2026 CS 332 Subject 9

NJIT

New Jersey's Science & Technology University

THE EDGE IN KNOWLEDGE

Alex. Gerbessiotis

CS Department

The yellowish/brownish notes are copyright the textbook author.

The bluish notes are copyright the instructor and are add-ons and provide more explanation or additional details

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

Module 1

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

Operating Systems: Internals and Design Principles

Chapter 11 I/O Management and Disk Scheduling

Ninth Edition
By William Stallings

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Differences in I/O Devices

- Devices differ in a number of areas:

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

Block or character device

- 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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

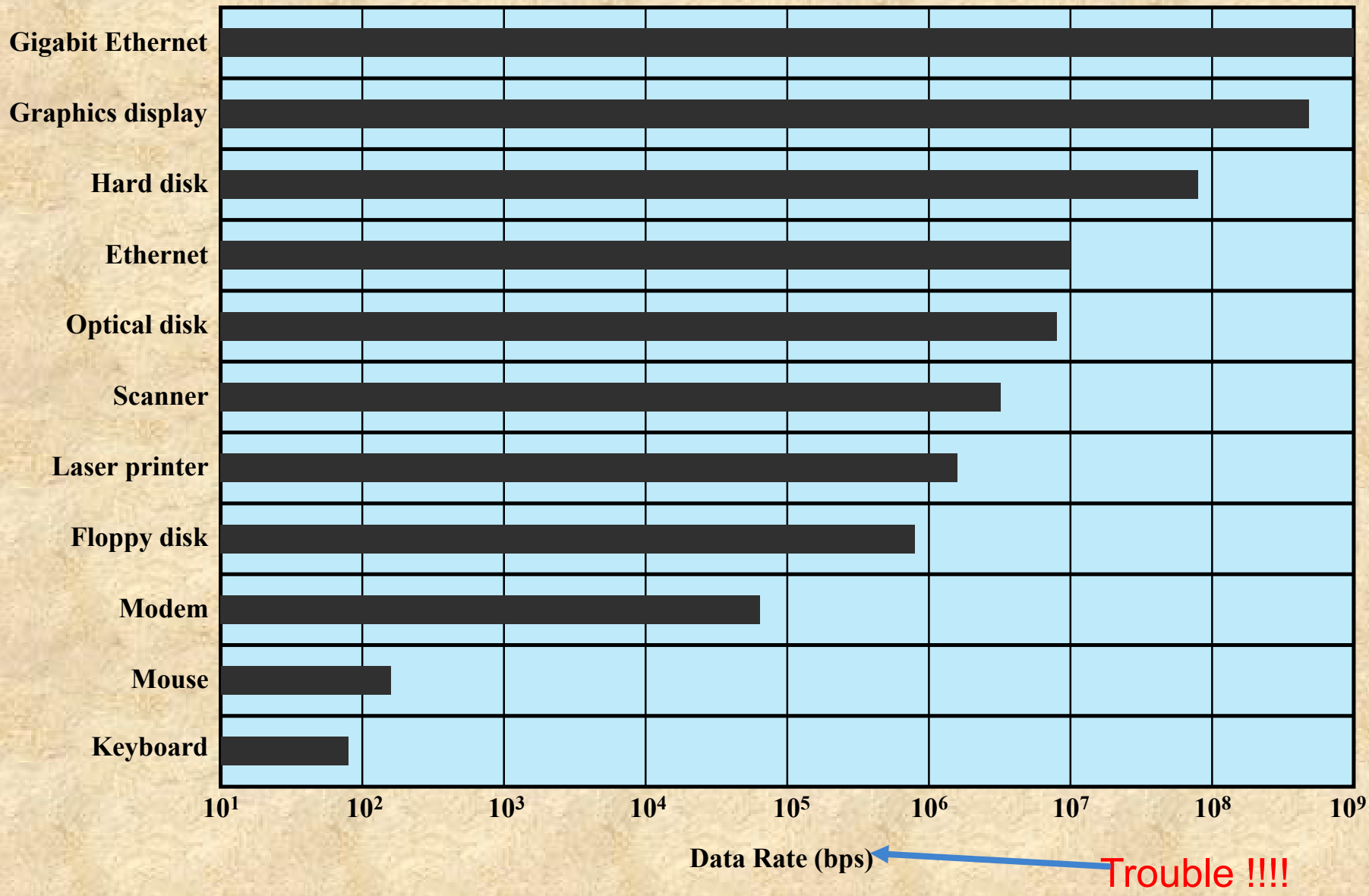


Figure 11.1 Typical I/O Device Data Rates

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Organization of the I/O Function

- Three techniques for performing I/O are:

Chapter 1 refresh

- **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**

One operation: multiple intrpts

  - The processor issues an I/O command on behalf of a process
    - If non-blocking – processor 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
- **Direct Memory Access (DMA)**

One operation: one intrpt at end!

  - A DMA module controls the exchange of data between main memory and an I/O module

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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)

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Linux / Unix

File descriptor fd

Per Process Open File Table

fd indexes POFT[fd]

POFT[3]=47

fd : 3 maps to pofte: 47

(pofte : POFT entry)

Inode in

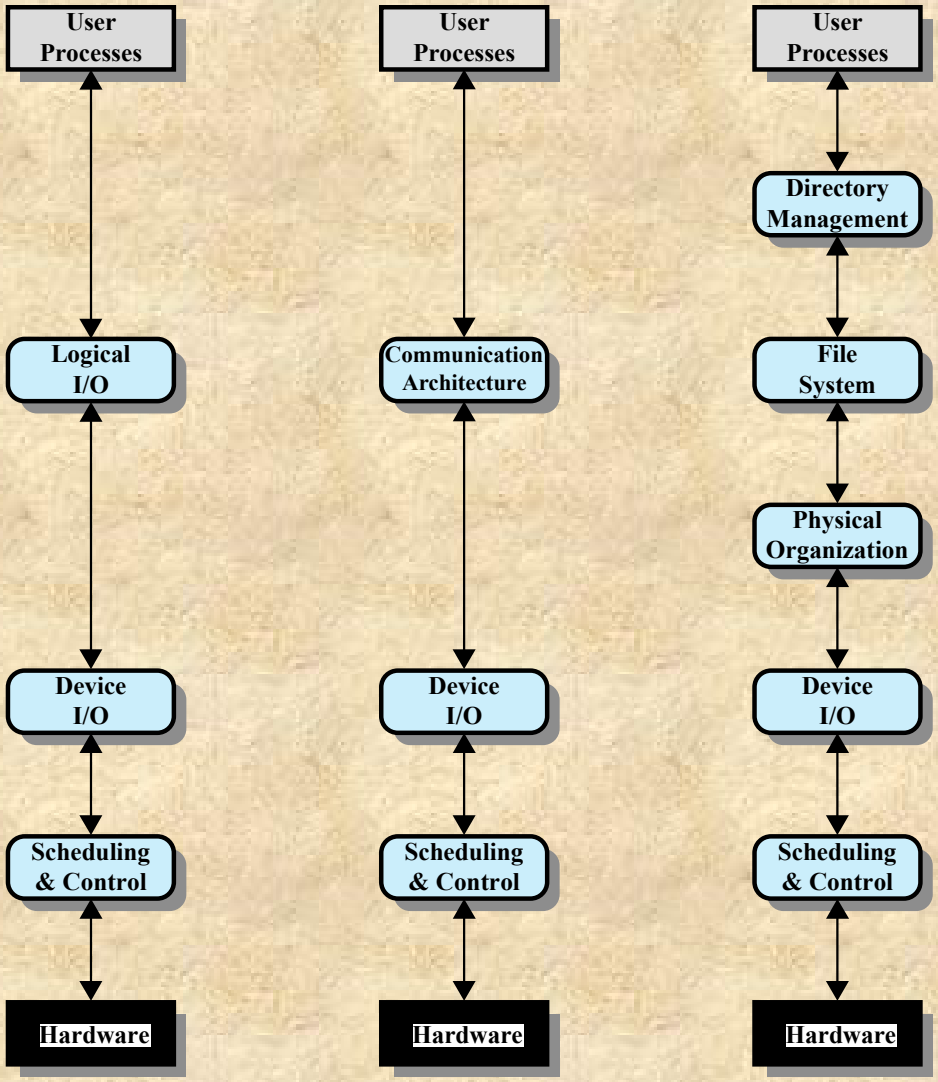
Kernel Global File Table

pofte indexes KGFT[pofte]

KGFT[47]=19

offset 47 of KGFT has a 19

One file has multiple fds in different processes but ONE pofte (47) that maps to a distinct inode (19)



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

% cp file1 file2

... fd=open("file1","r");

write(3, data);

writeblock(47, 123)

writesector(drv1, 0x14)

fd 0 : stdin

fd 1 : stdout

fd 2 : stderr

default fd

Figure 11.4 A Model of I/O Organization

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Buffering

- To avoid overheads and inefficiencies, it is sometimes convenient to perform input transfers in advance of requests being made, and to 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, mouse and other pointing devices, and most other devices that are not secondary storage are examples

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

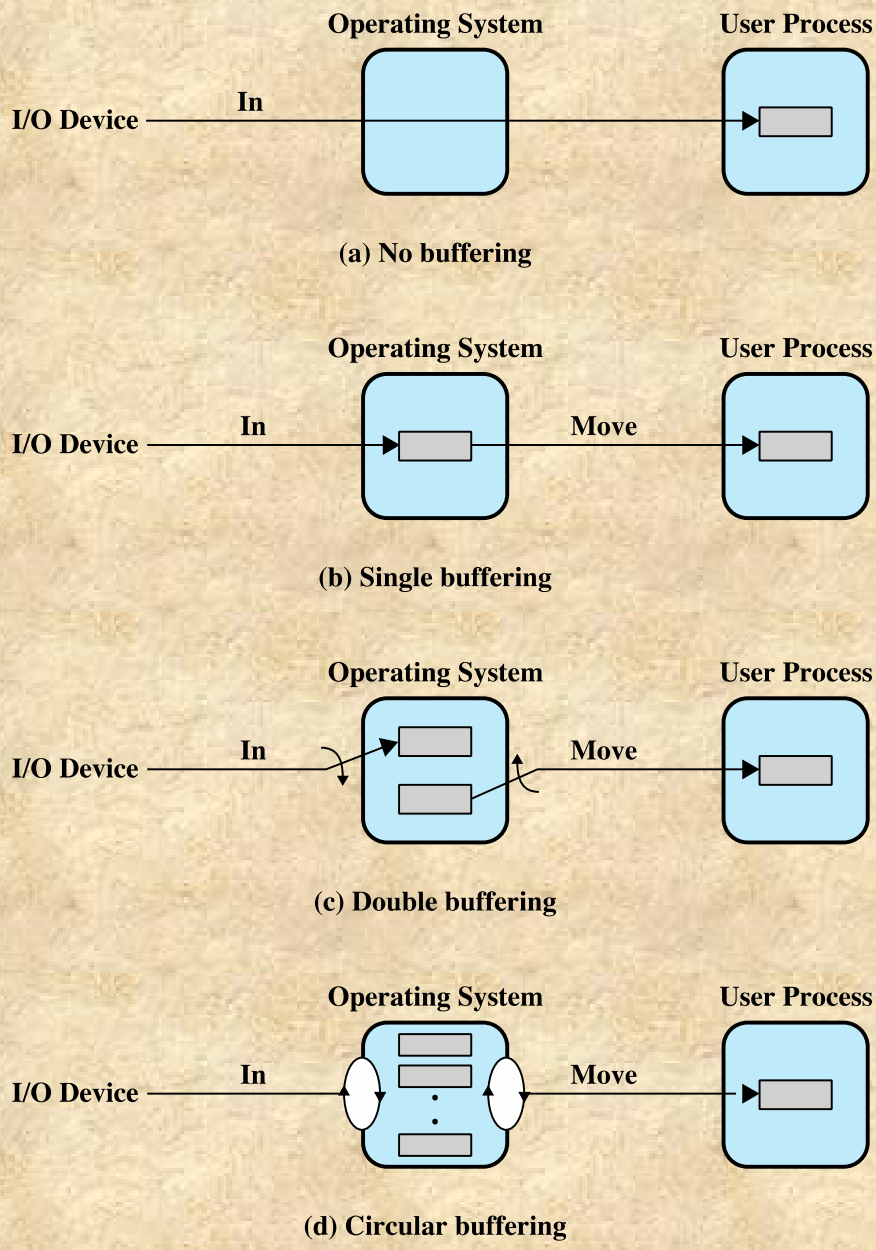
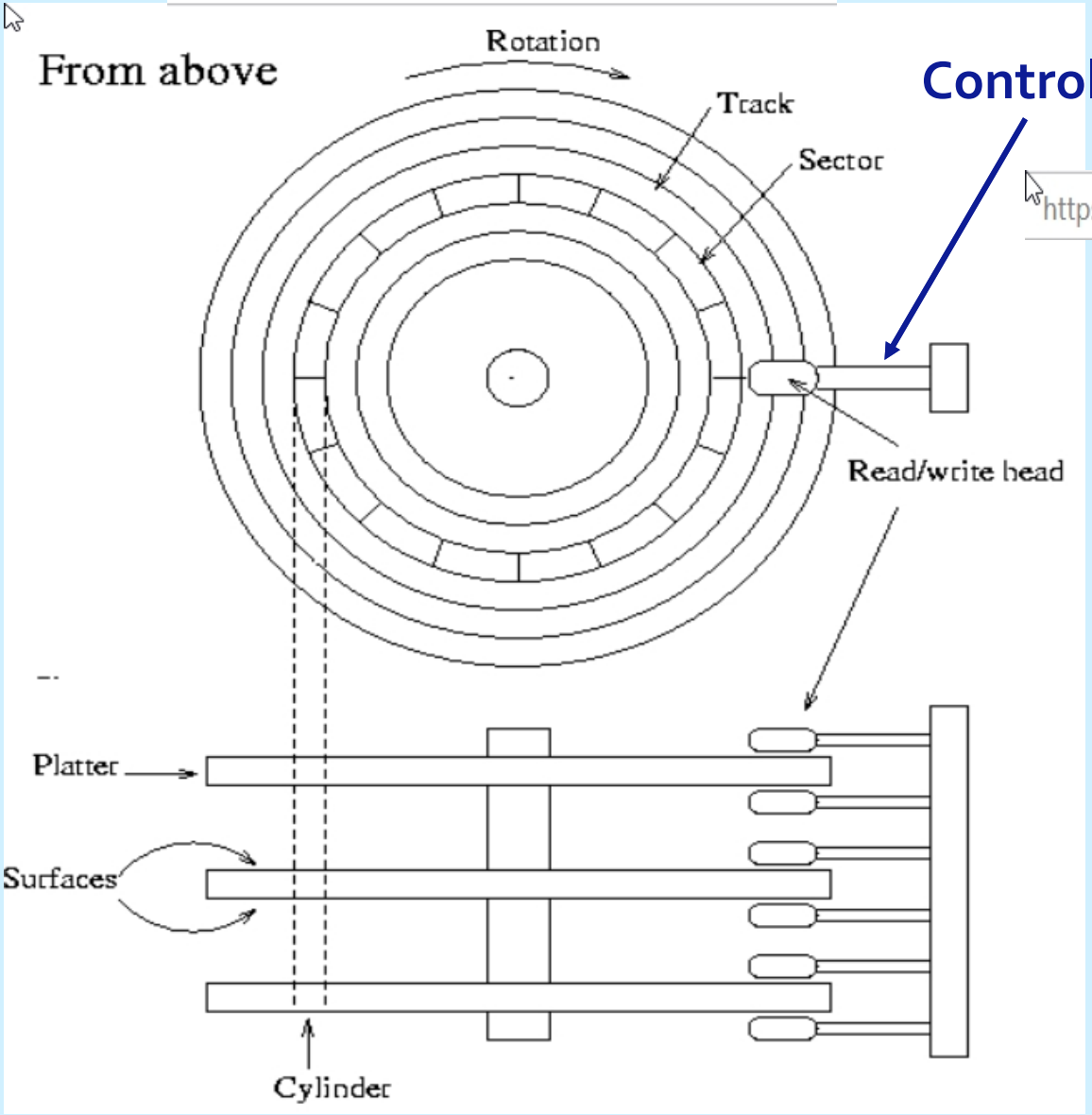


Figure 11.5 I/O Buffering Schemes (input)

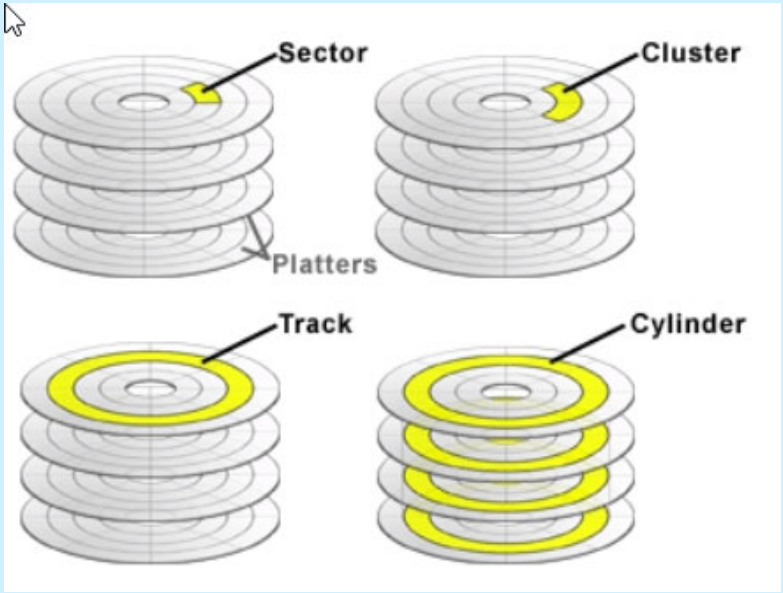
© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

https://www.tldp.org/LDP/sag/html/hard-disk.html



Controller Arm

https://superuser.com/questions/974581/chs-to-lba-mapping-disk-storage



Important Note:
Only one head can read/write at any one time in most (if not all) HDD

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

HDD : Names

Platter: Circular disk ; two surfaces (aka sides):up and down ; both read/write ; all platters rotate in unison. (One platter is for control purposes; remaining for data)

Arm and Heads. Arm contains disk controller. Attached to arm are the heads;
Number of heads is number of platters * two.
Heads move in unison; only ONE head active for read/write
Arms/Head move parallel to surface of platters.

Track: Concentric circular bands (regions) on platter (surface/side).
Density: KTPI (thousands of tracks per inch)
Outtermost to innermost numbering or the other way around
Track 5 is not a unique name; Every surface (2/platter) of every platter has a track 5. Thus,

Cylinder: All tracks of the same radius from center of platter, over all surfaces of all platters form a cylinder. Track 5 thus is cylinder 5 unless we specify surface and platter/disk number.

Thus: # tracks is equal to # cylinders

Sector: Every track has a fixed number of sectors
sector is a piece of a track at a given arc range.

Cluster: A set of consecutive sectors of a track

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

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

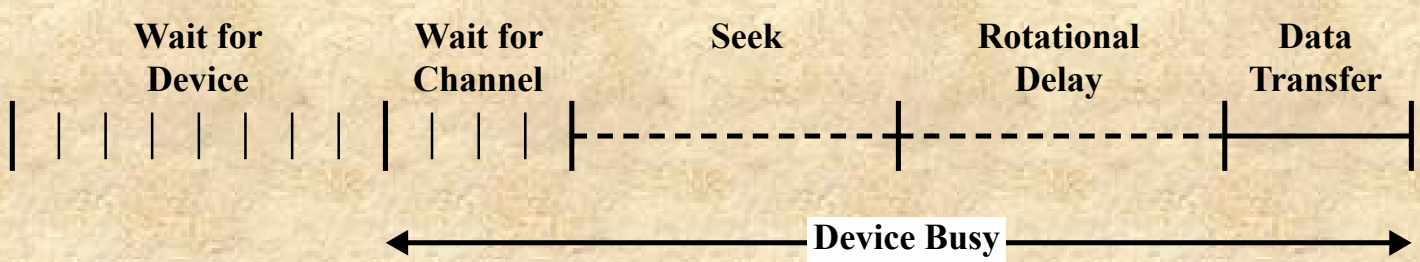


Figure 11.6 Timing of a Disk I/O Transfer

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

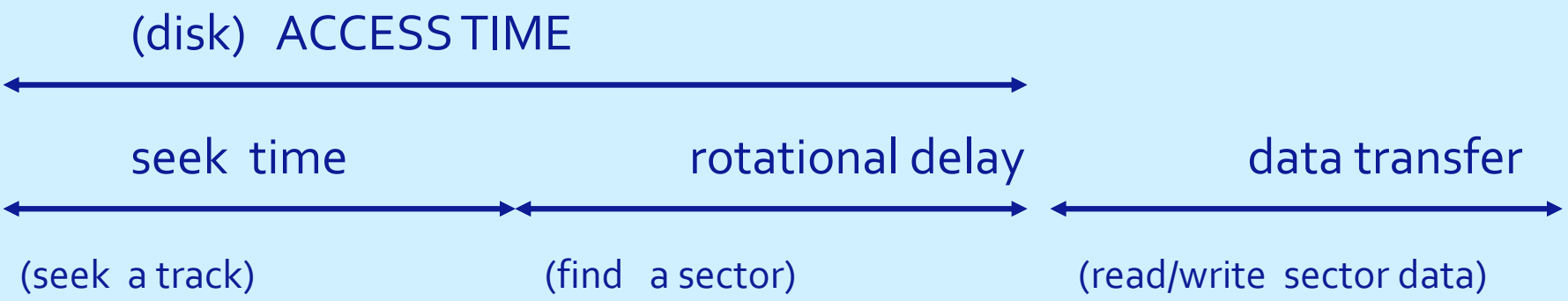
Disk Performance Parameters

- 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** (includes settle time) + controller overhead
- The time it takes for the beginning of the sector to reach the head is known as **rotational delay** (also know rotational latency)!
- The sum of the seek time and the rotational delay equals the **access time** **(note a difference in use in CS 332)**

More on slides 20 and later

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

HDD time(s)



Device Driver Request: **Read or Write**
sector X of track Y of platter Z of surface A (Up or Down)

Controller **Will develop a plan of action (controller overhead time)**

Total time to Read/Write Sector = Disk Access time + Data Transfer time

Disk Access time is sum of

Seek time : Move head from current(source) position to destination track Y
Seek time includes the following three components
(a) **controller overhead** [sometimes]
(b) **seek head movement time**
(c) settle (**settling time**) of one head Z coming closer to surface A

RotationalDelay : Find sector X on destination track Y i.e time for appropriate sector to be under or over the designated head

Data Transfer time **is time to transfer data from sector (read) or alternatively to sector (write)**

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

Access time: seek time

(disk) ACCESS TIME

seek (track) time : primarily time to move head from source (current) track to destination (desired) track but also includes controller overhead+settling

Seek (track) time components

a. **Controller overhead** (plan of action): : 2ms (typical)

b. **Seek time** (accelerate-steadyspeed-decelerate):

Average seek time (Used in practice) : 1/3 of Maximum seek time

Rarely used for MIN-TRACK to MAX-TRACK seek : Maximum seek time (special cases)

(if from say track 5 to track 6) : 0.5ms (typical for small moves)

c. **settle (or settling) time** : 1ms (typical)

Notes

Maximum seek time :Baseline and useless most of the times : 24-32ms (nowadays)

1950/1960s : 600ms, 1970s: 25ms, PC-HDD@1980s: 120ms

PC-HDD@20x0: 24-30ms server-HDD@20x0: 8-10ms

Track to track: (eg track 5 to track 6) : 0.5ms

Average seek time: Used in practice : 1/3 of Maximum seek time

Proof: Average seek= $\sum_{i,j} |i-j| / N^2$ ~ N/3

i: every starting position (not just innermost)

j: every ending position(not just outermost)

N : number of tracks (1 .. N or 0 .. N-1)

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

HDD and Rotational Delay

Rotational delay/ or rotational latency time: **Time for appropriate sector to be under or over designated head**
[it is 'head' singular: only one head is active for read or write operation at any one time]

Typical rpm values for HDD	**:**	**4800rpm, 5400rpm, 7200rpm**	
Best Case (Minimum rotational delay)	**:**	**0ms**	**(no wait)**
Worst Case (Maximum rotational delay)	**:**	**1 rotation**	**(max-wait)**
Average Case (Typical value used in practice):		**1/2 * rotation**	**(half-rotation)**

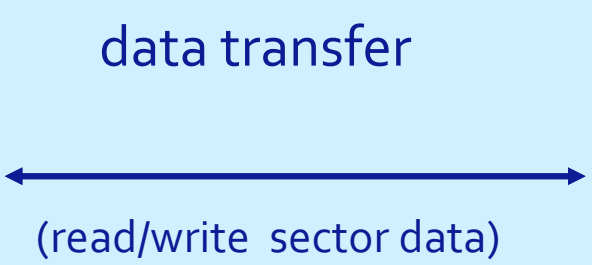
Example.
Maximum rotational delay for a 7200rpm disk (time for one rotation):

time 1 mn 60s 60x1000ms 60000ms
------- = -------- =------- = ---------- = -------- = **8.33 ms/R** (R indicates rotation)
rotation 7200 R 7200R 7200R 7200R

Example.
Average rotational delay for 7200rpm disk is 1/2 * rotation = 1/2 *8.33ms/R ~ 4.166 ~ 4.17ms

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

HDD and Transfer time



Typical values nowadays: 100,000-200,000 KiB/s for read (slower for write)

ExampleTransfer time: SectorSize/ 100,000KiB/s

Sometimes Transfer time: Time of one rotation since all data of all sectors of destination track can be read!
In reality there would be errors and would require a couple of rotation(s) to read all sector(s) of a track

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

Seek Time

- The time required to move the disk arm to the required track
- Consists of two key components:
  - The initial startup time
  - The time taken to traverse the tracks that have to be crossed once the access arm is up to speed
- Settling time
  - Time after positioning the head over the target track until track identification is confirmed
- Much improvement comes from smaller and lighter disk components
- A typical average seek time on contemporary hard disks is under 10ms

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Disk Performance

- 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
  - Disks rotate at speeds ranging from 3,6000 rpm (for handheld devices such as digital cameras) up to 15,000 rpm

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Practice 1

1.1. Work out the details of the example on the next two pages!

1.2 Then work out the questions of the third page from this page!

Practice 1: A 10TB disk

A 7200RPM server hard disk drive with capacity (10TB or 10TiB?) has
7 disks (a synonym for platters)

14 heads (7*2) ; 14 sides

Data density nowadays	: 1.5TiB per platter (~0.75TiB per surface)
(Logical) sector size:	: 512B
(Physical) sector size:	: 4KiB (4096B)
Physical sector emulates 8 logical sector	: (8x512=4096)

Controller overhead: 2ms
Settle time: 1 ms (also known as settling time)
Average seek time: 8ms What is the max seek time ?
Average rot. delay: 4.17ms (for 7200rpm)
I/O transfer rate: 200,000 KiB/s <<< tricky: practice makes perfect

(Disk) access time = seek-time + rotational-delay (latency) [textbook]
In seek time we include controller overhead + settle time [if given]

Total time to read one sector = Disk access time + Data transfer time

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

Practice 1: A 10TB disk

Total time-toread-1logical-sector = Disk access time + Data Transfer time

Disk-access-time= Seek time + Rotational delay

= (controller-overhead + seek-time+settle-time) + Avg.rot.delay

= (2ms + 8ms+1ms) + (1/2) * (60*1000ms)/7200R =

= 11ms+ 4.17ms = 15.17ms

Transfer-time = (512B) / 200,000KiB/s = 512B /(200,000*1,024 B/s) =

= 512 s / (200,000x1024) = 512*1000ms / (200,000x1024) =

= 1/400 ms = 0.0025ms

Time-toread-512B = 15.17ms + 0.0025ms = 15.1725ms

~ 16ms (rounded-up to next millisecond)

Transfer time is negligible!

Effective Transfer Rate = Actual Bytes transferred/ Time = 512B / 15.1725ms =

= 512B x 1000 / 15.1725s ~ 33745 B/s ~ 33KiB/s

We used on purpose 200,000 KiB/s which is 200,000 x 1024B/s to make calculations easier ; 200MiB/s is 200x1024x1024B/s

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

Practice 1': A 10TB disk

Question 1: How long to read 1000 sectors (located on random tracks/surfaces/platters)?
Hint: 1000x 16ms = 16s

Question 2: Let's say a track has 1000sectors and all of them are on one track.
How long to read those 1000 sectors?
Hint: The moment you hit the track start reading all sectors; rearrange them later in software (eg in MM)

Challenge questions : Search Web for answers (not an exam question!)
How many logical sectors in HDD?
How many physical sectors in HDD?
How many tracks ? (physical sector)

Transfer of Information: **A sector**

Challenge questions: **Practice 2 (next page)**
If a sector is 512B of data how many B does the head read when it transfers a sector?

If a sector is 4KiB of data how many B does the head read when it transfers a sector?

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

Practice 2

A 512B sector is preceded by 15B of gap, sync, and sector address data, and followed by 50B of ECC (Error Correcting Code) data (40 10-bit).Therefore a head effectively reads 512+15+50 = 577B (512/577 = 88% of it is data)

Gap / Sync / Sector Address : 15B | 512B of sector data | 50 B ECC (40 10-bit)

Gap / Sync / Sector Address : 15B | 4096B of sector data | 100 B ECC (80 10-bit)

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

Module 2

Minimize Seek (Head Movements)

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

Disk Scheduling

Controller received requests for sectors involving tracks
1 (innermost)
n (outermost)
2, n-1, 3, n-2, 4, n-3 , etc
thus a batch of n requests has arrived in that order.

What is a better track management
(a) FirstComeFirstServed : 1,n,2,n-1,3,n-2, etc
Or
(b) 1,2,3,4,… n-3,n-2,n-1, n ?

Determining between (a) vs (b) is disk scheduling!

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

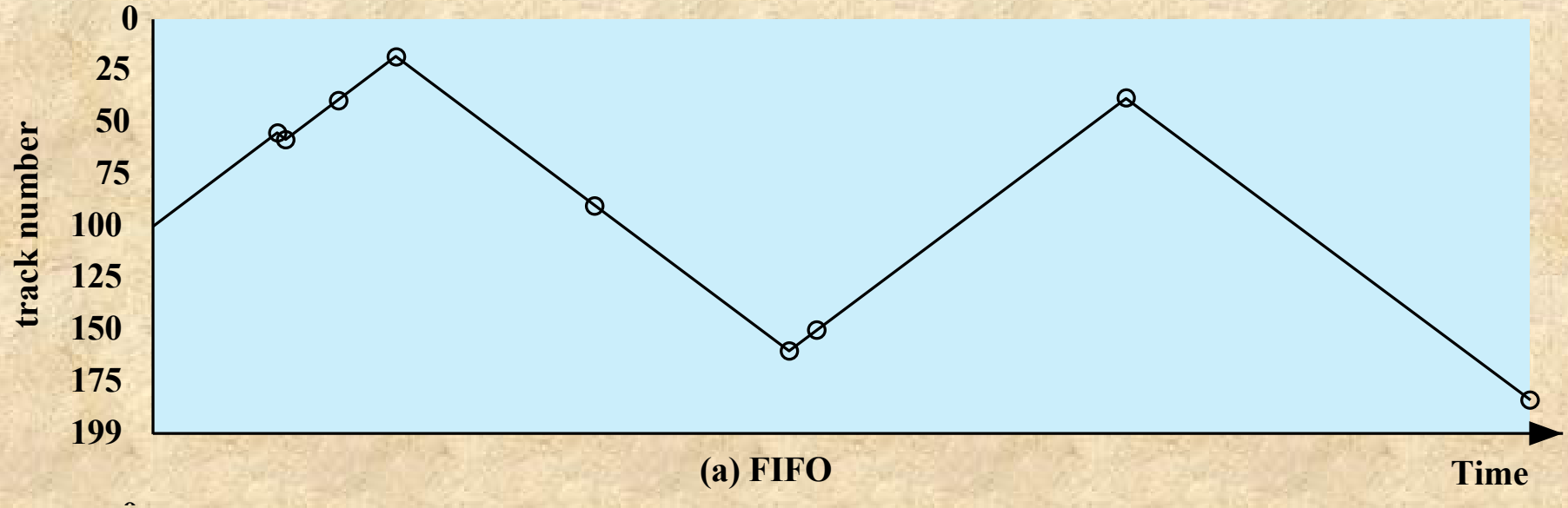
Disk Scheduling

For a batch of requests (in the example number of requests is 9) we are allowed to reorder to optimized head (track) movements

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

First-In, First-Out (FIFO)

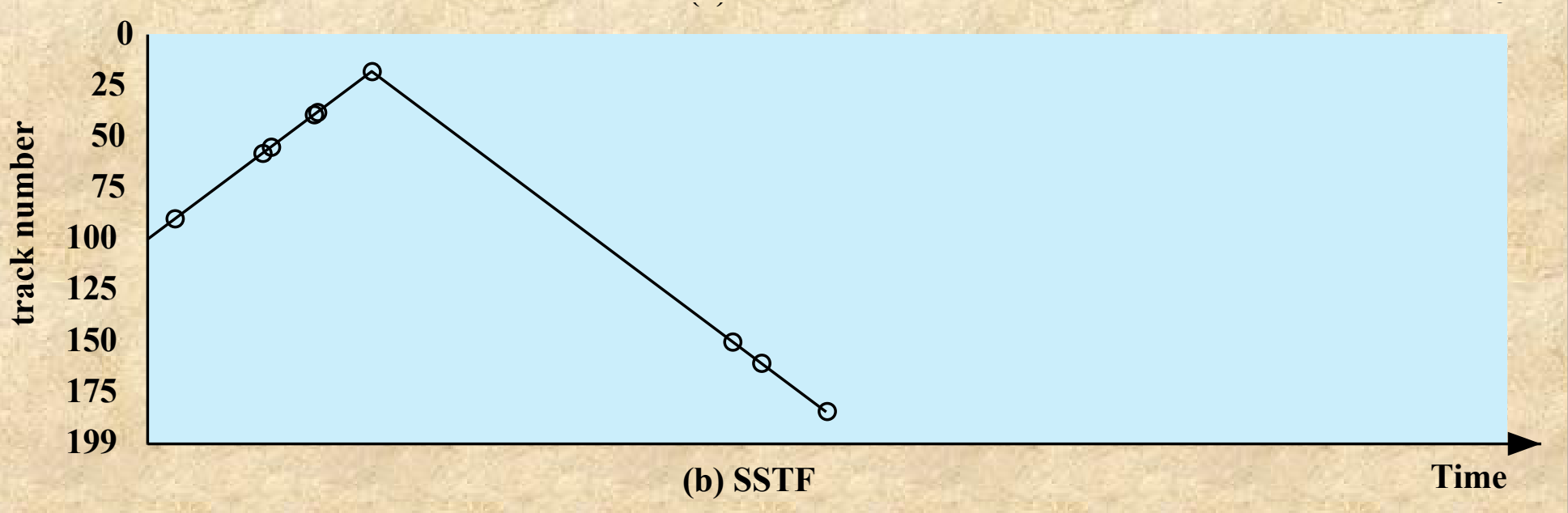
- Processes in sequential order
- Fair to all processes
- Approximates random scheduling in performance if there are many processes competing for the disk



© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

SCAN

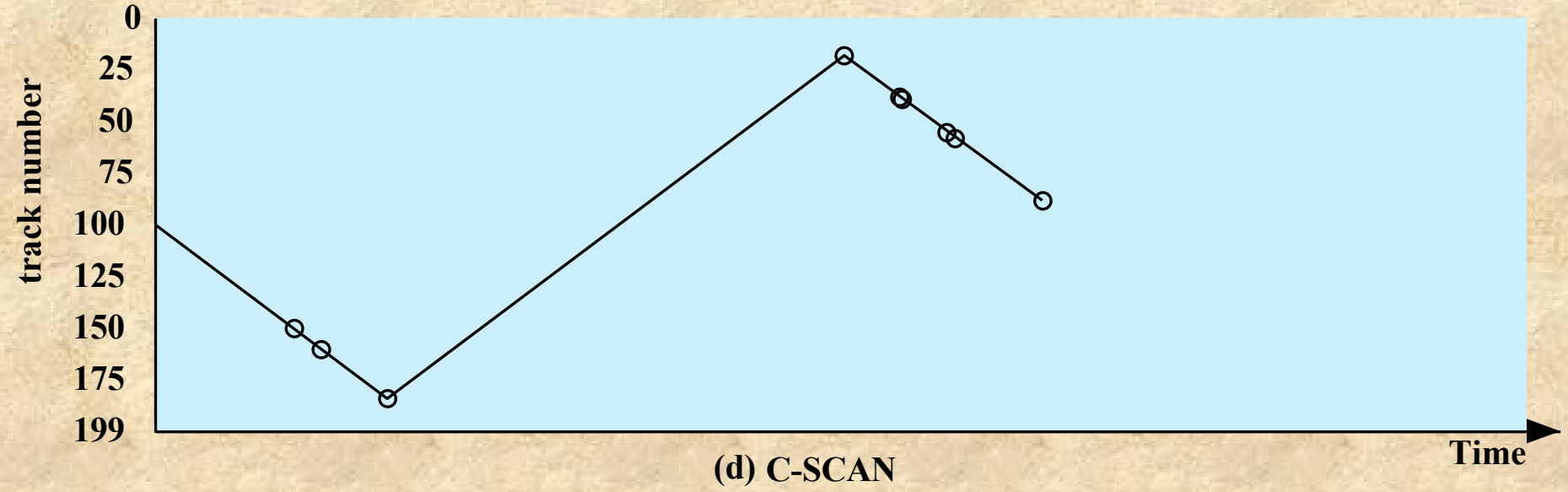
- Also known as the elevator algorithm
- Arm moves in one direction only
  - Satisfies all outstanding requests until it reaches the last track in that direction then the direction is reversed
- Favors jobs whose requests are for tracks nearest to both innermost and outermost tracks and favors the latest-arriving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

(c) SCAN

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

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

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

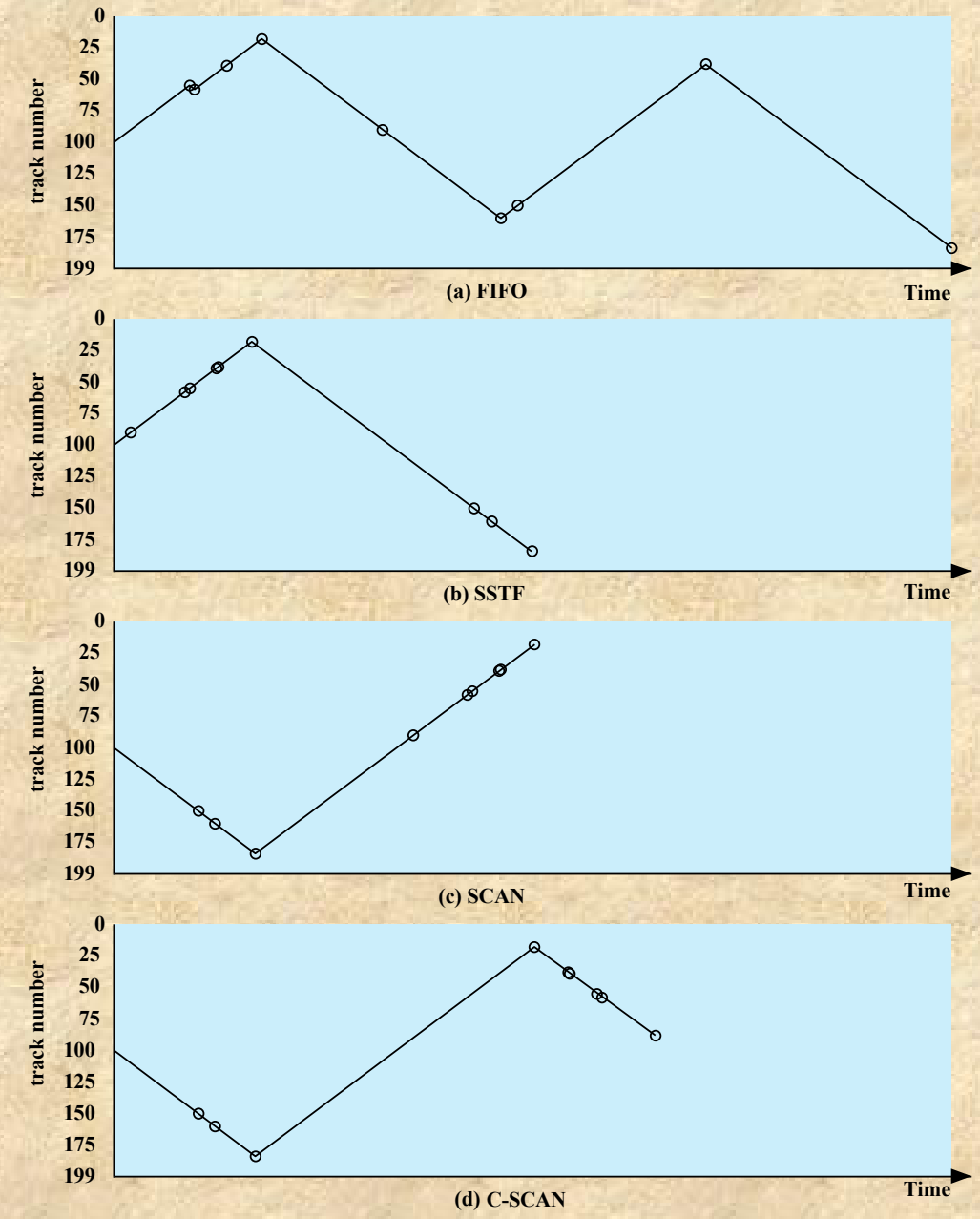


Figure 11.7 Comparison of Disk Scheduling Algorithms (see Table 11.3)

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

(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

Table 11.2 Comparison of Disk Scheduling Algorithms

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Name	Description	Remarks
Selection according to requestor		
Random	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

Table 11.3 Disk Scheduling Algorithms

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Practice 3

3.1 Just like Table 11.2 (slide 38) head is at track 100.
Requests are arriving for tracks 20, 40, 60, 120, 110.
What is the average seek time for a FIFO disk scheduling algorithm?

3.2 Repeat for SSTF.

3.3 Repeat for SCAN.

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

RAID

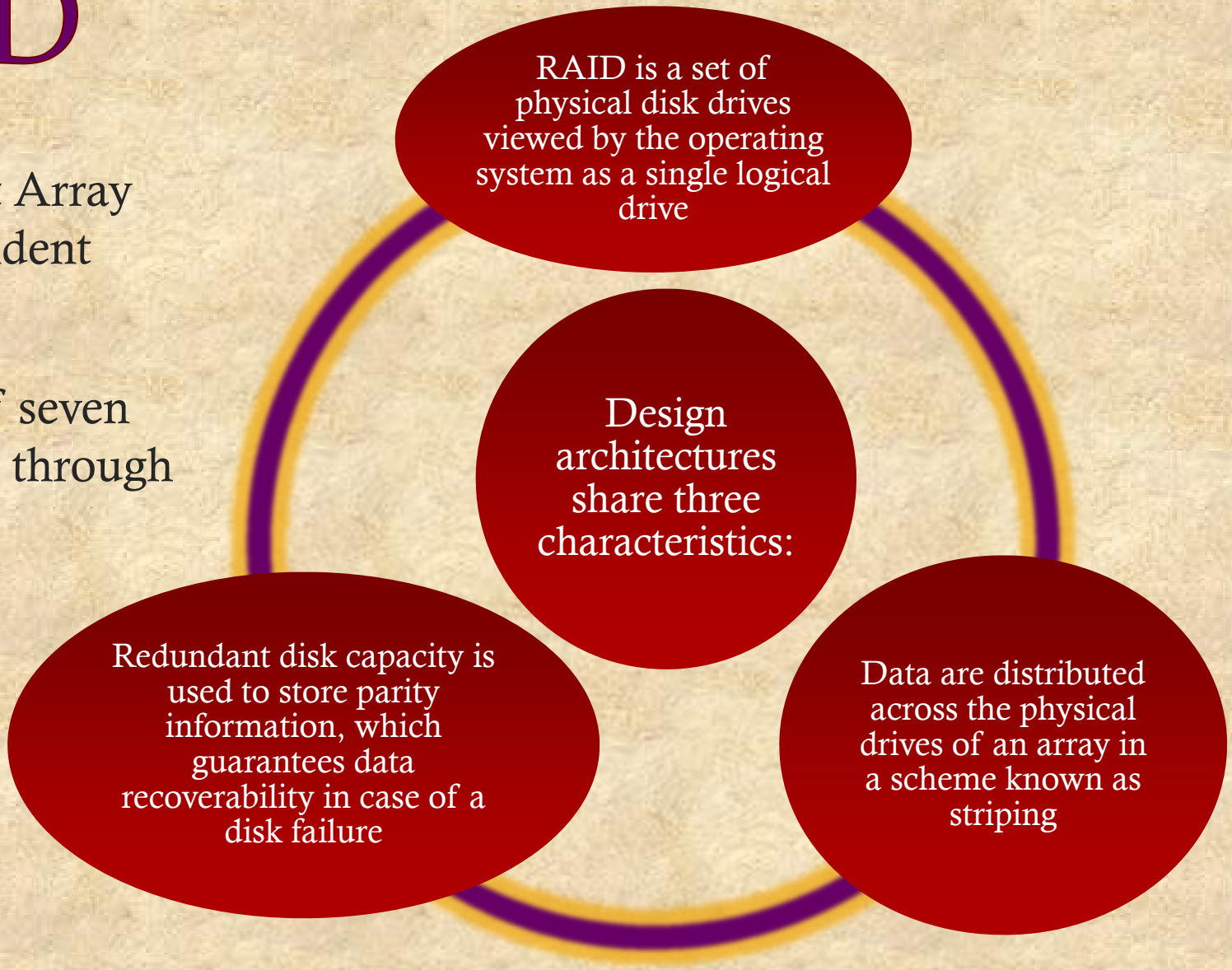
RAID : Redundant Array of Inexpensive Disks

Multiple Physical DISK DRIVES viewed as ONE logical DISK DRIVE

© 2019-2027. Alex. Gerbessiotis. All rights reserved.

RAID

- Redundant Array of Independent Disks
- Consists of seven levels, zero through six



© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

RAID

- The term was originally coined in a paper by a group of researchers at the University of California at Berkeley
  - The paper outlined various configurations and applications and introduced the definitions of the RAID levels
- Strategy employs multiple disk drives and distributes data in such a way as to enable simultaneous access to data from multiple drives
  - Improves I/O performance and allows easier incremental increases in capacity
- The unique contribution is to address effectively the need for redundancy
- Makes use of stored parity information that enables the recovery of data lost due to a disk failure

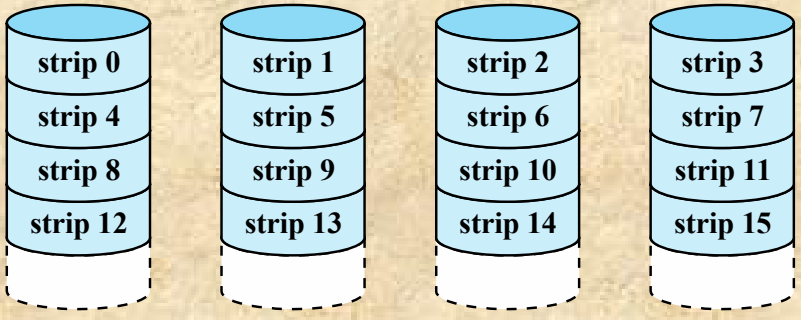
© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Category	Level	Description	Disks required	Data availability	Large I/O data transfer capacity	Small I/O request rate
Striping	0	Nonredundant	N	Lower than single disk	Very high	Very high for both read and write
Mirroring	1	Mirrored	$2N$	Higher than RAID 2, 3, 4, or 5; lower than RAID 6	Higher than single disk for read; similar to single disk for write	Up to twice that of a single disk for read; similar to single disk for write
Parallel access	2	Redundant via Hamming code	$N + m$	Much higher than single disk; comparable to RAID 3, 4, or 5	Highest of all listed alternatives	Approximately twice that of a single disk
	3	Bit-interleaved parity	$N + 1$	Much higher than single disk; comparable to RAID 2, 4, or 5	Highest of all listed alternatives	Approximately twice that of a single disk
Independent access	4	Block-interleaved parity	$N + 1$	Much higher than single disk; comparable to RAID 2, 3, or 5	Similar to RAID 0 for read; significantly lower than single disk for write	Similar to RAID 0 for read; significantly lower than single disk for write
	5	Block-interleaved distributed parity	$N + 1$	Much higher than single disk; comparable to RAID 2, 3, or 4	Similar to RAID 0 for read; lower than single disk for write	Similar to RAID 0 for read; generally lower than single disk for write
	6	Block-interleaved dual distributed parity	$N + 2$	Highest of all listed alternatives	Similar to RAID 0 for read; lower than RAID 5 for write	Similar to RAID 0 for read; significantly lower than RAID 5 for write

N = number of data disks; m proportional to log N

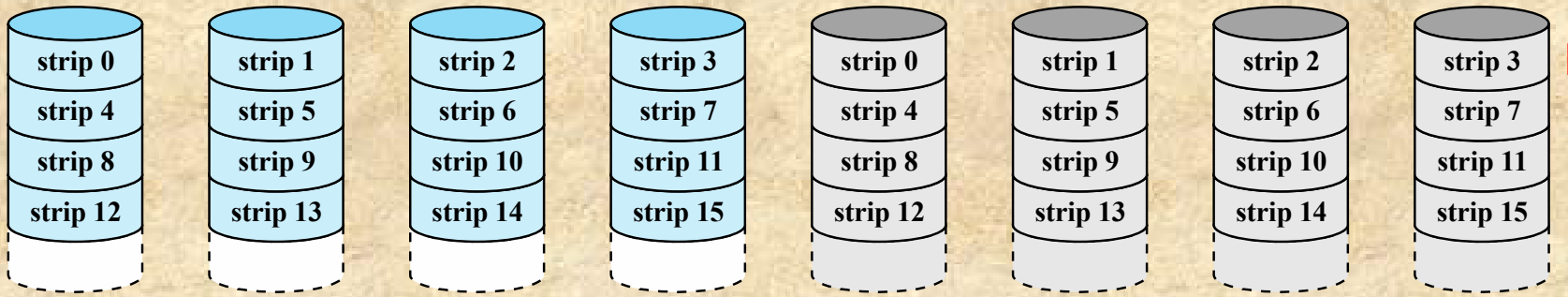
Table 11.4 RAID Levels (Page 498 in textbook)

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.



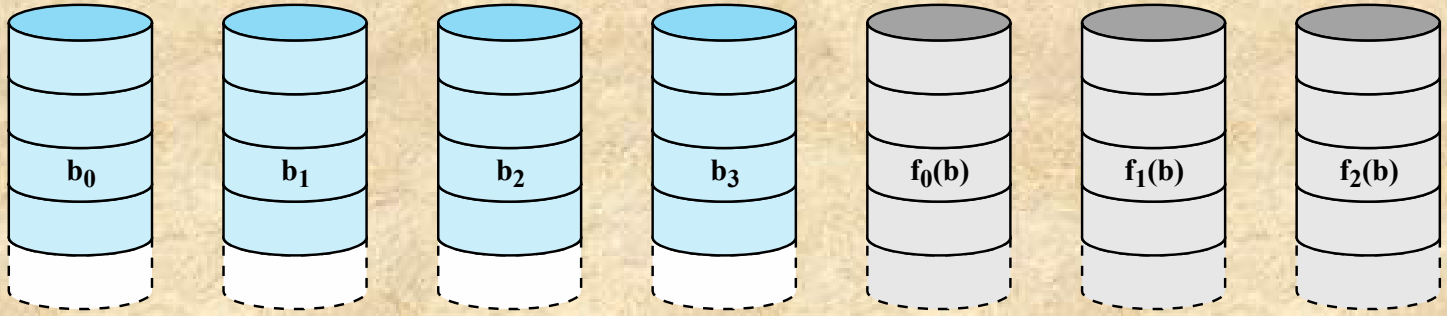
RAID 0: Think of ' Buy 4 (or X) disks' ' 1/4 or 1/X of every file on each disk'

(a) RAID 0 (non-redundant)



RAID 0 times 2

(b) RAID 1 (mirrored)



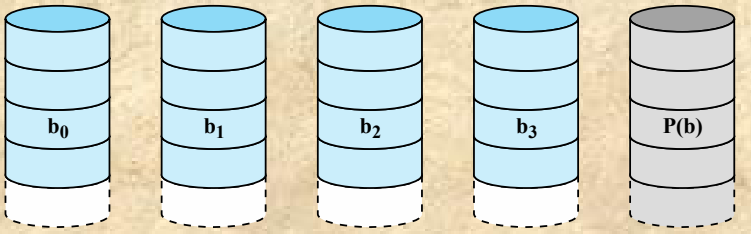
RAID 2 forget it no one is using it

(c) RAID 2 (redundancy through Hamming code)

Figure 11.8 RAID Levels (page 1 of 2)

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

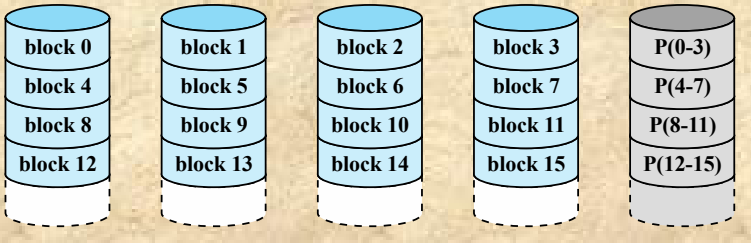
RAID 3
All work together



(d) RAID 3 (bit-interleaved parity)

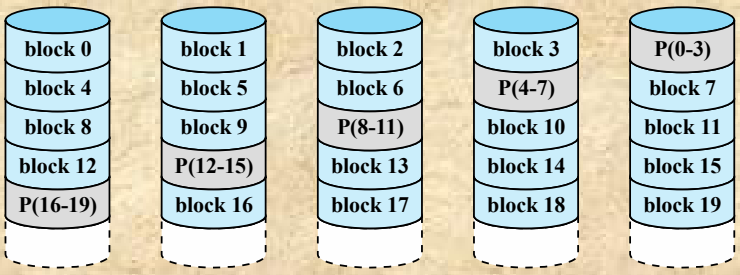
RAID 3
1 disk out:
can be reconstructed
from others

RAID 4
Disks not working in
unison



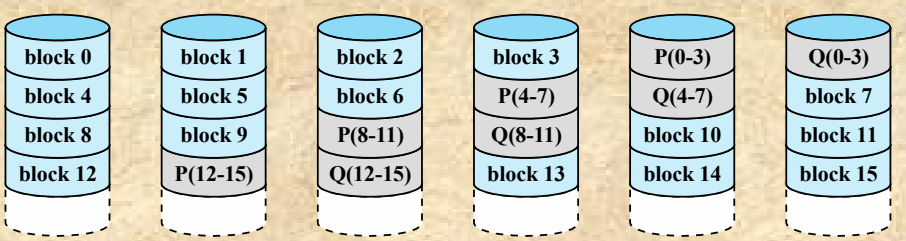
(e) RAID 4 (block-level parity)

RAID 4
Similar to RAID 3 or2
But: Block 2 written,
P(0-3) needs update



(f) RAID 5 (block-level distributed parity)

RAID 5
Fancier RAID 4
Spreads out parity
updates if there are many
writes!



(g) RAID 6 (dual redundancy)

RAID 6
Redundant RAID 5
If block2 and P(0-3) goes
bad, use Q(0-3) to recover

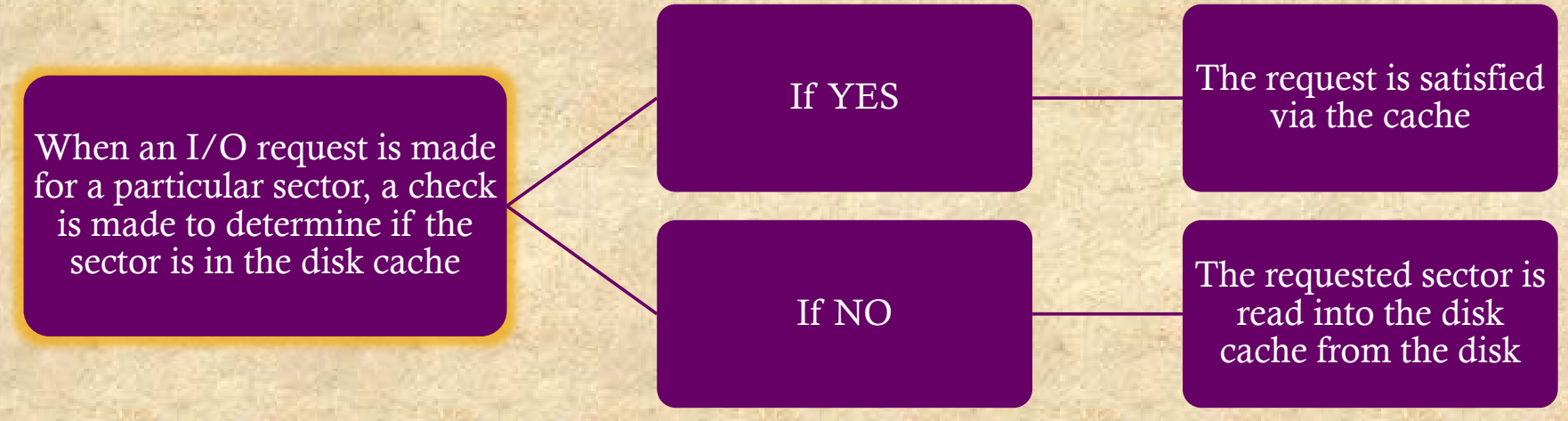
Figure 11.8 RAID Levels (page 2 of 2)

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Disk Cache

- *Cache memory* is used to apply to a memory that is smaller and faster than main memory and that is interposed between main memory and the processor
- Reduces average memory access time by exploiting the principle of locality
- *Disk cache* is a buffer in main memory for disk sectors
- Contains a copy of some of the sectors on the disk

IF YOU THOUGHT YOU WERE DONE WITH CACHES IN CHAPTER 1 or 8...



© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Least Recently Used (LRU)

- Most commonly used algorithm that deals with the design issue of replacement strategy
- The block that has been in the cache the longest with no reference to it is replaced
- A stack of pointers reference the cache
  - Most recently referenced block is on the top of the stack
  - When a block is referenced or brought into the cache, it is placed on the top of the stack

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

Least Frequently Used (LFU)

- The block that has experienced the fewest references is replaced
- A counter is associated with each block
- Counter is incremented each time block is accessed
- When replacement is required, the block with the smallest count is selected

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

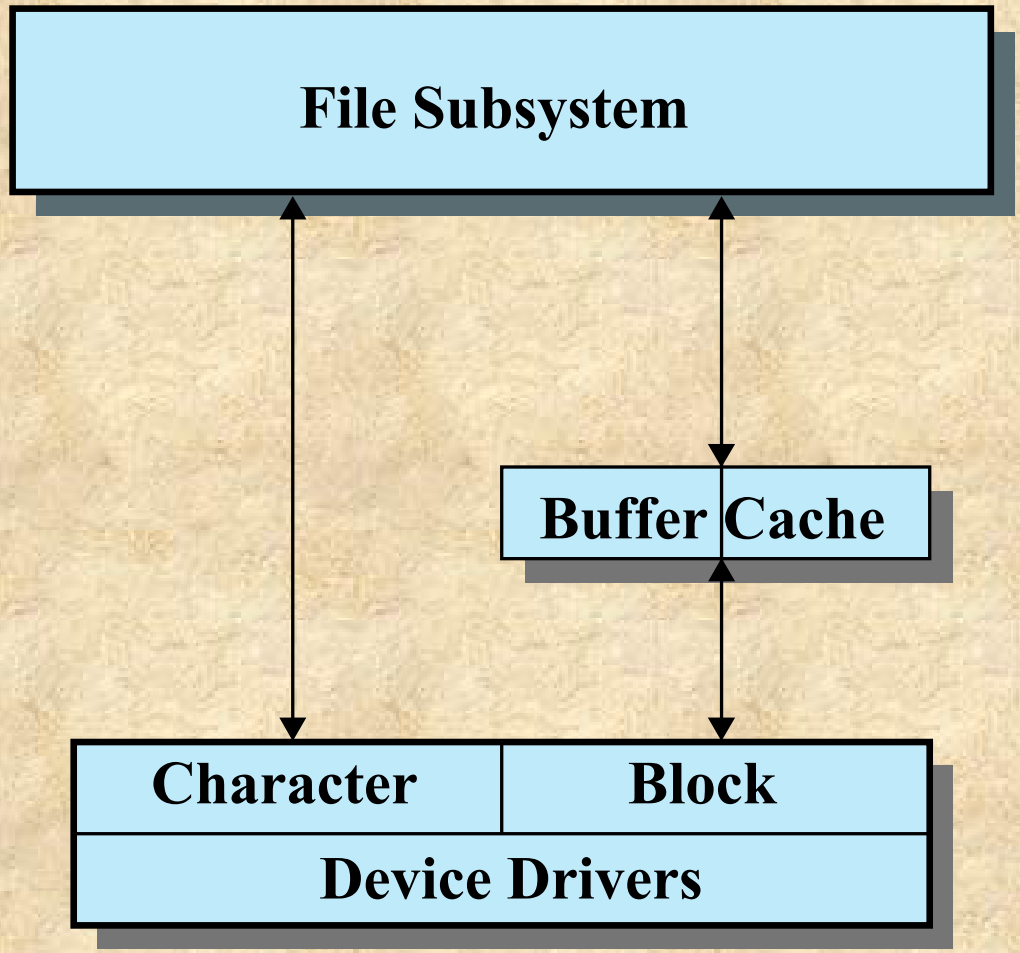


Figure 11.12 UNIX I/O Structure

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

UNIX Buffer Cache

- Is essentially a disk cache
  - I/O operations with disk are handled through the buffer cache
- The data transfer between the buffer cache and the user process space always occurs using DMA
  - Does not use up any processor cycles
  - Does consume bus cycles
- Three lists are maintained:
  - Free list
    - List of all slots in the cache that are available for allocation
  - Device list
    - List of all buffers currently associated with each disk
  - Driver I/O queue
    - List of buffers that are actually undergoing or waiting for I/O on a particular device

© 2017 Pearson Education, Inc., Hoboken, NJ. All rights reserved.

New Jersey's Science &
Technology University

THE EDGE IN KNOWLEDGE

Alex. Gerbessiotis

CS Department