Chapter 4

Device Management

Learning Objectives

After completing this chapter, you should be able to describe:

- Features of dedicated, shared, and virtual devices
- Differences between sequential and direct access media
- Concepts of blocking and buffering and how they improve I/O performance
- Roles of seek time, search time, and transfer time in calculating access time

Learning Objectives (cont'd.)

- Differences in access times in several types of devices
- Critical components of the input/output subsystem, and how they interact
- Strengths and weaknesses of common seek strategies, including FCFS, SSTF, SCAN/LOOK, C-SCAN/C-LOOK, and how they compare
- Different levels of RAID and what sets each apart from the others

Types of Devices

- **Dedicated Devices**
  - Device assigned to one job at a time
    - For entire time job is active (or until released)
    - Example: tape drives, printers, and plotters
  - Disadvantage
    - Inefficient if device is not used 100%
    - Allocated for duration of job's execution

Types of Devices (cont'd.)

- **Shared Devices**
  - Device assigned to several processes
    - Example: direct access storage device (DASD)
      - Processes share DASD simultaneously
      - Requests interleaved
  - Device manager supervision
    - Controls interleaving
      - Predetermined policies determine conflict resolution

Types of Devices (cont'd.)

- **Virtual Devices**
  - Dedicated and shared device combination
  - Dedicated devices transformed into shared devices
    - Example: printer
      - Converted by spooling program
  - Spooling
    - Speeds up slow dedicated I/O devices
    - Example: **universal serial bus (USB) controller**
      - Interface between operating system, device drivers, applications, and devices attached via USB host

Types of Devices (cont'd.)

- **Storage media**
  - Two groups
    - **Sequential access media**
      - Records stored sequentially
    - **Direct access storage devices (DASD)**
      - Records stored sequentially
      - Records stored using direct access files
  - Vast differences
    - Speed and sharability

Direct Access Storage Devices

- Directly read or write to specific disk area
  - Random access storage devices
- Four categories
  - Magnetic disks
  - Optical discs
  - Flash memory
  - Magneto-optical disks
- Access time variance
  - Not as wide as magnetic tape
  - Record location directly affects access time

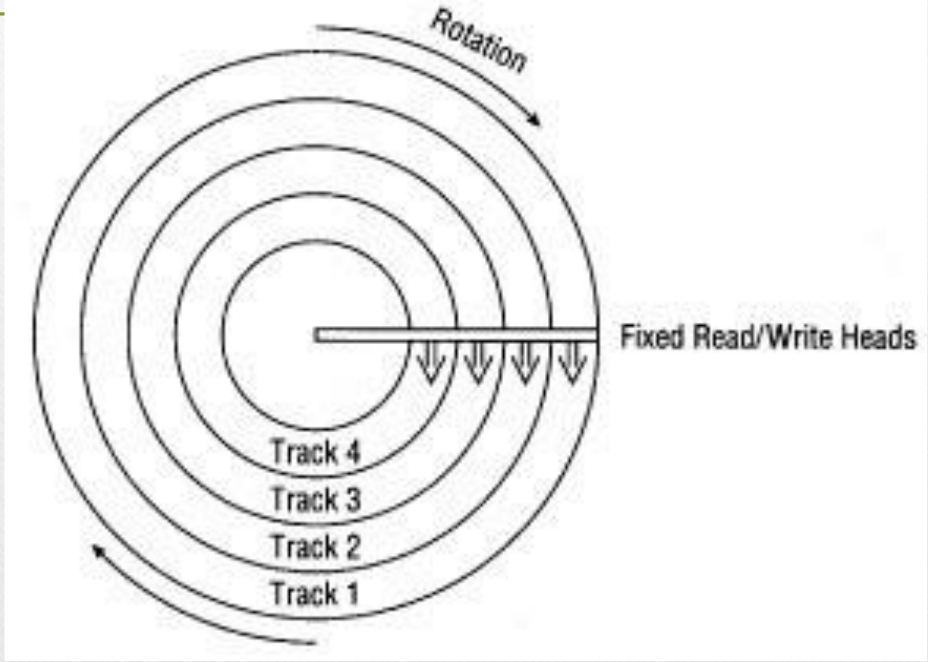
Fixed-Head Magnetic Disk Storage

- Looks like a large CD or DVD
  - Covered with magnetic film
  - Formatted
    - Both sides (usually) in concentric circles called **tracks**
  - Data recorded serially on each track
    - Fixed read/write head positioned over data
- Advantages
  - Fast (more so than movable head)
- Disadvantages
  - High cost and reduced storage

Fixed-Head Magnetic Disk Storage (cont'd.)

(figure 7.4)

A fixed-head disk with four read/write heads, one per track.



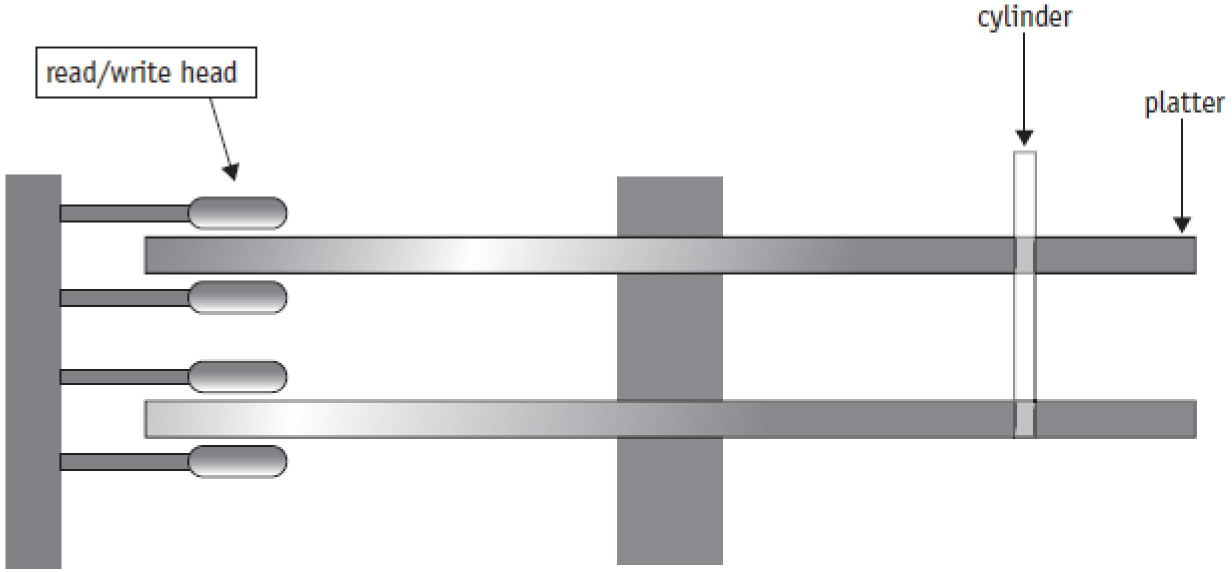
Movable-Head Magnetic Disk Storage

- One read/write head floats over disk surface
  - Example: computer hard drive
  - Disks
    - Single platter
    - Part of disk pack (stack of platters)
- **Disk pack platter**
  - Two recording surfaces
    - Exception: top and bottom platters
  - Surface formatted with concentric tracks
  - Track number varies
    - 1000+ (high-capacity disk)

Movable-Head Magnetic Disk Storage (cont'd.)

- Disk pack platter (cont'd.)
  - Track surface number
    - Track zero: outermost concentric circle on each surface
    - Center: contains highest-numbered track
  - Arm moves over all heads in unison
    - Slower: fill disk pack surface-by-surface
    - Faster: fill disk pack track-by-track
  - Virtual cylinder: fill track zero
- Record access system requirements
  - Cylinder number, surface number, record number

Movable-Head Magnetic Disk Storage (cont'd.)



(figure 7.5)

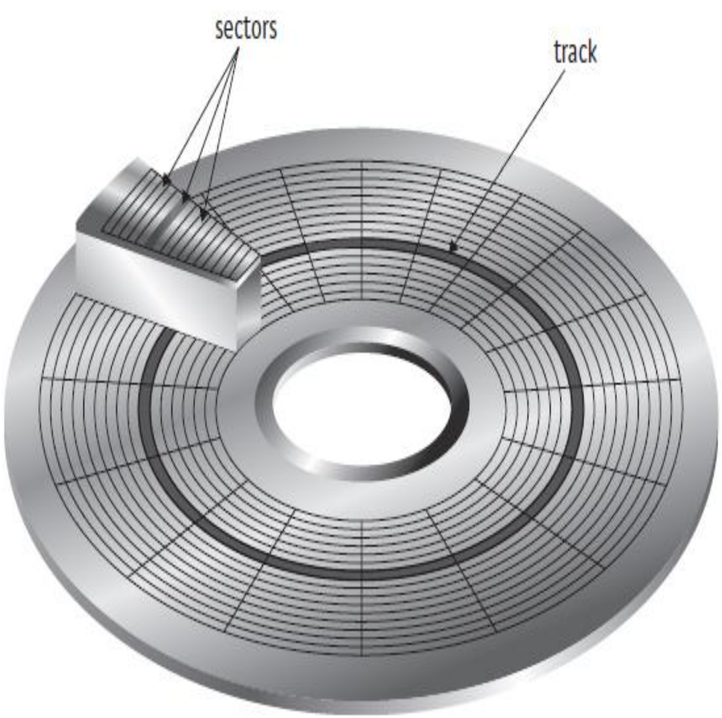
A disk pack is a stack of magnetic platters. The read/write heads move between each pair of surfaces, and all of the heads are moved in unison by the arm.

Optical Disc Storage

- Design difference
  - Magnetic disk
    - Concentric tracks of sectors
    - Spins at constant angular velocity (CAV)
    - Wastes storage space but fast data retrieval

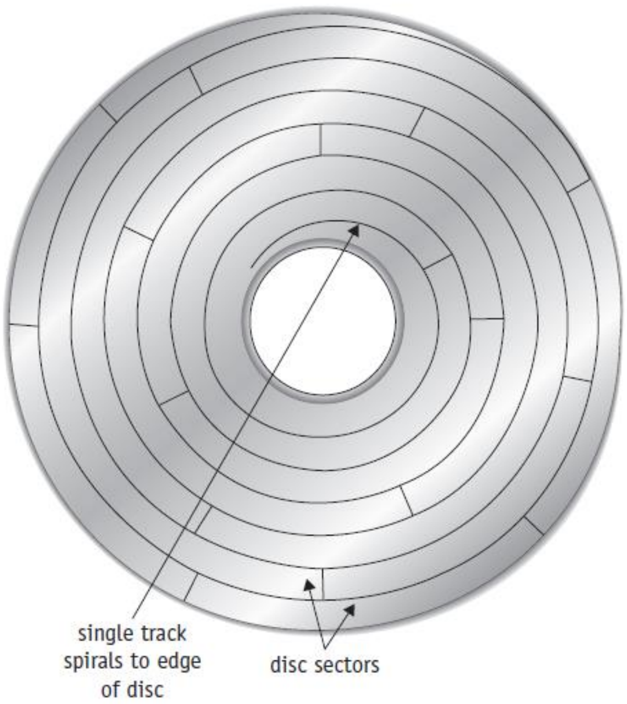
(figure 7.7)

On a magnetic disk, the sectors are of different sizes: bigger at the rim and smaller at the center. The disk spins at a constant angular velocity (CAV) to compensate for this difference. Some optical discs can read and write on multiple layers, greatly enhancing storage capacity.



Optical Disc Storage (cont'd.)

- Design features
  - Optical disc
    - Single spiralling track of same-sized sectors running from center to disc rim
    - Spins at constant linear velocity (CLV)
    - More sectors and more disc data



(figure 7.8)

On an optical disc, the sectors (not all sectors are shown here) are of the same size throughout the disc. The disc drive changes speed to compensate, but it spins at a constant linear velocity (CLV).

Optical Disc Storage (cont'd.)

- Two important performance measures
  - **Sustained data-transfer rate**
    - Speed to read massive data amounts from disc
    - Measured in megabytes per second (Mbps)
    - Crucial for applications requiring sequential access
  - **Average access time**
    - Average time to move head to specific disc location
    - Expressed in milliseconds (ms)
- Third feature
  - **Cache size** (hardware)
    - Buffer to transfer data blocks from disc

CD and DVD Technology

- **CD**
  - Data recorded as zeros and ones
    - **Pits**: indentations
    - **Lands**: flat areas
  - Reads with low-power laser
    - Light strikes land and reflects to photodetector
    - Pit is scattered and absorbed
    - Photodetector converts light intensity into digital signal

CD and DVD Technology (cont'd.)

- **CD-Recordable technology** (CD-R)
  - Requires expensive disk controller
  - Records data using write-once technique
  - Data cannot be erased or modified
  - Disk
    - Contains several layers
    - Gold reflective layer and dye layer
    - Records with high-power laser
    - Permanent marks on dye layer
    - CD cannot be erased after data recorded
  - Data read on standard CD drive (low-power beam)

CD and DVD Technology (cont'd.)

- **CD-Rewritable technology** (CD-RW)
  - Data written, changed, erased
  - Uses phase change technology
    - Amorphous and crystalline phase states
  - Record data: beam heats up disc
    - State changes from crystalline to amorphous
  - Erase data: low-energy beam to heat up pits
    - Loosens alloy to return to original crystalline state
  - Drives read standard CD-ROM, CD-R, CD-RW discs
  - Drives store large quantities of data, sound, graphics, multimedia

CD and DVD Technology (cont'd.)

- **DVD technology** (Digital Versatile Disc)
- CD-ROM comparison
  - Similar in design, shape, size
  - Differs in data capacity
    - Dual-layer, single-sided DVD holds 13 CDs
    - Single-layer, single-sided DVD holds 8.6 GB (MPEG video compression)
  - Differs in laser wavelength
    - Uses red laser (smaller pits, tighter spiral)
- DVDs cannot be read by CD or CD-ROM drives
- DVD-R and DVD-RW provide rewritable flexibility

Blu-Ray Disc Technology

- Same physical size as DVD/CD
- Smaller pits
- More tightly wound tracks
- Use of blue-violet laser allows multiple layers
- 50GB-500GB
- 432 Mbps
- Formats: BD-ROM, BD-R, BD-RE

Flash Memory Storage

- Electronically erasable programmable read-only memory (EEP)
  - Nonvolatile and removable
  - Emulates random access
    - Difference: data stored securely (even if removed)
- Data stored on microchip card or "key"
  - Compact flash, smart cards, memory sticks
  - Often connected through USB port
- Write data: electric charge sent through floating gate
- Erase data: strong electrical field (flash) applied

Components of the I/O Subsystem

- **I/O Channel**
  - Programmable units
    - Positioned between CPU and control unit
  - Synchronizes device speeds
    - CPU (fast) with I/O device (slow)
  - Manages concurrent processing
    - CPU and I/O device requests
  - Allows overlap
    - CPU and I/O operations
  - Channels: expensive because so often shared

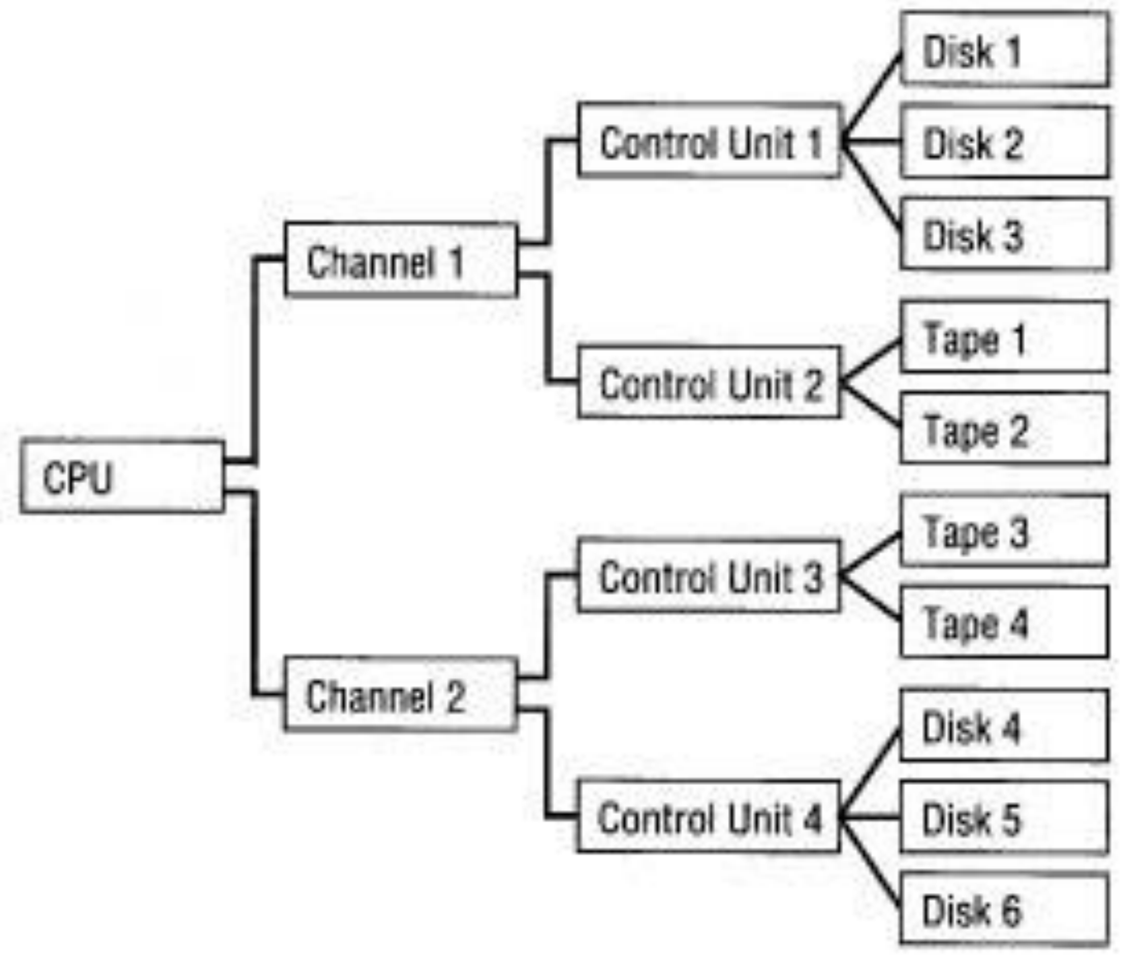
Components of the I/O Subsystem (cont'd.)

- **I/O channel programs**
  - Specifies action performed by devices
  - Controls data transmission
    - Between main memory and control units
- **I/O control unit**: receives and interprets signal
- **Disk controller** (disk drive interface)
  - Links disk drive and system bus
- Entire path must be available when I/O command initiated
- I/O subsystem configuration
  - Multiple paths increase flexibility and reliability

Components of the I/O Subsystem

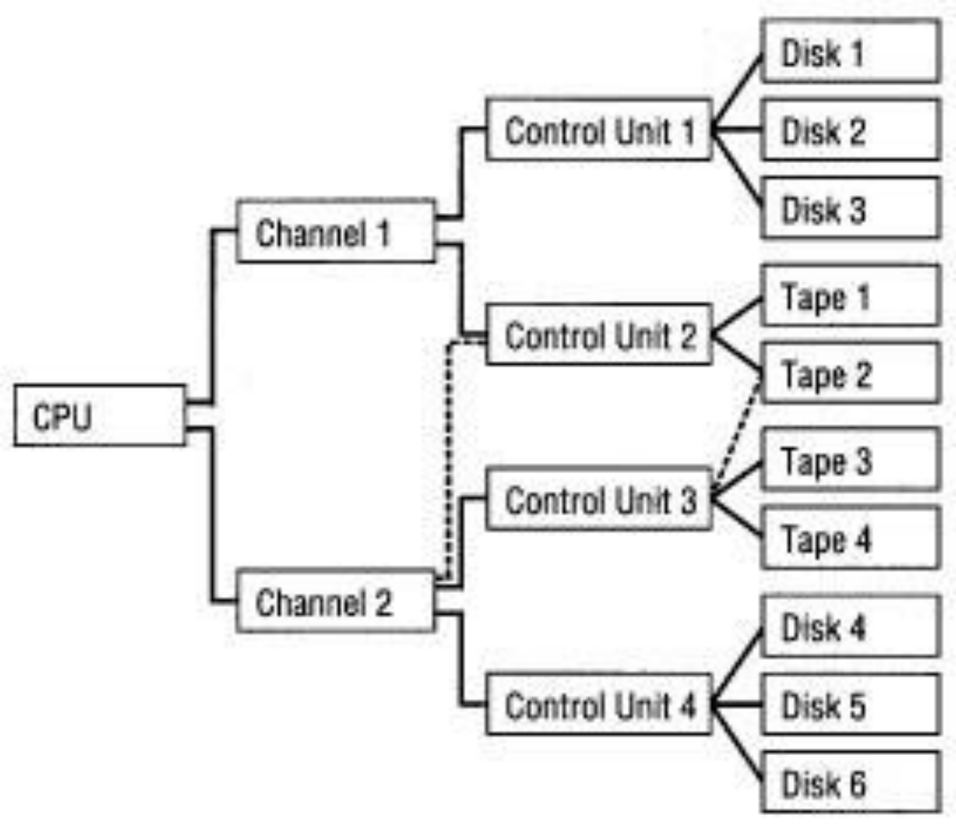
(figure 7.12)

Typical I/O subsystem configuration.



Components of the I/O Subsystem

.)



(figure 7.13)

I/O subsystem configuration with multiple paths, which increase both flexibility and reliability. With two additional paths, shown with dashed lines, if Control Unit 2 malfunctions, then Tape 2 can still be accessed via Control Unit 3.

Communication Among Devices

- Problems to resolve
  - Know which components are busy/free
    - Solved by structuring interaction between units
  - Accommodate requests during heavy I/O traffic
    - Handled by buffering records and queuing requests
  - Accommodate speed disparity between CPU and I/O devices
    - Handled by buffering records and queuing requests

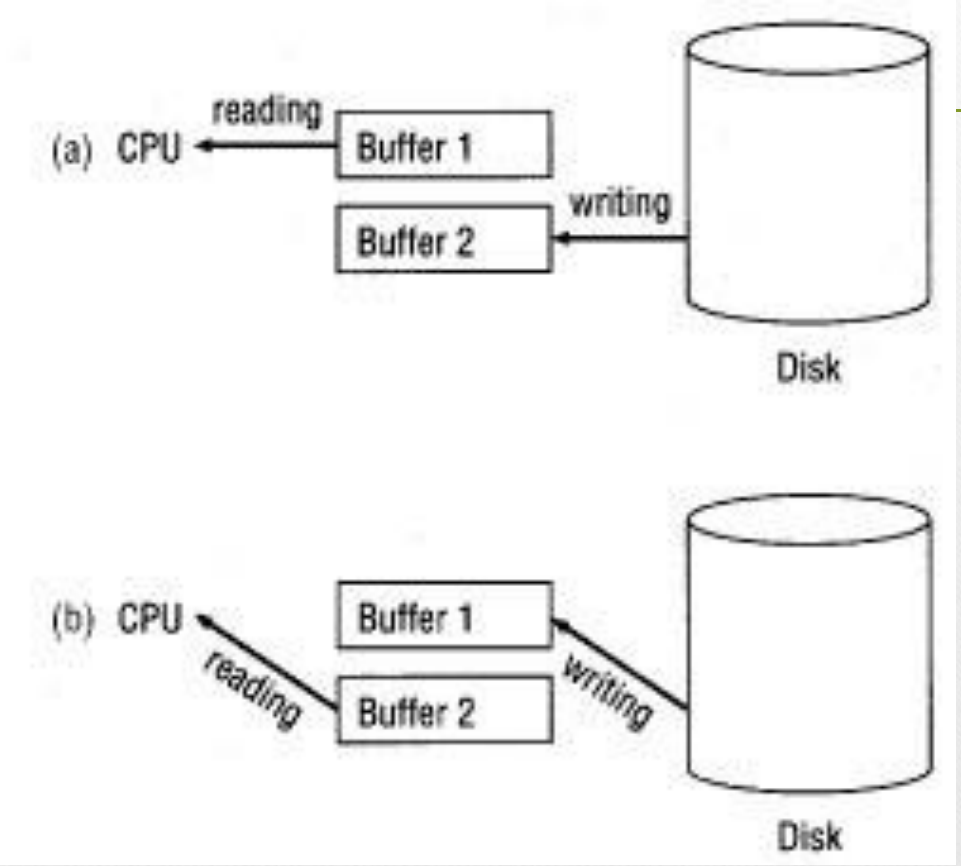
Communication Among Devices (cont'd.)

- I/O subsystem units finish independently of others
- CPU processes data while I/O performed
- Success requires device completion knowledge
    - Hardware flag tested by CPU
        - **Channel status word (CSW)** contains flag
        - Three bits in flag represent I/O system component (channel, control unit, device)
        - Changes zero to one (free to busy)
    - Flag tested using polling and interrupts
        - **Interrupts** are more efficient way to test flag

Communication Among Devices (cont'd.)

- Direct memory access (DMA)
  - Allows control unit main memory access directly
  - Transfers data without the intervention of CPU
  - Used for high-speed devices (disk)
- Buffers
  - Temporary storage areas in main memory, channels, control units
  - Improves data movement synchronization
    - Between relatively slow I/O devices and very fast CPU
  - Double buffering: processing of record by CPU while another is read or written by channel

Communication Among Devices (cont'd.)



(figure 7.14)

Example of double buffering: (a) the CPU is reading from Buffer 1 as Buffer 2 is being filled; (b) once Buffer 2 is filled, it can be read quickly by the CPU while Buffer 1 is being filled again.

Management of I/O Requests

- **I/O traffic controller**
  - Watches status of devices, control units, channels
  - Three main tasks
    - Determine if path available
    - If more than one path available, determine which one to select
    - If paths all busy, determine when one is available
  - Maintain database containing unit status and connections

Management of I/O Requests (cont'd.)

- **I/O scheduler**
  - Same job as process scheduler (Chapter 4)
  - Allocates devices, control units, channels
  - If requests greater than available paths
    - Decides which request to satisfy first: based on different criteria
  - In many systems
    - I/O requests not preempted
  - For some systems
    - Allow preemption with I/O request subdivided
    - Allow preferential treatment for high-priority requests

Management of I/O Requests (cont'd.)

- **I/O device handler**
  - Performs actual data transfer
    - Processes device interrupts
    - Handles error conditions
    - Provides detailed scheduling algorithms
  - Device dependent
  - Each I/O device type has device handler algorithm

Device Handler Seek Strategies

- Predetermined device handler
  - Determines device processing order
  - Goal: minimize seek time
- Types
  - First-come, first-served (FCFS), shortest seek time first (SSTF), SCAN (including LOOK, N-Step SCAN, C-SCAN, and C-LOOK)
- Scheduling algorithm goals
  - Minimize arm movement
  - Minimize mean response time
  - Minimize variance in response time

First Come First Serve (FCFS)

- We illustrate scheduling algorithms with a request queue (0-199)

98, 183, 37, 122, 14, 124, 65, 67

Head pointer 53

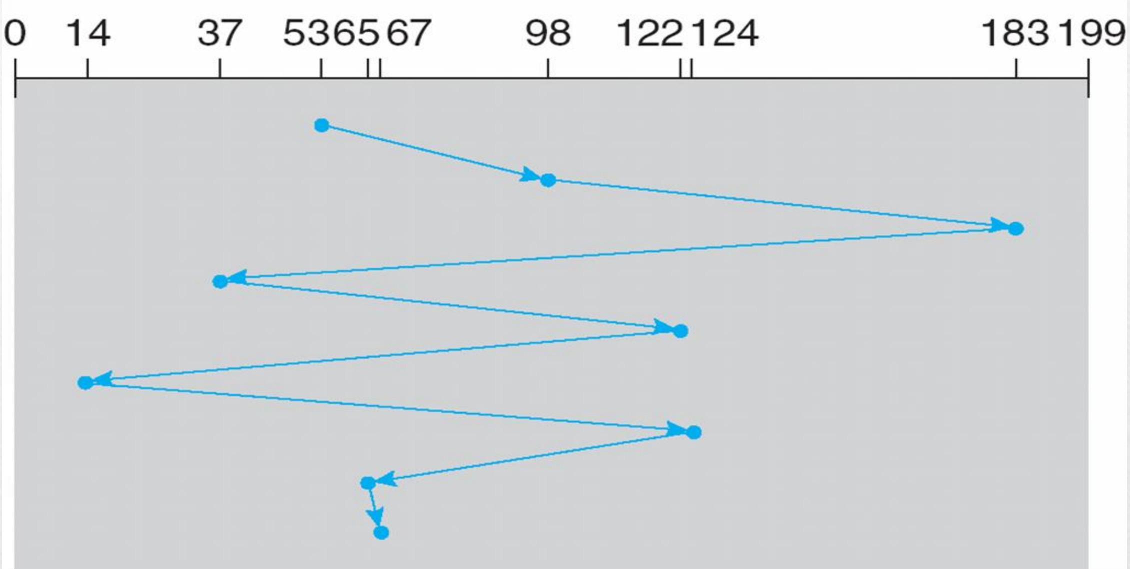


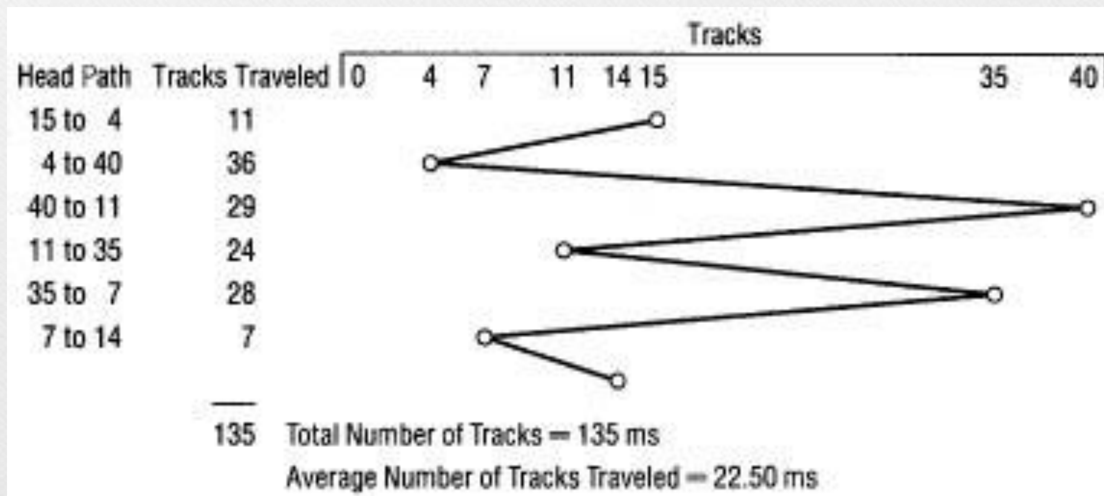
Illustration shows total head movement of 640 cylinders

Device Handler Seek Strategies (cont'd.)

- **FCFS**
  - On average: does not meet three seek strategy goals
  - Disadvantage: extreme arm movement

(figure 7.15)

The arm makes many time-consuming movements as it travels from track to track to satisfy all requests in FCFS order.



Shortest Seek Time First (SSTF)

- Shortest Seek Time First selects the request with the minimum seek time from the current head position

- We illustrate scheduling algorithms with a request queue (0-199)

  98, 183, 37, 122, 14, 124, 65, 67

  Head pointer 53

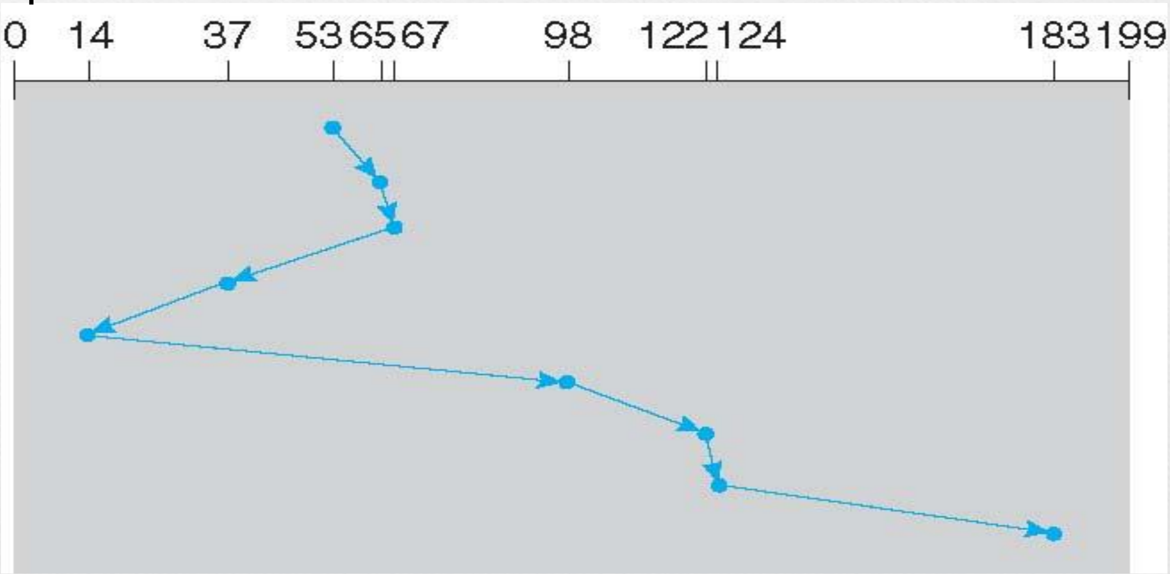
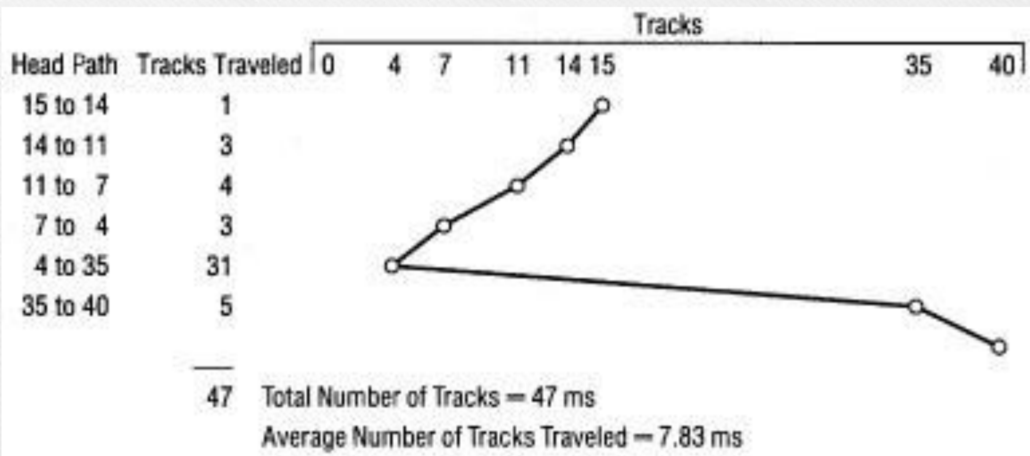


Illustration shows total head movement of 236 cylinders

Device Handler Seek Strategies (cont'd.)

- **Shortest Seek Time First (SSTF)**
  - Request with track closest to one being served
  - Minimizes overall seek time
  - Postpones traveling to out of way tracks



(figure 7.16)

Using the SSTF algorithm, with all track requests on the wait queue, arm movement is reduced by almost one third while satisfying the same requests shown in Figure 7.15 (using the FCFS algorithm).

SCAN

- **SCAN**
  - Directional bit
    - Indicates if arm moving toward/away from disk center
  - Algorithm moves arm methodically
    - From outer to inner track, services every request in its path
    - If reaches innermost track, reverses direction and moves toward outer tracks
    - Services every request in its path
- The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.
- **SCAN algorithm** Sometimes called the **elevator algorithm**

Device Handler Seek Strategies (cont'd.)

- We illustrate scheduling algorithms with a request queue (0-199). Assume arm move to the beginning of the queue

98, 183, 37, 122, 14, 124, 65, 67

Head pointer 53

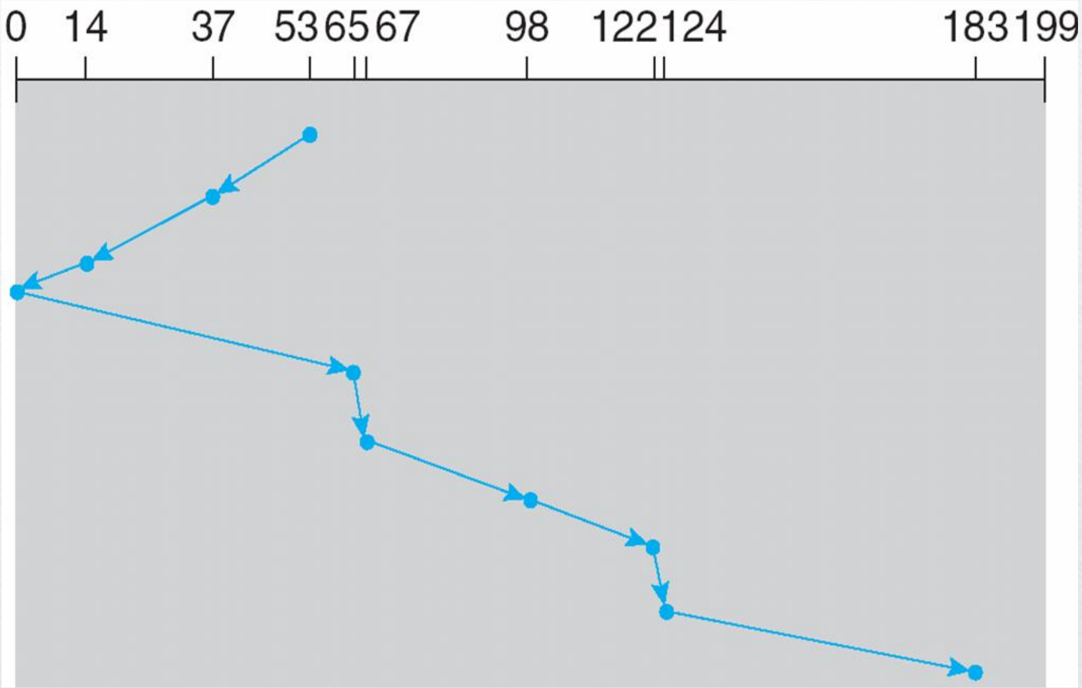


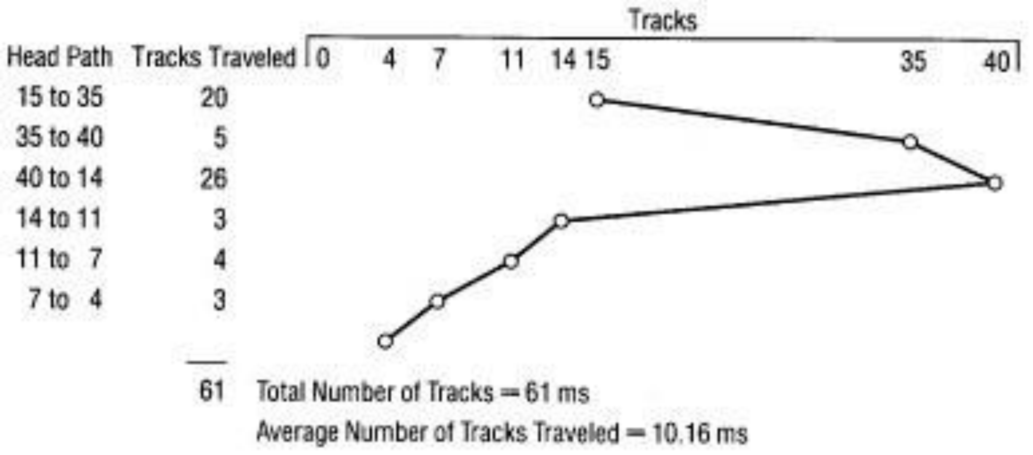
Illustration shows total head movement of 208 cylinders

Device Handler Seek Strategies (cont'd.)

- **LOOK**
  - Arm does not go to either edge
    - Unless requests exist
  - Eliminates indefinite postponement

(figure 7.17)

The LOOK algorithm makes the arm move systematically from the first requested track at one edge of the disk to the last requested track at the other edge. In this example, all track requests are on the wait queue.



Circular SCAN

- **C-SCAN (Circular SCAN)**
  - The head moves from one end of the disk to the other, servicing requests as it goes
  - When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip
  - Provides more uniform wait time

Device Handler Seek Strategies (cont'd.)

- We illustrate scheduling algorithms with a request queue (0-199). Assume arm move to the end of the disk

98, 183, 37, 122, 14, 124, 65, 67

Head pointer 53

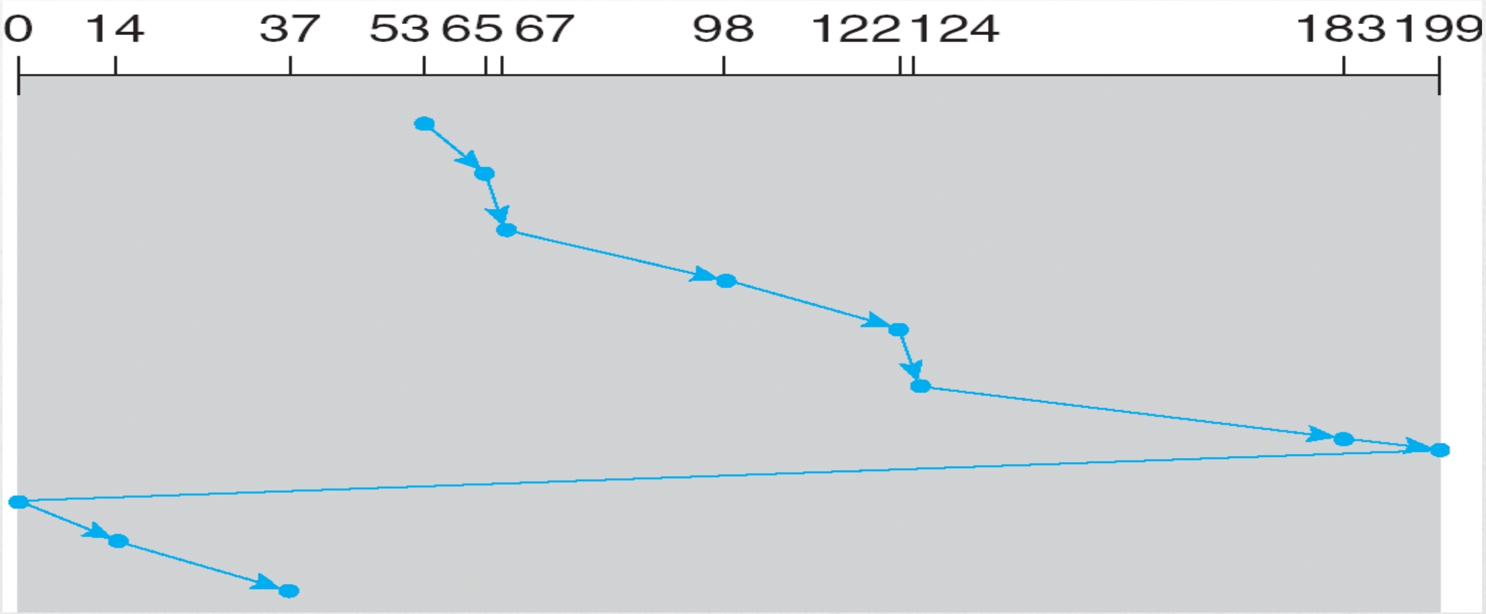


Illustration shows total head movement of 382 cylinders

Circular LOOK

- **C-LOOK**
  - LOOK a version of SCAN, C-LOOK a version of C-SCAN
  - Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk

Device Handler Seek Strategies (cont'd.)

- We illustrate scheduling algorithms with a request queue (0-199).Assume arm move towards the end of the disk

98, 183, 37, 122, 14, 124, 65, 67

Head pointer 53

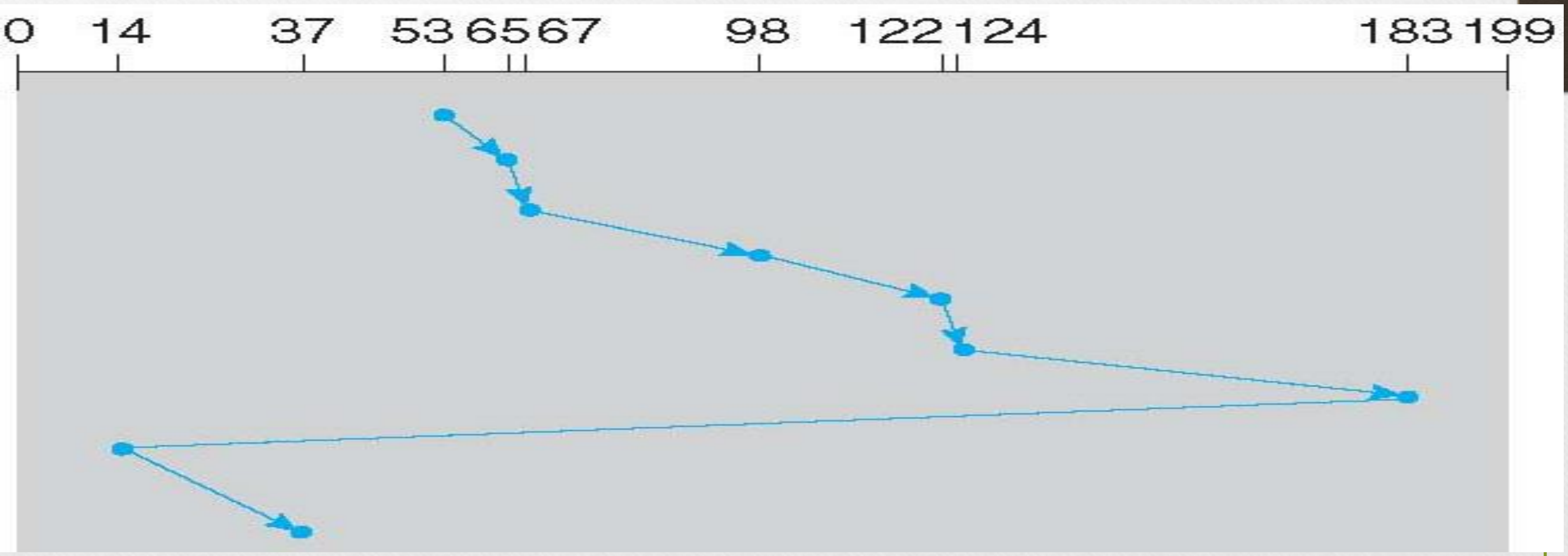


Illustration shows total head movement of 338 cylinders

Device Handler Seek Strategies (cont'd.)

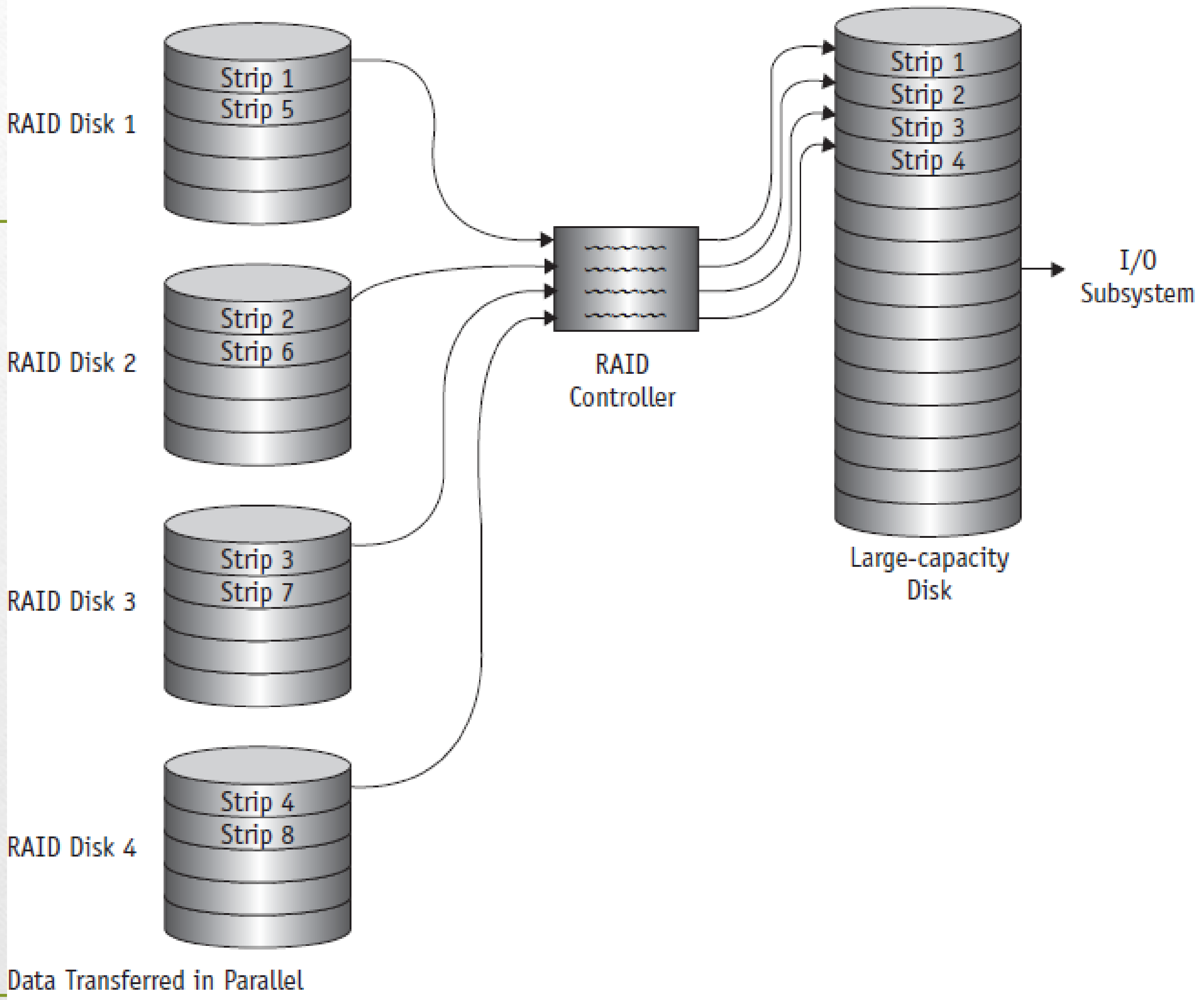
- Best strategy
  - FCFS best with light loads
    - Service time unacceptably long under high loads
  - SSTF best with moderate loads
    - Localization problem under heavy loads
  - SCAN best with light to moderate loads
    - Eliminates indefinite postponement
      - Throughput and mean service times SSTF similarities
  - C-SCAN best with moderate to heavy loads
    - Very small service time variances

RAID

- Physical disk drive set viewed as single logical unit
  - Preferable over few large-capacity disk drives
- Improved I/O performance
- Improved data recovery
  - Disk failure event
- Introduces redundancy
  - Helps with hardware failure recovery
- Significant factors in RAID level selection
  - Cost, speed, system's applications
- Increases hardware costs

(figure 7.19)

Data being transferred in parallel from a Level 0 RAID configuration to a large-capacity disk. The software in the controller ensures that the strips are stored in correct order.



RAID (cont'd.)

RAID Level	Error Correction Method	I/O Request Rate	Data Transfer Rate
0	None	Excellent	Excellent
1	Mirroring	Read: Good Write: Fair	Read: Fair Write: Fair
2	Hamming code	Poor	Excellent
3	Word parity	Poor	Excellent
4	Strip parity	Read: Excellent Write: Fair	Read: Fair Write: Poor
5	Distributed strip parity	Read: Excellent Write: Fair	Read: Fair Write: Poor
6	Distributed strip parity and independent data check	Read: Excellent Write: Poor	Read: Fair Write: Poor

(table 7.7)

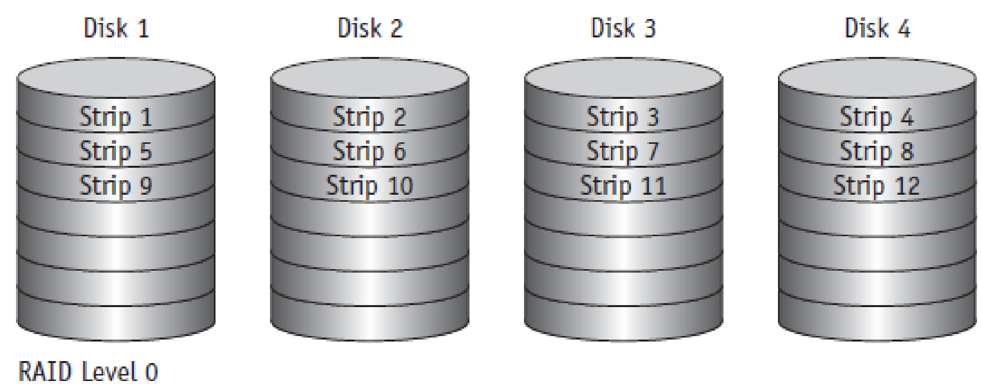
The seven standard levels of RAID provide various degrees of error correction. Cost, speed, and the system's applications are significant factors to consider when choosing a system.

Level Zero

- Uses data striping (not considered true RAID)
  - ~~No parity and error corrections~~
  - No error correction/redundancy/recovery
- Benefits
  - Devices appear as one logical unit
  - Best for large data quantity non-critical data

(figure 7.20)

RAID Level 0 with four disks in the array. Strips 1, 2, 3, and 4 make up a stripe. Strips 5, 6, 7, and 8 make up another stripe, and so on.



RAID 0 – Striping

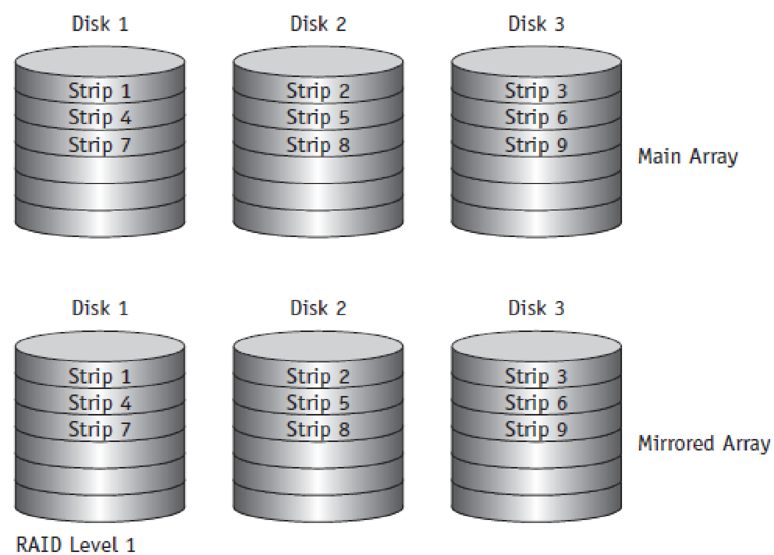
- **Concept**: Data is split into blocks and distributed (striped) across multiple disks. There is **no redundancy or parity**.
- **Read Performance**: Very high, as multiple disks can be read in parallel. Ideal for scenarios requiring fast sequential or random read access.
- **Write Performance**: Excellent, as data is written in parallel across all drives, allowing for high throughput.
- **Fault Tolerance**: **None**. If any one disk fails, all data is lost because there is no redundancy or parity information.
- **Recovery**: Impossible without backups. RAID 0 is **not suitable for critical data storage**.
- **Use Case**: Best for high-performance needs where **data loss is acceptable**, such as temporary files, gaming, or video editing scratch drives.

Level One

- Uses data striping (considered true RAID)
  - Mirrored configuration (backup)
    - Duplicate set of all data (expensive)
  - Provides redundancy and improved reliability

(figure 7.21)

RAID Level 1 with three disks in the main array and three corresponding disks in the backup array, the mirrored array.

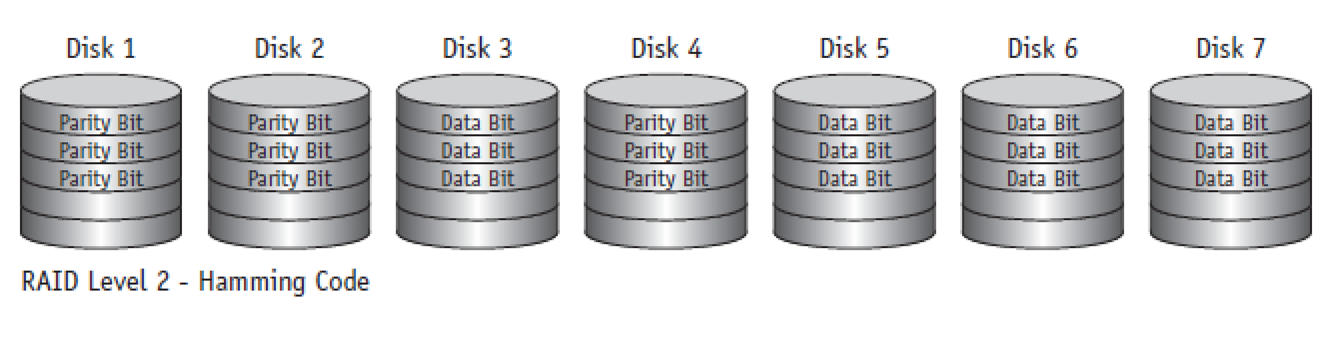


RAID 1 – Mirroring

- **Concept**: Data is duplicated across two (or more) drives. Every write is simultaneously written to each mirror disk.
- **Read Performance**: Very good. Some RAID controllers can read from both drives simultaneously to improve performance.
- **Write Performance**: Slower than RAID 0. All data must be written to all mirror disks, which adds write latency.
- **Fault Tolerance**: Very high. RAID 1 can **tolerate the failure of one disk**. The system continues to operate using the remaining mirror.
- **Recovery**: Simple. Once the failed disk is replaced, data is copied from the working disk to restore the mirror.
- **Use Case**: Ideal for systems that require **high availability and simple recovery**, such as OS boot drives, financial systems, or small business servers.

Level Two

- Uses small stripes (considered true RAID)
- **Hamming code**: error detection and correction
- Expensive and complex
  - Size of strip determines number of array disks



(figure 7.22)

RAID Level 2. Seven disks are needed in the array to store a 4-bit data item, one for each bit and three for the parity bits. Each disk stores either a bit or a parity bit based on the Hamming code used for redundancy.

RAID 2 – Bit-level Striping with Hamming Code (Obsolete)

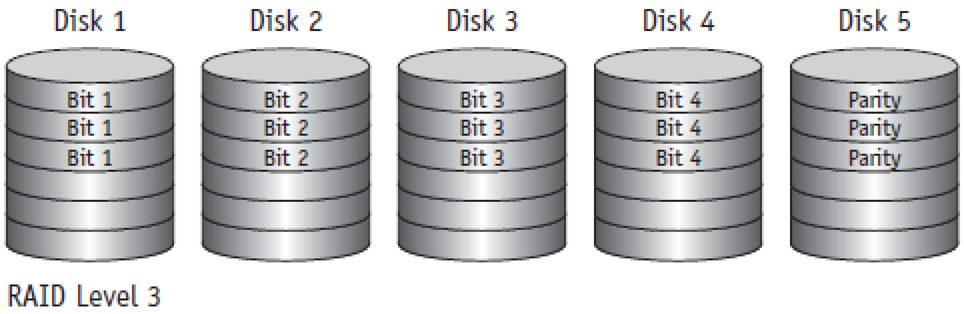
- **Concept**: Data is striped at the **bit level** across multiple disks, and **Hamming code** is used for error correction.
- **Read Performance**: Good, as it can read in parallel, though the bit-level striping is inefficient.
- **Write Performance**: Slower due to complexity of Hamming code error correction and synchronization requirements.
- **Fault Tolerance**: Good. It can **detect and correct errors**, even when multiple bits are corrupted.
- **Recovery**: Complex and slow. Requires decoding using error-correcting codes.
- **Use Case**: Rarely used today. Was originally designed for mainframe environments but has been replaced by simpler and faster RAID levels.

Level Three

- Modification of Level 2
  - Requires one disk for redundancy
    - One parity bit for each strip

(figure 7.23)

RAID Level 3. A 4-bit data item is stored in the first four disks of the array. The fifth disk is used to store the parity for the stored data item.



RAID 3 – Byte-level Striping with Dedicated Parity (Obsolete)

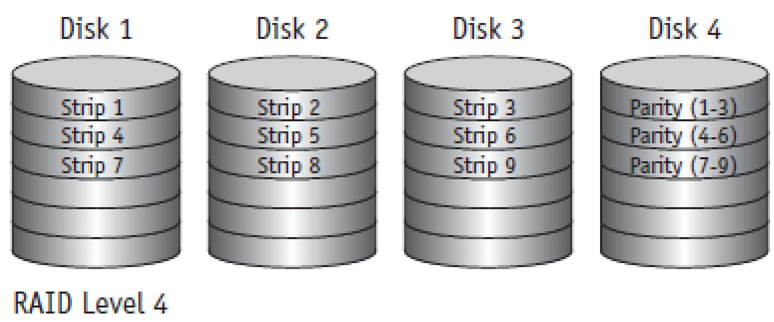
- **Concept**: Data is striped at the **byte level** across all disks, with one dedicated disk used for parity.
- **Read Performance**: Fair. Parallel read is possible but limited by byte-level striping.
- **Write Performance**: Slower due to bottleneck at the parity disk. Every write requires parity update.
- **Fault Tolerance**: Can **survive the failure of one disk**. Parity allows data reconstruction.
- **Recovery**: Requires reading from all other disks and reconstructing lost data using parity – relatively slow.
- **Use Case**: Obsolete. Replaced by RAID 5 which distributes parity and uses block-level striping.

Level Four

- Same strip scheme as Levels 0 and 1
  - Computes parity for each strip
  - Stores parities in corresponding strip
    - Has designated parity disk

(figure 7.24)

RAID Level 4. The array contains four disks: the first three are used to store data strips, and the fourth is used to store the parity of those strips.



RAID 4 – Block-level Striping with Dedicated Parity (Obsolete)

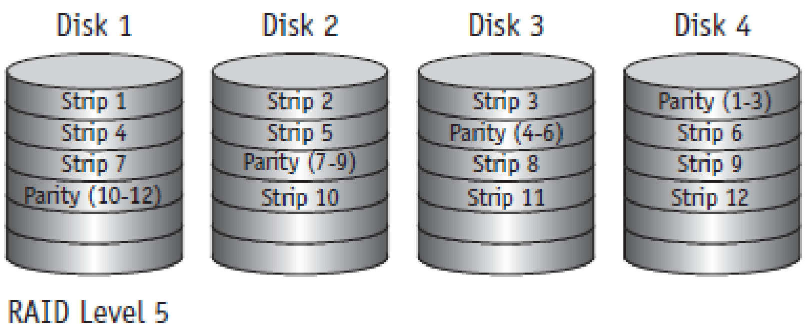
- **Concept**: Like RAID 3 but with **block-level** striping instead of byte-level. One disk stores parity.
- **Read Performance**: Good. Data can be read from multiple disks in parallel.
- **Write Performance**: Slower for small writes due to parity disk bottleneck. Requires reading old data and parity, recalculating and writing.
- **Fault Tolerance**: Can tolerate **one disk failure**. Parity helps in recovery.
- **Recovery**: Similar to RAID 3 – relatively slow and parity-disk dependent.
- **Use Case**: Rarely used today. RAID 5 is preferred for better performance and balanced load.

Level Five

- Modification of Level 4
- Distributes parity strips across disks
  - Avoids Level 4 bottleneck
- Disadvantage
  - Complicated to regenerate data from failed device

(figure 7.25)

RAID Level 5 with four disks. Notice how the parity strips are distributed among the disks.

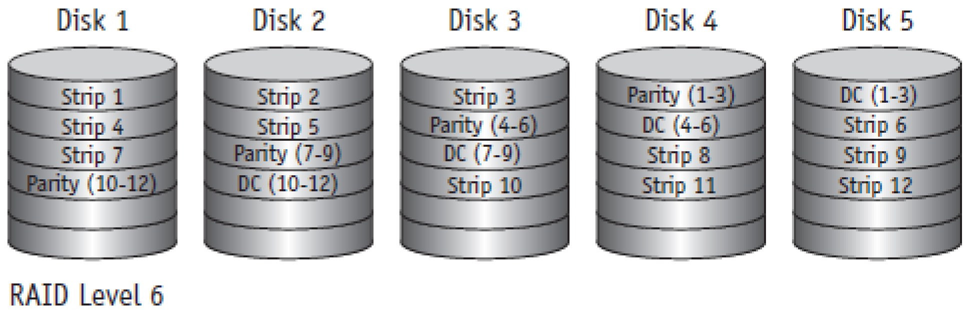


RAID 5 – Block-level Striping with Distributed Parity

- **Concept**: Data and parity are striped across all disks. Parity blocks are distributed to avoid a bottleneck.
- **Read Performance**: Very good. Can read from all disks simultaneously.
- **Write Performance**: Moderate. Every write involves reading old data and parity, recalculating, then writing data and parity – causing some overhead.
- **Fault Tolerance**: Can **tolerate the failure of one disk**. Parity enables recovery.
- **Recovery**: Slower than RAID 1. Reconstructs lost data using parity, which involves reading from all surviving disks.
- **Use Case**: Popular in enterprise storage where **good performance and fault tolerance** are required without high cost.

Level Six

- Provides extra degree of error protection/correction
  - Two different parity calculations (double parity)
    - Same as level four/five and independent algorithm
  - Parities stored on separate disk across array
    - Stored in corresponding data strip
- Advantage: data restoration even if two disks fail



(figure 7.26)

RAID Level 6. Notice how parity strips and data check (DC) strips are distributed across the disks.

RAID 6 – Block-level Striping with Double Distributed Parity

- **Concept**: Similar to RAID 5, but with **two parity blocks** distributed across all disks. Offers higher fault tolerance.
- **Read Performance**: Very good. Similar to RAID 5, though slightly reduced due to more parity.
- **Write Performance**: Slower than RAID 5. Two parity blocks need to be calculated and written, increasing overhead.
- **Fault Tolerance**: Can **tolerate the failure of two disks** simultaneously.
- **Recovery**: Slower and more compute-intensive due to dual-parity reconstruction.
- **Use Case**: Ideal for large storage arrays or **mission-critical systems** where extended fault tolerance is needed, such as archival or backup servers.

RAID COMPARISON

RAID Level	Read Performance	Write Performance	Fault Tolerance	Recovery
RAID 0 (Striping)	**Excellent** – Data is read from multiple disks in parallel.	**Excellent** – Data is written across all disks simultaneously.	❌ **None** – If one disk fails, all data is lost.	❌ **No recovery** possible.
RAID 1 (Mirroring)	**Very Good** – Can read from either disk (some controllers optimize read).	**Moderate** – Writes must occur on both disks simultaneously.	✅ **High** – Can survive **one disk** failure.	✅ **Fast recovery** – Simply copy data from the remaining mirror.
RAID 2 (Hamming ECC)*	**Good** – Parallel reads like RAID 0.	**Slower** – Parity checking adds overhead.	✅ **High** – Can detect and correct some errors.	⚠️ **Complex** – Rarely used; replaced by modern ECC systems.
RAID 3 (Byte-level striping with dedicated parity)*	**Good** – Parallel read possible.	**Slower** – Single parity disk becomes a bottleneck during writes.	✅ **Can tolerate 1 disk failure.**	⚠️ **Slower recovery** – Requires parity reconstruction.
RAID 4 (Block-level striping with dedicated parity)*	**Good** – Data read from multiple disks.	**Slower** – Like RAID 3, single parity disk is a bottleneck.	✅ **Can tolerate 1 disk failure.**	⚠️ **Recovery via parity disk reconstruction.**
RAID 5 (Block-level striping with distributed parity)	**Good to Excellent** – Read from all disks.	**Moderate** – Parity calculation adds overhead.	✅ **Can tolerate 1 disk failure.**	✅ **Recovery via distributed parity – longer but feasible.**
RAID 6 (Like RAID 5 + extra parity)	**Good** – Slightly slower than RAID 5 due to more parity data.	**Slower** – Two parity calculations per write.	✅✅ **Can tolerate 2 disk failures.**	⚠️ **Slower recovery** – More complex due to dual parity.

RAID COMPARISON

RAID Level	Read Speed	Write Speed	Fault Tolerance	Storage Efficiency	Best Use Case
RAID 0	⭐⭐⭐⭐	⭐⭐⭐⭐	❌ None	⭐⭐⭐⭐ (100%)	Speed > Safety (e.g., video editing)
RAID 1	⭐⭐⭐	⭐⭐	✅ 1 disk	⭐⭐ (50%)	Simple redundancy (e.g., OS drives)
RAID 2	⭐⭐	⭐	✅ ECC	⭐⭐	Legacy use (mainframes – obsolete)
RAID 3	⭐⭐	⭐	✅ 1 disk	⭐⭐⭐	Sequential access workloads (obsolete)
RAID 4	⭐⭐	⭐	✅ 1 disk	⭐⭐⭐	Rare today – parity bottleneck
RAID 5	⭐⭐⭐	⭐⭐	✅ 1 disk	⭐⭐⭐ (N-1/N)	Balanced (SMEs, NAS, servers)
RAID 6	⭐⭐	⭐	✅✅ 2 disks	⭐⭐ (N-2/N)	High fault tolerance (large arrays)

RAID COMPARISON

RAID Level	Read Speed	Write Speed	Fault Tolerance	Storage Efficiency	Best Use Case
RAID 0	⭐⭐⭐⭐	⭐⭐⭐⭐	❌ None	⭐⭐⭐⭐ (100%)	Speed > Safety (e.g., video editing)
RAID 1	⭐⭐⭐	⭐⭐	✅ 1 disk	⭐⭐ (50%)	Simple redundancy (e.g., OS drives)
RAID 2	⭐⭐	⭐	✅ ECC	⭐⭐	Legacy use (mainframes – obsolete)
RAID 3	⭐⭐	⭐	✅ 1 disk	⭐⭐⭐	Sequential access workloads (obsolete)
RAID 4	⭐⭐	⭐	✅ 1 disk	⭐⭐⭐	Rare today – parity bottleneck
RAID 5	⭐⭐⭐	⭐⭐	✅ 1 disk	⭐⭐⭐ (N-1/N)	Balanced (SMEs, NAS, servers)
RAID 6	⭐⭐	⭐	✅✅ 2 disks	⭐⭐ (N-2/N)	High fault tolerance (large arrays)

Which RAID Is Better? (Contextual Evaluation)

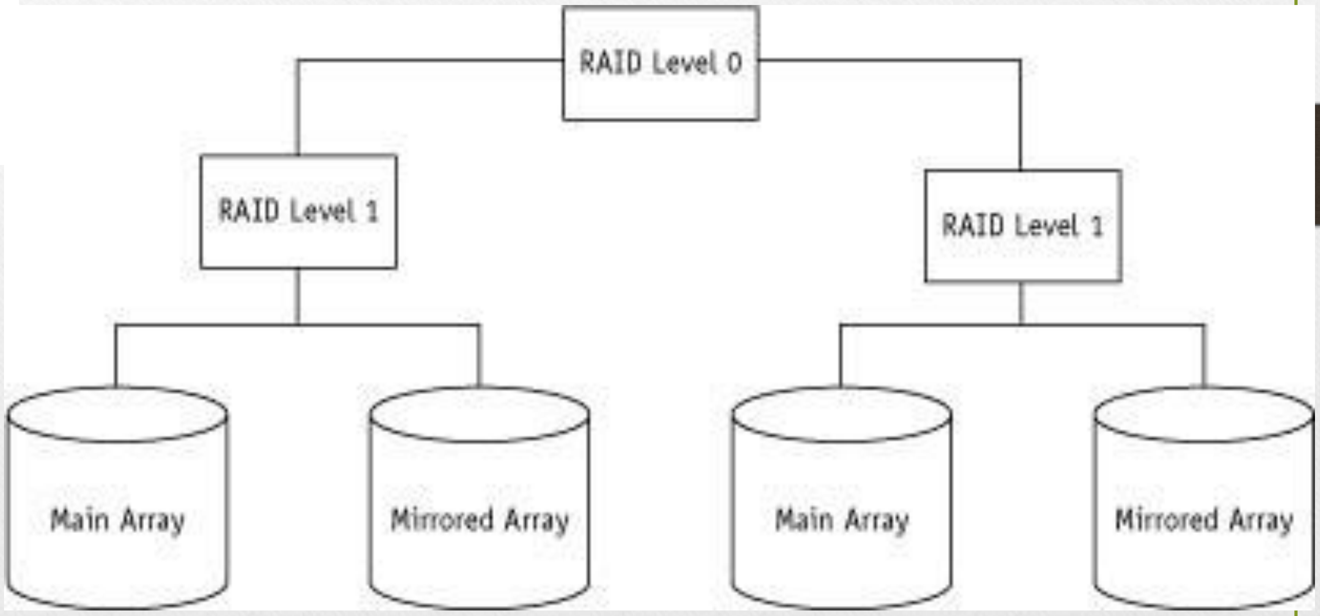
- **1. Best for Performance**
  - Winner: RAID 0
  - Highest read/write performance.
  - No fault tolerance – not for critical data.
- **2. Best for Simple Redundancy**
  - Winner: RAID 1
  - Easy to set up and recover.
  - Only 50% disk space efficiency.
- **3. Best for Balance (Performance + Fault Tolerance + Efficiency)**
  - Winner: RAID 5
  - Offers good read performance, decent write speed.
  - Can tolerate 1 disk failure with efficient disk usage.
- **4. Best for Maximum Fault Tolerance**
  - Winner: RAID 6
  - Can survive 2 simultaneous disk failures.
  - Slightly slower writes and more disk space used for parity

Nested RAID Levels

- Combines multiple RAID levels (complex)

(figure 7.27)

A RAID Level 10 system.



Nested RAID Levels (cont'd.)

RAID Level	Combinations
01 (or 0+1)	A Level 1 system consisting of multiple Level 0 systems
10 (or 1+0)	A Level 0 system consisting of multiple Level 1 systems
03 (or 0+3)	A Level 3 system consisting of multiple Level 0 systems
30 (or 3+0)	A Level 0 system consisting of multiple Level 3 systems
50 (or 5+0)	A Level 0 system consisting of multiple Level 5 systems
60 (or 6+0)	A Level 0 system consisting of multiple Level 6 systems

(table 7.8)

Some common nested RAID configurations. Important: RAID 01 and 03 are not to be confused with RAID Levels 1 and 3, respectively.

Summary

- Device Manager
  - Manages every system device effectively as possible
- Devices
  - Vary in speed and sharability degrees
  - Direct access and sequential access
- Magnetic media: one or many read/write heads
  - Heads in a fixed position (optimum speed)
  - Move across surface (optimum storage space)
- Optical media: disk speed adjusted
  - Data recorded/retrieved correctly

Summary (cont'd.)

- Flash memory: device manager tracks USB devices
  - Assures data sent/received correctly
- I/O subsystem success dependence
  - Communication linking channels, control units, devices
- SCAN: eliminates indefinite postponement problem
  - Best for light to moderate loads
- C-SCAN: very small service time variance
  - Best for moderate to heavy loads
- RAID: redundancy helps hardware failure recover
  - Consider cost, speed, applications

Strategy	Advantages	Disadvantages
FCFS	• Easy to implement • Sufficient for light loads	• Doesn't provide best average service • Doesn't maximize throughput
SSTF	• Throughput better than FCFS • Tends to minimize arm movement • Tends to minimize response time	• May cause starvation of some requests • Localizes under heavy loads
SCAN/LOOK	• Eliminates starvation • Throughput similar to SSTF • Works well with light to moderate loads	• Needs directional bit • More complex algorithm to implement • Increased overhead
N-Step SCAN	• Easier to implement than SCAN	• The most recent requests wait longer than with SCAN
C-SCAN/C-LOOK	• Works well with moderate to heavy loads • No directional bit • Small variance in service time • C-LOOK doesn't travel to unused tracks	• May not be fair to recent requests for high-numbered tracks • More complex algorithm than N-Step SCAN, causing more overhead

(table 7.9)

Comparison of DASD seek strategies discussed in this chapter.