OPERATING SYSTEM

Unit V — I/O Management & File Systems

Topics Covered in this Unit

System I/O Architecture: Memory Bus, General I/O Bus (PCIe) & Peripheral Buses (USB, SATA, NVMe)
The Canonical Device Model: Status, Command & Data Hardware Registers and Internal Microcontroller
The Canonical I/O Protocol: Polling vs. Interrupt-Driven I/O & Interrupt Service Routines (ISRs)
Direct Memory Access (DMA): Offloading High-Volume Data Transfers from CPU to Dedicated Hardware Controller
Methods of Device Interaction: Port-Mapped I/O (in/out assembly) vs. Memory-Mapped I/O (MMIO)
Device Drivers in the OS Architecture: Uniform Block and Character Device Abstractions & IDE Disk Driver Case Study
Hard Disk Drives (HDD): Geometry (Platters, Tracks, Sectors, Cylinders) & Latency Components (Seek, Rotation, Transfer)
Disk Scheduling Algorithms: FCFS, SSTF, SCAN (Elevator), C-SCAN, LOOK & C-LOOK
File System Architecture: Inodes (Metadata & Direct/Indirect Pointers), Directory Trees & Hard vs. Soft Links
The Fast File System (FFS): Cylinder Groups & Spatial Locality Optimization
Crash Consistency, Disk Failure Modes, Latent Sector Errors & Data Integrity Checksums (Fletcher, CRC32, ZFS)
High-Yield University Exam Question Bank with Detailed Model Answers & Unit Summary

Compiled in a structured, easy-to-understand, textbook style format
Specially curated for B.Tech / B.E. / BCA / MCA / B.Sc Computer Science & Information Technology Students

© Copyright — abhyasmitra.in — All Rights Reserved

COURSE SYLLABUS & MAPPING

[UNIT V SYLLABUS — PRESCRIBED UNIVERSITY CURRICULUM]

This section contains the official syllabus for Unit V. Verify with your university curriculum mapping.

Unit V Syllabus Breakdown:

Unit V I/O Management: System architecture, A Canonical device, The Canonical protocol, CPU virtualization, Lowering CPU Overhead with Interrupts, More efficient data movement With DMA, Methods of device interaction, Fitting into the OS: The device driver, Case Study: A simple IDE disk driver, Hard disk drives, files and directories, The fast file system, file system implementation, disk failure modes, handling latent sector error, detecting corruption: the checksum, using checksums

• Course Code: _____ • Semester / Year: _____

TABLE OF CONTENTS

5.1 System Architecture, Canonical Device & Protocols

The Canonical Device Architecture

A hardware I/O device presents two core components to the system:

1. Hardware Interface (Registers): Status Register (reads device state/busy), Command Register (tells device what task to perform), and Data Register (passes data in/out).

2. Device Internals: Hardware microcontroller, internal firmware, buffer RAM, and physical mechanical/electronic transducers.

The Canonical I/O Protocols: Polling vs. Interrupts vs. DMA

I/O Protocol	Operating Mechanism	CPU Overhead & Efficiency	Best Application
Programmed I/O (Polling / Busy-Waiting)	CPU repeatedly reads Status register in a tight loop until device indicates READY.	100% CPU utilization while waiting; wastes millions of CPU cycles on slow devices.	Ultra-fast devices where I/O completes in nanoseconds (network NIC in high-frequency trading).
Interrupt-Driven I/O	CPU issues command to device and immediately context-switches to another ready process. Device raises a hardware Interrupt line upon completion.	Zero wasted CPU cycles during I/O wait; small interrupt handler context-switch overhead.	Moderate/slow I/O devices (Keyboards, mice, disk operations).
Direct Memory Access (DMA)	Dedicated DMA Controller transfers entire multi-kilobyte data blocks directly between device buffer and physical RAM without CPU involvement.	CPU is interrupted only ONCE per large multi-block transfer, maximizing throughput.	High-bandwidth disk transfers, video streaming, Gigabit networking.

Methods of Device Interaction: Port I/O vs. MMIO

- • **Port-Mapped I/O (PMIO):** Uses dedicated CPU instructions (`in` and `out` in x86) with a separate I/O port address space.
- • **Memory-Mapped I/O (MMIO):** Hardware device registers are mapped directly into the CPU's physical memory address space. The OS interacts with devices using standard memory read/write instructions (`MOV`), simplifying device driver programming.

5.2 Hard Disk Drives (HDD) & Disk Scheduling

HDD Physical Latency Equation

Total Disk Access Time T_I/O is composed of three physical mechanical phases:

T_I/O = T_seek + T_rotational + T_transfer

- *Seek Time (T_seek): Time to physically move the read/write drive arm to the target concentric Track (typically 4-10ms - DOMINANT BOTTLENECK).*
- *Rotational Delay (T_rotational): Time for the target Sector to rotate under the head (Average = 1 / (2 × RPM)).*
- *Transfer Time (T_transfer): Time to read/write data bits as platter rotates under head.*

Disk Head Scheduling Algorithms

Algorithm	Selection Rule	Strengths & Gotchas
FCFS (First-Come, First-Served)	Serves disk I/O requests in the exact order of arrival.	Fair, zero starvation; causes wild, erratic seek swings across platters.
SSTF (Shortest Seek Time First)	Selects the request closest to the current head position (minimizing seek distance).	Significantly reduces average seek time; prone to Starvation of distant outer tracks.
SCAN (Elevator Algorithm)	Head sweeps continuously in one direction servicing requests until reaching disk edge, then reverses direction.	Fairer than SSTF; bounded wait times; edge tracks serviced less frequently.
C-SCAN (Circular SCAN)	Head sweeps in one direction servicing requests; upon reaching end, immediately returns to start without servicing requests on return.	Provides strictly uniform waiting time distribution across all tracks.
LOOK and C-LOOK	Enhanced versions of SCAN/C-SCAN that reverse or reset direction immediately after servicing the last request in current direction (without traveling to physical disk edge).	Industry standard for mechanical magnetic hard drive scheduling.

5.3 File Systems Implementation: Inodes & Directories

Definition: Inode (Index Node)

An Inode is the fundamental metadata data structure in Unix/Linux file systems that describes a file's attributes (File Size, Owner UID/GID, Permissions R/W/X, Timestamps: access, modify, change) and stores disk block pointers containing the actual file data.

Multi-Level Indexing in Inodes

- • **Direct Pointers:** Typically 12 direct pointers pointing straight to data blocks (supports small files with zero indirection overhead).
- • **Single Indirect Pointer:** Points to a disk block containing an array of direct pointers (e.g., 4KB block / 4-byte pointer = 1024 data blocks = 4 MB).
- • **Double Indirect Pointer:** Points to a block of single-indirect blocks (supports 1024 × 1024 × 4KB = 4 GB).
- • **Triple Indirect Pointer:** Points to a block of double-indirect blocks (supports 1024^3 × 4KB = 4 TB files).

Directories & Links

- • **Directory:** A special file containing a list of (Filename, Inode Number) mapping pairs.
- • **Hard Link:** Creates a new directory entry pointing to an EXISTING Inode number (increments Inode reference count). File data is only deleted when reference count reaches 0. Cannot span across different file systems.
- • **Soft Link (Symbolic Link):** A standalone independent file with its own unique Inode whose data block contains the pathname string of target file. Can cross file systems; becomes a Dangling Link if target is deleted.

5.4 Crash Consistency, Failure Modes & Checksums

1. The Crash Consistency Problem

Appending data to a file requires 3 separate disk writes: (1) Inode update, (2) Data block write, and (3) Data Bitmap update. If a system crash/power loss occurs mid-way, file system metadata is left in an inconsistent corrupted state.

- • **Journaling (Write-Ahead Logging - WAL):** Before modifying on-disk structures, the file system writes a transaction log to a dedicated Journal on disk, commits the transaction, and then checkpoints (writes) data to final locations. Upon reboot, the OS simply replays the journal in seconds.

2. Disk Failure Modes & Checksums

Definition: Checksum & Silent Data Corruption

• *Latent Sector Error (LSE): Physical drive damage where a sector becomes completely unreadable (returns I/O error).*

• *Silent Data Corruption (Bit Rot): Drive reads data successfully, but physical bits have flipped due to cosmic rays, electromagnetic interference, or buggy firmware.*

• *Checksum: A mathematical hash function C = f(Block) stored alongside data to detect bit corruption (e.g., XOR, Fletcher Checksum, CRC32, SHA-256). If f(Read_Data) ≠ Stored_Checksum, corruption is detected and repaired via RAID mirrors (ZFS / Btrfs).*

5.5 High-Yield University Exam Question Bank & Model Answers

Part A: Short Answer Questions (2–3 Marks)

- **Q1. What is Direct Memory Access (DMA)? Why is it important?**

Answer: DMA is a hardware architecture where a dedicated DMA Controller transfers high-volume data blocks directly between I/O device buffers and physical RAM without passing through the CPU. It frees the CPU from byte-by-byte data copying, allowing it to execute other user processes concurrently.

- **Q2. Differentiate between Hard Links and Soft (Symbolic) Links in Linux.**

Answer: A Hard Link is a directory entry pointing directly to the existing file's Inode number (shares Inode and permissions; cannot cross file systems). A Soft Link is an independent file with its own Inode whose data block contains the text path of the target file (can cross file systems; breaks if target is moved).

- **Q3. State the three physical components of Disk Access Time.**

Answer: (1) Seek Time (time to move drive head to target track - dominant latency), (2) Rotational Latency (time for sector to rotate under head), and (3) Transfer Time (time to read/write data bits).

- **Q4. How does Journaling solve the File System Crash Consistency problem?**

Answer: Journaling writes intent and metadata updates to a sequential Write-Ahead Log (Journal) on disk before writing to file system blocks. If a power crash occurs, the OS inspects the journal upon reboot and replays committed transactions, recovering consistency in seconds without full disk scans (fsck).

Part B: Long Answer Questions (8–10 Marks)

- **Q5. Explain the Inode structure in Unix File Systems with a complete multi-level index block diagram. Calculate maximum file size supported by 12 direct, 1 single, 1 double, and 1 triple indirect pointers with 4KB block size.**

Model Answer Outline:

- 1. Definition and purpose of Inode: File metadata and disk block pointers.
- 2. Multi-level indexing architecture: 12 Direct pointers, 1 Single indirect pointer, 1 Double indirect pointer, 1 Triple indirect pointer.
- 3. Mathematical capacity calculation (Block size = 4KB = 4096 bytes, Pointer size = 4 bytes → 1024 pointers/block):
- - Direct: 12 × 4KB = 48 KB.
- - Single Indirect: 1024 × 4KB = 4 MB.
- - Double Indirect: 1024 × 1024 × 4KB = 4 GB.
- - Triple Indirect: 1024 × 1024 × 1024 × 4KB = 4 TB.
- - Total Max File Size ≈ 4.004 TB.
- 4. Complete block pointer hierarchy diagram.

- **Q6. Describe Disk Scheduling Algorithms: FCFS, SSTF, SCAN, C-SCAN, LOOK, and C-LOOK. Solve a disk track seek problem with request queue [98, 183, 37, 122, 14, 124, 65, 67] starting at Head 53.**

Model Answer Outline:

- 1. Purpose of disk scheduling: Minimizing mechanical head movement and seek time.
- 2. Detailed operational rules for all 6 algorithms.
- 3. Step-by-step track seek numerical solution for SSTF, SCAN, and C-SCAN:
- - Head path sequence trace.
- - Calculation of Total Head Movements (cylinder seeks) for each algorithm.
- 4. Comparative evaluation matrix analyzing fairness, starvation risk, and throughput.

5.6 Unit V Summary & Quick Revision Sheet

I/O & File System Domain	Key Theoretical & Hardware Takeaways for Exams
Canonical Device & DMA	Registers (Status, Command, Data). Polling (Busy-wait) vs Interrupts vs DMA (Direct high-volume RAM transfer).
Disk Latency & Scheduling	T_I/O = T_seek (dominant) + T_rot + T_trans. Schedulers: SSTF (starvation), SCAN (elevator), C-SCAN (uniform wait), C-LOOK.
Inodes & File Systems	Inodes (Metadata + 12 Direct, 1 Single, 1 Double, 1 Triple indirect pointers). Directories map Name → Inode.

I/O & File System Domain	Key Theoretical & Hardware Takeaways for Exams
Links	Hard Link (Shares Inode, ref count, same fs) vs Soft Link (New Inode storing target path string, cross fs).
Crash Consistency & Integrity	Journaling (Write-Ahead Logging). Silent Data Corruption detected via Checksums (Fletcher, CRC32, ZFS tree).