NUB
NAHDA UNIVERSITY
IN BENI SUEF
جامعة النهضة . ببني سويف

Operating System 2- CS402

Lecture 4
I/O Interface, Kernel I/O Subsystem

2025

Learning Objectives

- Explain structuring techniques and interfaces for the operating system that enable I/O devices to be treated in a standard, uniform way.
- Discuss the main functions of Kernel I/O Subsystem.
- Explain the performance aspects of I/O hardware and software.

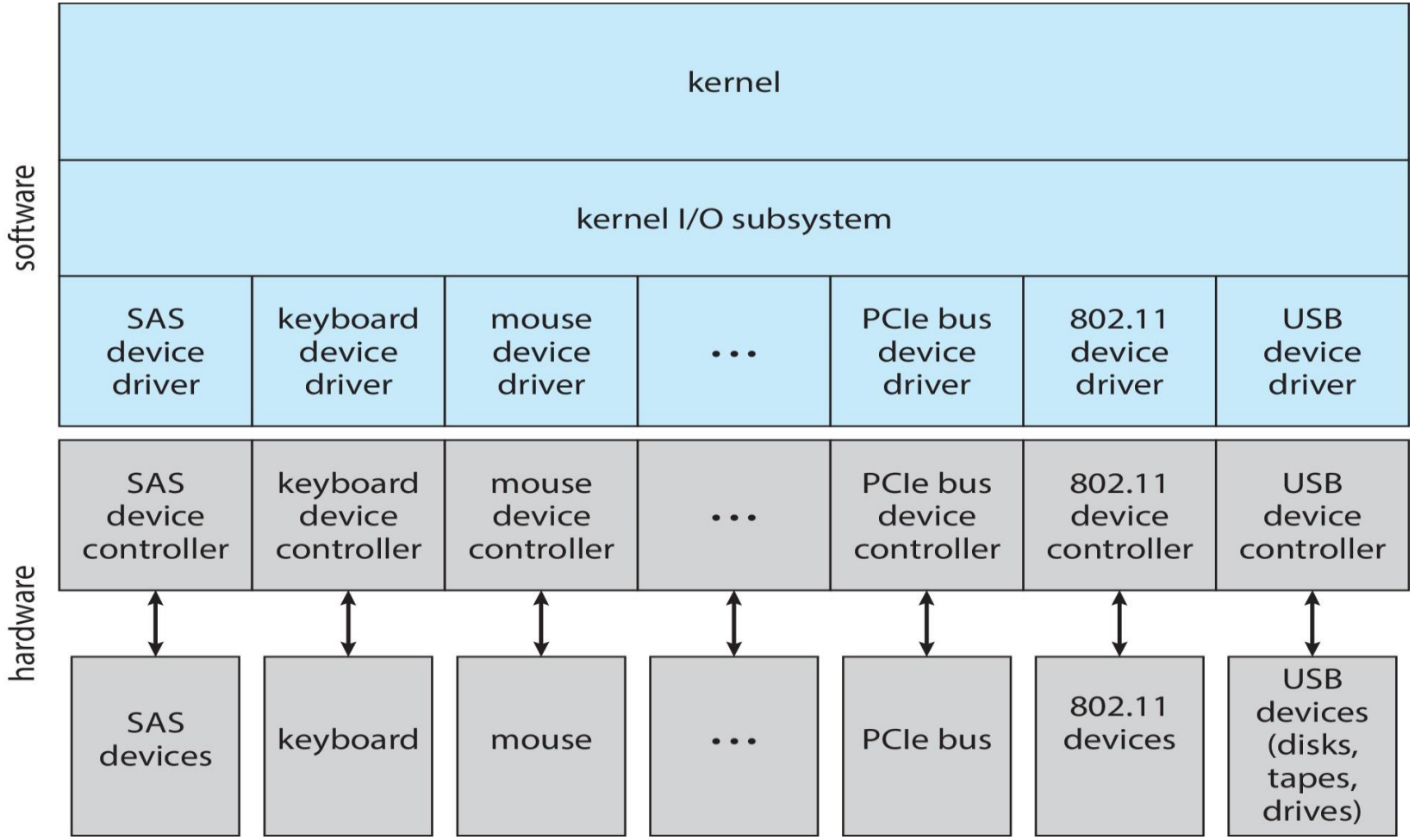
Uniform I/O Device Management

- How an application can open a file on a disk without knowing what kind of disk it is?
- How new disks and other devices can be added to a computer without disruption of the operating system?

Uniform I/O Device Management (cont.)

- Like other complex software-engineering problems, the approach here involves:
  - Abstraction: OS provides a unified interface (e.g., system calls like **open(), read(), write()**) to interact with devices without knowing hardware details.

A Kernel I/O Structure



How the I/O-related portions of the kernel are structured in software layers.

Characteristics of I/O devices

- Devices vary in many **dimensions**:
- Data-transfer mode:
  - A **character-stream** device transfers bytes one by one (keyboard, mouse).
  - A **block** device transfers a block of bytes as a unit (hard drives, SSDs).
- Access method:
  - A **sequential** device transfers data in a fixed order defined by the device (magnetic tapes, modem).
  - A **random-access** device user can instruct the device to seek to any of the available data storage locations (hard drives, SSDs).
- Transfer schedule:
  - A **synchronous** device performs data transfers with predictable response times (magnetic tapes).
  - An **asynchronous** device exhibits irregular or un predictable response times (keyboard, mouse).

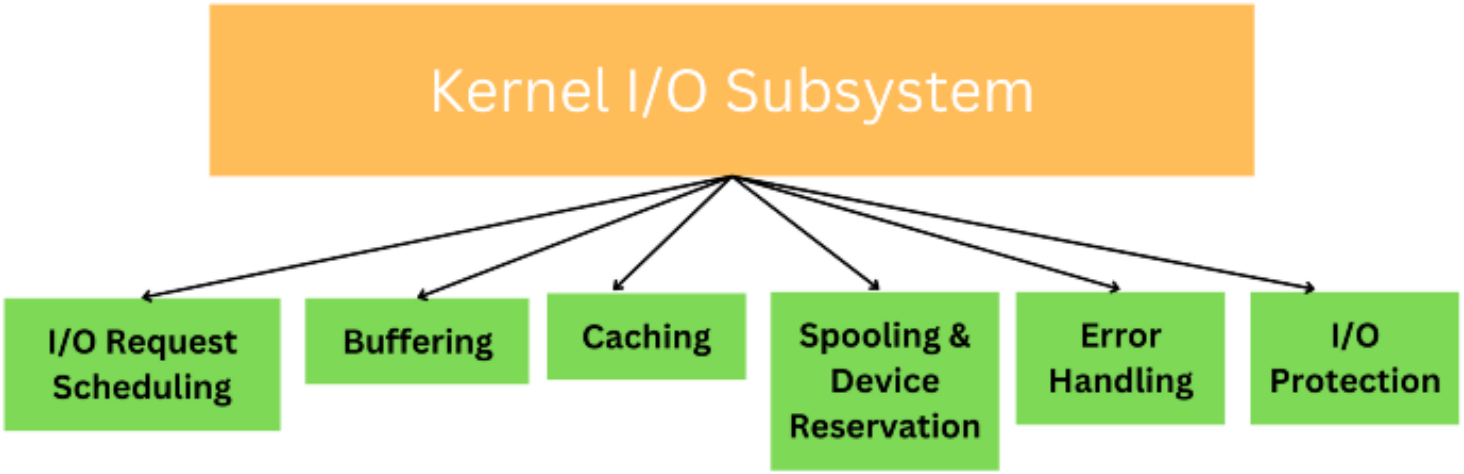
Characteristics of I/O devices

- Devices vary in many dimensions:
  - Sharing:
    - A **sharable** device can be used concurrently by several processes (hard drives).
    - A **dedicated** device cannot (keyboard, mouse).
  - Device speed:
    - Device speeds range from a few bytes per second to a few giga bytes per second.
    - Slow (keyboard, mouse), Fast (SSD)
  - I/O direction:
    - Read-write, Read only, or Write only.
    - Some devices perform both input and output, but others support only one data transfer direction.

Characteristics of I/O devices

aspect	variation	example
data-transfer mode	character block	terminal disk
access method	sequential random	modem CD-ROM
transfer schedule	synchronous asynchronous	tape keyboard
sharing	dedicated sharable	tape keyboard
device speed	latency seek time transfer rate delay between operations	
I/O direction	read only write only read–write	CD-ROM graphics controller disk

Kernel I/O Subsystem



kernel's I/O subsystem provide **services** to efficiently manage I/O operations:

Kernel I/O Subsystem

- Scheduling: Manages the **order and timing** of I/O requests to optimize system performance.
- Buffering: Temporarily holds data to accommodate **speed differences** between devices (e.g., keyboard input vs. disk storage).
- Caching: Stores frequently **accessed data in fast memory** to speed up retrieval.
- Spooling: Queues data for devices that process **one task at a time** (e.g., printer spooling).
- Device Reservation: Ensures **exclusive access** to devices when necessary to prevent conflicts.
- Error Handling: Detects, reports, and manages I/O errors for system stability.

Buffering in File Reading

- What is Buffering?
  - The process of **temporarily storing data** in memory to reduce direct disk access and improve performance.
  - It allows efficient reading and writing operations by **handling data in chunks instead of individual bytes**.
- Example: Buffered File Reading
  - Reading a large text file efficiently without excessive disk access.
- How It Works:
  1. OS **reads a block of data** (e.g., 4KB or 8KB) from the disk into a buffer in memory.
  2. The program reads data from this buffer instead of making frequent disk requests.
  3. When the **buffer is exhausted**, the OS fetches the next block from the disk.

Caching in the I/O Subsystem

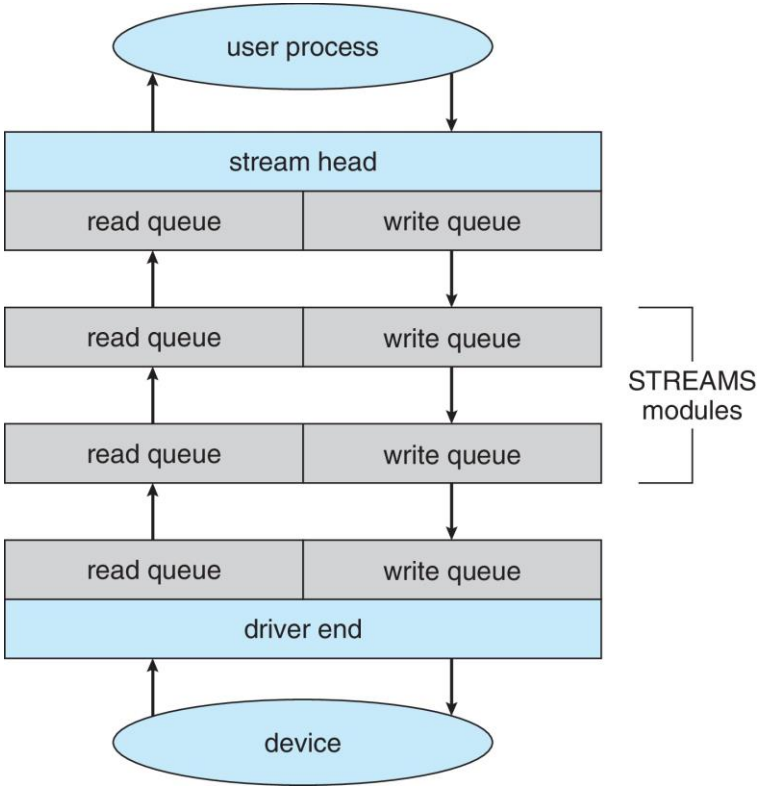
- A cache is **a high-speed memory** that stores frequently accessed data or instructions.
- It helps in **reducing access time** and improving system performance.
- Examples**:**
  - Web Caching: OS and web browsers cache web pages and resources to minimize network traffic and load times.
    - **Example**: When visiting a website for the second time, cached data speeds up the page load.
  - Video Caching: Streaming platforms preload parts of a video to avoid buffering.
    - **Example:** YouTube loads the next few seconds of a video in advance for smooth playback.

Caching in the I/O Subsystem

- Impact on Performance:
  - Reduces **disk I/O latency** by fetching data from memory instead of slow disks.
  - Improves **system responsiveness** for applications relying on frequent I/O operations.

Introduction to STREAMS

- **A stream is a full-duplex connection between a device driver and a user-level process.**
- Implemented in Unix System V and beyond.
- It consists of :
  - **Stream head:** interfaces with the user process.
  - **Driver end:** controls the device.
  - Zero or more **stream modules** between the stream head and the driver end.



How STREAMS Work

- STREAMS use **message-based communication.**
- **Data transferred between queues as messages.**
- Message Flow in STREAMS: Each module in a STREAM has two **queues**:
  - **Read Queue** → Handles incoming data (device → user).
  - **Write Queue** → Handles outgoing data (user → device).
- Messages are passed through these queues **asynchronously.**

The End