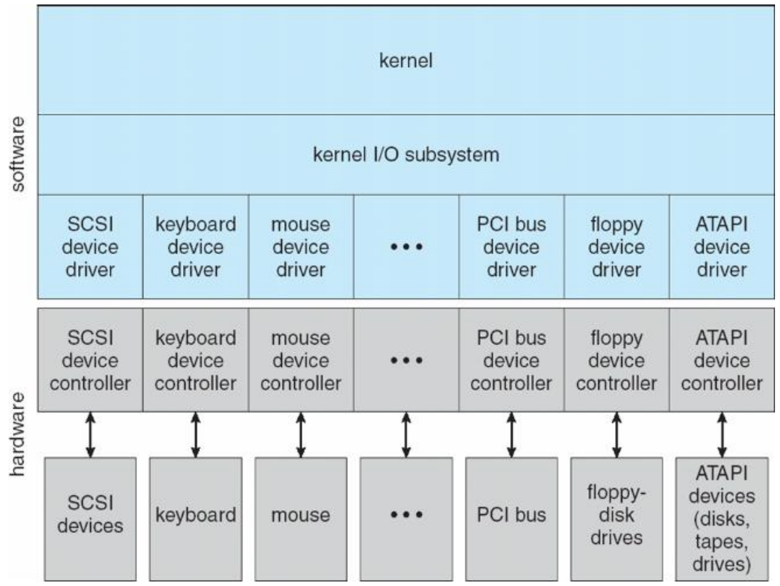
L42- IO DEVICES PERFORMANCE

Objectives

- Explore the structure of an operating system’s I/O subsystem
- Discuss the principles of I/O hardware and its complexity
- Provide details of the performance aspects of various types of I/O devices

Application I/O Interface

- I/O system calls encapsulate device behaviors in generic classes
- Device-driver layer hides differences among I/O controllers from kernel
- New devices use already-implemented protocols. No extra work.
- Each OS has its own I/O subsystem structures and device driver frameworks



Kernel I/O Structure

Category of I/O Devices

- Devices vary in many dimensions
- **Character-stream** or **block**
- **Sequential** or **random-access**
- **Synchronous** or **asynchronous**
- **Sharable** or **dedicated**
- **Speed of operation**
- **read-write**, **read only**, or **write only**

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

Characteristics of I/O Devices

- Broadly I/O devices can be grouped by the OS into
  - Block I/O
  - Character I/O (Stream)
  - Memory-mapped file access
  - Network sockets

Block and Character Devices

- Block devices include disk drives
  - Commands include read, write, seek
  - Raw I/O, direct I/O, or file-system access
  - File mapped to virtual memory and brought via demand paging
  - DMA
- Character devices include keyboards, mice, serial ports
  - Commands include **get()**, **put()**
  - Buffering and editing services

Network Devices

- Linux, Unix, Windows and many others include **socket** interface
  - Separates network protocol from network operation
  - Includes **select()** functionality to choose sockets for send and receive
- Pipes, FIFOs, streams, queues, mailboxes

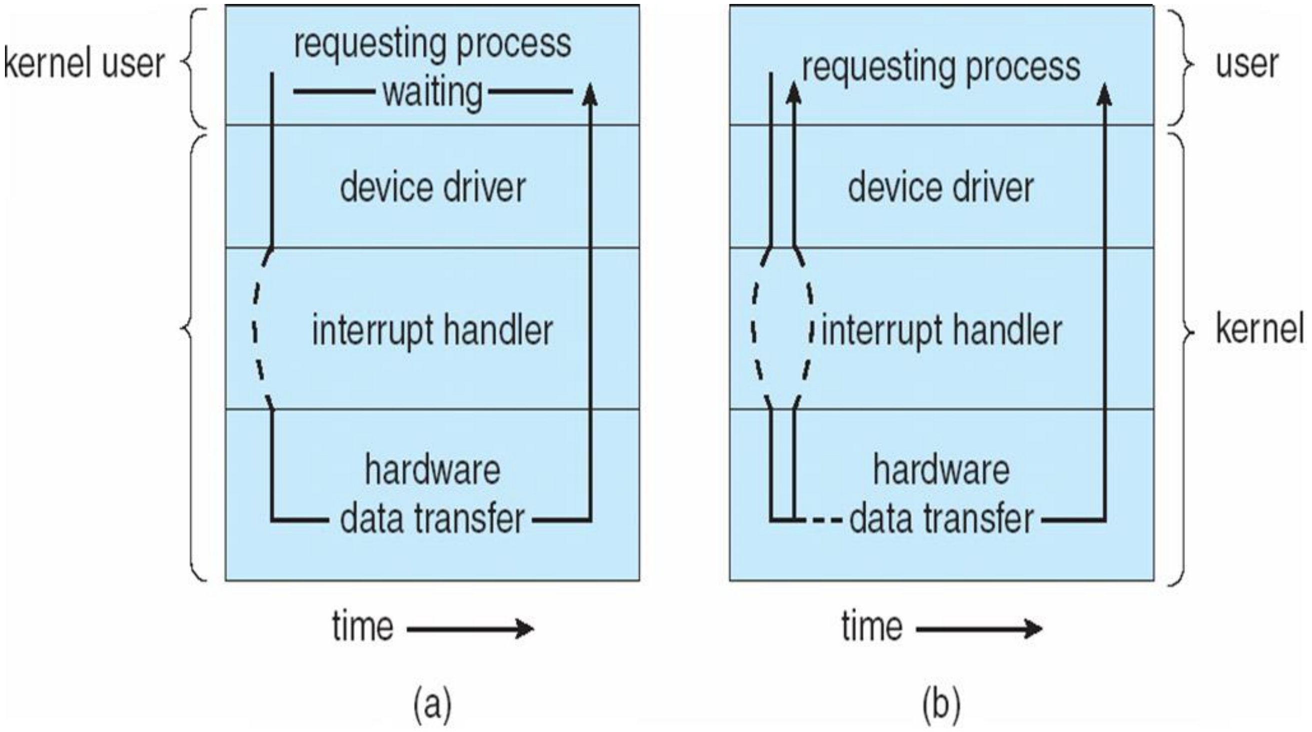
Clocks and Timers

- Provide current time, elapsed time, set a timer to trigger ***X*** and ***T***
- Normal resolution about 1/60 second
- Some systems provide higher-resolution timers
- Programmable interval timer used for timings, periodic interrupts

Nonblocking and Asynchronous I/O

- **Blocking** - process suspended until I/O completed
  - Application from running to waiting and to ready states
- **Nonblocking** - I/O call returns as much as available I/O data
  - User interface, data copy (buffered I/O)
  - Implemented via multi-threading
  - Returns quickly with count of bytes read or written
  - **select()** to find if data ready then **read()** or **write()** to transfer
- **Asynchronous** - process runs while I/O executes
  - Difficult to use
  - I/O subsystem signals process when I/O completed

Two I/O Methods



Synchronous   Asynchronous

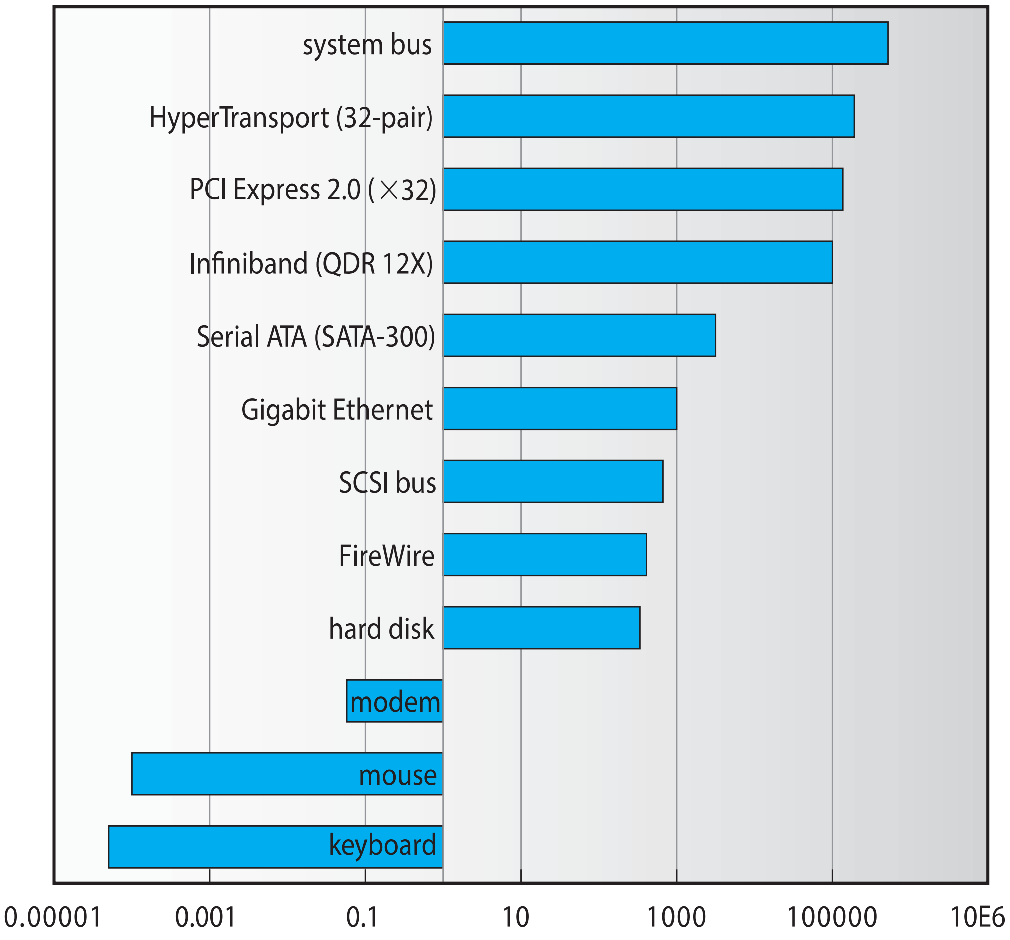
Kernel I/O Subsystem

- **Scheduling**
  - Some I/O request ordering via per-device queue
  - Some OSs try fairness
  - Some implement Quality Of Service

Kernel I/O Subsystem

- **Buffering** - store data in memory while transferring between devices
  - To cope with device speed mismatch
  - To cope with device transfer size mismatch
  - To maintain copy semantics
- **Double buffering** – two copies of the data
  - Kernel and user
  - Varying sizes
  - Full / being processed and not-full / being used
  - Copy-on-write can be used for efficiency in some cases

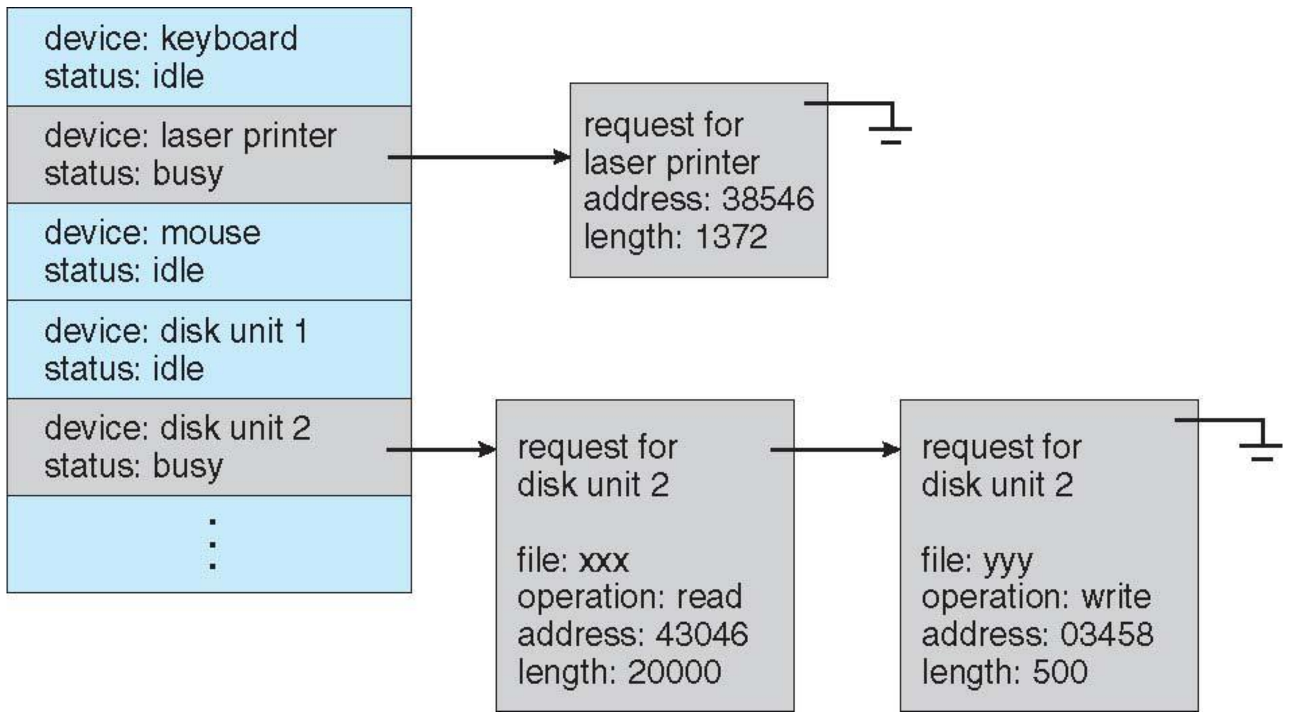
Sun Enterprise 6000 Device-Transfer Rates



Kernel I/O Subsystem

- **Caching** - faster device holding copy of data
  - Always just a copy
  - Key to performance
  - Sometimes combined with buffering
- **Spooling** - hold output for a device
  - If device can serve only one request at a time
  - i.e., Printing
- **Device reservation** - provides exclusive access to a device
  - System calls for allocation and de-allocation
  - Watch out for deadlock

Device-Status Table

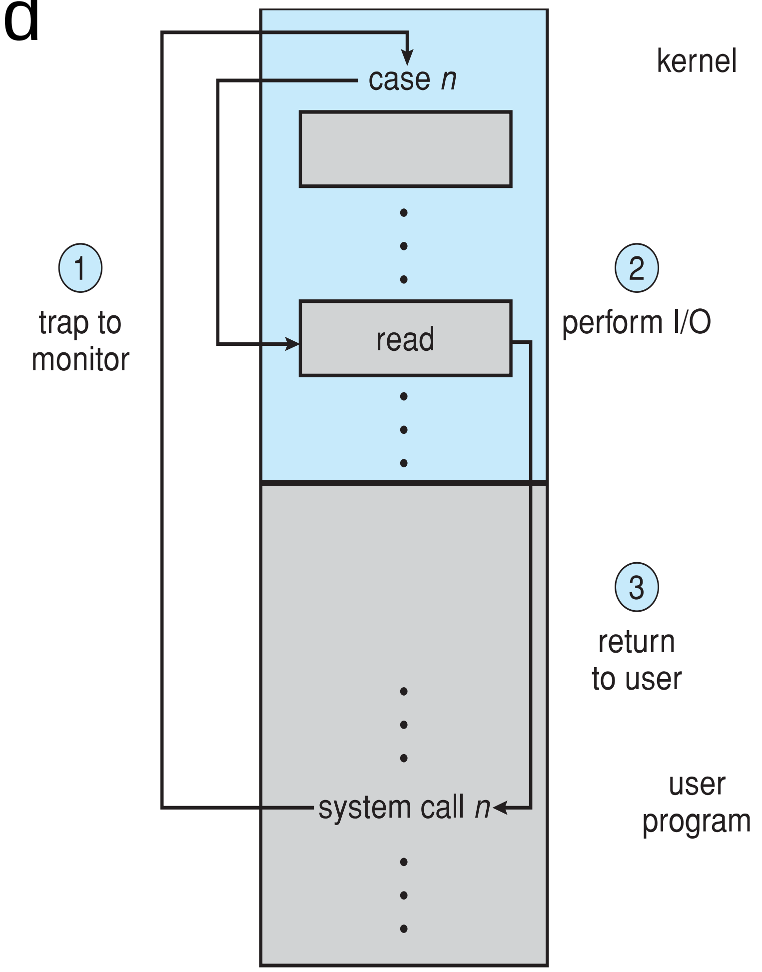


Error Handling

- OS can recover from disk read, device unavailable, transient write failures
  - Retry a read or write, for example
  - Some systems more advanced – Solaris FMA, AIX
    - Track error frequencies, stop using device with increasing frequency of retry-able errors
- Most return an error number or code when I/O request fails
- System error logs hold problem reports

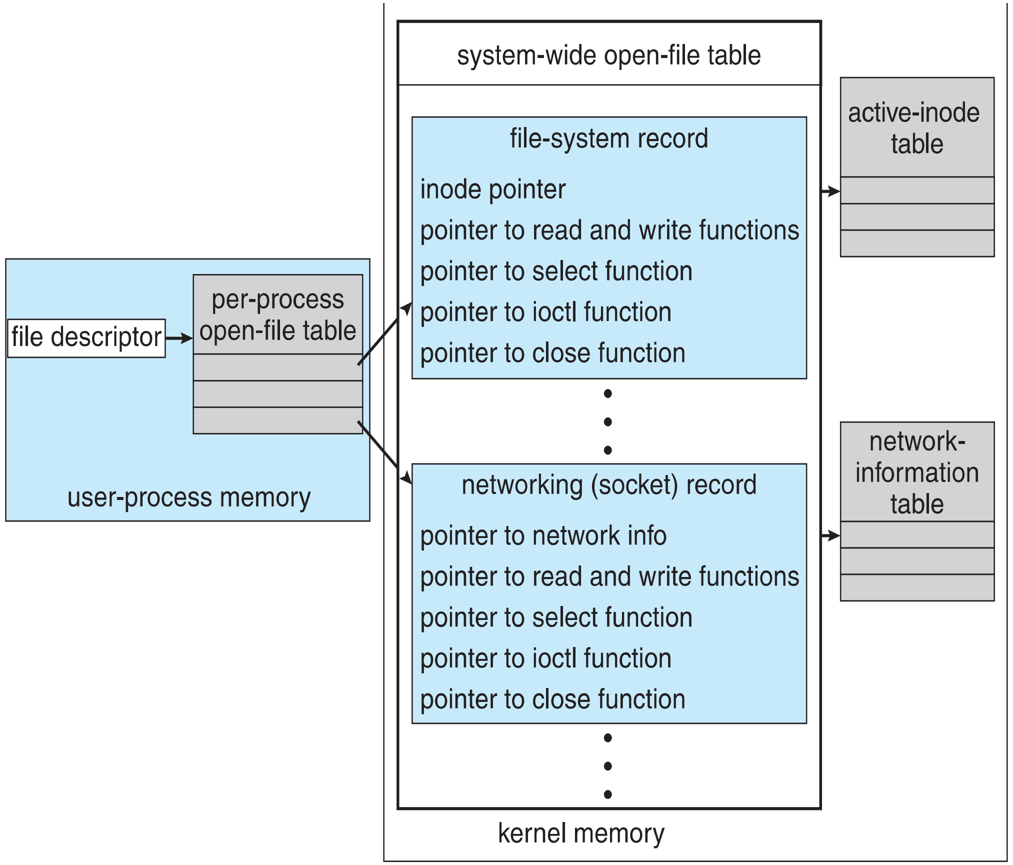
System Call to Perform I/O

- All I/O instructions defined to be privileged
- I/O must be performed via system calls



Kernel Data Structures

- Kernel keeps state information for I/O components, including open file tables, network connections, character device state



Power Management

- Not strictly domain of I/O, but much is I/O related
- Computers and devices use electricity, generate heat, frequently require cooling
- OS can help manage and improve use
- Mobile computing has power management

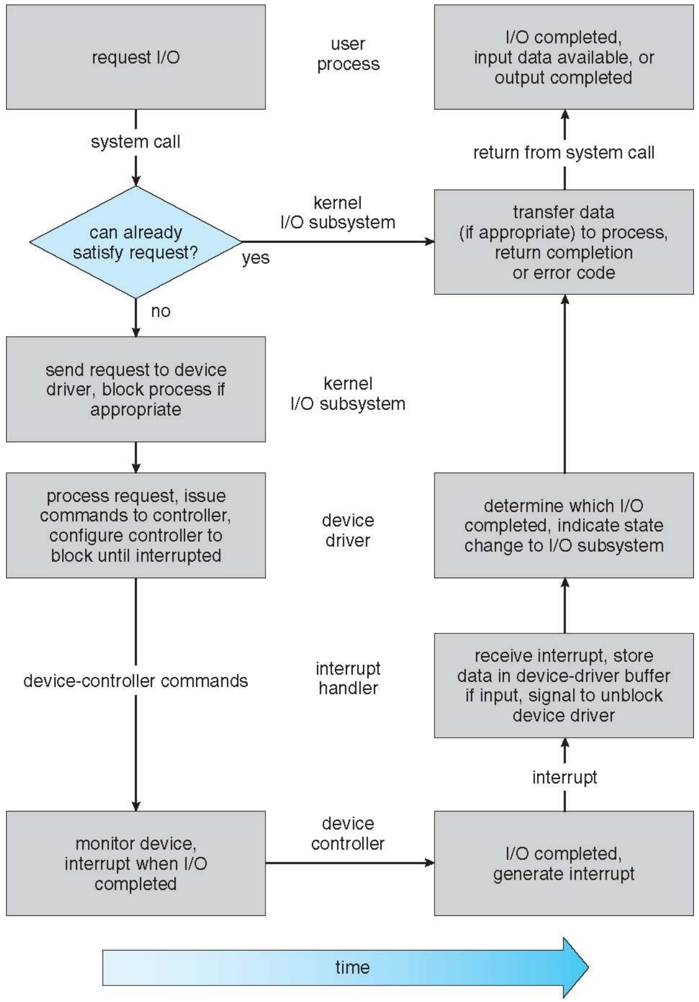
Power Management on Mobile Platforms

- ❖ Understands relationship between components
- ❖ Build device tree representing physical device topology
- ❖ System bus -> I/O subsystem -> {flash, USB storage}
- ❖ Device driver tracks state of device, whether in use
- ❖ Unused component – turn it off
- ❖ All devices in tree branch unused – turn off branch
- ❖ Wake locks – like other locks but prevent sleep of device when lock is held
- ❖ Power collapse – put a device into very deep sleep
    - ❖ Only awake enough to respond to external stimuli

I/O Requests to Hardware Operations

- Consider reading a file from disk for a process:
  - Determine device holding file
  - Translate name to device representation
  - Physically read data from disk into buffer
  - Make data available to requesting process
  - Return control to process

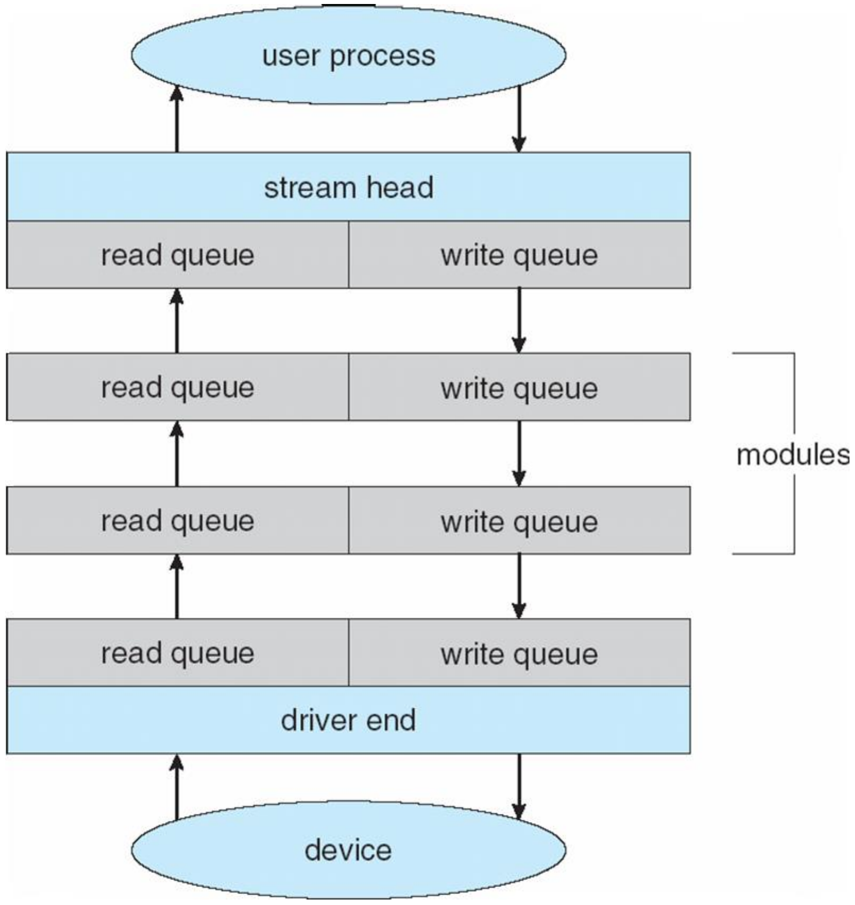
Life Cycle of An I/O Request



STREAMS

- **STREAM** – a full-duplex communication channel between a user-level process and a device in Unix System
- A STREAM consists of:
  - STREAM head interfaces with the user process
  - driver end interfaces with the device
  - zero or more STREAM modules between them
- Each module contains a **read queue** and a **write queue**
- Message passing is used to communicate between queues
  - **Flow control** option to indicate available or busy
- Asynchronous internally, synchronous where user process communicates with stream head

The STREAMS Structure



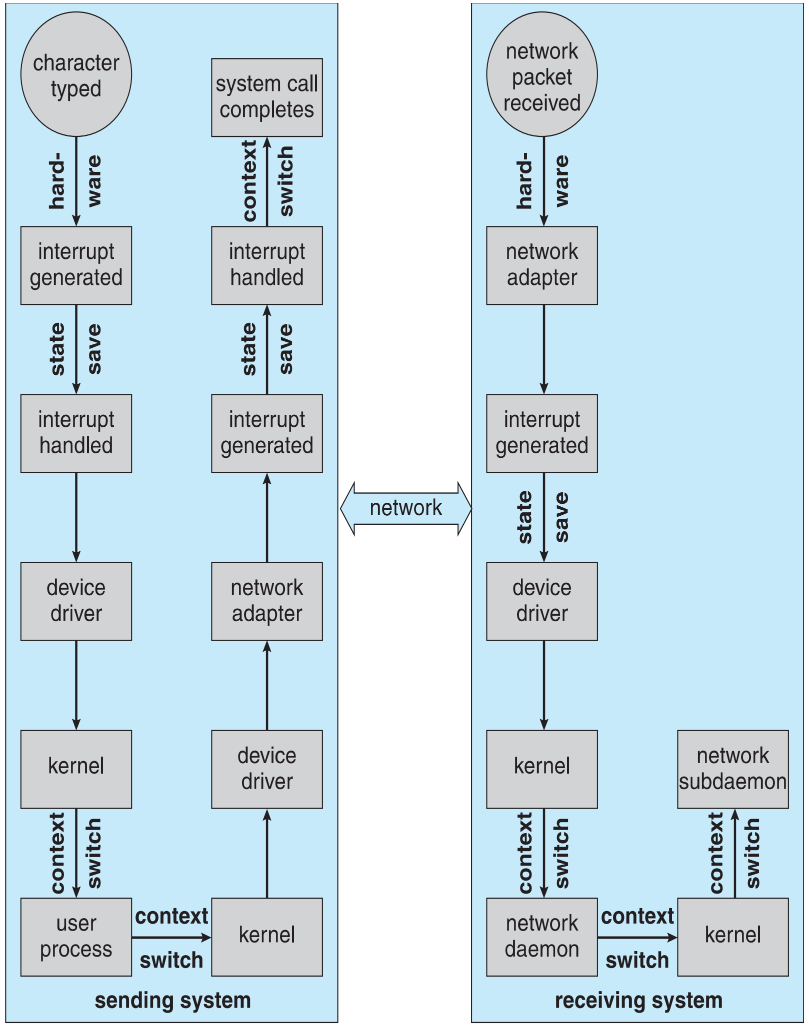
Performance

- I/O a major factor in system performance:
  - Demands CPU to execute device driver, kernel I/O code
  - Context switches due to interrupts
  - Data copying
  - Network traffic especially stressful

Improving Performance

- Reduce number of context switches
- Reduce data copying
- Reduce interrupts by using large transfers, smart controllers, polling
- Use DMA
- Use smarter hardware devices
- Balance CPU, memory, bus, and I/O performance for highest throughput
- Move user-mode processes / daemons to kernel threads

Intercomputer Communications



Thank You