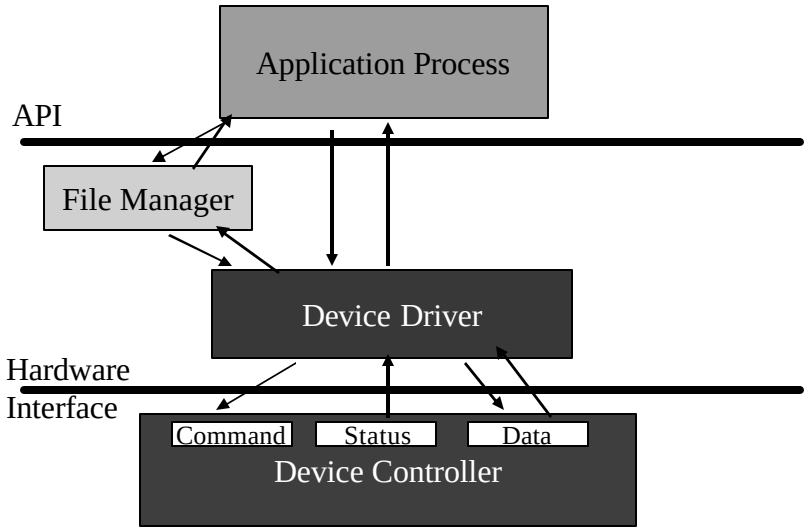
# Operating Systems

Lecture 12 : Input-Output Systems

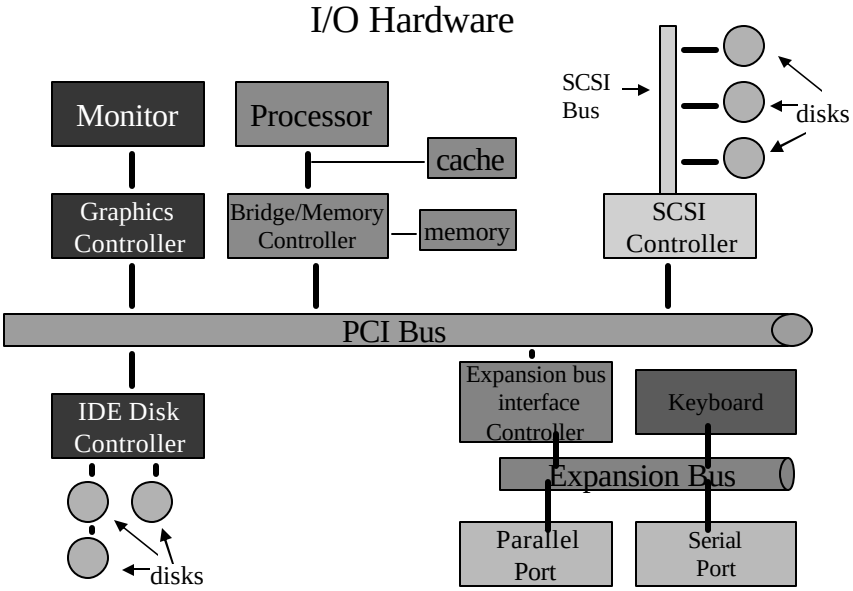
## I/O and Operating Systems

- Controlling input/output devices is complicated by the wide variety of devices.
- The variety of different methods used to control these devices comprise the I/O subsystem.
- This wide variety of devices conflict the push toward the standardization of interfaces, leading to the use of device driver modules.

# Device Management Organization




# I/O Hardware



## Device Controllers : An Example

The NEC PD 765 disk controller has a 16-command language, using 1 to 9 bytes in a device register. Commands include:

- Reading and writing data
- Moving the disk arm
- Formatting tracks
- Initializing, calibrating, sensing and resetting the controller and its drives.

## Using A Device Controller

READ and WRITE commands require parameters such as disk block address, # of sectors per track, recording mode, intersector gap spacing, etc.

The PD 765 returns 23 status and error fields in 7 bytes.

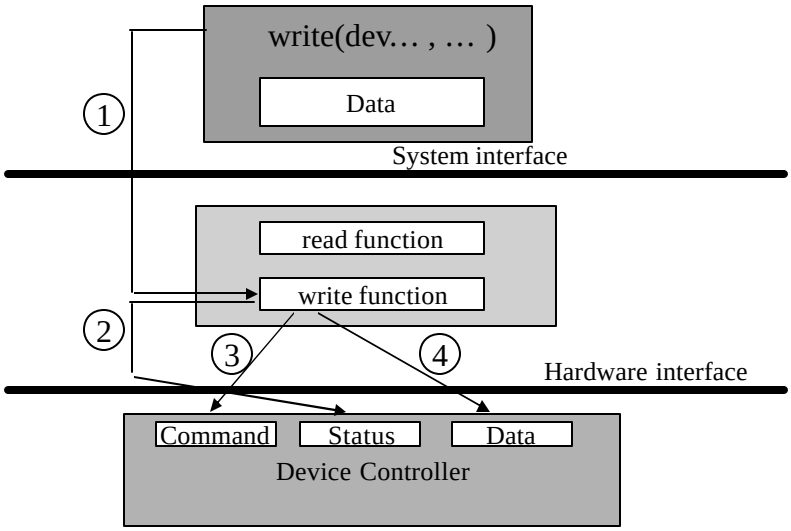
# I/O Port Addresses

| I/O Address Range | Device |
|---|---|
| 000-00F | DMA Controller |
| 020-021 | Interrupt Controller |
| 040-043 | Timer |
| 200-20F | Game Controller |
| 2F8-2FF | Secondary Serial Port |
| 320-32F | Hard Disk Controller |
| 378-37F | Parallel Port |
| 3D0-3DF | Graphics Controller |
| 3F0-3F7 | Floppy Disk Controller |
| 3F8-3FF | Primary Serial Port |

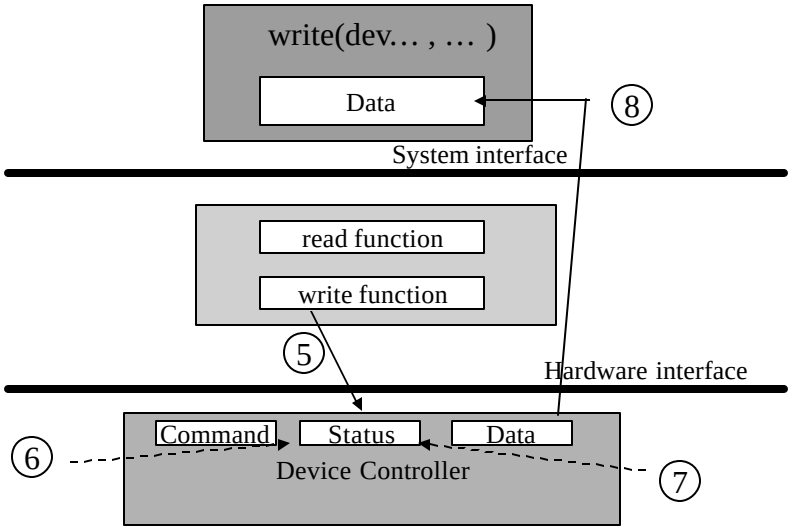
# Typical I/O Port

- The typical I/O port consists of 4 registers:
  - Status – contains bits read by the host system.
  - Control – contains bits written by the host system
  - Data-in – read by the host as input
  - Data-out – written by the host as output

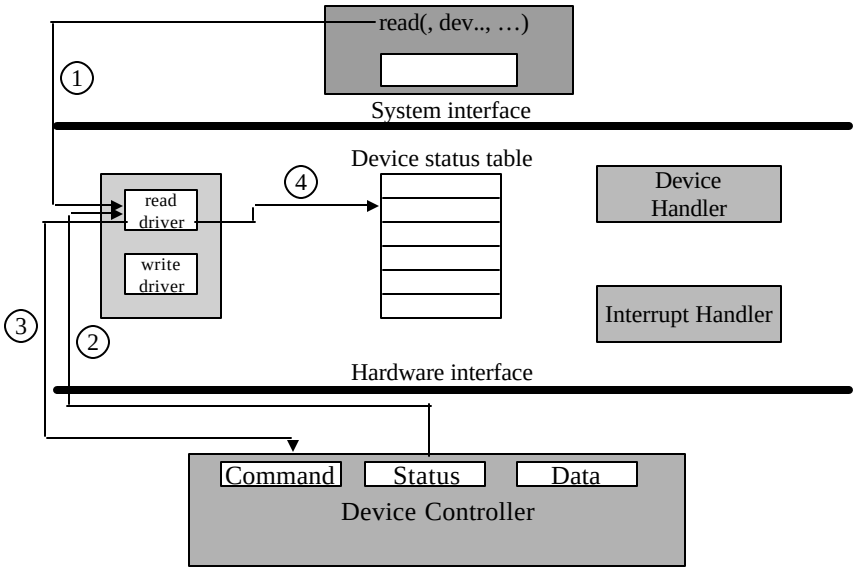
# Using Polling To Start An I/O Operation



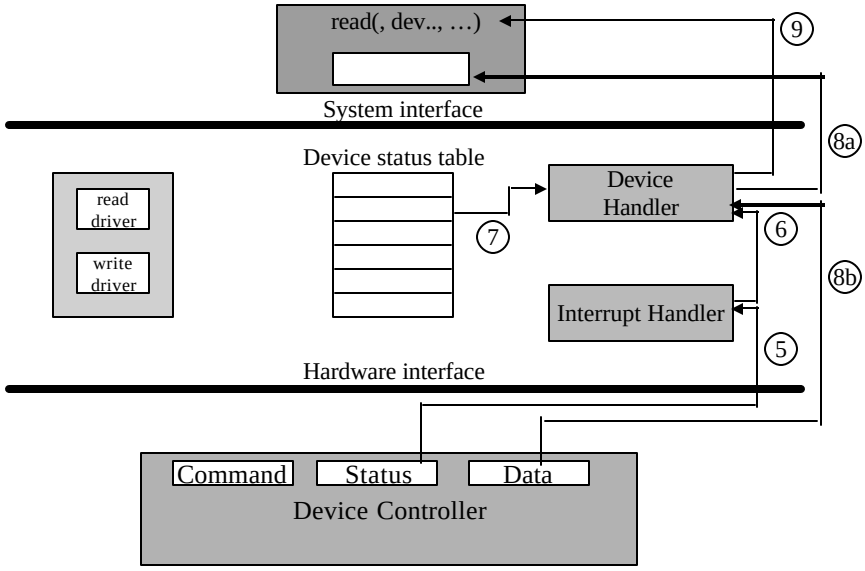

# I/O Performed By Polling



# Using Interrupts To Start An I/O Operation



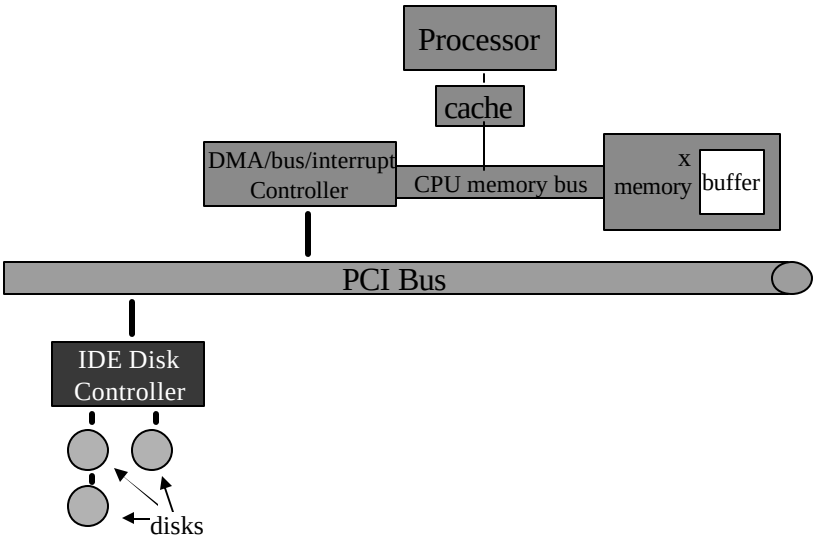

# Completing Interrupt-Driven I/O



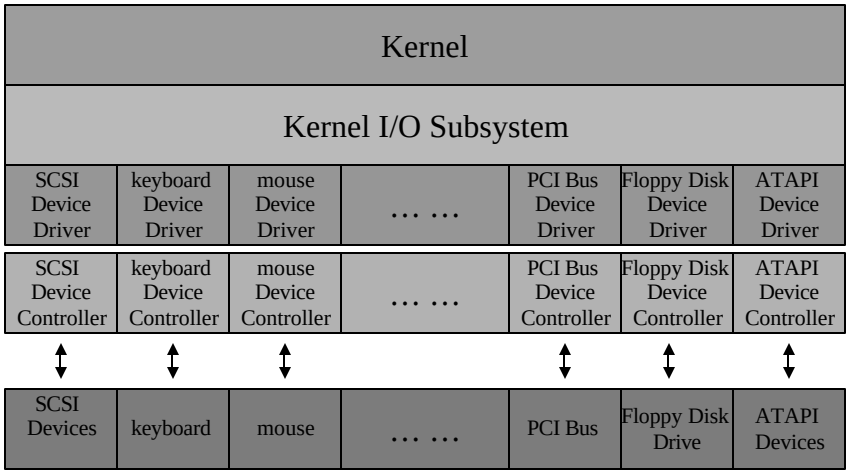
# Maskable vs. NonMaskable Interrupts

| Vector Number | Description |
|---|---|
| 0 | divide error |
| 1 | debug exception |
| 2 | null interrupt |
| 3 | breakpoint |
| 4 | INTO-detected overflow |
| 5 | bound range exception |
| 6 | invalid opcode |
| 7 | device not available |
| 8 | Double fault |
| 15, 19-31 | Intel reserved |
| 32-255 | Maskable interrupts |

# Direct Memory Access



# Application I/O Interface




# I/O Device Characteristics

| **Aspect** | **Variation** | **Example** |
|---|---|---|
| Data-Transfer Mode | Character<br>Block | Terminal<br>Disk |
| Access Method | Sequential<br>Random | Modem<br>CD-ROM |
| Transfer Schedule | Synchronous<br>Asynchronous | Tape<br>Keyboard |
| Sharing | Dedicated<br>Sharable | Tape<br>Keyboard |
| I/O Direction | Read only<br>Write Only<br>Read-Write | CD-ROM<br>Graphics controller<br>Disk |

## Block and Character Devices

- The essential operations of block devices are ***read***, ***write*** and ***seek***.
- ***Raw I/O*** – accessing a block devices as an array of blocks
- Character-stream devices such as keyboard use the basic operations ***get*** and ***put***.

## Network Devices

- Sockets are a network interface used by many operating systems.
- Sockets operations includes:
  - creating a socket
  - connecting local socket to a remote address
  - listening for remote application to connect into local sockets
  - sending and receiving packets

## Clocks and Timers

- Computer clocks and timers have 3 basic operations:
  - Giving the current time
  - Giving the elapsed time
  - Triggering operation *X* at time *T*

## Blocking and Nonblocking I/O

- When an application calls for input or output, the operating system blocks the process until the I/O operation is completed. Such a system call is called a ***blocking system call***.
- Some application cannot work properly if system calls block the process, e.g., video applications. These use ***nonblocking system calls***.

## Implementing Nonblocking I/O

- Nonblocking I/O requires that execution and I/O occur concurrently. This can be implemented by writing ***multithreaded programs***.
- An alternative involves nonblocking system calls which do not wait for I/O to be completed.

## Kernel I/O Subsystem

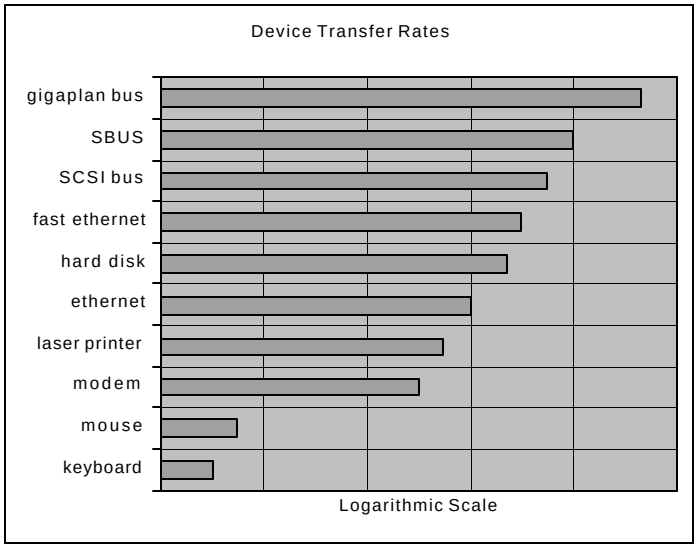
The I/O subsystem provides these services:

- I/O Scheduling
- Buffering
- Caching
- Spooling
- Device Reservation
- Error handling

# I/O Scheduling

- The order in which input/output requests are made is rarely the best order in which to perform them.
- I/O request scheduling algorithms seek to maximize performance while guaranteeing that all input/output are fulfilled within a reasonable time frame.

# Buffering



## Caching

- A cache is a copy of input/output data in memory that can be accessed more quickly than what is stored externally.
- Although buffering and caching are distinct functions, there are cases where the same area of memory is used for both purposes, e.g., when reading disk data.

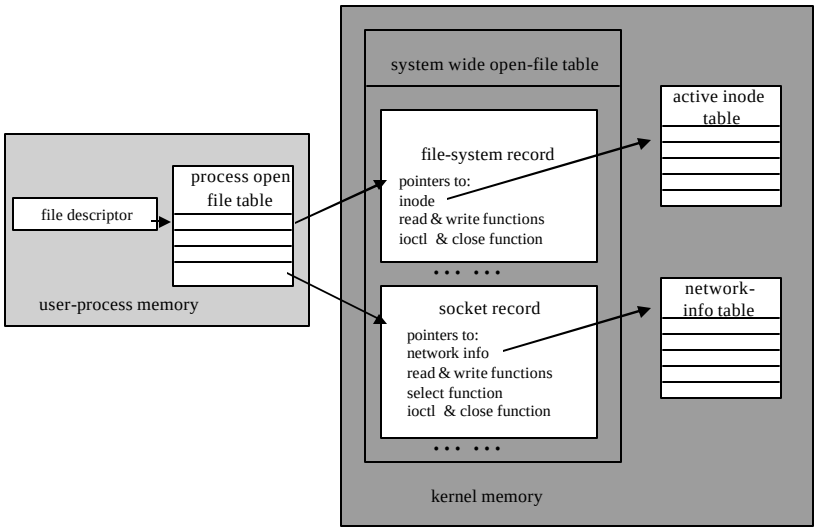
## Spooling and Device Reservation

- A spool is a buffer for output intended for a device that cannot accept interleaved data streams.
- Printers are an example of such devices.

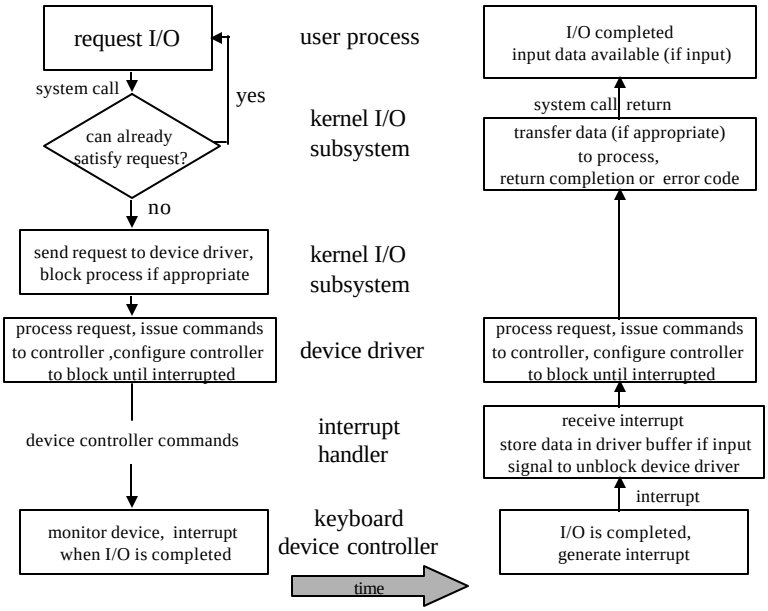
# Error Handling

- Operating systems should be able to handle transient hardware errors. This is usually done by trying the failed operation again.
- I/O system calls return a bit of information that indicates whether the operation was a success.
- In some cases either the operating system or the hardware itself may provide more specific information about a failure.

# Kernel Data Structures



## Transforming I/O Requests to Hardware Operations




# Performance

- I/O is a major factor in system performance.
- Interrupt handling is computationally expensive; anything that limits it can increase system performance.
- Network traffic can seriously affect system performance
- Some systems use front-end processor to terminal I/O.