# Part 1

# Introduction

1.1 What is an operating system
1.2 History of operating systems
1.3 Computer hardware review
1.4 The operating system zoo
1.5 Operating system concepts
1.6 System calls
1.7 Operating System Structure

# 1.1 What is an operating system

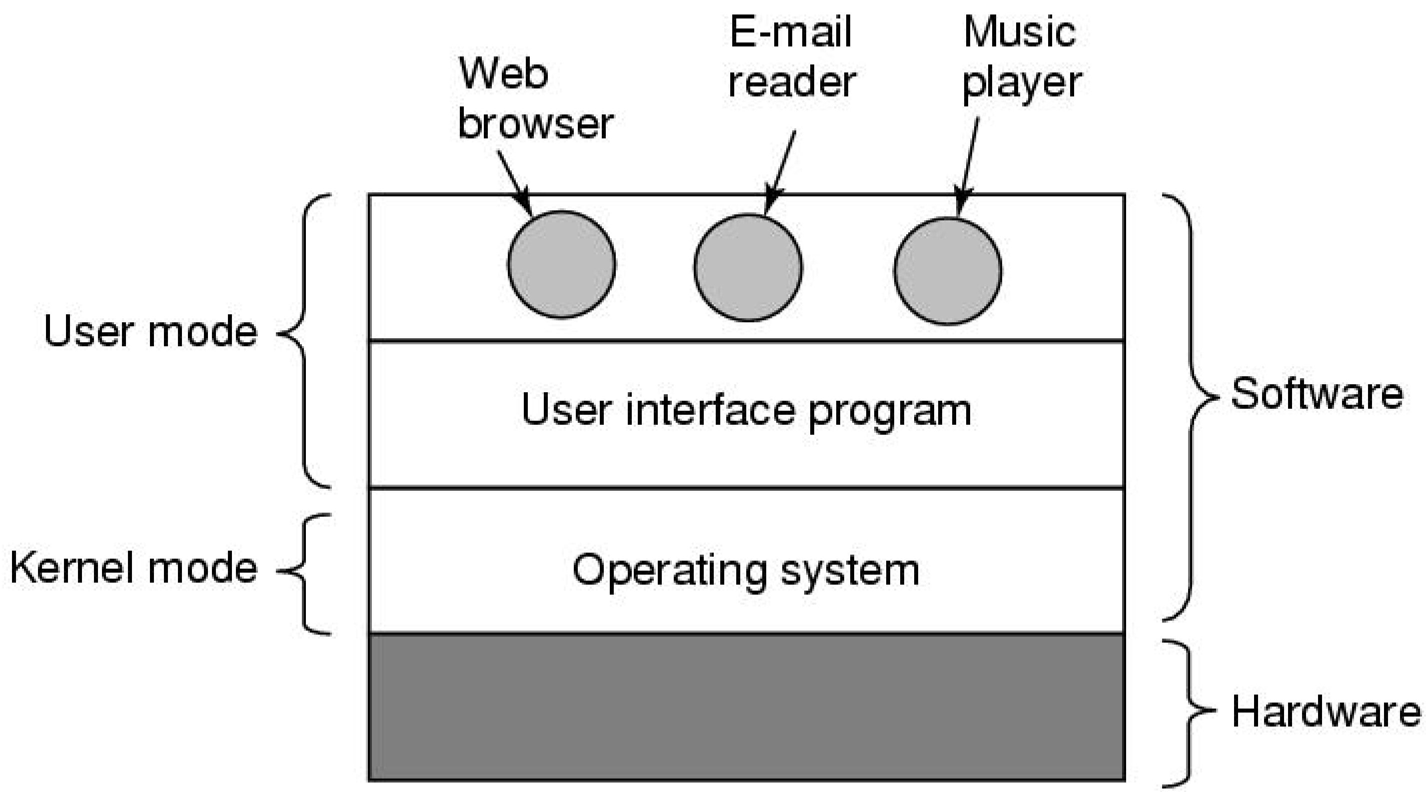
# What is An Operating System

A modern computer consists of:

- One or more processors
- Main memory
- Disks
- Printers
- Various input/output devices

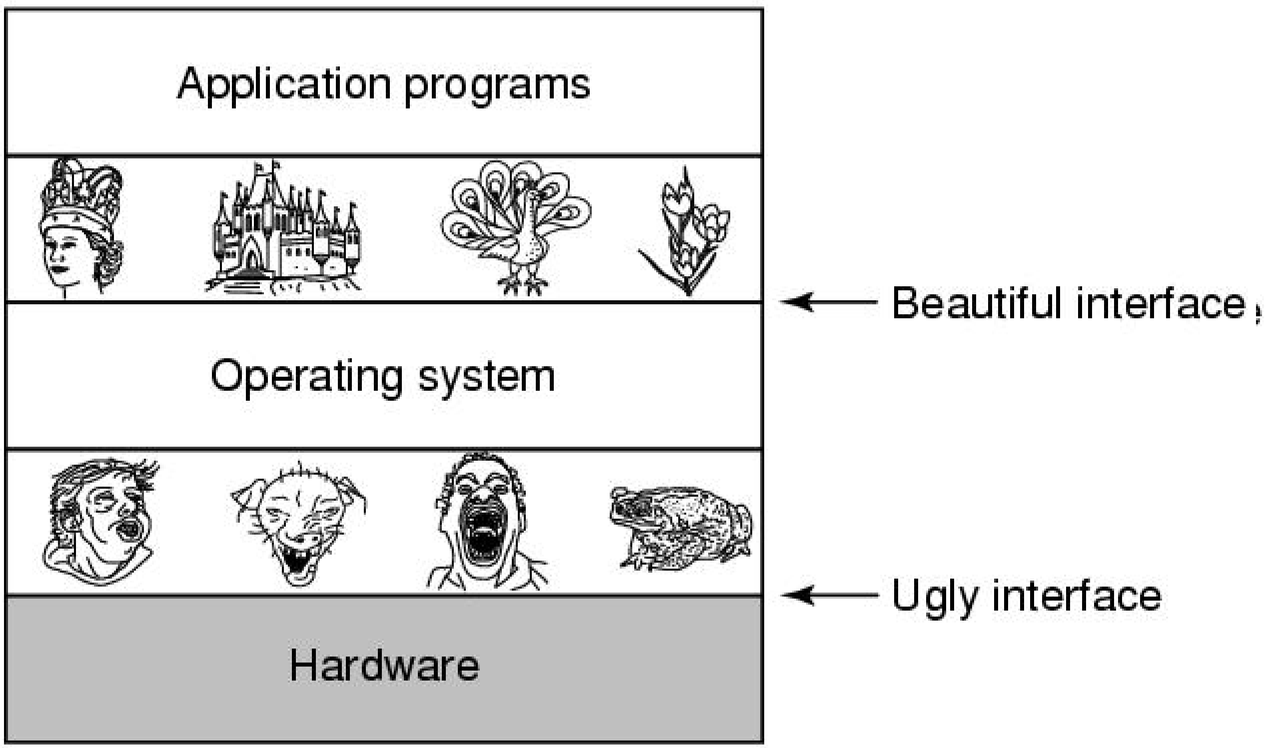
Managing all these components requires a layer of software – the **operating system**

# What is an Operating System



# The Operating System as an Extended Machine

- Hides the messy details which must be performed
- Presents user with a virtual machine, easier to use



# The Operating System as a Resource Manager

- Allow multiple programs to run at the same time
- Manage and protect memory, I/O devices, and other resources
- Includes multiplexing (sharing) resources in two different ways:
  - In time
  - In space

# 1.2 History of Operating Systems

# History of Operating Systems

- First generation 1945 - 1955
  - vacuum tubes, plug boards
- Second generation 1955 - 1965
  - transistors, batch systems
- Third generation  1965 – 1980
  - ICs and multiprogramming
- Fourth generation 1980 – present
  - personal computers

# History of Operating Systems
## First generation 1945 - 1955

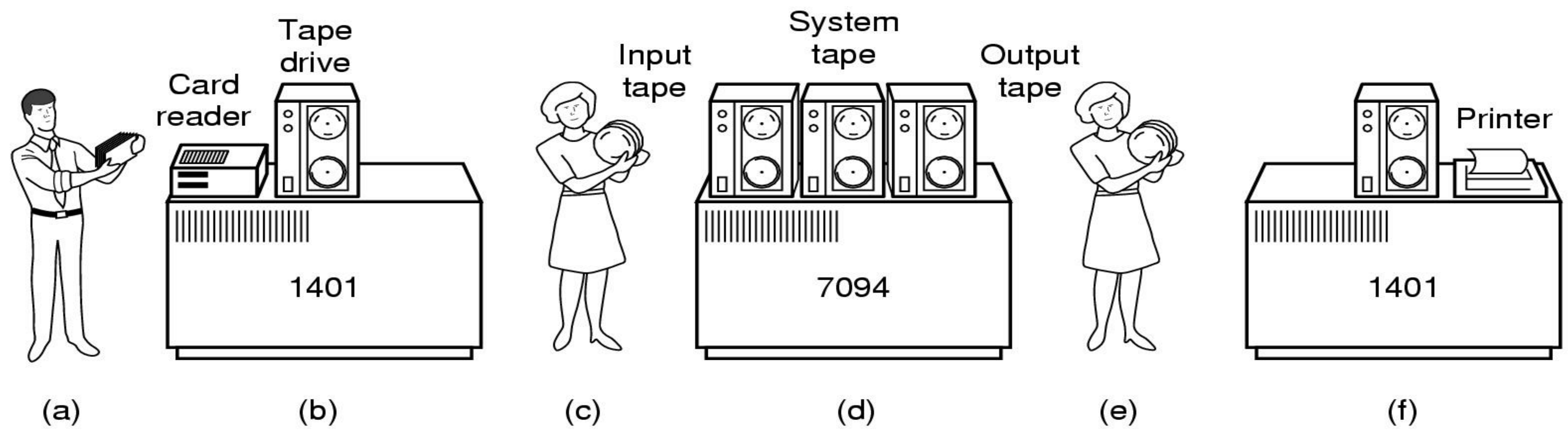
- Computers:ENIAC, UNIVAC…
- Operating System: No OS,
- Machine Language: plugboards
- Single group: designed, built, programmed, operated and maintained each machine

# History of Operating Systems
## Second Generation 1955 – 1965 (1)

- Computers: IBM 1401, IBM 7094…
- Operating System : FMS (Fortran Monitor System), IBSYS for Computer 7094
- Batch Systems
  - Function of Early Batch System
  - Structure of a typical FMS job
- Separation between designers, builders, programmers, operators and maintenance personnel

# History of Operating Systems

## Second Generation 1955 – 1965 (2)

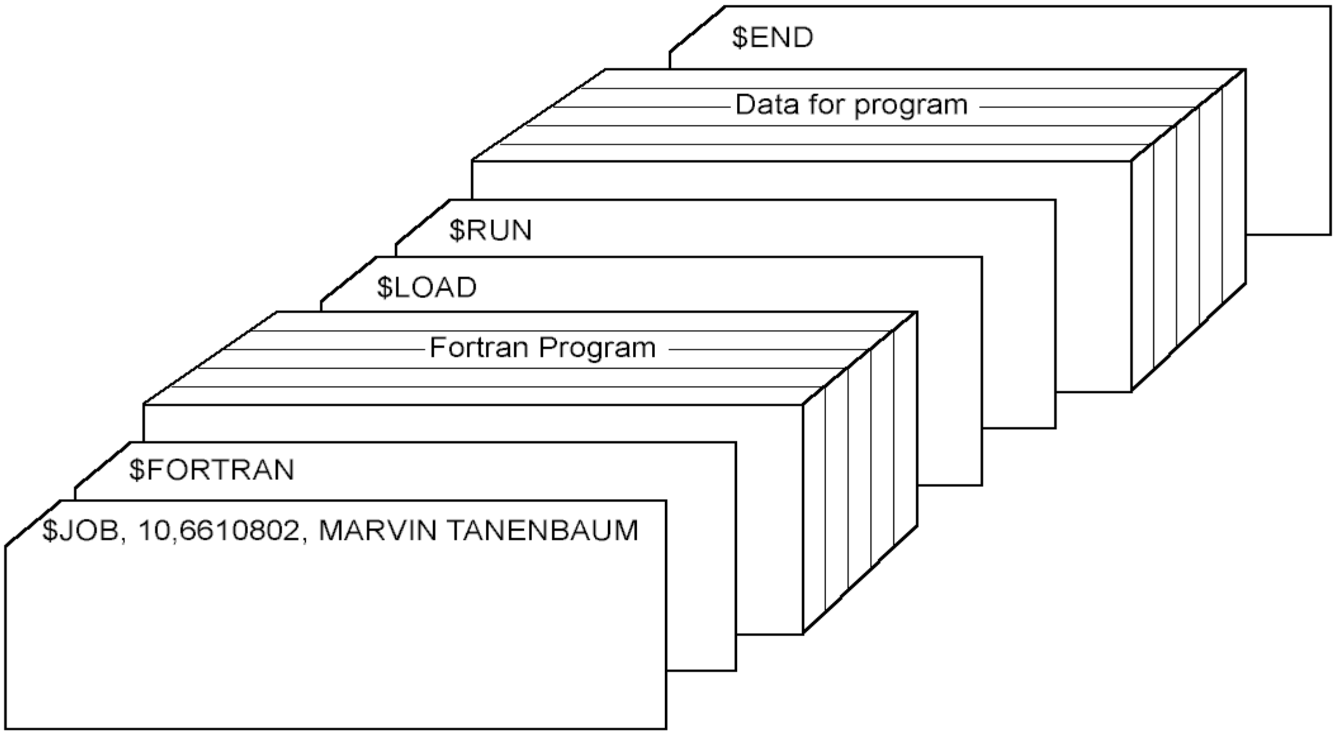



Early batch system

- bring cards to 1401
- read cards to tape
- put tape on 7094 which does computing
- put tape on 1401 which prints output

# History of Operating Systems

## Second Generation 1955 – 1965 (3)

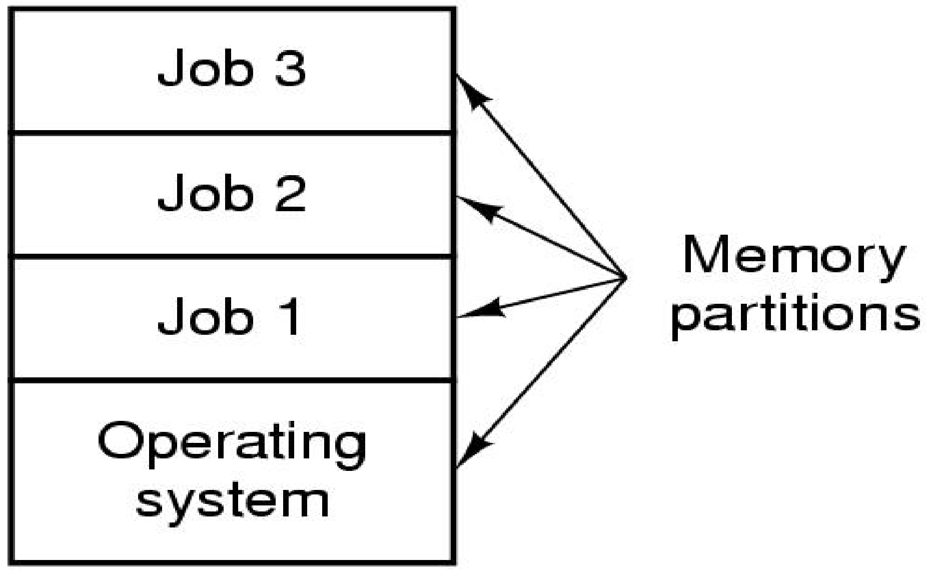



- Structure of a typical FMS job – 2nd generation

# History of Operating Systems
## Third generation 1965 – 1980 (1)

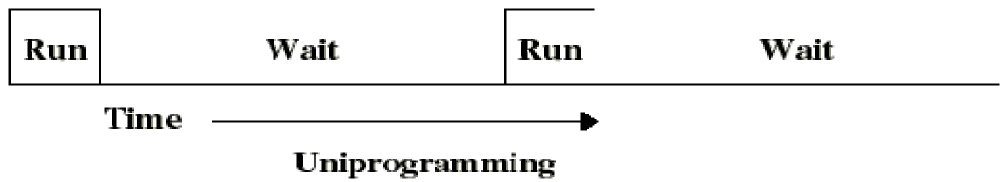
- Computers: System/360, IBM370, IBM4300…
- Operating System:
  - OS/360: to work on all models
  - Multiprogamming
  - Time Sharing:
    - CTSS (Compatible Time Sharing System),
    - MULTICS (MULTiplexed Information and Computing Service),
    - Unix

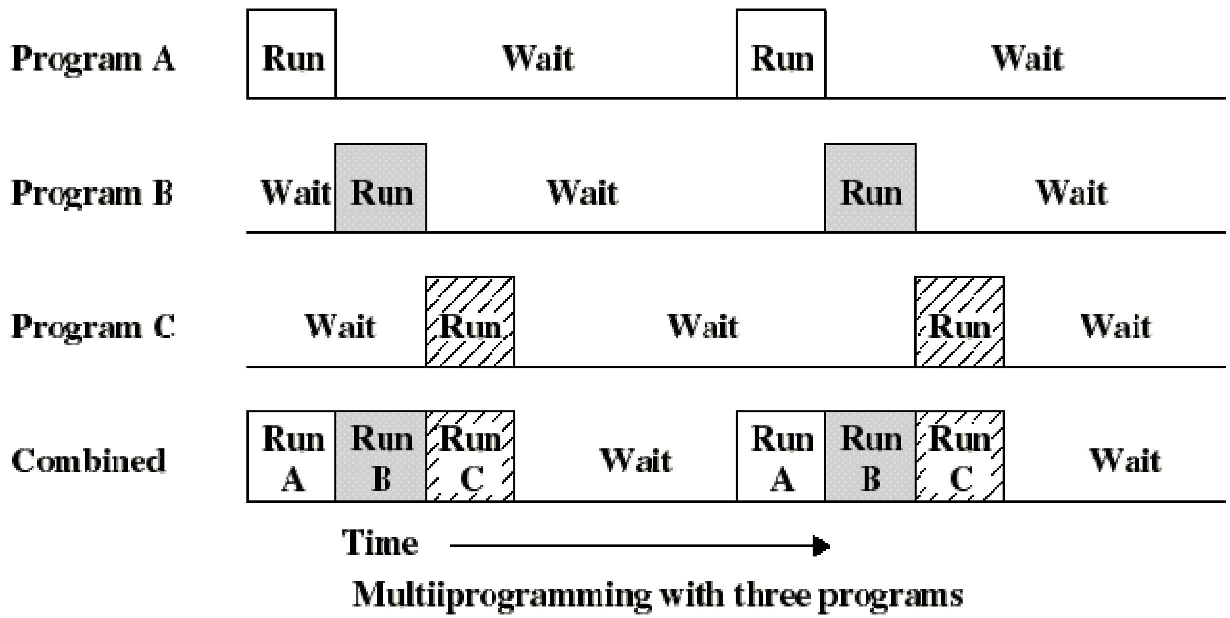
# History of Operating Systems
## Third generation 1965 – 1980 (2)




- Multiprogramming system
  - three jobs in memory – 3rd generation

# History of Operating Systems
## Third generation 1965 – 1980 (3)



# History of Operating Systems
## Fourth generation 1980 – present

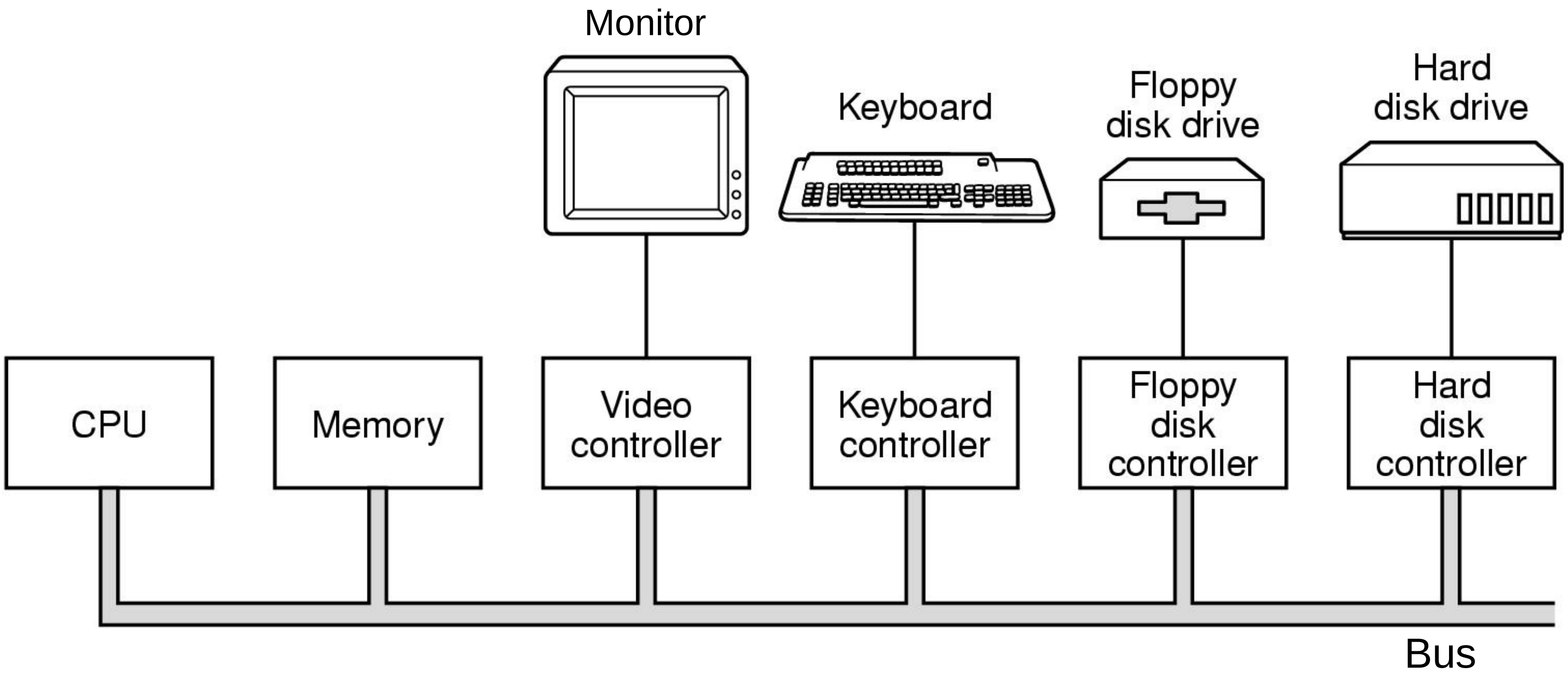
- Computers: IBM PC 80x86, Macintosh…
- Operating System:
  - 1977: CP/M (Control Program for Microcomputer)
  - 1980: DOS (Disk Operating System)
  - GUI with Macintosh
  - 1985-1995: Window 3.x
  - 1995: Window 95
  - 1996: Window NT 4.0
  - 1999: Window 2000
  - Window 2003
  - Unix

# 1.3 Computer Hardware Review

# Computer Hardware Review

- CPU
- Memory
- I/O Devices
- Buses

# Computer Hardware Review

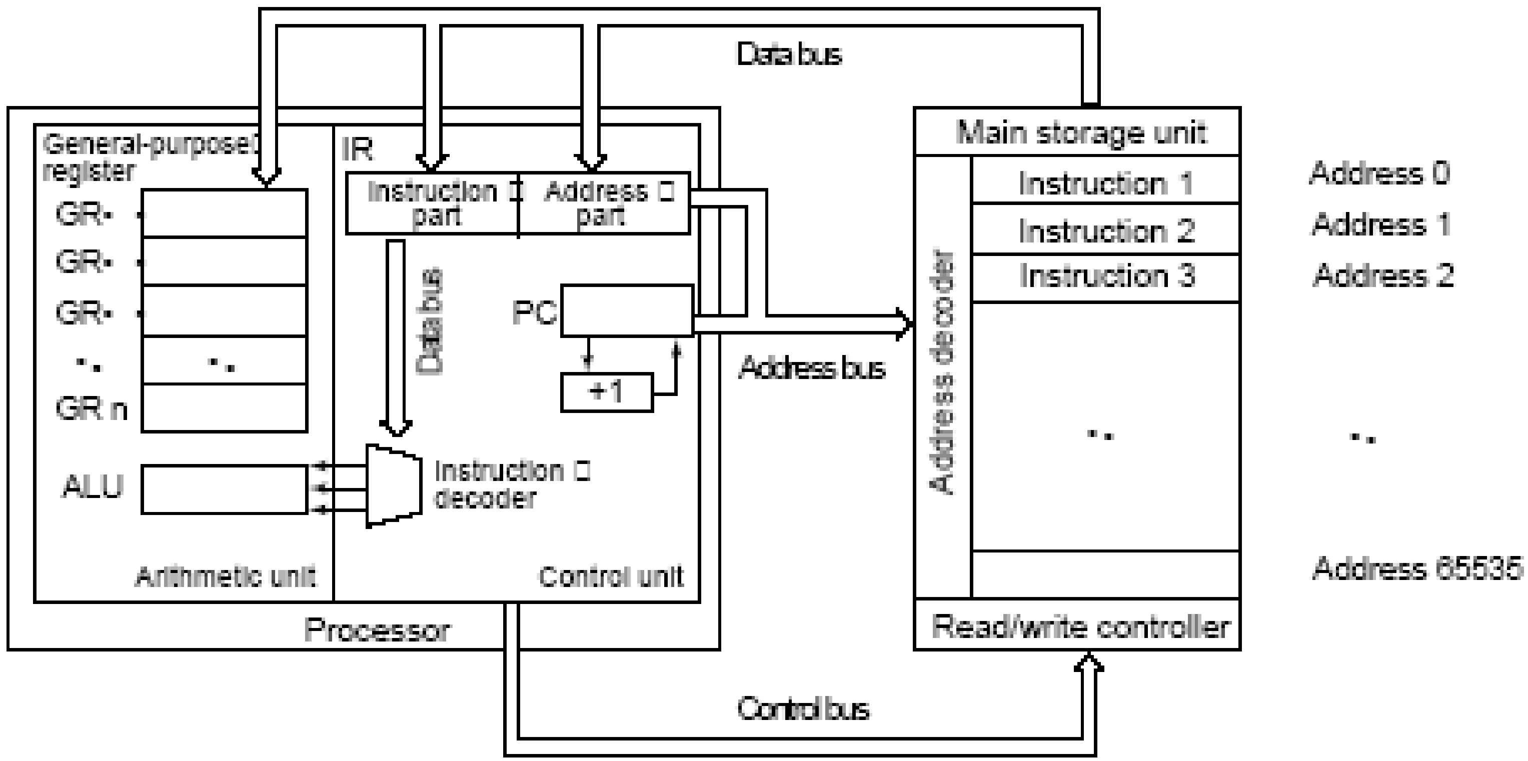



- Components of a simple personal computer

# Computer Hardware Review
## The CPU (1)

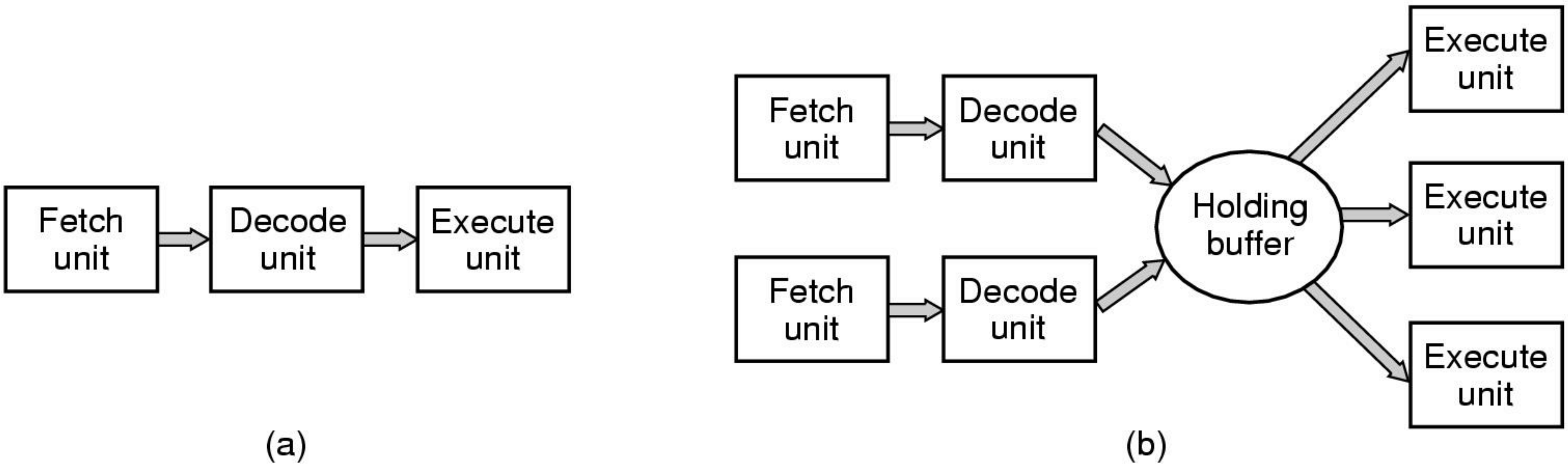
- PC Program Counter
- SP Stack Pointer
- PSW Program Status Word
- General Registers
- Instruction Cycle
- Pipeline
- Superscalar
- System Call

# Computer Hardware Review
## The CPU (2)



# Computer Hardware Review

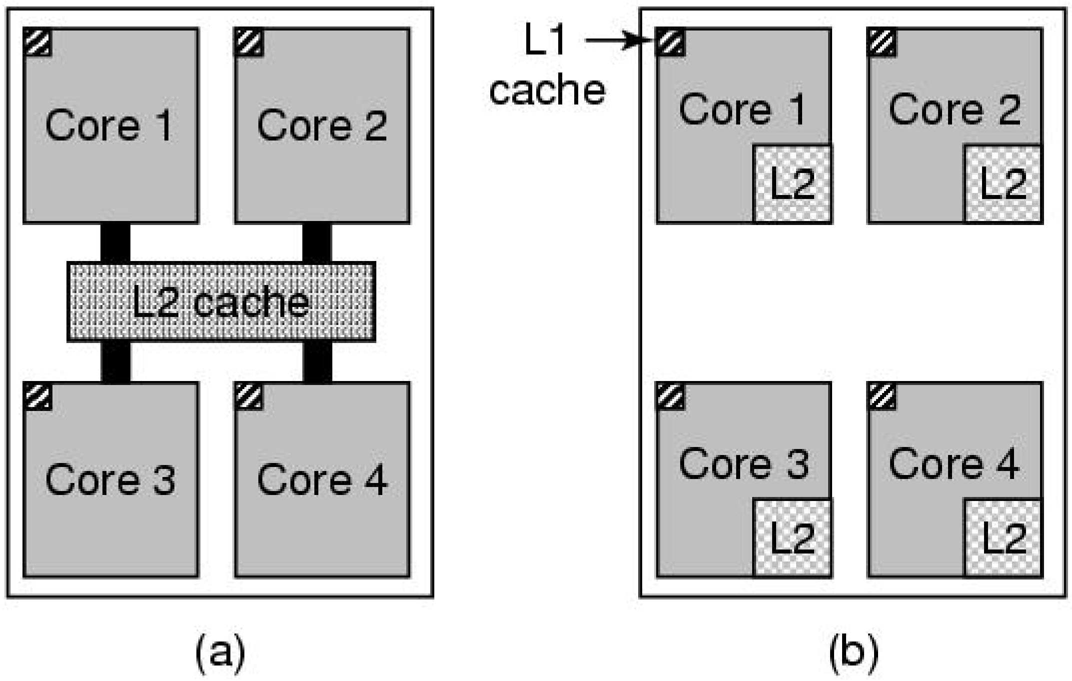
## The CPU (3)




(a) A three-stage pipeline
(b) A superscalar CPU

# Computer Hardware Review

## The CPU (4)

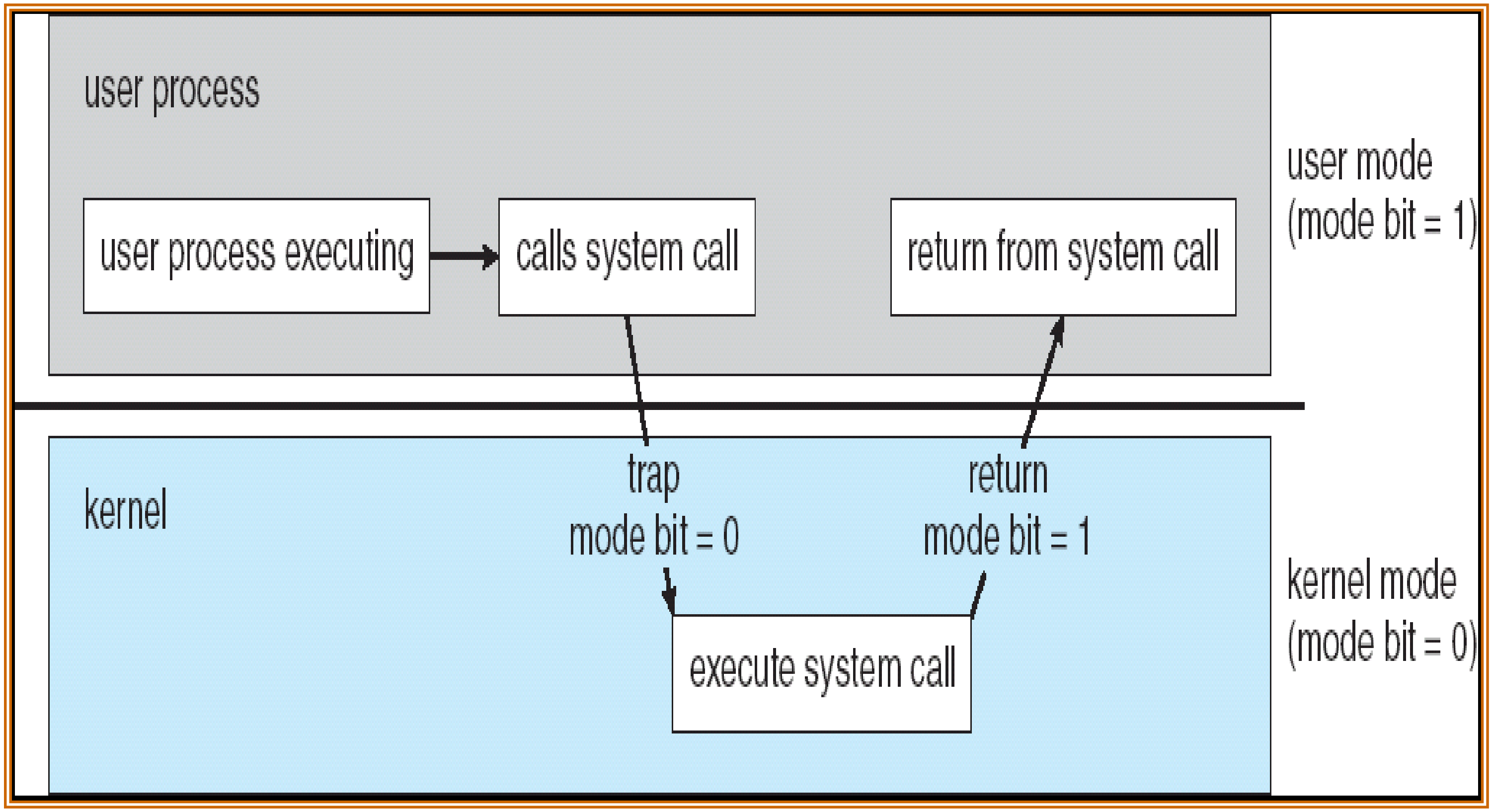


# Computer Hardware Review
## The CPU (5)

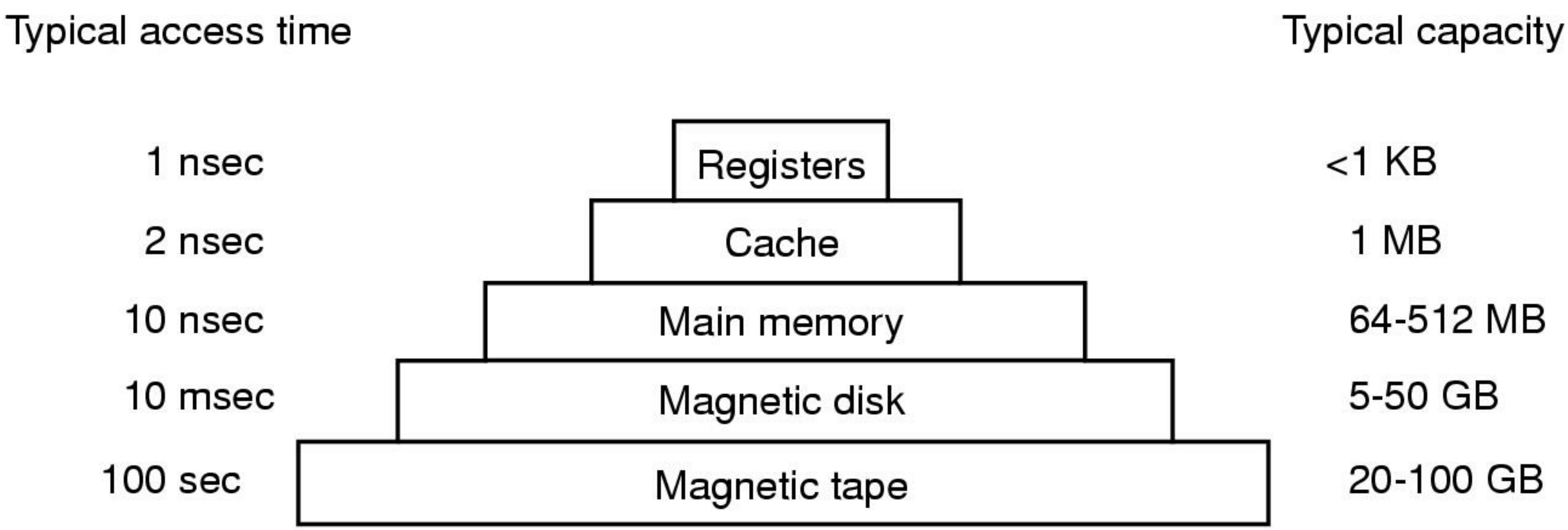
- **Dual-mode** operation allows OS to protect itself and other system components
  - **User mode** and **kernel mode**
  - **Mode bit** provided by hardware
    - Provides ability to distinguish when system is running user code or kernel code
    - Some instructions designated as **privileged**, only executable in kernel mode
    - System call changes mode to kernel, return from call resets it to user

# Computer Hardware Review
## The CPU (6)

- Transition from User to Kernel Mode



# Computer Hardware Review
## Memory (1)




- Typical memory hierarchy
  - numbers shown are rough approximations

# Computer Hardware Review
## Memory (2)

- Storage systems organized in hierarchy.
  - Speed
  - Cost
  - Size
  - Volatility

# Computer Hardware Review
## Memory (3)

**Registers:**

- Small number, Fast
- Capacity:
  - X*32 bit on 32-bit CPU
  - Y*64 bit on 64-bit CPU

# Computer Hardware Review
## Memory (4)

**Caching** – copying information into faster storage system; main memory can be viewed as a last *cache* for secondary storage.

# Computer Hardware Review
## Memory (5)

**Caching**

- Important principle, performed at many levels in a computer (in hardware, operating system, software)
- Information in use copied from slower to faster storage temporarily
- Faster storage (cache) checked first to determine if information is there
  - If it is, information used directly from the cache (fast)
  - If not, data copied to cache and used there
- Cache smaller than storage being cached
  - Cache management important design problem
  - Cache size and replacement policy

# Computer Hardware Review
## Memory (6)

### **Main Memory**

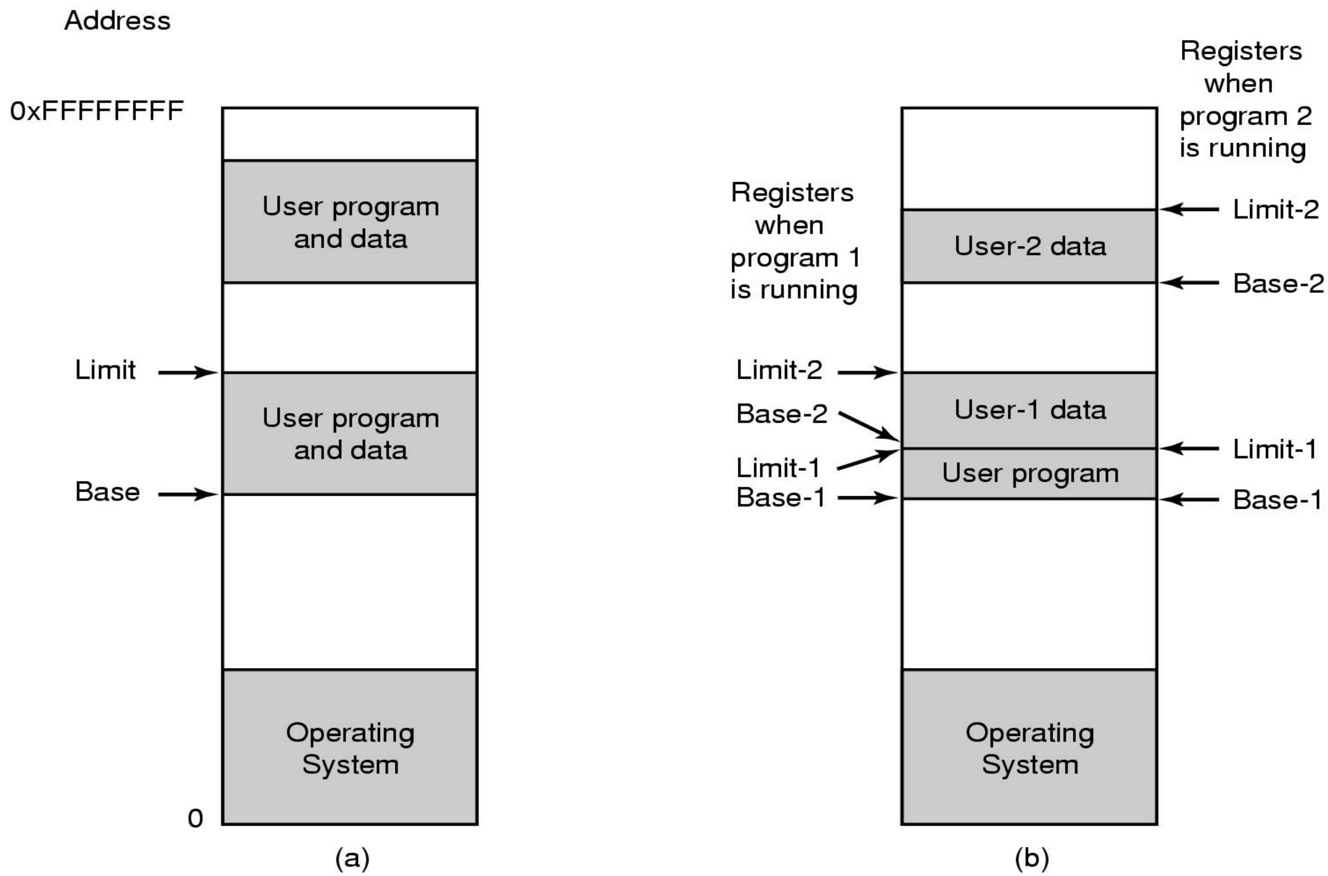
- RAM (Random Access Memory)
- Problem:
  - How protect the program from one another and the kernel from them all
  - How to handle relocation
- Solution: CPU equipped with two special registers
- Base Register and Limit Register
- MMU (Memory Management Unit) convert Virtual Address to Physical Address
- Context Switch; switching from one to another

# Computer Hardware Review
## Memory (7)

**Relocation and Protection**

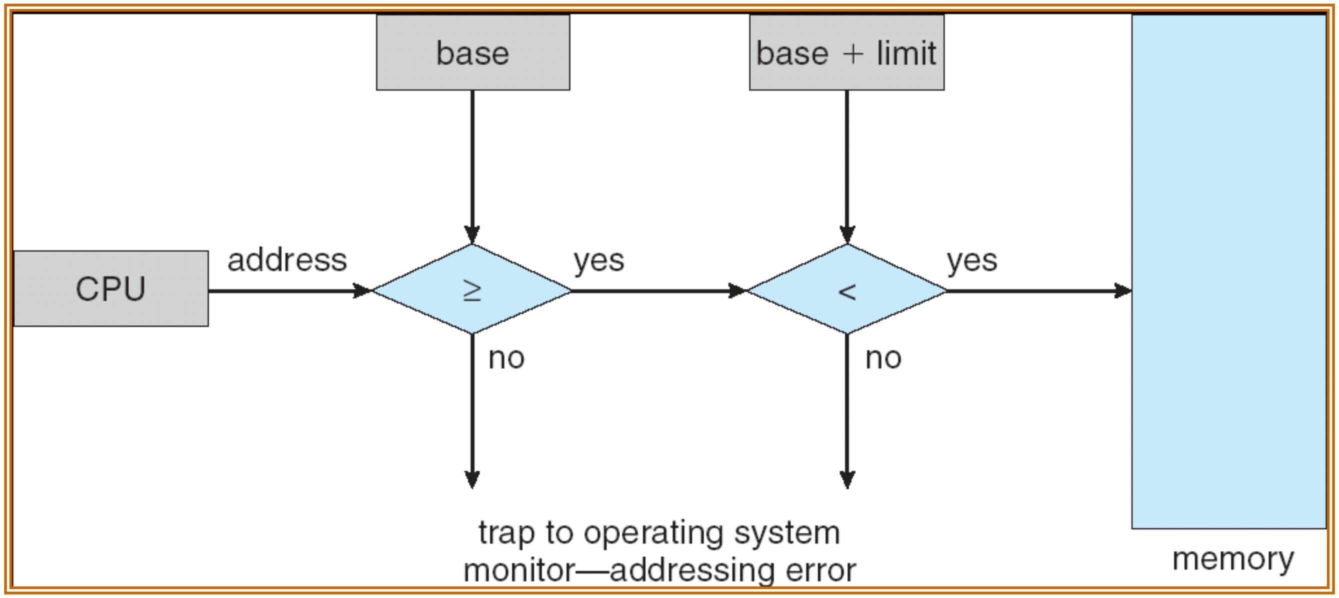
- Cannot be sure where program will be loaded in memory
  - address locations of variables,  code routines cannot be absolute
  - must keep a program out of other processes' partitions
- Use base and limit values
  - address locations added to base value to map to physical addr
  - address locations larger than limit value is an error

# Computer Hardware Review
## Memory (8)




One base-limit pair and two base-limit pairs

# Computer Hardware Review
## Memory (9)



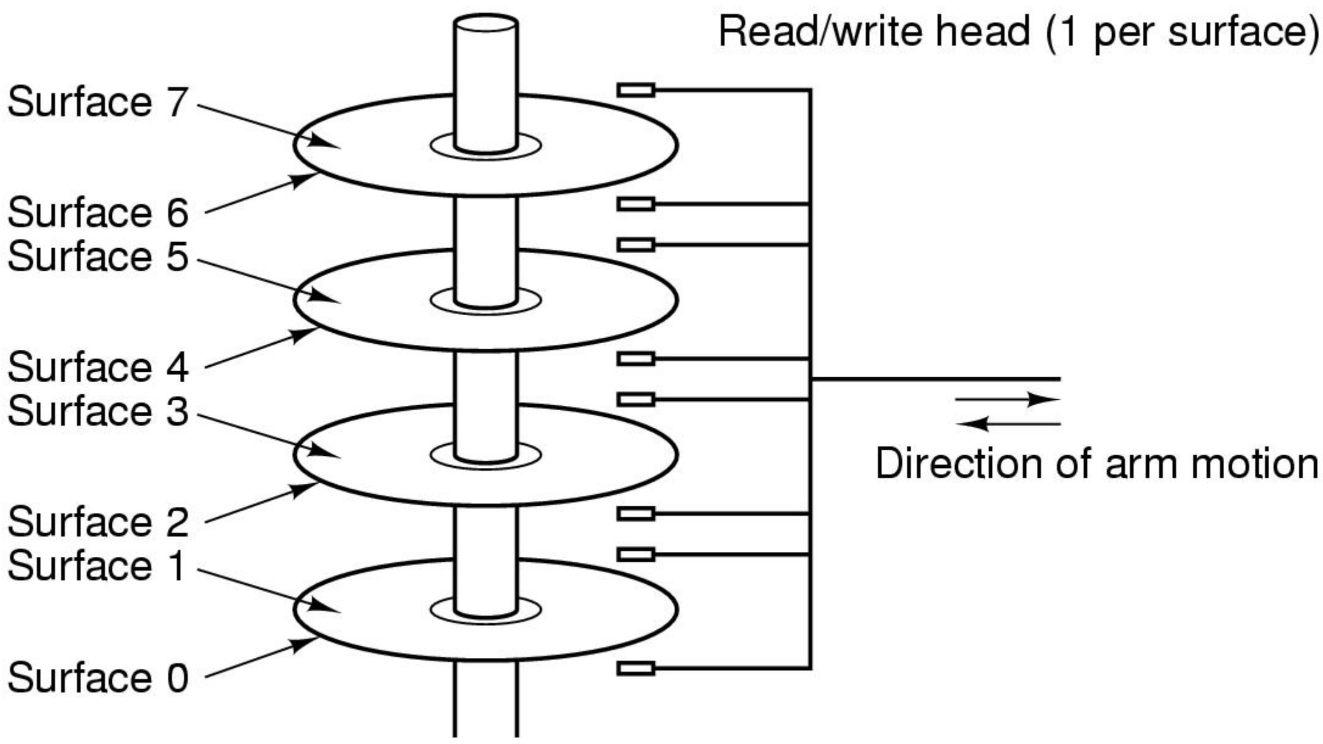
# Computer Hardware Review
## Memory (10)

Other memory in computer

- ROM (Read Only Memory)
- EEPROM (Electrically Erasable ROM):
  - BIOS Basic Input Output System
- CMOS:
  - Real time clock
  - Configuration Information

# Computer Hardware Review

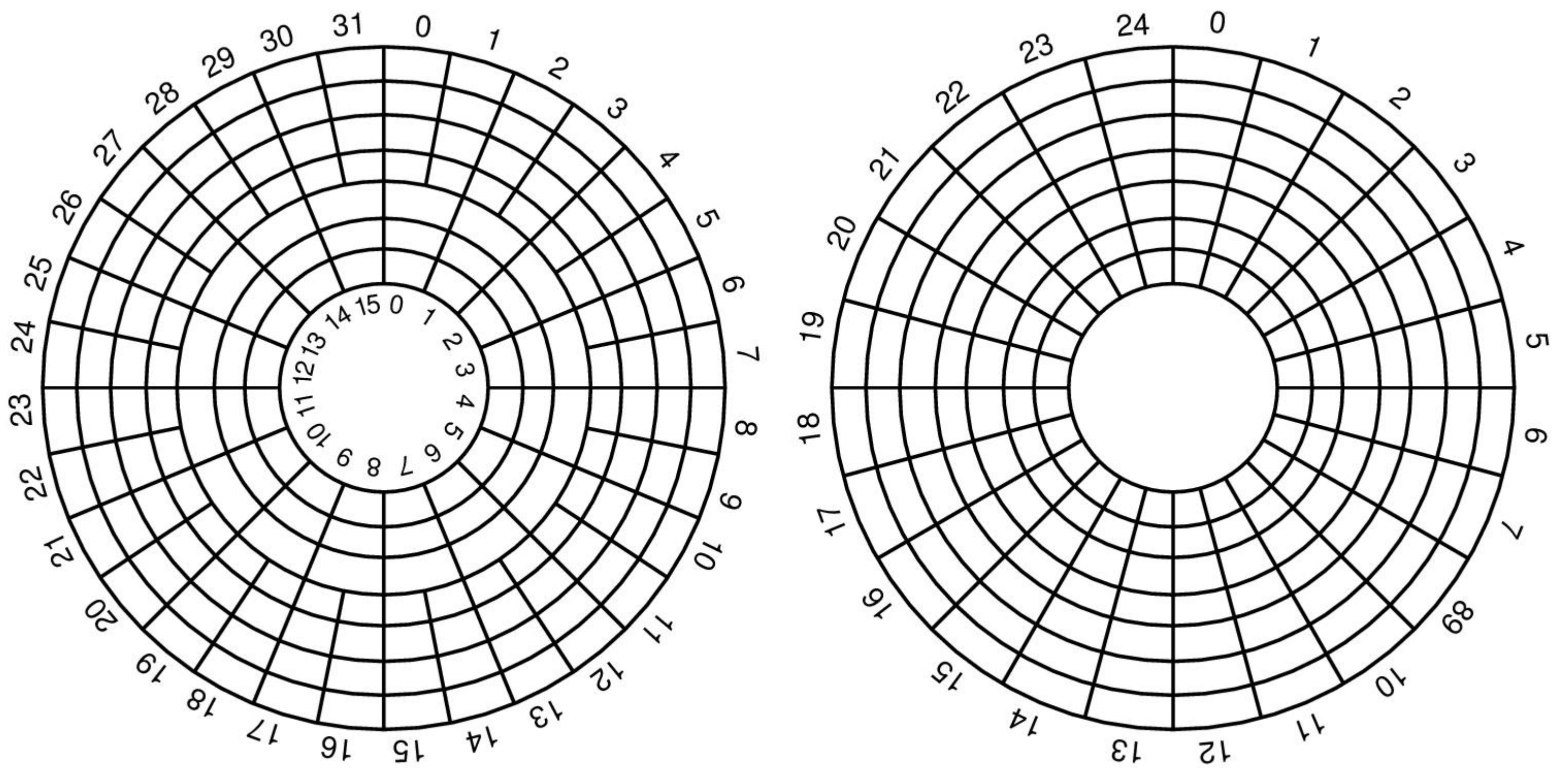
## Memory (11)




Structure of a disk drive

# Computer Hardware Review

## Memory (12)



Disk: Cylinder, Track, sector

# Computer Hardware Review

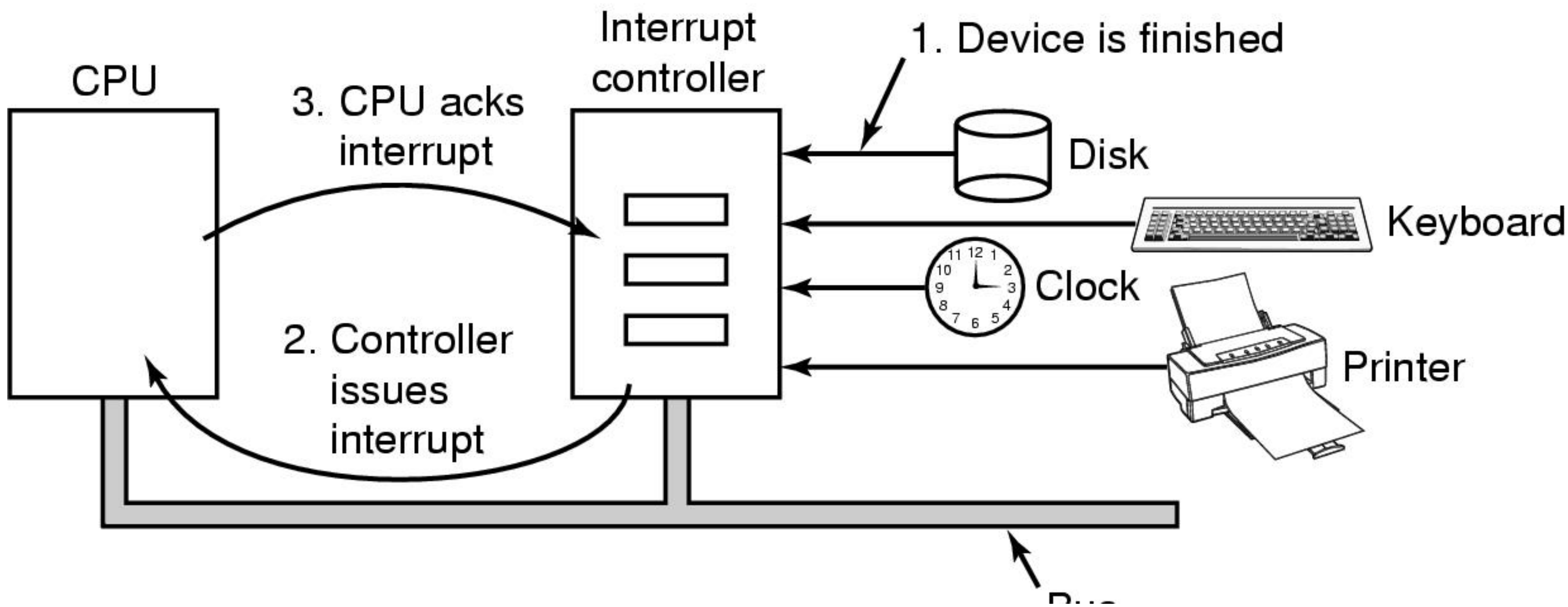
## I/O Devices (1)

Computer-System Operation

- I/O devices and the CPU can execute concurrently.
- Each device controller is in charge of a particular device type.
- Each device controller has a local buffer.
- CPU moves data from/to main memory to/from local buffers
- I/O is from the device to local buffer of controller.
- Device controller informs CPU that it has finished its operation by causing an *interrupt*.

# Computer Hardware Review

## I/O Devices (2)

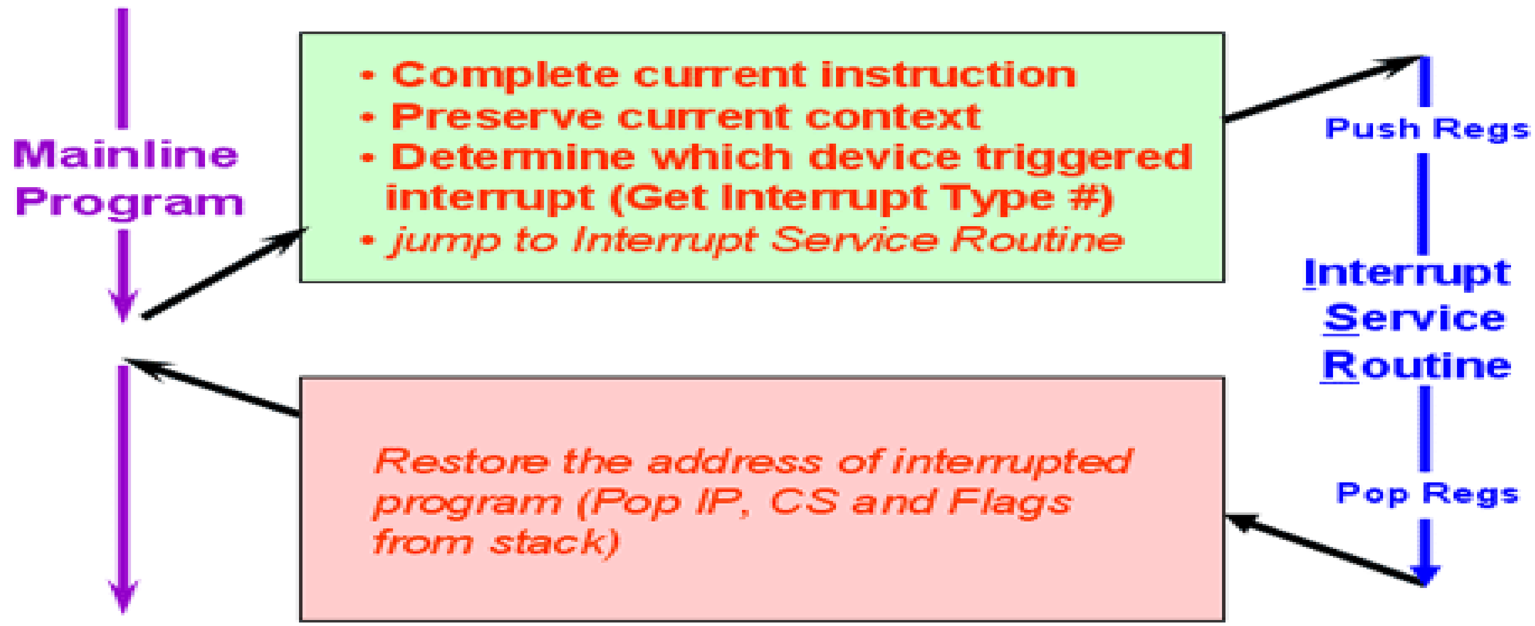



How interrupts happens. Connections between devices and interrupt controller actually use interrupt lines on the bus rather than dedicated wires

# Computer Hardware Review

## I/O Devices (3)

Example of Interrupt processing for PC



# Computer Hardware Review

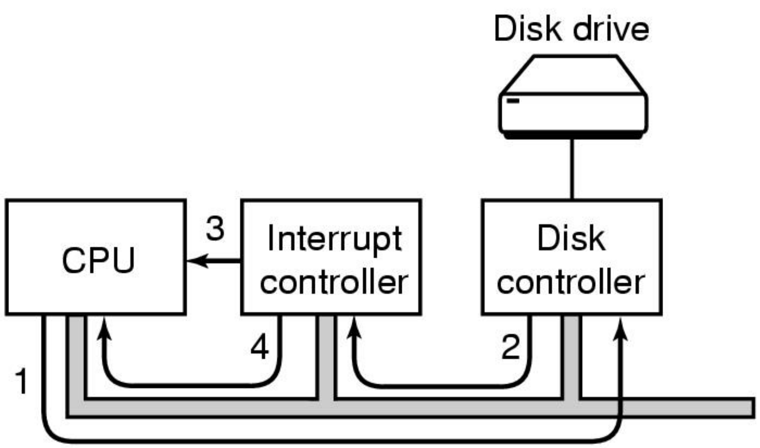
## I/O Devices (4)

### Common Functions of Interrupts

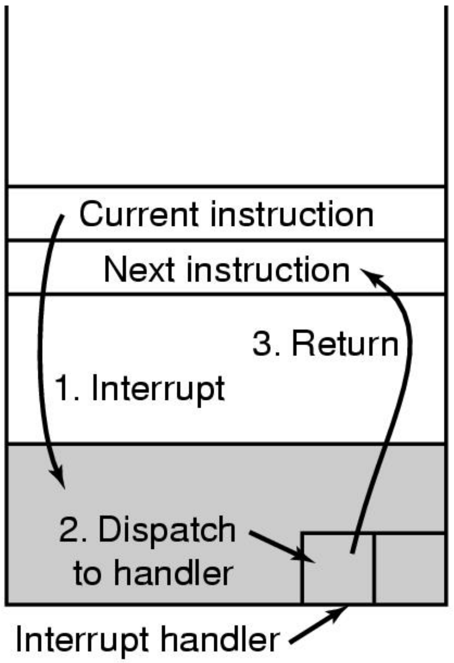
- Interrupt transfers control to the **interrupt service routine** generally, through the *interrupt vector table*, which contains the addresses of all the service routines.
- Interrupt architecture must save the address of the interrupted instruction.
- Incoming interrupts are *disabled* while another interrupt is being processed to prevent a *lost interrupt*.
- A *trap* is a software-generated interrupt caused either by an error or a user request.
- An operating system is *interrupt* driven.

# Computer Hardware Review

## I/O Devices (5)




(a)



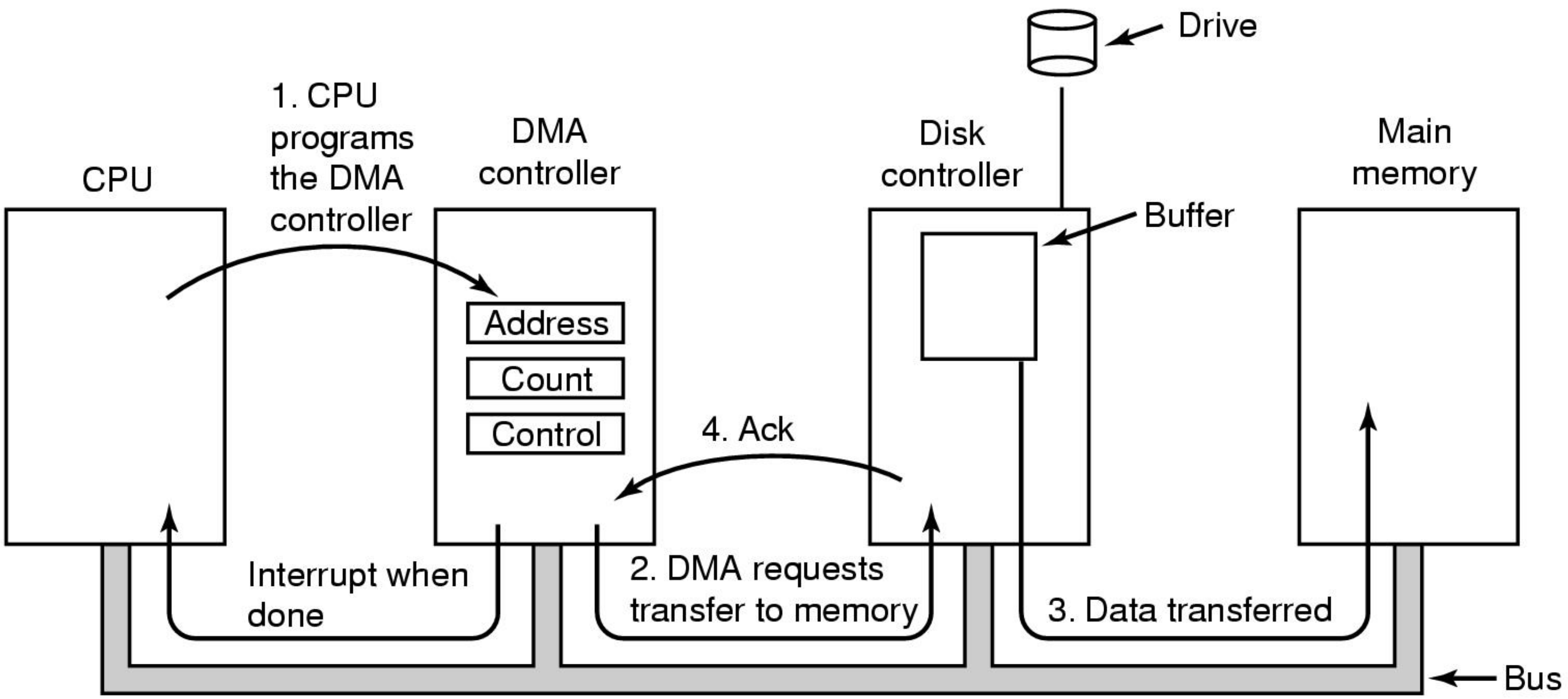

(b)

(a) Steps in starting an I/O device and getting interrupt

(b) How the CPU is interrupted

# Computer Hardware Review

## I/O Devices (6)




Operation of a DMA transfer
DMA (Direct Memory Access)

# Computer Hardware Review

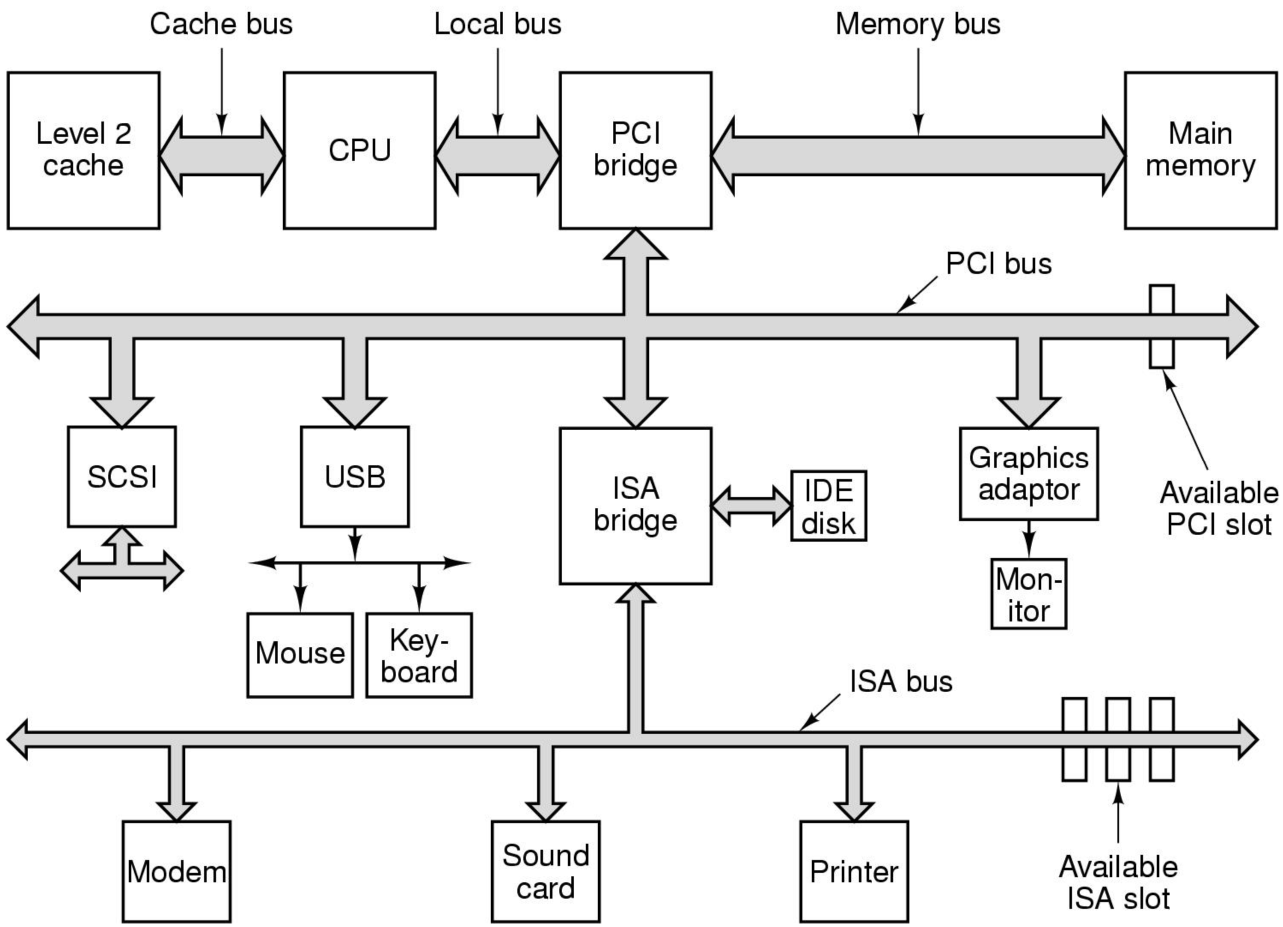
## I/O Devices (7)

**DMA**

- Used for high-speed I/O devices able to transmit information at close to memory speeds.
- Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention.
- Only one interrupt is generated per block, rather than the one interrupt per byte.

# Computer Hardware Review

## Structure of a large Pentium system



# Computer Hardware Review
# BUS

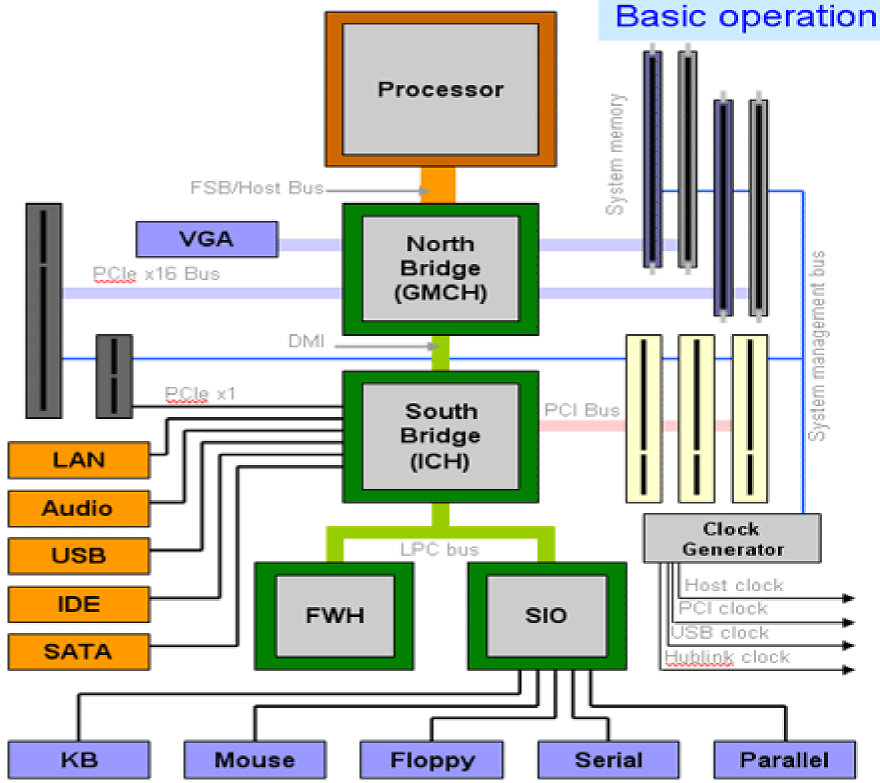
- Cache BUS
- Local BUS
- System BUS
- ISA Industry Standard Architecture
- PCI Peripheral Component  Interconnect
- USB Universal Serial BUS
- SCSI Small Computer System Interface
- IDE Integrated Drive Electronic

# Computer Hardware Review

## Personal Computer (PC)

### Chipset: North Bridge, South Bridge, & Firmware Hub

Basic operational design is called its architecture




- North Bridge
  - HOST-PCI Bridge
  - DRAM Controller
  - AGP Interface
- South Bridge (ICH)
  - PCI-PCI Bridge
  - 8254 Timer; 8259 Intr. Ctlr; 8237 DMAC; RTC
  - EIDE Interface
  - USB Ctlr
- Super I/O
  - Kybd/Mouse Ctlr
  - Floppy Ctlr
  - Serial & IR port
  - Parallel ports

# 1.4 The Operating System Zoo

# The Operating System Zoo

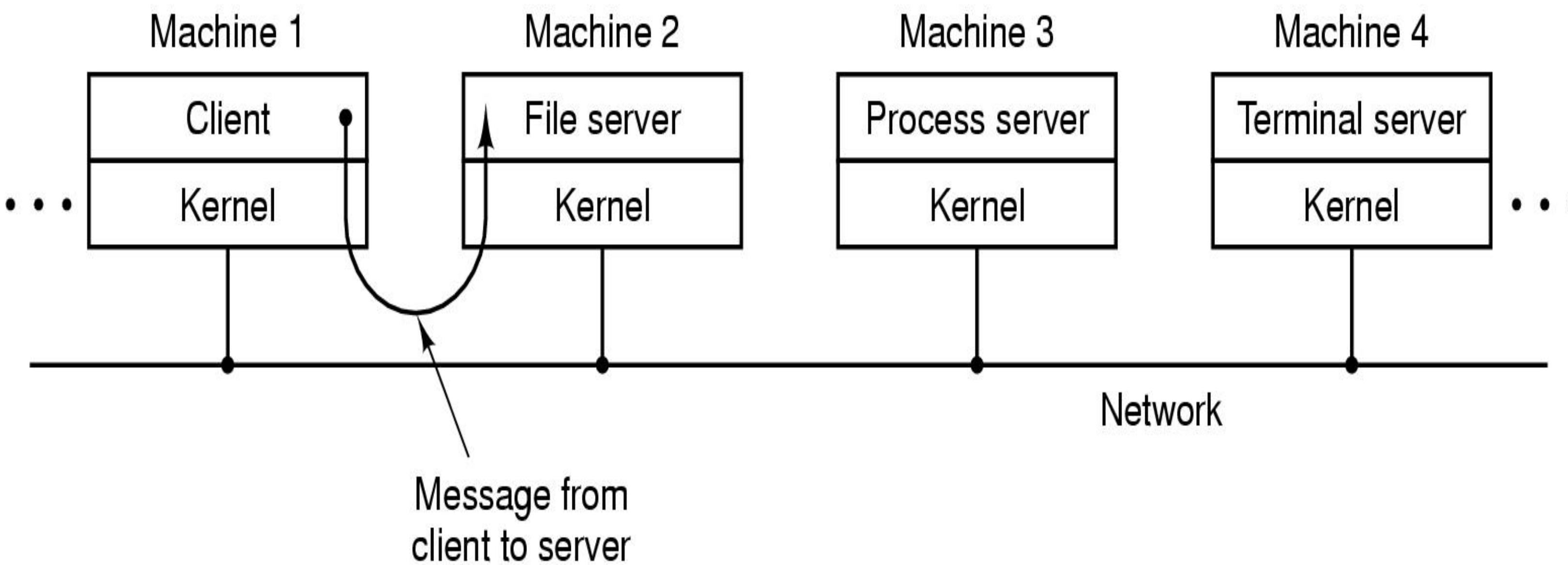
- Mainframe operating systems
- Server operating systems
- Multiprocessor operating systems
- Personal computer operating systems
- Handheld computer operating systems
- Embedded operating systems
- Sensor Node operating systems
- Real-time operating systems
- Smart card operating systems

# The Operating System Zoo
## Mainframe operating systems

- batch
- multiprogrammed
- time-sharing, multitasking
- Application: High-End Web Server, Servers for Business-To- Business transactions
- Example: OS/390

# The Operating System Zoo

## Server operating systems



# The Operating System Zoo
## Multiprocessor operating systems

- Multiple CPU
- Share computer bus, clock
- Advantage:
  - High system throughput
  - High availability
  - Multiprocessor system and Multicomputer system

# The Operating System Zoo
## Personal computer operating systems

- Many I/O devices
- Interface to single user
- Modern Personal computer operating systems support multiprogramming
- Examples: MS Windows, Mac OS, Solaris, Linux,....

# The Operating System Zoo
## Handheld Computer operating systems

- Personal digital assistant (PDA): Palmtop, Pocket-PC, Cellular phones
- Restriction of memory size, speed of  CPU, screen size, powers
- Operating System: PalmOS, Windows CE (Consumer Electronic)

# The Operating System Zoo

## Embedded operating systems

- Run on the computers that control device
- Typical examples are microwave ovens, TV set, MP3 players..
- All the software is in ROM, no untrusted software
- Examples: QNX, VxWork

# The Operating System Zoo

## Sensor Node Operating systems

- Network of tiny sensor nodes
  - Sensor nodes are the computer that communicate with each other and with a base station using wireless commnication
  - Examples: guard national borders, measure temperature for weather forecasting
- Restriction of memory size, speed of CPU, Powers (Batteries)
- Example: TinyOS

# The Operating System Zoo
## Real-time operating systems

- Time is a key parameter
- Two types of real-time system
  - Hard real-time system for industrial process control system…
  - Soft real-time system for multimedia system

# The Operating System Zoo
## Smart card operating systems

- CPU chips on Card
- Severe processing power and memory constraint
- Specific Application:
  - Single function: electronic payments
  - Multiple function: proprietary systems
  - Java oriented: holds interpreter JVM

# 1.5 Operating System Concepts

# OS Components

- Process Management
- Main Memory Management (Address Spaces)
- File management
- I/O Management
- Protection and Security
- Operating System Services (The Shell)

# Operating System Concepts
## OS Components (1)

- **Process Management**
  - Process is a Program in execution
  - A process needs certain resources, including CPU time, memory, files, and I/O devices, to accomplish its task.
  - Tasks of process management of OS:
    - Process creation and deletion.
    - process suspension and resume
    - Provision of mechanisms for:

      - Process synchronization

      - Interprocess communication

      - Prevent or avoid deadlock

# Operating System Concepts
## OS Components (2)

- **Main Memory Management**:
  - Motivations:
    - Increase system performance
    - Maximize memory utilization
  - Task of main memory management:
    - Keep track of which parts of memory are currently being used and by whom.
    - Decide which processes to load when memory space becomes available.
    - Allocate and deallocate memory space as needed

# Operating System Concepts
## OS Components (3)

- **File management**
  - File system
    - File
    - Directory
  - Task of file management of OS:
    - Create and delete File/Dicrectory
    - Manipulate: rename, copy, move, new,...
    - Mapping files onto secondary storage.
    - File backup on stable (nonvolatile) storage media

# Operating System Concepts
## OS Components (4)

- **I/O Management**
  - Hide the specialty of H/W devices
  - Task of I/O Management of OS:
    - Manage main memory for the devices using caching, buffering, and spooling
    - Maintain and provide a general device-driver interfaces
    - Drivers for specific hardware devices.

# Operating System Concepts
## OS Components (5)

### Protection and Security

- **Protection** – any mechanism for controlling access of processes or user’s resources defined by the OS
- **Security** – defense of the system against internal and external attacks
  - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service

# Operating System Concepts
## OS Components (7)

### **Protection and Security**

- Systems generally first distinguish among users, to determine who can do what
  - User identities (**user IDs**, security IDs) include name and associated number, one per user
  - User ID then associated with all files, processes of that user to determine access control
  - Group identifier (g**roup ID**) allows set of users to be defined and controls managed, then also associated with each process, file
  - **Privilege escalation** allows user to change to effective ID with more rights

# Operating System Concepts
## Operating System Services

- **One set of operating-system services provides functions that are helpful to the user:**
  - User interface - Almost all operating systems have a user interface (UI)
    - Varies between Command-Line (CLI), Graphics User Interface (GUI)
  - Program execution - The system must be able to load a program into memory and to run that program, end execution, either normally or abnormally (indicating error)
  - Control access to I/O device.
  - File-system manipulation

# Operating System Concepts
## Operating System Services

- Communications – Processes may exchange information, on the same computer or between computers over a network
- Error detection – OS needs to be constantly aware of possible errors
- ….

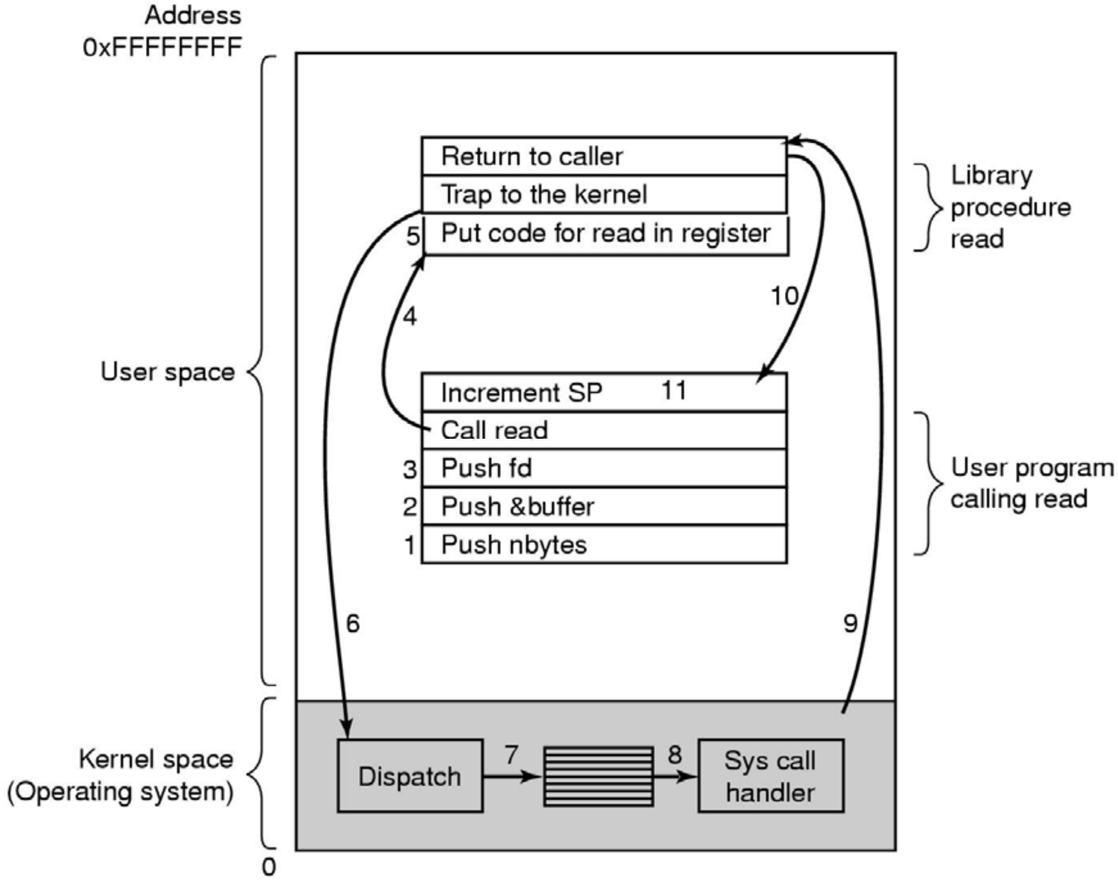
# 1.6 System Calls

# System Calls

- Making System Calls
- Major POSIX System Calls (Lab)
- Examples of System Calls  (Lab)

# System Calls

## Making System Calls (1)



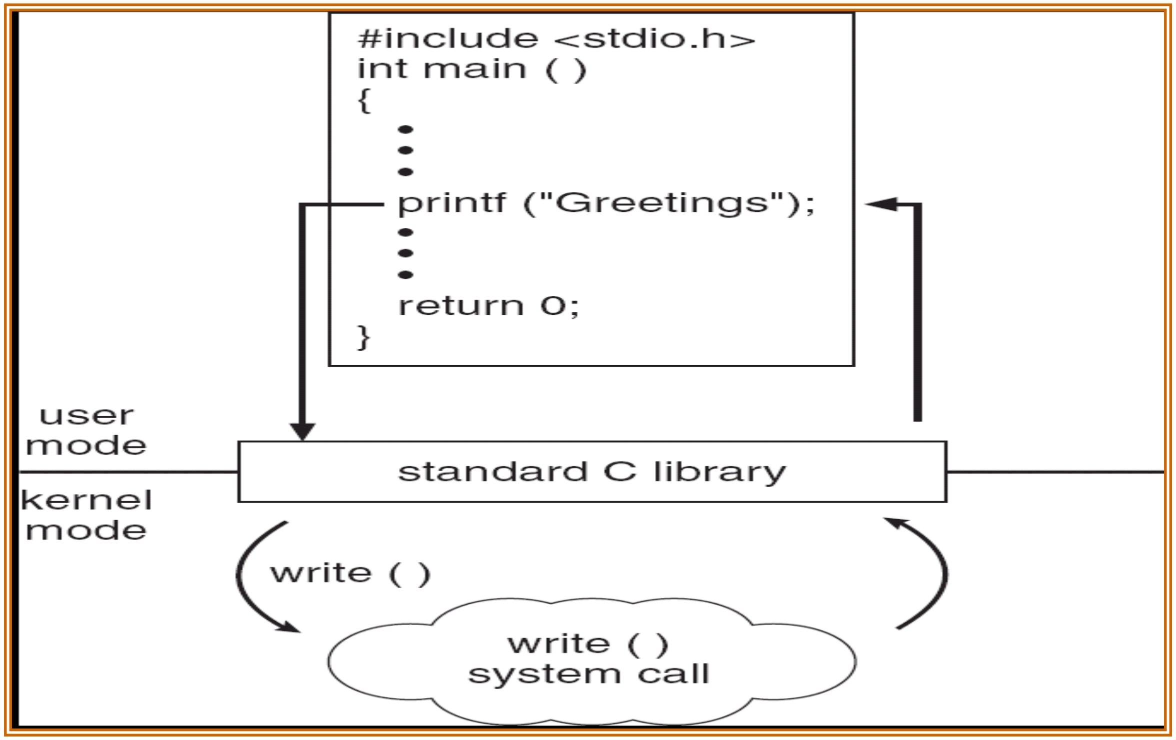

There are 11 steps in making the system call

read (fd, buffer, nbytes)

# System Calls

## Making System Calls (2)

- C program invoking printf() library call, which calls write() system call



# System Calls
## Examples

| UNIX | Win32 | Description |
|---|---|---|
| fork | CreateProcess | Create a new process |
| waitpid | WaitForSingleObject | Can wait for a process to exit |
| execve | (none) | CreateProcess = fork + execve |
| exit | ExitProcess | Terminate execution |
| open | CreateFile | Create a file or open an existing file |
| close | CloseHandle | Close a file |
| read | ReadFile | Read data from a file |
| write | WriteFile | Write data to a file |
| lseek | SetFilePointer | Move the file pointer |
| stat | GetFileAttributesEx | Get various file attributes |
| mkdir | CreateDirectory | Create a new directory |
| rmdir | RemoveDirectory | Remove an empty directory |
| link | (none) | Win32 does not support links |
| unlink | DeleteFile | Destroy an existing file |
| mount | (none) | Win32 does not support mount |
| umount | (none) | Win32 does not support mount |
| chdir | SetCurrentDirectory | Change the current working directory |
| chmod | (none) | Win32 does not support security (although NT does) |
| kill | (none) | Win32 does not support signals |
| time | GetLocalTime | Get the current time |

The Win32 API calls that roughly correspond to the UNIX calls

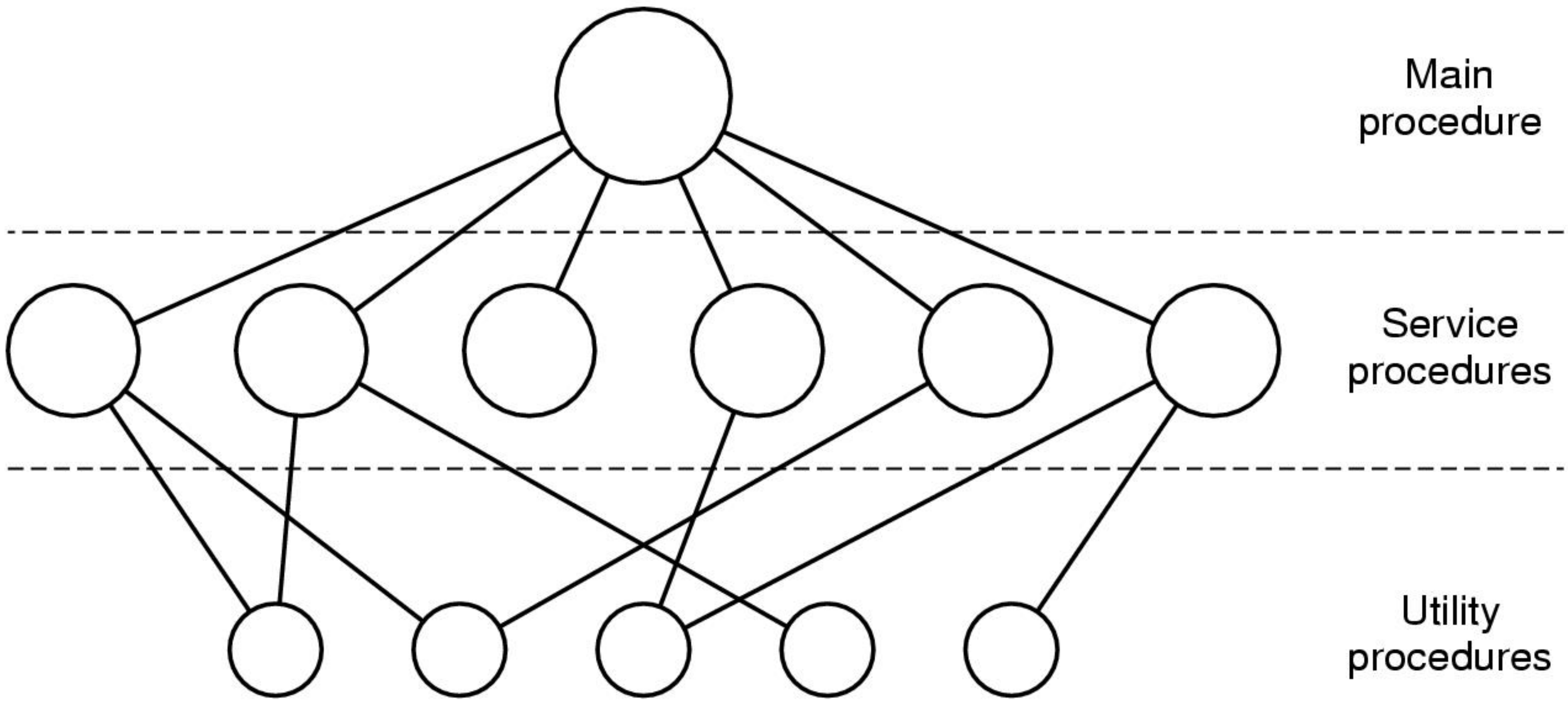
# 1.7 Operating System Structure

# Operating System Structure

- Monolithic systems
- Layered Systems
- Microkernel
- Client-server model
- Virtual Machines
- Exokernel

# Operating System Structure

## Monolithic system (1)




Simple structuring model for a monolithic system

# Operating System Structure

## Monolithic system (2)

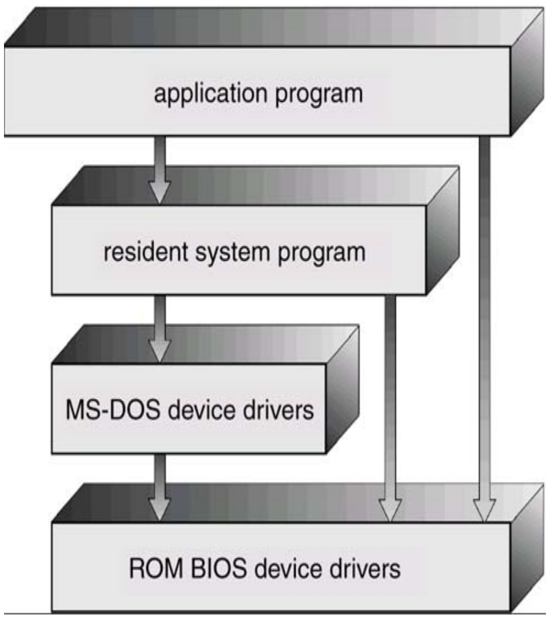
Structure of Operating System:

- A main program that invokes the requested service procedure.
- A set of service procedures that carry out the system calls.
- A set of utility procedures that help the service procedures.

# Operating System Structure

## Monolithic system (3) : Example

- Monolithic
  - MS-DOS – written to provide the most functionality in the least space:
    - not divided into modules;
    - Although MS-DOS has some structure, its interfaces and levels of functionality are not well separated



# Operating System Structure
## Layered System (1)

- Many Layers
- Each layer has well defined functions
- Upper layer can only calls functions of closely lower layer
- Advantages:
  - Easier to extend
  - Easier to debug from lower to upper layer

# Operating System Structure
## Layered System (2): Example

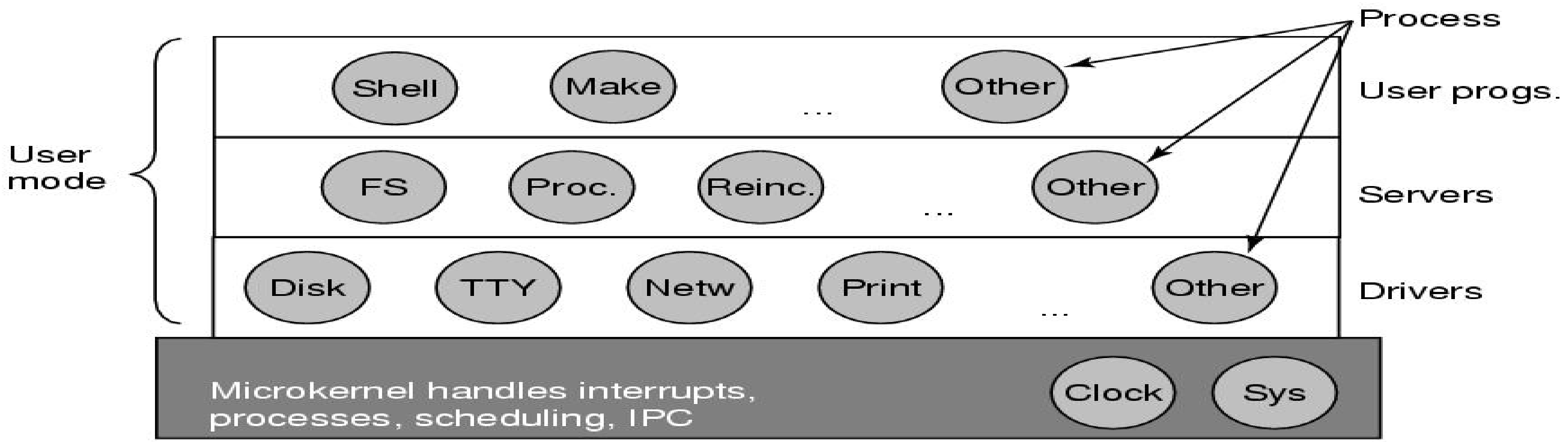
| Layer | Function |
|---|---|
| 5 | The operator |
| 4 | User programs |
| 3 | Input/output management |
| 2 | Operator-process communication |
| 1 | Memory and drum management |
| 0 | Processor allocation and multiprogramming |

Structure of the THE operating system

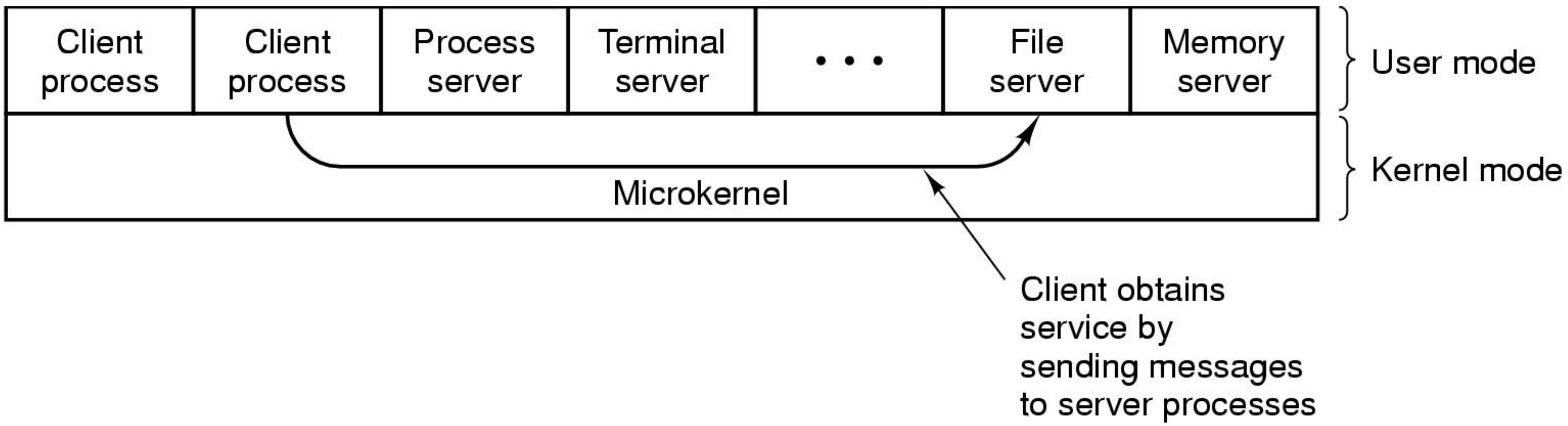
# Operating System Structure
## Microkernel: Example

### Microkernel

- Structure of the MINIX 3 system.
  - Moves as much from the kernel into “*user*” space
  - kernel → microkernel



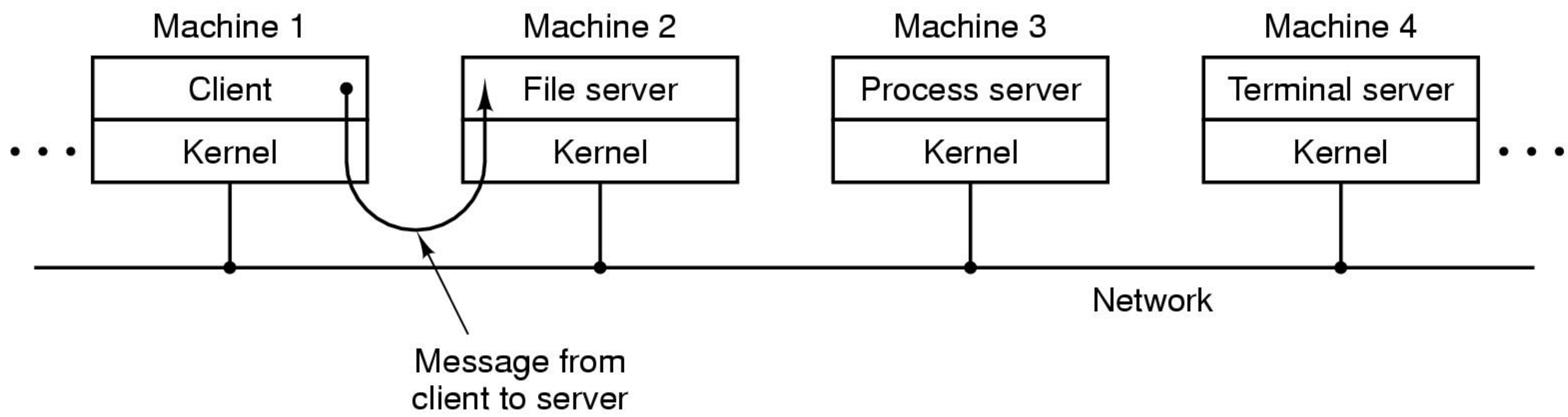
# Operating System Structure
## Client-server model (1)




The client-server model

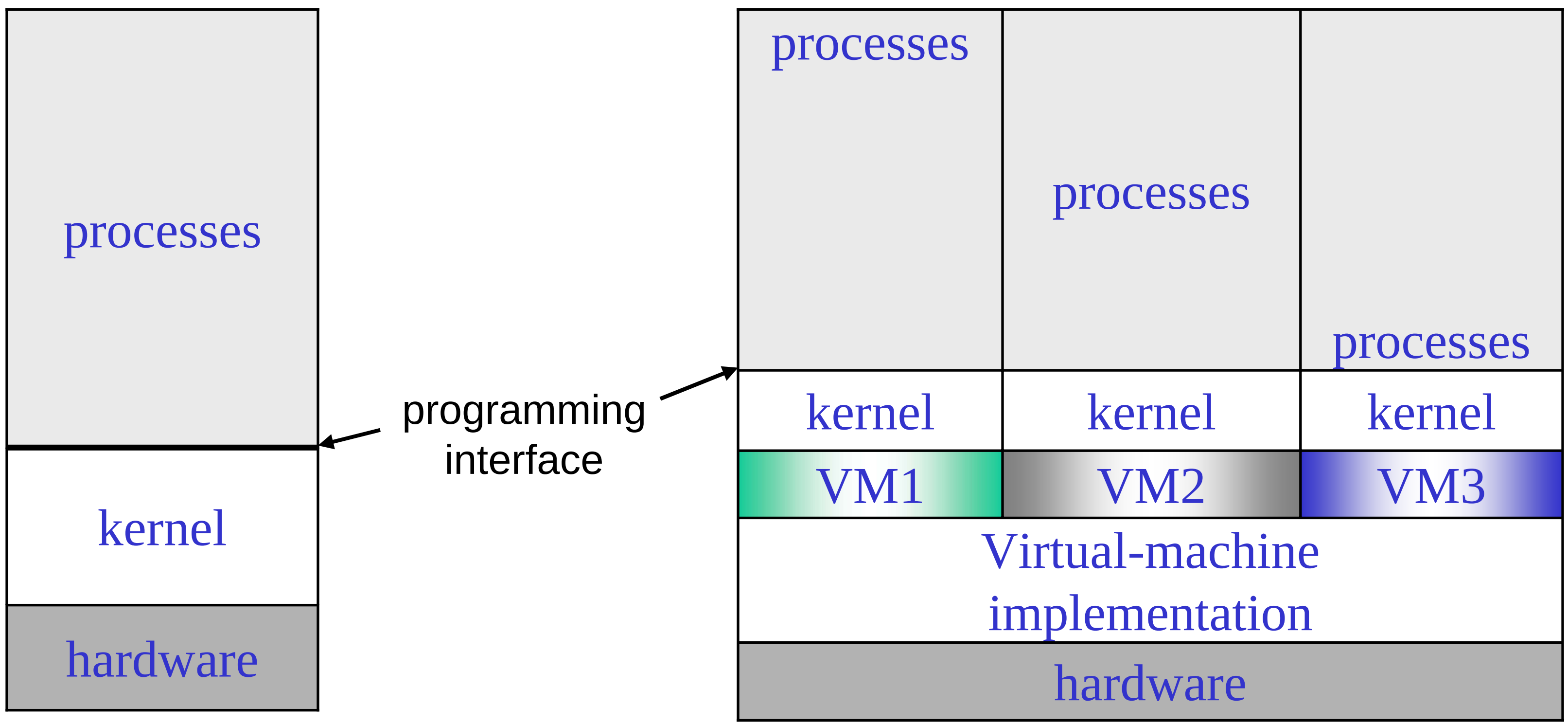
# Operating System Structure

## Client-server model (2)




The client-server model in a distributed system

# Operating System Structure
## Virtual Machine (1)




Non-virtual machine
system model

Virtual machine system model

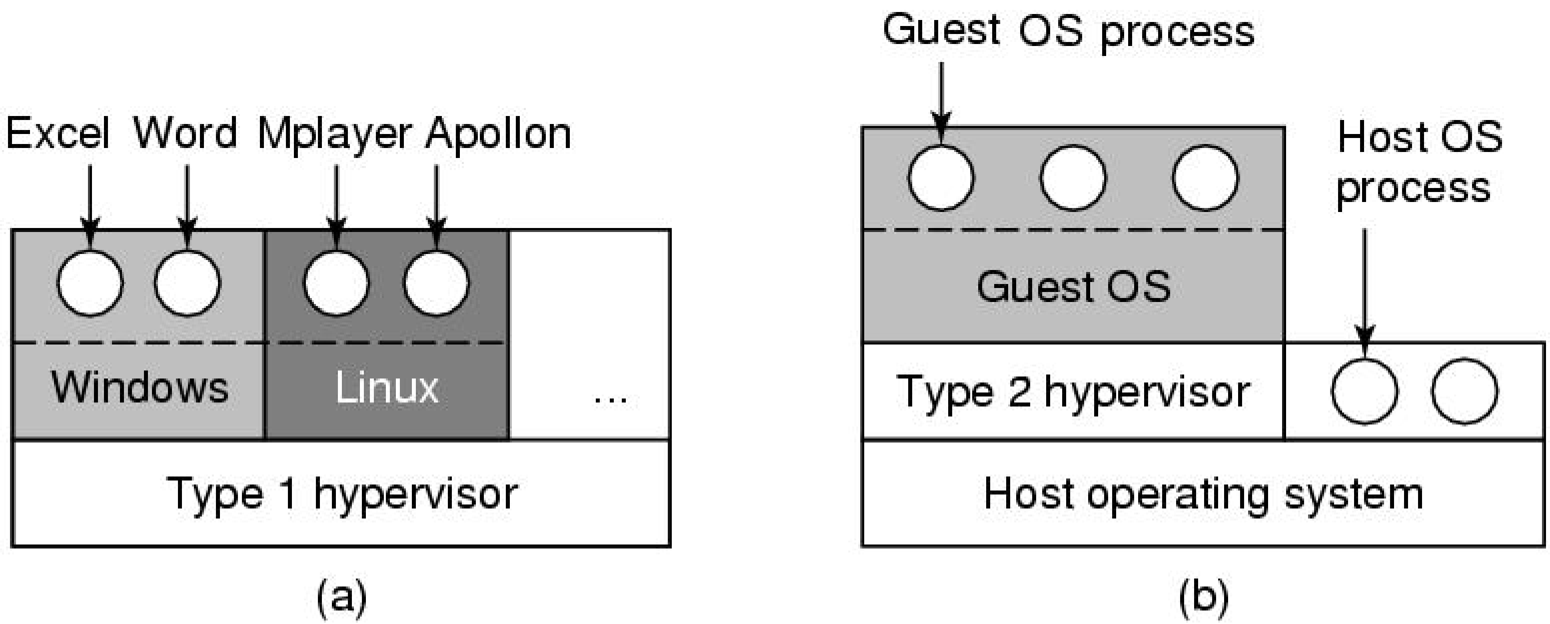
# Operating System Structure
## Virtual Machine (2)

- A *virtual machine* takes the layered approach to its logical conclusion. It treats hardware and the operating system kernel as though they were all hardware
- A virtual machine provides an interface *identical* to the underlying bare hardware
- The operating system creates the illusion of multiple processes, each executing on its own processor with its own (virtual) memory

# Operating System Structure
## Virtual Machine (3)

- The resources of the physical computer are shared to create the virtual machines
  - CPU scheduling can create the appearance that users have their own processor
  - Spooling and a file system can provide virtual card readers and virtual line printers
  - A normal user time-sharing terminal serves as the virtual machine operator's console

# Operating System Structure
# Virtual Machine (4)




(a) A type 1 hypervisor. (b) A type 2 hypervisor.

# Operating System Structure
## Exokernel

- Give user a subset of the resources
- The program, running in kernel mode called the exokernel
- Exokernel need only keep track of which virtual machine assigned which resource
- Example: Pentium virtual 8086