Operating Systems

CSE-2008

Dr Sunil Kumar Singh

Assistant Professor

School - SCOPE

VIT-AP Amaravati

sunil.singh@vitap.ac.in

Cabin - Room-223 (CB)

Module No. 1

VIT-AP UNIVERSITY

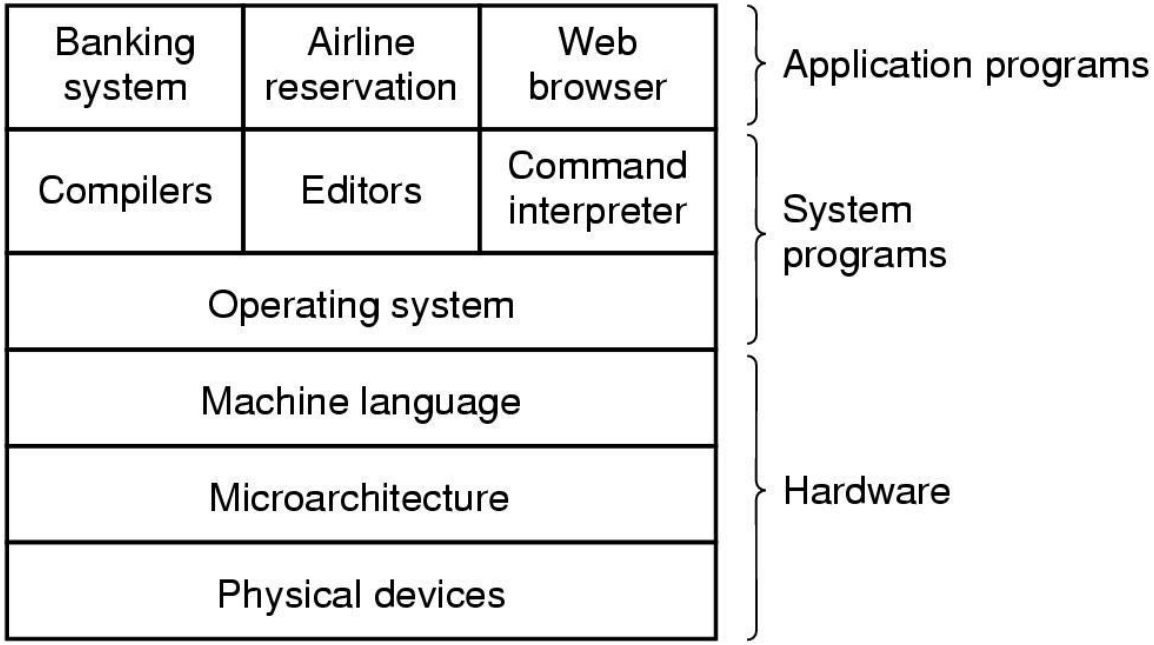
Introduction

1.1 What is an operating system
1.2 History of operating systems
1.3 Types of Operating Systems
1.4 Computer System Organization
1.5 Operating system operations
1.6 System calls

What is an Operating System

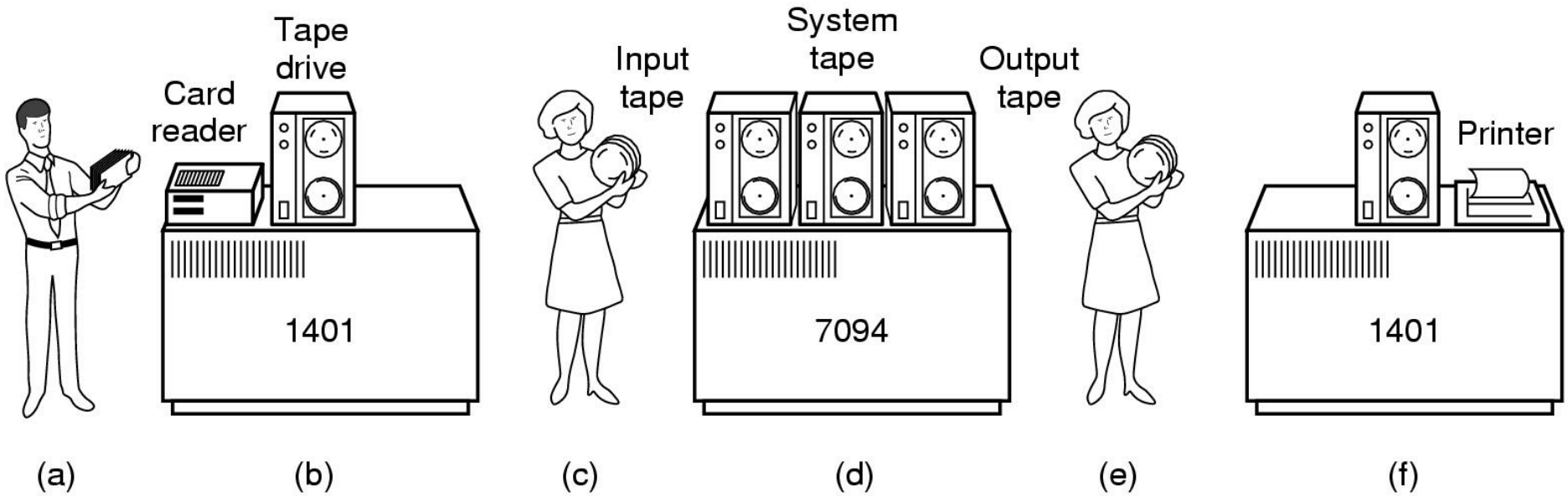
- An operating system (OS) is a collection of programs that achieve *effective utilization* of a computer system by providing
  - Convenient methods of using a computer
  - Efficient use of the computer
- A modern OS supports different classes of users with diverse requirements

Introduction



- A computer system consists of
  - hardware
  - system programs
  - application programs

History of Operating Systems (1)



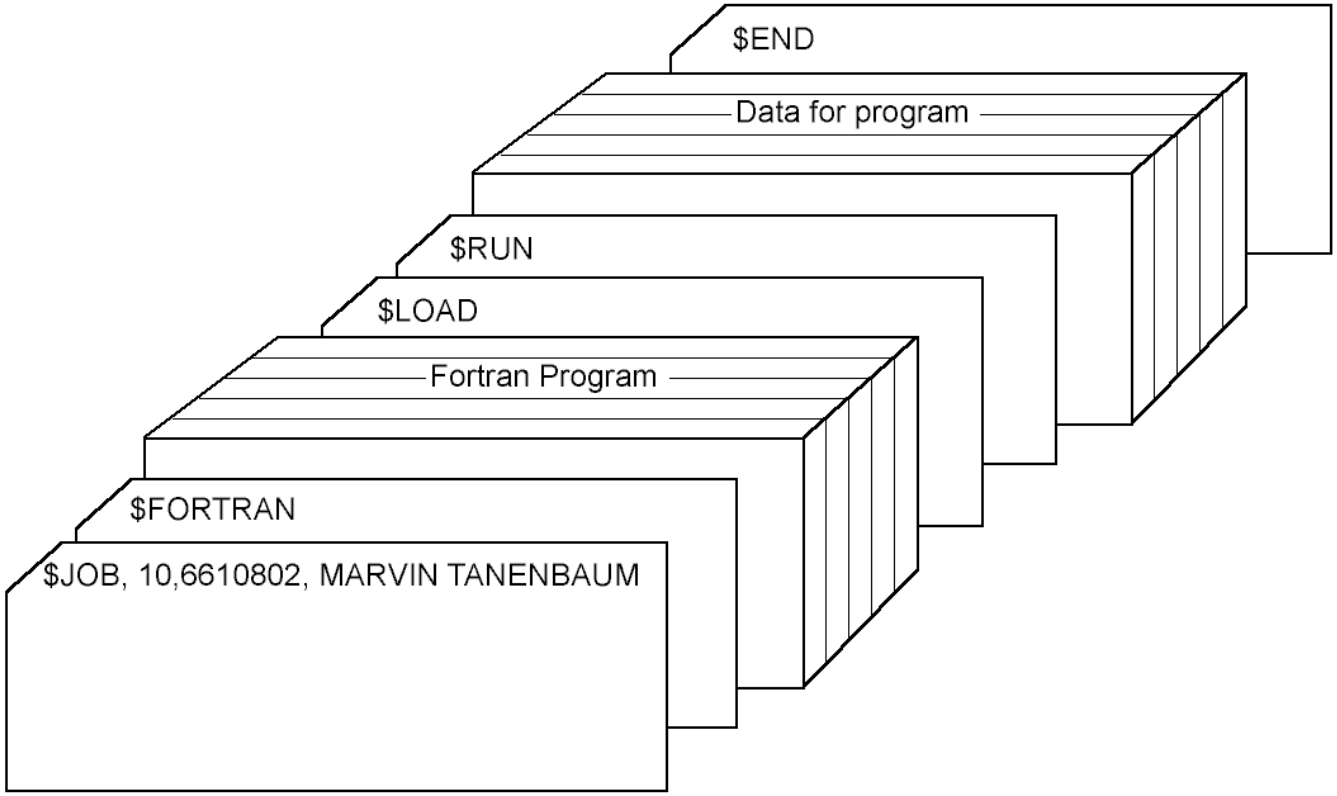
Early batch system

- 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 (2)

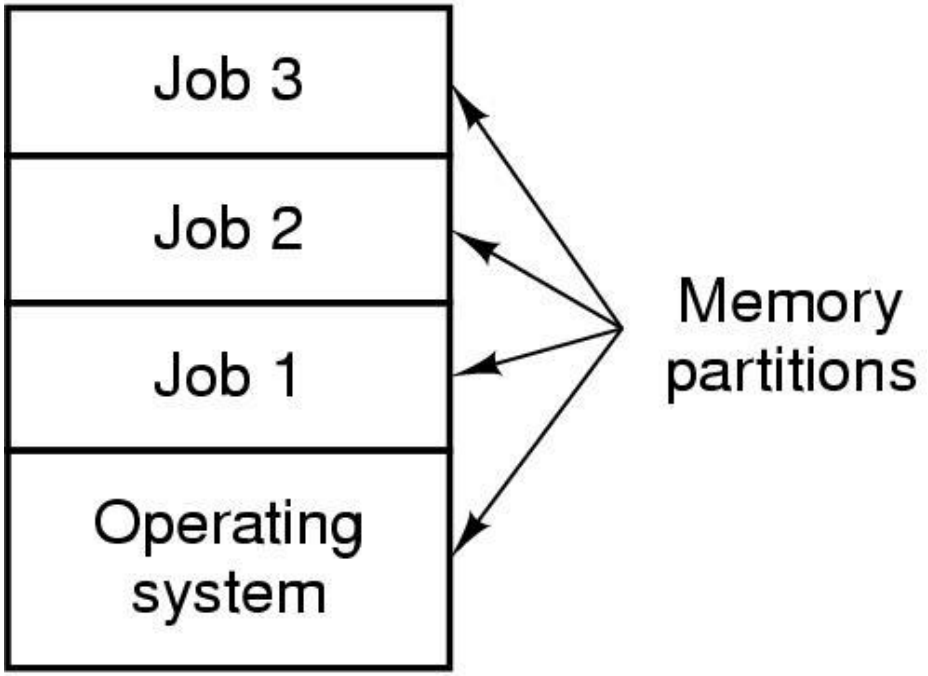
- First generation 1945 - 1955
  - vacuum tubes, plug boards
- Second generation 1955 - 1965
  - transistors, batch systems
- Third generation 1965 – 1980
  - ICs and multiprogramming
- Fourth generation 1980 – present
  - personal computers

History of Operating Systems (3)



- Structure of a typical FMS job – 2nd generation

History of Operating Systems (4)

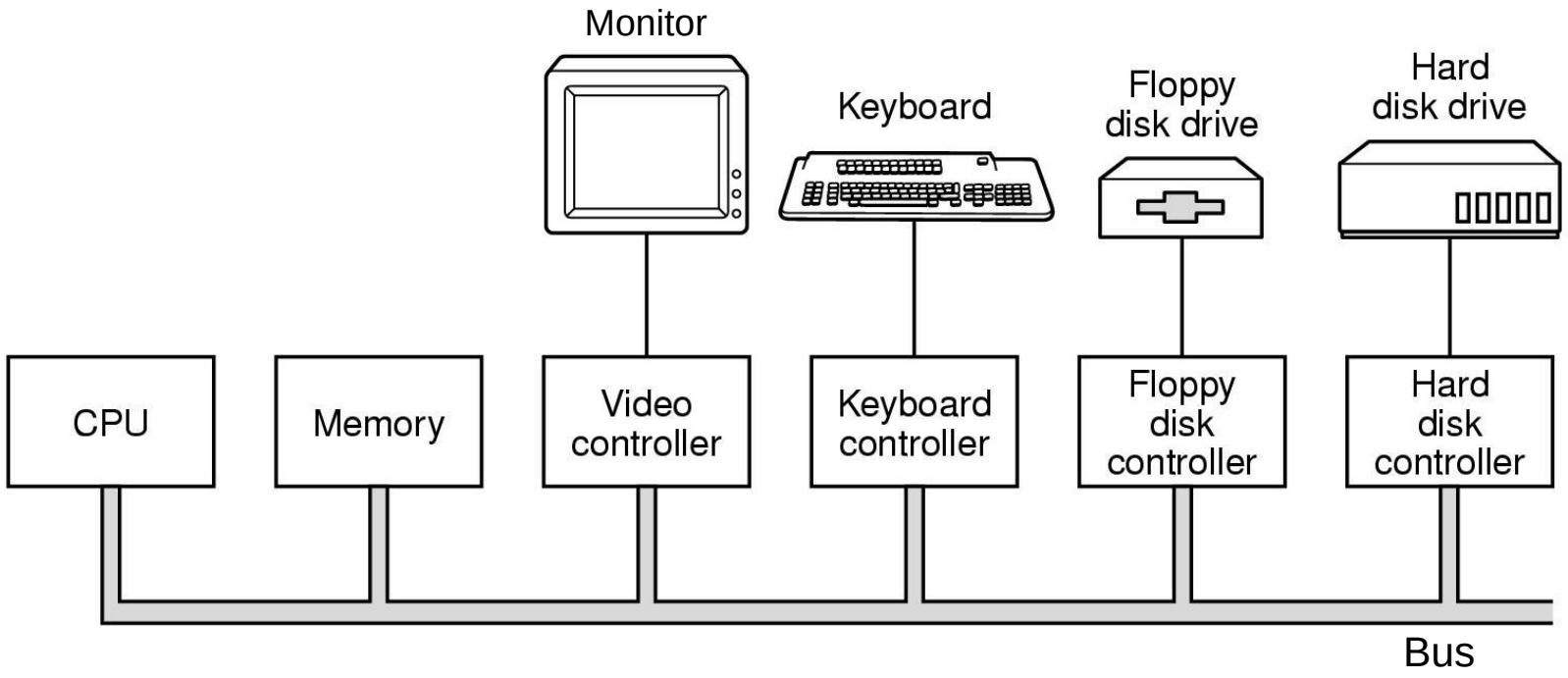


- Multiprogramming system
  - three jobs in memory – 3rd generation

Goals of an OS

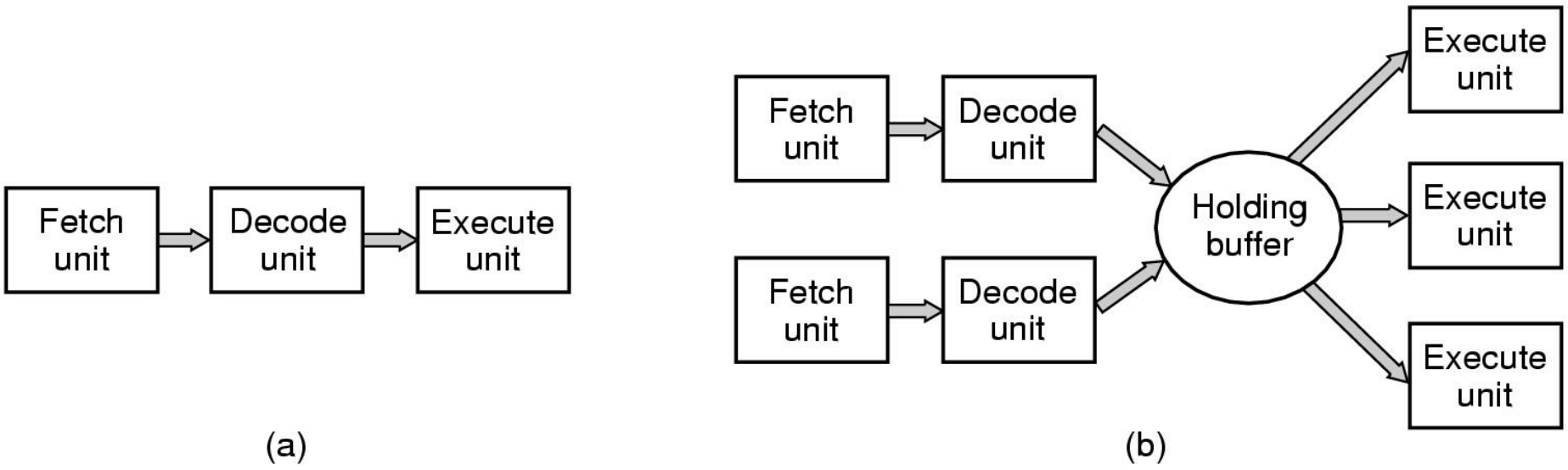
- Two primary goals of an OS are
  - Efficient use of the computer's resources
    - To ensure cost-effectiveness of the computer
  - User convenience
    - A user should find it easy to use the computer
- These two goals sometimes conflict
  - e.g. a user desires prompt service, which can be provided by not permitting other users to use the system simultaneously; however, efficient use requires sharing of a computer's resources among many users
- An OS designer decides which of the two goals is more important under what conditions.
  - That is why we have so many operating systems!

Computer System Organization



- Components of a simple personal computer

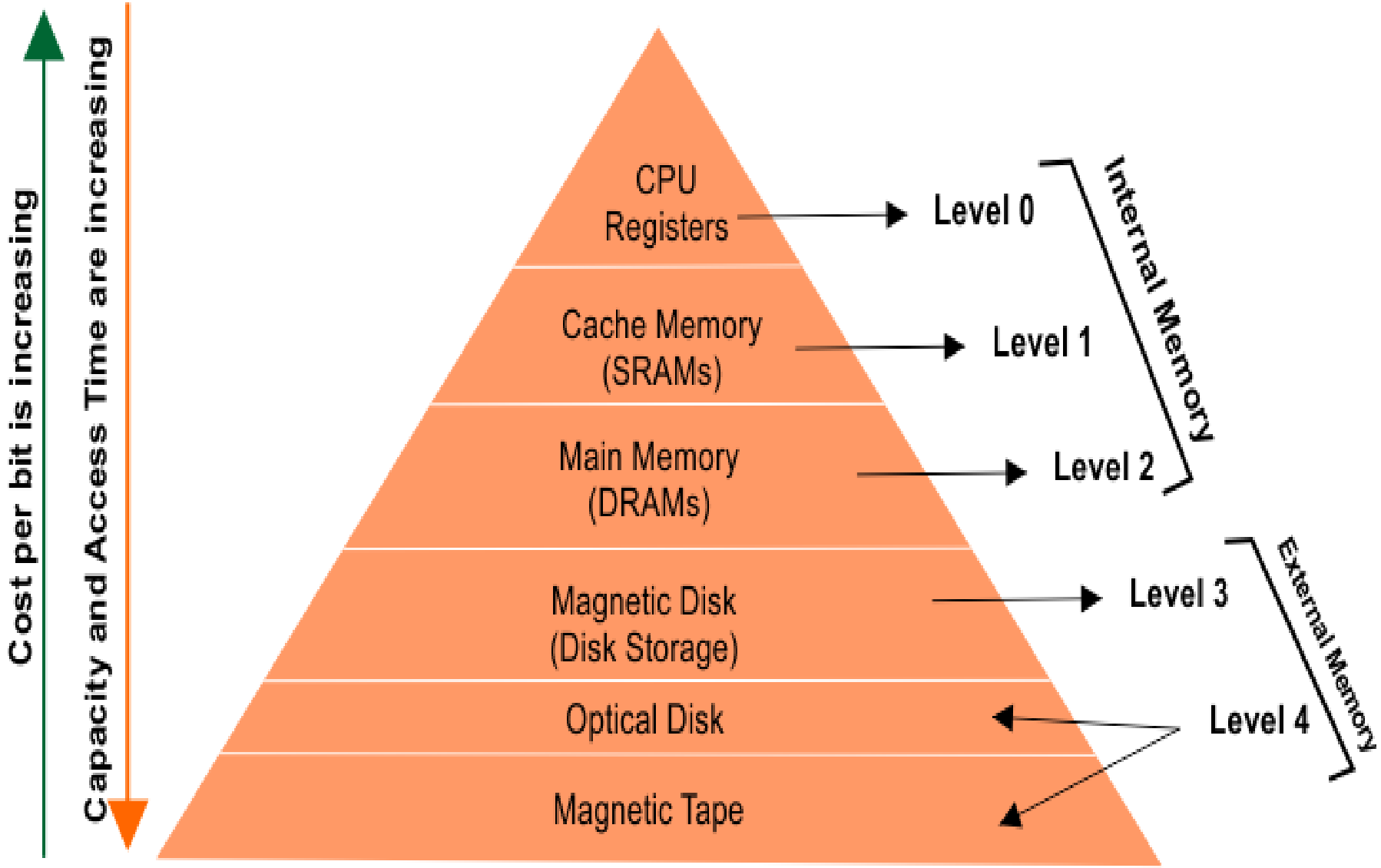
Computer Hardware (1)



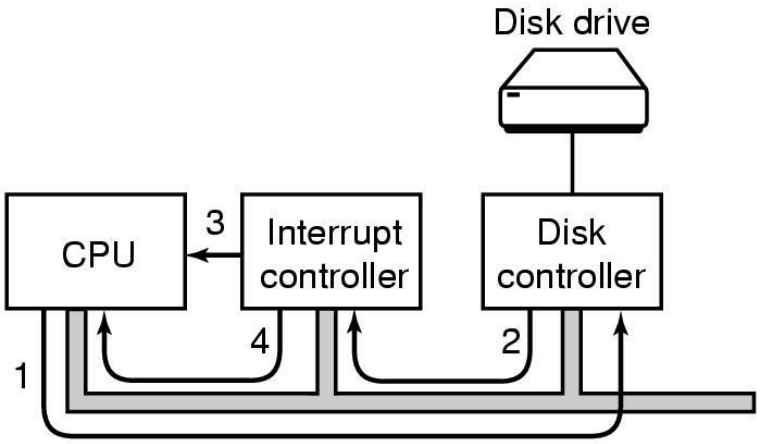
(a) A three-stage pipeline

(b) A superscalar CPU

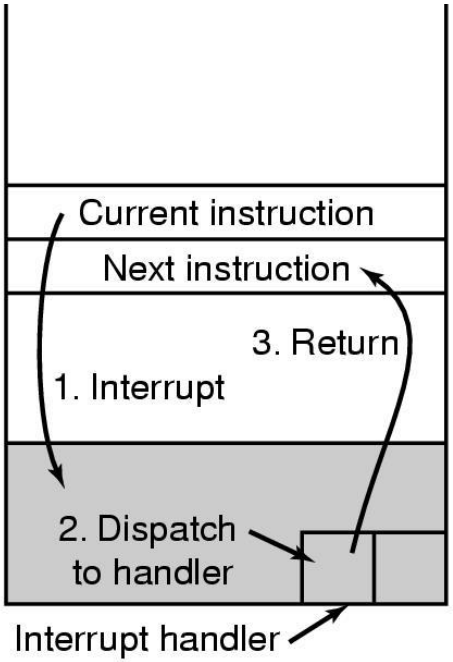
Memory Hierarchy



Computer Hardware (5)



(a)



(b)

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

(b) How the CPU is interrupted

What operating systems do?

- Memory Management
- Process Management
- Device Management
- File Management
- Security
- Control over system performance
- Coordination between other software and users

User View

- The user view depends on the system interface that is used by the users.
- Some systems are designed for a single user to monopolize the resources to maximize the user's task.
- In these cases, the OS is designed primarily for ease of use, with little emphasis on quality and none on resource utilization.

The User View:

- Single User View Point
- Multiple User View Point
- Handheld User View Point

System View

- The OS may also be viewed as just a resource allocator.
- A computer system comprises various sources, such as hardware and software, which must be managed effectively.
- The operating system manages the resources, decides between competing demands, controls the program execution, etc.

The System View:

- Resource Allocation
- Control Program

Operating system operations

- Process Management
- Main Memory Management
- File Management
- I/O System Management
- Secondary Management
- Protection System
- Command-Interpreter System

Process Management

- A *process* is a program in execution. A process needs certain resources, including CPU time, memory, files, and I/O devices, to accomplish its task.
- The operating system is responsible for the following activities in connection with process management.
  - Process creation and deletion.
  - process suspension and resumption.
  - Provision of mechanisms for:
    - process synchronization
    - process communication

Main Memory Management

- Memory is a large array of words or bytes, each with its own address. It is a repository of quickly accessible data shared by the CPU and I/O devices.
- Main memory is a volatile storage device. It loses its contents in the case of system failure.
- The operating system is responsible for the following activities in connections with 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 (allocation policy)
  - Allocate and deallocate memory space as needed.

File Management

- A file is a collection of related information defined by its creator. Commonly, files represent programs (both source and object forms) and data.
- The operating system is responsible for the following activities in connections with file management:
  - File creation and deletion.
  - Directory creation and deletion.
  - Support of primitives for manipulating files and directories.
  - Mapping files onto secondary storage.
  - File backup on stable (nonvolatile) storage media.

I/O System Management

- The I/O system consists of:
  - A buffer-caching system
  - A general device-driver interface
  - Drivers for specific hardware devices
  - SPOOLing (simultaneous peripheral operations online) for *virtual devices* (e.g., a shared printer)

Secondary-Storage Management

- Most modern computer systems use disks as the principle on-line secondary storage medium, for both programs and data.
  - Bulk memory (large size)
  - Low cost
  - Access time in between DRAM and tapes (variable over a small range) (1 msec ~ 60mec per 1Kbyte on a single magnetic disk surface)
- The operating system is responsible for the following activities in connection with disk management:
  - Free space management
  - Storage allocation
  - Disk scheduling

Protection & Security System

- *Protection* refers to a mechanism for controlling access by programs, processes, or users to both system and user resources.
- The protection mechanism must:
  - distinguish between authorized and unauthorized usage.
  - specify the controls to be imposed.
  - provide a means of enforcement.

Types of Operating Systems

- Batch O.S.
- Multi programmed O.S.
- Time Sharing O.S.
- Distributed O.S.
- Virtualization
- Real Time Embedded Systems
- Open source O.S.

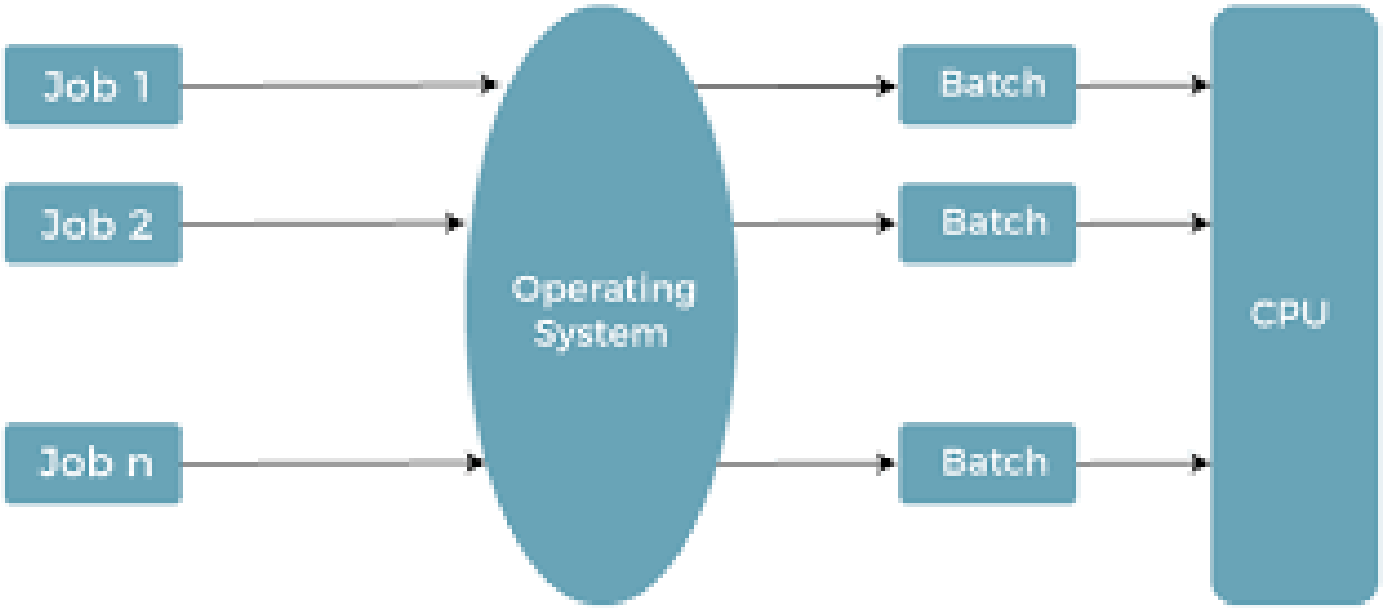
Batch O.S.

- It is the First operating system of the second generation computer.
- Batch operating system took the input on the punch card.
- Each punch card had the different form of data.
- System executed the jobs one by one in batch.
- When one job from the batch executed, then the second job has taken from it and so on.
- The process of placing the jobs in queue for execution is known as spooling.

Batch O.S.

- The batch processing system used where we want to update the data related to any transactions or any record.

- Transactions can be related to any customer orders, receipts, invoices, payments, online transactions, data entry, payroll system, banks etc.



Pros and Cons of Batch OS

VIT-AP UNIVERSITY

Advantages of Batch Operating System:

- It is very difficult to guess or know the time required for any job to complete. Processors of the batch systems know how long the job would be when it is in queue
- Multiple users can share the batch systems
- The idle time for the batch system is very less
- It is easy to manage large work repeatedly in batch systems

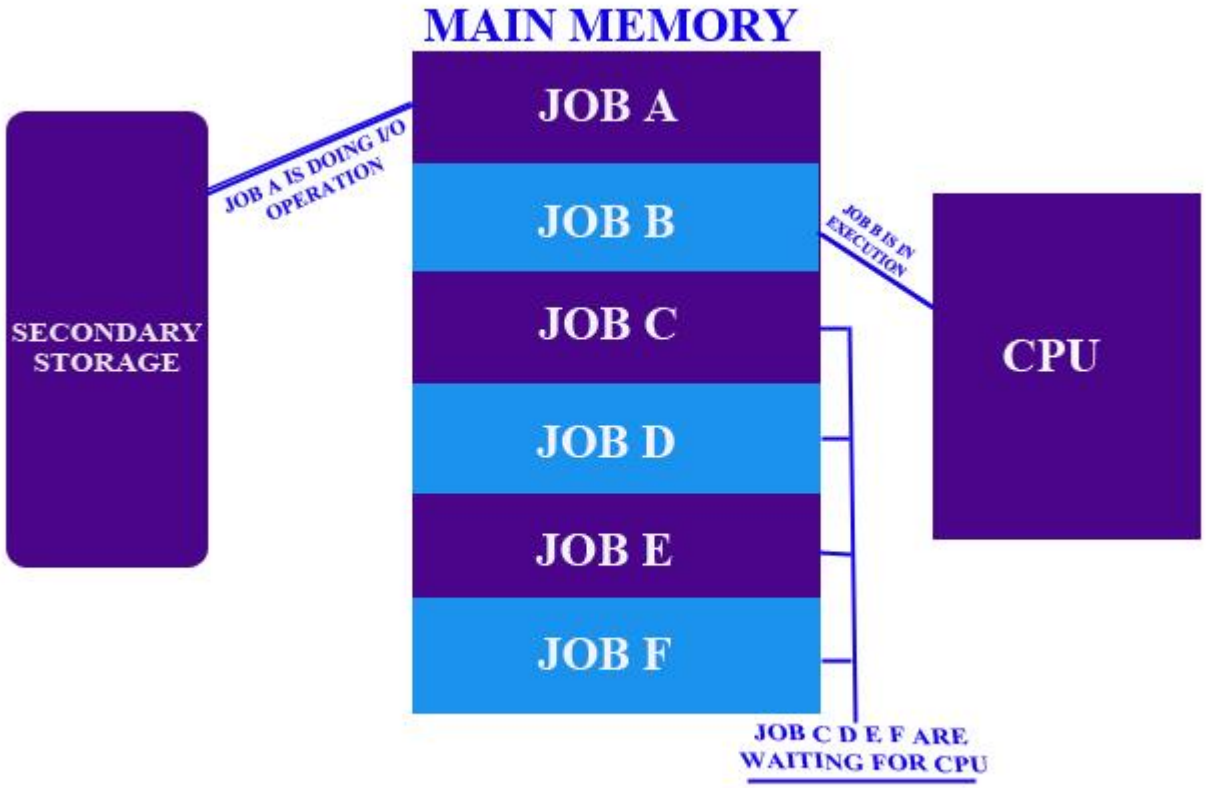
Disadvantages of Batch Operating System:

- The computer operators should be well known with batch systems
- Batch systems are hard to debug
- It is sometimes costly
- The other jobs will have to wait for an unknown time if any job fails

Multi programmed O.S.

- **Multiprogramming OS** is an ability of an operating system that executes more than one program using a single processor machine.
- More than one task or program or jobs are present inside the main memory at one point of time.
- Buffering and spooling can overlap I/O and CPU tasks to improve the system performance but it has some limitations that a single user cannot always keep CPU or I/O busy all the time.

Multi programmed O.S.



Multiprogramming Operating System

OS Features Needed for Multiprogramming

- I/O routine supplied by the system.
- Memory management – the system must allocate the memory to several jobs.
- CPU scheduling – the system must choose among several jobs ready to run.
- Allocation of devices.

- Advantage : CPU Utilization increased due to reduced CPU wait time for I/O operations

  Example : Multiprogram four identical jobs, each with 50% CPU wait time for I/O operations

  Resulting CPU wait time is < 5%

Types of the Multiprogramming Operating System

There are mainly two types of multiprogramming operating systems. These are as follows:

1. **Multitasking Operating System**
2. **Multiuser Operating System**

Multiprogramming Operating System

VIT-AP UNIVERSITY

Multitasking Operating System

- A multitasking operating system enables the execution of two or more programs at the same time. The operating system accomplishes this by shifting each program into and out of memory one at a time. When a program is switched out of memory, it is temporarily saved on disk until it is required again.

Multiuser Operating System

- A multiuser operating system allows many users to share processing time on a powerful central computer from different terminals. The operating system accomplishes this by rapidly switching between terminals, each of which receives a limited amount of processor time on the central computer. The operating system changes among terminals so quickly that each user seems to have continuous access to the central computer. If there are many users on a system like this, the time it takes the central computer to reply can become more obvious.

Pros and Cons of Multiprogramming OS

Advantages

1. It provides less response time.
2. It may help to run various jobs in a single application simultaneously.
3. It helps to optimize the total job throughput of the computer.
4. Various users may use the multiprogramming system at once.
5. Short-time jobs are done quickly in comparison to long-time jobs.
6. It may help to improve turnaround time for short-time tasks.
7. It helps in improving CPU utilization and never gets idle.
8. The resources are utilized smartly.

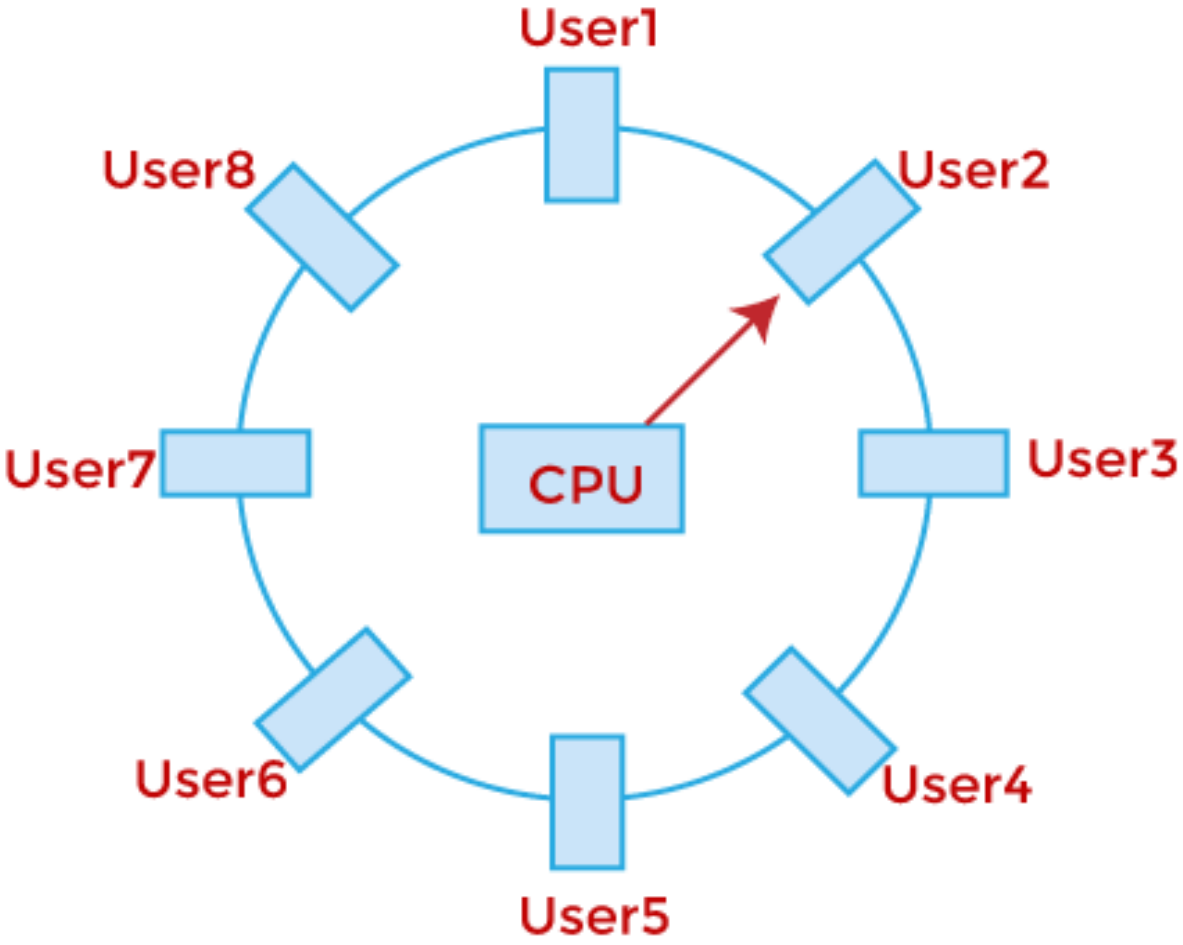
Disadvantages

1. It is highly complicated and sophisticated.
2. The CPU scheduling is required.
3. Memory management is needed in the operating system because all types of tasks are stored in the main memory.
4. The harder task is to handle all processes and tasks.
5. If it has a large number of jobs, then long-term jobs will require a long wait.

Time-Sharing OS

- A **time shared operating system** uses CPU scheduling and multi-programming to provide each with a small portion of a shared computer at once.
- Each user has at least one separate program in memory.
- A program loaded into memory and executes, it performs a short period of time either before completion or to complete I/O.
- This short period of time during which user gets attention of CPU is known as **time slice, time slot or quantum**.
- It is typically of the order of 10 to 100 milliseconds.
- Time shared operating systems are more complex than multiprogrammed operating systems.
- In both, multiple jobs must be kept in memory simultaneously, so the system must have memory management and security.

Time-Sharing OS



Timesharing in case of 8 users

Time-Sharing OS

VIT-AP UNIVERSITY

Advantages of Time Sharing Operating System

- The time-sharing operating system provides effective utilization and sharing of resources.
- This system reduces CPU idle and response time.

Disadvantages of Time Sharing Operating System

- Data transmission rates are very high in comparison to other methods.
- Security and integrity of user programs loaded in memory and data need to be maintained as many users access the system at the same time.

Distributed O.S

- Distributed systems use multiple central processors to serve multiple real-time applications and multiple users. Data processing jobs are distributed among the processors accordingly.
- The processors communicate with one another through various communication lines (such as high-speed buses or telephone lines). These are referred as **loosely coupled systems** or distributed systems.
- Processors in a distributed system may vary in size and function. These processors are referred as sites, nodes, computers, and so on.

Distributed O.S

The advantages of distributed systems are as follows −

- With resource sharing facility, a user at one site may be able to use the resources available at another.
- Speedup the exchange of data with one another via electronic mail.
- If one site fails in a distributed system, the remaining sites can potentially continue operating.
- Better service to the customers.
- Reduction of the load on the host computer.
- Reduction of delays in data processing.

Virtualization

What Is Virtualization ?

- Virtualization is a Technology that transforms hardware into software.
- Virtualization allows to run multiple operating systems as virtual machines.
  - Each copy of an operating system is installed in to a virtual machine.

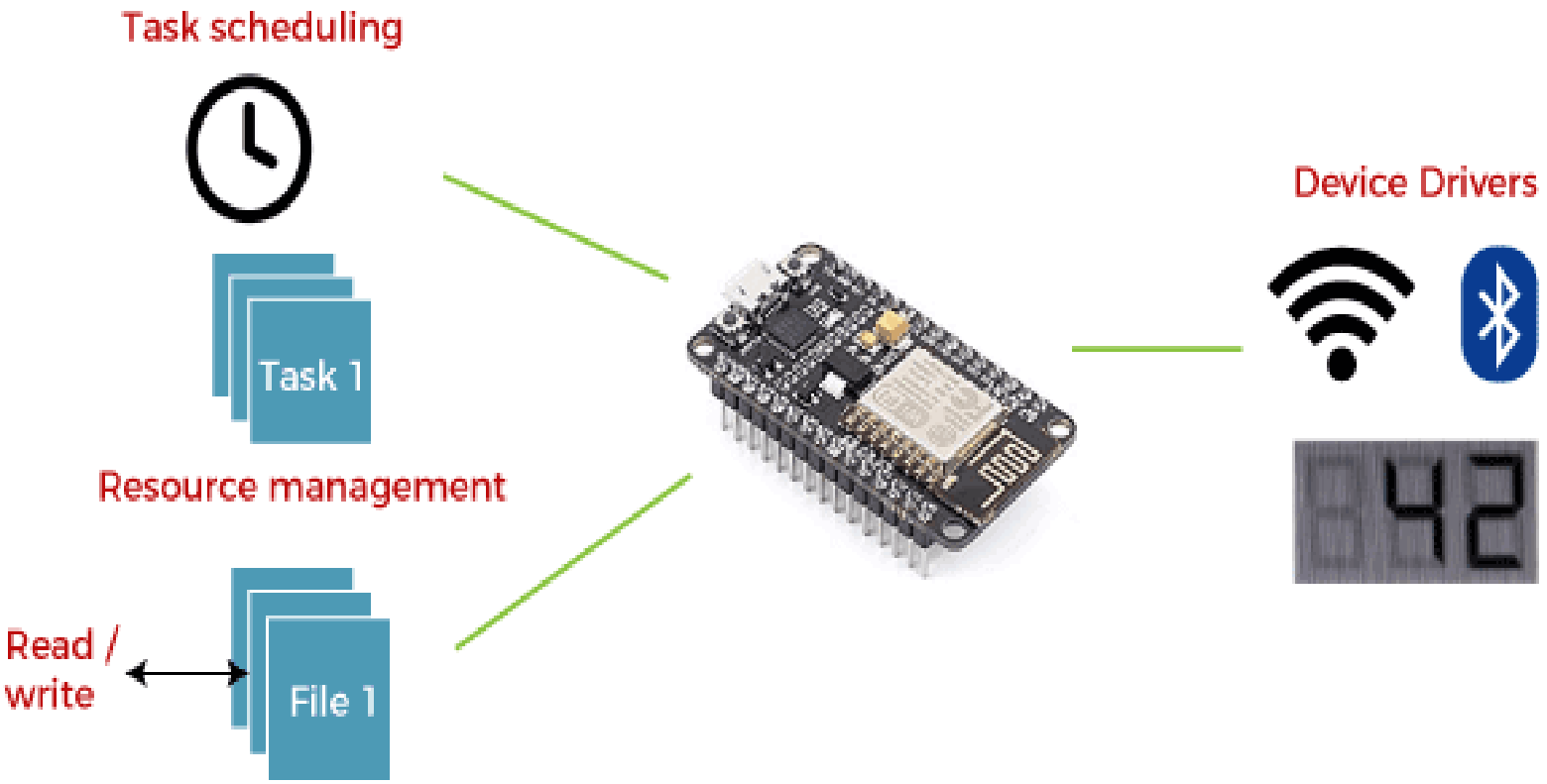
Advantages/Disadvantages of Virtual Machines

- The virtual-machine concept provides complete protection of system resources since each virtual machine is isolated from all other virtual machines. This isolation, however, permits no direct sharing of resources.
- A virtual-machine system is a perfect vehicle for operating-systems research and development. System development is done on the virtual machine, instead of on a physical machine and so does not disrupt normal system operation.
- The virtual machine concept is difficult to implement due to the effort required to provide an *exact* duplicate to the underlying machine.

Real Time Embedded Systems

- In Real-Time Systems, each job carries a certain deadline within which the job is supposed to be completed, otherwise, the huge loss will be there, or even if the result is produced, it will be completely useless.
- Often used as a control device in a dedicated application such as controlling scientific experiments, nuclear reactor, medical imaging systems, industrial control systems, some display systems.
- Well-defined fixed-time constraints.
- Real-Time systems may be either hard or soft real-time

Real Time Embedded Systems



Types of RTOS

- Real-Time systems may be either hard or soft real-time

- Hard real-time:
  - Secondary storage limited or absent, data stored in short term memory, or read-only memory (ROM)
  - Conflicts with time-sharing systems, not supported by general-purpose operating systems.

- Soft real-time
  - Limited utility in industrial control or robotics
  - Useful in applications (multimedia, virtual reality) requiring advanced operating-system features

Open Source OS

- The term "**open source**" refers to computer software or applications where the owners or copyright holders enable the users or third parties to use, see, and edit the product's source code.
- The source code of an open-source OS is publicly visible and editable.
- The usually operating systems such as Apple's iOS, Microsoft's Windows, and Apple's Mac OS are closed operating systems
- The user may modify or change those codes and develop new applications according to the user requirement.
- Some basic examples of the open-source operating systems are Linux, Open Solaris, Free RTOS, Open BDS, Free BSD, Minix, etc.

Open Source OS

Advantages

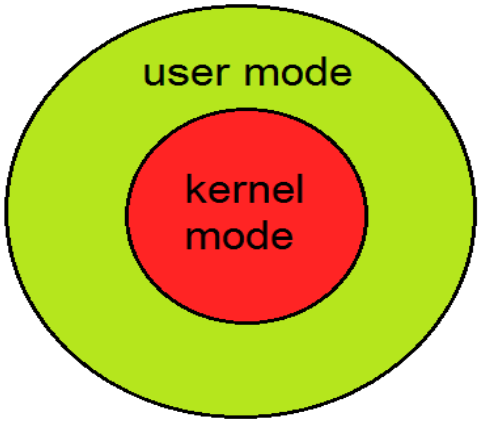
- Reliable and efficient
- Cost-efficient
- Flexibility

Disadvantages

- Complicated
- Security risk
- No support

System Call in OS

Every modern operating system supports these two modes.

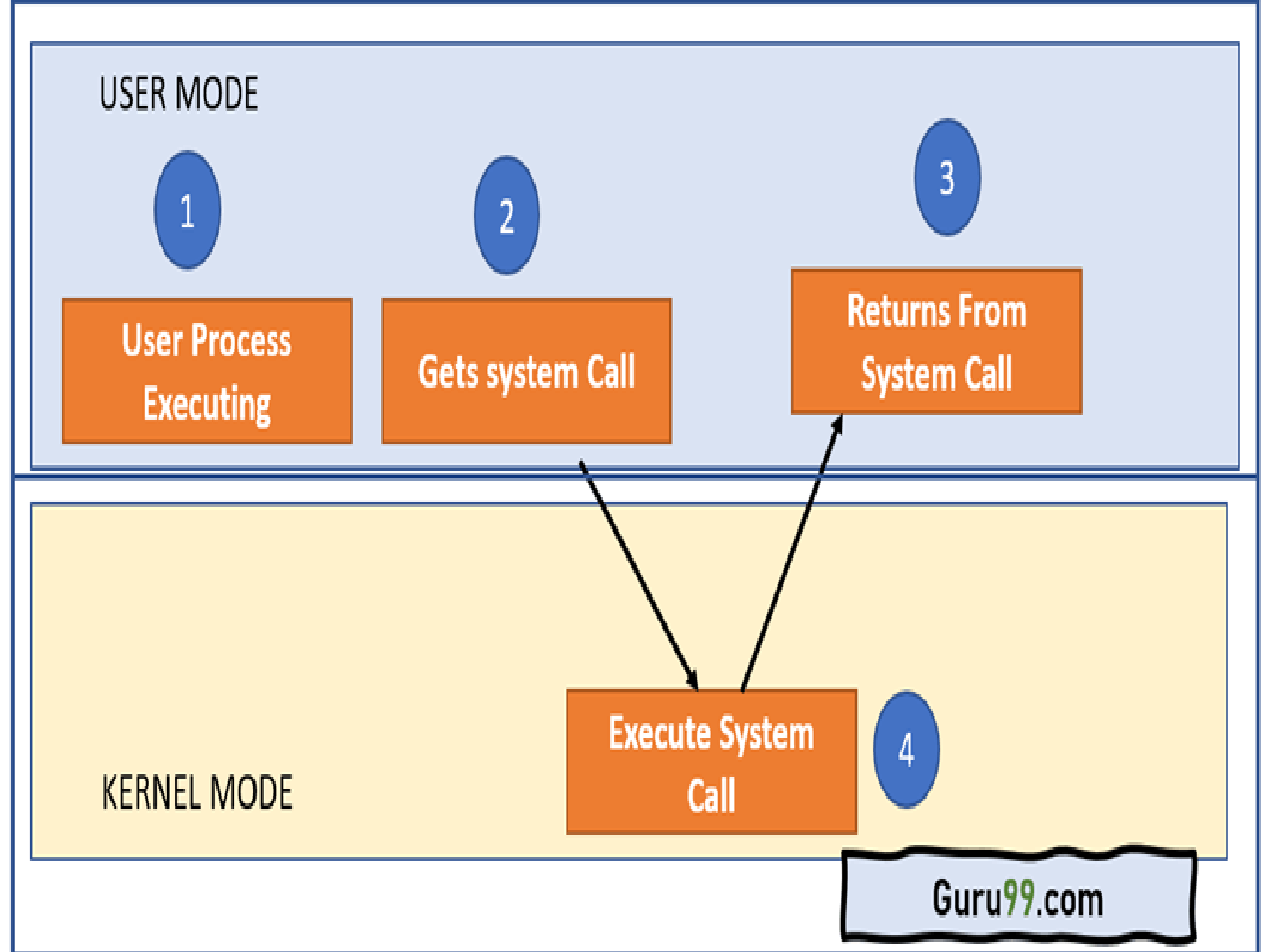


Kernel Mode

• When CPU is in **kernel mode**, the code being executed can access any memory address and any hardware resource.
• Hence kernel mode is a very privileged and powerful mode.
• If a program crashes in kernel mode, the entire system will be halted.

User Mode

• When CPU is in **user mode**, the programs don't have direct access to memory and hardware resources.
• In user mode, if any program crashes, only that particular program is halted.
• That means the system will be in a safe state even if a program in user mode crashes.
• Hence, most programs in an OS run in user mode.


USER MODE
1
User Process Executing
2
Gets system Call
3
Returns From System Call
KERNEL MODE
Execute System Call
4
Guru99.com

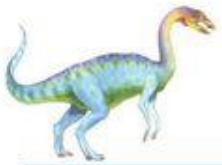
System Call in OS

VIT-AP UNIVERSITY

- When a program in user mode requires access to RAM or a hardware resource, it must ask the kernel to provide access to that resource. This is done via something called a system call.
- When a program makes a system call, the mode is switched from user mode to kernel mode. This is called a context switch.
- Then the kernel provides the resource which the program requested. After that, another context switch happens which results in change of mode from kernel mode back to user mode.

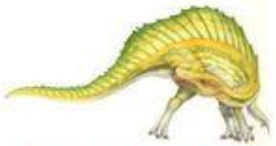
Types of System Calls

- File management
  - create file, delete file
  - open, close file
  - read, write, reposition
  - get and set file attributes
- Device management
  - request device, release device
  - read, write, reposition
  - get device attributes, set device attributes
  - logically attach or detach devices

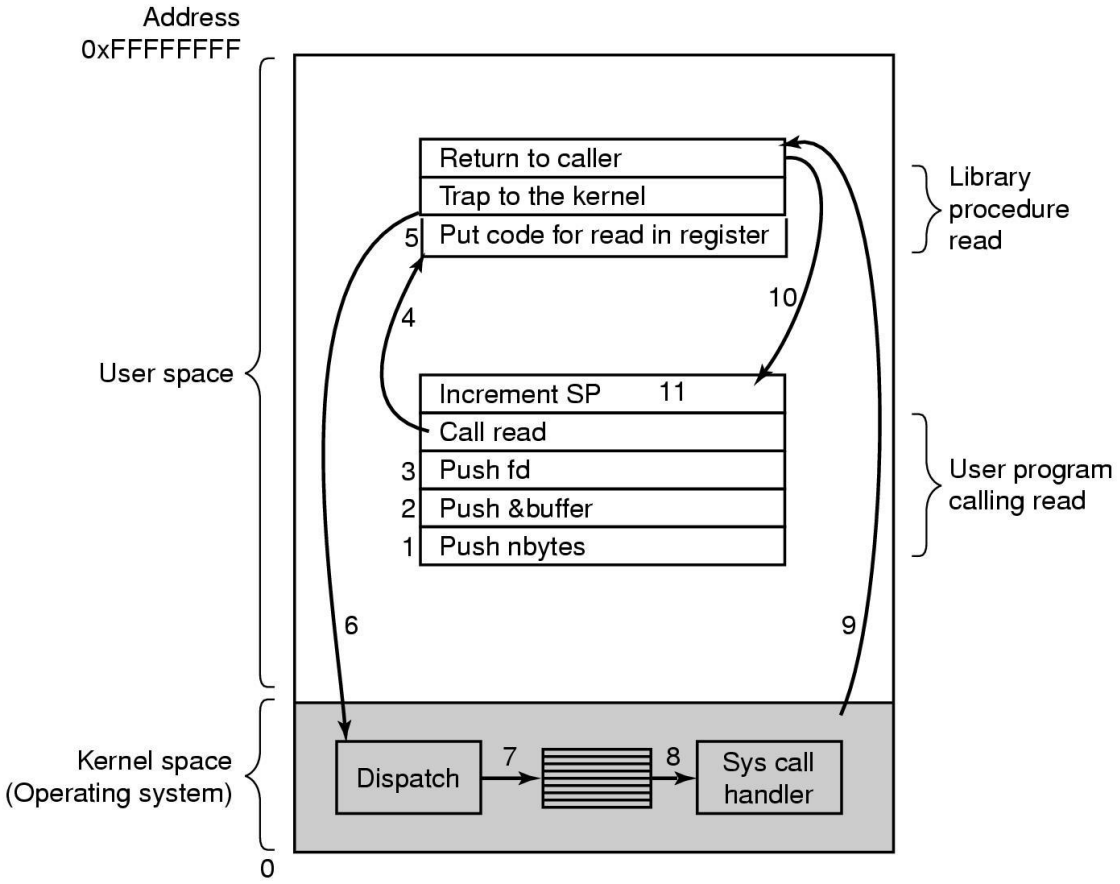


Types of System Calls (Cont.)

- Information maintenance
  - get system data, set system data
    - For example, most systems have a system call to return the current time() and date().
  - get and set process, file, or device attributes
    - For example, calls are also used to reset the process information (**get process attributes**() and **set process attributes**().
- Communications
  - create, delete communication connection between the processes
  - send, receive messages from process to another
  - transfer status information (process ID, process Name,…)
- Devices Protection
  - Control access to resources
  - Get and set permissions
  - Allow and deny user access



Steps in Making a System Call



There are 11 steps in making the system call read (fd, buffer, nbytes)

Some System Calls For Process Management

Process management

Call	Description
pid = fork( )	Create a child process identical to the parent
pid = waitpid(pid, &statloc, options)	Wait for a child to terminate
s = execve(name, argv, environp)	Replace a process' core image
exit(status)	Terminate process execution and return status

The Fork() System Call

1. A call to **fork()** will create a completely separate sub-process which will be exactly the same as the parent.
2. The process that initiates the call to fork is called the parent process.
3. The new process created by fork is called the child process.
4. The child gets a copy of the parent's text and memory space.
5. They do not share the same memory .

Fork System Call

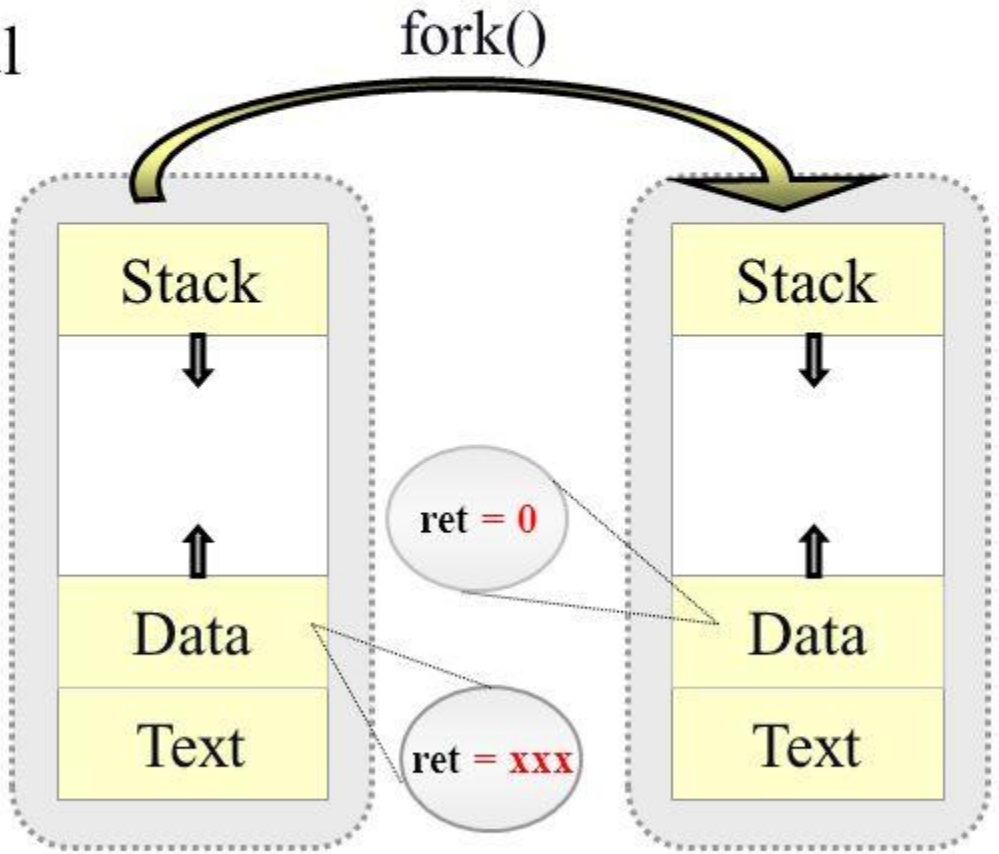
- The child process inherits from parent
  - identical copy of memory
  - CPU registers
  - all files that have been opened by the parent

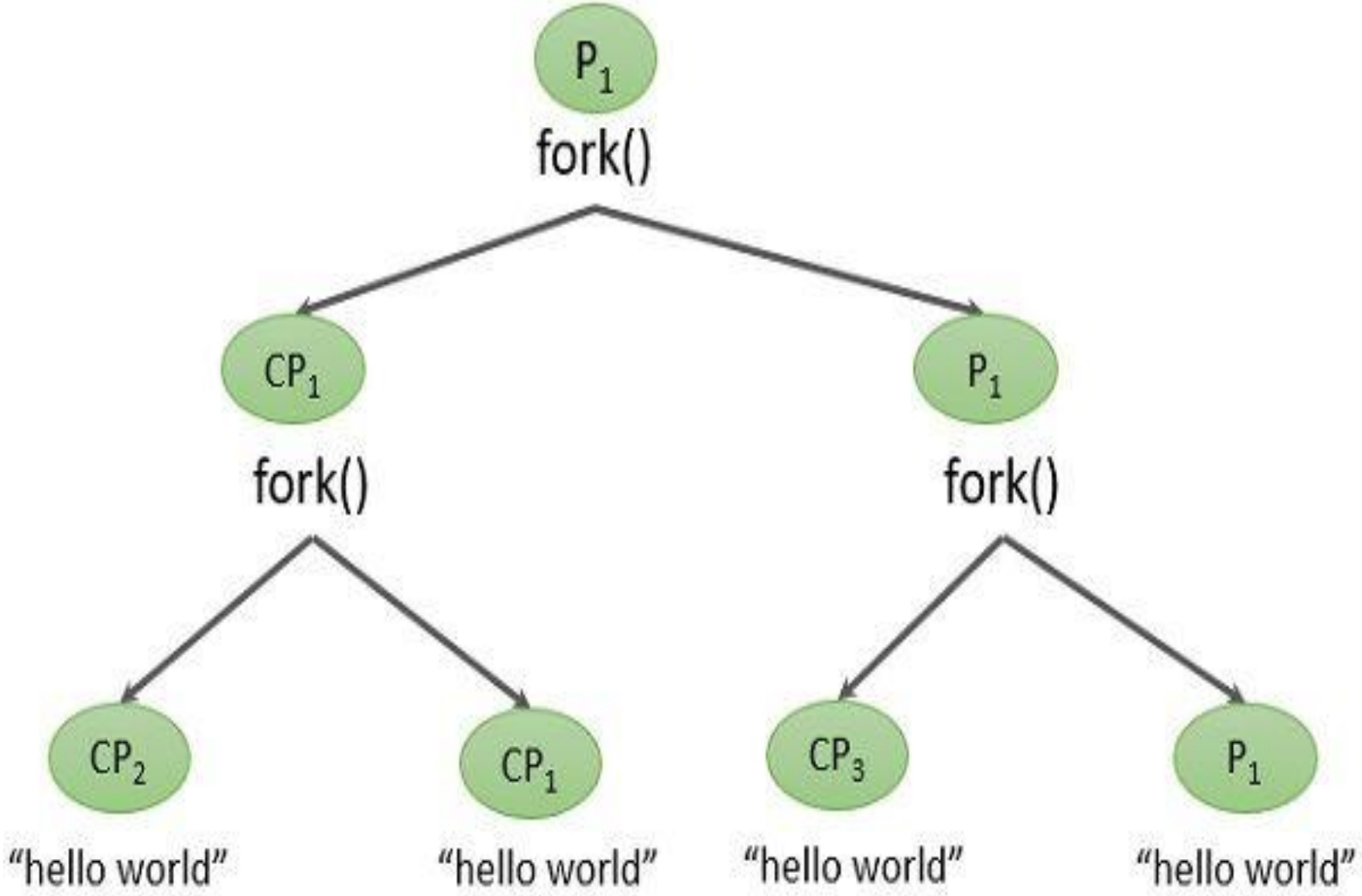
- Execution proceeds concurrently with the instruction following the fork system call

- The execution context (PCB) for the child process is a copy of the parent's context at the time of the call

Fork System Call

- Current process split into 2 processes: parent, child
- Returns -1 if unsuccessful
- Returns 0 in the child
- Returns the child's identifier in the parent




P1
fork()
CP1
P1
fork()
fork()
CP2
CP1
CP3
P1
"hello world"
"hello world"
"hello world"
"hello world"

C code for fork() system call

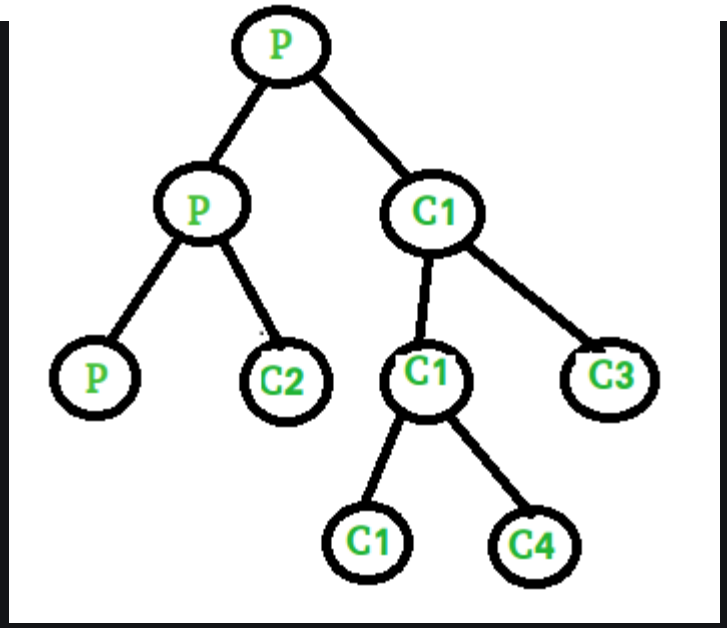
```
main(){
        fork();
        fork();
         printf("\n hello world");
       }
```

```c
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
int main(){
fork();
printf("\n Process id = %d", getpid());
return 0;
}
```

```
Process id = 273
Process id = 274
```

Examples

```
#include <stdio.h>
#include <unistd.h>
int main() {
        if (fork() && fork());
        fork();
        printf(“Hello\n");
        return 0;
}
```



1. It will create two process one parent P (has process ID of child process)and other is child C1 (process ID = 0).
2. In if statement we used OR operator(||) and in this case second condition is evaluated when first condition is false.
3. Parent process P will return positive integer so it directly execute statement and create two more processes (one parent P and other is child C2). Child process C1 will return 0 so it checks for second condition and second condition again create two more processes(one parent C1 and other is child C3).
4. C1 return positive integer so it will further create two more processes (on parent C1 and other is child C4). Child C3 return 0 so it will directly print 1.