# Computers & Operating Systems

## LS-01. OS Introduction (Evolution, Definition, Types).

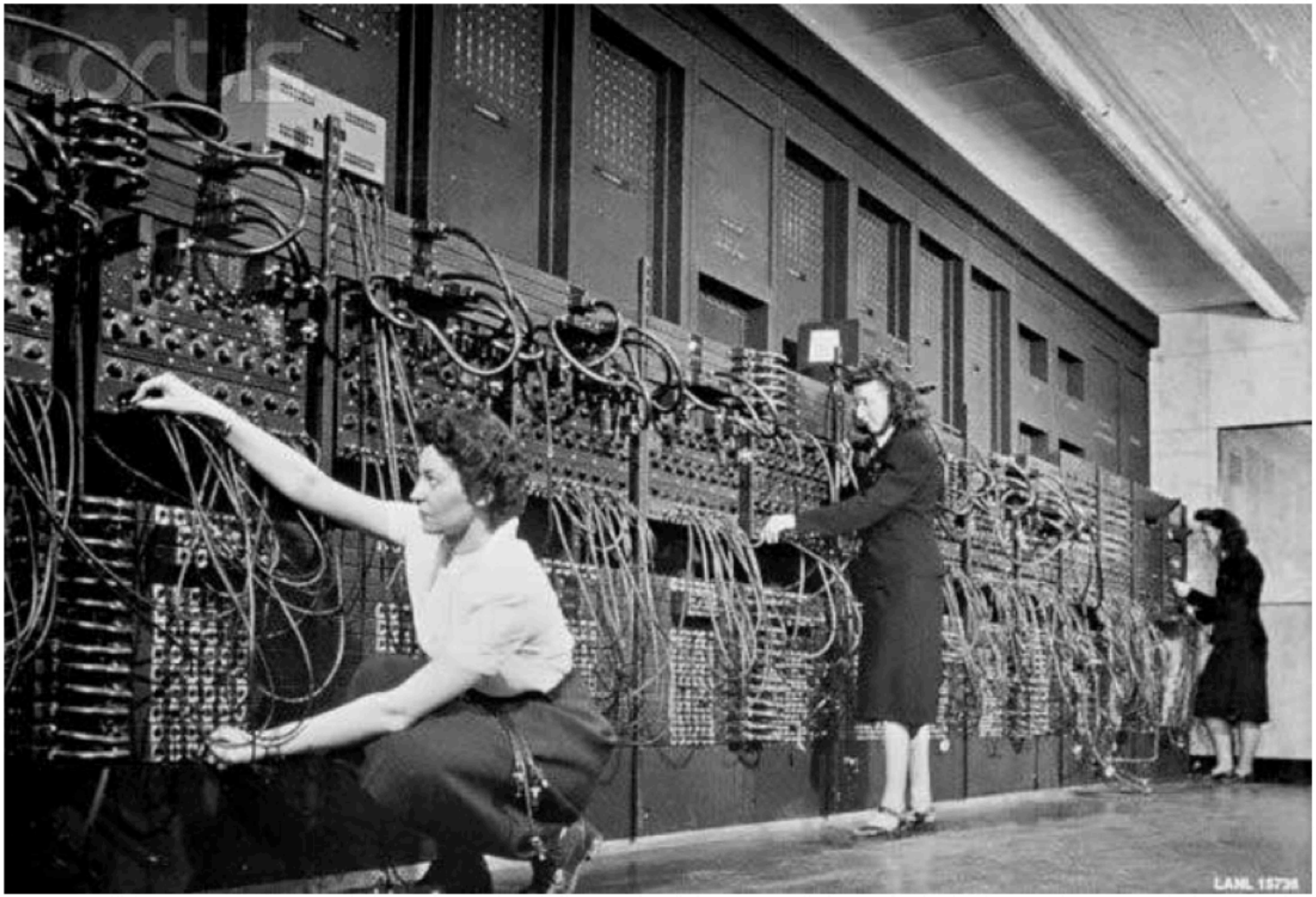
# 1. Evolution of Computers and Operating System

# There were no operating systems

"Preparing ENIAC for a series of runs was an incredibly involved process. First, detailed instructions had to be written defining the problem and a procedure for solving it. These instructions were programmed by adjusting switches manually and inserting thousands of cables into as many as forty large plug boards. A team of five operators might work several days on the external wiring and many more days searching for errors and correcting them."

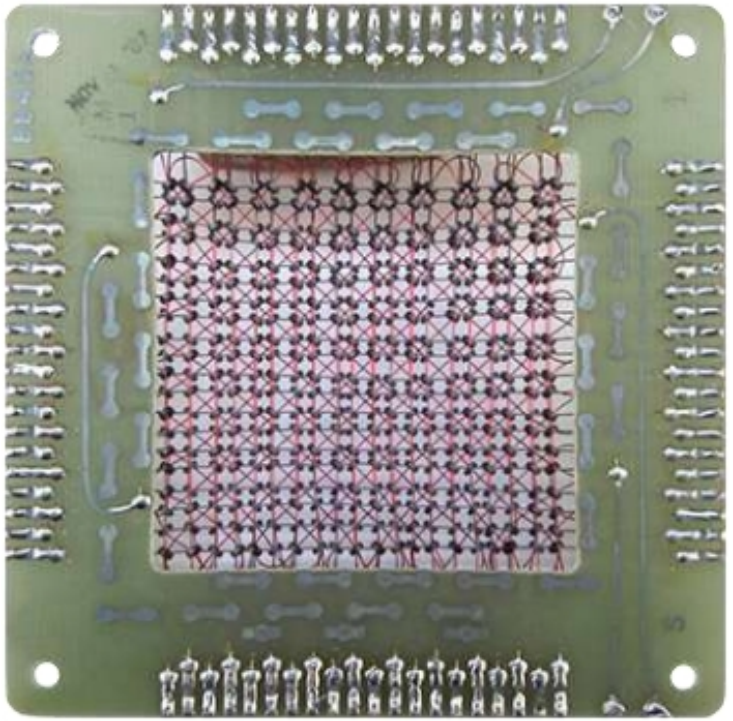
— *Breakthrough to the Computer Age, Harry Wulforst, Charles Scribner's & Sons Pub., 1982*

# Programming the ENIAC



# Ferrite Core Memory

- Fast, random-access memory
  - Non-volatile
  - Write-after-read to preserve bit
- First used in MIT's Whirlwind-1
  - 1953
  - 32×32×16 bits
- Used through ~1980



# Late 1940s – 1950s

- Stored program concept: reload a program
- Reusable code ("subroutines")
- IBM SHARE (Society to Help Alleviate Redundant Effort)
- The OS emerges
  - I/O Control System (IOCS): Common I/O routines for device access
    - Precursor to *device drivers*
  - Batch systems (1956)
    - "Control cards" after a deck of punched cards to terminate one job and prepare for the next
      - Programmatic transition to reduce overhead of starting new jobs
    - Branch to a location in the OS that would cause the next program to get loaded and run
    - Job control languages to define resource needs

# The Interrupt

- 1951 – UNIVAC I: exception handling
  - Transfer control on arithmetic overflow

- 1956 – UNIVAC 1103A
  - Hardware interrupt support
  - Interrupt writes PC to memory location & transfers control to an Interrupt Service Routine

# 1960s

- Goal: improve throughput
  - Use every possible second of CPU time
- Multiprogramming
  - Keep several programs in memory at once; switch between them
  - Works because of the speed mismatch between I/O and CPU
- 1961: Time sharing: preemption
  - CTSS (Compatible Time-Sharing System): Process scheduling
- 1962: the System Call (Atlas I Computer, Manchester)
  - Privileged & unprivileged modes

# 1960s

- Interactive access
- User accounts and passwords
- Direct storage access (file systems)
- Transaction processing systems (SABRE)
  - IBM & American Airlines

# 1960s

- 1961: DEC PDP-1 – first minicomputer ($125,000+)
- 1964: IBM System/360
  - PCP/360: sequential jobs (batch)
  - MFT: Multiple job system, fixed number of tasks
  - MVT: Multiple jobs, variable number of tasks (direct memory)
  - Direct Address Translation
    (precursor of virtual memory & the Memory Management Unit)
  - *Channels*: specialized processors for transferring data between main memory and an I/O device
    (*precursor of DMA*)

# December 9, 1968: The Mother of All Demos

- Douglas Engelbart

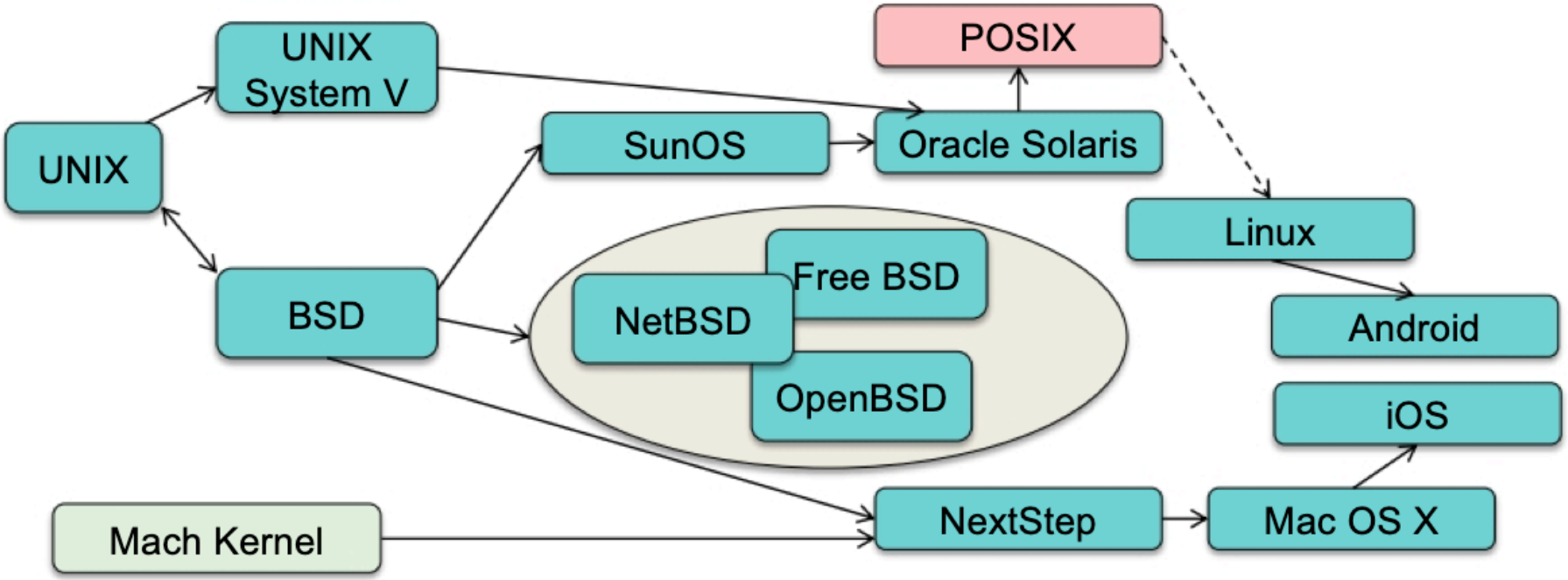
  Stanford Research Institute (SRI), Augmentation Research Center
- Presented at Fall Joint Computer Conference
- Introduced
  - Computer mouse
  - Windows
  - Video conferencing
  - WYSIWYG word processing (with cut & copy) & embedded objects
  - Collaborative editing
  - Version control
  - Hypertext


# 1964-1970: Multics

- Memory mirrored onto the disk and available via the file system
- Dynamic linking for code & data segments
- Interprocess communication via shared segments
- Multiprocessor support
- On-line reconfiguration of system HW without downtime
- Hierarchical security model using protection rings
- Hierarchical file system with arbitrary file names
- Symbolic links
- Command processor not part of the operating system
- Written in a high-level language
  – EPL, a subset of PL/1 with extensions
- I/O redirection to files and programs ("pipes")

# Late 1960s – 1970s

- 1970s: UNIX
  - Portable operating system
  - Written in an efficient high level language (C)
  - The UNIX programming environment: shell, pipes, “tools”



# 1972: Virtual Machines

- 1972: Virtual Machines (VM/370)
  - Run multiple operating systems on one machine
  - Each “machine” presents the same System/370 architecture
  - Hypervisor
    - Control program that runs on the physical hardware and creates the virtual machines
    - Intercepts & interprets all I/O operations and privileged instructions
    - Partitions memory

# 1973: Xerox Alto

- A *personal* computer (dedicated to one user)
  - Desktop UI metaphor and a mouse
  - Inspired by Douglas Englebart's On-Line System
- Specs
  - TI bit-slice processor
  - 128-512 KB RAM
  - 2.5MB removable hard disk
  - Ethernet
  - B&W CRT
  - 3-button mouse
  - Small fridge-sized cabinet
- Inspired the Mac & Microsoft Windows

# 1971 - 1975

- Microprocessors emerge
  - Intel 4004 → 8008 → 8080
  - Zilog Z-80, MOS Technology 6502, Motorola 6800, 6809
  - CP/M: dominant OS for 8080 family of machines
    - CCP: command interpreter
    - BDOS: file operations, printing, and console I/O
    - BIOS: character I/O, disk sector read/write

Initial cost
8080: $360
6502: $25

# Late 1970s: Home PCs

- 1975: Early PCs – targeted at hobbyists
  - Connect your own teletype or use a front panel
  - Build it from a kit
  - Write your own OS drivers


- 1977: Ready-to-use personal computers
  - Apple II
  - Commodore PET
  - Radio Shack TRS-80 Model I
  - Followed by:
    - Atari 400, Atari 800, TI-99/4A, Commodore Vic 20, Commodore 64, …

# 1980s

- 1981: IBM PC
  - Open architecture; Microsoft OS
  - Only proprietary component was the BIOS
- 1982: BIOS was reverse engineered
  - PC clones (Compaq, Columbia, Dell, HP, …)
- 1984: Apple Macintosh

# 1980s

- Client-server networking
  - Personal workstations
  - Network file systems

- 1985: Intel 80386
  - Virtual memory with paging
  - Virtual 8086 mode for multiple legacy programs

# 1990s

- 1990: Windows 3.0
  - Takes advantage of virtual memory provided by 80386
- 1993: Window NT
  - New OS built from scratch
- Open Source Operating Systems
  - Linux, FreeBSD, NetBSD, OpenBSD
- 1995: Windows 95
  - Built-in Internet support (networking usually via modem)

# 1990s

- PCI bus: Plug & Play hardware
  – Adding hardware becomes easy
- Laptops become mainstream: *power usage is important*
- 1993: NCSA Mosaic – the web browser
- Network PC, thin clients
  – Failed … but resurrected with the Google Chromebook

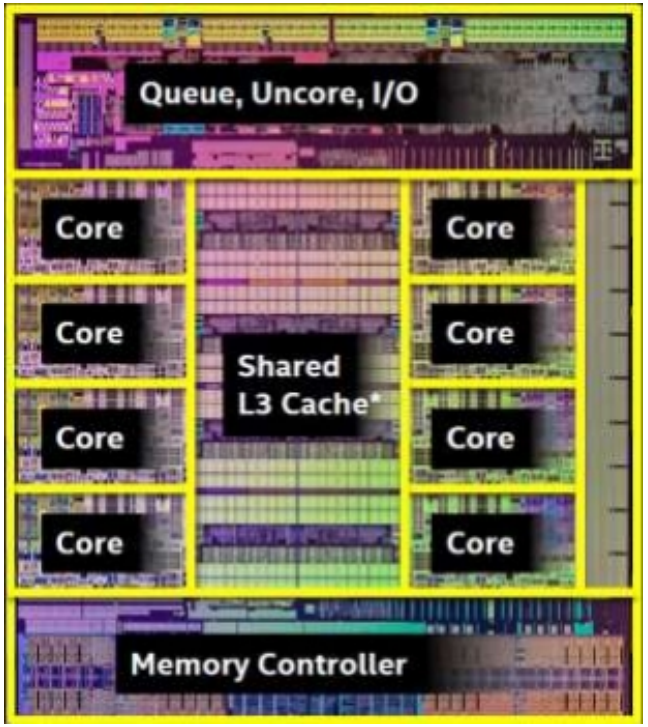
# 2000s

- PC-based machine virtualization
  - Virtualization support added by Intel & AMD (2006)
  - Virtual machine migration
- Cloud computing, on-demand data centers
- Security
  - Hardware authentication, Storage encryption, digital rights management: Trusted Platform Module
  - Personal firewalls
  - Address space layout randomization

# Multi-core Architectures

- 2005: Intel Introduces dual core Pentium D
  - 90nm process – Pentium Extreme Edition
  - 230 million transistors
  - 2 MB L2 cache

- Late 2014: Intel Haswell-E i7-5960X
  - 8 Cores
  - Hyperthreading
  - 2.6 Billion 22nm tri-gate 3-D transistors
  - 2133 MHz DDR4 memory interface
  - 20 MB L3 cache (shared across cores)



# 2000s

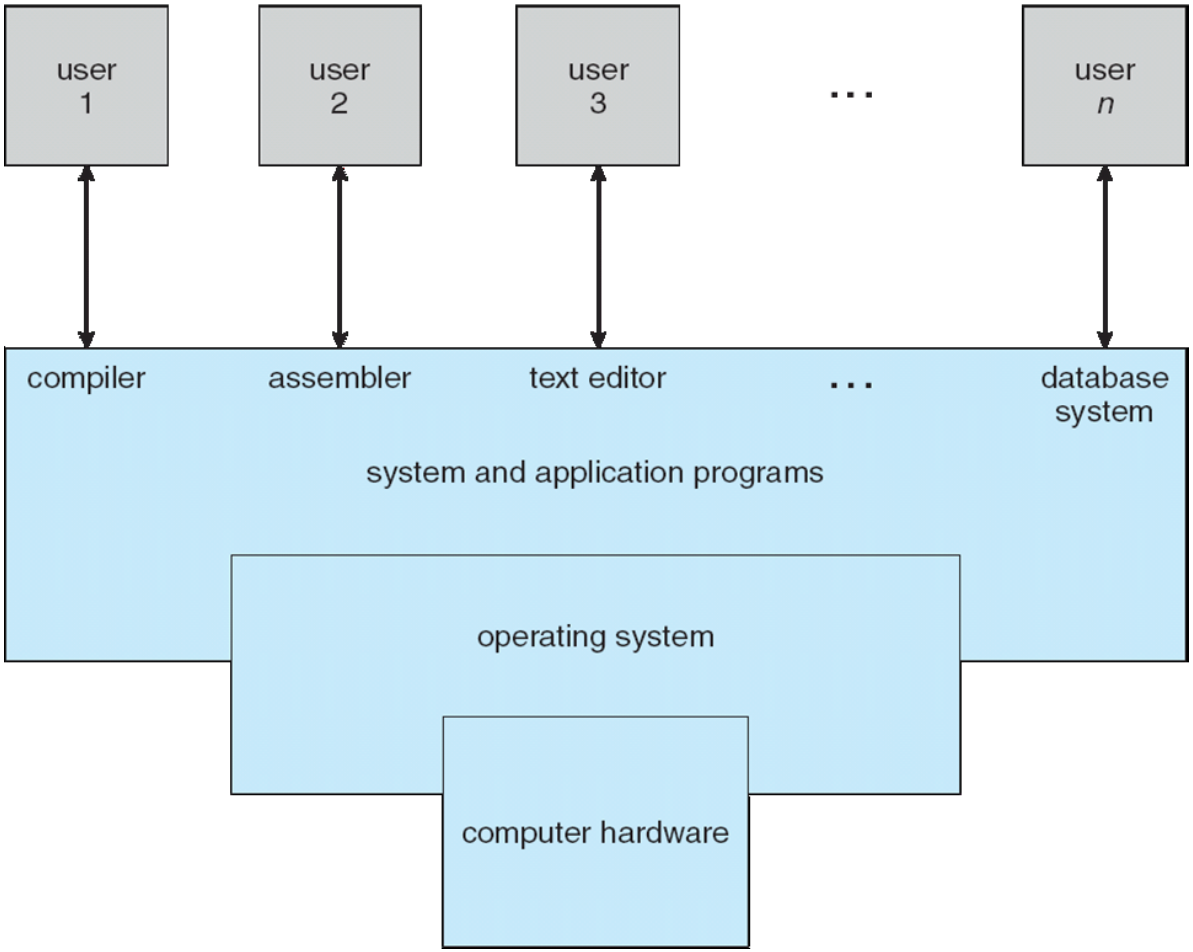
- Focus on mobility
  - Tablets
    - 1991 AT&T EO Personal Communicator
    - 1999: Microsoft Tablet PC
    - 2010: Apple iPad
  - PDAs → smartphones
    - iOS, Android, BlackBerry OS, Windows Mobile
- Increased focus on embedded systems
  - Machine-to-machine (M2M), Internet of Things, Arduino, …
- Cloud computing
  - Large scale data centers, reconfigurable virtual machines

# 2. The Operating System Definition

# The Computer System

- Computer system can be divided into four components:
  - Hardware – provides basic computing resources
    - CPU, memory, I/O devices
  - Operating system
    - Controls and coordinates use of hardware among various applications and users
  - Application programs – define the ways in which the system resources are used to solve the computing problems of the users
    - Word processors, compilers, web browsers, database systems, video games
  - Users
    - People, machines, other computers

# Four Components of a Computer System



# What is an operating system?

- The first run program?
- A program that lets you run other programs?
- A program that provides controlled access to resources?
  - CPU
  - Memory
  - Display, keyboard, mouse
  - Persistent storage
  - Network

    This includes: naming, sharing, protection, communication

# Operating System Definition

- No universally accepted definition.
  - 1. May be, OS is a resource allocator.
    - Manages all resources,
    - Decides between conflicting requests for efficient and fair resource use.
  - 2. Or may be, OS is a control program.
    - Controls execution of programs to prevent errors and improper use of the computer.
    - This includes: naming, sharing, protection, communication
  - 3. Or may be, “Everything a vendor ships when you order an operating system” is good approximation, but varies wildly.
  - 4. My definition - OS is a software package that performs the following special tasks and functions:

# Operating System Management Tasks

1. Processor management which involves putting the tasks into order and pairing them into manageable size before they go to the CPU.
2. Memory management which coordinates data to and from RAM (random-access memory) and determines the necessity for virtual memory.
3. Device management which provides interface between connected devices.
4. Storage management which directs permanent data storage.
5. Application which allows standard communication between software and your computer.
6. User interface which allows you to communicate with your computer.

# Operating System Functions

1. It boots the computer
2. It performs basic computer tasks e.g. managing the various peripheral devices e.g. mouse, keyboard
3. It provides a user interface, e.g. command line, graphical user interface (GUI)
4. It handles system resources such as computer's memory and sharing of the central processing unit (CPU) time by various applications or peripheral devices.
5. It provides file management which refers to the way that the operating system manipulates, stores, retrieves and saves data.
6. Error Handling is done by the operating system. It takes preventive measures whenever required to avoid errors.

# 3. The Operating System Types

# Types of Operating Systems

By Functionality:

- Zero User – Single Tasking OS (Embedded)
- Single User – Single Tasking OS (MS DOS)
- Single User – Multitasking OS (Windows 98)
- Multi-User – Multitasking OS (UNIX, Linux, Windows Server

By Working process:

- Interactive
- Not Interactive

By Architecture:

- Simple Batch System
- Multiprogramming Batch System
- Multiprocessor System
- Distributed Operating System
- Realtime Operating System

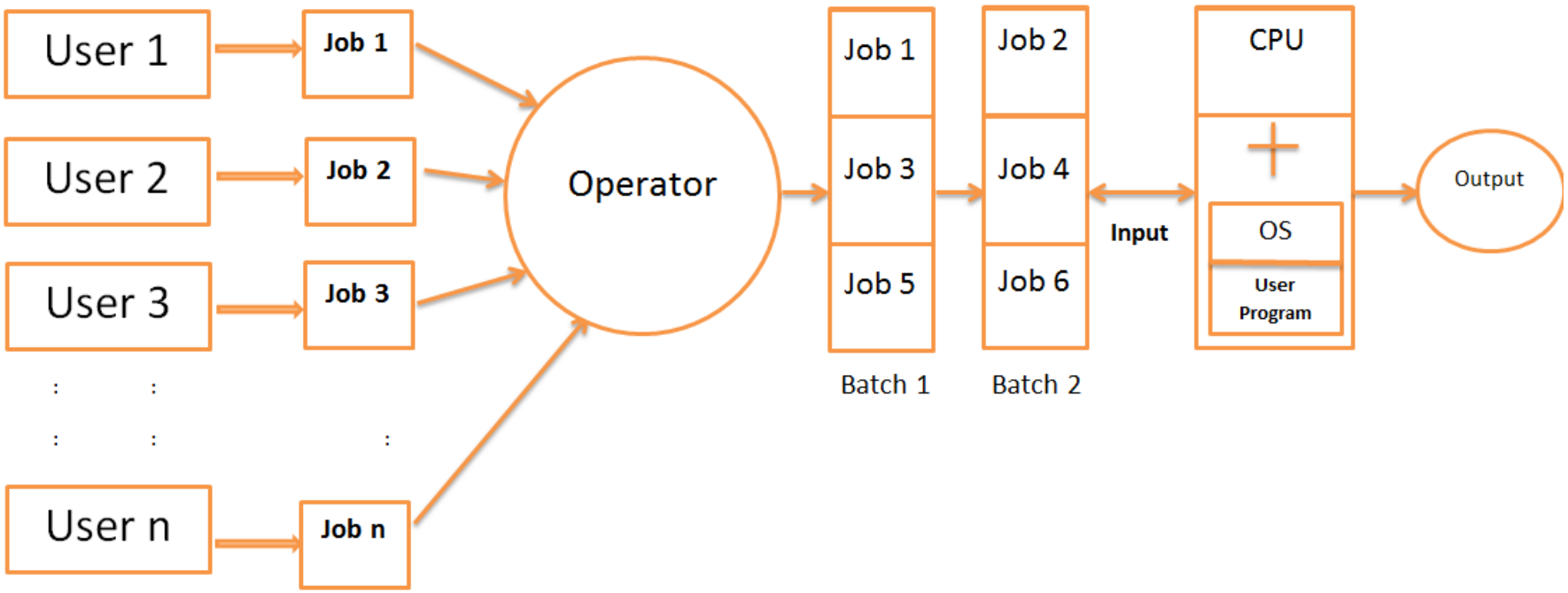
# Simple Batch System

- In this type of system, there is no direct interaction between user and the computer.
- The user has to submit a job (written on cards or tape) to a computer operator.
- Then computer operator places a batch of several jobs on an input device.
- Jobs are batched together by type of languages and requirement.
- Then a special program, the monitor, manages the execution of each program in the batch.
- The monitor is always in the main memory and available for execution.

Following are some disadvantages of this type of system :

- a. Zero interaction between user and computer.
- b. No mechanism to prioritize processes.

# Simple Batch System



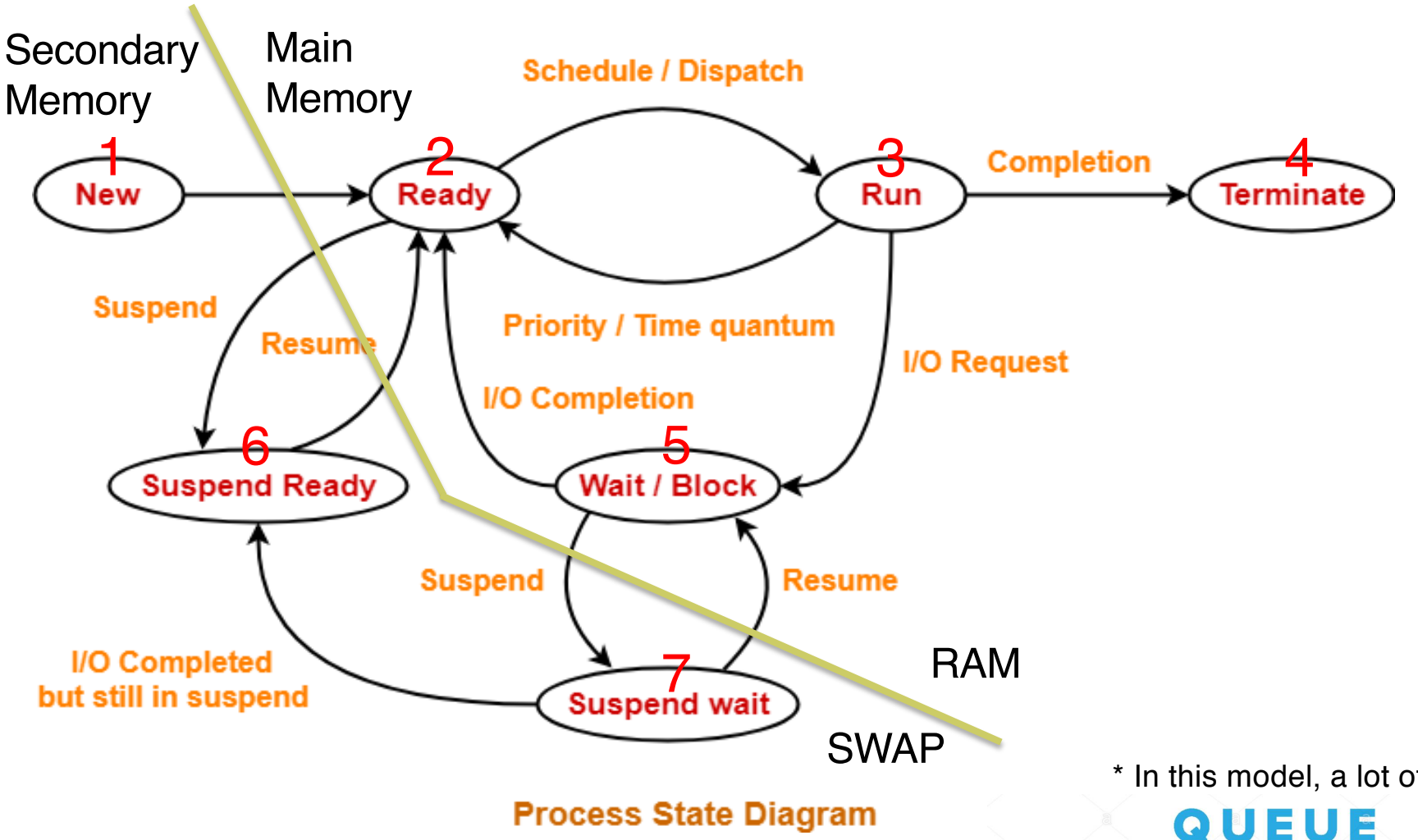
# Multiprogramming Batch System

- In this the operating system, picks and begins to execute one job from memory.
- Once this job needs an 1/0 operation operating system switches to another job (CPU and OS always busy).
- Jobs in the memory are always less than the number of jobs on disk(Job Pool).
- If several jobs are ready to run at the same time, then system chooses which one to run (CPU Scheduling).
- In Non-multiprogrammed system, there are moments when CPU sits idle and does not do any work.
- In Multiprogramming system, CPU will never be idle and keeps on processing.

Time-Sharing Systems are very similar to Multiprogramming batch systems. In fact time sharing systems are an extension of multiprogramming systems.

- In time sharing systems the prime focus is on minimizing the response time, while in multiprogramming the prime focus is to maximize the CPU usage.

# Multiprogramming Batch System




Process State Diagram

* In this model, a lot of the

# Multiprogramming Batch System

- **1. New State -** A process is said to be in new state (1) when a program present in the secondary memory is initiated for execution.
- **2. Ready State -** A process moves from new state (1) to queue (очередь) of ready state (2) after it is loaded into the main memory and is ready for execution. In ready state, the process waits for its execution by the processor. In multiprogramming environment, many processes may be present in the ready state.
- **3. Run State -** A process moves from ready state (2) to run state (3) after it is assigned the CPU for execution.
- **4. Terminate State -** A process moves from run state (3) to terminate state (4) after its execution is completed. After entering the terminate state, context of the process (process descriptor) is deleted by the operating system.
- **5. Block Or Wait State -** A process moves from run state (3) to block or wait state (5) if it requires an I/O operation or some blocked resource during its execution. After the I/O operation gets completed or resource becomes available, the process moves to the ready state (2).
- **6. Suspend Ready State -** A process moves from ready state (2) to suspend ready state (6) if a process with higher priority has to be executed but the **main memory is full** (Memory Swapping).

  Moving a process with lower priority from ready state to suspend ready state clear a room for higher priority process in the ready state.

  The process remains in the suspend ready state until the main memory becomes available. When main memory becomes available, the process is brought back to the ready state (2).
- **7. Suspend Wait State -** A process moves from wait state (5) to suspend wait state (7) if a process with higher priority has to be executed but the main memory is full.

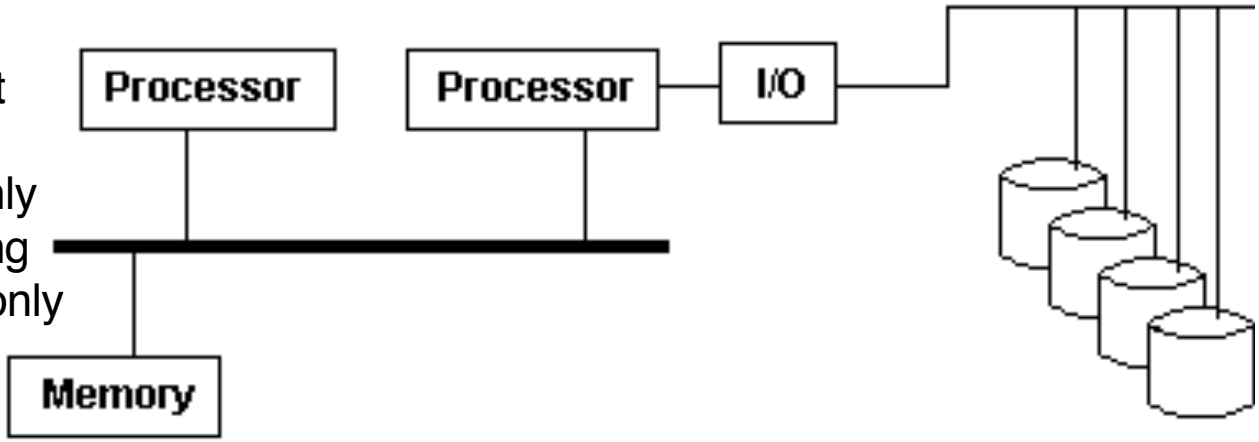
  Moving a process with lower priority from wait state to suspend wait state clear a room for higher priority process in the ready state.

  After the resource becomes available, the process is moved to the suspend ready state (6). After main memory becomes available, the process is moved to the ready state.

# Multiprocessor System

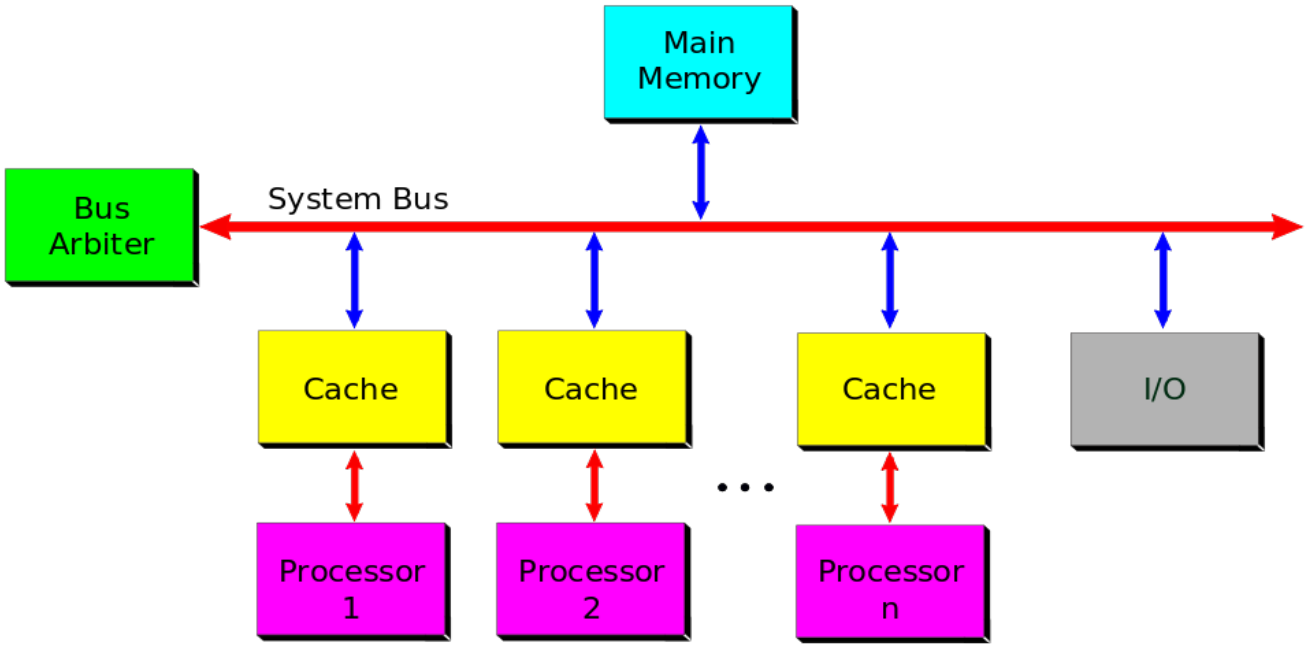
- A multiprocessor system consists of several processors that share a common physical memory and all processors operate under single OS.
- General advantage of this type of system - **Enhanced performance**:
  - Execution of several tasks by different processors concurrently, increases the system's throughput without speeding up the execution of a single task.
  - If possible, system divides task into many subtasks and then these subtasks can be executed in parallel in different processors. Thereby speeding up the execution of single tasks.
- Old **Asymmetric multiprocessing** (**AMP** or **ASMP**) system is a multiprocessor computer system where not all of the multiple interconnected central processing units (CPUs) are treated equally.

For example, a system might allow (either at the hardware or operating system level) only one CPU to execute operating system code or might allow only one CPU to perform I/O operations.



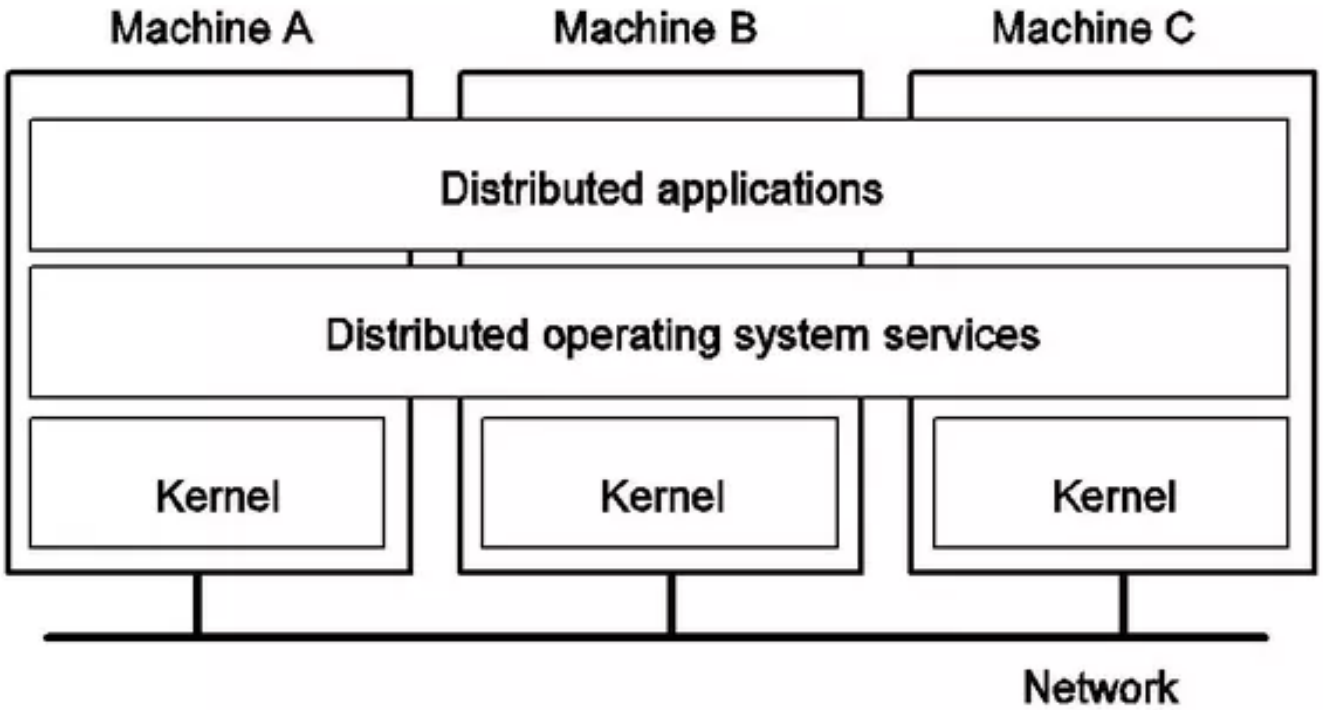
# Multiprocessor System

- Modern **Symmetric multiprocessing** (**SMP**) include a hardware and software architecture where many identical processors are connected to a single, shared main memory, have full access to all input and output devices, and are controlled by a single operating system instance that treats all processors equally, reserving none for special purposes.
- Most multiprocessor systems today use an SMP architecture.
- In the case of multi-core processors, the SMP architecture applies to the cores, treating them as separate processors.



# Distributed Operating System

- The motivation behind developing distributed operating systems is the availability of powerful and inexpensive microprocessors and advances in communication technology.
- Distributed systems consists of many computers that are interconnected by communication networks.
- The main advantage of distributed systems is its low price/performance ratio of computing. As there are multiple systems involved, user at one site can utilize the resources of systems at other sites for resource-intensive tasks.



# Realtime Operating System (RTOS)

- It is defined as an operating system known to give maximum time for each of the critical operations that it performs, like OS calls and interrupt handling.
- The Real-Time Operating system which guarantees the maximum time for critical operations and complete them on time are referred to as **Hard Real-Time** Operating Systems (RTLinux, QNX).
- While the real-time operating systems that can only guarantee a average of the time, i.e. the critical task will get priority over other tasks, but no assurity of completeing it in a defined time. These systems are referred to as **Soft Real-Time** Operating Systems (Linux, Windows).

- The biggest drawback of RTOS is that the system only concentrates on a few tasks.
- For example. In multimedia, audio has a high priority (sound clicks are a problem), but video and subtitles are lower priority (frame loss is not a problem).

# Realtime Operating System (RTOS)

## Difference between in GPOS and RTOS

Here are important differences between GPOS and RTOS:

| General-Purpose Operating System (GPOS) | Real-Time Operating System (RTOS) |
|---|---|
| It used for desktop PC and laptop. | It is only applied to the embedded application. |
| Process-based Scheduling. | Time-based scheduling used like round-robin scheduling. |
| Interrupt latency is not considered as important as in RTOS. | Interrupt lag is minimal, which is measured in a few microseconds. |
| No priority inversion mechanism is present in the system. | The priority inversion mechanism is current. So it can not modify by the system. |
| Kernel's operation may or may not be preempted. | Kernel's operation can be preempted. |
| Priority inversion remain unnoticed | No predictability guarantees |

# The End