UFV

UFV.ES

Operating Systems

MODULE 2 / UNIT 6 / 0.9

MOISES M. MARTINEZ

FUNDAMENTALS OF COMPUTER ENGINEERING

2025/2026

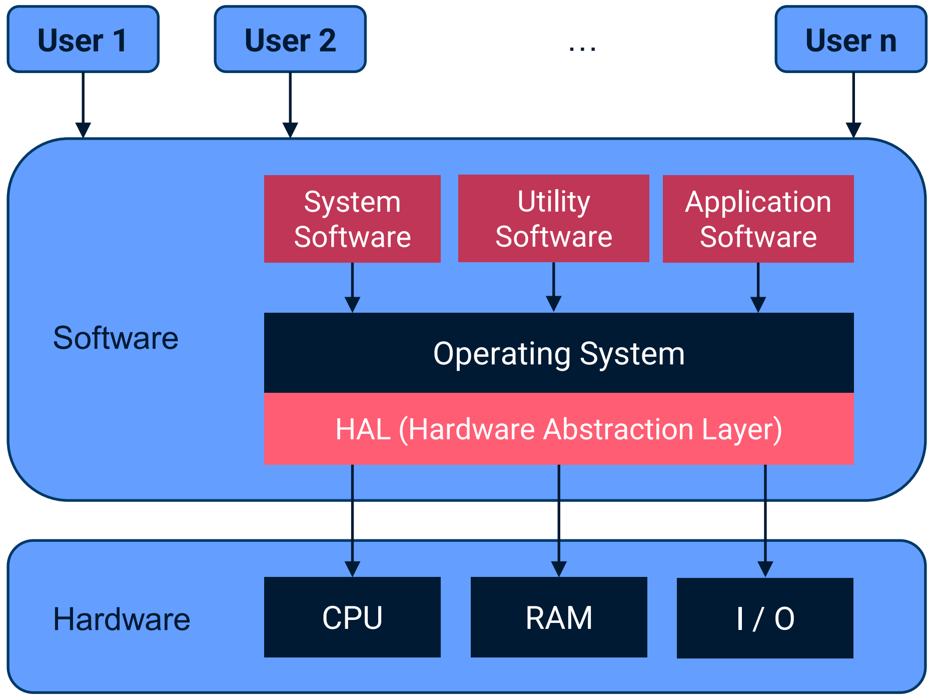
What is Operating Systems?

Fundamental Concepts

An operating system (OS) represents system software tasked with the management of computer hardware and software resources, as well as the provision of essential services for computer programs. In essence, it functions as an interface between computer users and the underlying computer hardware.

The responsibilities of an operating system encompass:

- Resource Manager
  - Manage all the hardware resources.
  - Manage user requests.
- Control program
  - Controls the programs' execution.
  - Manage the life cycle of the application software.
  - Distribute the CPU time between apps.



Fundamental Concepts

Types of Operating Systems

Operating systems (OS) can be categorized based on various criteria, including user interaction, task handling, and processor management:

- Batch OS: A batch OS allows a single user to perform one task at a time sequentially. An example of such a system is MS-DOS.
- Multiuser OS: A multiuser OS permits numerous users to concurrently utilize a computer's resources.
- Multitasking OS: A multitasking OS enables users to execute multiple computer tasks, such as running multiple application programs, concurrently. Notable examples of such operating systems include Microsoft Windows 10, IBM's OS/390, and Linux, which facilitate multitasking, a capability common to most modern operating systems.
- Multiprocessing OS: A multiprocessing OS is one in which two or more central processing units (CPUs) collectively manage the computer's functions. Each CPU operates with its copy of the OS, and these instances communicate to coordinate system operations.

Fundamental Concepts

Types of Operating Systems

Operating systems (OS) can be categorized based on various criteria, including user interaction, task handling, and processor management:

- Distributed Operating System: A Distributed OS represents a specialized type of operating system that extends the functionality of a network operating system. It facilitates enhanced levels of communication and integration among machines within a network. This entails the utilization of multiple processors distributed across various machines to deliver rapid computation capabilities to its users.
- Real-Time Operating System: A Real-Time OS (RTOS) is an operating system tailored for real-time applications characterized by stringent time constraints on data processing and event handling. These RTOSs can be categorized into two main groups: (1) Event-driven, and (2) Time-sharing.
- Mobile Operating System: A Mobile OS is a distinct operating system designed specifically for smartphones, tablets, and wearable devices, each of which possesses unique features and requirements.

OS funtions

01

OS functions

- Process Management which assists the operating system (OS) in the creation and termination of processes. It also facilitates synchronization and communication mechanisms between processes.
- Memory Management which is responsible for the allocation and deallocation of memory space required by programs.
- I/0 Management which involves monitoring and control of all connected devices. This function is typically carried out by a module known as the I/O controller.
- File Management which oversees various file-related operations, including storage organization, retrieval, naming, sharing, and file protection.

OS functions

- Networking: A distributed system is a group of processors which do not share memory, hardware devices, or a clock. The processors communicate with one another through a internal network.
- Job accounting: it keeps track of time & resource used by various job and users.
- Communication management: Coordination and assignment of compilers, interpreters, and another software resource of the various users of the computer systems.

OS functions

Loading an OS …

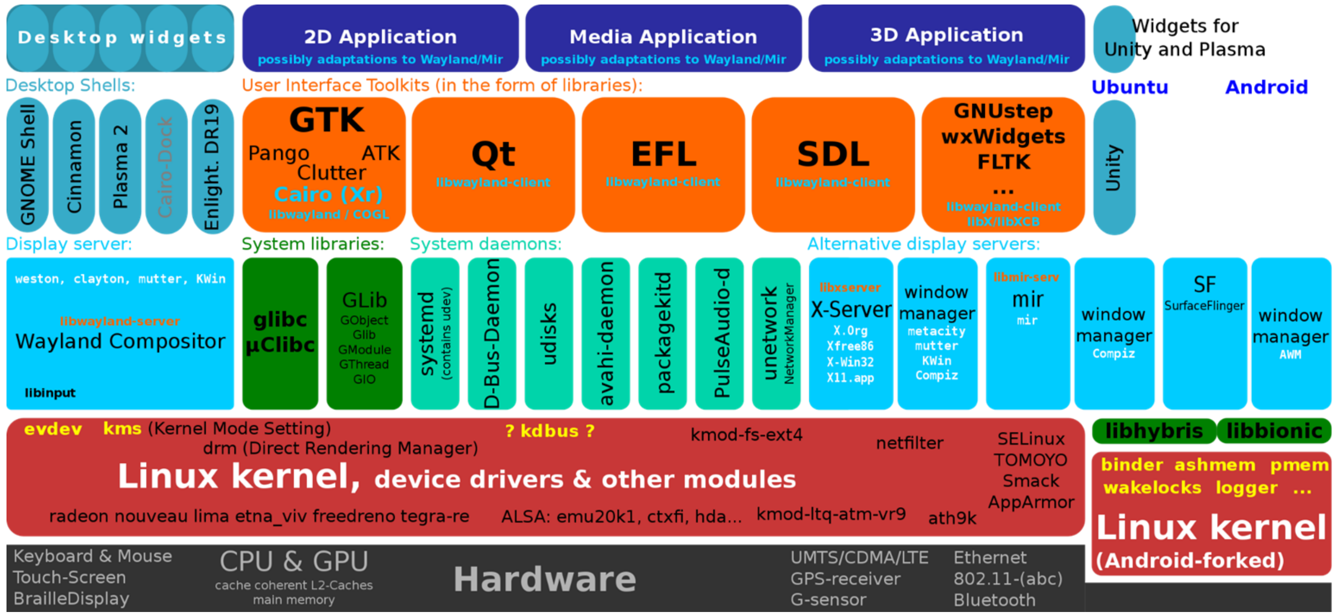
When we start a computer, the process of initialization (more or less the same in all OS) follow the next step:

- Reading: Main code is read from non-volatile storage (ROM). This code detects the boot device and reads (from the boot sector) the code to detect the location (position in the memory) of the OS.
- Loading: The **OS kernel** is loaded into the computer.
- Starting: The OS takes control by starting all services of the system (programs or system daemons) and booting the user interaction interface.

OS functions

Loading an OS …

A daemon can be defined as a computer program that operates as a background process, independent of direct interaction with a user.



In Unix-based environments, it is common for the parent process of a daemon to be the init process, although exceptions exist.

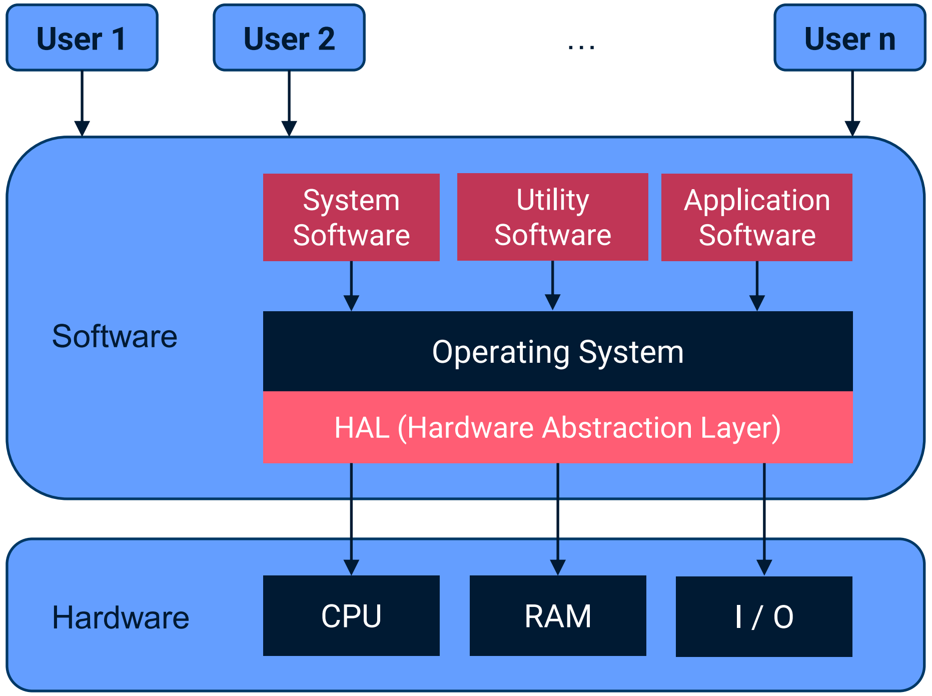
Main operations

02

Main Operations

An operating system (OS) primarily executes four fundamental operations:

- Process Management.
- Memory Management.
- Input/Output (I/O) Operations Management.
- File Management.



These core functions collectively provide the foundation for the proper functioning of an operating system.

Main operations

Process Management

03

Main Operations

Process Management

Process management involves a range of essential tasks related to the processes, such as process creation, scheduling, termination, and deadlock resolution. These tasks are crucial for enabling the concurrent execution of multiple processes, allowing a computer to run several programs simultaneously, rather than being limited to a single program at any given time.

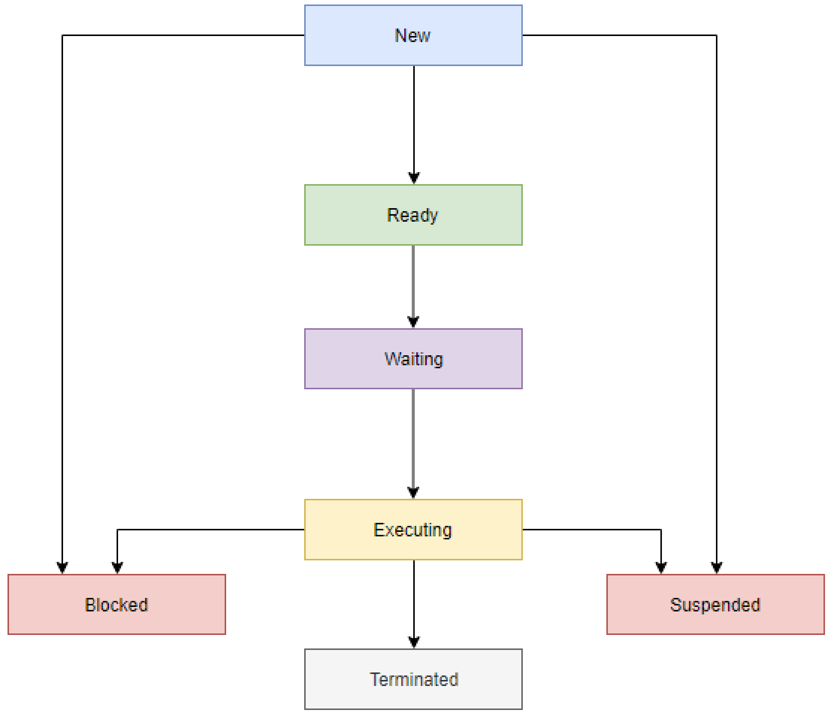
A process is a program that is under execution.

Additionally, it provides essential mechanisms for **synchronization and coordination among processes**, ensuring they work together efficiently and execute smoothly. These functions are critical for maintaining system stability and optimizing performance in a multitasking environment to facilitate **interprocess communication** and data exchange.

Main Operations

Process Management

- New
- Ready
- Waiting
- Executing
- Blocked
- Suspended
- Terminated

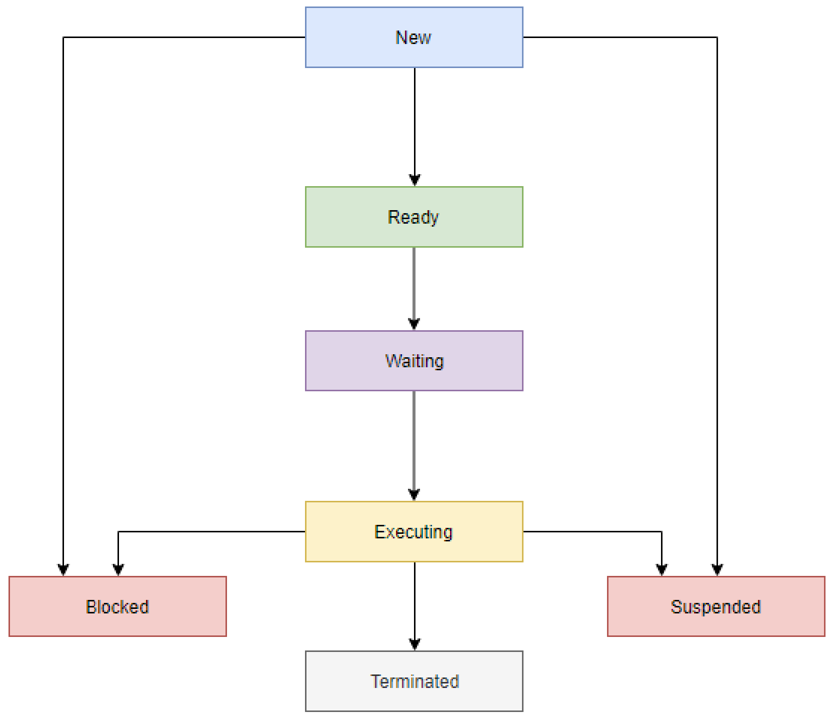


After completing each phase, a process **releases all the resources it was allocated**, leading to the freeing up of system memory. This resource deallocation is essential for maintaining system efficiency, as it allows the memory and resources to be reallocated to other processes.

Main Operations

Process Management

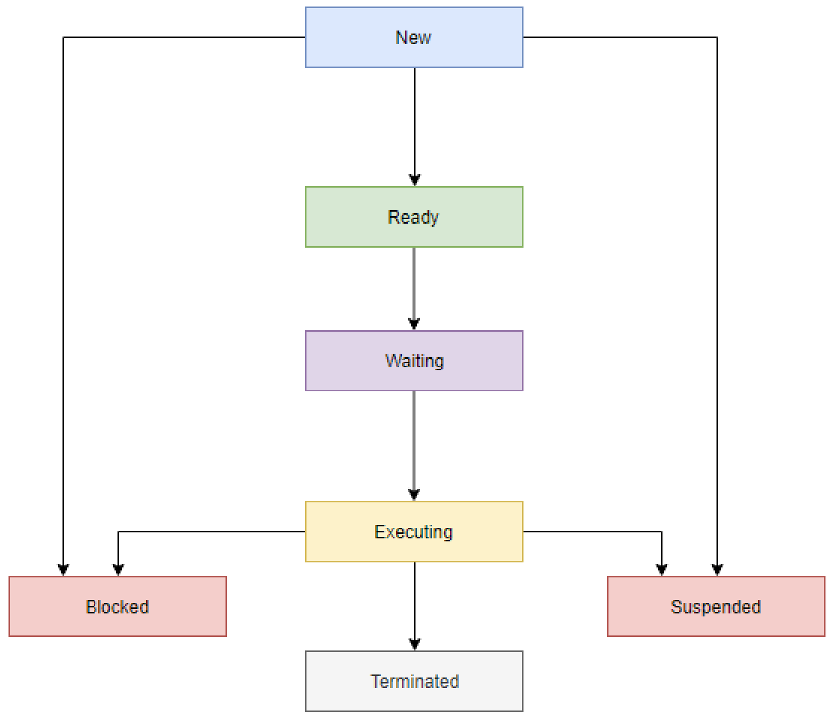
- New: A process is created when a program is initiated, leading to the transfer of necessary data from secondary storage (such as a hard disk) to primary memory (RAM).
- Ready: The process is fully prepared and loaded into primary memory, waiting for the operating system to schedule its execution.
- Waiting: The process is in a state of anticipation, awaiting the allocation of CPU time or other critical resources before it can begin execution.
- Executing: The process is actively running, with its instructions being executed by the CPU.



Main Operations

Process Management

- Blocked: The process is in a waiting state, usually pending the occurrence of a specific event, such as the completion of I/O operations, before it can continue execution.
- Suspended: The process is ready for execution but has been temporarily removed from the ready queue by the operating system, **typically to free up system resources or because higher-priority processes need to be scheduled**.
- Terminated: The process has finished its execution and has been removed from the system's active process list, meaning it is no longer running.

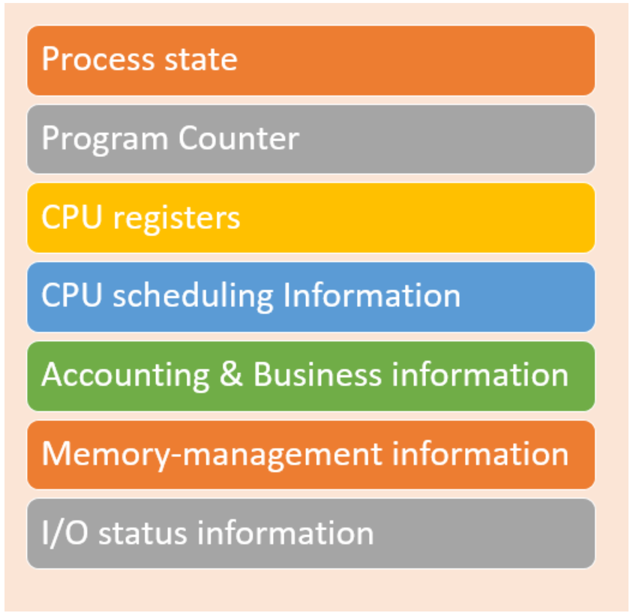


Main Operations

Process Management

In the context of an operating system, processes are represented by a data structure called Process Control Block (PCB), or Task Control Block (TCB). The PCB/TCB contains essential information about a process, including its state, program counter, memory allocation, CPU registers, and scheduling information, enabling the operating system to manage and control process execution effectively.

- Process State indicates the **current status of a process**, such as new, ready, running, waiting, or terminated, which helps the operating system manage process execution.
- Program Counter holds the **address of the next instruction that the CPU will execute for the process**, enabling the operating system to track the process's execution flow.

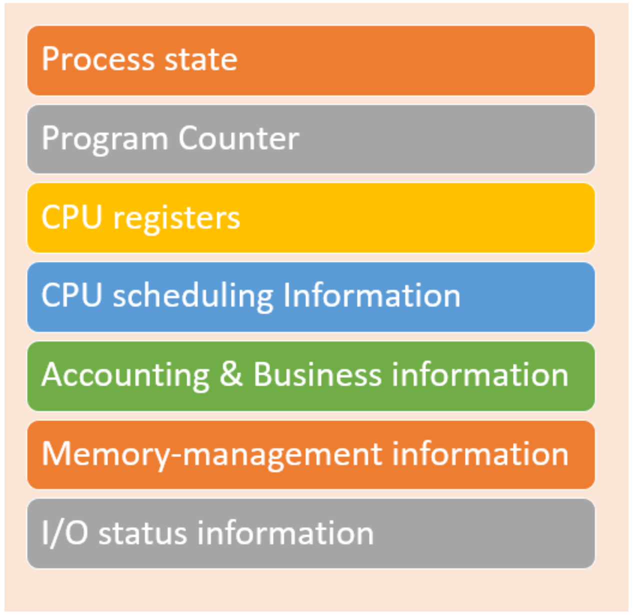

Process state
Program Counter
CPU registers
CPU scheduling Information
Accounting & Business information
Memory-management information
I/O status information

Main Operations

Process Management

In the context of an operating system, processes are represented by a data structure called Process Control Block (PCB), or Task Control Block (TCB). The PCB/TCB contains essential information about a process, including its state, program counter, memory allocation, CPU registers, and scheduling information, enabling the operating system to manage and control process execution effectively.

- CPU Register include **accumulators**, **index registers**, **general-purpose registers**, and condition codes, storing essential data and state information required for process execution.
- CPU Scheduling Information includes details such as the **process's priority**, pointers to **scheduling queues**, and other parameters essential for determining how and when the process should be scheduled for execution by the CPU.

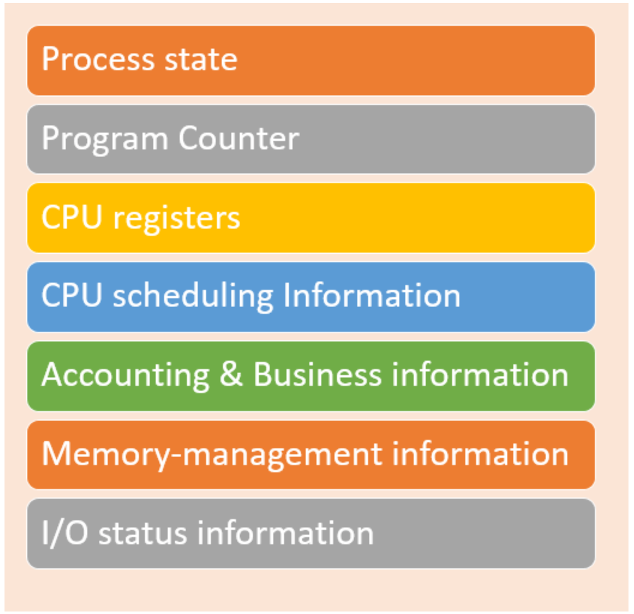

Process state
Program Counter
CPU registers
CPU scheduling Information
Accounting & Business information
Memory-management information
I/O status information

Main Operations

Process Management

In the context of an operating system, processes are represented by a data structure called Process Control Block (PCB), or Task Control Block (TCB). The PCB/TCB contains essential information about a process, including its state, program counter, memory allocation, CPU registers, and scheduling information, enabling the operating system to manage and control process execution effectively.

- Accounting and Business Information stores data such as the amount of **CPU time utilized**, **real-time usage**, and identifiers **like job or process** numbers, which are used for **tracking resource consumption** and process management within the system.
- I/O Status Information contains a list of open files and the I/O devices allocated to the process, along with other relevant details. It tracks the **process's interactions with input/output devices** and manages its access to necessary resources.

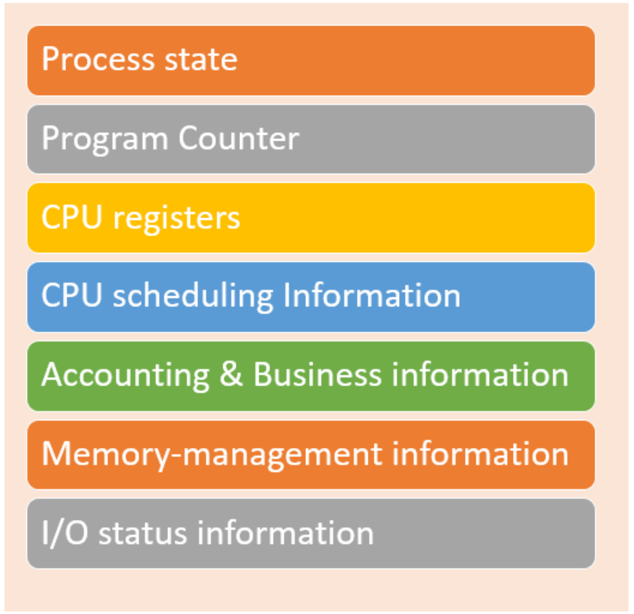

Process state
Program Counter
CPU registers
CPU scheduling Information
Accounting & Business information
Memory-management information
I/O status information

Main Operations

Process Management

In the context of an operating system, processes are represented by a data structure called Process Control Block (PCB), or Task Control Block (TCB). The PCB/TCB contains essential information about a process, including its state, program counter, memory allocation, CPU registers, and scheduling information, enabling the operating system to manage and control process execution effectively.

- Memory-Management Information includes details such as the values of **base and limit registers**, as well as **page** or **segment tables**, depending on the memory management system used by the operating system. This information is crucial for managing the process's memory allocation and ensuring that it accesses only its designated memory space.


Process state
Program Counter
CPU registers
CPU scheduling Information
Accounting & Business information
Memory-management information
I/O status information

Main operations

Memory Management

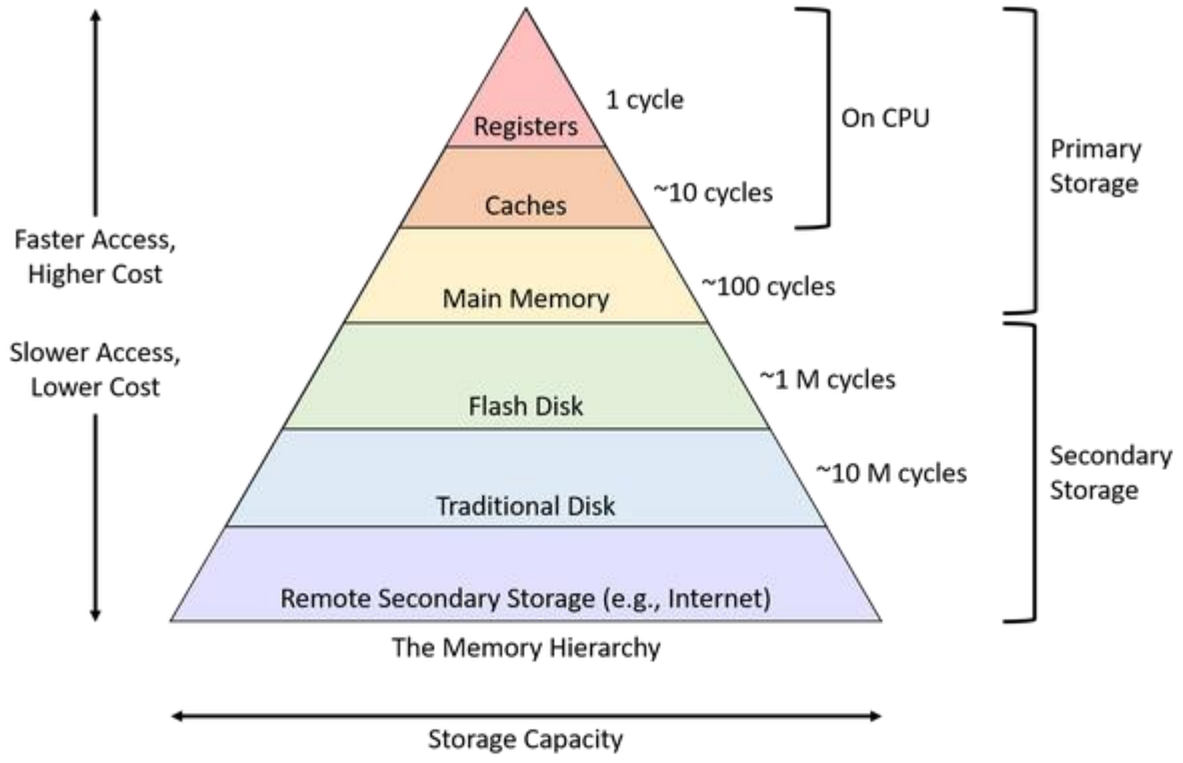
04

Main Operations

Memory Management

Memory management involves managing the address space, which consists of a set of virtual addresses allocated for executing instructions and storing data.

- Primary storage/memory (on-CPU) resides on the CPU and includes registers and cache, provides the fastest access to data directly within the CPU, crucial for immediate processing.
- Primary storage/memory (Off-CPU) resides on the motherboard and serves as the main memory (RAM and ROM) for active processes and essential firmware.
- Secondary storage/memory, like HDDs, SSDs, and external drives, offers large-capacity, long-term data storage, accessible even after the computer is powered off.



Main Operations

Memory Management

The address space allows the operating system to efficiently allocate memory, ensuring that each process has access to the necessary resources while maintaining isolation and stability within the system. It is utilized, comprising a range of virtual addresses specifically designated for executing instructions and storing data.

1. Programs are stored on secondary storage/memory as binary executable files.
2. Before execution, these programs must be loaded into memory and encapsulated within a process. Depending on the memory management scheme in use, a process may be relocated between disk and memory during its execution.
3. The **collection of processes that reside on the disk**, waiting to be transferred into memory for execution, is known as the **input queue**.

We do not have enough memory to keep all programs in primary memory simultaneously.

Main Operations

Memory Management

Memory management is based on the concept of an **address space**, which is a defined range of virtual addresses allocated for executing instructions and storing data. There are different strategies for memory management:

- Virtual Memory strategy leverages mass storage devices like hard drives or SSDs to extend the capabilities of physical memory.
- Paging strategy divides the memory into fixed-size blocks called **pages** and the physical memory into **frames**. When a program needs to execute, its pages are loaded into available frames in physical memory.
- Segmentation strategy divides memory into variable-sized segments (Each segment is loaded into memory separately) based on the logical divisions of a program, such as code, data, and stack segments.
- Swapping involves moving entire processes between the primary storage and secondary storage (such as a hard disk). When the memory is full, a process that is not currently needed can be swapped out to disk, freeing up memory for other processes.

Main Operations

Memory Management

The primary strategy for modern memory management is **Virtual Memory**, which utilizes mass storage devices, such as hard drives or SSDs, to extend the capabilities of a computer's physical memory (RAM).

- It combines hardware and software to manage situations where the physical memory is insufficient to handle all the running processes.
- It works by temporarily transferring data that is not immediately needed in RAM to disk storage, thereby freeing up physical memory for more critical tasks.
- It operates by dividing memory into blocks and mapping these blocks to corresponding files on disk. When a process requires more memory than is available in RAM, some of its data can be moved to disk, allowing the computer to continue operating as though it has more RAM than is physically installed.

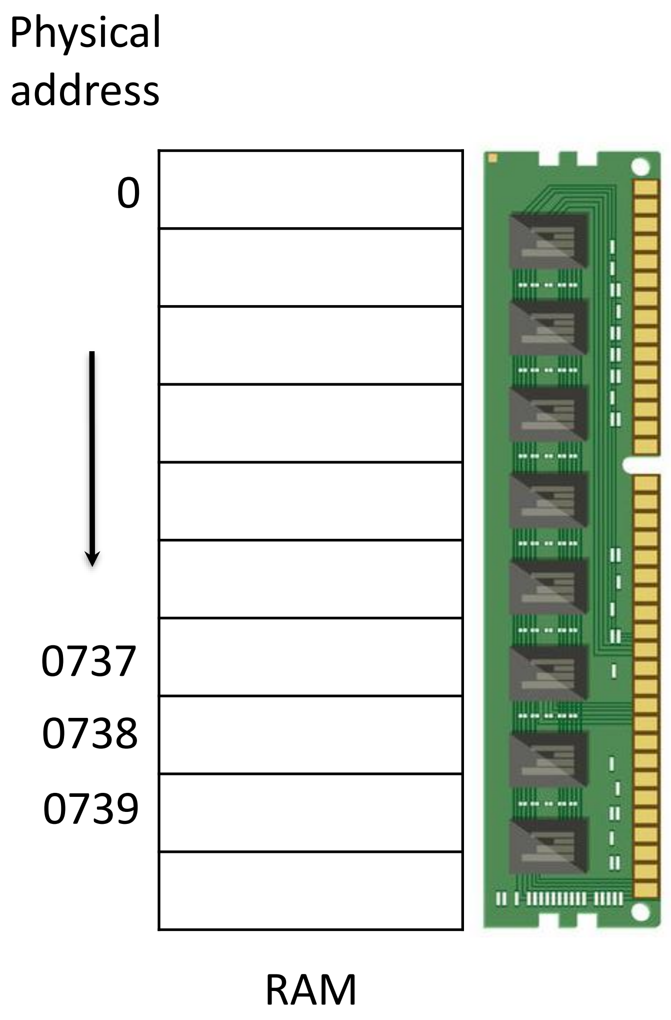
Virtual memory effectively treats the disk space as an extension of RAM, enabling the system to run larger applications or multiple programs simultaneously without running out of memory.

Main Operations

Memory Management

Virtual memory extends the capacity of main memory (RAM) for process execution without the need to add more physical memory to the computer. This extension is accomplished by utilizing the hard disk (or other secondary or mass storage devices) as a supplementary resource to RAM.

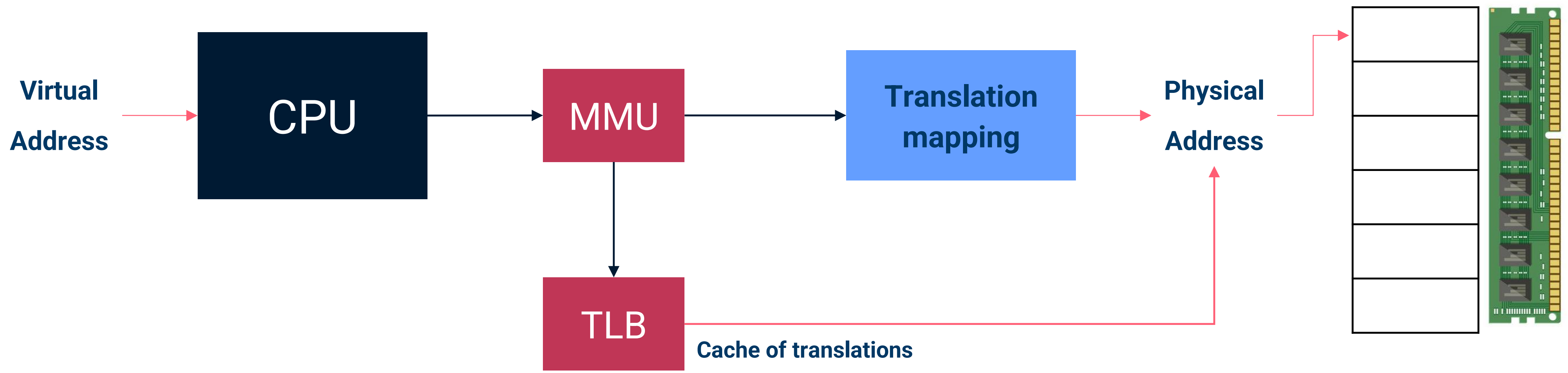
Random Access Memory (RAM) is structured as an array of discrete cells or storage units, each capable of holding exactly **one byte of data**, which is equivalent to **8 bits**. Every cell in RAM has a unique memory address, allowing the system to access and store data in any cell directly and efficiently. This structure supports the rapid retrieval and storage of information, which is critical for the smooth operation of running programs.



Main Operations

Memory Management

The real-time translation from a virtual address to a physical address is accomplished through the utilization of the Memory Management Unit (MMU), which is a hardware component. The MMU employs the following mechanism to effectuate the translation of virtual addresses into their corresponding physical addresses.



The Translation Lookaside Buffer (TLB) is a specialized cache used to speed up the process of translating virtual addresses to physical addresses.

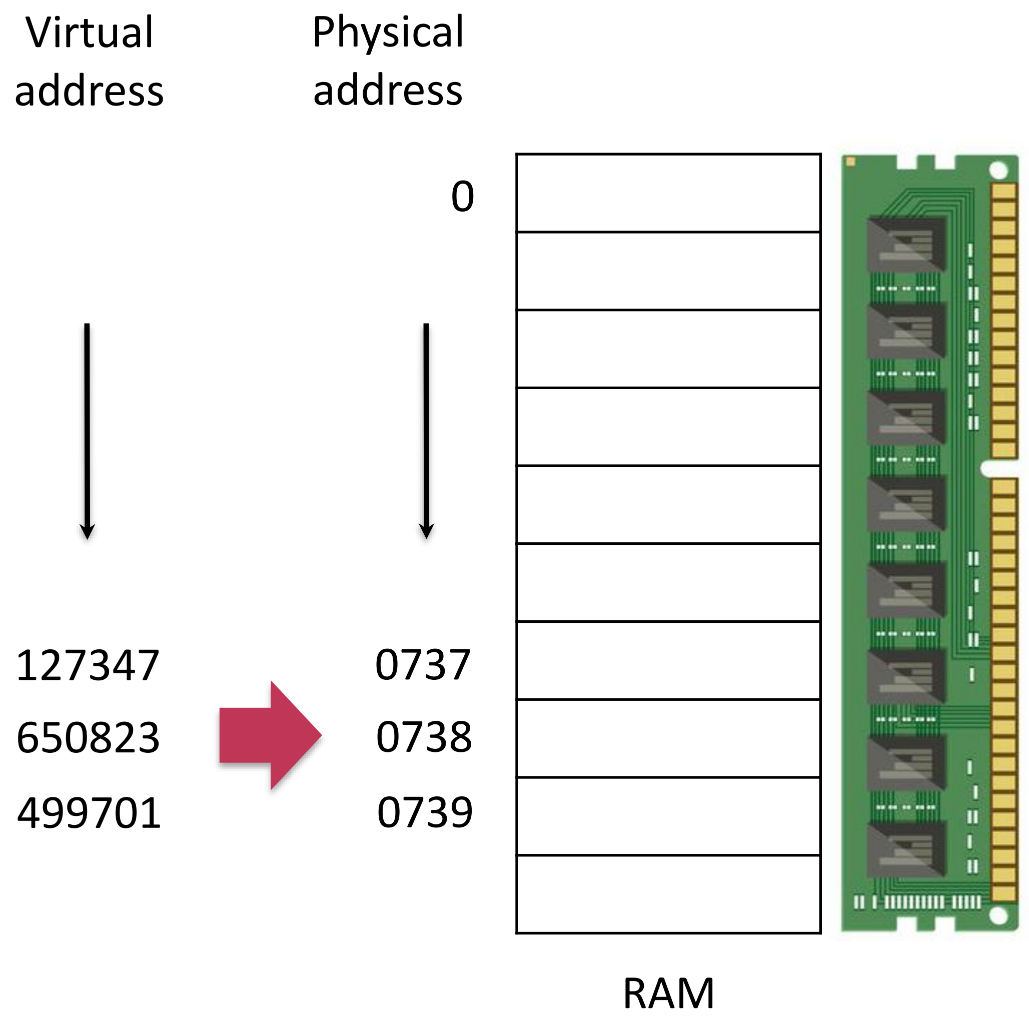
Main Operations

Memory Management

Processes do not directly use physical addresses for accessing memory; instead, they operate using virtual addresses, which are abstracted memory locations.

Virtual addresses are often larger in bit size compared to physical addresses, allowing the system to create a much larger virtual address space than the available physical memory.

This design enables the operating system to manage and allocate memory more flexibly, giving the illusion that each process has access to a vast, contiguous block of memory, even if the physical memory is limited.

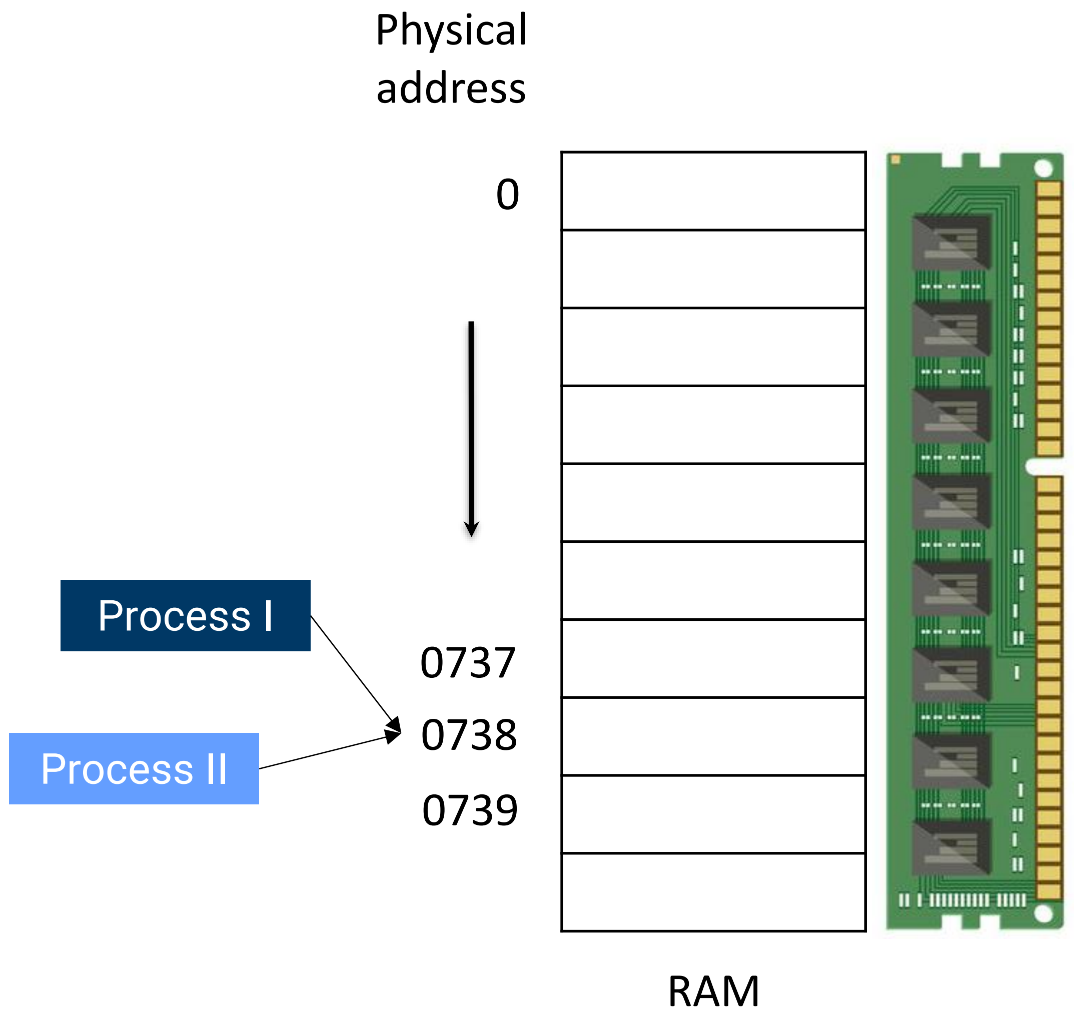


Main Operations

Memory Management

When a program is executed, it accesses memory without any inherent awareness of the need to share this memory with other processes. Programs operate in isolation, oblivious to the presence of other processes running concurrently on the system.

This lack of awareness can lead to potential **conflicts**, as two or more active processes might inadvertently attempt to use the same physical memory locations to store their data. Such conflicts can result in data corruption, unpredictable behaviour, and system instability. To prevent these issues, the operating system employs memory management techniques like virtual memory, which assigns each process its own virtual address space.



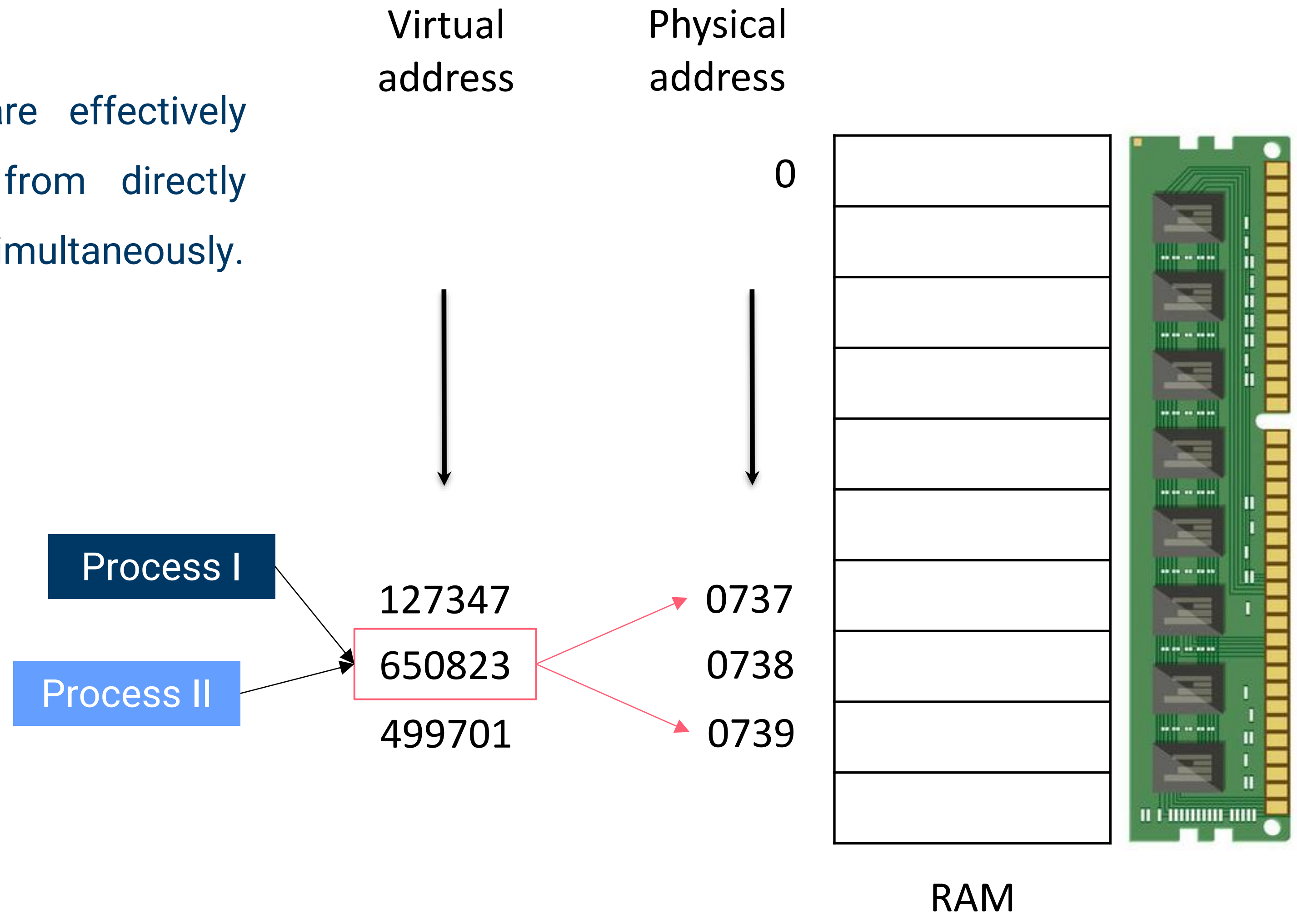
Main Operations

Memory Management

In virtual memory environments, processes are effectively isolated from each other, preventing them from directly accessing the same physical memory addresses simultaneously.

The same virtual address (e.g., 0x650823) in two different processes can exist simultaneously, but these addresses are mapped to different physical memory locations.

Each process is given its own virtual address space.



Main operations

I/O Management

05

Main Operations

I/O Management

I/O Management is specifically concerned with overseeing and coordinating Input/Output operations, ensuring controlled and efficient access to a variety of hardware devices. This involves managing the interaction between the system's software and peripheral devices like keyboards, mice, printers, storage drives, and network interfaces, ensuring that data is transmitted correctly, efficiently, and without conflicts between competing processes.

- Computers consist of CPUs and various **device controllers** that are connected through a shared bus channel, with device drivers acting as intermediaries.
- **Device drivers** provide an interface between the operating system and I/O devices, enabling smooth communication with the system hardware. They manage the data exchange between the CPU and peripheral devices, ensuring efficient operation and allowing controlled access to shared memory resources.

Device controllers

Main Operations

I/O Management

Input/Output management is facilitated by several key components, each responsible for specific aspects of I/O operations.

- The **I/O Traffic Controller** maintains a detailed record of the operational status of all devices, control units, and communication channels within the system. By monitoring these elements, it ensures that data traffic flows smoothly and that resources are utilized effectively, preventing conflicts and bottlenecks.
- The **I/O Scheduler** implements the policies set by the operating system to allocate and manage access to devices, control units, and communication channels. It prioritizes I/O tasks based on system requirements, ensuring that resources are distributed efficiently and that critical operations receive the necessary attention.
- The **I/O Device Handler** is responsible for managing device interrupts and overseeing the data transfer process. By handling device-specific tasks and coordinating data interactions, it ensures that data is transferred efficiently and that the system responds promptly to I/O requests, playing a pivotal role in maintaining overall system performance.

Main operations

File Management

06

Main Operations

File Management

File management enables the efficient storage, retrieval, and protection of vast amounts of data.

- **Secondary storage/memory** allows computers to store immense quantities of data, from gigabytes to petabytes. Access to these storage devices is controlled by Disk Storage Management, which handles tasks such as partition creation, deletion, and formatting to prepare the storage medium for file storage.
- The **file system** is essential for organizing and managing data on secondary storage, serving multiple critical functions:
  - Hierarchical Organization: It manages the hierarchical structure of folders and directories, ensuring an orderly and logical arrangement for storing and retrieving files.
  - File Naming Conventions: It defines the syntax for naming files, including rules for permissible characters and length restrictions, to maintain consistency and compatibility across the system.
  - Access Controls: It incorporates robust access control mechanisms to protect data from unauthorized access, ensuring that only authorized users can view or modify sensitive information.
  - Data Integrity and Reliability: It implements safeguards against data loss due to system crashes or failures. These mechanisms help preserve the integrity of stored information, ensuring that data remains intact and recoverable in the event of an unexpected disruption.

Main Operations

UFV

File Management

Key Concepts of a File System:

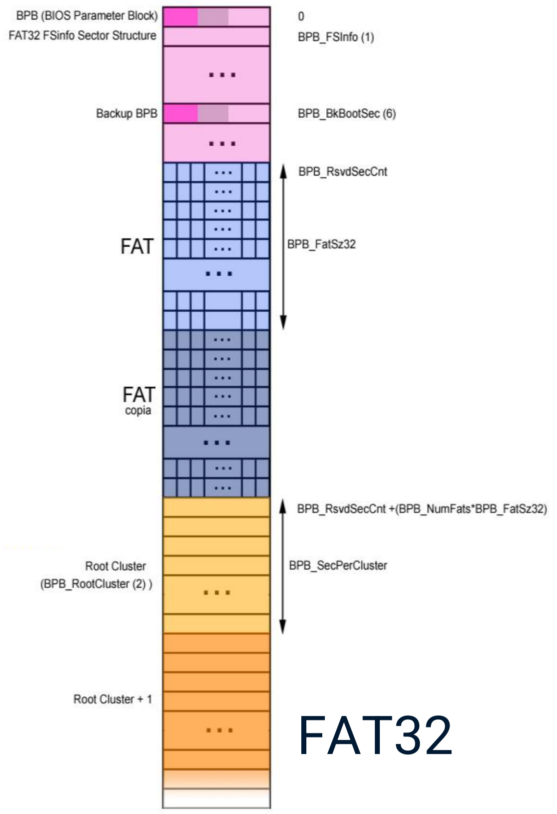
- A **file** is a logical unit of data created by processes during execution, serving as a container for information that can be stored, retrieved, and manipulated by the system.
- A **directory/folder** is a structured storage location that can contain multiple files, providing an organized framework for managing and categorizing data within the file system.
- A **partition** is a defined segment of the storage medium that functions as a separate and independent unit, isolated from other partitions, allowing for better management and organization of storage resources.
- The **access mechanism** is the process managed by the operating system to control and authorize user or process access to files, ensuring security and proper usage of the system's resources.
- A **file extension** is a suffix added to a file name, typically following a dot, that indicates the file's type and the kind of data it contains, helping the operating system and users identify the file's purpose.

Main Operations

File Management

Different operating systems utilize various file systems, each designed to meet specific needs for data organization, security, and efficiency:

- Windows File Systems: FAT32 (File Allocation Table 32, exFAT (Extended File Allocation Table) and NTFS (New Technology File System).
- MacOS File Systems: FAT32, exFAT, HFS+ (Hierarchical File System Plus) and APFS (Apple File System).
- Linux File Systems: ext2 (Second Extended File System, ext3 (Third Extended File System, ext4 (Fourth Extended File System, FAT32 and exFAT.
- Unix File Systems: UFS (Unix File System), ext2, ext3, ext4 and ZFS (Zettabyte File System).


BPB (BIOS Parameter Block)
FAT32 FSinfo Sector Structure
Backup BPB
FAT
FAT copia
Root Cluster (BPB_RootCluster (2))
Root Cluster + 1
0
BPB_FSInfo (1)
BPB_BkBootSec (6)
BPB_RsvdSecCnt
BPB_FatSz32
BPB_RsvdSecCnt +(BPB_NumFats*BPB_FatSz32)
BPB_SecPerCluster
FAT32

UFV