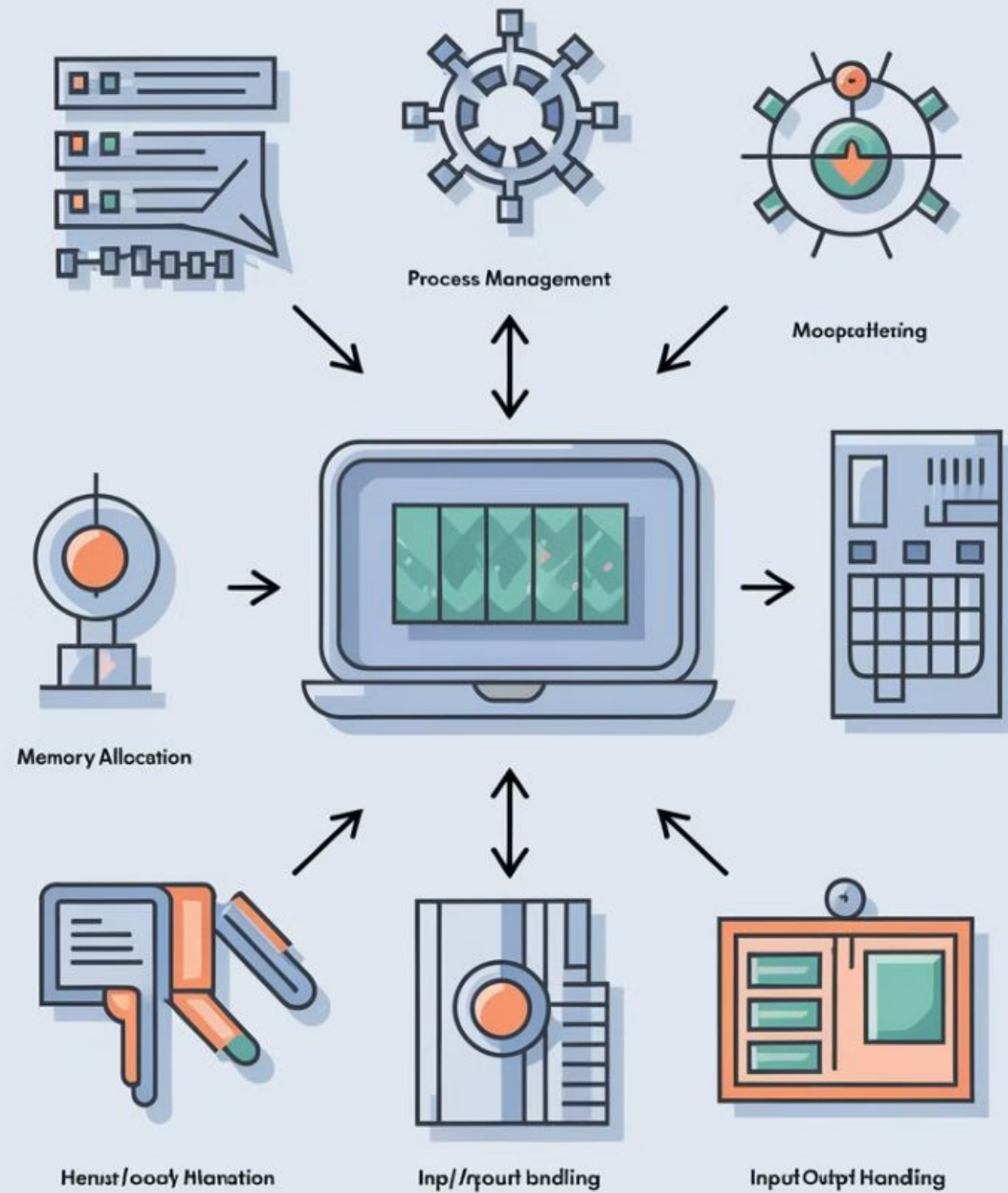


# OPERATING SYSTEM



## Sistemas Operativos: Introducción y Conceptos Fundamentales

Este curso analiza los principios, diseño e implementación de los sistemas operativos, esenciales para gestionar el hardware y ejecutar aplicaciones. Los estudiantes comprenderán su arquitectura, componentes y funciones, incluyendo gestión de procesos, memoria, almacenamiento y entrada/salida.



# ¿Qué es un Sistema

## Operativo?

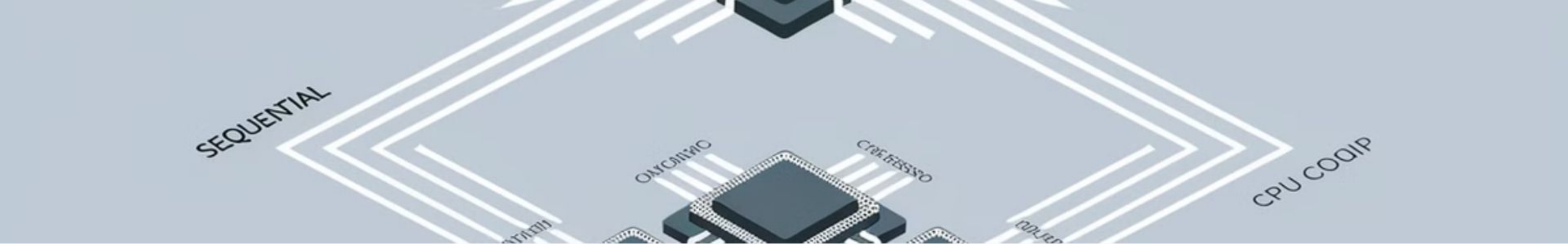
Definición Básica

Un sistema operativo (SO) es un **conjunto de programas de software que gestionan los recursos de hardware de una computadora** y proporcionan servicios comunes a las aplicaciones de software.



## Intermediario

Actúa como **intermediario entre el usuario y el hardware**: El SO crea una **interfaz más conveniente y eficiente** para que los usuarios interactúen con el hardware de la computadora y para que las aplicaciones utilicen estos recursos. Sin un SO, las aplicaciones tendrían que interactuar directamente con el complejo hardware, lo cual sería una tarea ardua y propensa a errores.



# Temario del Curso



## Introducción

Repaso de la arquitectura de computadoras

Concepto e historia de los Sistemas

Operativos

Objetivos, funciones y evolución de los

Sistemas Operativos

## Procesos.

- Procesos. Descripción y control.



- Hilos.
- Comunicación de procesos.
- Sincronización entre procesos.
- Transacciones atómicas.
- Scheduling.

## Administración de memoria

- Asignación dinámica de memoria.
- Memoria principal.
- Memoria virtual.



- Memoria de usuario.
- Memoria de kernel.

## Almacenamiento masivo

1. Servidor de sistema de archivos.
2. Servicios del sistema operativo para almacenamiento.
3. Sistema de archivos UNIX y WINDOWS.

# Arquitectura de Computadoras: Una Revisión

## Importancia para los SO

El diseño y la funcionalidad de un sistema operativo están **íntimamente ligados a la arquitectura de la computadora** que gestiona. Comprender cómo interactúan el procesador, la memoria y los dispositivos de E/S es fundamental para entender las decisiones de diseño y los mecanismos implementados en los SO.

## Componentes Básicos

Los componentes fundamentales de una computadora son la **Unidad Central de Procesamiento (CPU) o procesador**, la **memoria principal** y los **dispositivos de Entrada/Salida (E/S)**. El sistema operativo es responsable de coordinar y gestionar estos componentes para la ejecución de programas.

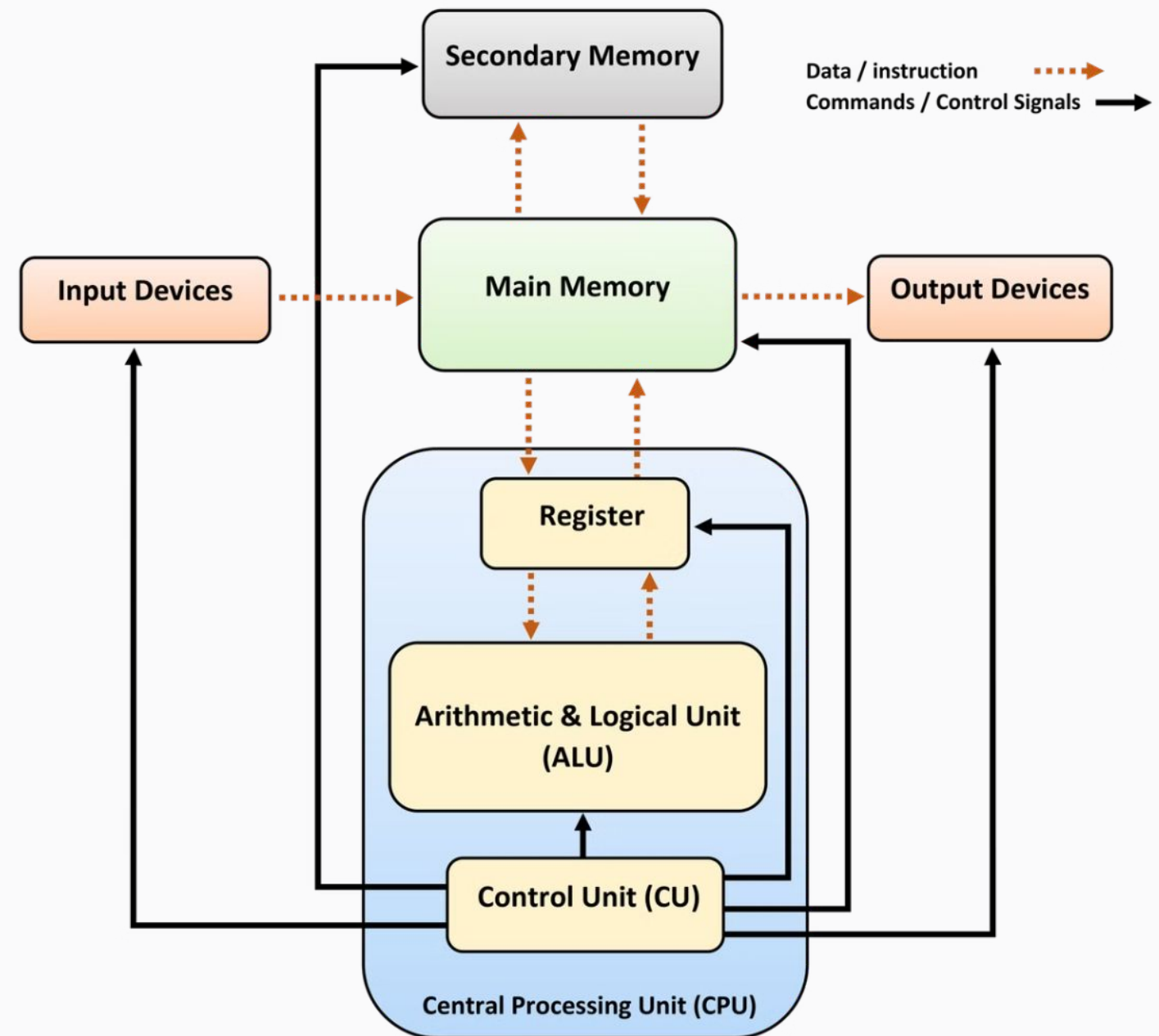


Image By Amila Ruwan 20

# Arquitectura de Computadoras - Procesador y Memoria



## Procesador (CPU)

La CPU es el **cerebro de la computadora**, responsable de ejecutar las instrucciones de los programas. Está compuesta por la unidad de control, la unidad aritmético-lógica (ALU), registros y caché.

La **unidad de control** dirige la secuencia de operaciones

La **ALU** realiza cálculos

Los **registros** almacenan datos temporales

La **caché** acelera el acceso a la memoria



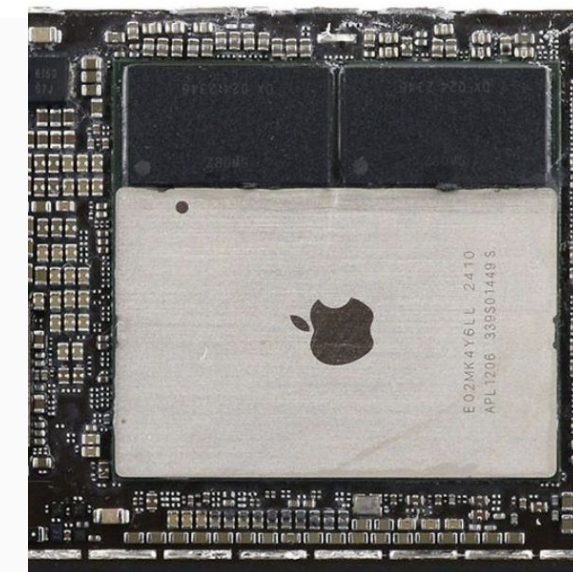
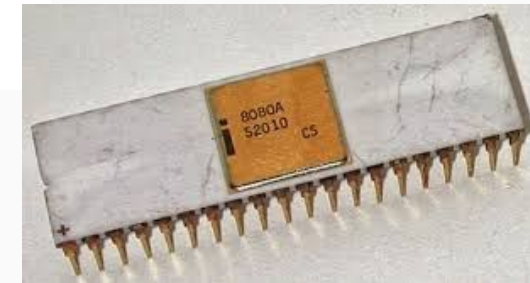
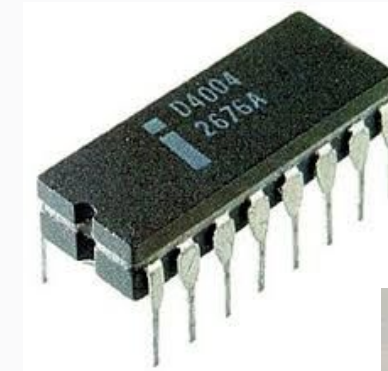
## Memoria Principal

También conocida como **RAM (Random Access Memory)**, es el almacenamiento de acceso rápido donde se cargan los **datos y las instrucciones de los programas que se están ejecutando**.

La RAM permite el acceso directo a cualquier ubicación de memoria, lo que es crucial para la eficiencia en la ejecución de programas. Es importante destacar que la RAM es **volátil**, lo que significa que su contenido se pierde cuando la computadora se apaga.

# Evolución del Procesador

- 1971 - Intel 4004: Primer microprocesador comercial, con 4 bits y 2,300 transistores.
- 1974 - Intel 8080: Base de las primeras computadoras personales, con arquitectura de 8 bits.
- 1978 - Intel 8086: Introducción de la arquitectura x86, base de los procesadores modernos.
- 1985 - Intel 80386: Primer procesador de 32 bits, con capacidad de multitarea avanzada.
- 1993 - Intel Pentium: Mejoras en rendimiento con arquitectura superscalar.
- 2000 - AMD Athlon 64: Primer procesador de 64 bits para consumidores.
- 2006 - Intel Core 2 Duo: Popularización de la arquitectura multinúcleo.
- 2011 - Intel Sandy Bridge: Integración de gráficos en el procesador.
- 2017 - AMD Ryzen: Revolución en el mercado con múltiples núcleos y alta eficiencia.
- 2020 - Apple M1: Transición a arquitectura ARM en computadoras de alto rendimiento.



# Capacidades Actuales de los Procesadores

## Apple M4:

- Proceso de fabricación: 3 nanómetros de segunda generación
- Núcleos: Hasta 10 (combinación de núcleos de alto rendimiento y eficiencia)
- Memoria unificada: Hasta 16 GB
- Neural Engine: Capaz de realizar hasta 38 billones de operaciones por segundo

## Intel Core Ultra 9 285K:

- Proceso de fabricación: 3 nanómetros
- Núcleos: 24 (8 de rendimiento y 16 de eficiencia)
- Frecuencia base: 3.7 GHz
- Frecuencia turbo: Hasta 5.5 GHz
- Memoria soportada: DDR5-6400, hasta 192 GB
- Consumo de energía (TDP): 125 W

## AMD Ryzen 9 7950X:

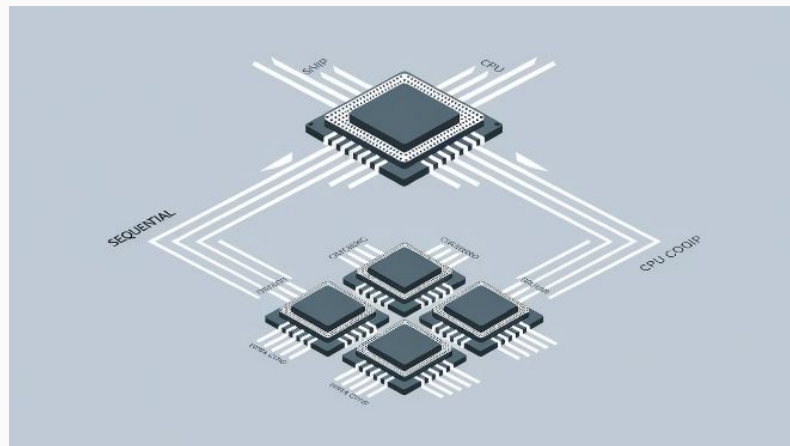
- Proceso de fabricación: 5 nanómetros
- Núcleos: 16
- Frecuencia base: 4.5 GHz
- Frecuencia turbo: Hasta 5.7 GHz
- Memoria soportada: DDR5, hasta 128 GB
- Consumo de energía (TDP): 170 W
- Estas especificaciones reflejan las capacidades avanzadas de cada procesador en sus respectivas plataformas.



# Arquitecturas de Computadoras - Arquitecturas Nowadays

## Arquitecturas Mononúcleo vs. Multinúcleo

- Aunque las arquitecturas mononúcleo (con un solo núcleo de procesamiento) todavía existen, la mayoría de los sistemas modernos, desde computadoras personales hasta servidores y dispositivos móviles, utilizan arquitecturas multinúcleo.
- Los procesadores multinúcleo ofrecen un mayor rendimiento para cargas de trabajo paralelas y multitarea.



## Arquitecturas RISC vs. CISC

- Históricamente, las arquitecturas de procesadores se dividieron en RISC y CISC.
- RISC (Reduced Instruction Set Computing) utiliza un conjunto de instrucciones más pequeño y simple, lo que facilita una ejecución más rápida.
- CISC (Complex Instruction Set Computing), por otro lado, utiliza un conjunto de instrucciones más grande y complejo.
- Si bien la distinción se ha difuminado con el tiempo, muchas arquitecturas modernas incorporan principios de ambas.

# Ejecución de Instrucciones - Proceso Básico



El procesador contiene una **unidad de control** que coordina las diferentes etapas del ciclo de instrucción, así como una **unidad aritmético-lógica (ALU)** que realiza las operaciones aritméticas y lógicas especificadas por las instrucciones. El procesador utiliza registros internos para almacenar operandos y resultados temporales durante la ejecución.

# Ejecución de Instrucciones - Tipos de Instrucciones

## Instrucciones de Transferencia de Datos

Estas instrucciones mueven datos entre diferentes ubicaciones de memoria (incluyendo registros, caché y RAM) y entre la memoria y los dispositivos de E/S.

Ejemplos incluyen **LOAD** (cargar un valor de la memoria a un registro)

**STORE** (almacenar un valor de un registro en la memoria)

## Instrucciones Aritmético-Lógicas

Estas instrucciones realizan operaciones aritméticas y lógicas en los datos.

Aritméticas: **ADD, SUBTRACT, MULTIPLY, DIVIDE**

Lógicas: **AND, OR, NOT**

## Instrucciones de Control de Flujo

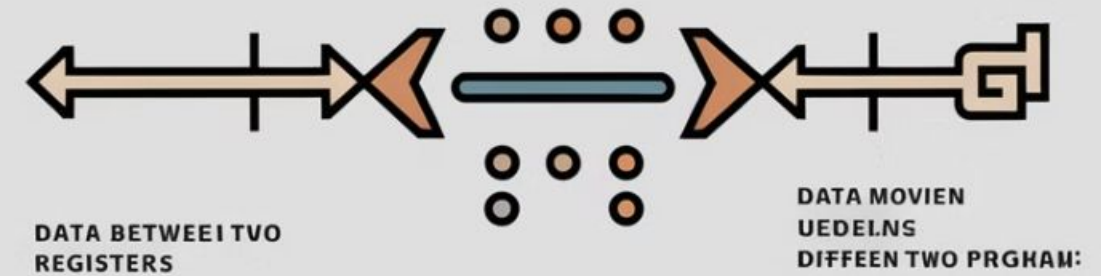
Estas instrucciones alteran la secuencia normal de ejecución de las instrucciones, permitiendo la implementación de estructuras de control.

Saltos (**JUMP**)

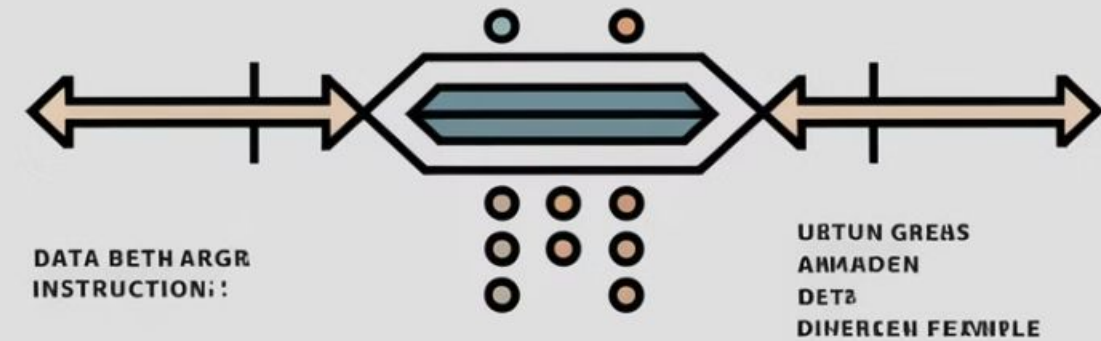
Bifurcaciones condicionales (**BRANCH**)

Llamadas a subrutinas (**CALL**)

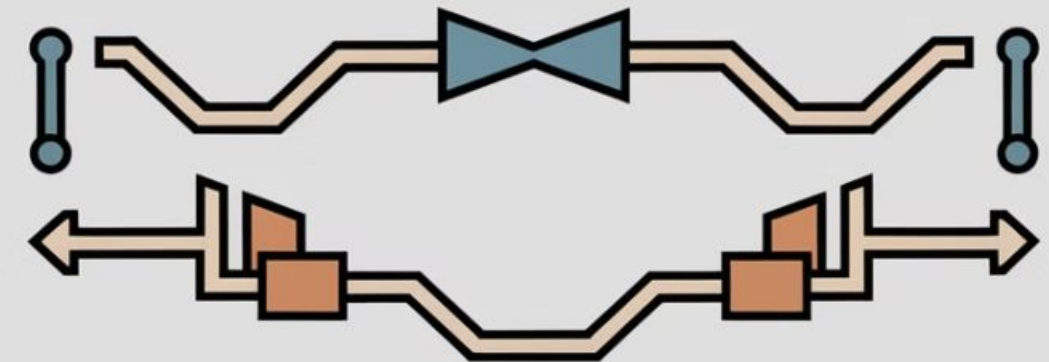
## DATA TRANSFER



## ARITHMETIC-LOGIC



## CONTROL FLOW

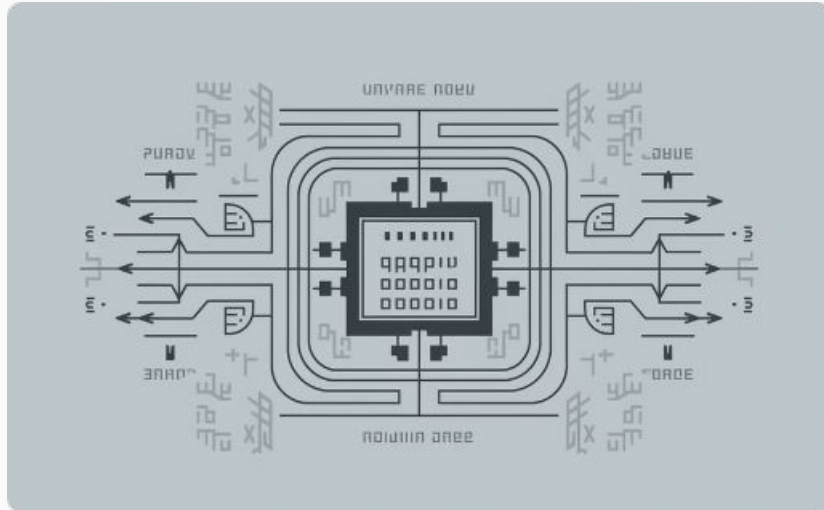


# Jerarquía de Memoria - Introducción



La jerarquía de memoria organiza los diferentes tipos de almacenamiento en niveles según su velocidad, costo por bit y capacidad. Los registros dentro del CPU constituyen el nivel más rápido y pequeño. Le siguen la memoria caché (con múltiples niveles) y la memoria principal (RAM), que ofrecen un equilibrio entre velocidad y capacidad. Finalmente, el almacenamiento secundario, como discos duros o SSDs, proporciona mayor capacidad a menor costo y velocidad. Esta estructura permite a los sistemas optimizar el rendimiento y la eficiencia en la gestión de datos.

# Memoria Caché - Concepto



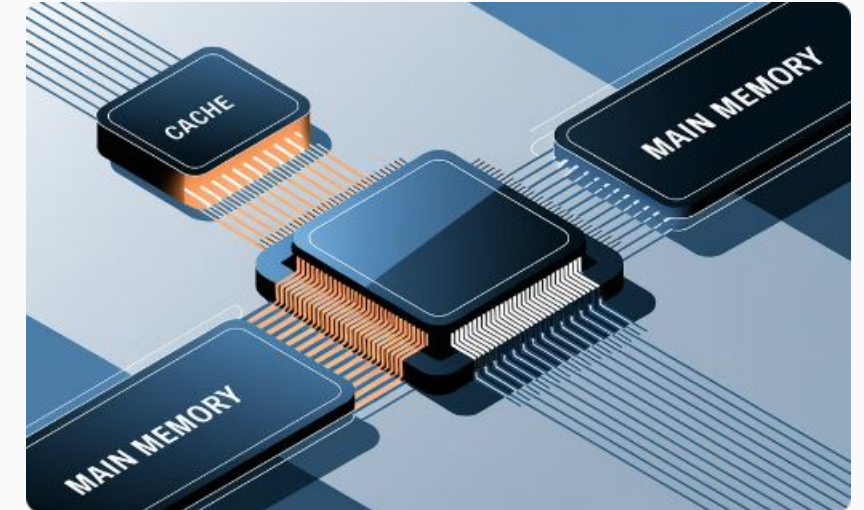
## Buffer de Alta Velocidad

La caché actúa como un **buffer de alta velocidad** para la memoria principal, aprovechando el principio de **localidad** (la tendencia de los programas a acceder a un conjunto limitado de ubicaciones de memoria durante un período de tiempo).



## Almacenamiento de Copias

Almacena **copias de datos** de la memoria principal que se utilizan con frecuencia. Cuando el CPU necesita acceder a un dato, primero verifica si una **copia del mismo está presente en la caché**.



## Reducción de Latencia

Objetivo: **reducir la latencia de acceso a la memoria**. Al proporcionar acceso más rápido a los datos utilizados con frecuencia, la caché **disminuye el tiempo promedio que tarda el CPU en acceder a la memoria**, lo que mejora significativamente el rendimiento general del sistema.

# Memoria Caché - Funcionamiento Básico

## Golpe de Caché (Cache Hit)

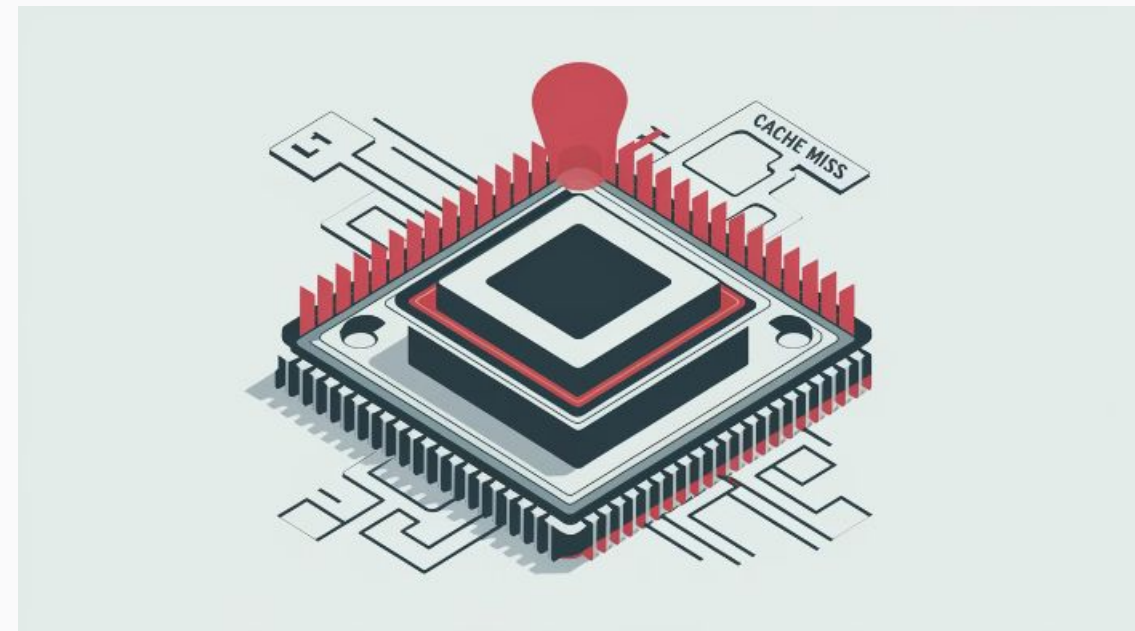
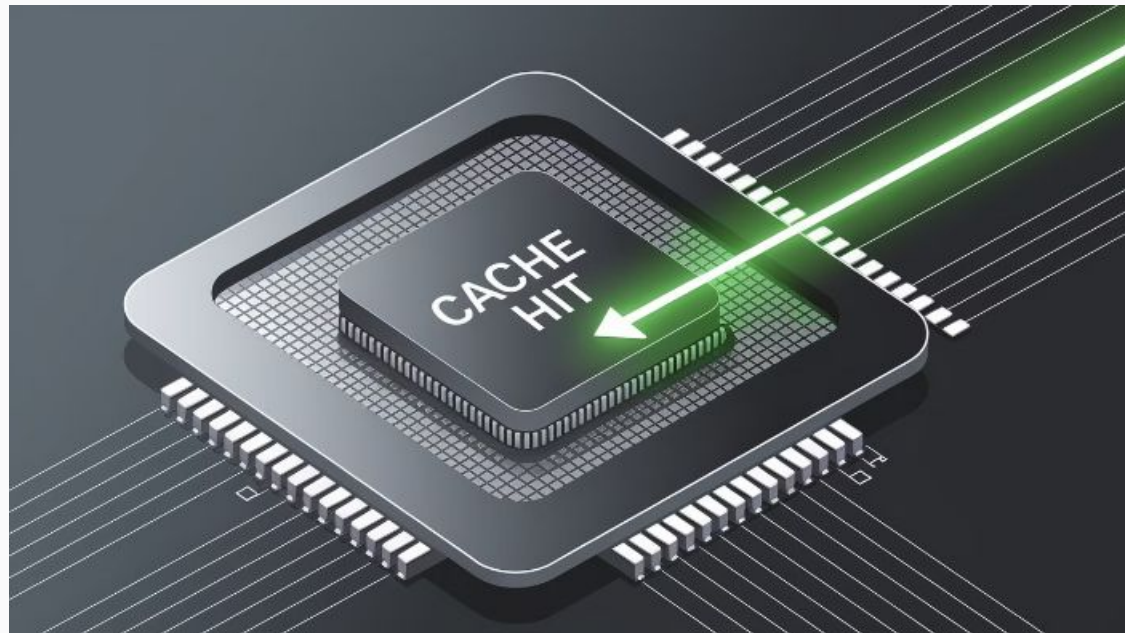
Cuando el CPU solicita un dato y lo encuentra en la caché, se produce un **golpe de caché**. El dato se entrega al CPU **rápidamente** desde la caché, sin necesidad de acceder a la memoria principal.

La **tasa de hits** (la proporción de accesos a memoria que son hits de caché) es un factor clave en el rendimiento del sistema.

## Fallo de Caché (Cache Miss)

Si el dato solicitado por el CPU no se encuentra en la caché, se produce un **fallo de caché**. En este caso, el CPU debe **acceder a la memoria principal** para obtener el dato, lo que implica una **latencia mucho mayor**.

Una vez que el dato se recupera de la RAM, se suele almacenar una copia en la caché para futuros accesos, basándose en las políticas de reemplazo.



# Memoria Caché - Organización (Mapeo)



## Mapeo Directo

En el mapeo directo, cada bloque de la memoria principal solo se puede mapear a una **única línea específica de la caché**.

Es la organización más simple pero puede sufrir de **conflictos** si se accede repetidamente a bloques de memoria que se mapean a la misma línea de caché.



## Mapeo Asociativo

En el mapeo asociativo, un bloque de la memoria principal se puede mapear a **cualquier línea de la caché**.

Esto ofrece mayor flexibilidad y reduce los conflictos, pero la lógica de búsqueda en la caché es más compleja y costosa.



## Mapeo Asociativo por Conjuntos

Este es un enfoque intermedio que divide la caché en un número fijo de **conjuntos**, y cada bloque de la memoria principal solo se puede mapear a una línea dentro de un **conjunto específico**.

Dentro de un conjunto, el mapeo es asociativo, ofreciendo un buen compromiso entre flexibilidad y complejidad. La caché se organiza como un número de conjuntos, cada uno conteniendo un número fijo de líneas. Un bloque de memoria principal se mapea a un conjunto específico basándose en una función de su dirección de memoria. Dentro del conjunto, el bloque se puede colocar en cualquier línea libre.

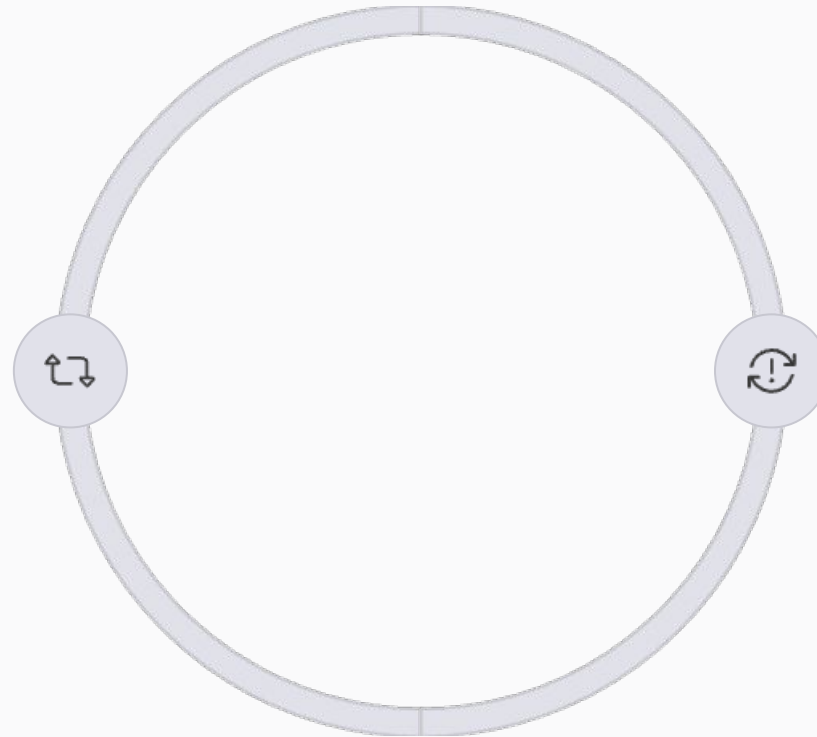
# Memoria Caché - Políticas

## Políticas de Reemplazo

Cuando una caché está llena y necesita almacenar un nuevo bloque, es necesario reemplazar uno de los bloques existentes. Las políticas de reemplazo determinan **qué bloque se debe desalojar**.

**LRU (Least Recently Used):** reemplaza el bloque que no se ha utilizado durante más tiempo

**FIFO (First-In, First-Out):** reemplaza el bloque que se cargó primero



## Políticas de Escritura

Cuando el CPU escribe datos en una ubicación de memoria que también está presente en la caché, es necesario mantener la coherencia entre la caché y la memoria principal. La política de escritura determina **cuándo y cómo se actualiza la memoria principal**.

**Write-Through:** cada escritura a la caché también se escribe inmediatamente en la memoria principal

**Write-Back:** las escrituras solo se realizan en la caché y la memoria principal se actualiza solo cuando el bloque modificado se reemplaza de la caché

# Memoria de Acceso Directo (RAM) - Concepto



Memoria Principal del Computador  
La RAM es la **memoria de trabajo** del computador, donde se almacenan temporalmente los datos y las instrucciones que los programas necesitan durante su ejecución. Su tamaño influye directamente en la cantidad de programas y datos que pueden estar activos simultáneamente.



Acceso Aleatorio  
Una característica fundamental de la RAM es que permite el **acceso aleatorio** a cualquier ubicación de memoria. Esto significa que el tiempo necesario para acceder a cualquier dato en la RAM es constante, independientemente de su ubicación física.

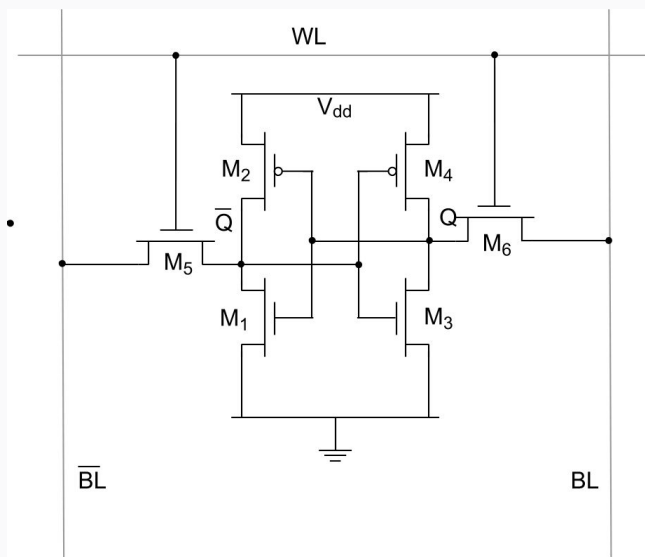


Volátil  
A diferencia del almacenamiento secundario, la RAM es **volátil**. Esto significa que los datos almacenados en la RAM se pierden cuando se interrumpe el suministro de energía a la computadora (es decir, al apagarla). Por lo tanto, es necesario el almacenamiento secundario para guardar los datos de forma permanente.

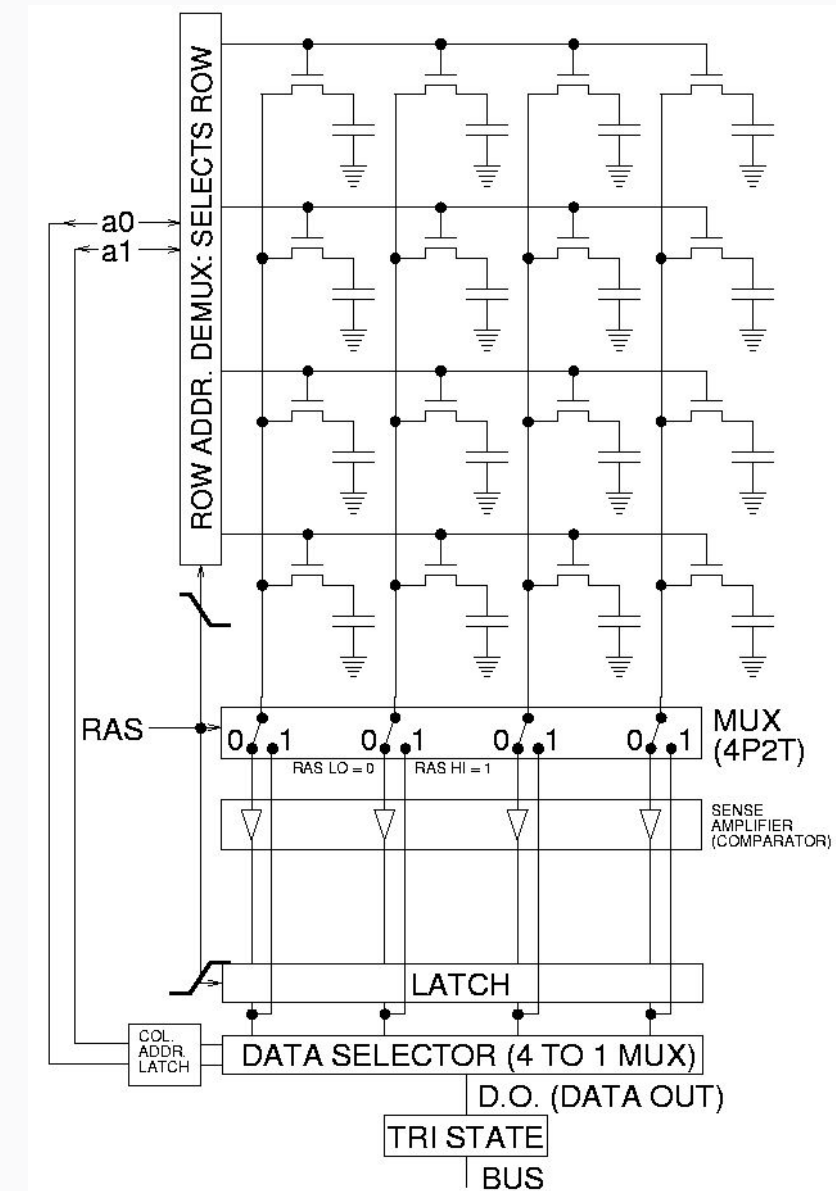
# Memoria de Acceso Directo (RAM) - Tipos

**DRAM** (Dynamic RAM): La DRAM es el tipo más común de RAM utilizado en las computadoras personales. Almacena cada bit de datos en un condensador dentro de una celda de memoria, que necesita ser **recargado (refrescado)** periódicamente para mantener los datos. Es relativamente barata y ofrece alta densidad de almacenamiento.

**SRAM** (Static RAM): La SRAM utiliza flip-flops para almacenar cada bit de datos y **no necesita ser refrescada**. Es significativamente **más rápida y costosa** que la DRAM y se utiliza principalmente para la memoria caché debido a su velocidad.



Evolución: A lo largo del tiempo, la tecnología DRAM ha evolucionado significativamente, dando lugar a diferentes generaciones como **DDR (Double Data Rate), DDR2, DDR3, DDR4 y la más reciente DDR5**. Cada nueva generación ofrece mayores velocidades de transferencia de datos y menor consumo de energía.



# Memoria de Acceso Directo (RAM) - Funcionamiento

## Gestión por el SO

El sistema operativo es responsable de **administrar la memoria principal**, asignando bloques de RAM a los procesos que los necesitan y liberándolos cuando los procesos terminan o ya no requieren esa memoria. Esta gestión eficiente de la memoria es crucial para el rendimiento y la estabilidad del sistema.

## Carga de Programas

Antes de que un programa pueda ser ejecutado por el CPU, sus **instrucciones y los datos que necesita deben cargarse desde el almacenamiento secundario a la memoria RAM.**

## Almacenamiento Temporal

Durante la ejecución de un programa, la RAM sirve como un **almacén temporal para los datos con los que el CPU está trabajando activamente y las instrucciones que está ejecutando.** El CPU accede a la RAM (directamente o a través de la caché) para obtener las instrucciones y los datos necesarios para realizar las operaciones.

# Memoria de Acceso Directo (RAM) - Relación con la Jerarquía

2

Nivel en la Jerarquía

En la jerarquía de memoria, la RAM se sitúa **inmediatamente por debajo de la caché**. Es el siguiente nivel de memoria al que el CPU accede cuando un dato no se encuentra en la caché.

10X

Latencia vs. Caché

Cuando se produce un **fallo de caché**, el CPU debe **acceder a la memoria RAM** para recuperar el bloque de datos o la instrucción solicitada. La latencia de acceso a la RAM es significativamente mayor que la latencia de acceso a la caché, por lo que una alta tasa de fallos de caché puede degradar el rendimiento del sistema.

8-32GB

Capacidad Típica

La **cantidad de memoria RAM instalada en un computador** tiene un impacto directo en su rendimiento general. Una cantidad insuficiente de RAM puede llevar a una situación de **"paging"** o **"swapping"**, donde el sistema operativo mueve datos entre la RAM y el almacenamiento secundario (que es mucho más lento) para liberar memoria, lo que puede ralentizar significativamente el sistema.