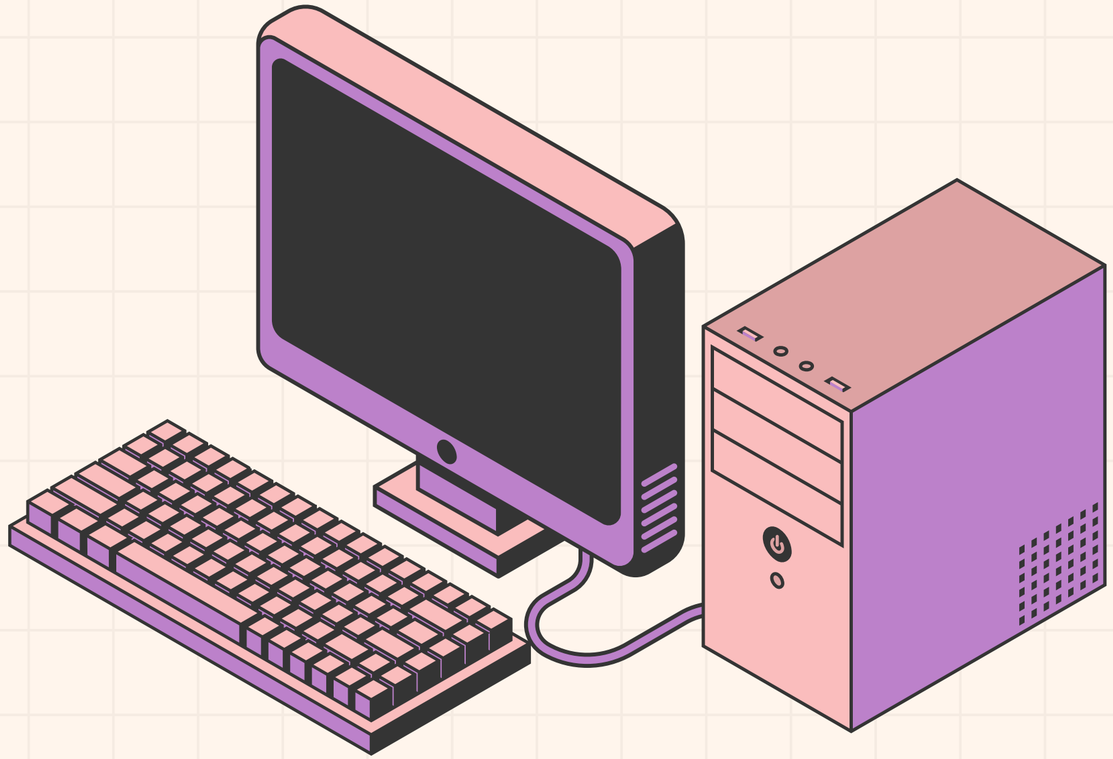
IIC2343
ARQUITECTURA

HOME SERVICE ABOUT US CONTACT US

# CACHÉ

## JERARQUÍA DE MEMORIA

**Joaquín Peralta**

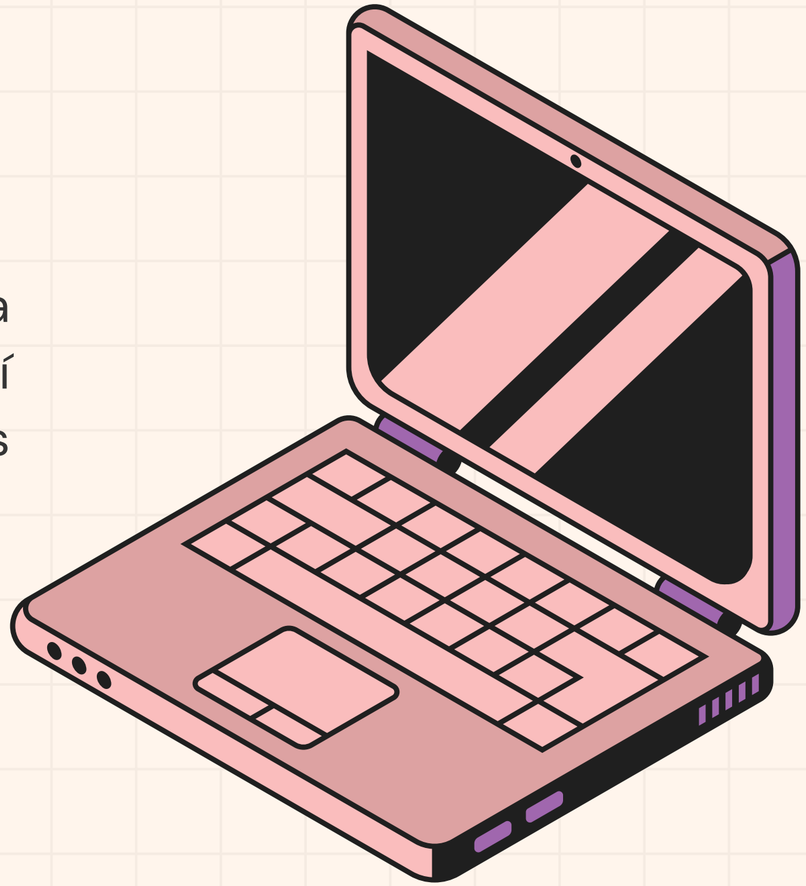


IIC2343
ARQUITECTURA

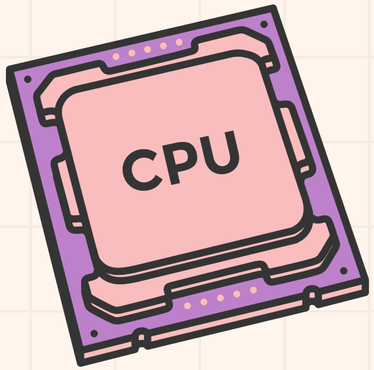
HOME SERVICE ABOUT US CONTACT US

# ACCESOS A MEMORIA

La motivación para crear una jerarquía de memoria es para poder **agilizar la obtención de datos** y así reducir los tiempos de ejecución general en las instrucciones

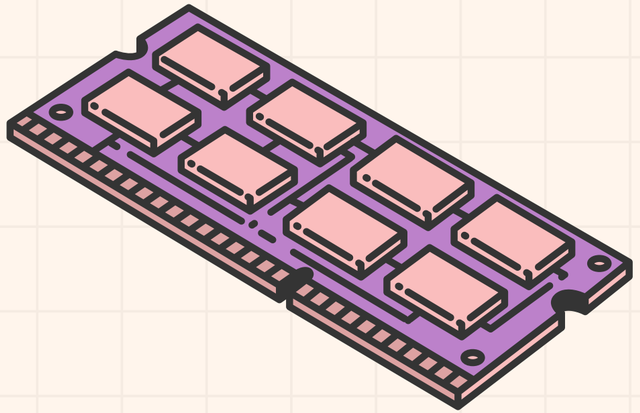


# PRINCIPIOS DE LOCALIDAD




**TEMPORAL**

Es probable que un dato obtenido en memoria sea usado posteriormente en otra operación

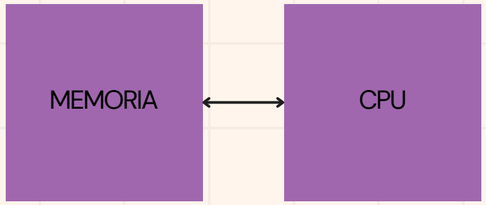


**ESPACIAL**

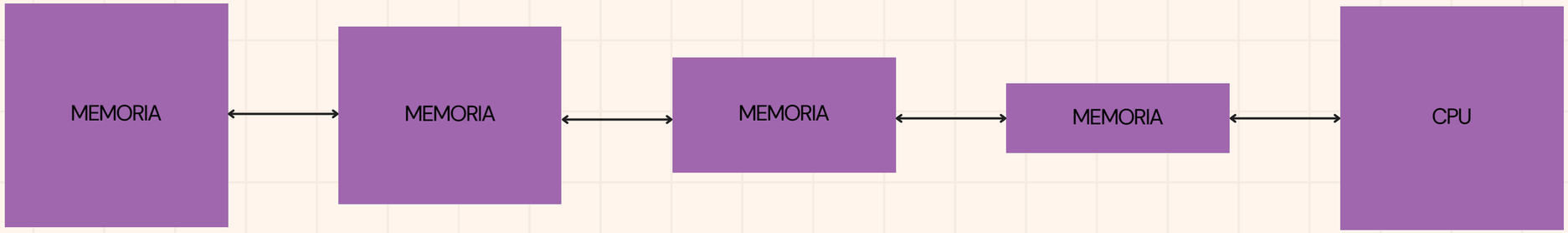
Es probable que datos cercanos al buscado también sean usados (bloque de memoria)

# JERARQUÍA DE MEMORIA

REALIZAMOS UNA TRANSFORMACIÓN DE ESTE ESQUEMA




**A ESTE ESQUEMA**




**POR EJEMPLO** HDD SSD DRAM SRAM

Ojo: CPU no sabe que caché existe, dado que ahora se comunica con el **controlador de Caché**

# JERARQUÍA DE MEMORIA
## EVALUACIÓN

- **Hit**: Búsqueda exitosa de un dato en la memoria caché.
- **Miss**: Búsqueda fallida de un dato en la memoria caché.
- **Hit-rate (HR)**: #Hits ÷ #Accesos a memoria
- **Miss-rate (MR)**: 1 - HR
- **Hit-time (HT)**: Tiempo que toma buscar un dato en memoria caché, independiente de si se encuentra o no.
- **Miss-penalty (MP)**: Tiempo que toma, luego de un miss en caché, copiar un bloque de memoria del siguiente nivel en la jerarquía en la caché y luego acceder al dato buscado desde esta.

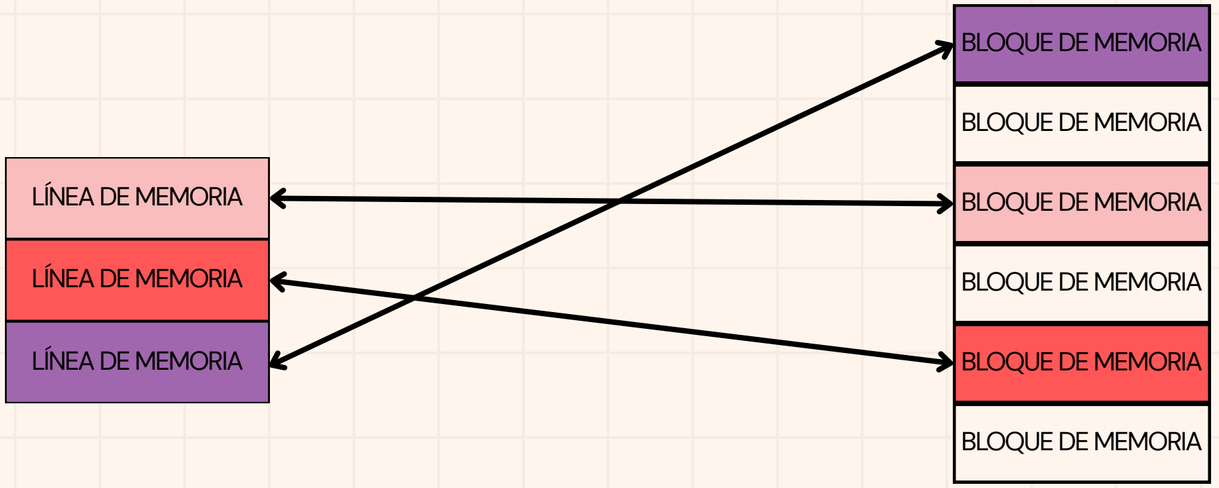
$$TP = HR \cdot HT + (1 - HR) \cdot (HT + MP)$$

# MEMORIA CACHÉ
# CARACTERÍSTICAS

- Es el **primer nivel** de la jerarquía de memoria (más cercano a CPU)
- La CPU no sabe que la memoria caché existe. Un **controlador de caché** se encarga de su manejo
- **Se divide en líneas de memoria** (la división física de la caché), las cuales se asocian a bloques de memoria de la memoria principal.
- Cada línea de la caché se rige por un **bit de validez**, que indica si el contenido es válido o no
- La CPU solo conoce el tamaño de la memoria principal (MP)

# MEMORIA CACHÉ
## BLOQUES Y LÍNEAS DE MEMORIA

- Las **líneas** y los **bloques** de la caché deben ser del **mismo tamaño** que los bloques de la memoria principal (MP)
- Al acceder a un dato, si no está en la caché, se va a la MP y se extrae todo el bloque que contiene al dato, lo cual respeta el principio de localidad espacial



# MEMORIA CACHÉ
# CONTROLADOR DE CACHÉ

Se encarga de la comunicación y coordinación de la información. Posee dos funciones principales:

- **Mecanismo de acceso de datos**: Funciones de correspondencia y Políticas de reemplazo
- **Política de escritura**: Write-through y Write-back

# **CONTROLADOR DE CACHÉ**
# FUNCIONES DE CORRESPONDENCIA

Es la asociación entre un bloque de memoria y una línea de la caché, lo cual depende de:

- Tamaño de bloque y línea
- Cantidad de líneas en la caché
- Tamaño de la memoria principal (MP)

Además, cada función utilizará un **tag**

# FUNCIONES DE CORRESPONDENCIA

## TAG

Es el **identificador único** que nos permite identificar si el dato de una dirección de la MP **se encuentra en caché o no.** Es el dato que revisamos al realizar las búsquedas en los accesos.

**El cómo** se compone el tag dependerá del tipo de función de correspondencia a utilizar

STUDIO SHODWE



# FUNCIONES DE CORRESPONDENCIA



## DIRECTLY MAPPED

Cada bloque de memoria se asocia **únicamente** a una línea de la caché.

**TAG | ID LINEA | OFFSET**



## FULLY ASSOCIATIVE

Cada bloque de memoria se puede asociar a **cualquier línea de caché**

**TAG | OFFSET**



## N-WAY ASSOCIATIVE

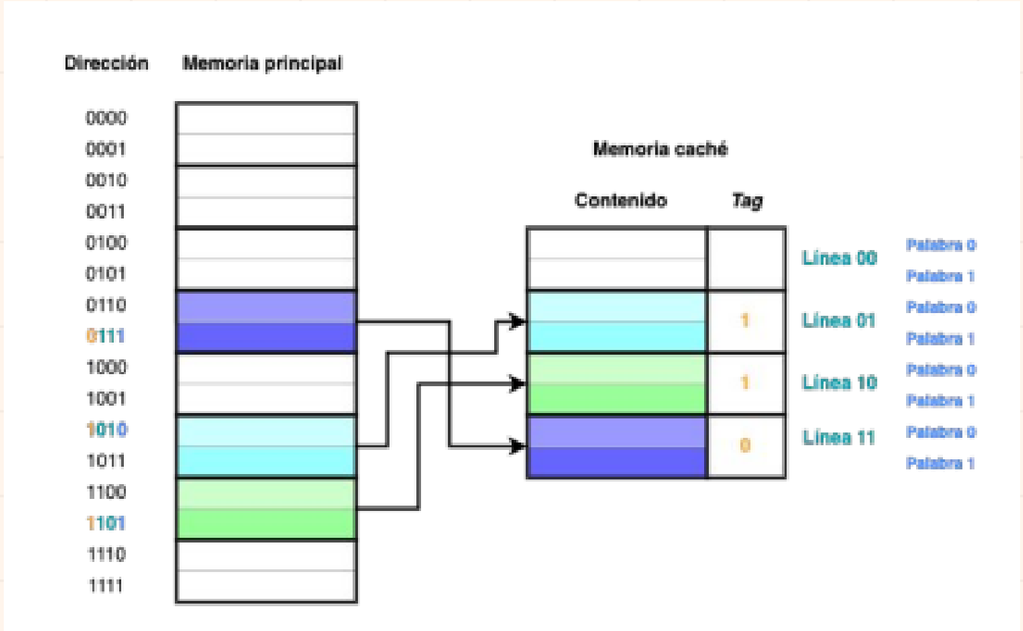
Cada bloque de memoria se asocia a un **conjunto de líneas de la caché**

**TAG | ID CONJUNTO | OFFSET**

# **DIRECTLY** MAPPED

- Cada bloque de MP se asigna a una **única linea** de caché
- **Fórmula**:
  - n°línea = n°bloque MOD #líneas
- **Tag**: Se obtiene de los bits restantes tras determinar offset e índice de línea

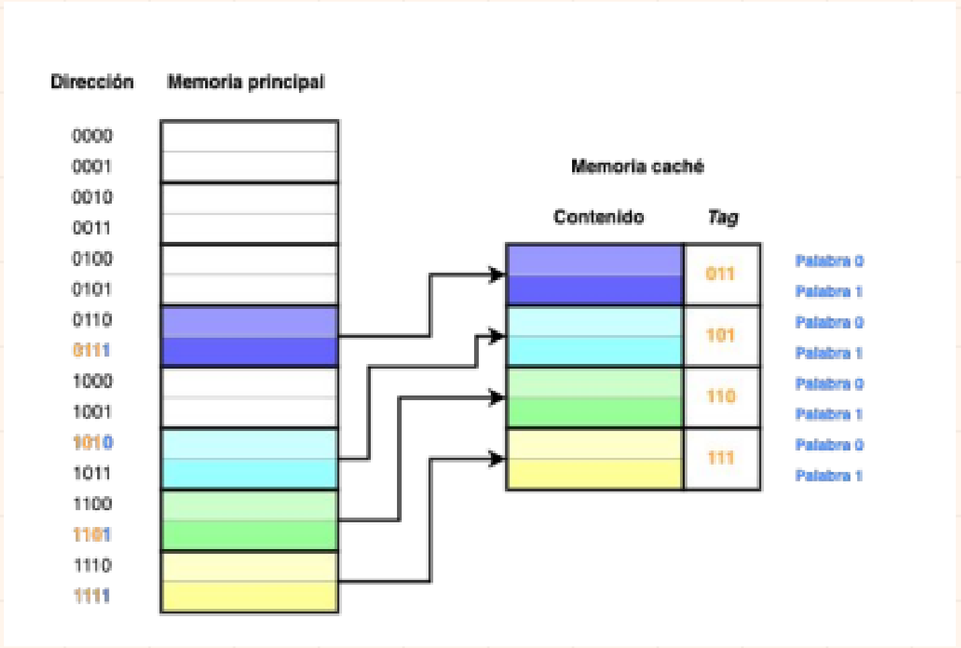
**Ventajas**: Simple y rápido
**Desventajas**: Alta tasa de conflictos, si varios bloques caen en la misma línea, se reemplazan constantemente



# FULLY ASSOCIATIVE

- Cada bloque de MP puede ir a cualquier línea de la caché
- Solo se necesita el tag
- Al buscar, se compara el tag en TODAS las líneas

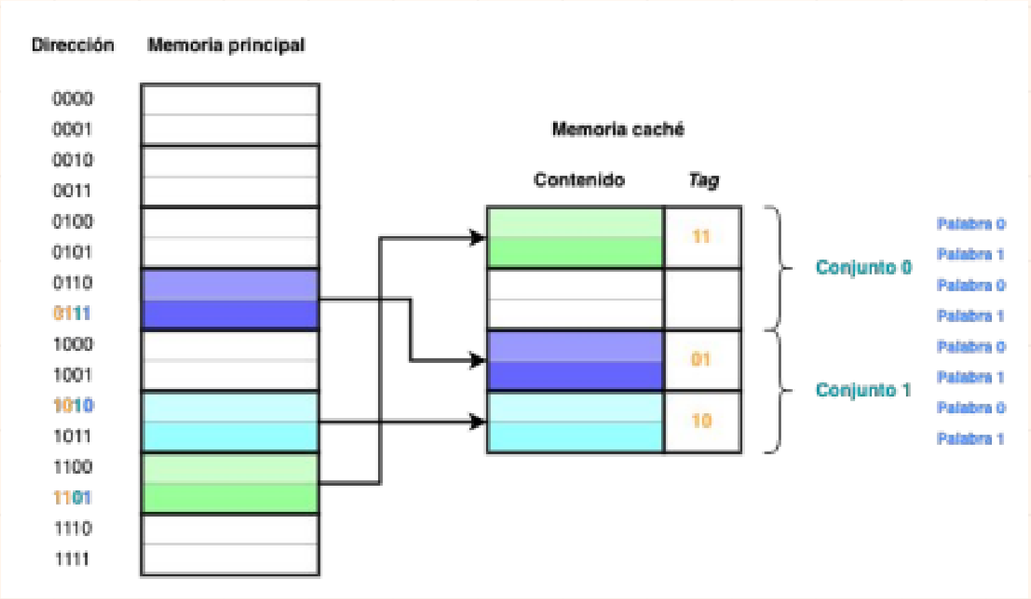
**Ventajas**: Uso óptimo de la caché (menor tasa de conflictos)
**Desventajas**: Más lento, se debe buscar en todas las líneas de la caché



# N-WAY ASSOCIATIVE

- Caché dividida en **conjuntos de N líneas**.
- Cada bloque se asigna a un conjunto específico (Directly Mapped), pero dentro del conjunto puede ir a cualquiera de sus líneas (Fully Associative)
- **Fórmulas**:
  - n°conjunto = n°bloque MOD #conjuntos
  - El tag se calcula con los bits restantes tras offset e índice de conjunto.

**Ventajas:** Buen balance entre rendimiento y complejidad, muetra menor tasa de conflictos que Directly Mapped, y más eficiente que Fully.
**Desventajas**: Complejidad intermedia en hardware.



# CONTROLADOR DE CACHÉ
## POLÍTICAS DE REEMPLAZO

**Bélády**: Se saca el bloque que se utilizará más lejos en el futuro. Ideal no alcanzable en la práctica.

**FIFO**: El primer bloque en entrar es el primero en salir

**LRU**: El bloque con mayor tiempo sin accesos se quita

**Random**: Es mejor que FIFO, pero peor que LRU

# CONTROLADOR DE CACHÉ
# POLÍTICAS DE ESCRITURA

Cuando el programa escribe datos en memoria, también debemos actuar.
Entonces, en este caso el controlador de caché puede usar:

- **Write-through**: El bloque modificado se escribe de inmediato en la MP, menos conveniente porque toma tiempo adicional en escritura.
- **Write-back**: El bloque modificado se escribe en la MP solo cuando su línea en caché es sustituida, mejora los tiempos, pero se pueden generar *problemas de sincronización* (habrá que hacerse cargo)

# MEMORIA CACHÉ
## TIPOS DE MEMORIA

Al existir Von Neumann y Harvard, surge la necesidad de tener dos tipos de memoria:

- **Caché unified**: Almacena datos e instrucciones, tiene la desventaja que probablemente su HR sea inferior al mezclar dos distribuciones de acceso juntas. (Esta la usamos en harvard para la memoria de datos)
- **Caché split**: Caché con división interna para datos e instrucciones. Puede tener mejor HR, pero tiene hardware más complejo, por ende, un mayor *overhead* de acceso. (Esta la podemos usar en una arquitectura Von Neumann)

# MEMORIA CACHÉ
## CACHÉ MULTINIVEL

Así como está la jerarquía de memoria, también se puede aplicar la idea en una caché. Se define una jerarquía de caché con varios niveles con tamaños sucesivamente mayores. Suelen haber tres niveles: L1, L2, L3.

STUDIO
SHODWE



# YA CACHÉ

