


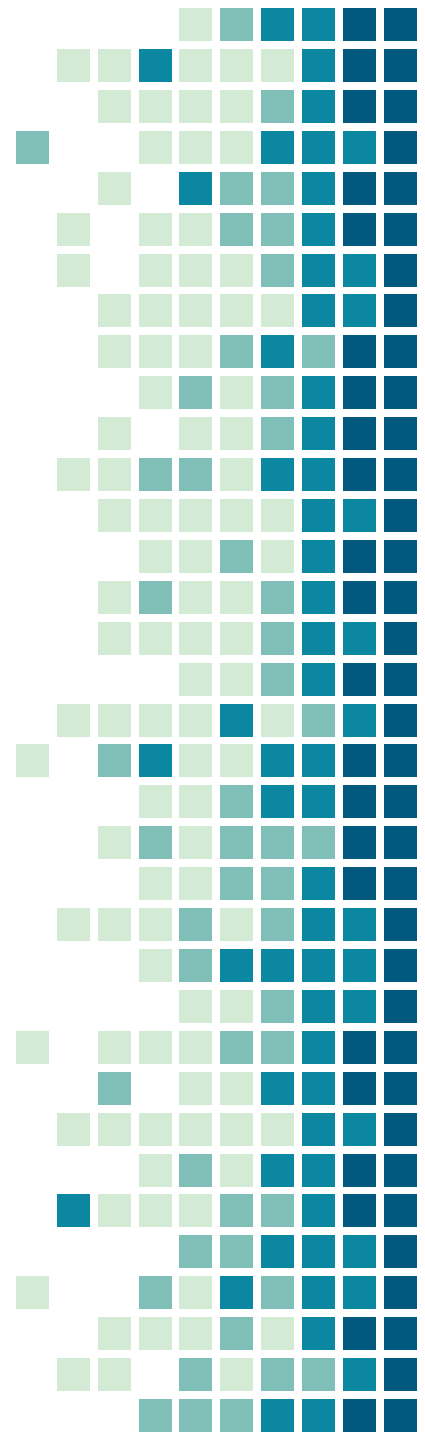
# Jerarquía de memoria



# Agenda

---

- 
- **Memoria.**
  - **Jerarquía de memoria:**
    - **Registros**
    - **Scratchpad**
    - **Caché**
    - **Memoria principal**
    - **Flash**



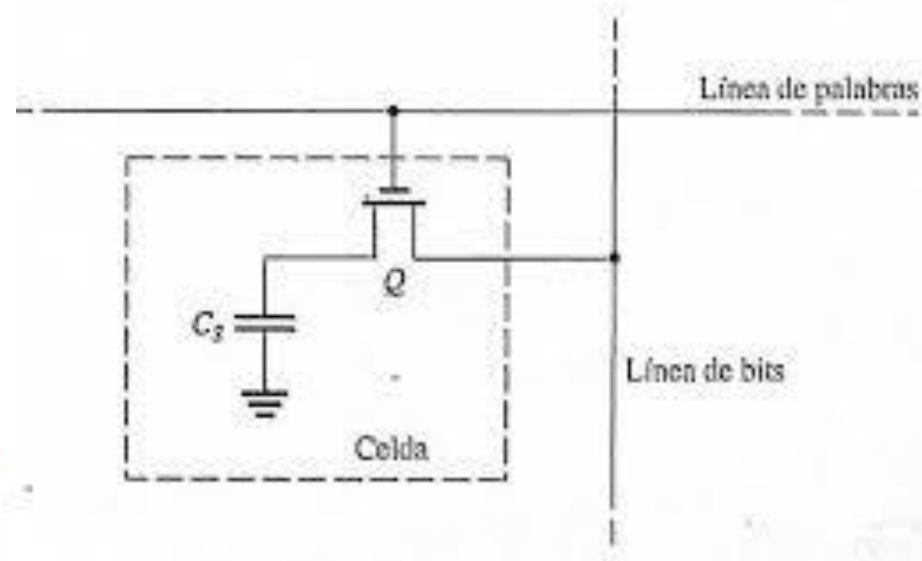
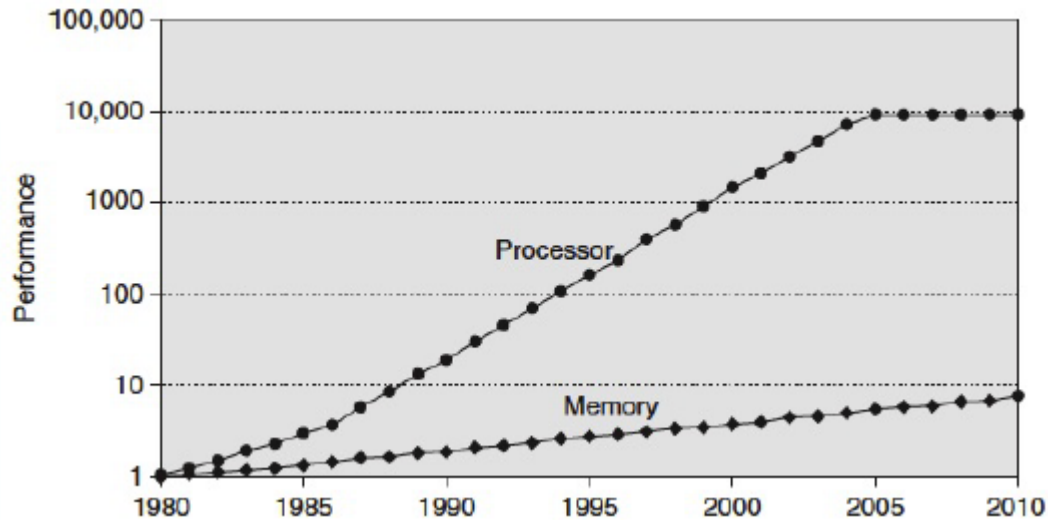
Pared de  
memoria



# ¿Por qué no se puede memorias más rápidas al igual que procesadores?

- **Efecto de pared de memoria.**
- ¿Cuál es la limitante?
- El capacitor no se puede hacer más pequeño porque si no se rompe el dieléctrico.
- Por condiciones físicas no se puede igualar la rapidez del procesador.

# ¿Por qué no se puede memorias más rápidas al igual que procesadores?



# Crecimiento de la DRAM

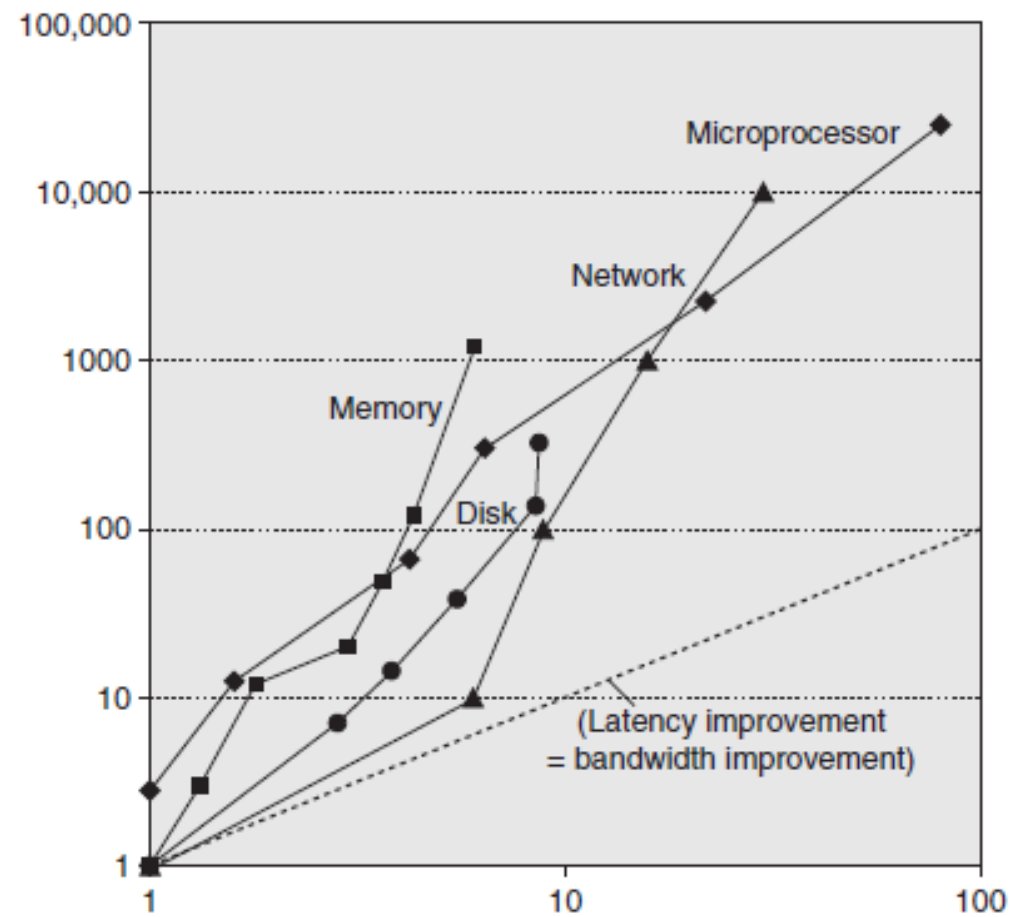
CA:AQA Edition	Year	DRAM growth rate	Characterization of impact on DRAM capacity
1	1990	60%/year	Quadrupling every 3 years
2	1996	60%/year	Quadrupling every 3 years
3	2003	40%–60%/year	Quadrupling every 3 to 4 years
4	2007	40%/year	Doubling every 2 years
5	2011	25%–40%/year	Doubling every 2 to 3 years

# Memoria Flash

- Su capacidad se dobla cada 2 años aproximadamente.
- Memoria no volátil.
- Especialmente utilizada en dispositivos móviles.
- 15 o 20 veces más barata que DRAM



# Comparación





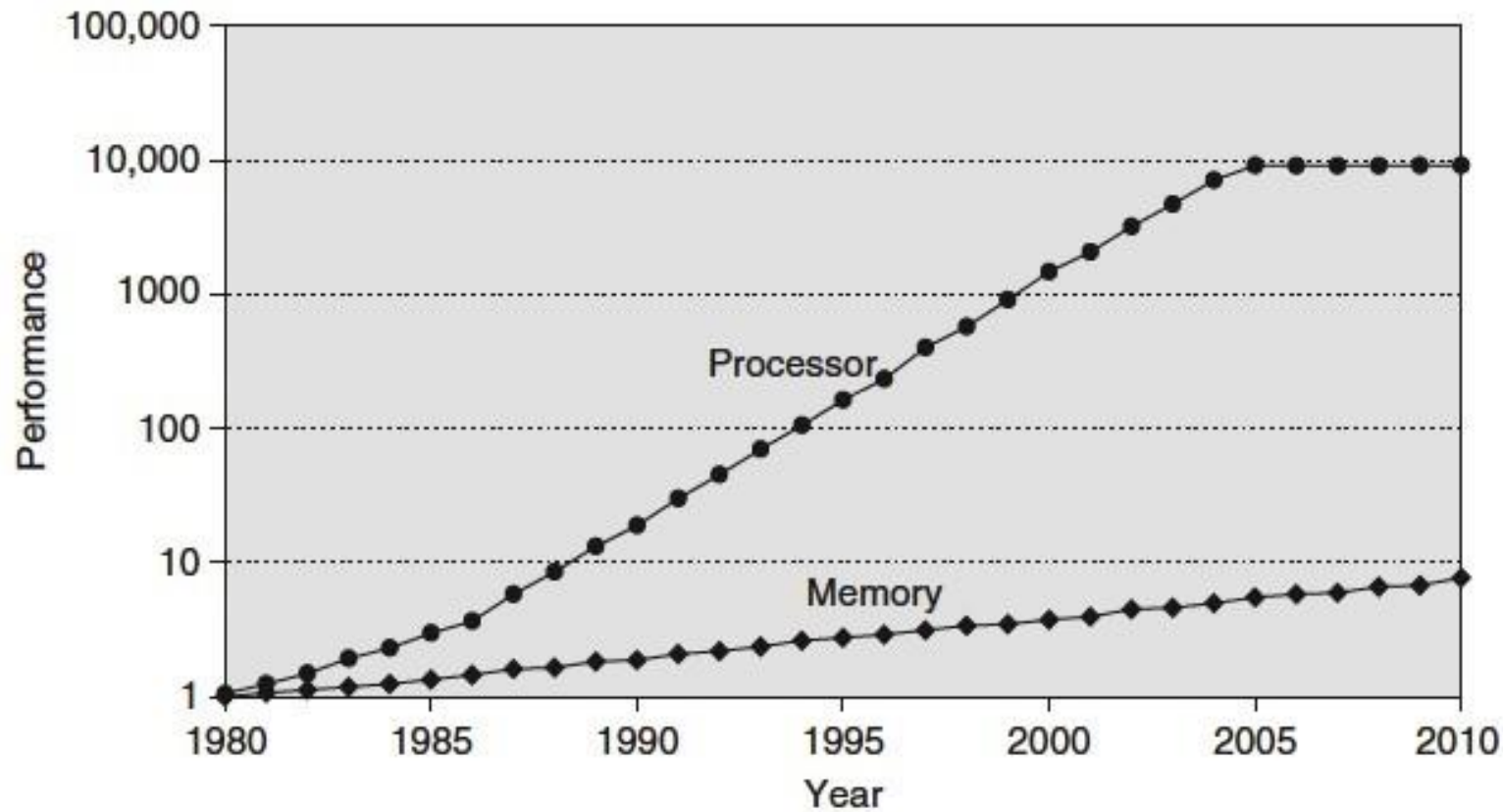
# Comparación

- Procesador 2014 CPU 3.9 GHz

Actividad	Tiempo	Escala humana
Ciclo CPU	0.256 ns	1 segundo
Caché L1	1.026 ns	4 segundos
Caché L2	3.077 ns	12 segundos
Caché L3	6.154 ns	24 segundos
RAM	8.4 ns	32 segundos
Disco duro- mejor caso	2.9 ms	132 días
Disco duro- peor caso	12 ms	1.5 años
SSD	85 us	3 días y 20 horas
Cambio de contexto	10 us	10.80 horas
Quantum	100 ms	12.4 años

**Problema:** Latencias en memoria son grandes comparadas con ciclos del procesador.

15 % de instrucciones son *lw/sw*. 1 acceso = cientos ciclos



Rendimiento de uP vs rendimiento de memoria RAM

# Memoria principal: DRAM

---

CA:AQA Edition	Year	DRAM growth rate	Characterization of impact on DRAM capacity
1	1990	60%/year	Quadrupling every 3 years
2	1996	60%/year	Quadrupling every 3 years
3	2003	40%–60%/year	Quadrupling every 3 to 4 years
4	2007	40%/year	Doubling every 2 years
5	2011	25%–40%/year	Doubling every 2 to 3 years

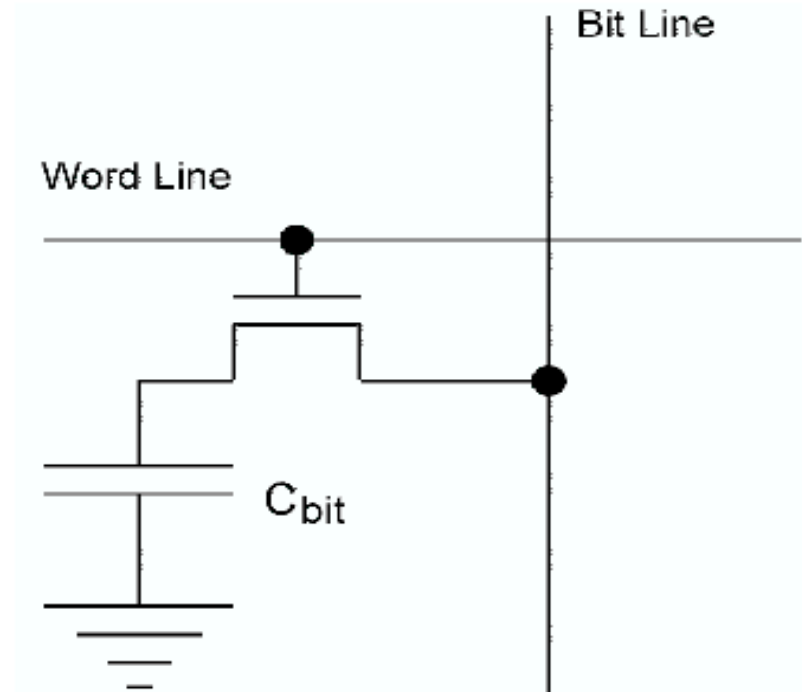
Recientemente la capacidad de las memorias RAM se ha ido incrementando en un 25-40 % por año. Sin embargo:



# Pared de memoria

Principal limitante: **tecnología**

- 🖨️ Área
- 🖨️ Costo
- 🖨️ Refrescamiento de memoria DRAM
- 🖨️ Tiempo de lectura/escritura relacionado con  $\tau=RC$



# Soluciones al problema de memoria

---



- **Aprovechar localidad.**

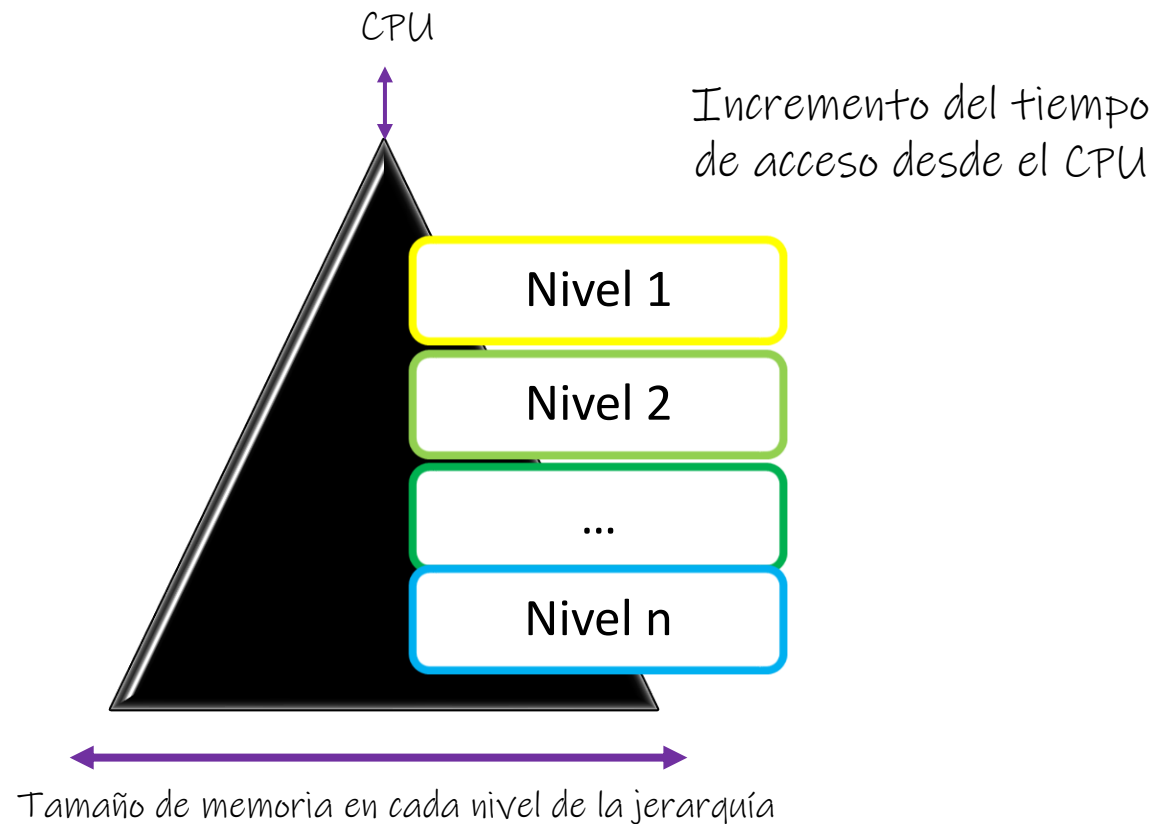
1. **Localidad temporal:** Cuando se hace referencia a un elemento de memoria, es probable que este elemento vuelva a ser referenciado pronto. Ej.: índices y variables dentro de ciclos.
2. **Localidad espacial:** Cuando se hace referencia a una posición de memoria, es probable que se haga referencia a posiciones cercanas. Ej.: Instrucciones, Arreglos.

- **Jerarquía:** Para aprovechar de mejor manera la localidad, se debe incluir niveles más bajos de memoria.

# Jerarquía de memoria

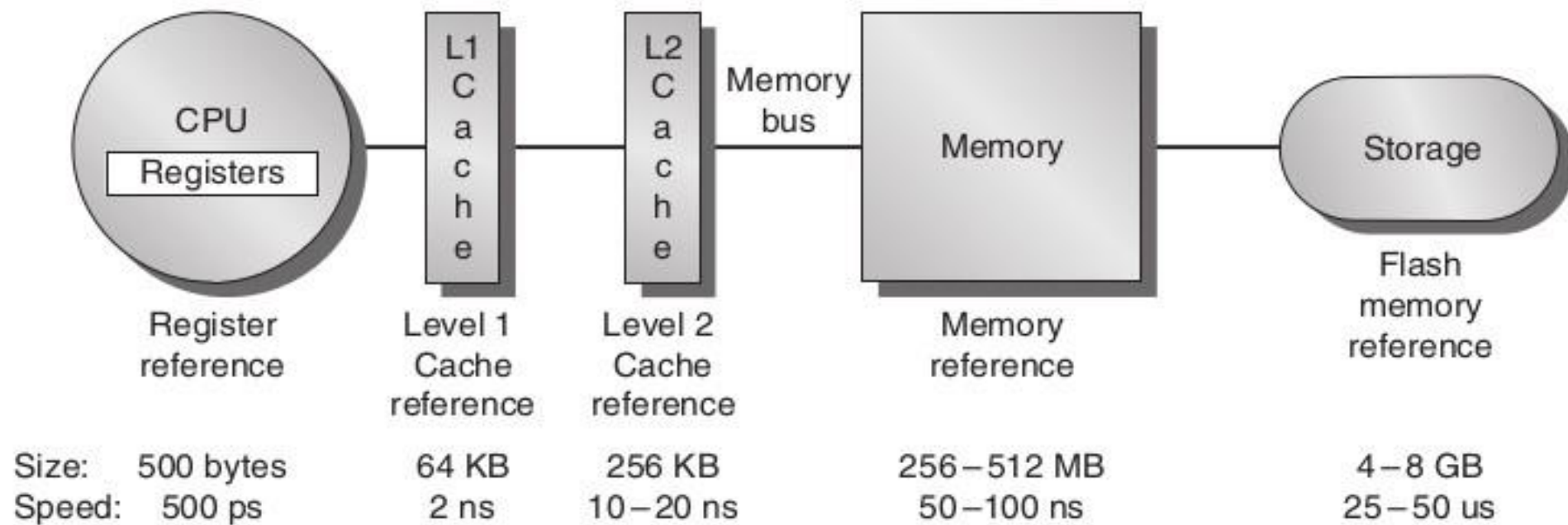
Brinda un **balance** entre rendimiento y costo

- ✎ Utilizar diferentes niveles de memoria con tamaño y velocidad diferente.
- ✎ **Objetivo:** proveer un tamaño de memoria de forma que se utilice tecnología barata, pero se tenga tecnología más rápida, de forma transparente.



# Jerarquía de memoria



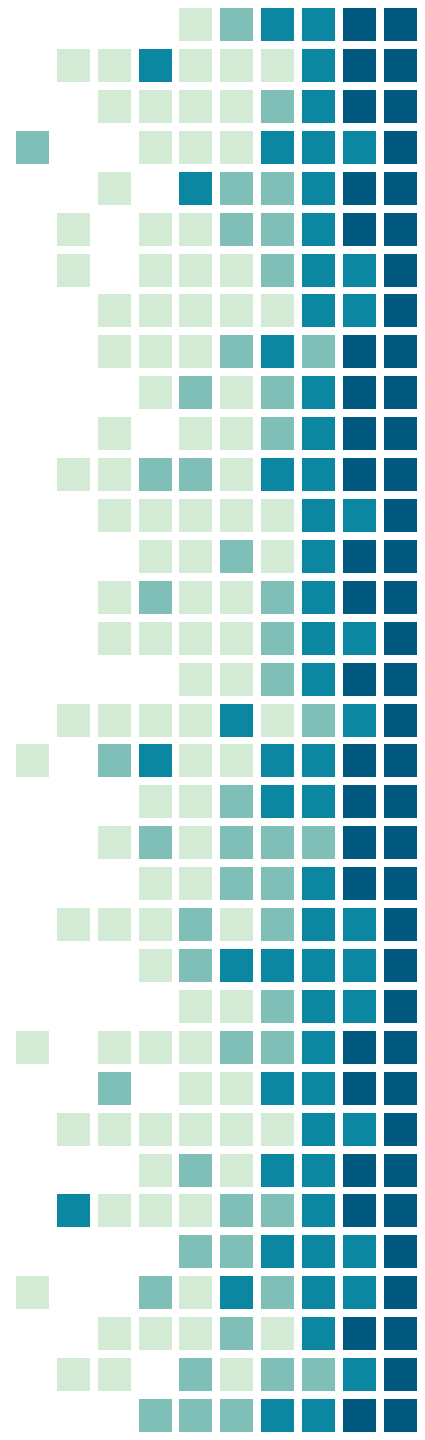




# Jerarquía de memoria

---

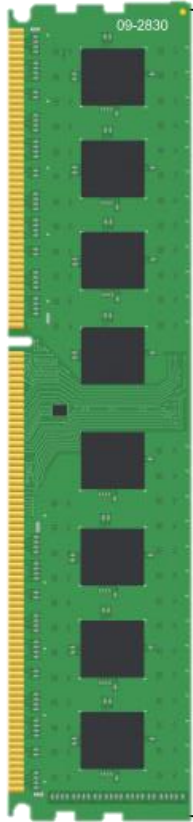
- ➡ Registros
- ➡ SPM - Scratchpad Memory
- ➡ Cache
- ➡ Memoria principal (DRAM)
- ➡ Memoria almacenamiento masivo: flash, disco duro ,etc.



# Registros

- Memoria de alta velocidad y poca capacidad, integrada al CPU en el microprocesador, que permite guardar valores muy usados.
- La cantidad de registros es pequeño no solo por su costo físico pero más importante su costo en bits dentro de una instrucción
- Pueden ser agrupados en un solo banco de memoria, usando tecnología SRAM.

# Memorias Scratchpad



Se encuentra tan cercana al CPU como la memoria Caché.

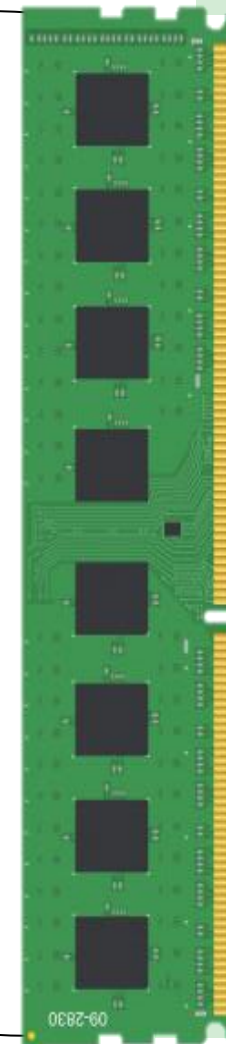
**Accesible** para el programador.

Tiene un conjunto de direcciones distinto a la memoria principal

Para aplicaciones con requisitos de tiempo-real, la Scratchpad puede garantizar la predictibilidad en los tiempos de acceso.

# Memoria caché

- Memoria de alta velocidad, cuyo objetivo es mejorar el rendimiento del CPU al realizar lecturas y escrituras desde y hacia la memoria principal.
- Cuando el CPU lee un dato por primera vez este es almacenado en la memoria caché. Es así como en la memoria caché se pueden encontrar los datos más recientemente utilizados por el CPU.
- Caché puede ser independiente para instrucciones (Caché-I) y para datos (Caché-D), organizada en niveles: L1, L2, etc.



# Conceptos básicos

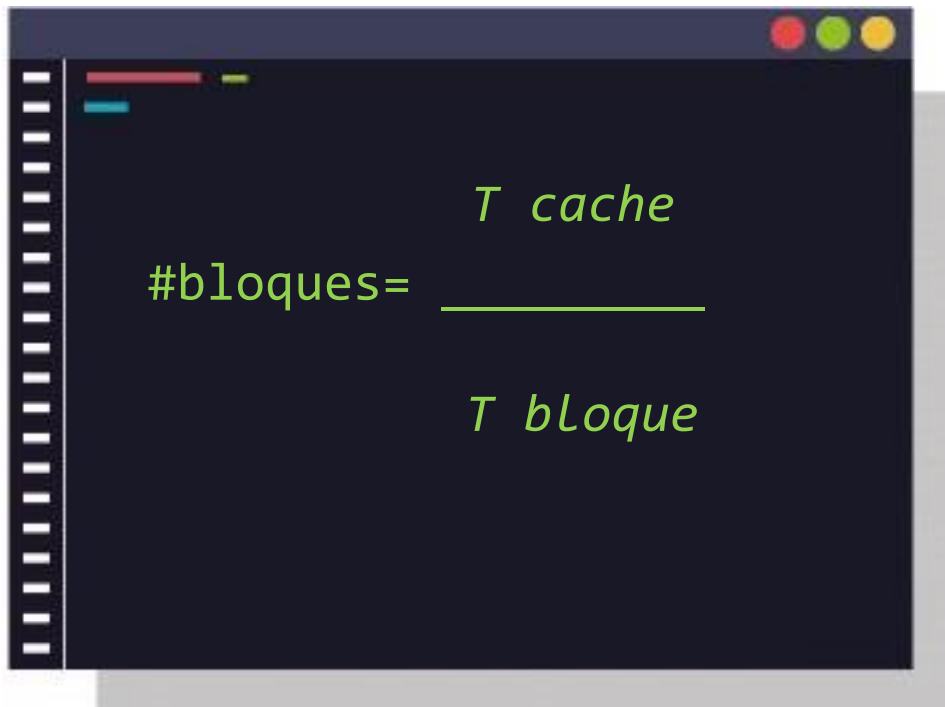
---

- ➡ Memoria caché es pequeña, rápida y de mayor costo que memoria principal.
- ➡ Cantidad de datos que se puede copiar en una sola escritura a caché se denomina **bloque** o **línea** de caché.
- ➡ Para explotar la localidad espacial, el bloque debe ser múltiplo del tamaño de palabra. Por ejemplo:
  - Tamaño de bloque 128 bits = 4 palabras de 32 bits.

# Conceptos básicos

---

- El número de bloques de caché está dado por:



Ejemplo:

- Tamaño de cache 64KB; tamaño de bloque 128 bits.
- # de bloques de caché: 4K = 4096

# Terminología de Caché

- 🧠 **Cache Hit:** Cuando el CPU busca un dato en la caché y este se encuentra ahí
- 🧠 **Cache Miss:** Cuando el CPU busca un dato en la caché y NO se encuentra o se encuentra inválido.
- 🧠 **Miss Penalty:** Penalización (tiempo que toma acceder a un nivel superior de memoria) producto de un miss de caché.
- 🧠 **Políticas de reemplazo:** Formas en las que la caché reemplaza bloques de memoria (caso miss)
- 🧠 **Correspondencia:** Método en que se mapean bloques de memoria principal a la memoria caché

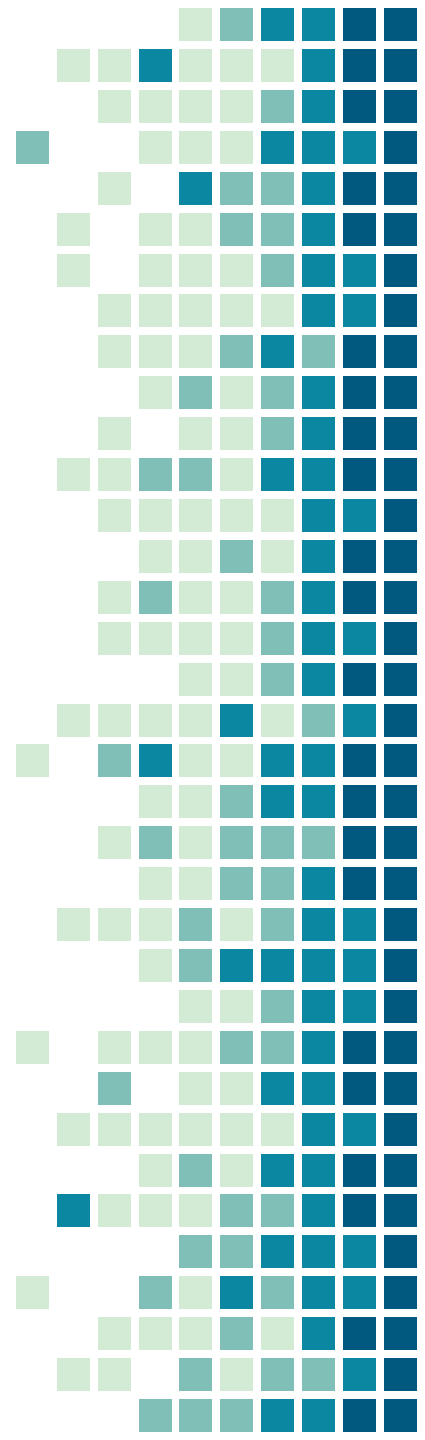
Directa (Direct mapped)  
Asociativa (Fully associative)  
Asociativa por set (n-way set associative)

# Terminología de caché

$$\text{Hit rate} = \frac{\#Hits}{\#accesos a memoria}$$

- Tasa de aciertos (Hit rate): número de accesos a memoria en los que los datos si se encuentran en caché.
- Tasa de desaciertos (Miss rate): número de accesos a memoria en los cuales los datos no encuentran en la caché

$$\text{Miss rate} = \frac{\#Misses}{\#accesos a memoria}$$





# Tiempo promedio de acceso a memoria\*

AMAT= Average Memory Access Time

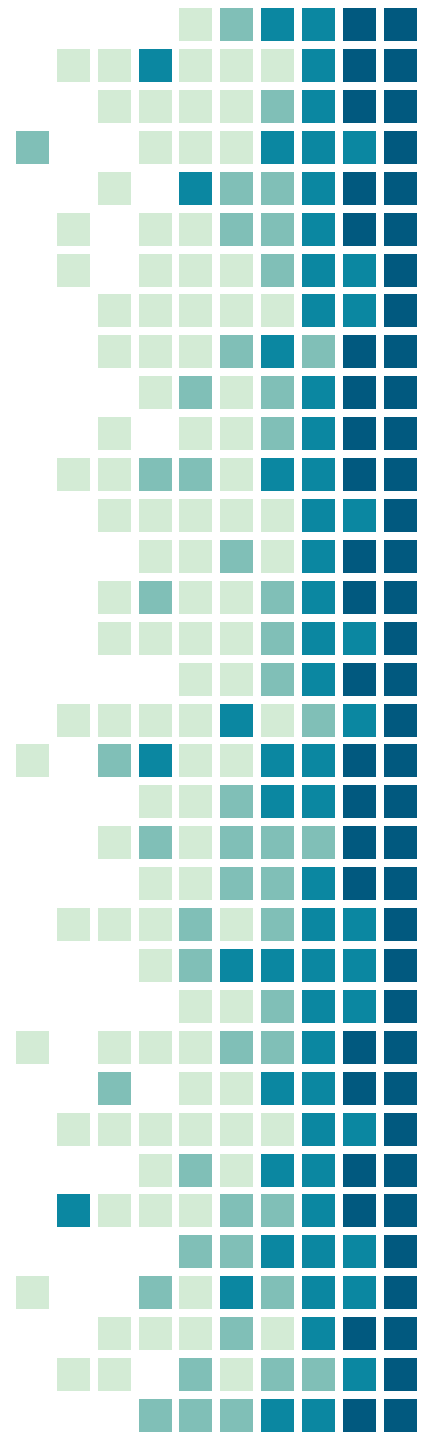
- $AMAT = \text{Hit rate} * \text{Hit time} + \text{Miss rate} * \text{Miss time}$

Donde

- $\text{Miss time} = \text{Hit time} + \text{Miss Penalty}$
- $\text{Hit rate} + \text{Miss rate} = 1$

Por lo que

- $AMAT = \text{Hit time} + \text{Miss rate} * \text{Miss penalty}$



# Ejemplo AMAT I

---

- Un computador posee una jerarquía de memoria dada por una cache L1 completamente asociativa y la memoria principal. Tomando en cuenta que el tiempo de acceso a la cache es de 1 ciclo y el de memoria es de 50 y por cada 100 accesos a memoria, se tienen 55 desaciertos en cache ¿Cuál es el tiempo promedio de acceso a memoria del sistema, en ciclos de reloj?

$$AMAT = \text{Hit time} + \text{Miss rate} * \text{Miss penalty}$$

$$AMAT = 1 + 0,55 * (50) = 28,5 \text{ ciclos}$$

# Ejemplo AMAT II

- Suponga que se tiene un computador con memoria principal RAM y con 2 niveles de cache unificada L1 y L2. Si el tiempo de acceso a las caches es 4 y 10 ciclos, respectivamente, la tasa de desaciertos en accesos a memoria es de 40 % para L1 y 15 % para L2 y el tiempo de acceso a memoria principal es de 100 ciclos, ¿Cuál es el tiempo promedio de acceso a memoria?

$$AMAT = \text{Hit time} + \text{Miss rate} * \text{Miss penalty}$$

$$AMAT = T_{aL1} + Mr_{L1} * (T_{aL2} + Mr_{L2} * (T_{aRAM}))$$

$$AMAT = 4 + 0,4 * (10 + 0,15 * (100)) = 14 \text{ ciclos}$$

# Estructura de memoria caché

---

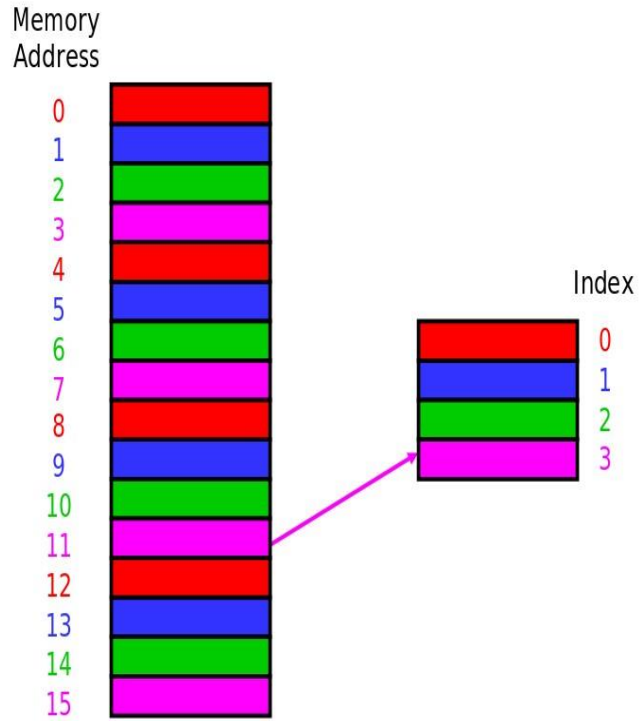
Una entrada en la memoria cache esta compuesta de tres partes:

- **Bit de validez (V):** indica si el bloque contiene datos validos o no. Al iniciar el sistema, todas las entradas son marcadas como invalidas. Una línea invalida implica en su lectura un miss de cache.
- **Etiqueta (tag):** contiene el valor que identifica la dirección de memoria (principal) correspondiente a los datos almacenados.
- **Datos:** Contiene copia de los datos (bloque o línea de cache)



# Caché: Correspondencia Directa

Cada bloque de la memoria principal está mapeado a un único bloque (línea) en la caché.



- $B_{cache} = B_{mem} \bmod Nb_{cache}$
- $B_{mem}$  : Bloque de memoria principal
- $B_{cache}$  : Bloque en caché
- $Nb_{cache}$  : Número de bloques de la caché

# Caché: Asociativa por Set

Cada bloque de memoria principal puede mapearse de **n** formas a la caché, donde **n** es el la asociatividad (n-way)

$$S_{cache} = B_{mem} \bmod NS_{cache}$$

$B_{mem}$  : Bloque de memoria principal

$S_{cache}$  : Set en caché

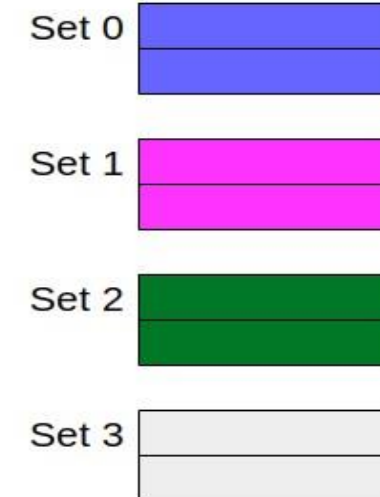
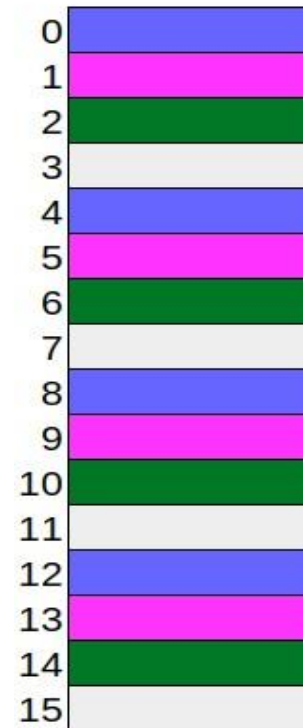
$NS_{cache}$  : Número de sets de la caché

$$NS_{cache} = NB_{cache} / n$$

$NB_{cache}$  : Número de bloques de la caché.

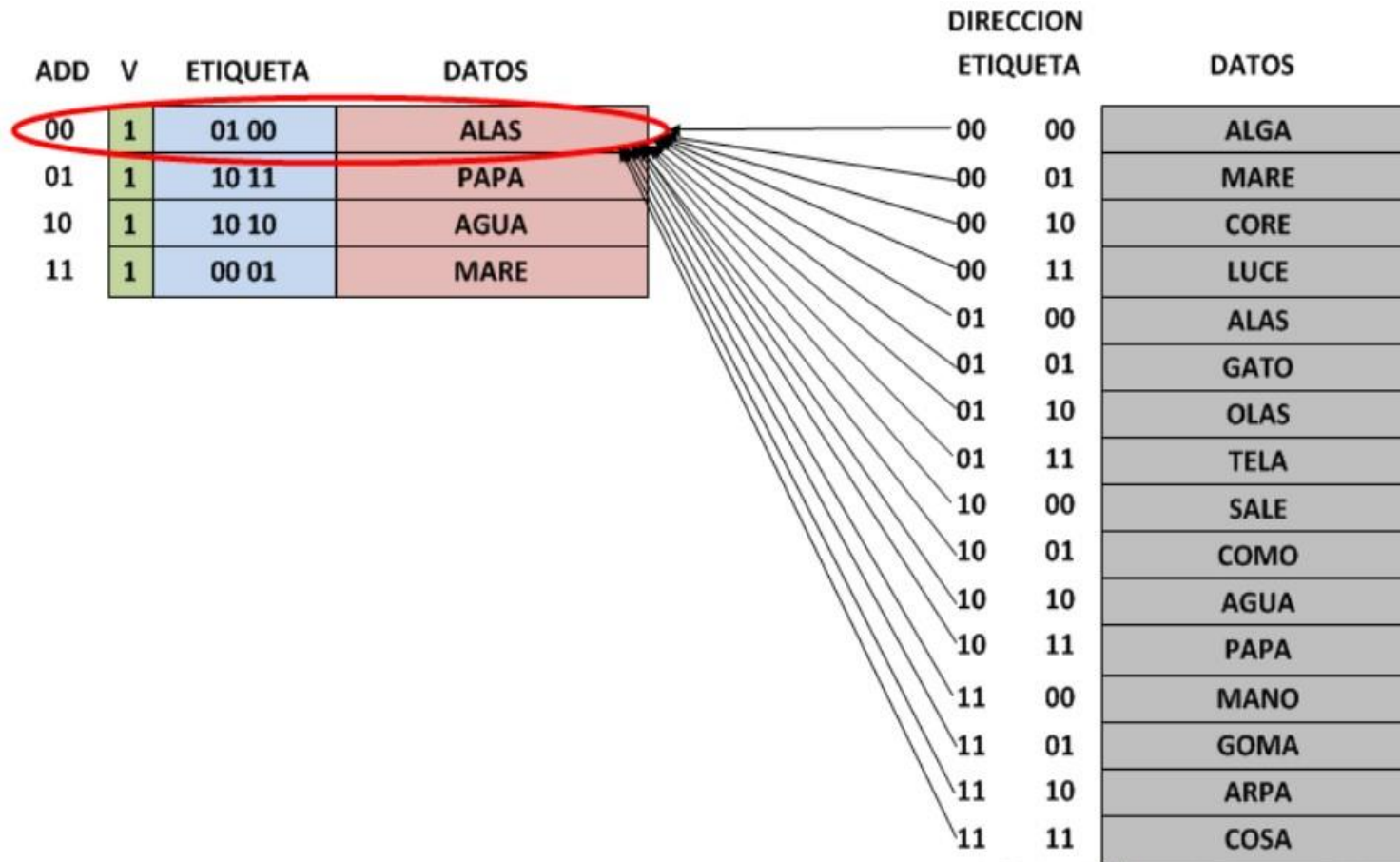
$n$ : Asociatividad

Dentro de un mismo set en caché,  
**cualquiera** de los bloques puede ser  
reemplazado



# Caché: Asociativa (completa)

Cada bloque de memoria puede ser mapeado a **cualquier** bloque de caché)

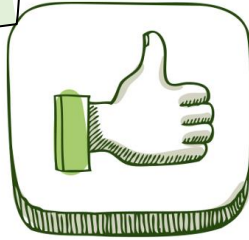


# Incremento de la asociatividad

---

## Ventaja

Reduce la tasa de desaciertos.



## Desventajas

Alto costo de implementación (más complejo identificar bloques/sets)

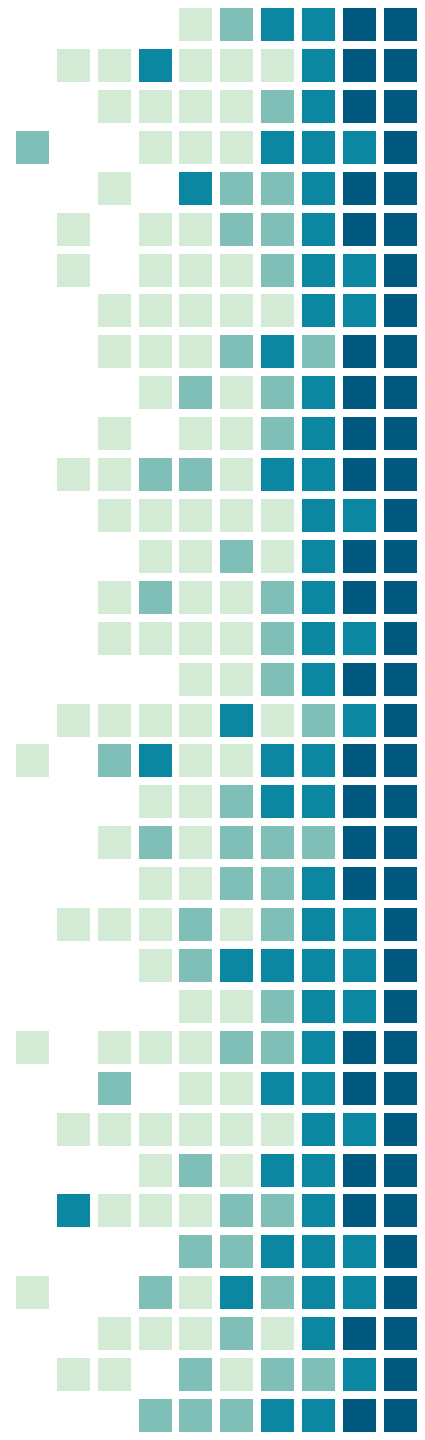
Incremento del tiempo de acierto (toma más tiempo identificar los sets/bloques)



# Políticas de reemplazo

---

- ✓ Caso **correspondencia directa**: Solo hay un posible candidato a reemplazar.
- ✓ Asociativa **por set**: Dentro del set correspondiente se debe seleccionar uno de los bloques.
- ✓ **Completamente** asociativa: Cualquier bloque es candidato



# Políticas de reemplazo

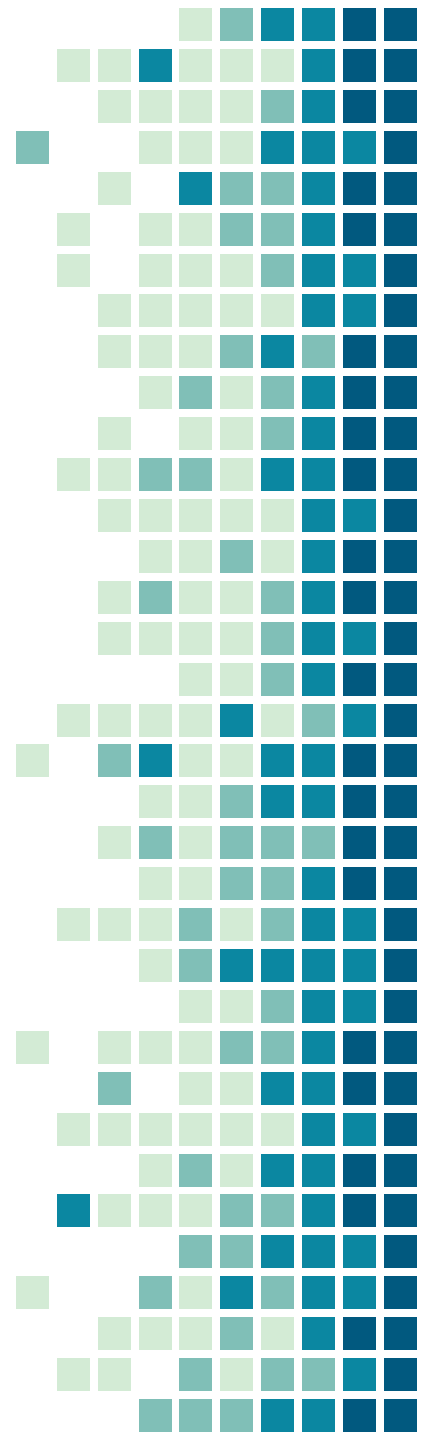
---

Las principales estrategias de reemplazo existentes son:

Random (al azar): entre los posibles bloques, se selecciona uno de manera aleatoria.

Menos usado recientemente (LRU): se reemplaza el bloque que haya sido utilizado menos recientemente

FIFO: Temporalmente, el primer bloque en ser escrito es el que será reemplazado.



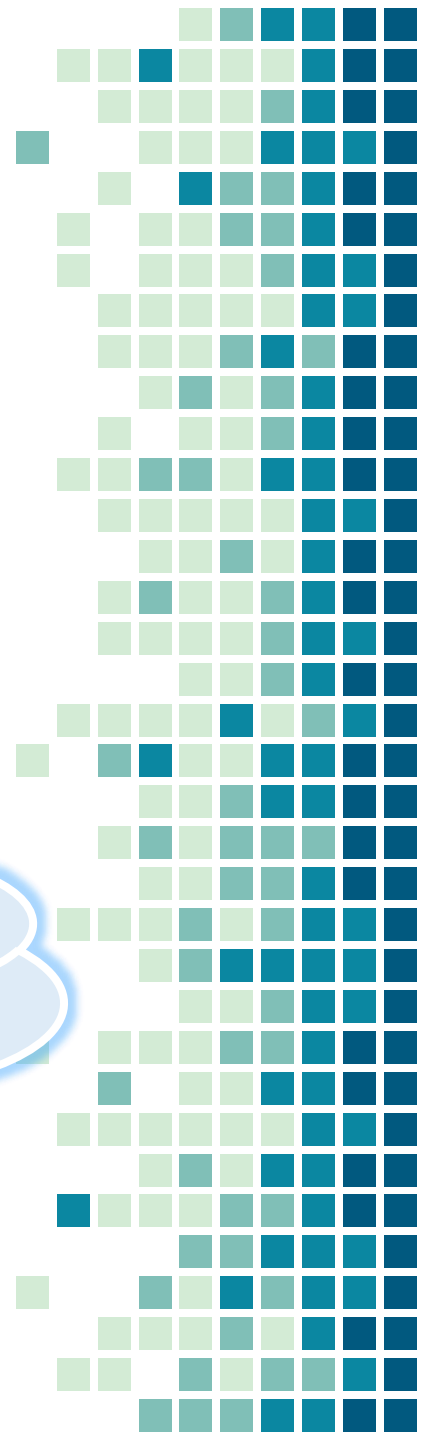
# Políticas de escritura

---

**Write Through:** Se escribe tanto en caché como en memoria principal

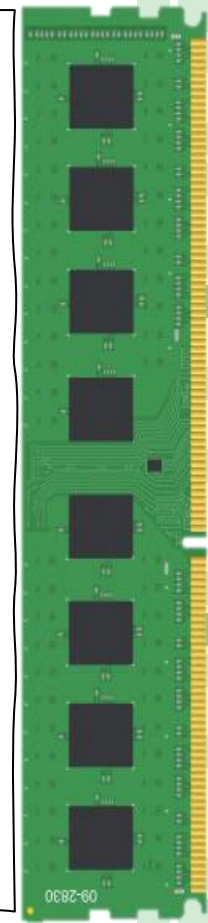
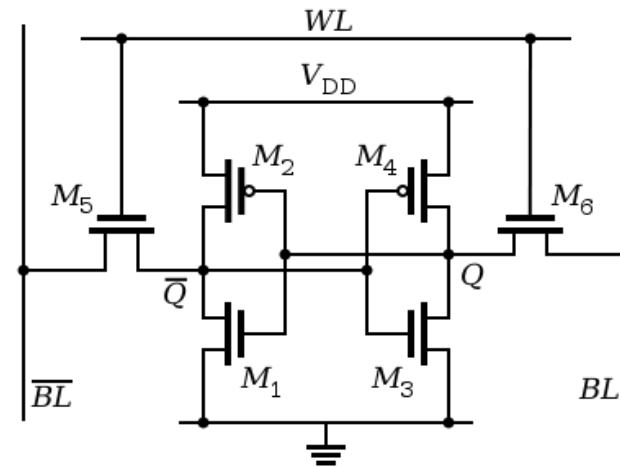
**Write back:** Se escribe solamente en caché. La escritura en memoria principal se da cuando se genera un desacierto (miss).

Produce incoherencia de caché

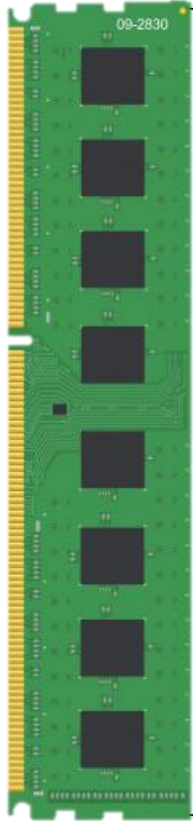


# Memoria principal - SRAM

- Requiere entre 4 y 6 transistores para almacenar un solo bit: + Área, + Costo.
- Mantiene la información mientras se mantenga la alimentación (memoria volátil), pero no necesita refrescamiento.
- Tiempos de acceso bajo



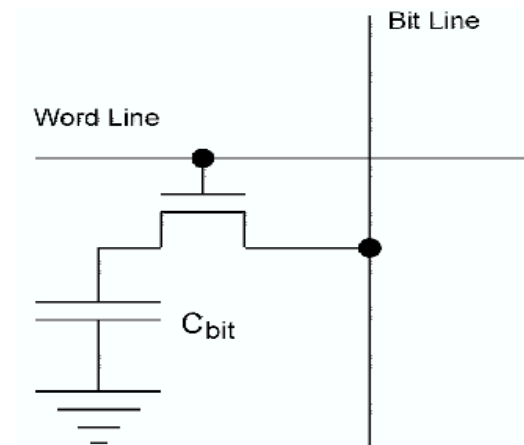
# Memoria principal - DRAM



1 transistor + 1 condensador: mayor densidad de almacenamiento, menor costo.

Tiempo limitado de la información (lectura destructiva): requiere refrescamiento

Tiempo de acceso mayor que SRAM



# Memoria Flash

- Evolución de el EEPROM
- Memoria no volátil
- Especialmente utilizada en dispositivos móviles
- 15 o 20 veces más barata que la DRAM
- Su capacidad se dobla cada 2 años.



# Almacenamiento masivo - Flash

	NAND	NOR
<b>Advantages</b>	Fast PROGRAMs	Random access
	Fast ERASEs	Byte PROGRAMs possible
<b>Disadvantages</b>	Slow random access	Slow PROGRAMs
	Byte PROGRAMs difficult	Slow ERASEs
<b>Applications</b>	File (disk) applications	Replacement of EPROM
	Voice, data, video recorder	Execute directly from nonvolatile memory
	Any large sequential data	

Comparación entre tecnologías de memoria flash



**NOR:** La escritura y lectura está basada en acceso aleatorio, es decir, byte a byte. Los tamaños típicos de los bloques son de 64, 128 o 256Kb.

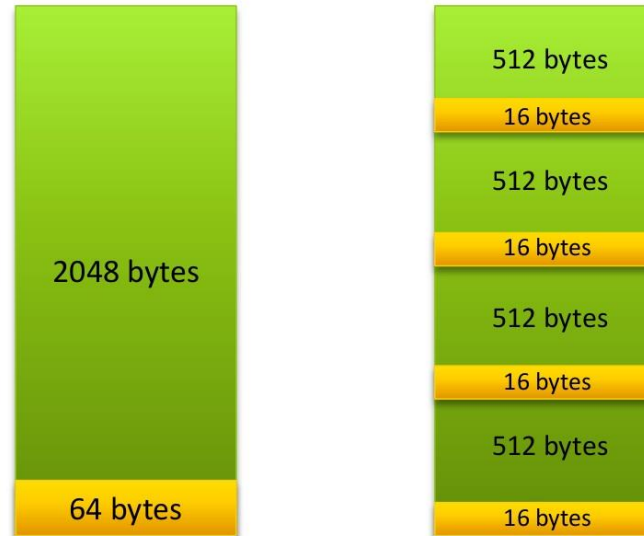


**NAND:** La escritura y lectura está basada en bloques. Los tamaños típicos de bloque son 16, 128, 256 o 512KB.

# Políticas de reemplazo

## Tamaños de bloque típicos:

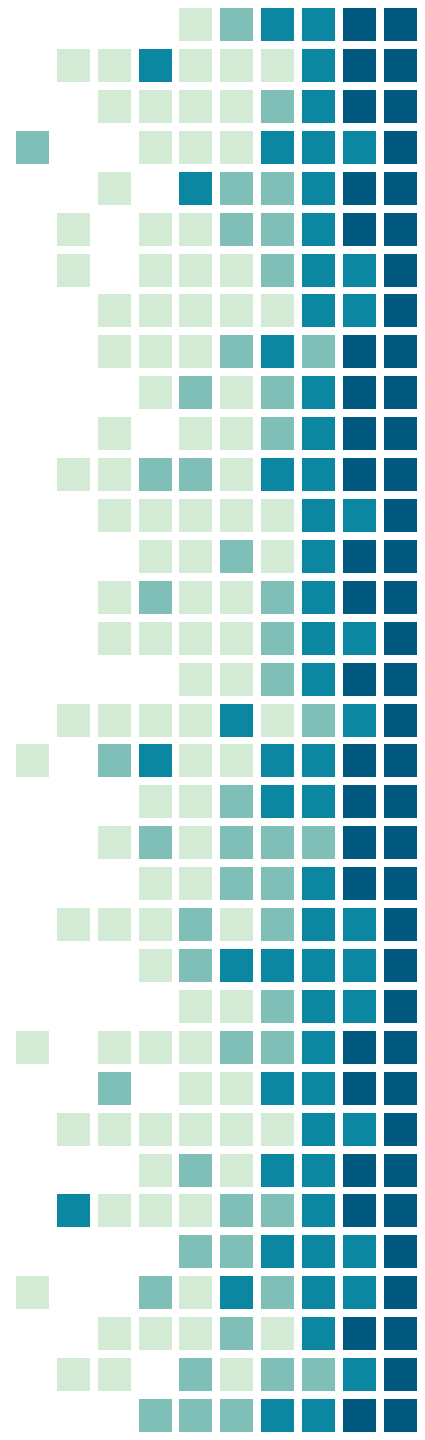
- 💡 32 páginas de 512 bytes, con un tamaño de bloque de 16 KB.
- 💡 64 páginas de 2048 bytes, con un tamaño de bloque de 128 KB.
- 💡 64 páginas de 4096 bytes, con un tamaño de bloque de 256 KB.
- 💡 128 páginas de 4096 bytes, con un tamaño de bloque de 512 KB.





# Referencias

- ...► Hennesy and David Patterson (2012)  
Computer Architecture: A Quantitative Approach. 5th Edition. Elsevier - Morgan Kaufmann.
- ...► M. Aguilar, C. Murillo (2010)  
Introducción a memoria caché.
- ...► William Stallings (2010)  
Computer organization and architecture: designing for performance. Pearson Education India



# ¿Preguntas?

Realizado por: Jason Leitón Jiménez.

---

Tecnológico de Costa Rica

Ingeniería en Computadores

2024

