

# Memorias

Tipos, Jerarquías, Caché

# Memorias:

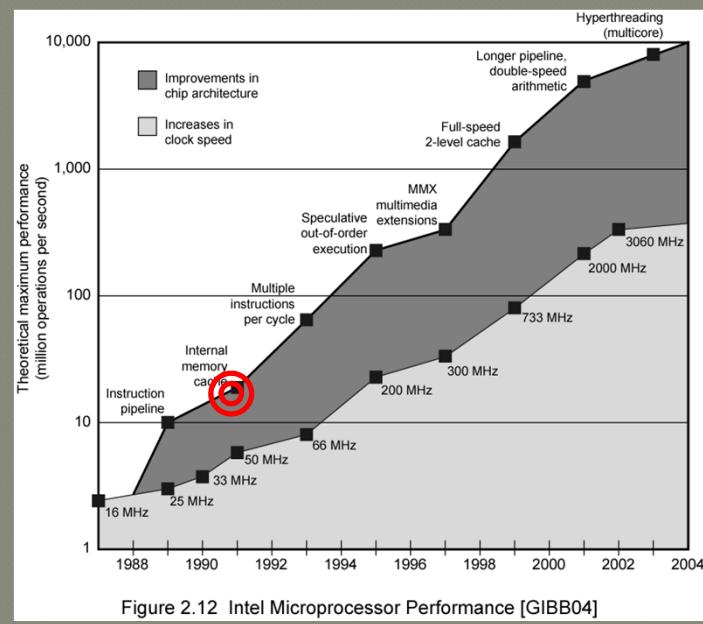
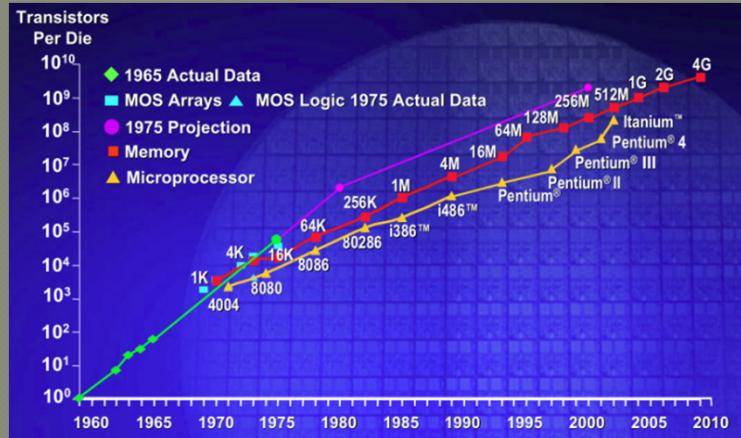
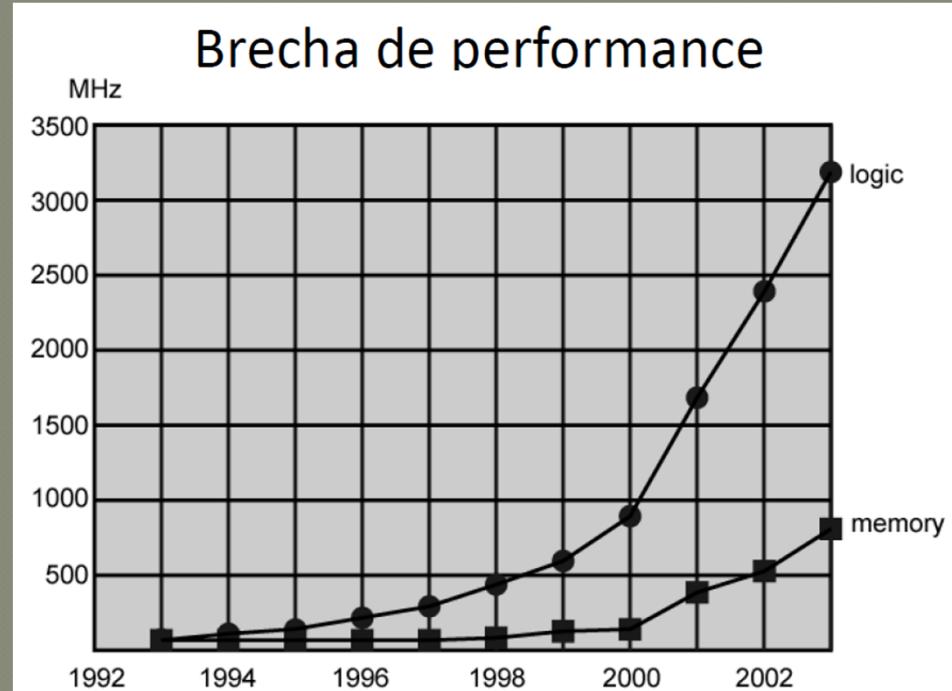
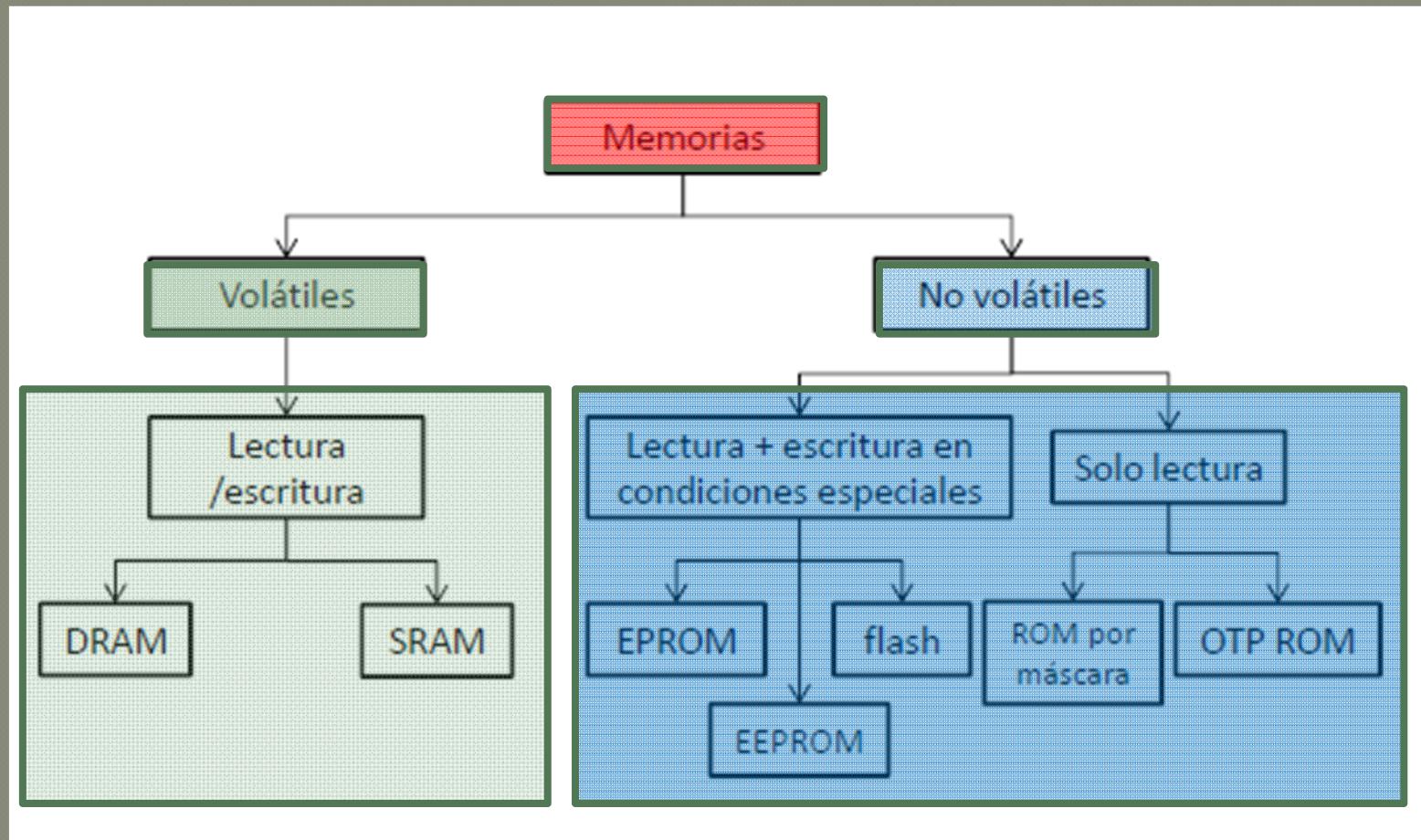


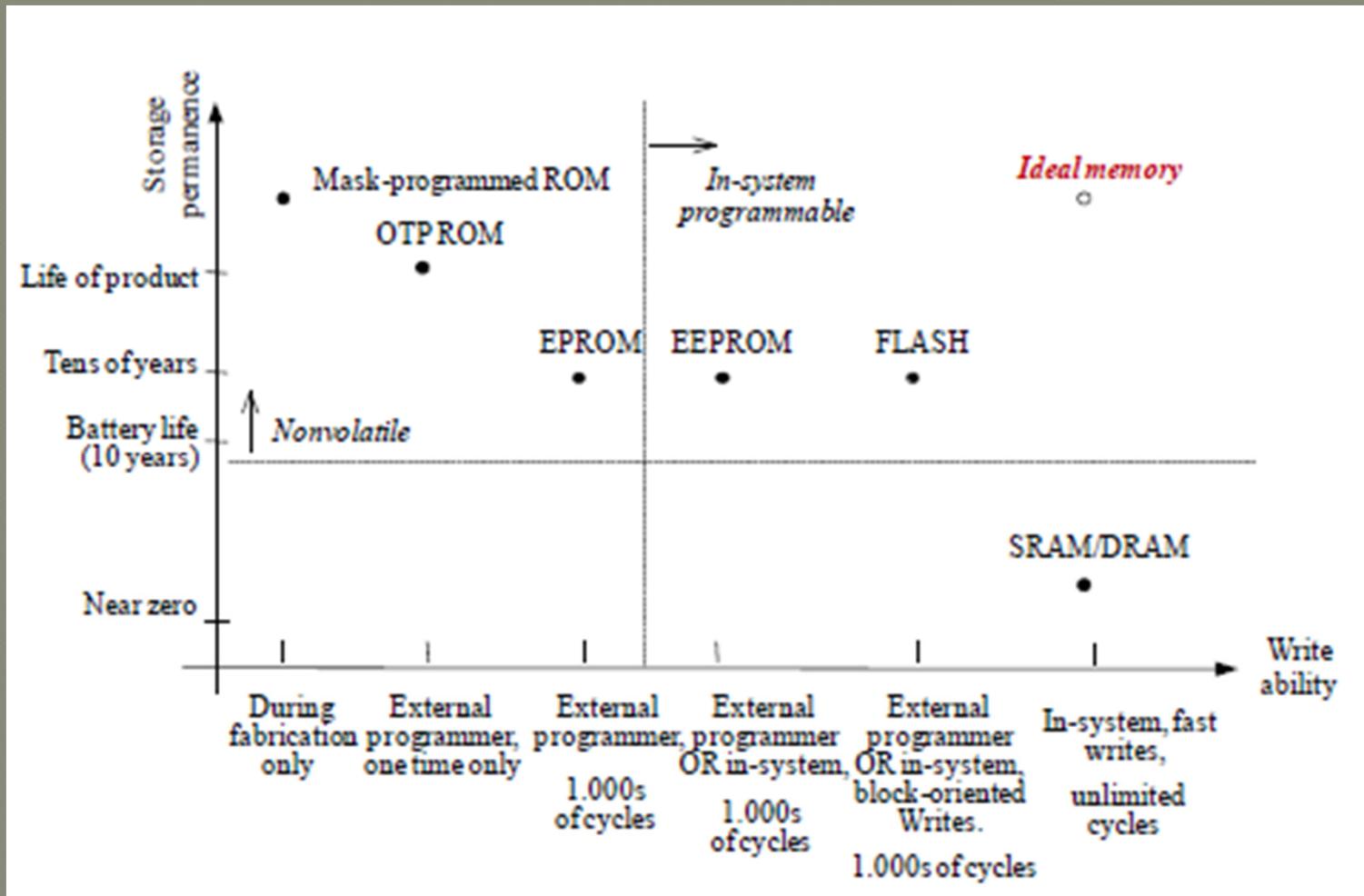
Figure 2.12 Intel Microprocessor Performance [GIBB04]



# Memorias: Tipos



# Memorias: Tipos



# Memorias: Tipos

## Duración de la información:

- Permanente / no volátil
- Volátil
- Lectura destructiva
- Refresco

## Modo de acceso:

- Aleatorio
- Secuencial
- Directo

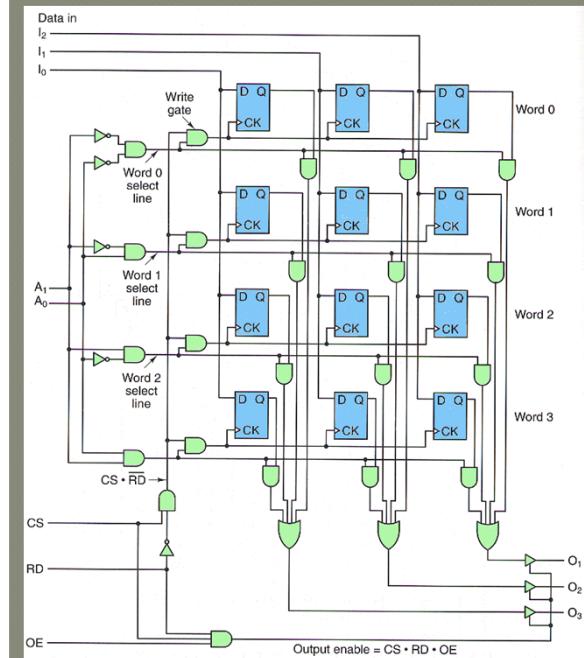
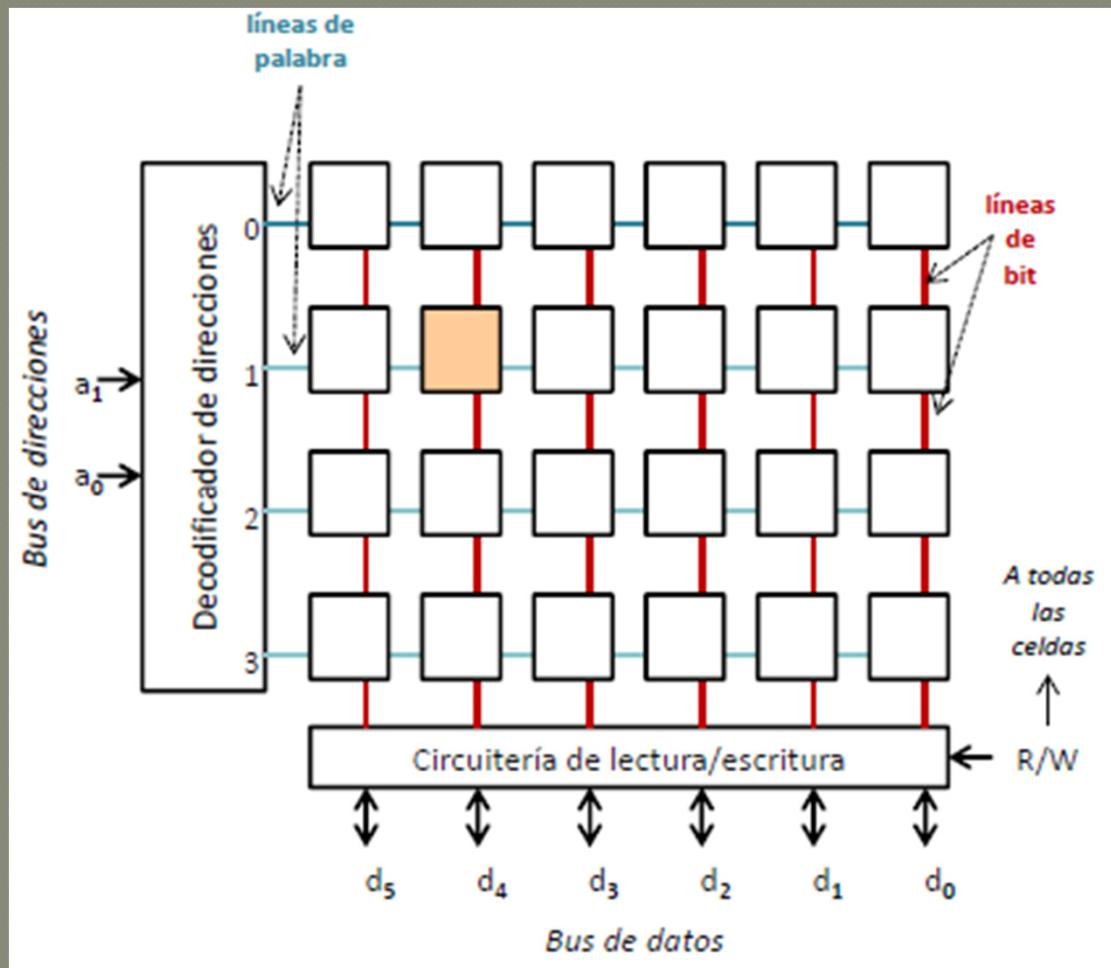
## Realización de operaciones:

- Por palabras
- Por bloques

## Forma de acceso:

- Por dirección
- Por contenido

# Memorias: Estructura básica

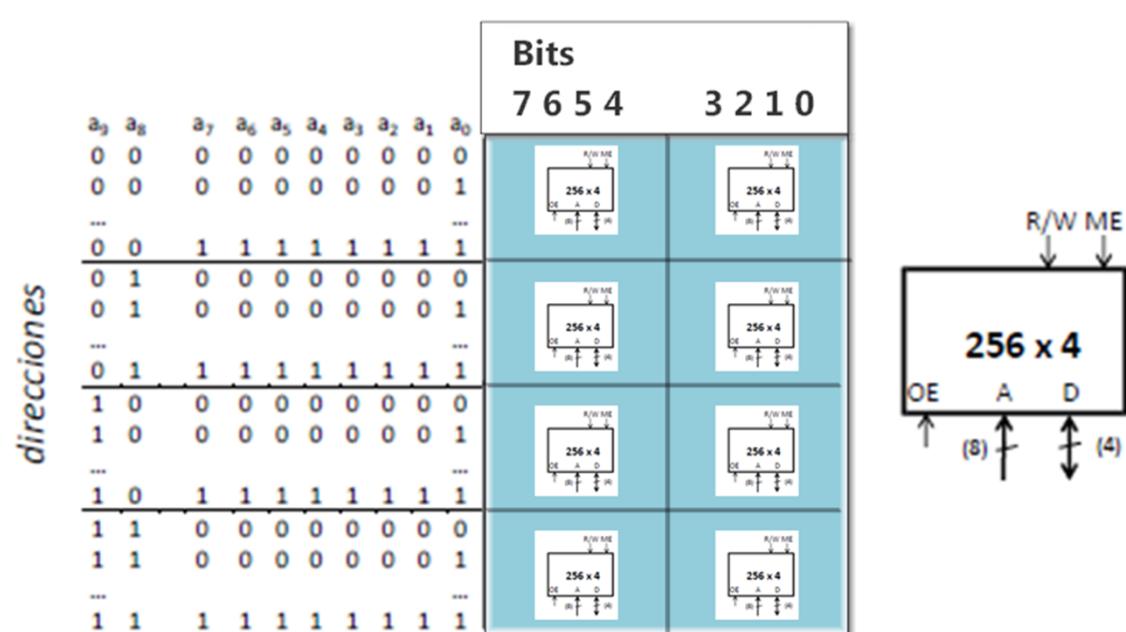


# Chips y Bancos de Memorias

## EJEMPLO:

Construir un banco de memorias de 1.024 palabras de 8 bits (1.024x8) utilizando chips de memoria SRAM de 256 palabras de 4 bits (256x4)

### MEMORY MAPPING



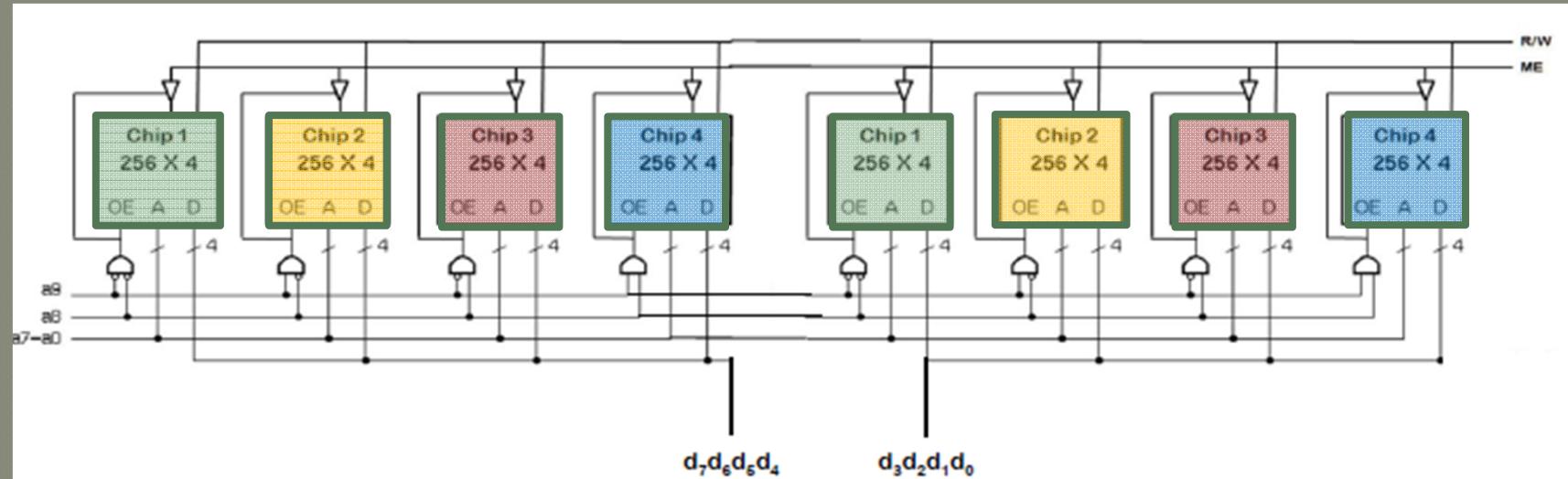
# Chips y Bancos de Memorias

Banco de memoria de 1k (1024) direcciones x 8 bits

Chips de 256 líneas x 4 bits

2 columnas de 4 bits → 8 bits

4 filas de 256 direcciones →  $1024B = 1 kB$



# Chips y Bancos de Memorias

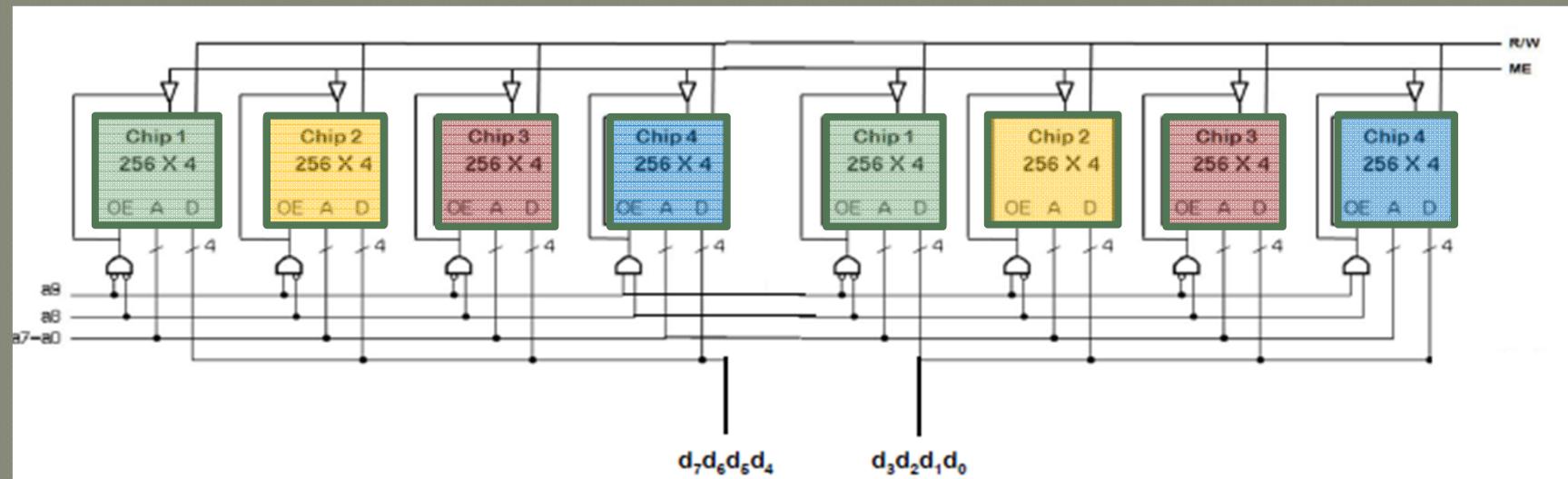
Banco de memoria de 1k (1024) direcciones x 8 bits

Chips de 256 líneas x 4 bits

Bus de Datos → 8 bits

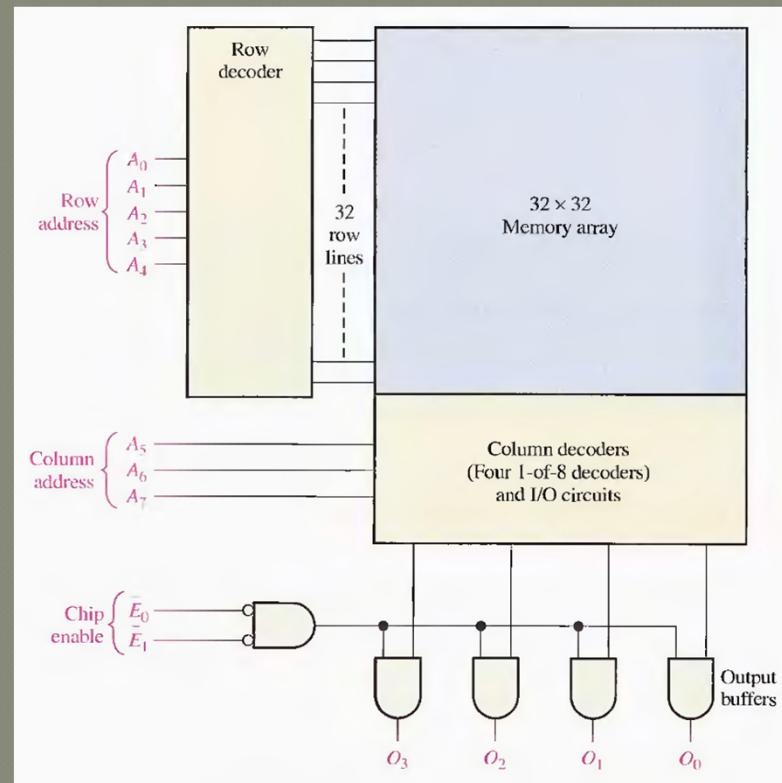
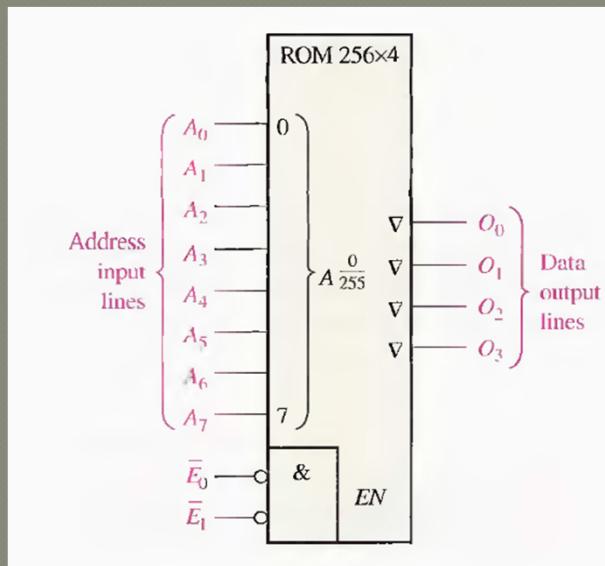
Bus de Direcciones Chip →  $2^8 = 256$  direcciones

Bus de Direcciones Total →  $2^{10} = 1024$  direcciones



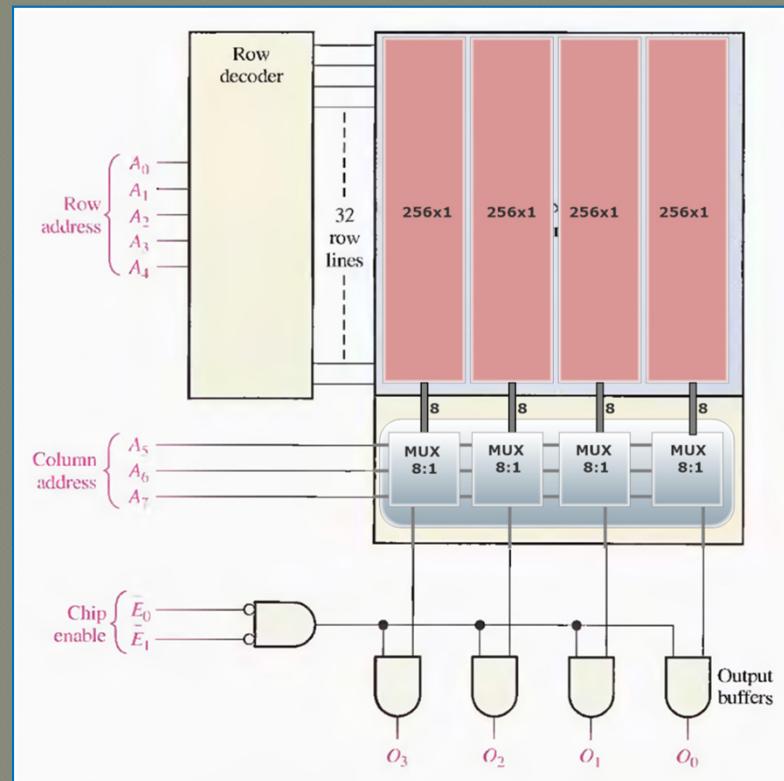
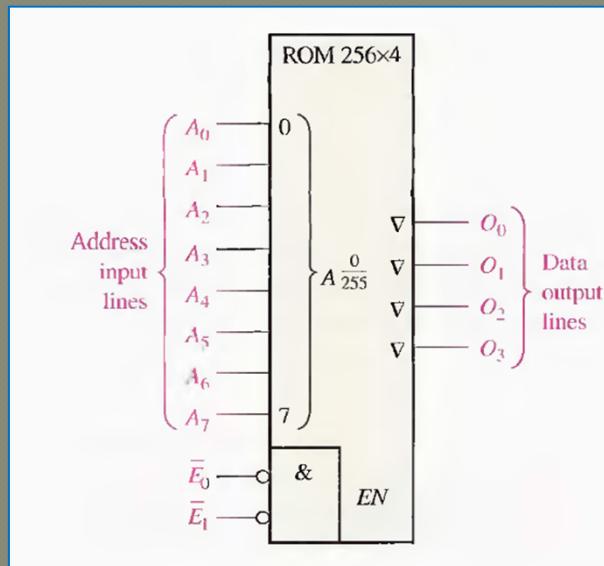
# • Memorias

- Banco de memoria de 1k (1024) direcciones x 8 bits
- Chips de 256 líneas x 4 bits



# • Memorias

- Banco de memoria de 1k (1024) direcciones x 8 bits
- Chips de 256 líneas x 4 bits

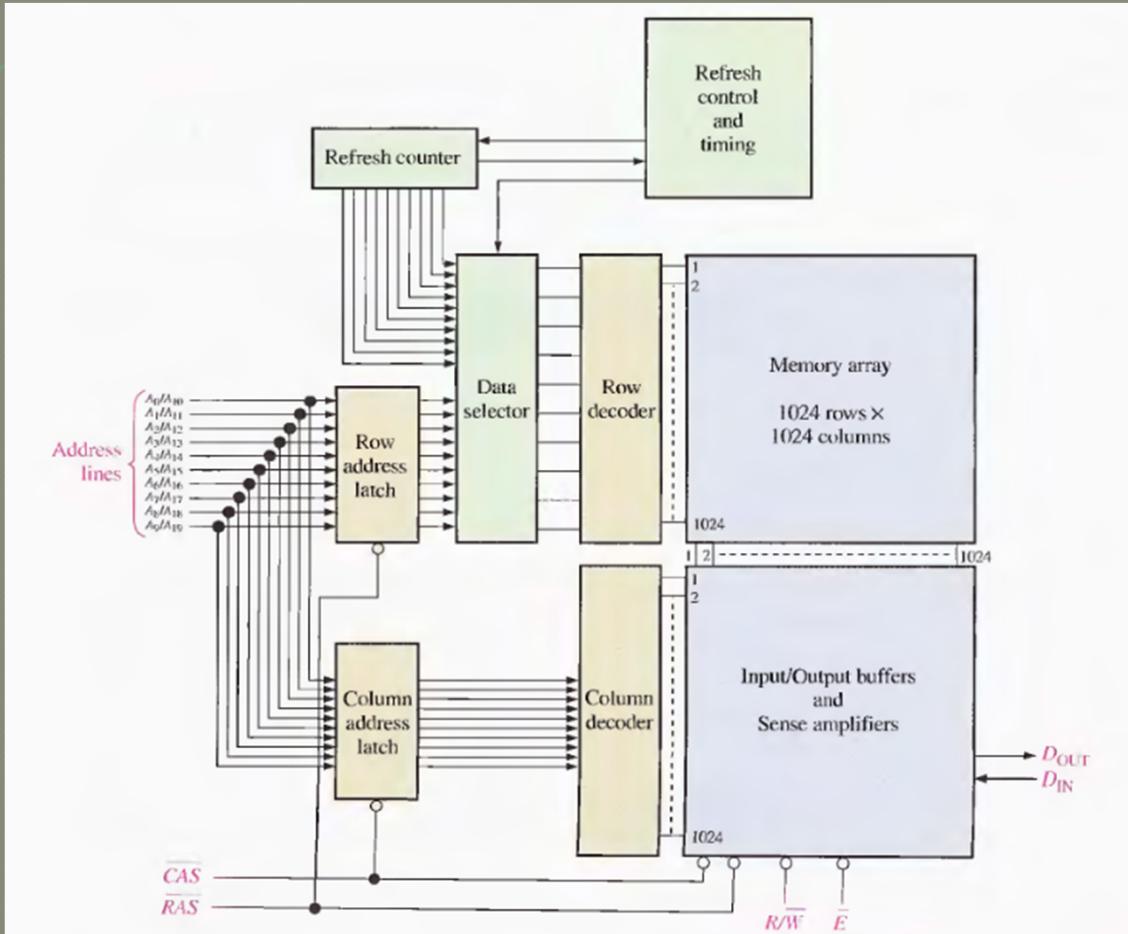


## • Memorias

1Mx1bit

10 líneas de dirección  
multiplexadas en el  
tiempo

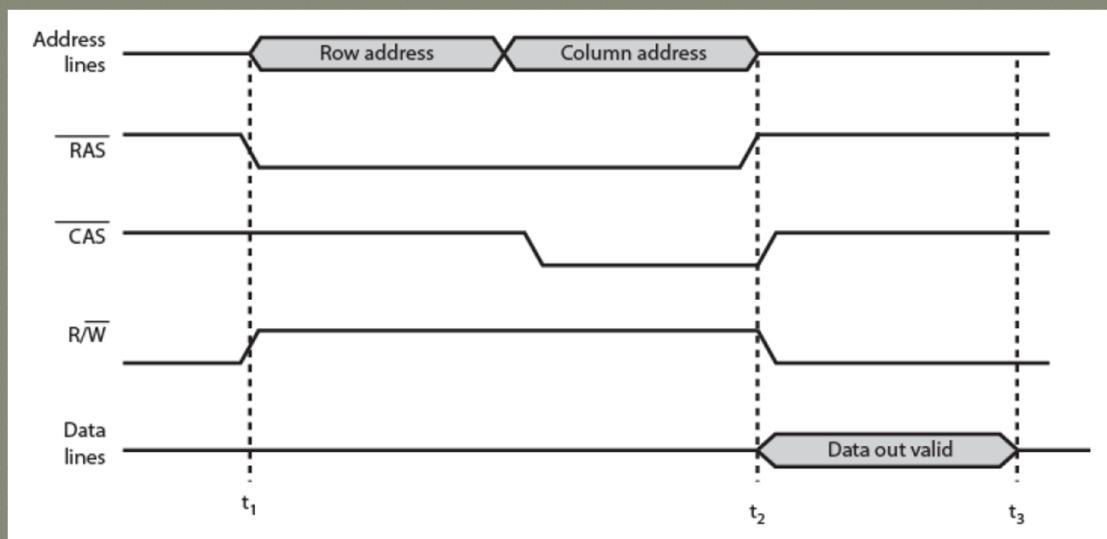
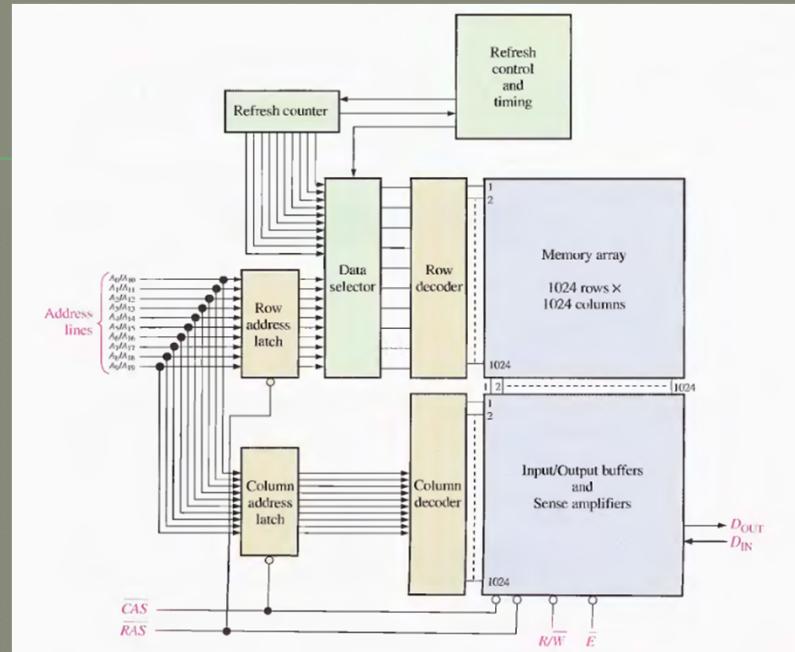
Organizada de esta manera,  
la matriz anterior de 32x32 y  
5 bits de dirección  
¿que memoria tendría?



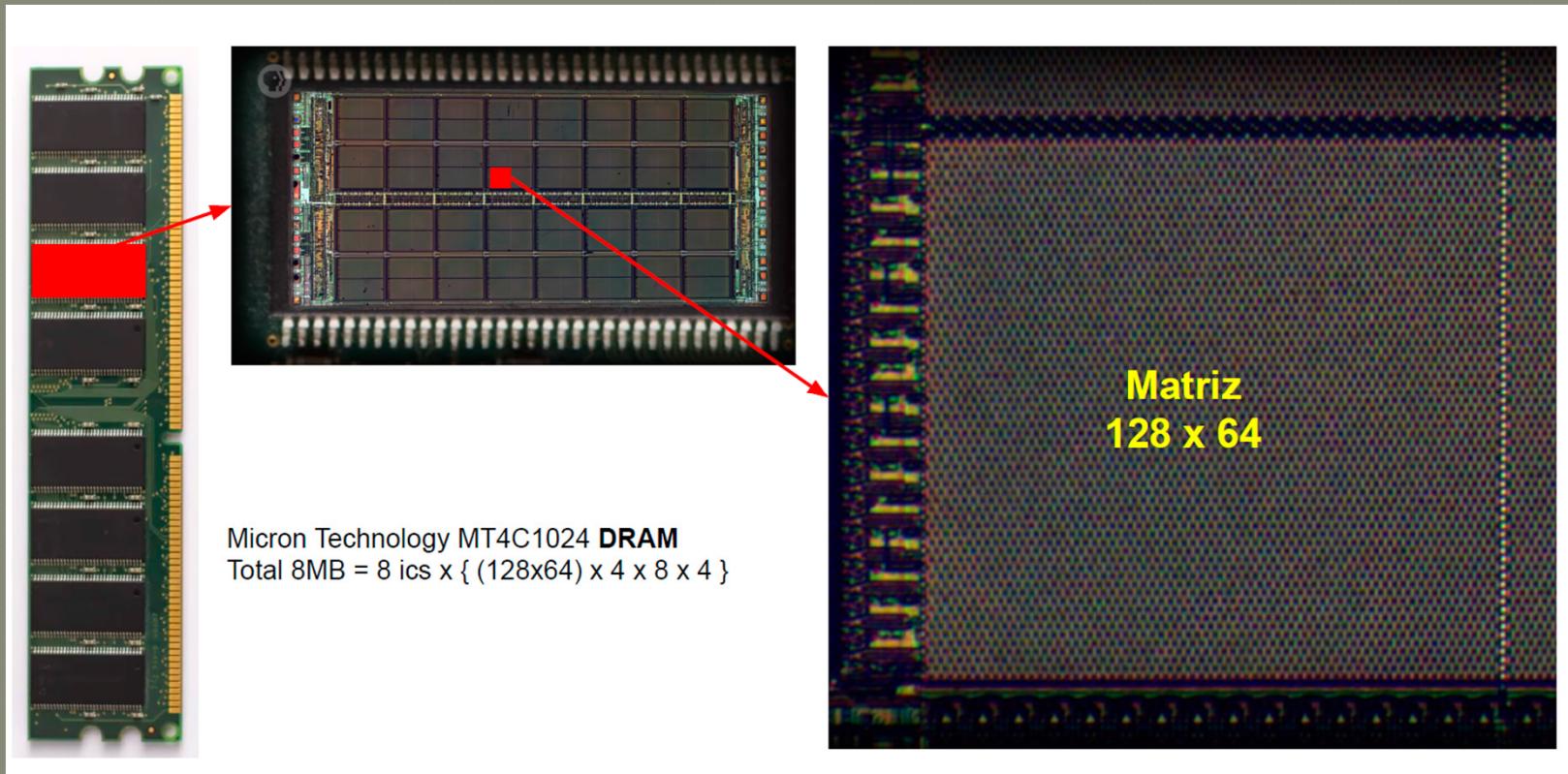
# • Memorias

1Mx1bit

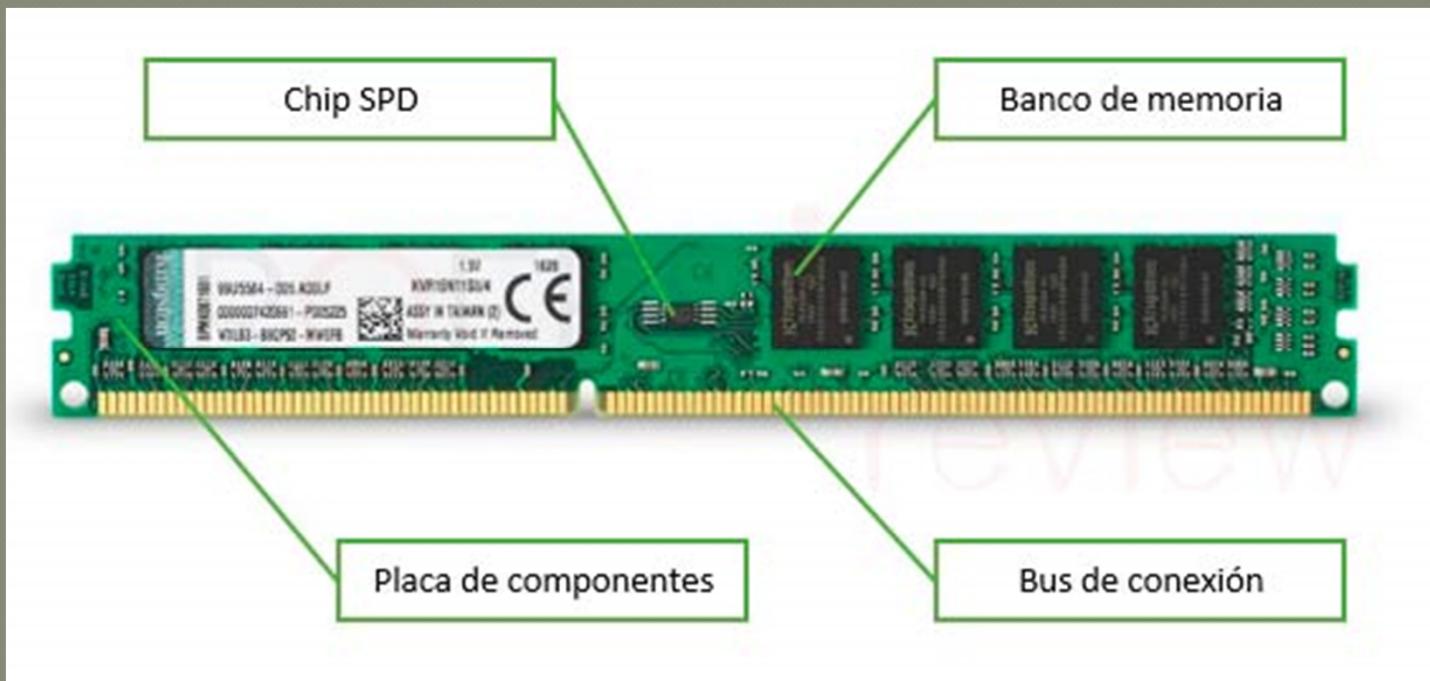
10 líneas de dirección  
multiplexadas en el  
tiempo



# Chips y Bancos de Memorias



# Chips y Bancos de Memorias



# Chips y Bancos de Memorias

## Memoria RAM

### Primeros Modelos de Ram

DIP Memory

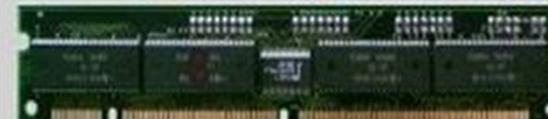
30 pin SIPP

30 pin SIMM

72 pin  
SIMM

168 pin DIMM

168 pin PC100



DIMM



DDR



DDR2



DDR3

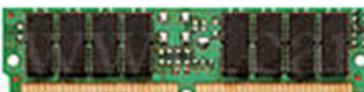


# Chips y Bancos de Memorias

Note, as well as the different number of pins, the different spacing of the slots in the connector-edge



30 pin SIMM



72 pin SIMM



MicroDIMM  
(rare)



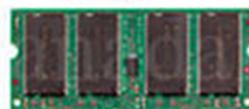
184 pin Rambus RDRAM RIMM



100 pin DIMM  
printer RAM



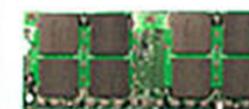
72 pin SODIMM  
(rare)



144 pin SDRAM  
SODIMM



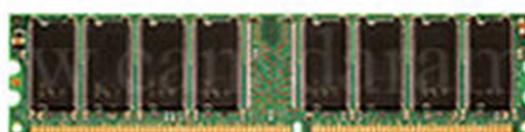
200 pin DDR  
SODIMM



200 pin DDR-2  
SODIMM



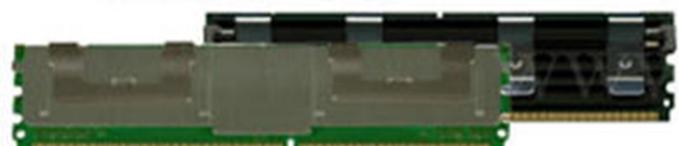
168 pin SDRAM DIMM



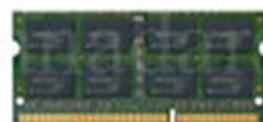
184 pin DDR DIMM



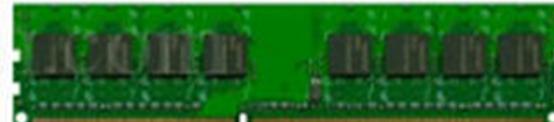
240 pin DDR-2 DIMM



240 pin DDR-2 FB-DIMM, standard & Apple heatsink



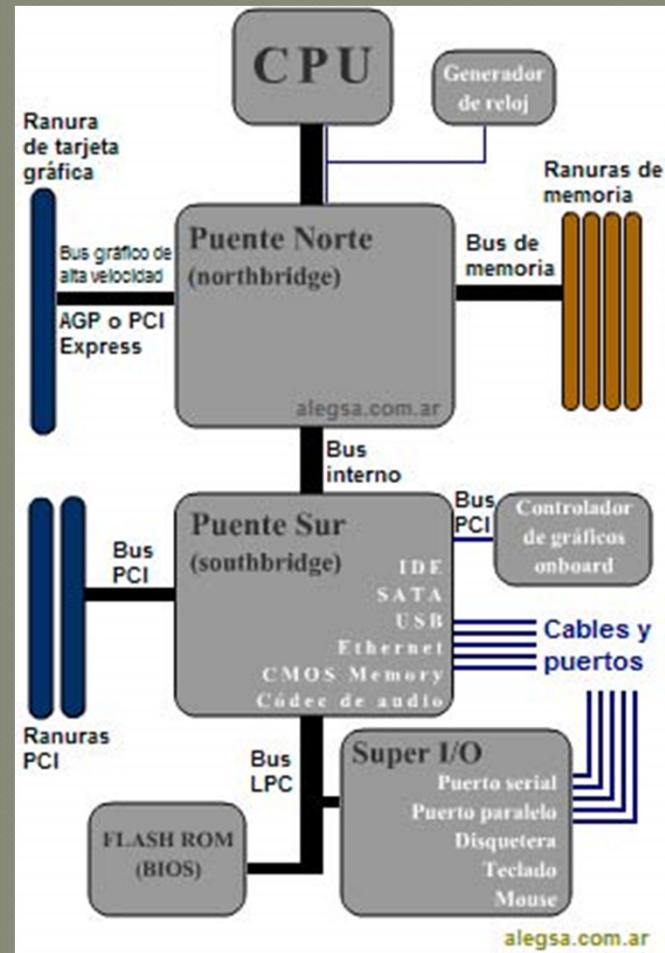
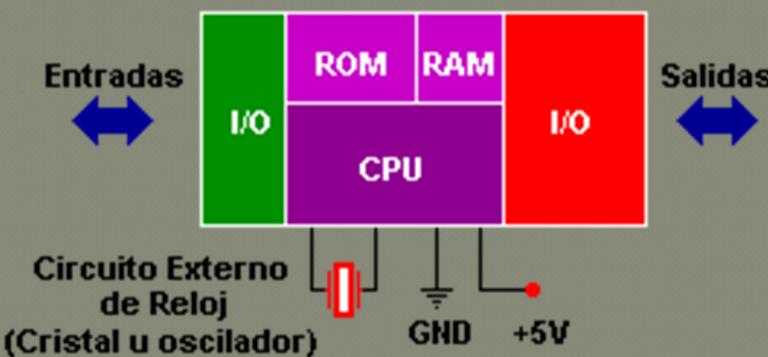
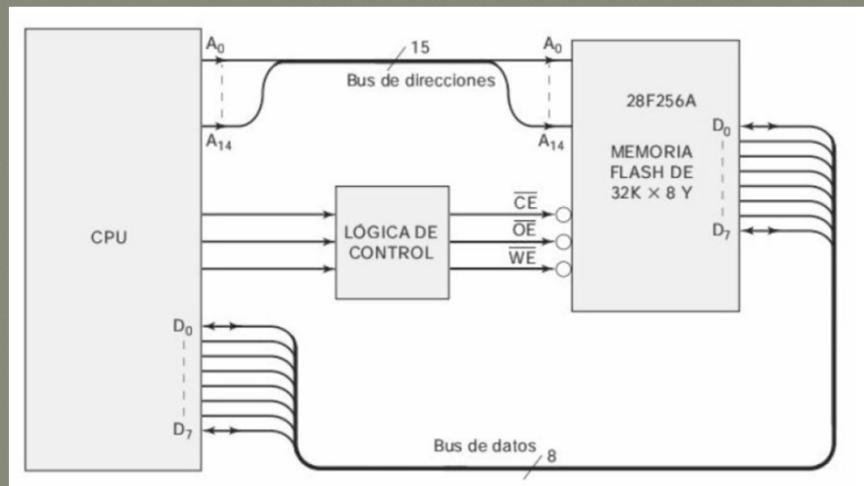
203 pin DDR-3 SODIMM



240 pin DDR-3 DIMM

(c) 2007-2009 www.canadaram.com

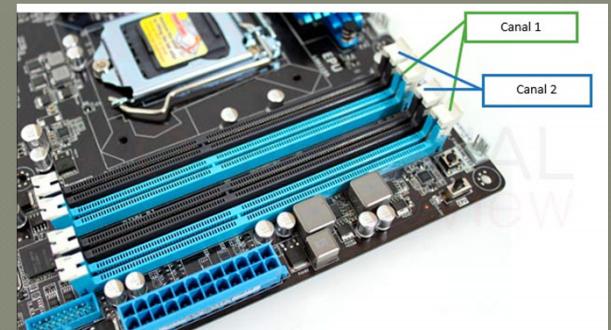
# Chips y Bancos de Memorias



# Chips y Bancos de Memorias

## ● Memoria Entrelazada

- Colección de chips DRAM
- Agrupados en bancos de memoria
- Los bancos resuelven independientemente requerimientos de lectura o escritura.
- N bancos resuelven N requerimientos de manera simultánea.
- K bancos de acceso en paralelo → Factor de entrelazado K (potencias de 2)



# Jerarquía de Memoria

---

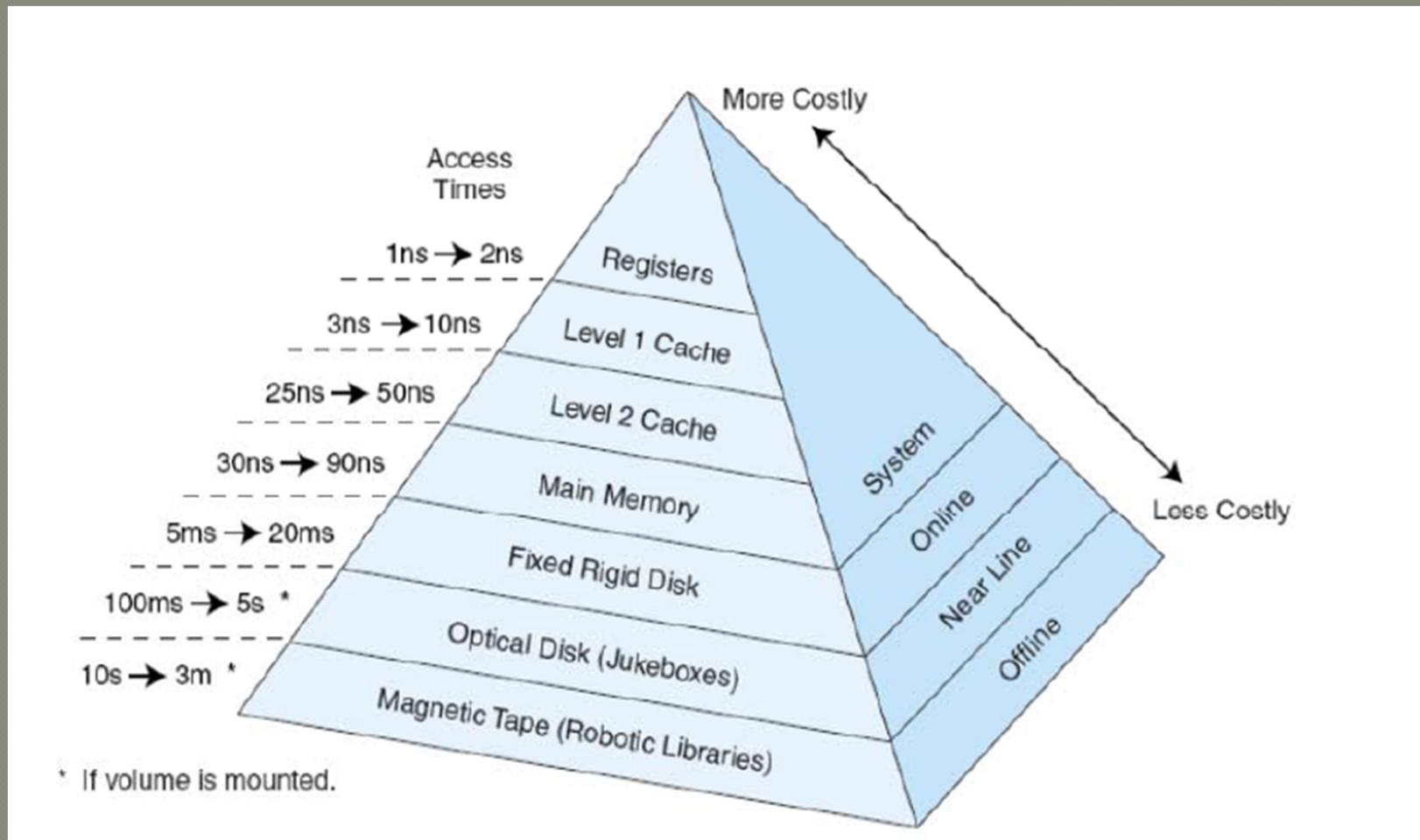


# Jerarquía de Memoria

---

- Métricas de las Memorias
  - Capacidad de Almacenamiento (B, MB, GB, TB)
  - Tiempo de Acceso (ms, ns)
  - Velocidad de Transferencia de Datos (B/seg)
  - Consumo de energía (W)
  - Tamaño Físico (cm<sup>3</sup>)
  - Costo total y costo por MB (\$, \$/MB)

# Jerarquía de Memoria



# Jerarquía de Memoria

---

**Objetivo:** la velocidad del sistema deberá ser, aproximadamente, la del nivel más rápido al costo del nivel mas barato.

- A medida que nos alejamos de la CPU, cada nivel inferior es más grande, más lento y más barato que el nivel previo (o superior) en la jerarquía.
- Debe haber correspondencia de direcciones en los distintos niveles.

Propiedades a cumplir

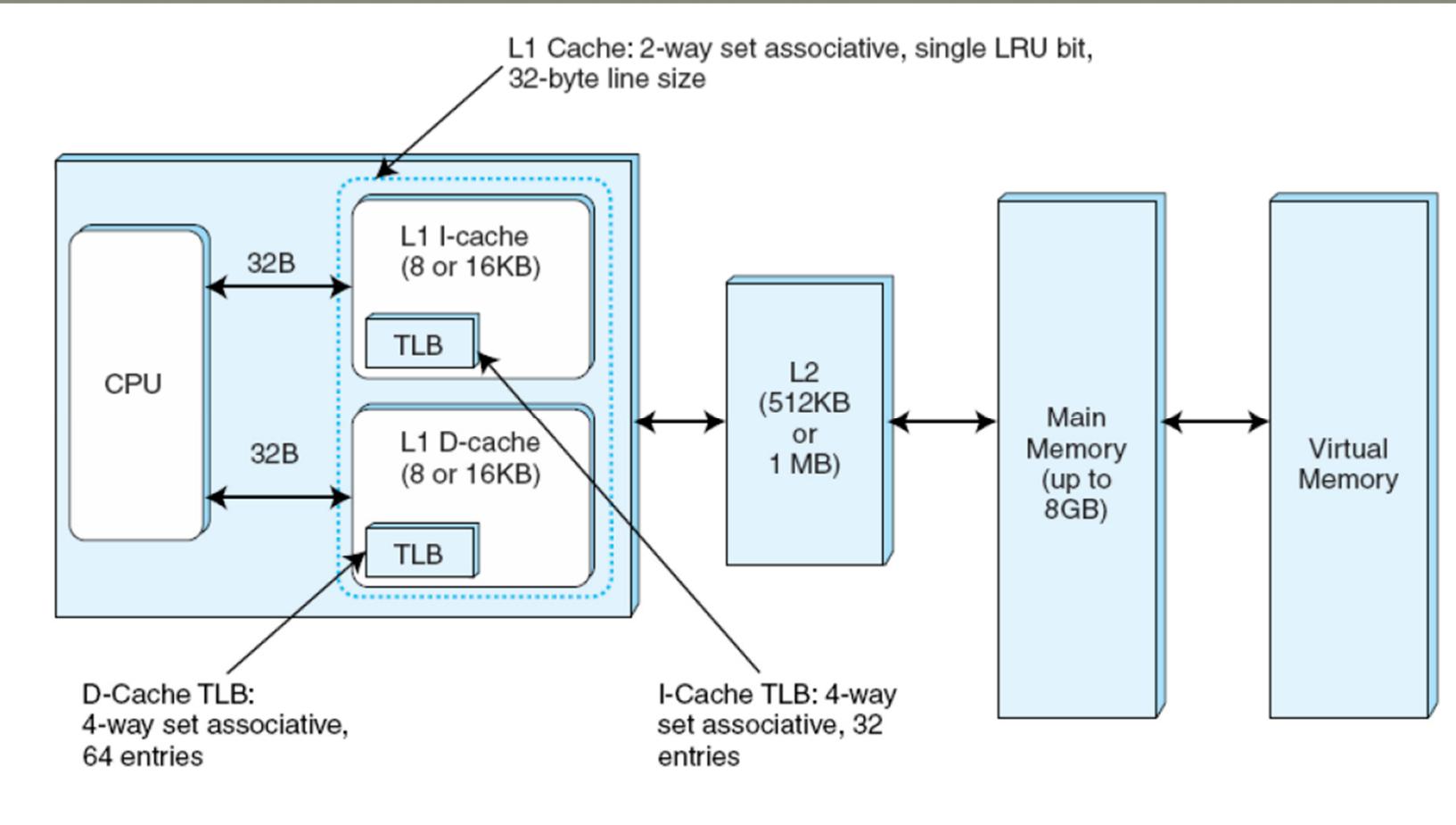
- **Inclusión**

Los datos almacenados en un nivel han de estar almacenados en los niveles inferiores a él

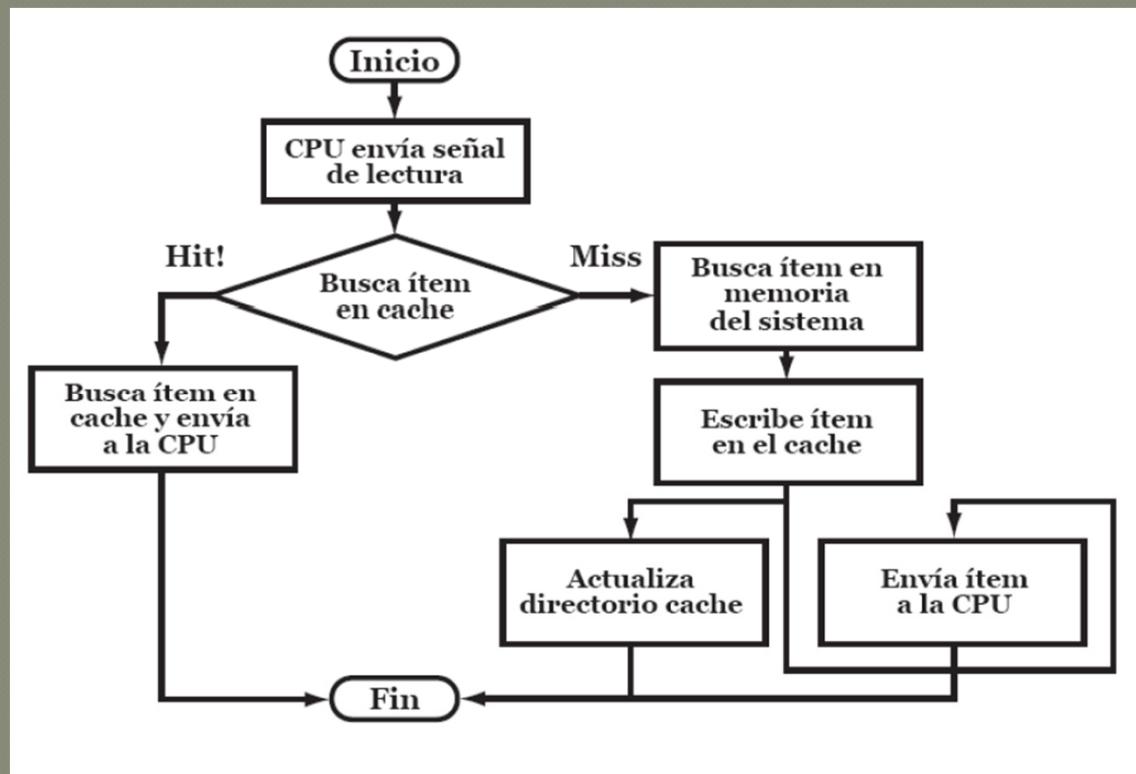
- **Coherencia**

Las copias de la misma información en los distintos niveles deben contener los mismos valores.

# Jerarquía de Memoria



# Jerarquía de Memoria



CPU solicita contenido de una dirección de memoria

La caché tiene ese dato?  
Si: lo obtiene (rápido)  
No: Se lee desde memoria  
y se escribe la caché

# Jerarquía de Memoria

- $T_a$  : Tiempo de Acceso Medio
- $T_c$  : Tiempo de Acceso a Caché
- $T_{mp}$  : Tiempo de Acceso a Memoria Ppal.
- $h$  : Probabilidad de Acierto (nro. Aciertos / nro. Intentos)
- $hf$  : probabilidad de Fallo =  $(1-h)$
- $h + hf = h + (1-h) = 1$  probabilidad total

Para una jerarquía de 2 niveles (memoria caché y memoria principal)

$$T_a = \frac{\text{nro. aciertos} \times \text{tiempo de acceso } J_1 + \text{nro. fallas} \times \text{tiempo penalización}}{\text{nro. intentos}}$$

$$T_a = T_c \cdot h + (T_c + T_{mp}) \cdot (1-h)$$

$$T_a = T_c + (1-h) T_{mp}$$

En general:

$$T_{a_n} = T_1 + (1-h_1)T_2 + (1-h_1)(1-h_2)T_3 + \dots + (1-h_1)\dots(1-h_{n-1})T_n$$

# Jerarquía de Memoria

---

- Ejemplo:
- $T_{mp} = 500 \text{ ns}$  ;  $T_c = 50 \text{ ns}$  ;  $h = 0,85$

$$T_a = T_c + (1-h) T_{mp} = 50 \text{ ns} + (1-0,85).500 \text{ ns} \\ = 125 \text{ ns}$$

$$\text{Speedup} = \frac{T \text{ sin mejora}}{T \text{ con mejora}} = 500 / 125 = 4$$

# Jerarquía de Memoria

---

## ● Localidad Espacial y Temporal

- **Localidad Espacial:** los elementos de memoria cuyas direcciones están próximas a los últimos referenciados serán referenciados.
- **Localidad Temporal:** los elementos de memoria referenciados recientemente (datos o instrucciones), volverán a serlo en un futuro próximo

**Instrucciones:** programa secuencial – Saltos

**Datos:** Bucles - Arreglos

# Memoria Caché

---

- **Política de Ubicación (correspondencia)**
  - ¿dónde se ubican los bloques de la MP en la caché?
  
- **Política de Sustitución**
  - ¿qué bloque reemplazar si la caché está llena?
  
- **Política de Escritura (actualización)**
  - ¿Qué hacer ante una operación de escritura?

# Memoria Caché

---

## ● Política de Ubicación (correspondencia)

- ¿dónde se ubican los bloques de la MP en la caché?
- Correspondencia Directa, Asociativa, Asociativa por conjuntos

## ● Política de Sustitución

- ¿qué bloque reemplazar si la caché está llena?
- LRU (Least Recently Used, menos recientemente referenciado), LFU (Least Frequently Used, bloque menos referenciado), FIFO, Random

## ● Política de Escritura (actualización)

- ¿Qué hacer ante una operación de escritura?
- Problema de coherencia en la actualización, escritura inmediata, diferida, con asignación, sin asignación

# Memoria Caché

---

## Política de Escritura

### Escritura en Acierto

#### Write-through (Escritura inmediata).

- Se actualizan simultáneamente la posición de la caché y de la memoria principal.
- con múltiples CPU, observar el tráfico a memoria principal para mantener actualizada cada cache local.
- se genera mucho tráfico y retrasa la escritura.

#### Write-back (Post-escritura).

- La información sólo se actualiza en la caché.
- Se marca como actualizada • bit de “sucio”.
- La memoria principal se actualiza en el reemplazo y puede contener información errónea en algún momento

# Memoria Caché

---

## Política de Escritura

### Escritura en Fallo

#### Write allocate (con asignación)

- La información se lleva de la memoria principal a la caché. Se sobreescribe en la caché
- Habitual con write-back

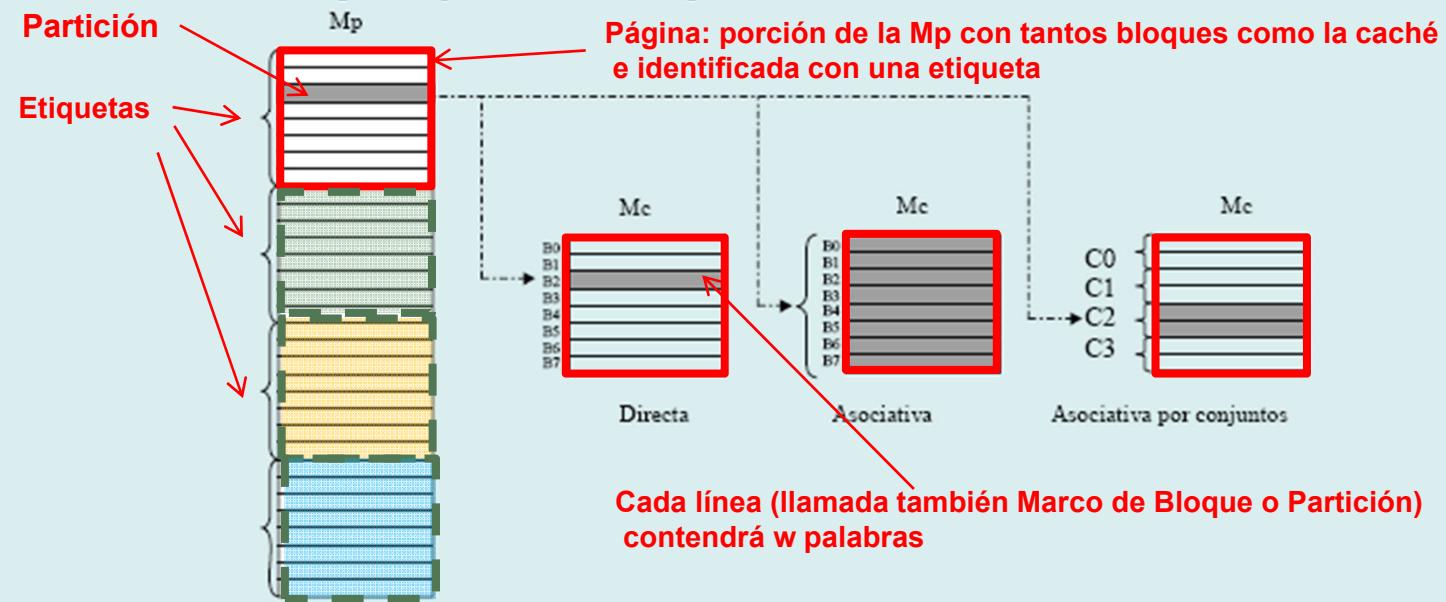
#### No-write allocate (sin asignación)

- El bloque no se lleva a la memoria caché. Se escribe directamente en la memoria principal.
- Habitual con write-through

# Memoria Caché : correspondencia

## Función de correspondencia

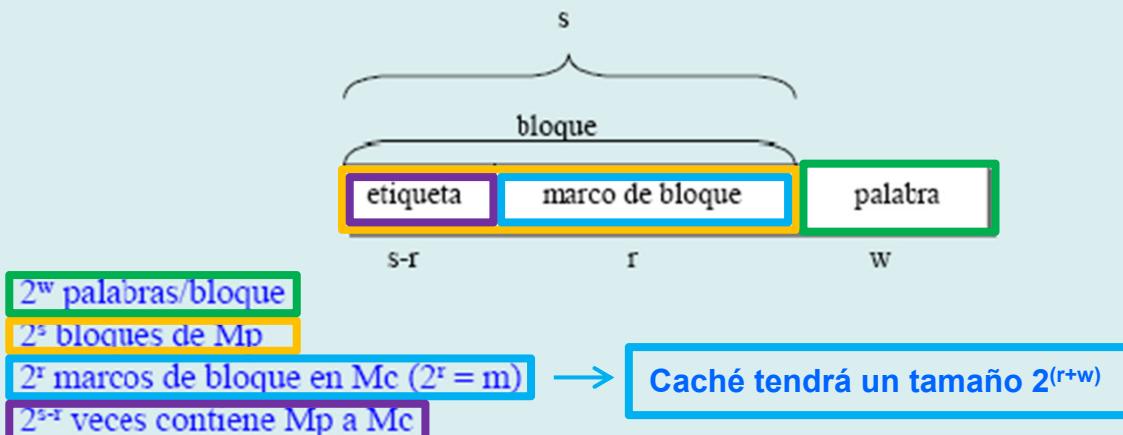
- Determina las posibles líneas de la caché (marcos de bloque) en las que se puede ubicar un determinado bloque de la memoria principal.
- Existen tres funciones de correspondencia: *directa, asociativa y asociativa por conjuntos*.
  - Directa: un bloque de Mp sólo puede ubicarse en una línea de la caché, aquella que coincide con el bloque cuando superponemos Mc sobre Mp respetando fronteras de Mc
  - Asociativa: un bloque de Mp puede ubicarse en cualquier línea de Mc.
  - Asociativa por conjuntos: es un compromiso entre las dos anteriores.



# Memoria Caché : correspondencia

## Correspondencia directa

- El bloque  $B_j$  de  $M_p$  se puede ubicar sólo en el marco de bloque  $MB_i$  tal que  $i = j \bmod m$   $m =$  número total de líneas que tiene la caché.



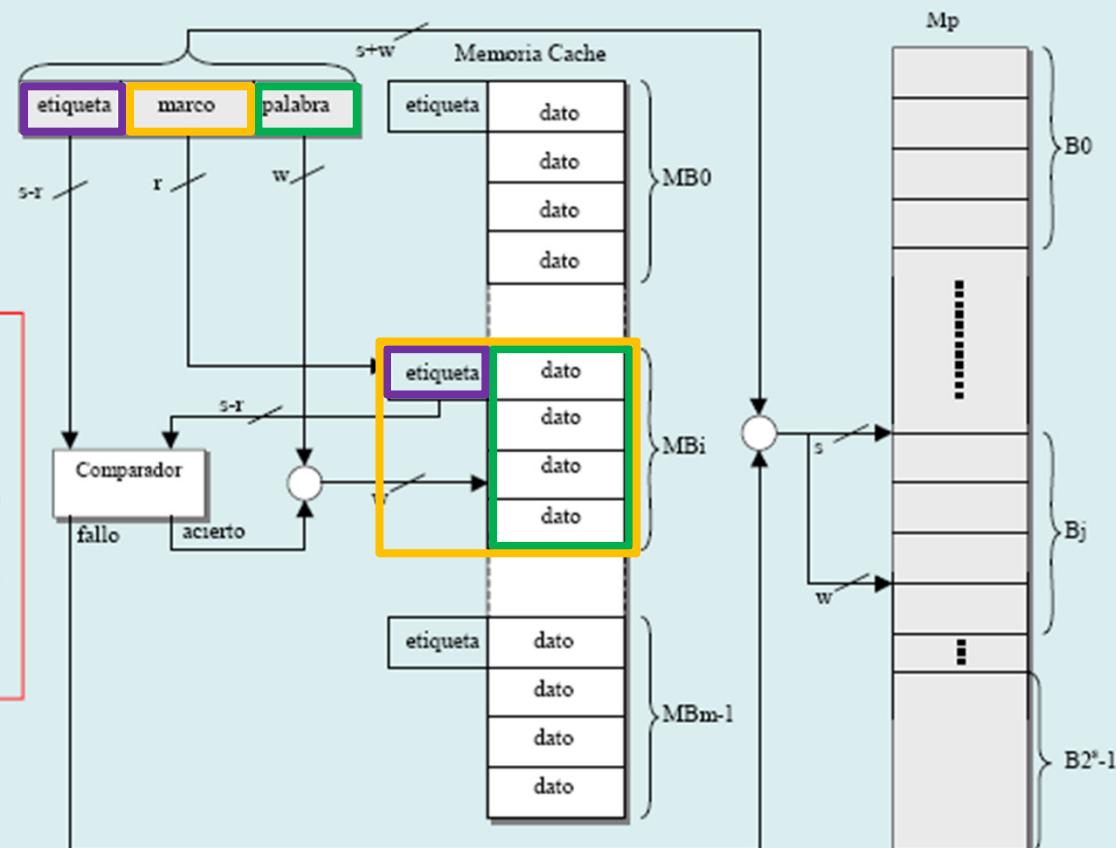
- Los  $s - r$  bits de la *etiqueta* diferenciarán a cada uno de los bloques de  $M_p$  que pueden ubicarse en el mismo marco de bloque de  $M_c$ .
- El directorio caché en correspondencia directa contendrá un registro de  $s - r$  bits por cada marco de bloque para contener la etiqueta del bloque ubicado en ese momento en dicho marco.

# Memoria Caché : correspondencia

## Correspondencia directa

- El esquema de acceso a una Mc con correspondencia directa es el siguiente:

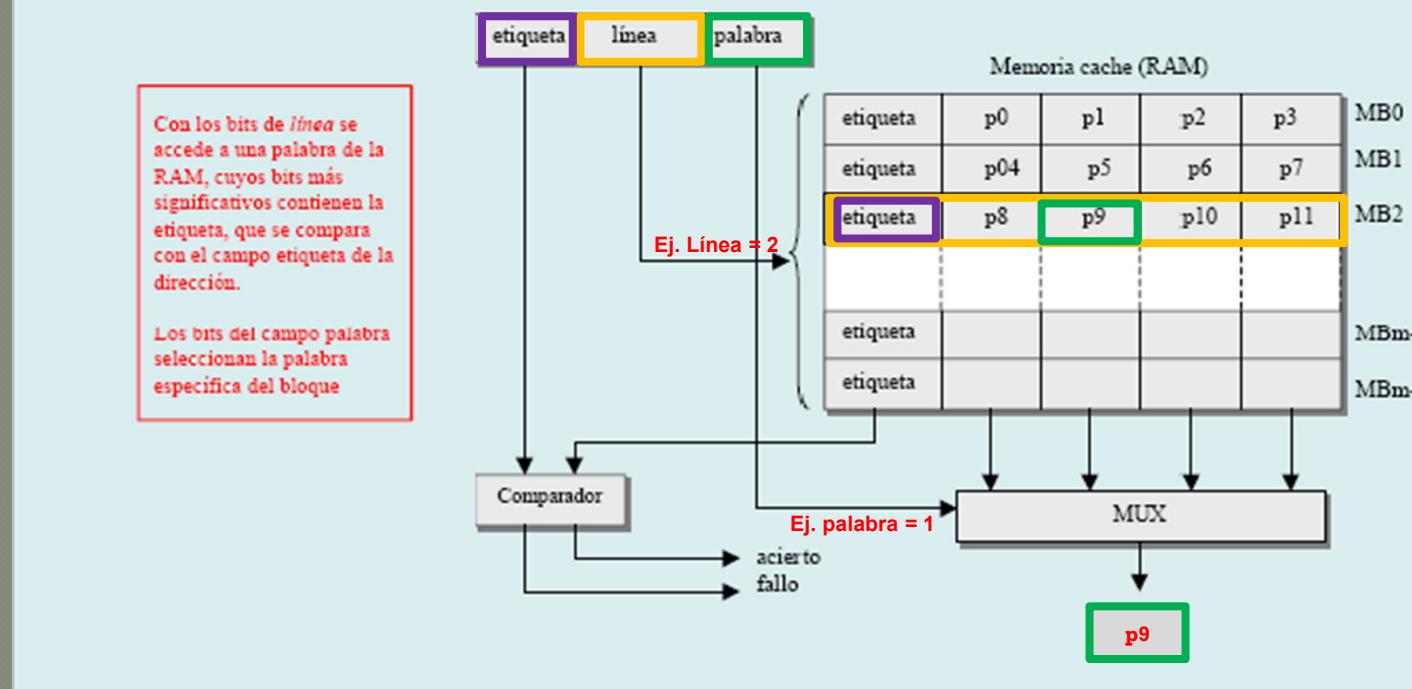
Los bits del campo *marco* (o *línea*) de la dirección determinan la línea leída. Si la etiqueta de la dirección coincide con la etiqueta contenida en la línea, hay acierto, es decir, el bloque contenido en la línea leída es el mismo al que pertenece la dirección que accede al sistema de memoria. En caso contrario se produce un fallo de caché.



# Memoria Caché : correspondencia

## Correspondencia directa

- Desde el punto de vista del diseño una Mc con correspondencia directa se organiza como una memoria RAM con longitud de palabra suficiente para contener una línea y los bits de etiqueta.
- El número de palabras lo determinará el número de líneas.
- El esquema de acceso es el siguiente:



# Memoria Caché : correspondencia

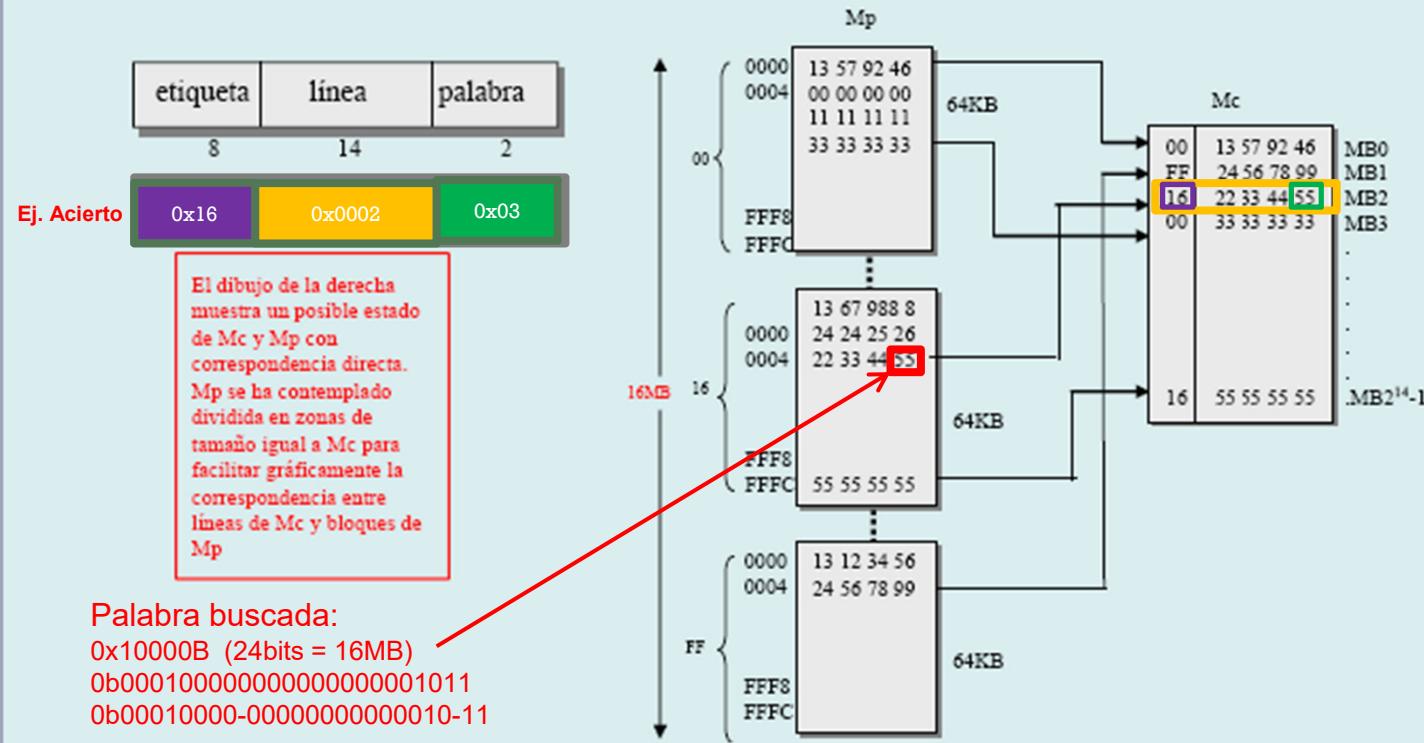
## Correspondencia directa

- Ejemplo: para los tres tipos de correspondencia utilizaremos los siguientes datos:

Tamaño de bloque K = 4 bytes =  $2^2 \Rightarrow w = 2$

Tamaño de Mc = 64 KBytes =  $2^{16}$  Bytes =  $2^{14}$  marcos de bloque  $\Rightarrow r = 14$

Tamaño de Mp = 16 MBytes =  $2^{24}$  Bytes =  $2^{22}$  bloques  $\Rightarrow s = 22$



# Memoria Caché : correspondencia

## Correspondencia asociativa

- Un bloque  $B_j$  de  $M_p$  se puede ubicar en cualquier marco de bloque de  $M_c$ .



$2^s$  bloques de  $M_p$

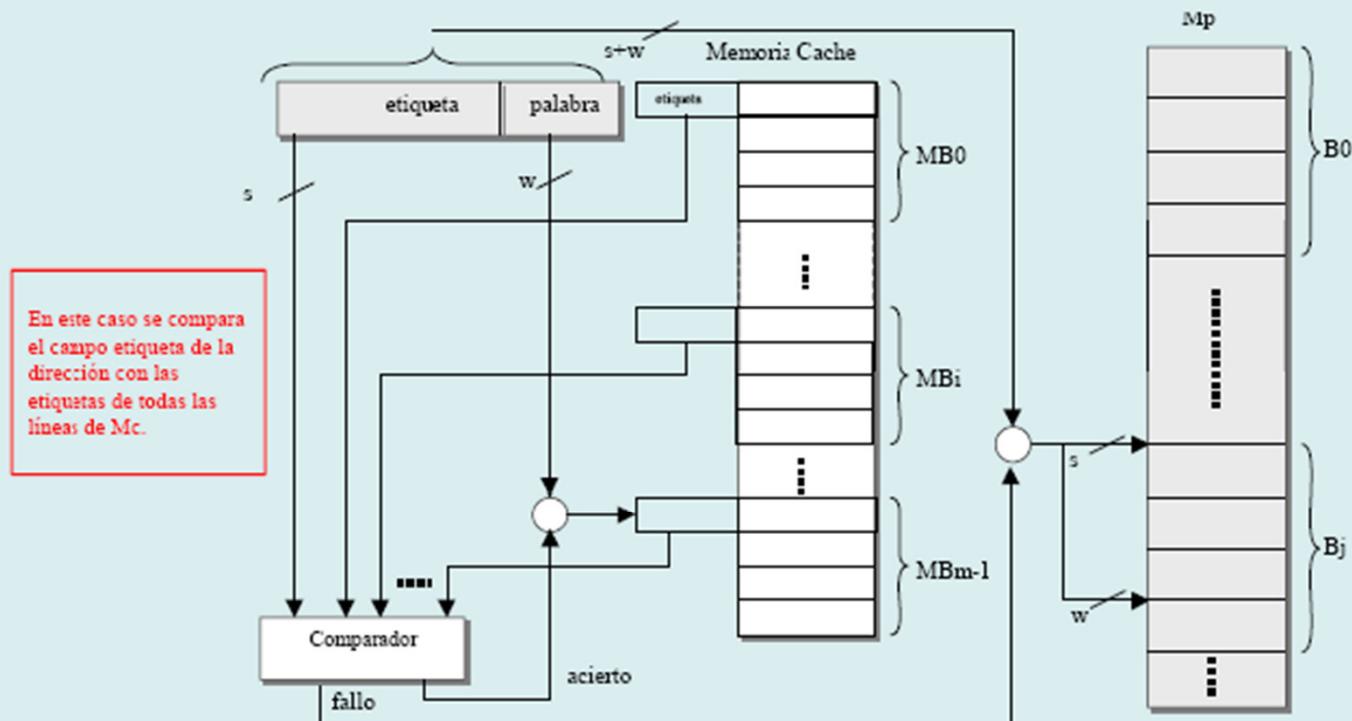
$2^r$  marcos de bloque de  $M_c$

- La etiqueta tiene  $s$  bits para poder diferenciar a cada uno de los bloques de  $M_p$  (todos) que pueden ubicarse en el mismo marco de bloque de  $M_c$ .
- El directorio caché en correspondencia asociativa contendrá, pues, un registro de  $s$  bits por cada marco de bloque para contener la etiqueta del bloque ubicado en ese momento en dicho marco

# Memoria Caché : correspondencia

## Correspondencia asociativa

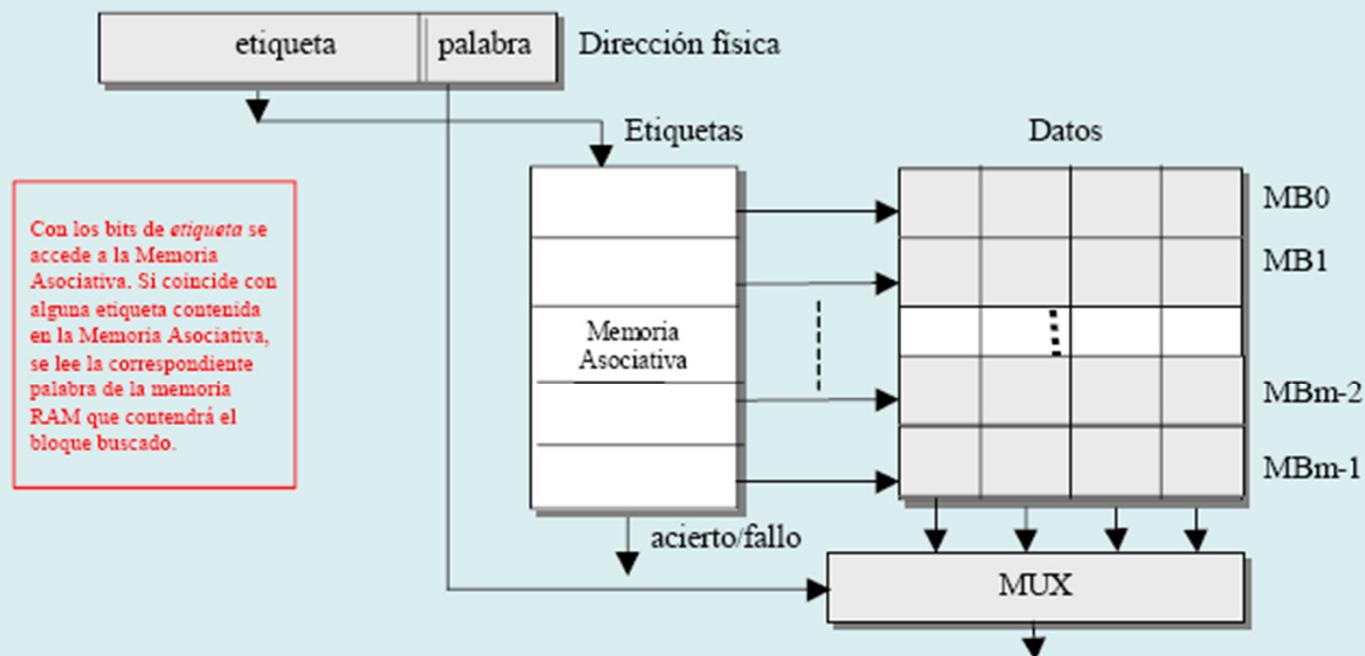
- El esquema de acceso a una Mc con correspondencia asociativa es el siguiente:



# Memoria Caché : correspondencia

## Correspondencia asociativa

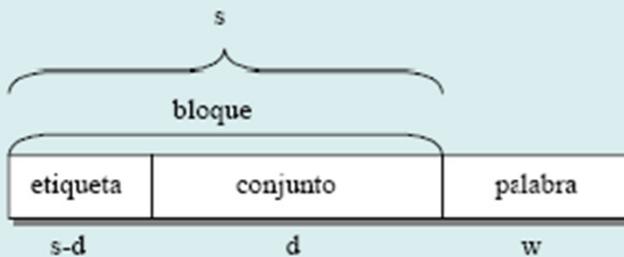
- Desde el punto de vista del diseño una Mc con correspondencia asociativa se organiza con una memoria RAM con longitud de palabra suficiente para contener una línea, y una Memoria Asociativa para contener las etiquetas de los bloques actualmente en Mc.
- El esquema de acceso es el siguiente:



# Memoria Caché : correspondencia

## Correspondencia asociativa por conjuntos

- Las líneas de  $Mc$  se dividen en  $v = 2^d$  conjuntos con  $k$  líneas/conjunto o *vías* cada uno.
- Se cumple que el número total de marcos de bloque (líneas) que tiene la caché  $m = v * k$ .
- Un bloque  $B_j$  de  $M_p$  se puede ubicar sólo en el conjunto  $C_i$  de  $Mc$  que cumple  $i = j \bmod v$ .



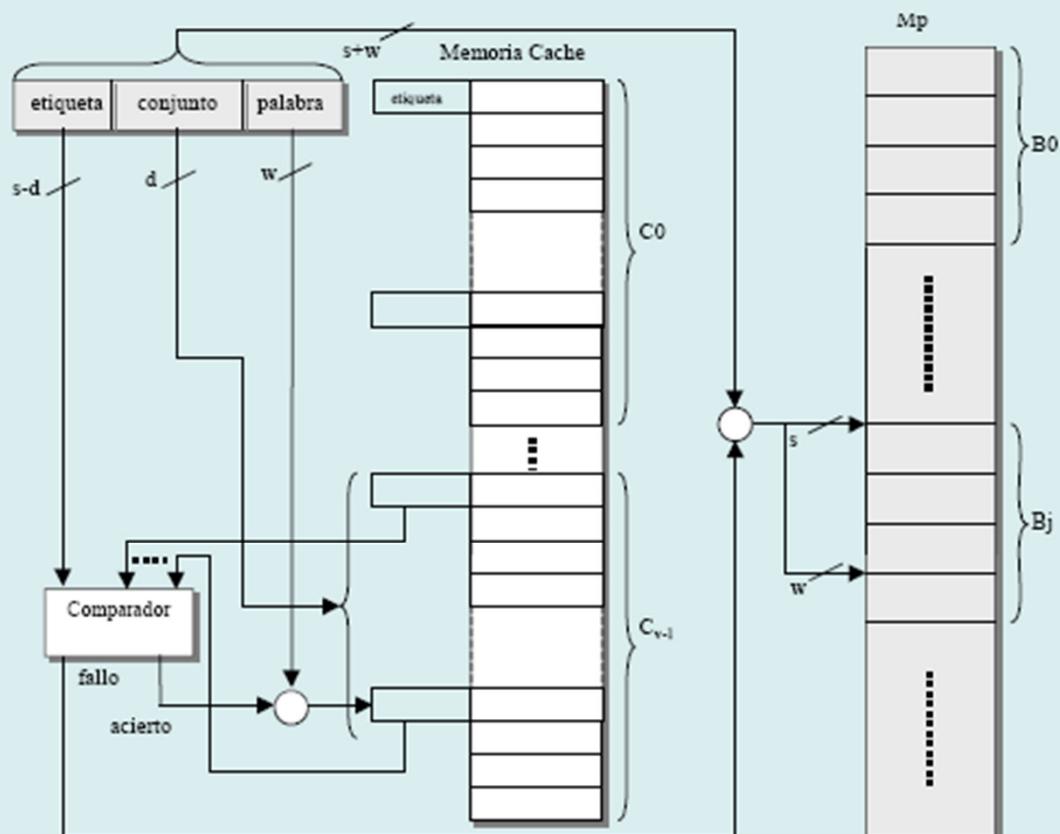
- La etiqueta tiene  $s - d$  bits para poder diferenciar a cada uno de los bloques de  $M_p$  que pueden ubicarse en el mismo conjunto de  $Mc$ .
- El directorio caché en correspondencia asociativa por conjuntos contendrá, pues, un registro de  $s - d$  bits por cada conjunto de líneas de  $Mc$ .

# Memoria Caché : correspondencia

## Correspondencia asociativa por conjuntos

- El esquema de acceso a una Mc con correspondencia asociativa por conjuntos es el siguiente:

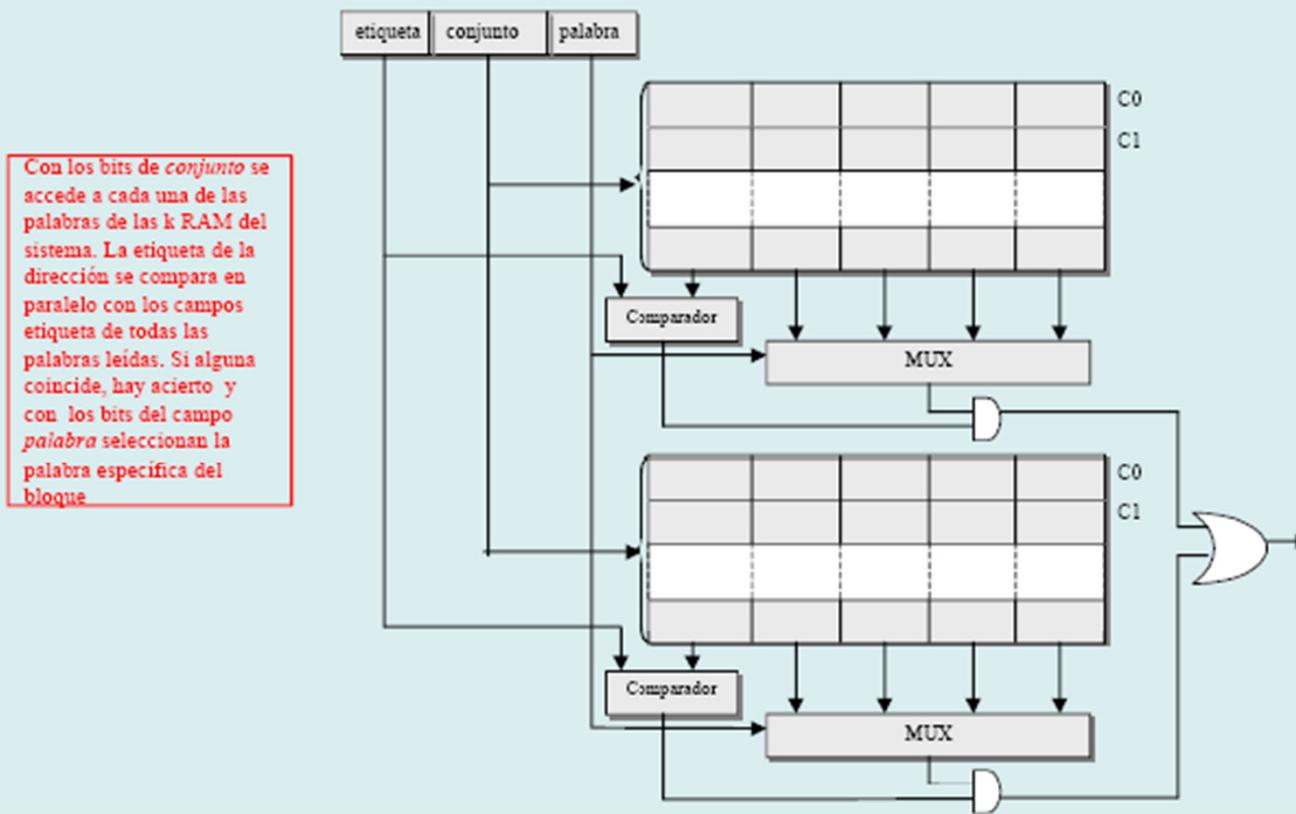
Los bits del campo *conjunto* de la dirección determinan el conjunto leído. Si la etiqueta de la dirección coincide con la etiqueta de alguna de las líneas del conjunto accedido, hay acierto, es decir, el bloque contenido en dicha línea es el mismo al que pertenece la dirección que accede al sistema de memoria. En caso contrario se produce un fallo de caché.



# Memoria Caché : correspondencia

## Correspondencia asociativa por conjuntos

- Una memoria de correspondencia asociativa de grado k se organiza como un conjunto de k memorias de correspondencia directa, cada una con su comparador.
- Para el caso k= 2 (2 vías) el esquema sería el siguiente:



# Memoria Caché : correspondencia

## Correspondencia asociativa por conjuntos

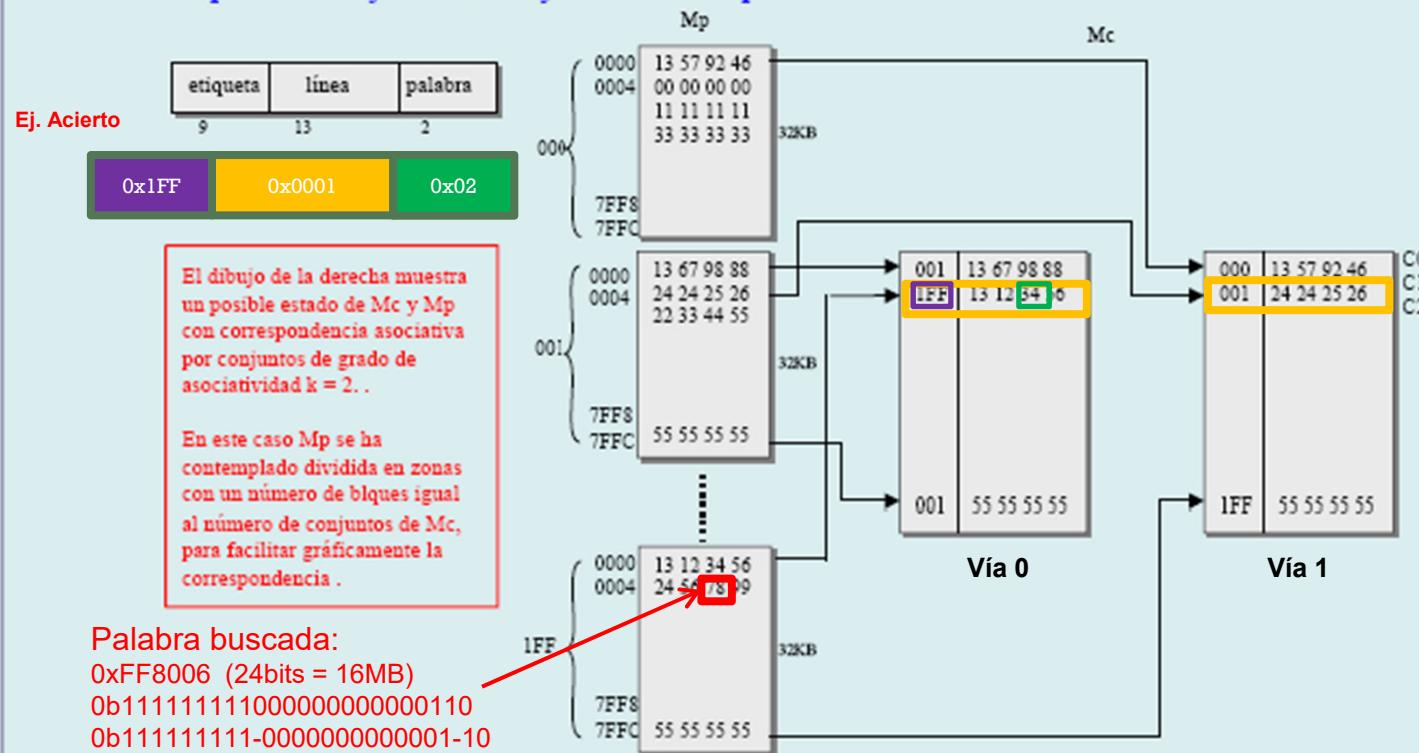
- Ejemplo (anterior).

$$k = 2 \Rightarrow v = m/k = 2^r/2 = 2^{13} \Rightarrow d = 13; w = 2; s = 22 \Rightarrow s - d = 9$$

$$\text{Bloque} = 4 \text{ bytes} = 2^2 \Rightarrow w = 2$$

$$M_c = 64 \text{ KBytes} = 2^{16} \text{ Bytes} = 2^{14} = 2^{14} \text{ líneas} \Rightarrow r = 14$$

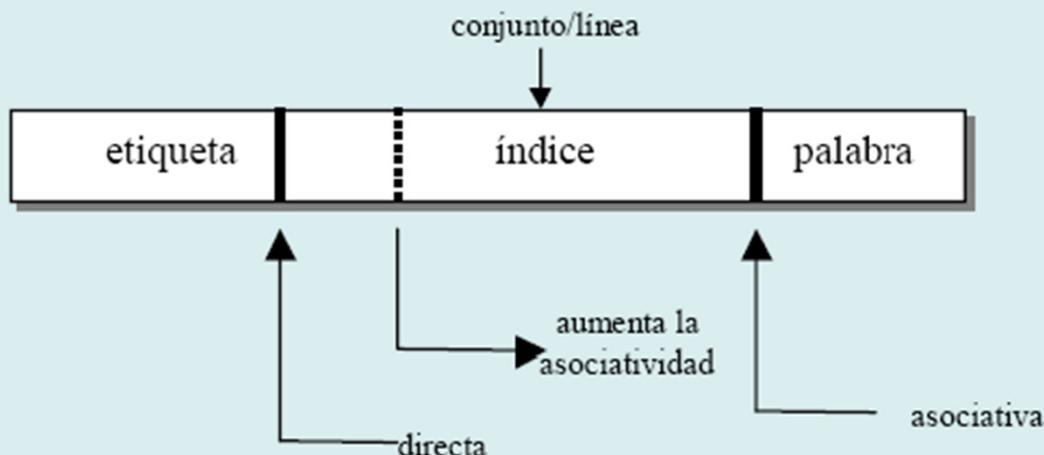
$$M_p = 16 \text{ MBytes} = 2^{24} \text{ Bytes} = 2^{22} \text{ bloques} \Rightarrow s = 22$$



# Memoria Caché : correspondencia

## Funciones de correspondencia unificadas: asociativa por conjunto

- Las tres funciones de correspondencia se pueden ver en realidad como una sola, la asociativa por conjunto, siendo las otras dos correspondencias casos extremos de ella:



Si  $k = 1 \Rightarrow m = v \Rightarrow$  asociativa por conjuntos de 1 vía = directa

Si  $v = 1 \Rightarrow m = k \Rightarrow$  asociativa por conjuntos de  $m$  vías = asociativa

# Memoria Caché : correspondencia

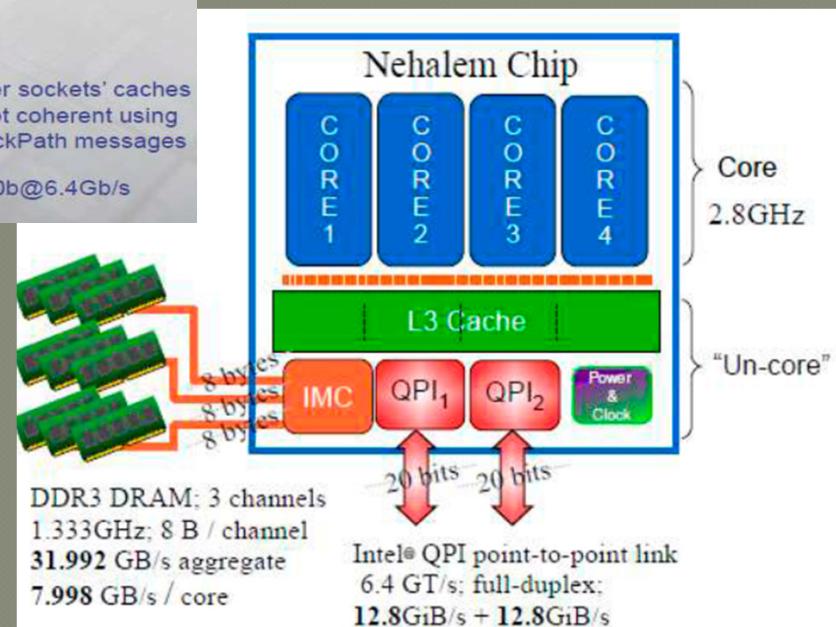
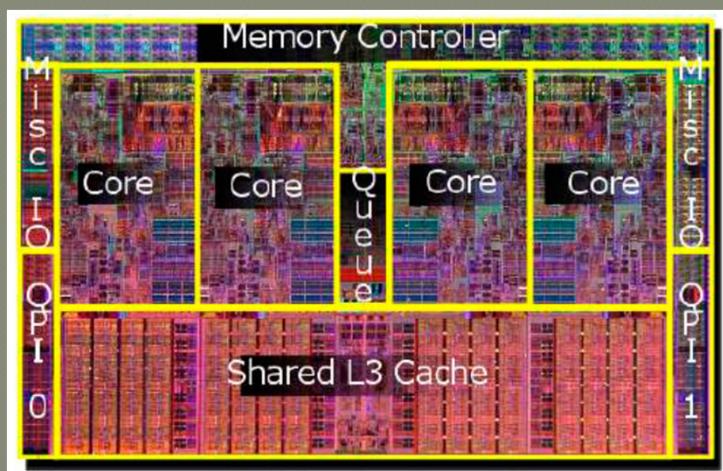
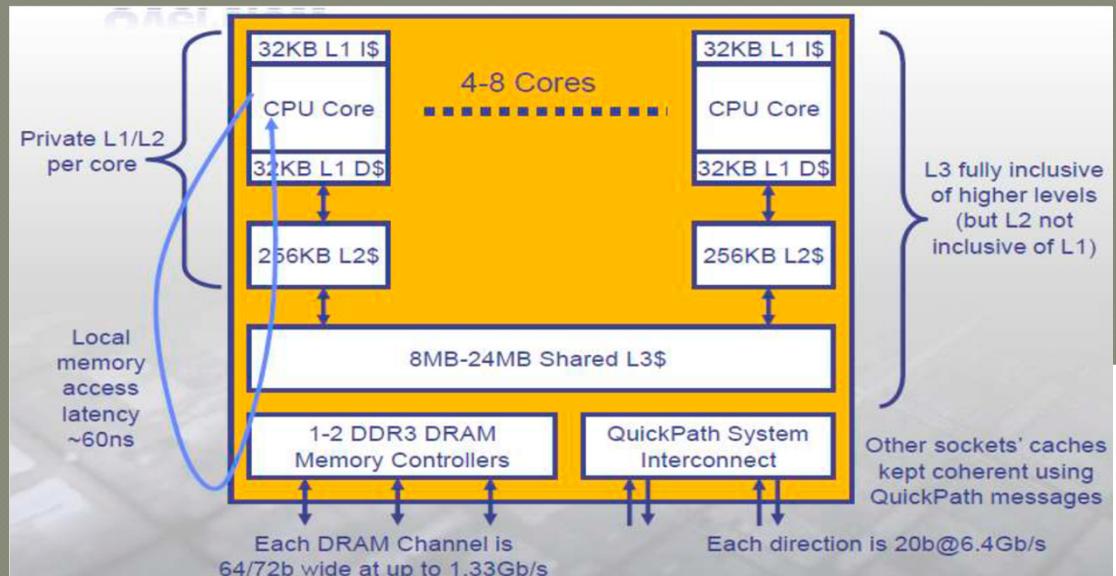
---

¿Cuál es mejor?

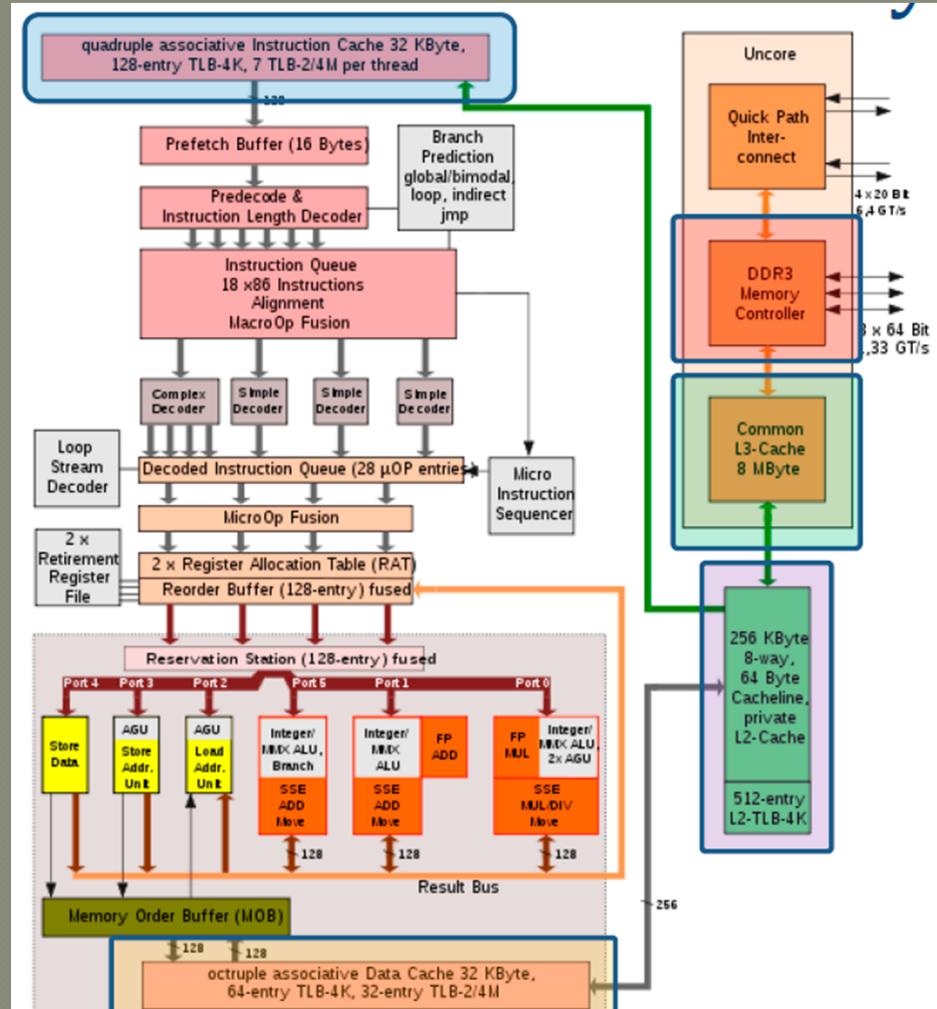
Compromiso entre:

- Eficacia : mayor o menor números de aciertos.
- Complejidad : más complejo es más caro y voluminoso.
- Velocidad: más grande es más lento con la misma tecnología.

# Memoria Caché : Intel I7-Nehalem



# Memoria Caché : Intel I7-Nehalem



1. 4-way set associative instruction cache
2. 8-way set associative L1 data cache (32 KB)
3. 8-way set associative L2 data cache (256 KB)
4. 16-way shared L3 cache (8 MB)
5. 3 DDR3 memory connections