

Jerarquía de Memoria

Estructura de Computadores
Semana 14

Bibliografía:

[BRY16] Cap.6 Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016
Signatura ESIT/C.1 BRY com

Transparencias del libro CS:APP, Cap.6

Introduction to Computer Systems: a Programmer's Perspective

Autores: Randal E. Bryant y David R. O'Hallaron

<http://www.cs.cmu.edu/afs/cs/academic/class/15213-f15/www/schedule.html>

Guía de trabajo autónomo (4h/s)

■ **Lectura:** del Cap.6 CS:APP (Bryant/O'Hallaron)

- Accessing Main Memory, Random Access Memory, *Enhanced DRAMs*
 - § 6.1.1 pp.623-625, 615-621, 621-622
- Locality, The Memory Hierarchy
 - § 6.2 – 6.3 pp.640-650
- Disk Storage, *Nonvolatile Memory*, Solid State Disks, *Storage Technology Trends*
 - § 6.1.2 – .1.3 pp.625-636, 622-623, 636-638-640

■ **Ejercicios:** del Cap.6 CS:APP (Bryant/O'Hallaron)

- Probl. 6.1 § 6.1.1, p.620
- Probl. 6.7 – 6.8 § 6.2.3, pp.644, 645
- Probl. 6.2 – 6.6 § 6.1.2, pp.628, 631, 632, 637, 640

Bibliografía:

[BRY16] Cap.6

Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016

Signatura ESIIT/[C.1 BRY com](#)

Memoria I: Jerarquía de memoria

- **La abstracción de memoria (concepto de Lectura y Escritura)**
- RAM: bloque constructivo de memoria principal
- Configuración y diseño de memorias utilizando varios chips
- Localidad de las referencias
- Jerarquía de memoria
- Tecnologías de almacenamiento, y tendencias

Lectura y Escritura (de/a) Memoria

■ Escritura

- Transferir datos de CPU a memoria

```
movq %rax, 8(%rsp)
```

- Operación “Store”

■ Lectura

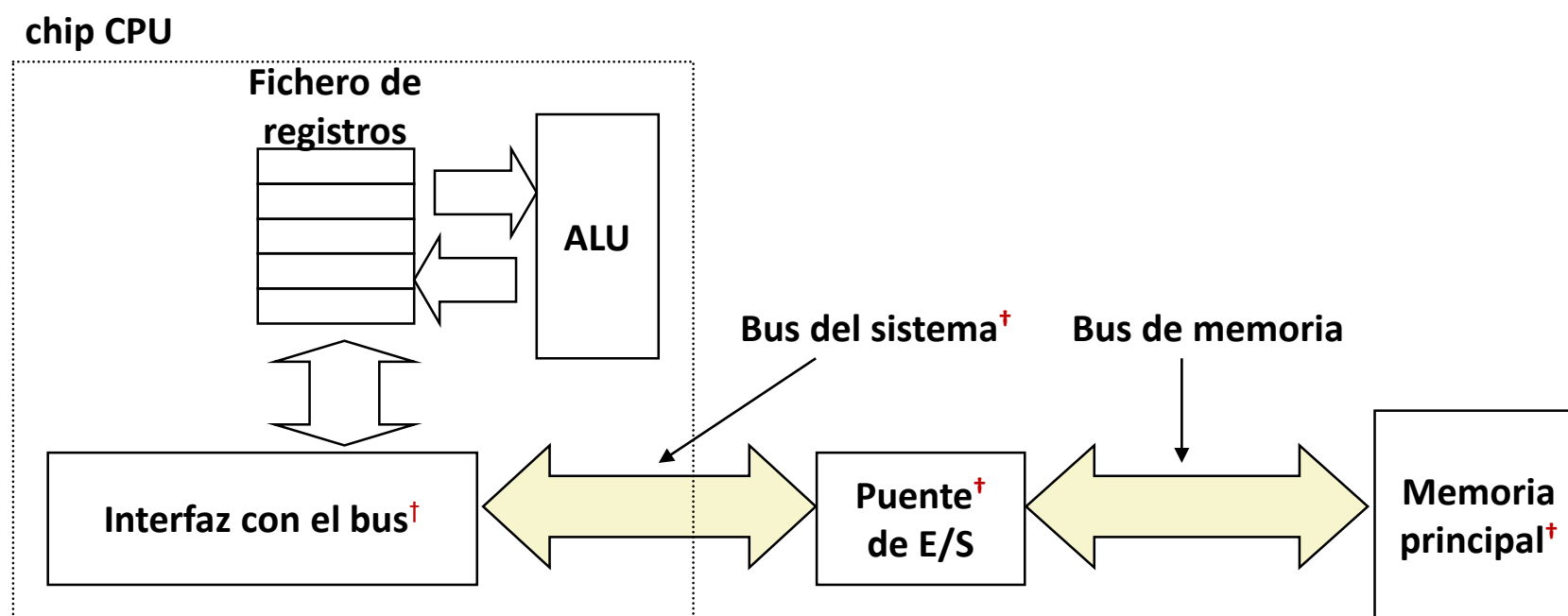
- Transferir datos de memoria a CPU

```
movq 8(%rsp), %rax
```

- Operación “Load”

Estructura de buses CPU – Memoria (tradicional[†])

- Un **bus** es un conjunto de cables en paralelo que transportan direcciones, datos, y señales de control
- Típicamente a un bus se conectan varios dispositivos

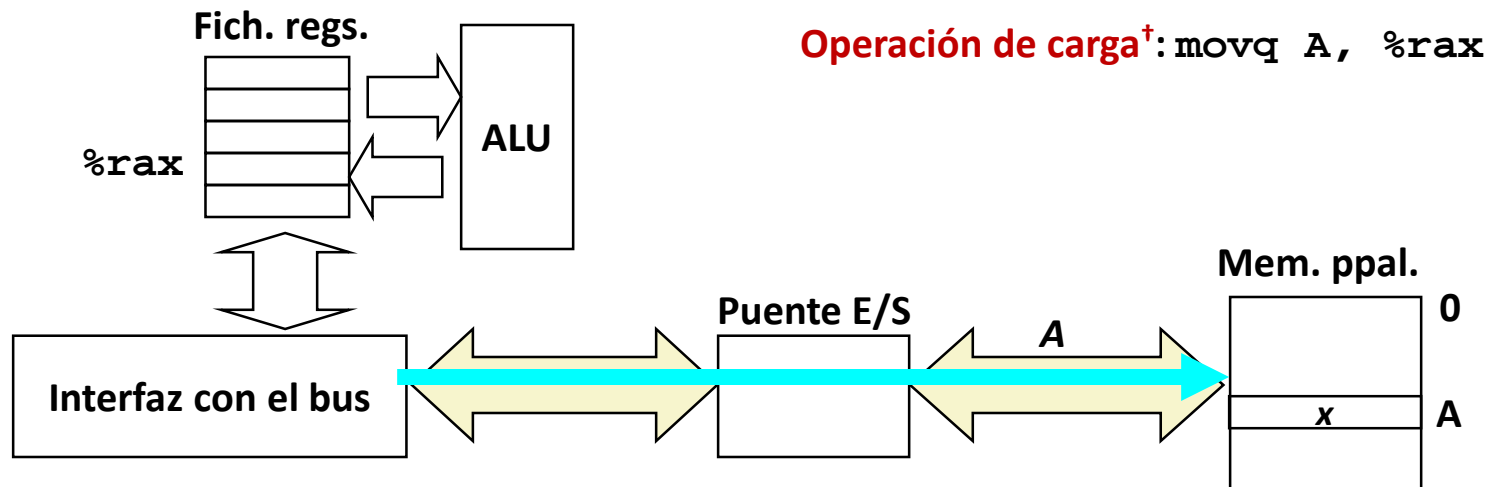


[†] "System bus", "Bus interface", "I/O bridge", "Main memory".

[‡] North/SouthBridge (1990), Intel Hub Arch (IHA=MCH/ICH 1999), Platform Controller Arch=(IMC/)PCH (2008), IOH/ICH (2008).

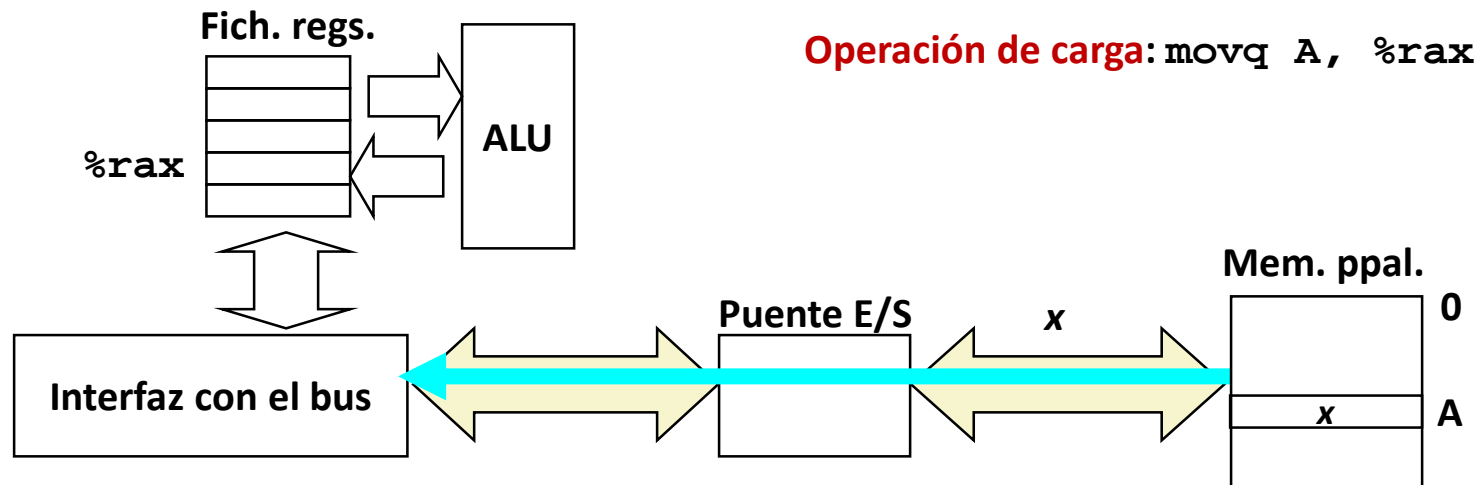
Transacción de Lectura de Memoria (1)

- La CPU pone la dirección A en el bus de memoria



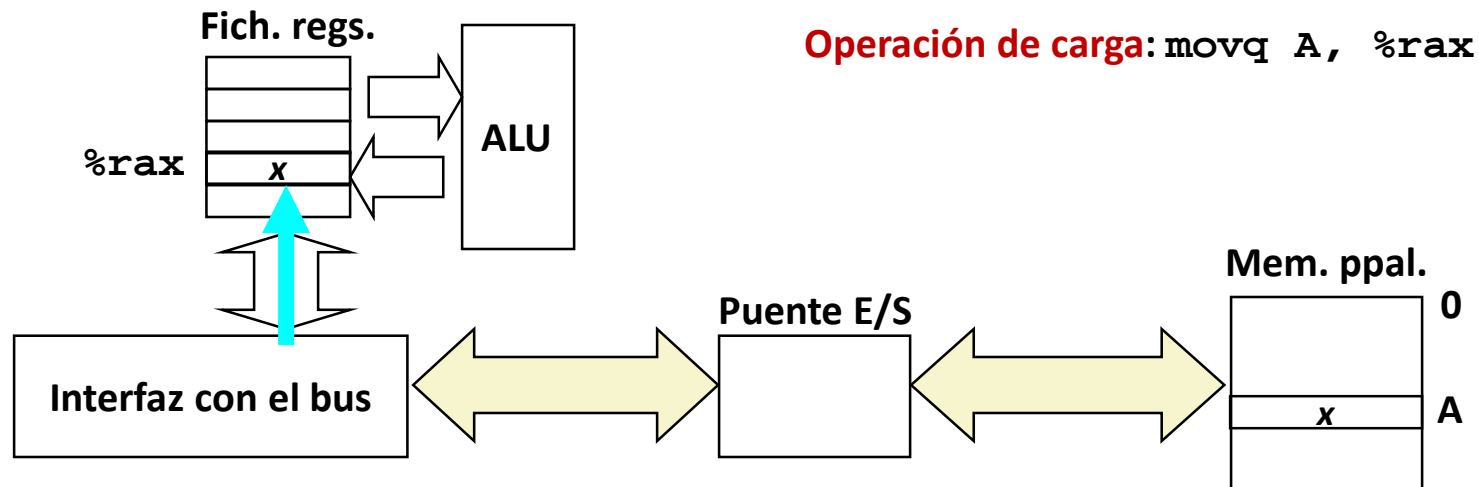
Transacción de Lectura de Memoria (2)

- La memoria principal recibe dirección A del bus de memoria, recupera la palabra x, y la pone en el bus



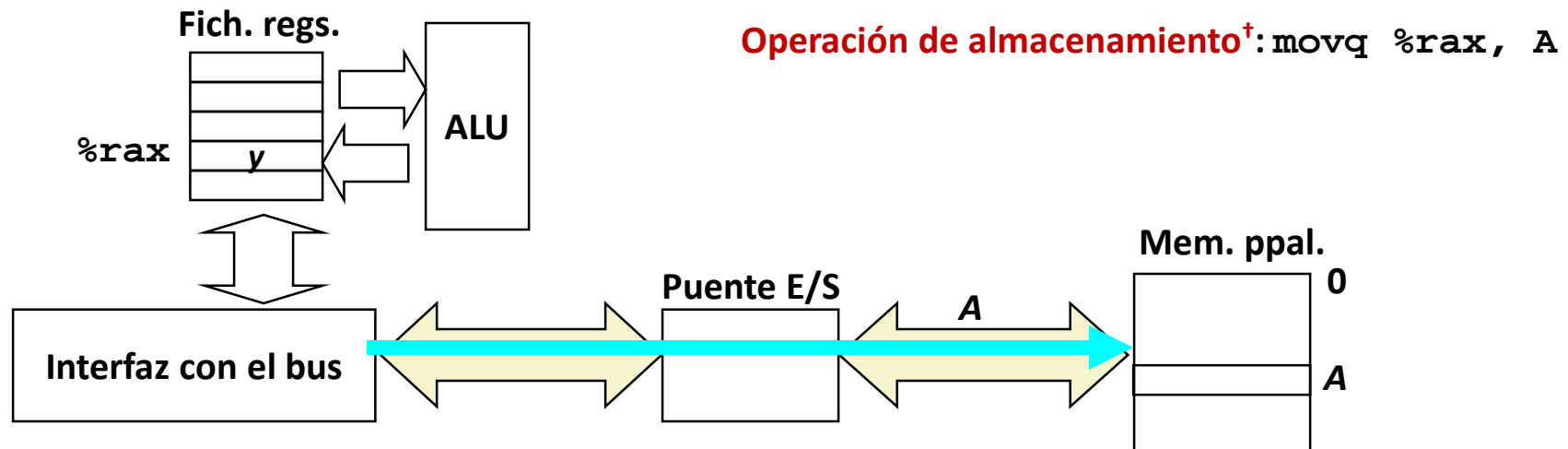
Transacción de Lectura de Memoria (3)

- La CPU lee palabra x del bus y la copia al registro `%rax`



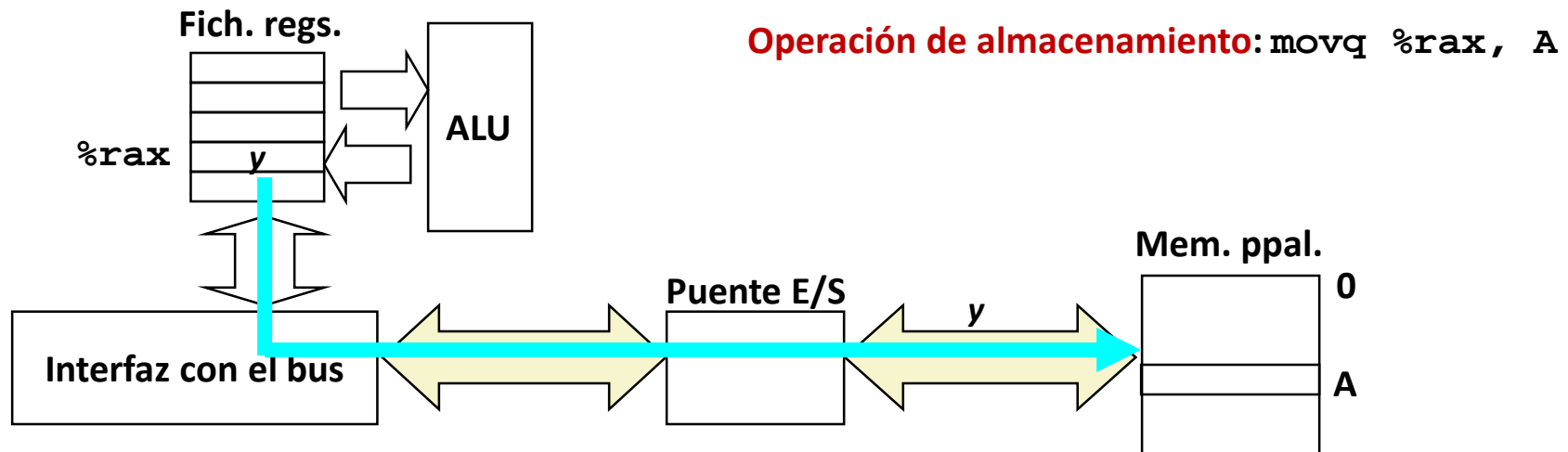
Transacción de Escritura a Memoria (1)

- La CPU pone dirección A en el bus. La mem. ppal. la recibe y espera a que llegue la palabra de datos correspondiente



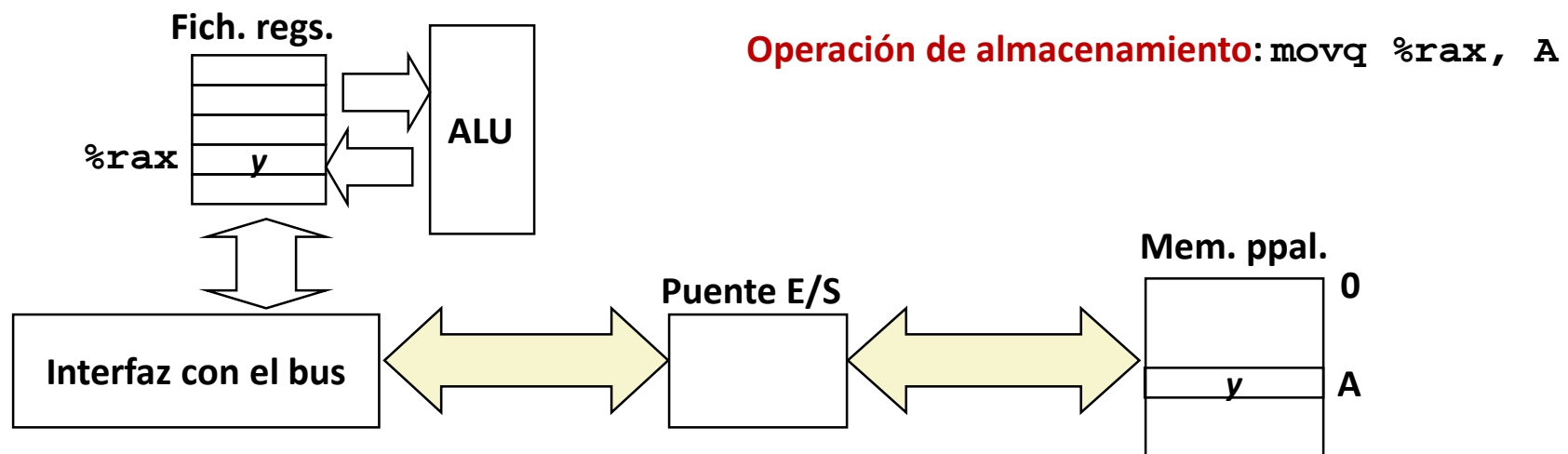
Transacción de Escritura a Memoria (2)

- La CPU pone la palabra de datos y en el bus



Transacción de Escritura a Memoria (3)

- La memoria principal recibe la palabra de datos y del bus y la almacena en la posición con dirección A



Memoria I: Jerarquía de memoria

- La abstracción de memoria (concepto de Lectura y Escritura)
- **RAM: bloque constructivo de memoria principal**
- Configuración y diseño de memorias utilizando varios chips
- Localidad de las referencias
- Jerarquía de memoria
- Tecnologías de almacenamiento, y tendencias

Algunas definiciones

■ Tiempo de acceso:

- tiempo que se requiere para leer (o escribir) un dato (palabra) en la memoria
 - medido en ciclos o en (n-μ-m) s

■ Ancho de banda: (de la memoria de un computador)

- Número de palabras a las que puede acceder el procesador (o que se pueden transferir entre el procesador y la memoria) por unidad de tiempo
 - medido en (K-M-G) B/s

Algunas definiciones

■ Métodos de acceso:

- **Aleatorio (RAM)**: tiempo de acceso independiente de posición a acceder
 - SRAM, ROM
- **Secuencial (SAM)**: t. acceso depende de posición de los datos a acceder
 - Cinta magnética
- **Directo** (semialeatorio, **DASD** – direct access storage device)
 - tiempo acceso tiene una componente aleatoria y otra secuencial
 - Discos giratorios (época en que lat. rotacional \gg t. búsqueda)
- actualmente muchos dispositivos tienen 2 o más componentes acceso
 - DRAM: $CL - T_{RCD} - T_{RP} - T_{RAS}$
 - (CAS latency, Row-Col delay, Row precharge, Row active)
 - No es lo mismo acceder a nueva fila, a otra columna de la misma fila, a ráfaga...

Memoria de Acceso Aleatorio (RAM)

■ Características principales

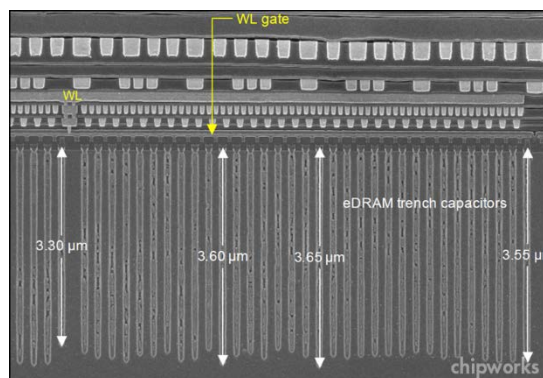
- La RAM tradicionalmente se empaqueta como un chip
 - o incluida[†] como parte de un chip procesador
- Unidad básica almacenamiento es normalmente una celda (1 bit/celda)
- Múltiples chips de RAM forman una memoria

■ La RAM tiene dos variedades:

- SRAM (RAM estática)
- DRAM (RAM Dinámica)

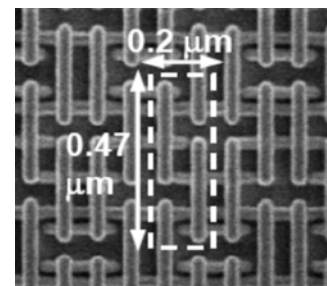
Tecnologías RAM

■ DRAM



- **(1 Transistor + 1 condensador) / bit**
 - Condensador orientado verticalmente
- **Debe refrescar estado periódicamente**

■ SRAM

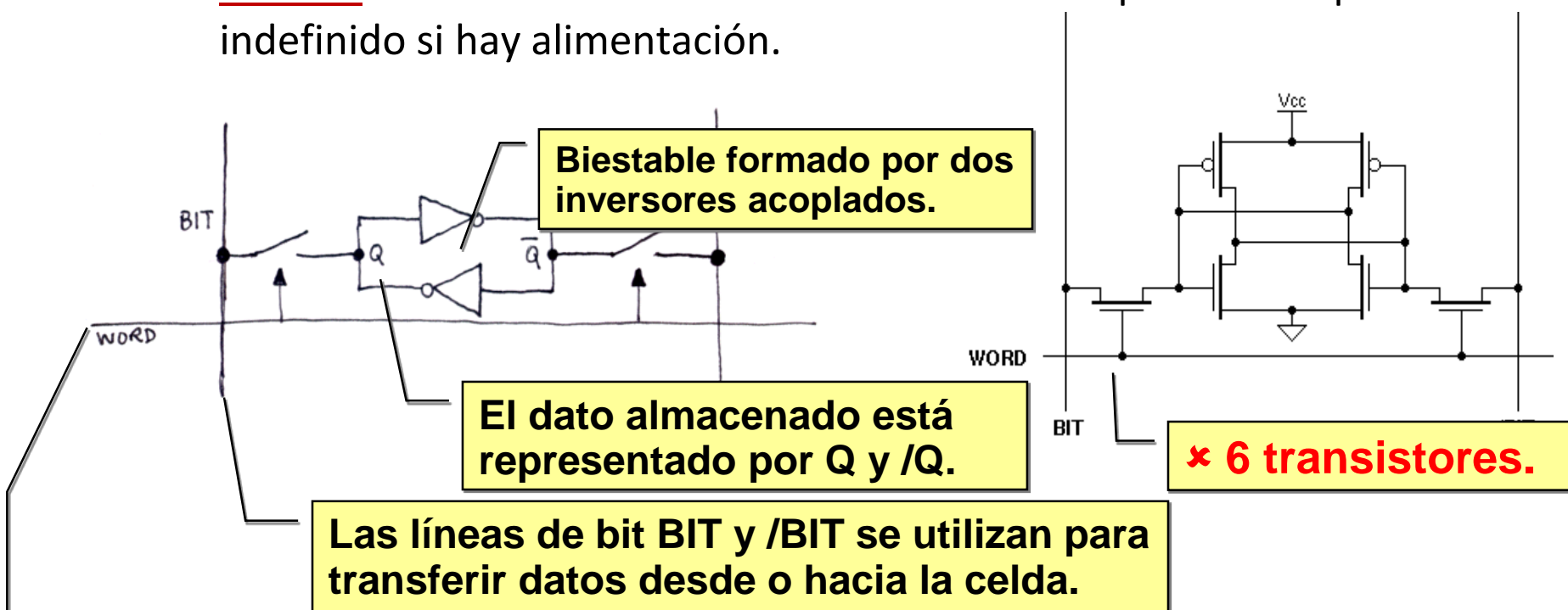


- **6 transistores / bit**
 - 2 inversores (x 2 tr) + 2 puertas de paso
- **Mantiene el estado indefinidamente**

SRAM

■ Celda de memoria SRAM

- Estática \Rightarrow los datos almacenados se mantienen por un tiempo indefinido si hay alimentación.

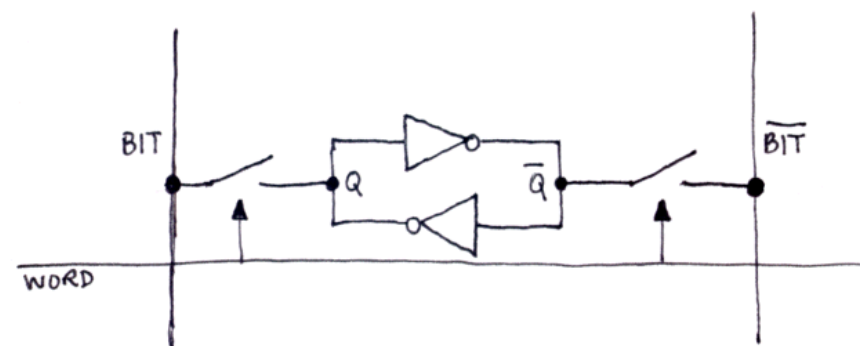


- Cuando la línea **WORD** esté a 1 se selecciona la celda para lectura o escritura.
- Cuando la línea **WORD** esté a 0 los dos transistores asociados no conducen, desconectando la celda de las líneas de bit.

SRAM

■ Celda de memoria SRAM

- Operación de lectura:
 - Se selecciona la celda poniendo WORD a 1 \Rightarrow Q se conecta a BIT y \bar{Q} a \bar{BIT} .
 - Si $Q = 1$ ($\bar{Q} = 0$) \Rightarrow la línea \bar{BIT} se pone a 0.
 - $BIT = 1$ y $\bar{BIT} = 0 \Rightarrow$ se lee un 1.
 - Si $Q = 0$ ($\bar{Q} = 1$) \Rightarrow la línea BIT se pone a 0.
 - $BIT = 0$ y $\bar{BIT} = 1 \Rightarrow$ se lee un 0.



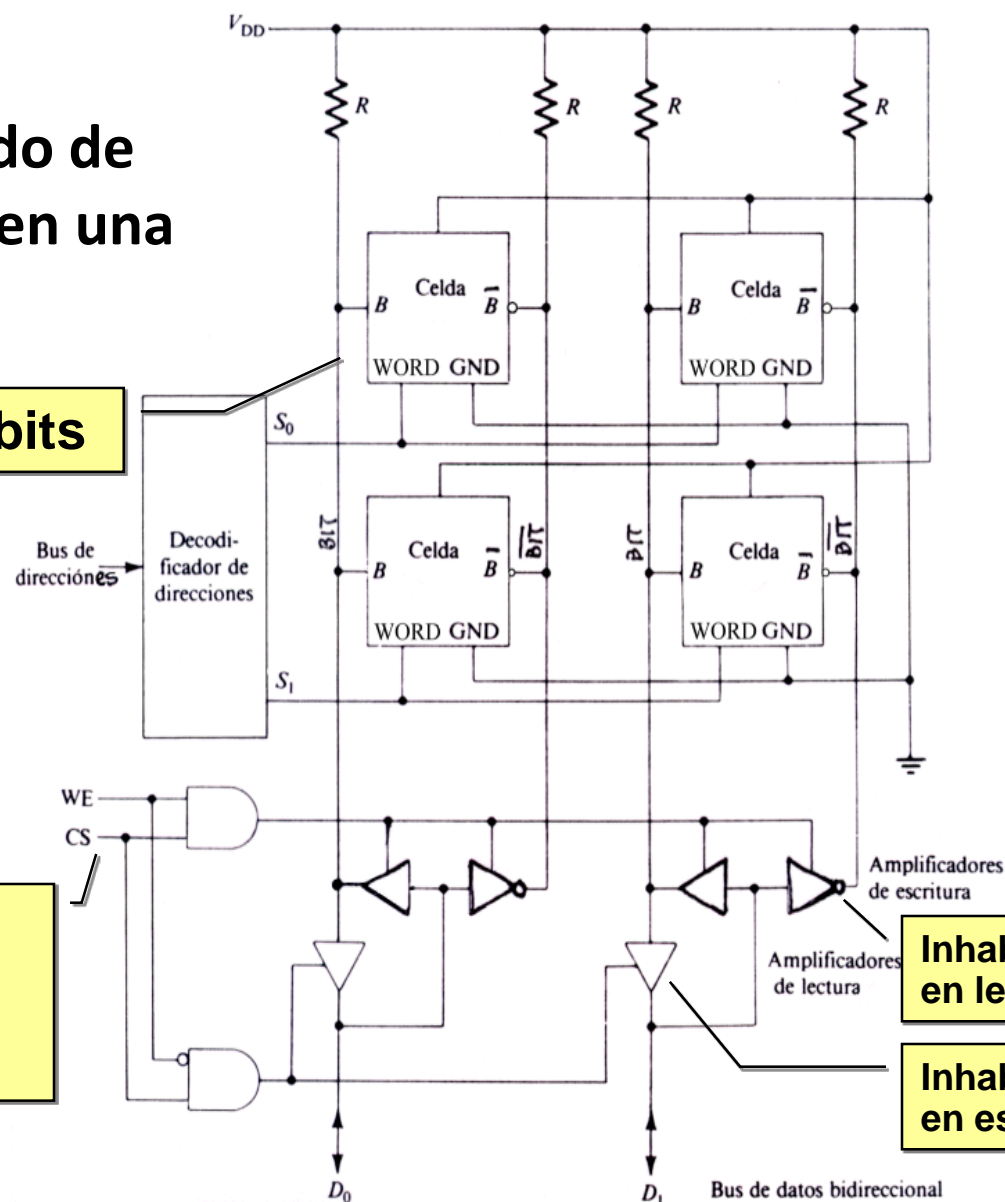
- ✓ **Lectura no destructiva.**
- ✓ **Mayor velocidad que DRAM.**

SRAM

- Esquema simplificado de conexión de celdas en una SRAM:

SRAM de 2 palabras de 2 bits

Si $CS = 1 \Rightarrow$
 $WE = 1 \Rightarrow$ Escritura
 $WE = 0 \Rightarrow$ Lectura



Inhabilitados en lectura

Inhabilitados en escritura

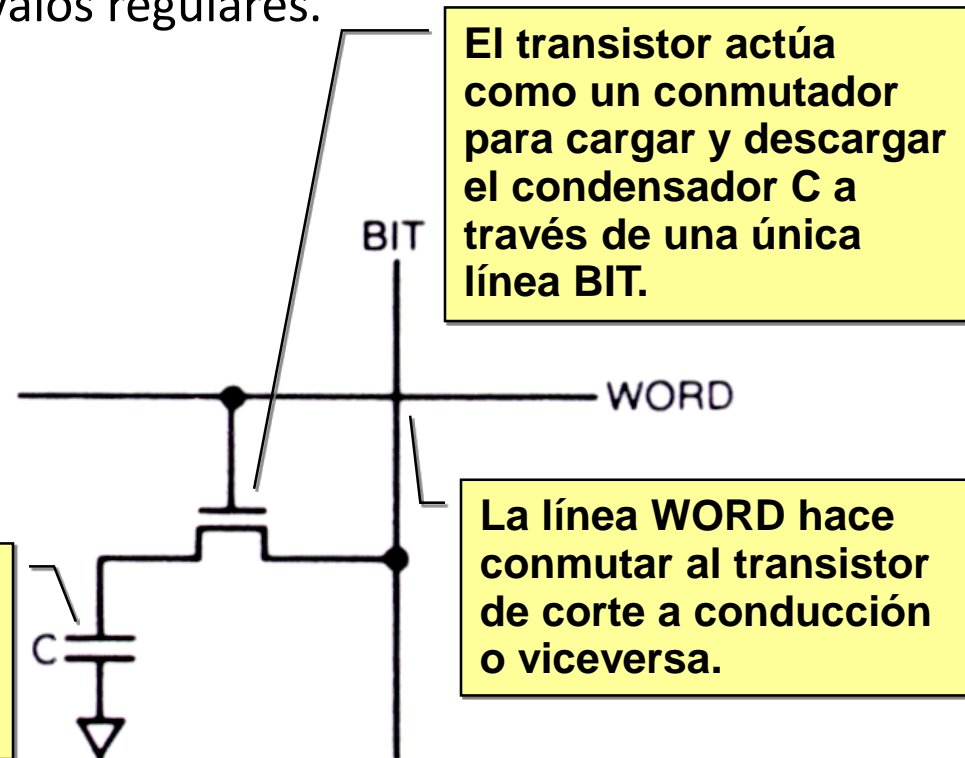
DRAM

■ Celda de memoria DRAM

- Dinámica \Rightarrow los datos almacenados decaen o se desvanecen y deben ser restaurados a intervalos regulares.

✓ Sencillez de la celda de almacenamiento.

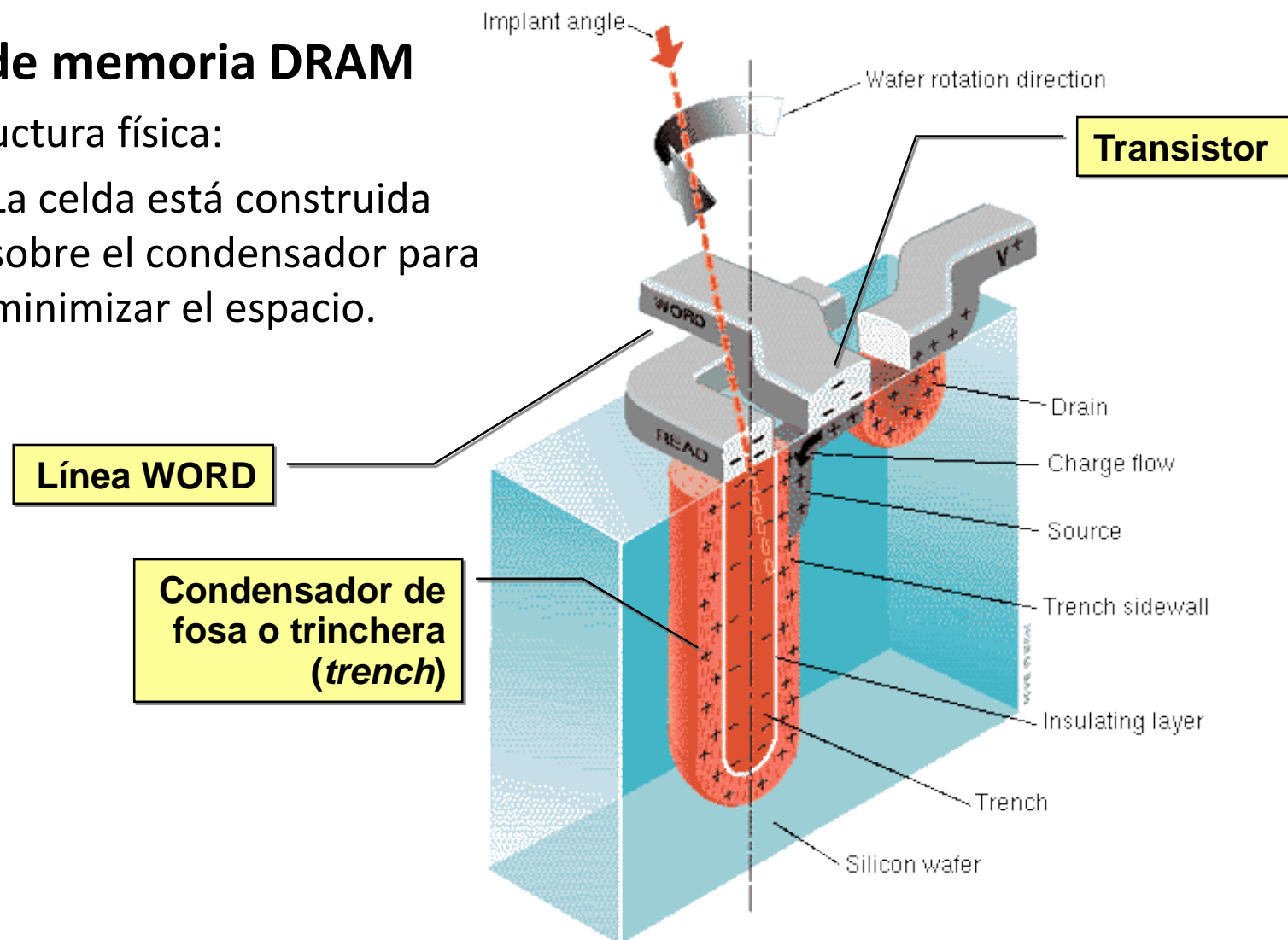
La información se almacena en forma de carga eléctrica en un condensador C. La presencia (o ausencia) de carga indica que hay almacenado un 1 (o un 0).



DRAM

■ Celda de memoria DRAM

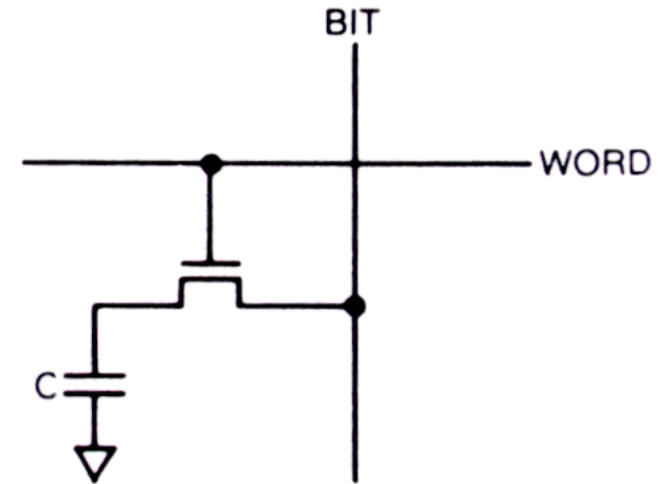
- Estructura física:
 - La celda está construida sobre el condensador para minimizar el espacio.



DRAM

■ Celda de memoria DRAM

- Operación de lectura:
 - La circuitería externa convierte a BIT en una línea de salida, seleccionándose la celda con WORD = 1.
 - Si C está cargado (= 1) \Rightarrow se descarga a través de la línea BIT \Rightarrow se produce un pulso de corriente que es detectado por un amplificador de salida ("**sense amplifier**") \Rightarrow aparece un 1 en la línea de datos de salida.
 - Si C está descargado (= 0) \Rightarrow no se produce pulso de corriente \Rightarrow aparece un 0 en la línea de datos de salida.



× Lectura destructiva.

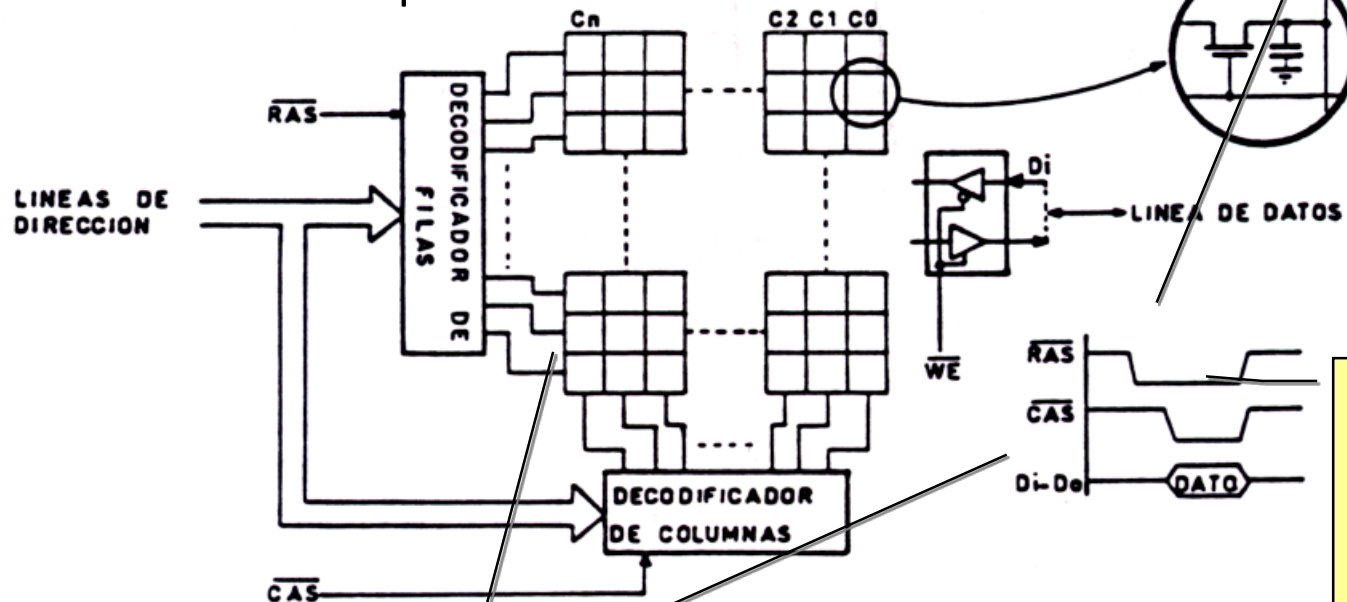
↓
Debe ir seguida de una escritura que restaure el estado original.

↓
Realizada automáticamente por los circuitos de control de la DRAM

DRAM

■ Circuitos de memoria DRAM

- Capacidad elevada de los chips de memoria DRAM \Rightarrow las direcciones han de proporcionarse multiplexadas en el tiempo.



Los bits correspondientes a las filas de la matriz de memoria se proporcionan en primer lugar, validándose por la señal **\overline{RAS} (Row Access Strobe)**.

Después se proporcionan los bits que direccionan las columnas, validados por **\overline{CAS} (Column Access Strobe)**.

Las celdas se organizan en una matriz lo más cuadrada posible.

No existe una señal CS como en las SRAM. Para dar por válido un acceso a memoria es necesaria la secuencia completa **\overline{RAS} - \overline{CAS}** junto con la señal **\overline{WE} (Write Enable)**.

DRAM

- Tiempo de acceso.
- Tiempo de ciclo.

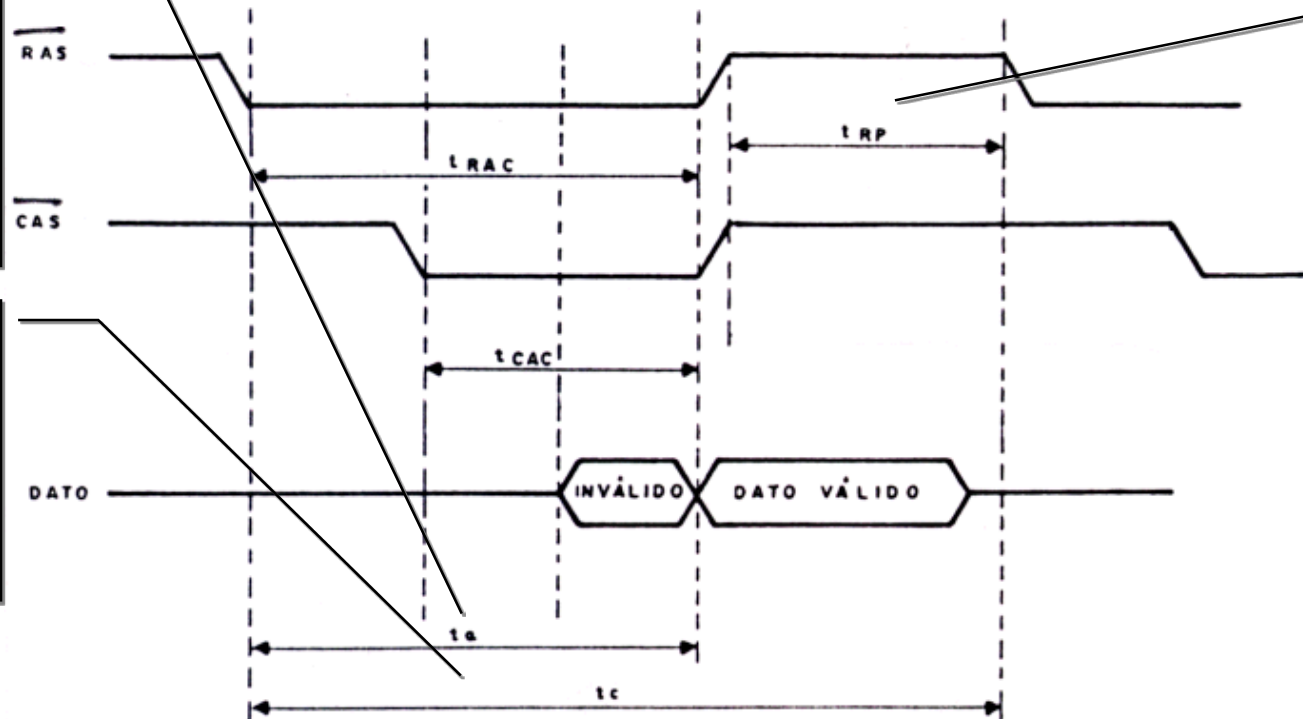
$t_{RAC} = t_a$
(tiempo de acceso):
 tiempo desde que se inicia una lectura ($/RAS\downarrow$) hasta que el dato está disponible en el bus de datos.

t_c (tiempo de ciclo):
 tiempo mínimo entre dos accesos consecutivos.

$$t_c \cong t_{RAC} + t_{RP}$$

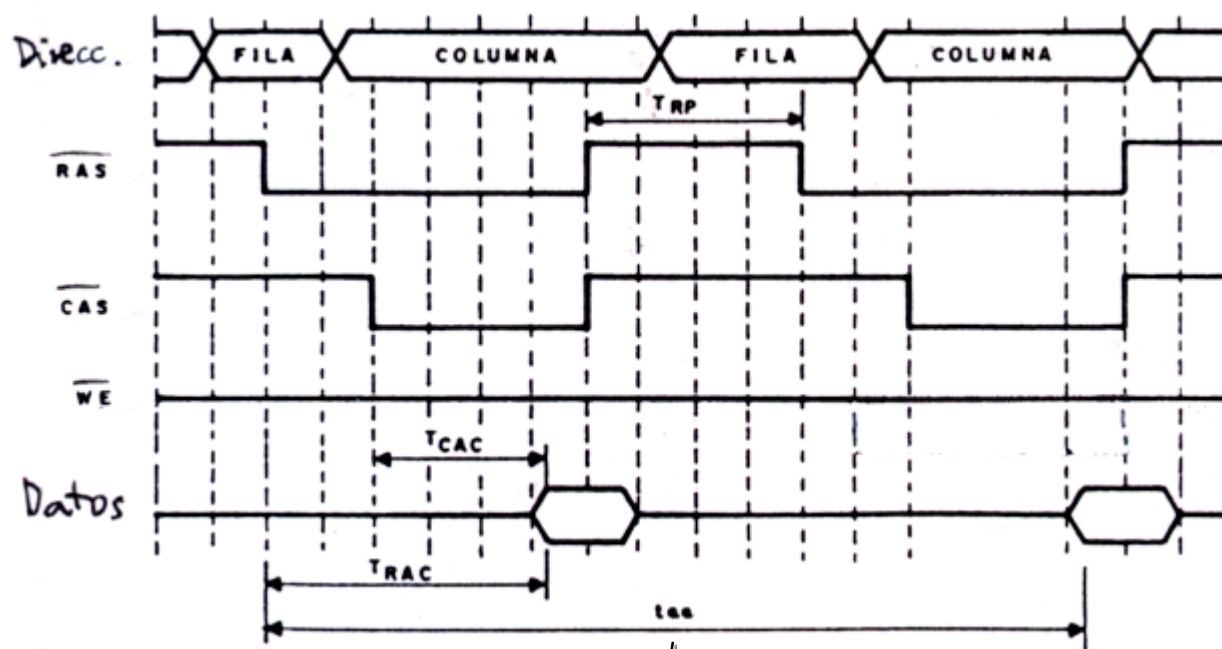
$$t_c > t_a$$

No se puede comenzar un segundo acceso tras t_a , dado que las lecturas son destructivas (el condensador de la celda de memoria se descarga) y es necesario esperar **t_{RP} (tiempo de precarga de fila)** para que la circuitería interna de la DRAM restaure el valor de las celdas de la fila correspondiente.



DRAM

- Acceso a dos palabras:



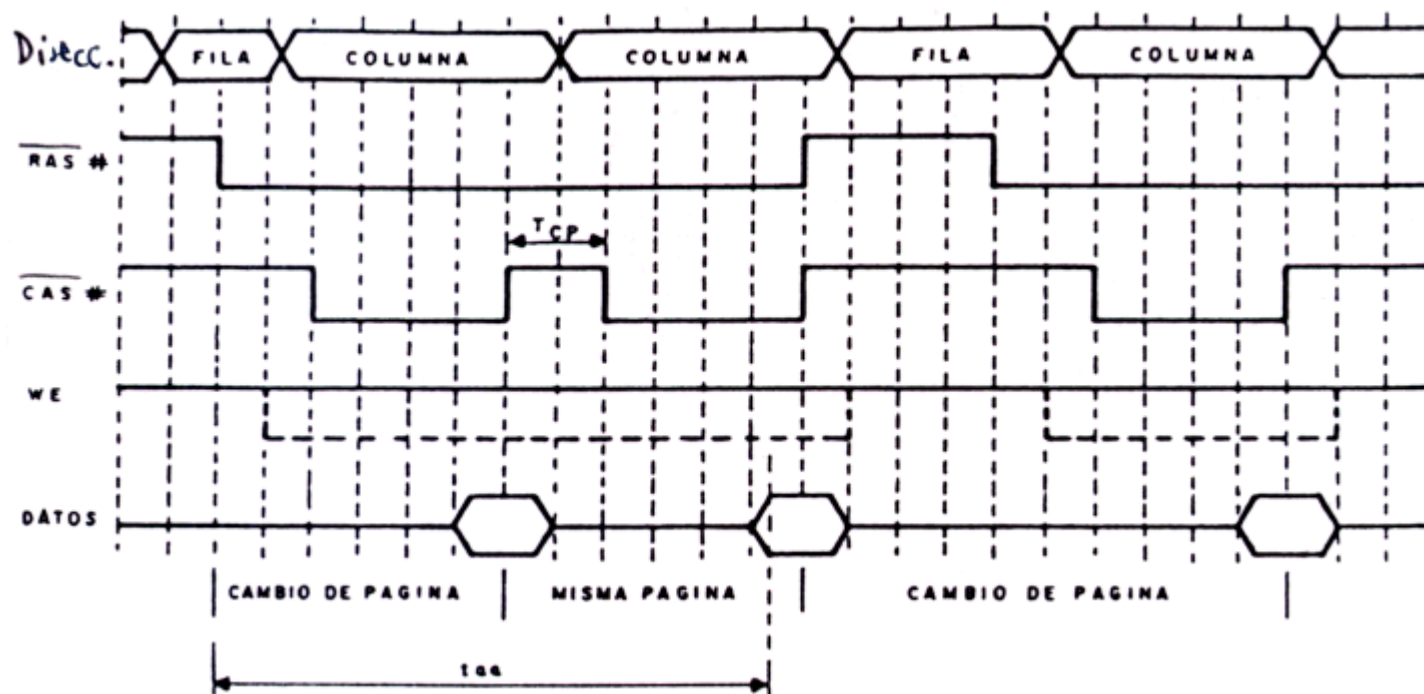
t_{aa} (tiempo de acceso a las dos palabras):

$$t_{aa} \cong t_{RAC} + t_{RP} + t_{RAC}$$

Ej.: $t_{aa} \cong 80 \text{ ns} + 60 \text{ ns} + 80 \text{ ns} = 220 \text{ ns}$

DRAM FPM

- Acceso en modo página rápido (*Fast Page Mode*):



t_{aa} (tiempo de acceso a dos palabras en modo página):

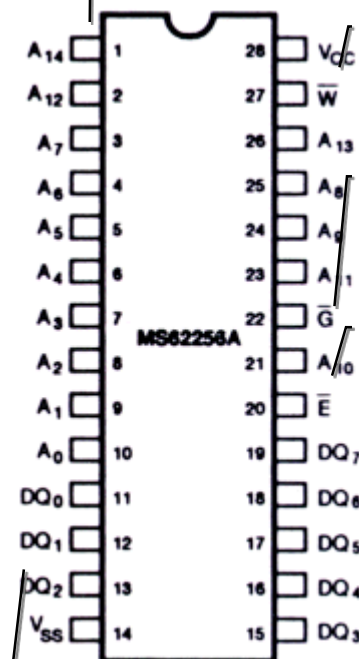
$$t_{aa} \cong t_{RAC} + t_{CP} + t_{CAC}$$

Ej.: $t_{aa} \cong 80 \text{ ns} + 10 \text{ ns} + 20 \text{ ns} = 110 \text{ ns}$

Ejemplo de SRAM

256 K bits (32 K palabras × 8 bits)

A_0 - A_{14} : 15 señales de dirección para seleccionar una de 32768 palabras de 8 bits.



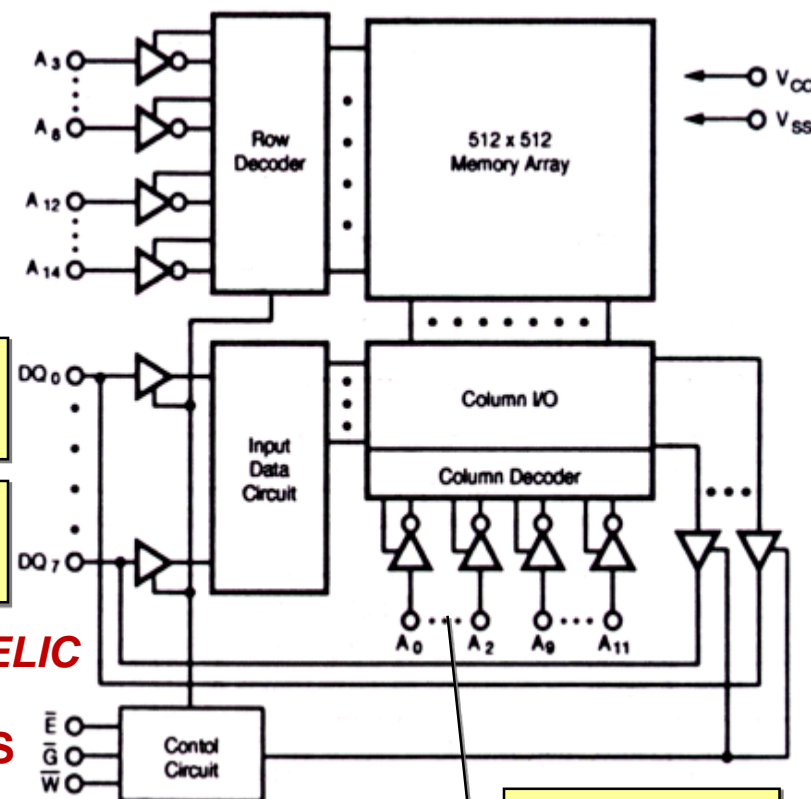
\overline{IW} (Write Enable) \equiv \overline{WE} . Se utiliza para indicar lectura o escritura.

\overline{IG} (Output Enable) \equiv \overline{OE} . Se selecciona ($=0$) para leer del chip.

\overline{IE} (Chip Enable) \equiv \overline{CS} . Si no está activa \Rightarrow el chip no está seleccionado y permanece en “standby”, pasando su consumo a 900 mW a 50 mW.

MOSEL-VITELIC
MS62256A
SRAM CMOS
32K × 8

Mode	\overline{E}	\overline{G}	\overline{W}	I/O Operation
Standby	H	X	X	High Z
Read	L	L	H	D_{OUT}
Output Disabled	L	H	H	High Z
Write	L	X	L	D_{IN}



6 líneas:
 $9 - 6 = 3 \Rightarrow$
se lee o
escribe una
palabra de
 $2^3 = 8$ bits

DQ_0 - DQ_7 (Data Input/Output)

Ejemplo de DRAM

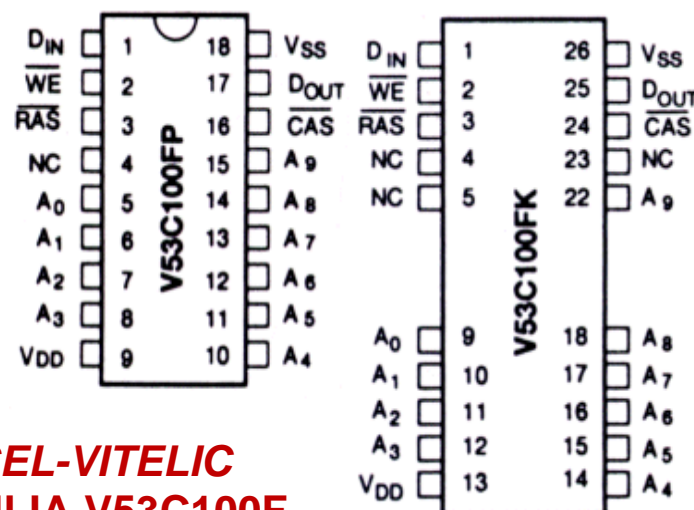
FPM 1 M bit (1 M palabras \times 1 bit)

Una dirección (20 bits) se divide (multiplexada en el tiempo) en 10 bits para la fila y 10 para la columna.

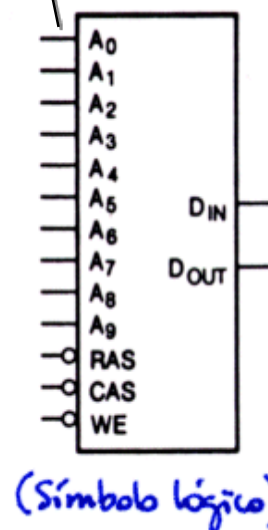
Tiempo de acceso: 60 ns

Tiempo de ciclo: 120 ns

T. de ciclo en modo página: 40 ns



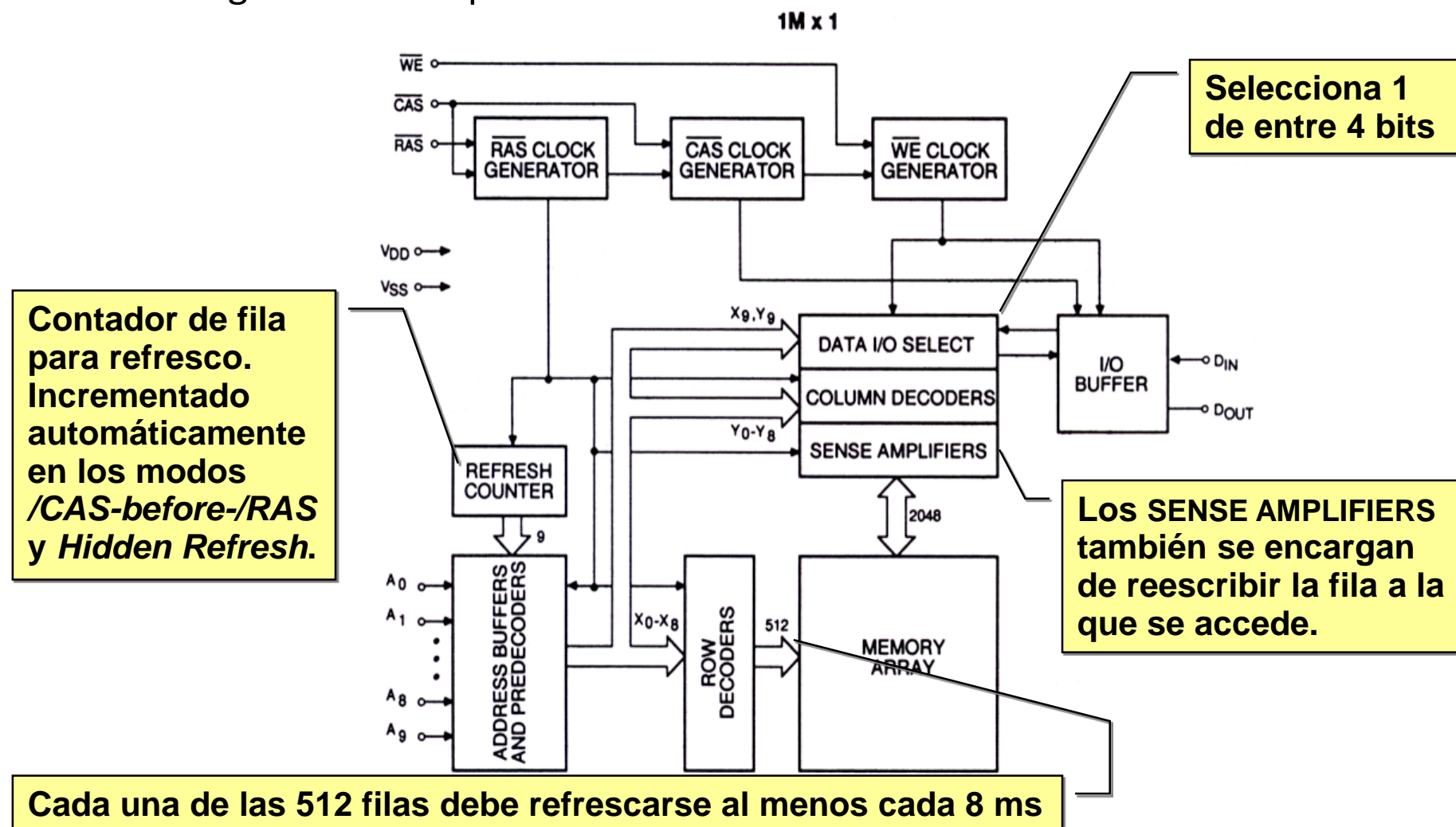
MOSEL-VITELIC
FAMILIA V53C100F
DRAM CMOS
FAST PAGE MODE
1M \times 1 BIT



A ₀ -A ₉	Address Inputs
$\overline{\text{RAS}}$	Row Address Strobe
$\overline{\text{CAS}}$	Column Address Strobe
$\overline{\text{WE}}$	Write Enable
D _{IN}	Data Input
D _{OUT}	Data Output
V _{DD}	+5V Supply
V _{SS}	0V Supply
NC	No Connect

Ejemplo de DRAM

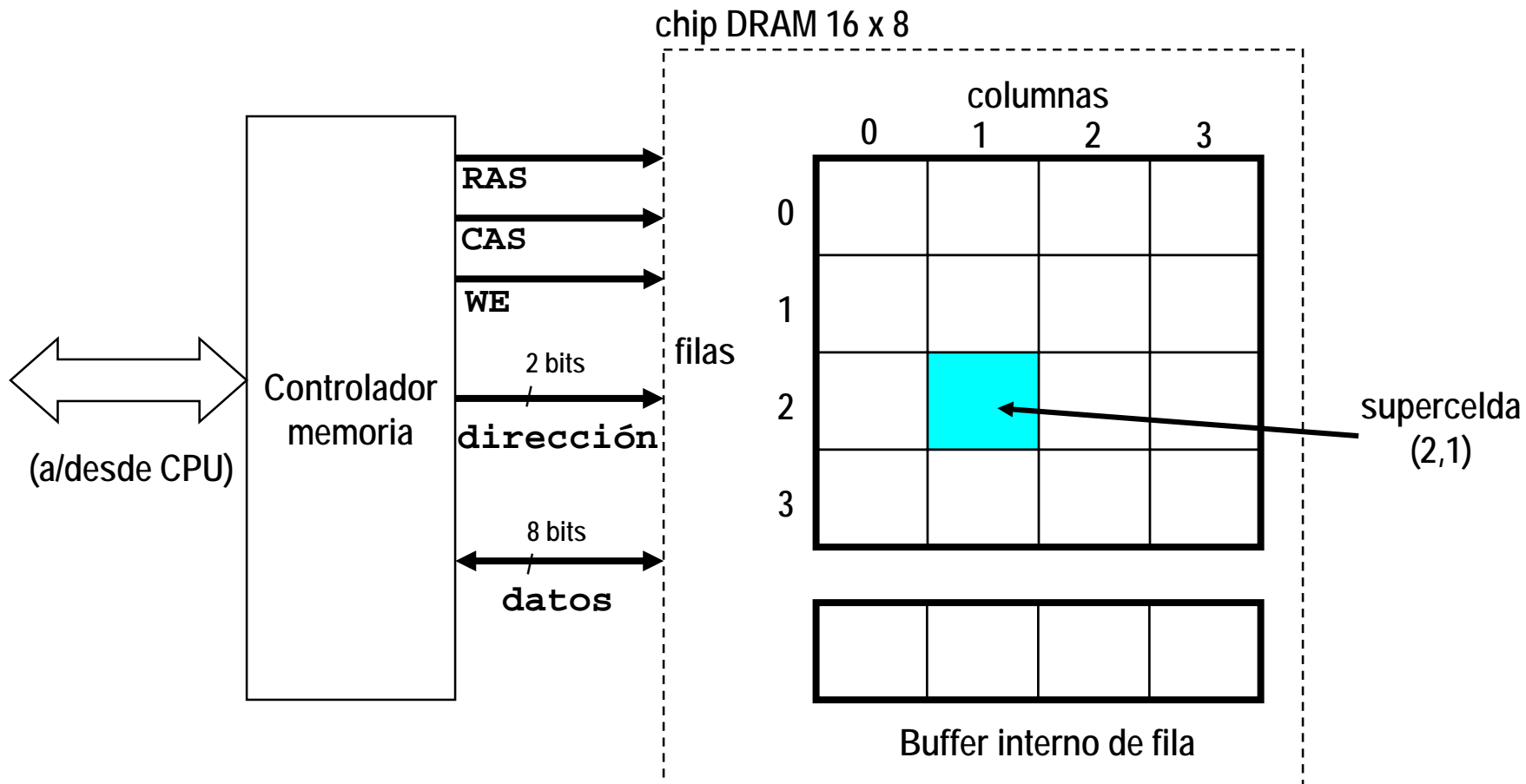
- Diagrama de bloques funcionales:



Organización DRAM convencional

■ DRAM $d \times w$:

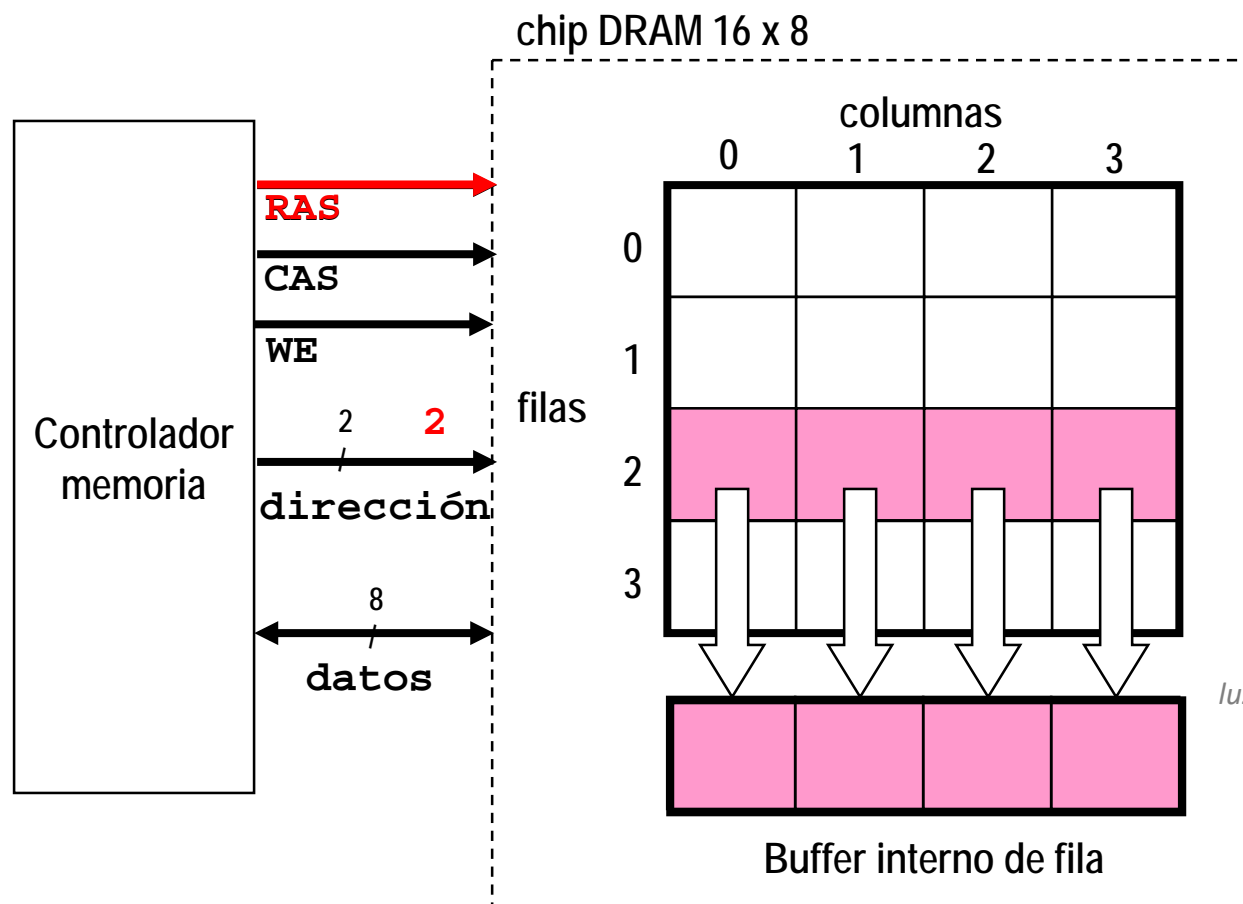
- $d \cdot w$ bits total organizados como d **superceldas** de tamaño w bits



Lectura de Supercelda DRAM (2,1)

Paso 1(a): **RAS** (Row address strobe[†], comando *Activate*) selecciona fila 2

Paso 1(b): Fila 2 copiada de array DRAM a buffer de fila



[†] strobe = luz q. parpadea rápidamente traducir como "señal dirección fila/columna" desde SDRAM, 3/4 líneas para formar un comando RAS, CAS, WE, A10

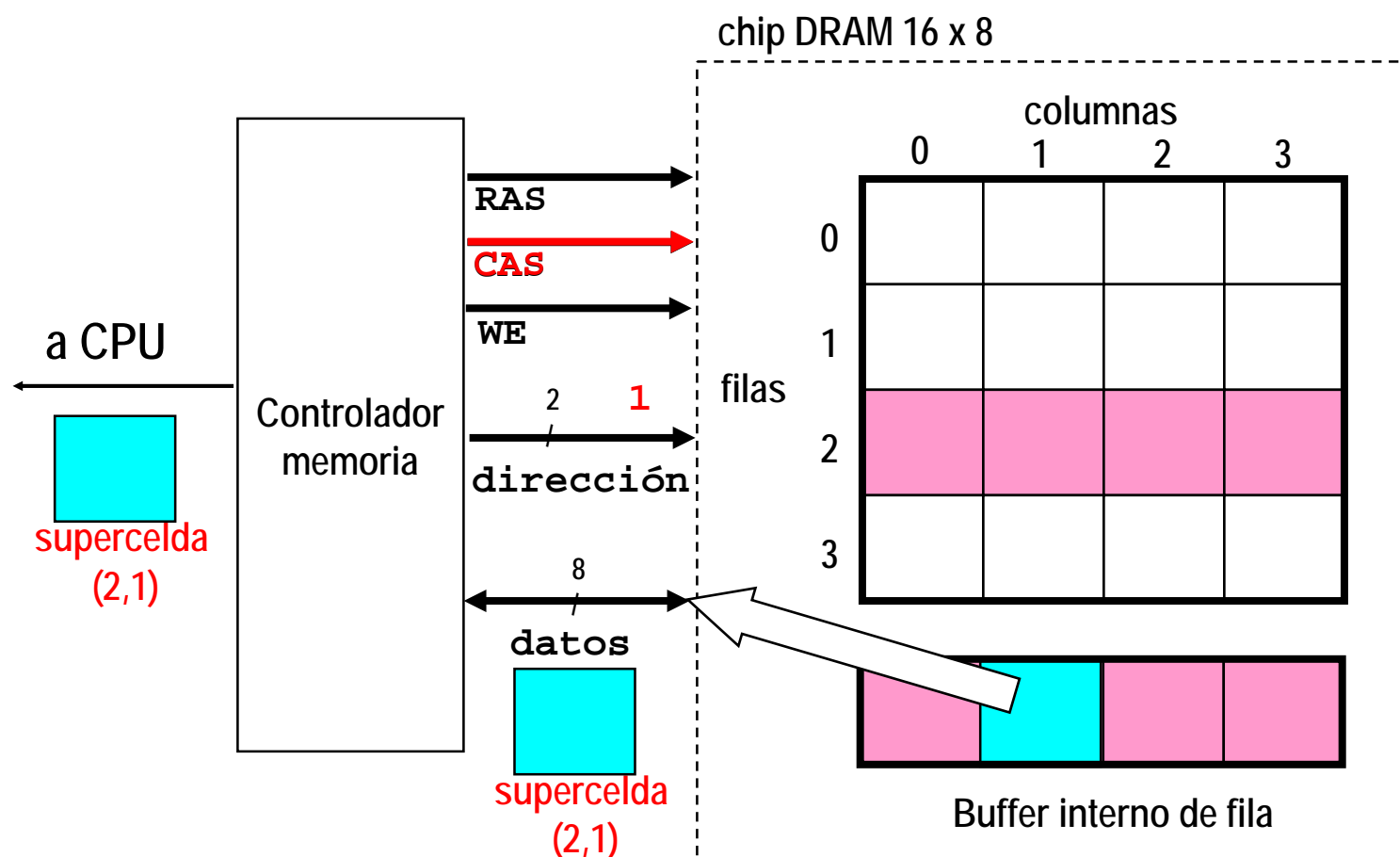
Ver https://en.wikipedia.org/wiki/Synchronous_dynamic_random-access_memory#Commands

Lectura de Supercelda DRAM (2,1)

Paso 2(a): **CAS** (Column address strobe, comando *Read*) selecciona columna 1

Paso 2(b): Supercelda (2,1) copiada de buffer a líneas bus datos → CPU.

Paso 3: Buffer copiado de vuelta a la fila para refrescar datos



Resumen SRAM vs DRAM

	Trans. por bit	tiempo acceso	necesita refresco?	necesita EDC [‡] ?	Coste	Aplicaciones
SRAM	6 ú 8	1x	No	Quizás	100x	memoria cache
DRAM	1	10x	Sí	Sí	1x	memoria principal, frame buffers [†]

EDC[‡]: Error detection and correction

■ Tendencias

- La SRAM escala con la tecnología de semiconductores
 - está llegando a sus límites
- Escalado DRAM limitado por mínima capacidad necesaria C (condensador)
 - razón de aspecto limita cómo de profundo se puede hacer el C
 - también llegando a su límite

[‡] Detección y corrección de errores

[†] buffer de fotogramas (vídeo) 33

DRAMs mejoradas

- **Funcionamiento celda DRAM no ha cambiado desde invención**
 - Comercializada por Intel en 1970
- **Núcleos DRAM con mejor lógica de interfaz y E/S más rápida:**
 - DRAM síncrona (**SDRAM**)
 - Usa una señal de reloj convencional en lugar de control asíncrono
 - puede aceptar un comando y transferir una palabra en cada ciclo reloj
 - puede hacer pipeline accediendo concurrentemente a distintos bancos
 - Reinterpreta señales RAS/CAS/WE/A10 como comando (Activate, Read...)[‡]
 - Añade 1..3 bits selección banco (BA_{0-2}) (2,4,8 “Memory Arrays”)[‡]
 - DRAM síncrona a doble velocidad datos (double data-rate, **DDR SDRAM**)
 - temporización a doble flanco envía 2 bits por ciclo por pin
 - diferentes tipos según el tamaño de un pequeño buffer precaptación:
 - **DDR** (2 bits), **DDR2** (4 bits), **DDR3** (8 bits), **DDR4** (8⁺ bits)
 - DDR4 cambia formato comandos (ACT, RAS/CAS/WE= A_{16-14} , BC/AP= $A_{12,10}$)
 - para 2010, estándar para mayoría sistemas sobremesa y servidor
 - Intel Core i7 admite SDRAM DDR3 y/o DDR4 (según modelo)

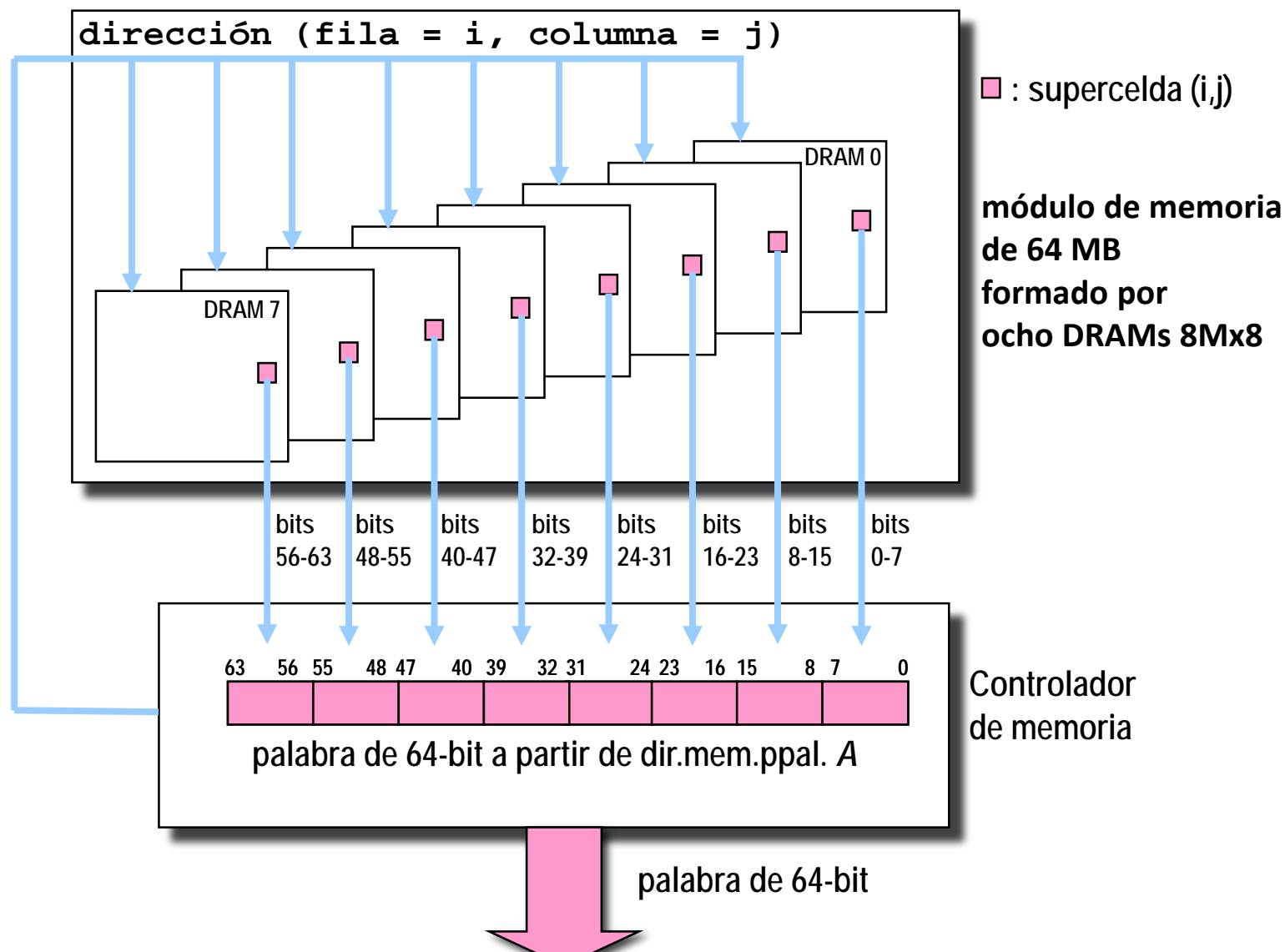
[‡] Ver https://en.wikipedia.org/wiki/Synchronous_dynamic_random-access_memory#Command_signals

Los entusiastas pueden consultar la tesis enlazada en “See also”: http://drum.lib.umd.edu/bitstream/1903/11269/1/Gross_umd_0117N_11844.pdf

https://en.wikipedia.org/wiki/DDR4_SDRAM#Command_encoding [†] formato de los comandos

.../DDR4_SDRAM#Features [†] 8n / bank group interleave **34**.

Módulos de Memoria



Memoria I: Jerarquía de memoria

- La abstracción de memoria (concepto de Lectura y Escritura)
- RAM: bloque constructivo de memoria principal
- **Configuración y diseño de memorias utilizando varios chips**
- Localidad de las referencias
- Jerarquía de memoria
- Tecnologías de almacenamiento, y tendencias

Ampliación de memoria

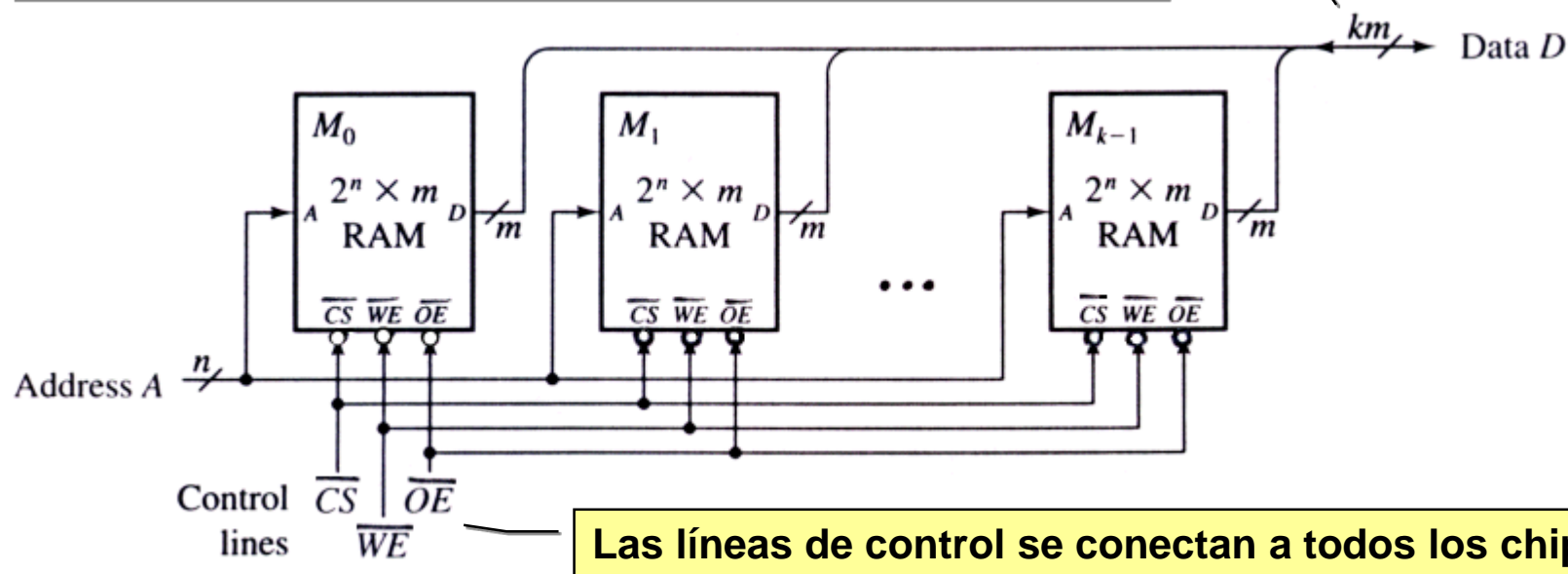
■ Problema:

- Construir una memoria de 2^N palabras de M bits a partir de chips de 2^n palabras de m bits.

① Incrementar el ancho de palabra de m a $k \cdot m = M$

- Necesitamos k circuitos de tamaño de palabra m :

Los k chips se conectan de forma “bit-slice”. Cada chip contribuye a m bits de la palabra de datos de $k \cdot m$ bits



Ampliación de memoria

② Incrementar el número de palabras de

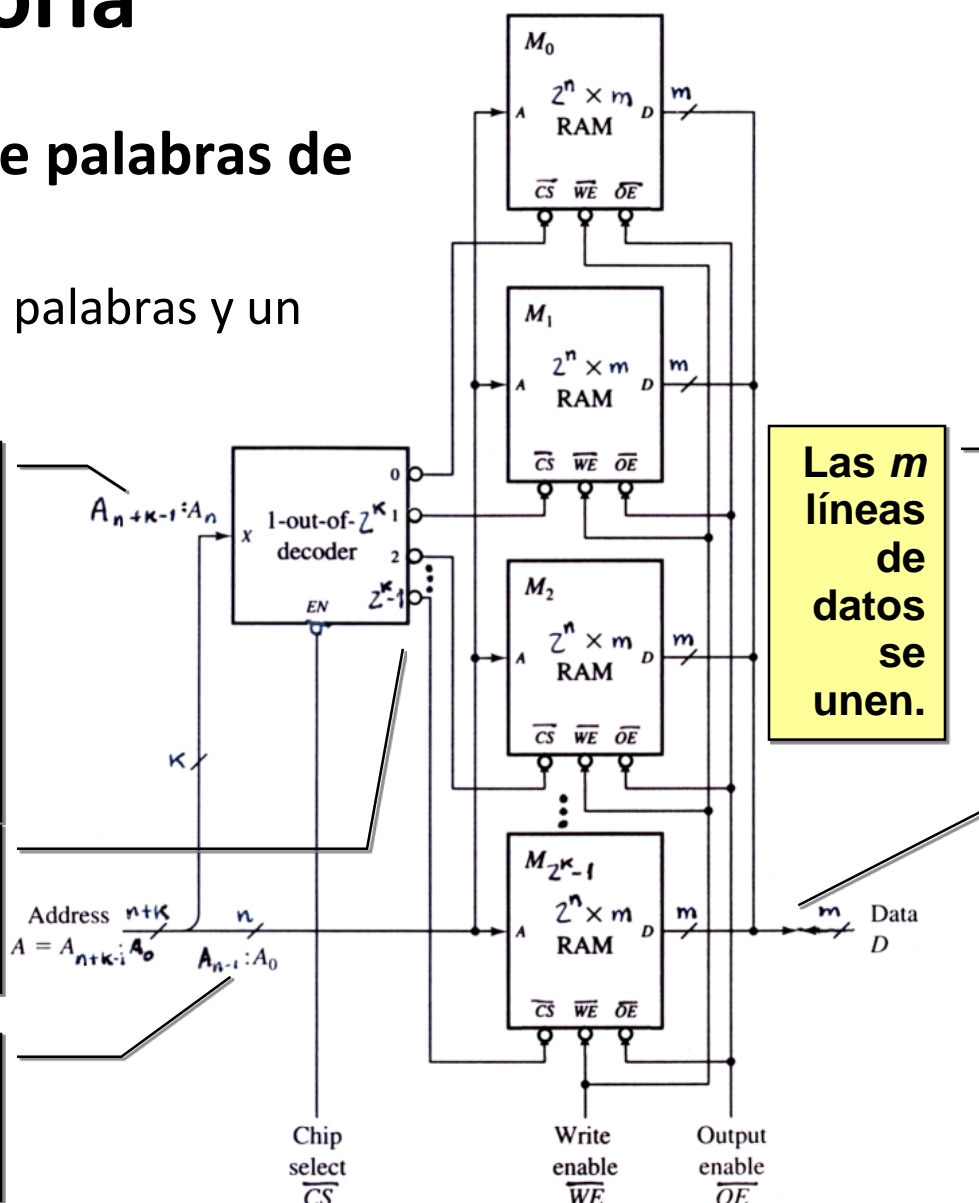
2^n a $2^{n+k}=2^N$

- Necesitamos 2^k circuitos de 2^n palabras y un decodificador de 1 entre 2^k .

Las k líneas de dirección de orden superior ($A_{n+k-1}:A_n$) se conectan a las entradas de un decodificador de k a 2^k
 \Rightarrow cada configuración de los $n+k$ bits hace que se seleccione solamente el circuito RAM indicado por los bits de dirección $A_{n+k-1}:A_n$.

Las 2^k líneas de salida del decodificador se conectan a las entradas de selección /CS de los 2^k circuitos.

Las n líneas de dirección de orden inferior se conectan en común a las líneas de dirección de los 2^k chips.



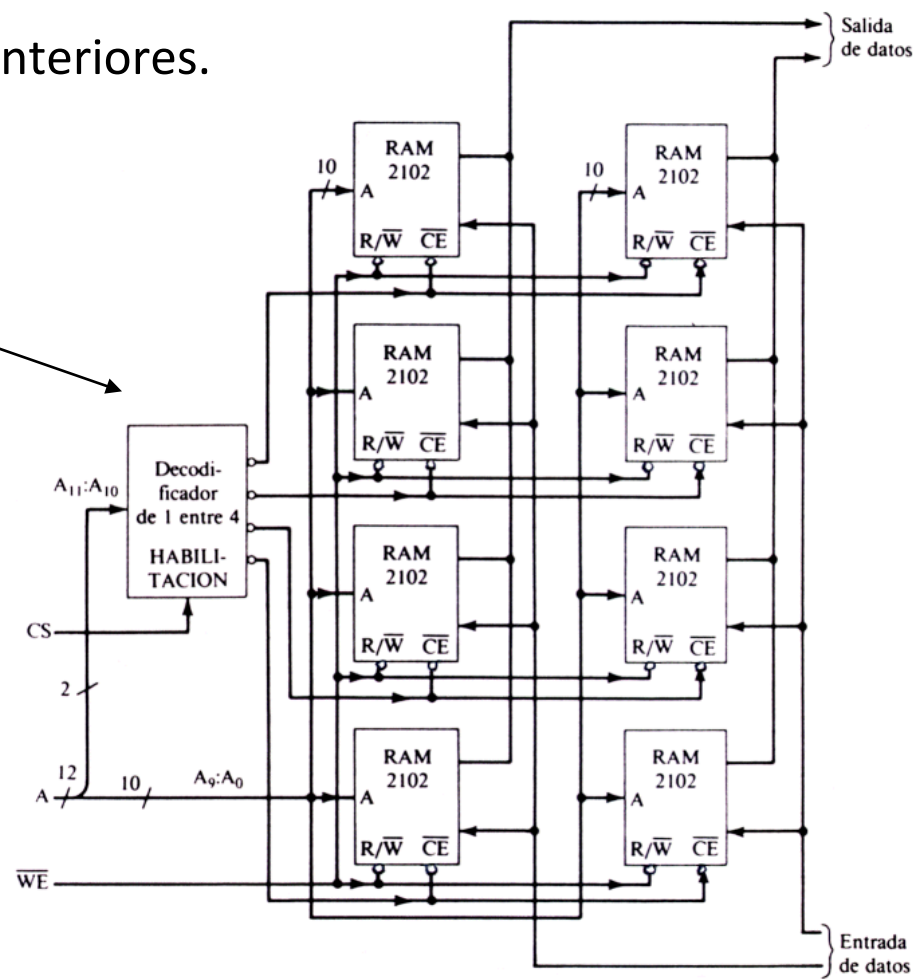
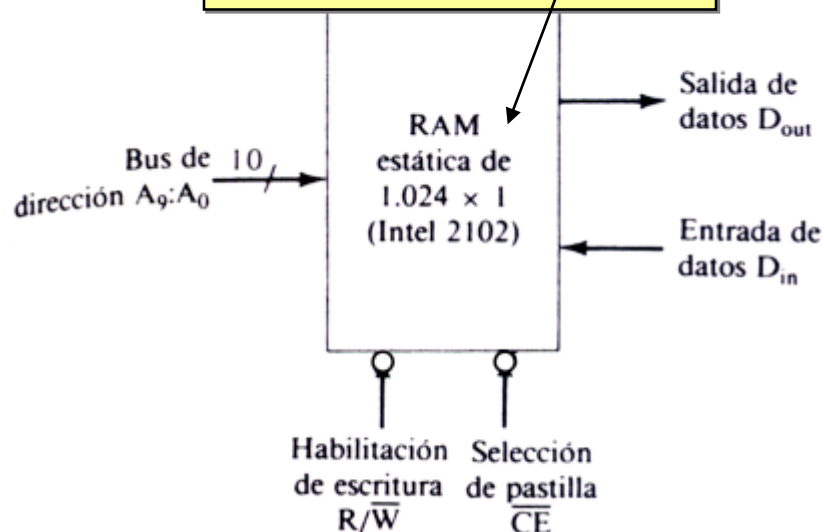
Ampliación de memoria

③ Incrementar simultáneamente el número de palabras y el ancho de palabra.

- Basta combinar las dos técnicas anteriores.

Ejemplo 1:

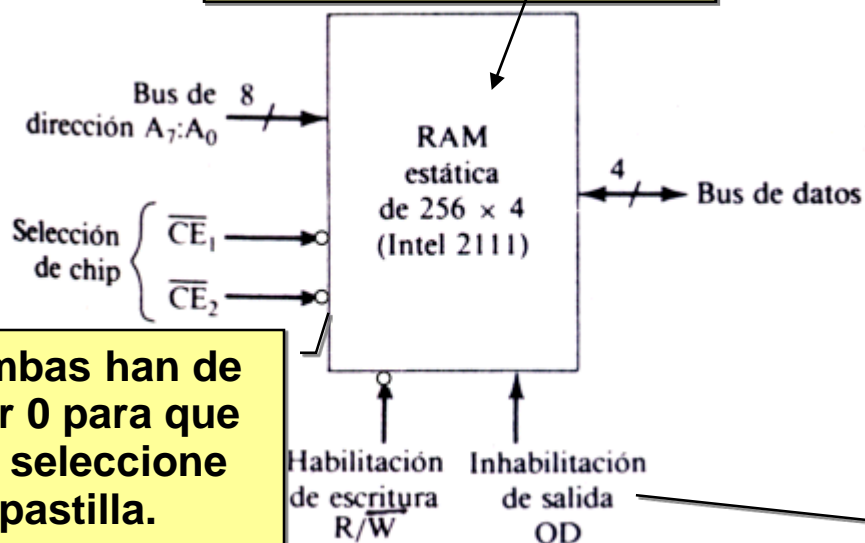
RAM de 4096×2
construida con 8
RAM de 1024×1 .



Ampliación de memoria

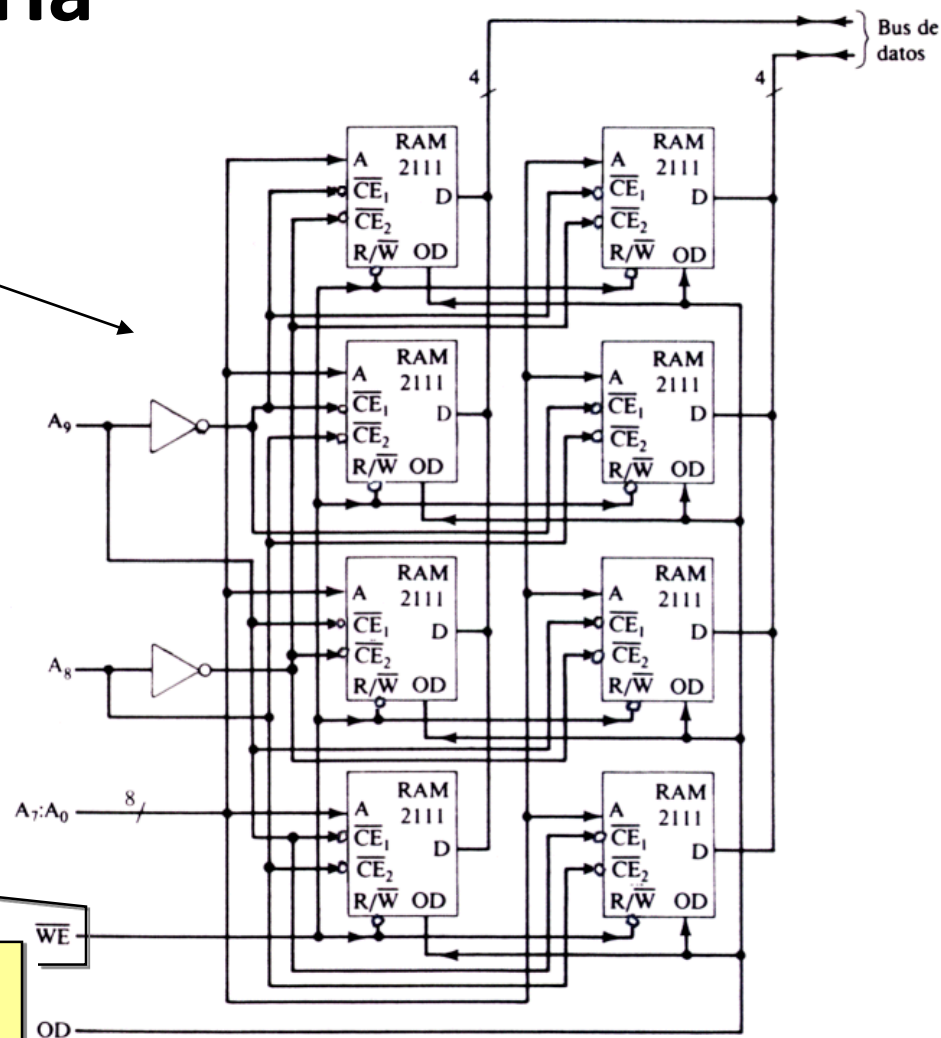
Ejemplo 2:

RAM de $1\text{ K} \times 8$
construida con 8
RAM de 256×4 .



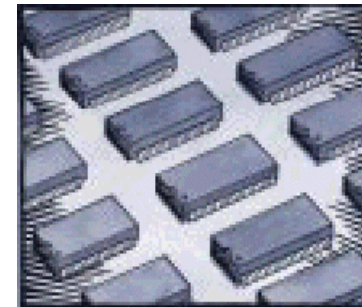
Ambas han de ser 0 para que se seleccione la pastilla.

Como hay un solo bus de datos \Rightarrow la línea OD (*Output Disable*) se activa durante los ciclos de escritura para evitar que se lean datos internos desde el chip al bus mientras el procesador usa el bus para salida datos.



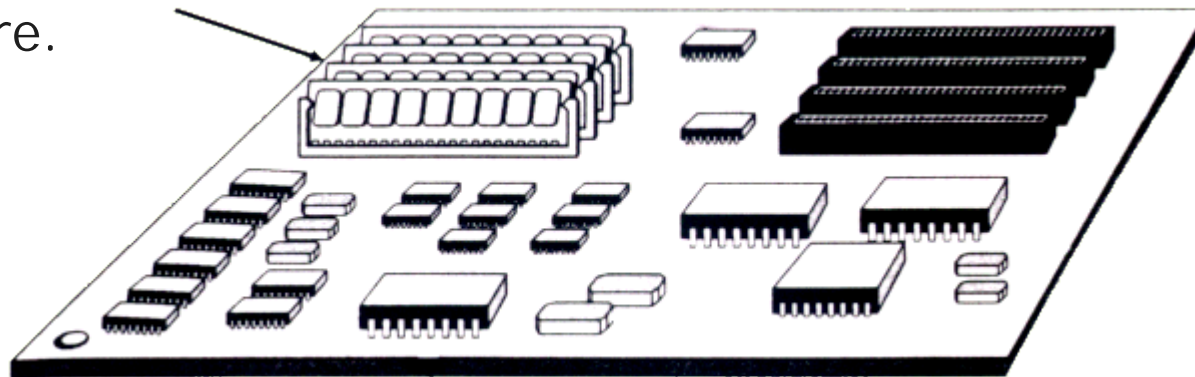
Módulos de memoria en línea

- En los 80, la memoria solía soldarse directamente en la placa madre del ordenador. Pero a medida que aumentaron los requisitos de memoria, esta técnica resultó poco factible.



- SIMM, DIMM, SODIMM:

- Varios chips DRAM en una pequeña placa de circuito impreso (PCB) que calza en un conector en la placa madre.



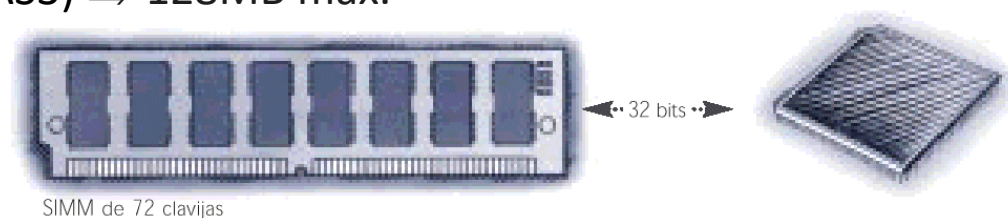
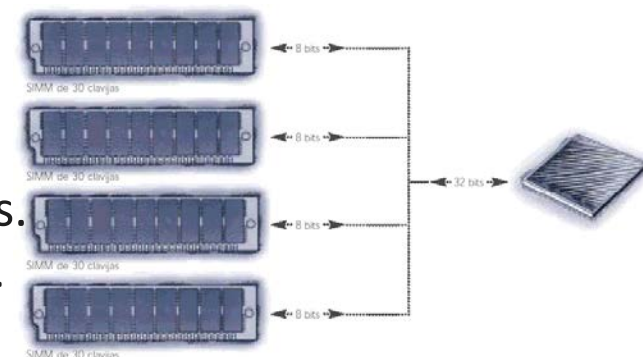
- ✓ método flexible para actualizar la memoria
- ✓ ocupa menos espacio en la placa madre.
- Normalmente sólo se ampliaba el bus de datos, no el de direcciones (no había necesidad de decodificador).

Módulos de memoria SIMM



■ **SIMM** (*Single In-line Memory Module*)

- En un SIMM, cada contacto de una cara está conectado con el alineado en la otra cara para formar un único contacto.
- 80's y 90's, FPM y EDO-RAM
- 30 contactos:
 - 8 bits de datos \Rightarrow 4 SIMM para bus 32 bits.
 - 12 pines dirección \Rightarrow 24 bits dir \Rightarrow 16MB máx.
- 72 contactos:
 - 32 bits de datos (24 bits dir) \Rightarrow 64MB máx.
 - Dos rangos[†] (con /RAS1, /RAS3) \Rightarrow 128MB máx.

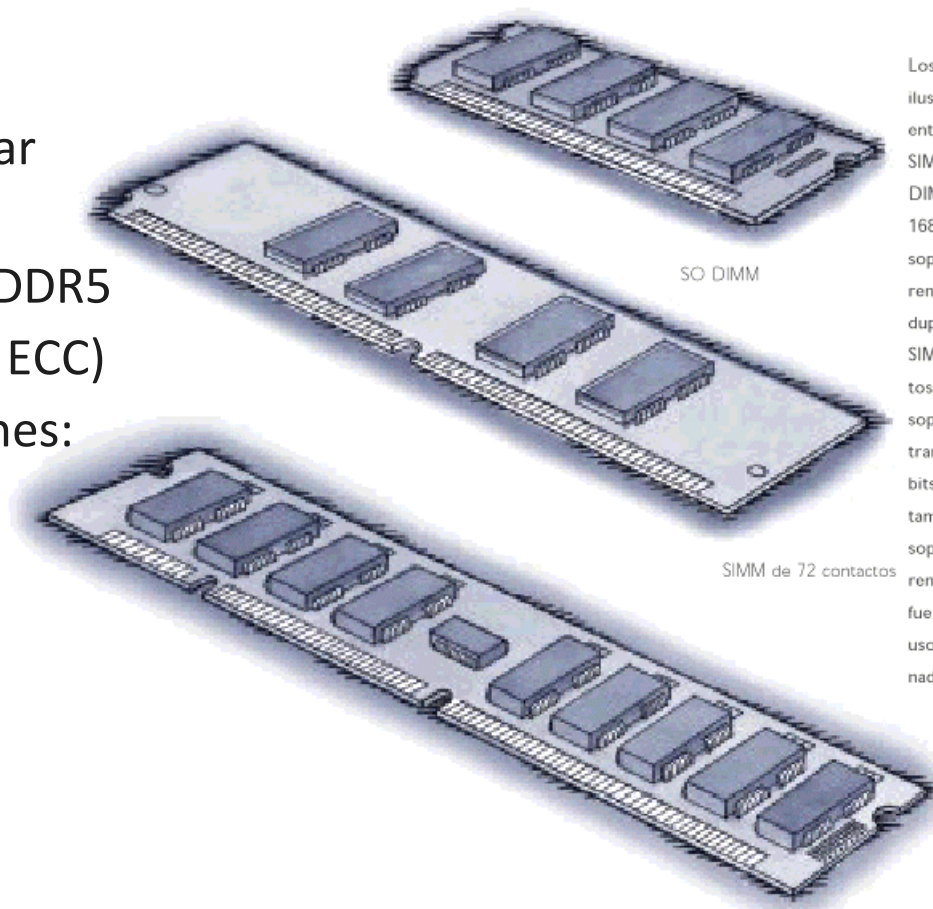


[†] Ver https://en.wikipedia.org/wiki/SIMM#72-pin_SIMMs
https://en.wikipedia.org/wiki/Memory_rank

Módulos de memoria DIMM

■ DIMM (*Dual In-line Memory Module*)

- En DIMM los contactos opuestos están aislados eléctricamente para formar **dos contactos separados**.
- 90's-hoy, SDRAM-DDR-...-DDR5
- 64 bits de datos (ó 72 con ECC)
- Incremento progresivo pines:
168-pin: FPM, EDO y SDRAM
184-pin: DDR
240-pin: DDR2, DDR3
288-pin: DDR4, DDR5



Los tres ejemplos ilustran las diferencias entre los productos SIMM, DIMM y SO DIMM. El DIMM de 168 contactos brinda soporte para transferencias de 64 bits, sin duplicar el tamaño del SIMM de 72 contactos, el cual brinda soporte sólo para transferencias de 32 bits. El SO DIMM también brinda soporte para transferencias de 32 bits y fue diseñado para su uso en los ordenadores portátiles.

DIMM de 168 contactos

Módulos de memoria SODIMM



■ SODIMM (*Small Outline DIMM*)

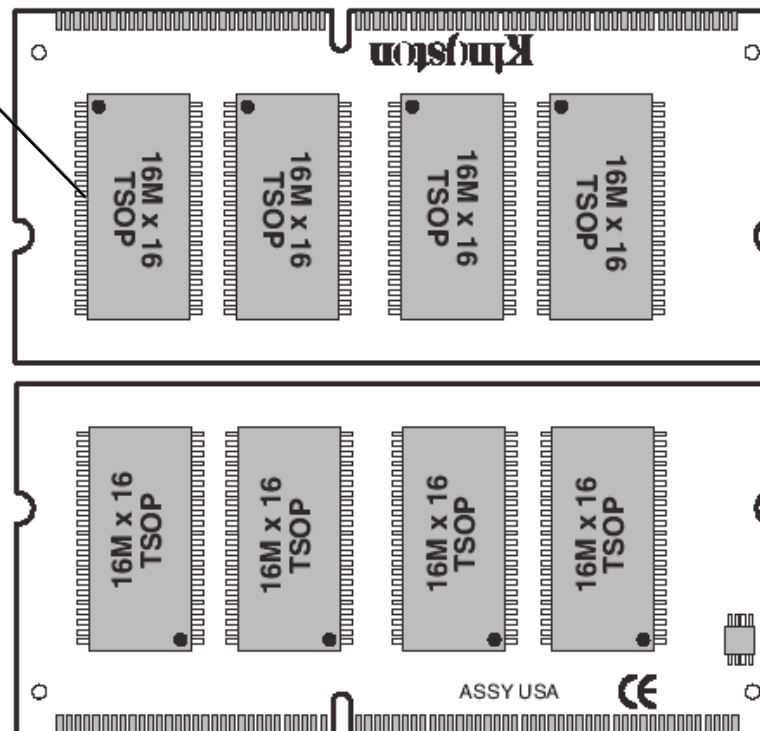
- Menos tamaño que un DIMM, menos contactos.
- Uso en portátiles. (100/144-pin SDR, 200-pin DDR-DDR2, 204-pin DDR3, 260-pin DDR4)
- Ejemplo: SODIMM SDR (144-pin) de 256 MB ($2 \times 16\text{M} \times 64$) a 133 MHz:

**8 chips (4 en cada cara)
SDRAM a 133 MHz de
 $16\text{M} \times 16$**

**Cada chip tiene 4
bancos de $4\text{M} \times 16$**

**Bus de datos: 64 bits =
 $4 \text{ chips} \times 16 \text{ bits/chip}$**

**El módulo tiene 2 ranks
(¡en este caso sí se
amplía el número de
direcciones!)**



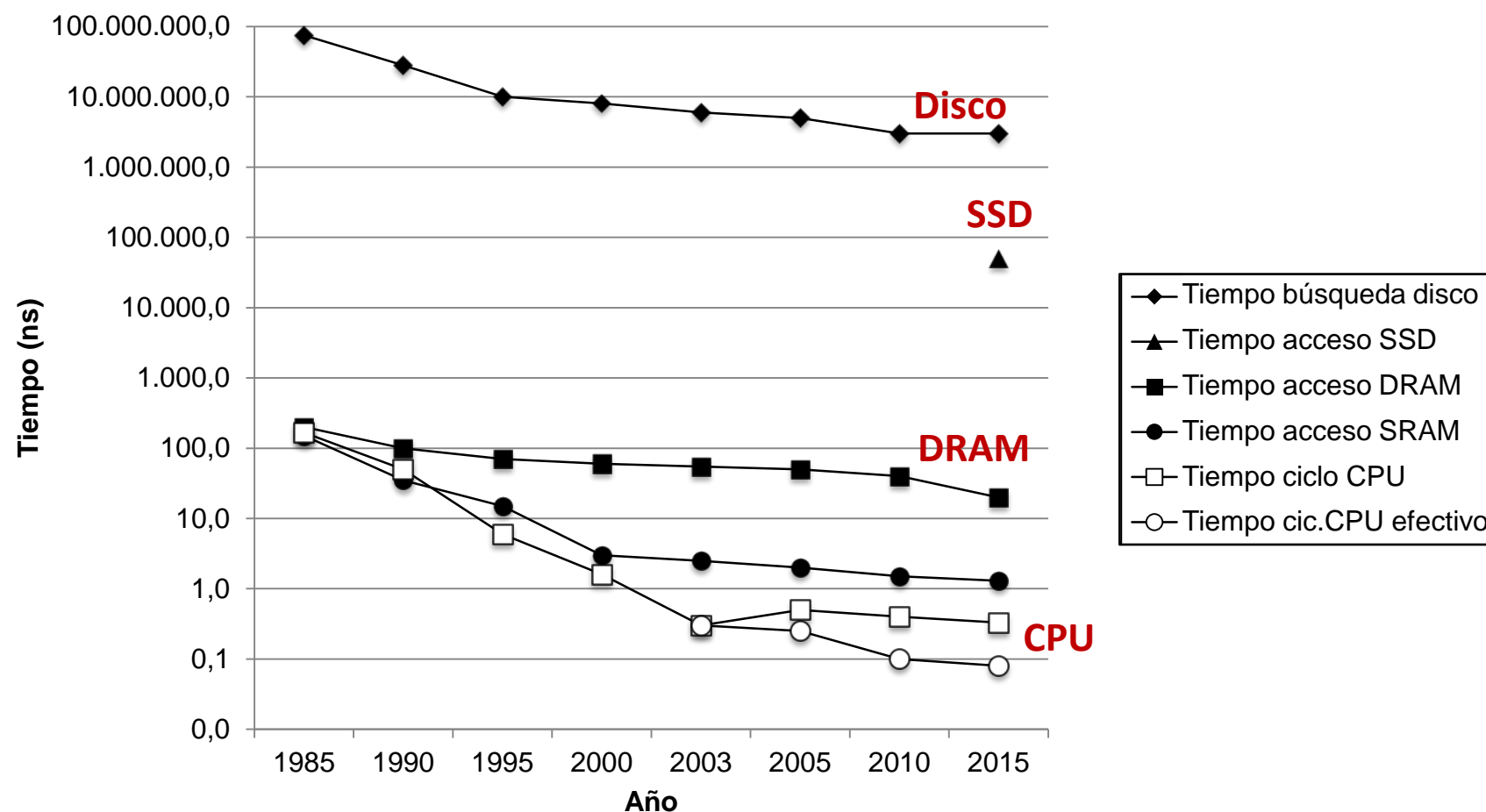
† Ver <https://en.wikipedia.org/wiki/SO-DIMM>
<https://en.wikipedia.org/wiki/DIMM#Ranking>

Memoria I: Jerarquía de memoria

- La abstracción de memoria (concepto de Lectura y Escritura)
- RAM: bloque constructivo de memoria principal
- Configuración y diseño de memorias utilizando varios chips
- **Localidad de las referencias**
- Jerarquía de memoria
- Tecnologías de almacenamiento, y tendencias

La brecha[†] CPU-Memoria

La brecha entre velocidades de disco, DRAM y CPU *se ensancha*.



¡Localidad al rescate!

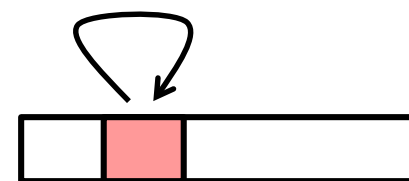
La clave para salvar esta brecha CPU-Memoria es una propiedad fundamental de los programas informáticos conocida como **localidad**

Localidad

- **Principio de localidad:** Los programas tienden a usar datos e instrucciones con direcciones iguales o cercanas a las que han usado recientemente

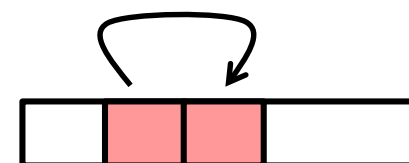
- **Localidad temporal:**

- Elementos referenciados recientemente probablemente serán referenciados de nuevo en un futuro próximo



- **Localidad espacial:**

- Elementos con direcciones cercanas tienden a ser referenciados muy juntos en el tiempo



Expresión matemática de localidad

- si en instante tiempo t se accede al dato/posición mem. $d(t)$...

■ Temporal

- $d(t + n) = d(t)$ con n pequeño

■ Espacial

- $d(t + n) = d(t) + k$ con n, k pequeños

Ejemplo de Localidad

```
sum = 0;  
for (i = 0; i < n; i++)  
    sum += a[i];  
return sum;
```

■ Referencias a datos

- Referenciar elementos array en sucesión (patrón de referencias de paso-1[†])
- Referenciar variable `sum` cada iteración

■ Referencias a instrucciones

- Referenciar instrucciones en secuencia
- Iterar bucle repetidamente

Localidad
Espacial o Temporal?

espacial

temporal

espacial

temporal

[†] *stride-1 reference pattern*

stride = paso, zancada 50

Estimaciones cualitativas de localidad

- **Afirmación:** Ser capaz de mirar un código y sacar una idea cualitativa de su localidad es una habilidad clave para un programador profesional
- **Pregunta:** ¿Tiene esta función buena localidad respecto al array a?

Pista: alm. por filas
(row-major order)

Respuesta: sí

```
int sum_array_rows(int a[M][N])
{
    int i, j, sum = 0;

    for (i = 0; i < M; i++)
        for (j = 0; j < N; j++)
            sum += a[i][j];

    return sum;
}
```

a		a	a		a		a		a
[0]	• • •	[0]	[1]	• • •	[1]	• • •	[M-1]	• • •	[M-1]
[0]		[N-1]	[0]		[N-1]		[0]		[N-1]

Ejemplo de localidad

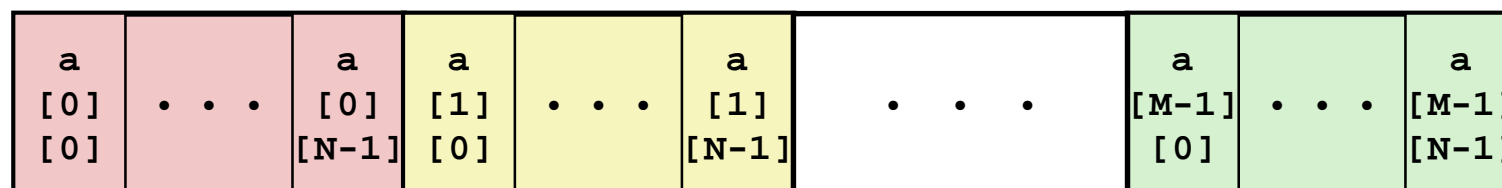
- **Pregunta:** ¿Tiene esta función buena localidad respecto al array a?

```
int sum_array_cols(int a[M][N])
{
    int i, j, sum = 0;

    for (j = 0; j < N; j++)
        for (i = 0; i < M; i++)
            sum += a[i][j];
    return sum;
}
```

Respuesta: no, salvo si...

**N es muy pequeño
...o M muy muy pequeño
y N no muy grande**



Ejemplo de Localidad

- **Pregunta:** Se pueden permutar los bucles de forma que la función recorra el array 3-d a con patrón de referencias de paso-1 (y tenga por tanto buena localidad espacial)?

```
int sum_array_3d(int a[M][N][N])
{
    int i, j, k, sum = 0;

    for (i = 0; i < N; i++)
        for (j = 0; j < N; j++)
            for (k = 0; k < M; k++)
                sum += a[k][i][j];
    return sum;
}
```

Respuesta: sí: for k,i,j, bucles en mismo orden índices
(segunda opción, peor) **for i,k,j**, haría NxM bucles (de N elems, paso-1)

Memoria I: Jerarquía de memoria

- La abstracción de memoria (concepto de Lectura y Escritura)
- RAM: bloque constructivo de memoria principal
- Configuración y diseño de memorias utilizando varios chips
- Localidad de las referencias
- **Jerarquía de memoria**
- Tecnologías de almacenamiento, y tendencias

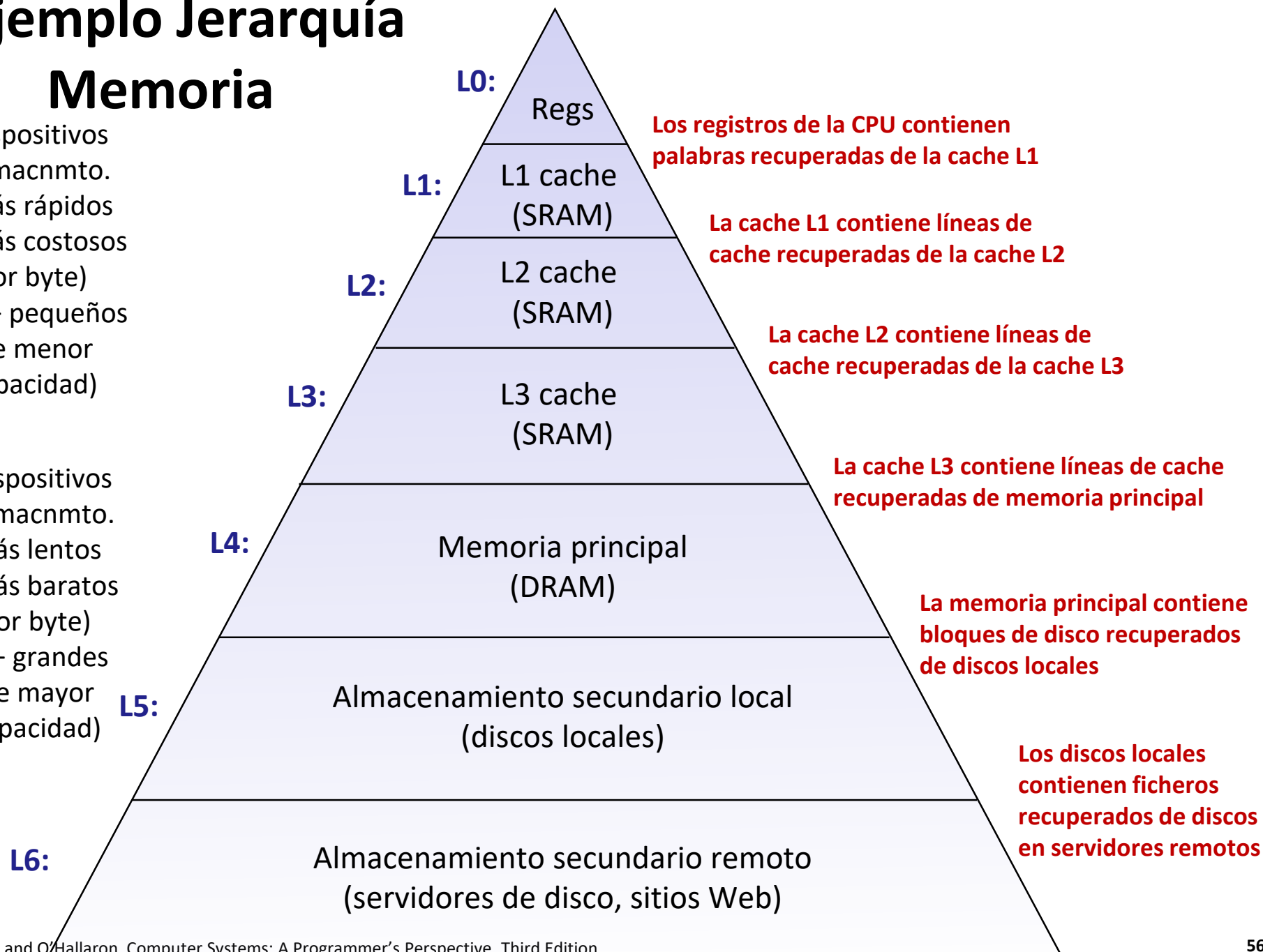
Jerarquía de Memoria

- **Algunas propiedades fundamentales y perdurables del hardware y software:**
 - Las tecnologías de almacenamiento rápidas cuestan más por byte, tienen menor capacidad y requieren más potencia (¡calor!)
 - La brecha velocidad CPU-memoria principal se está ampliando
 - Los programas bien escritos tienden a exhibir buena localidad
- **Estas propiedades fundamentales se complementan muy convenientemente**
- **Sugieren un enfoque para organizar sistemas de memoria y almacenamiento conocido como **jerarquía de memoria****

Ejemplo Jerarquía Memoria

↑
dispositivos
almacnmto.
más rápidos
más costosos
(por byte)
y + pequeños
(de menor
capacidad)

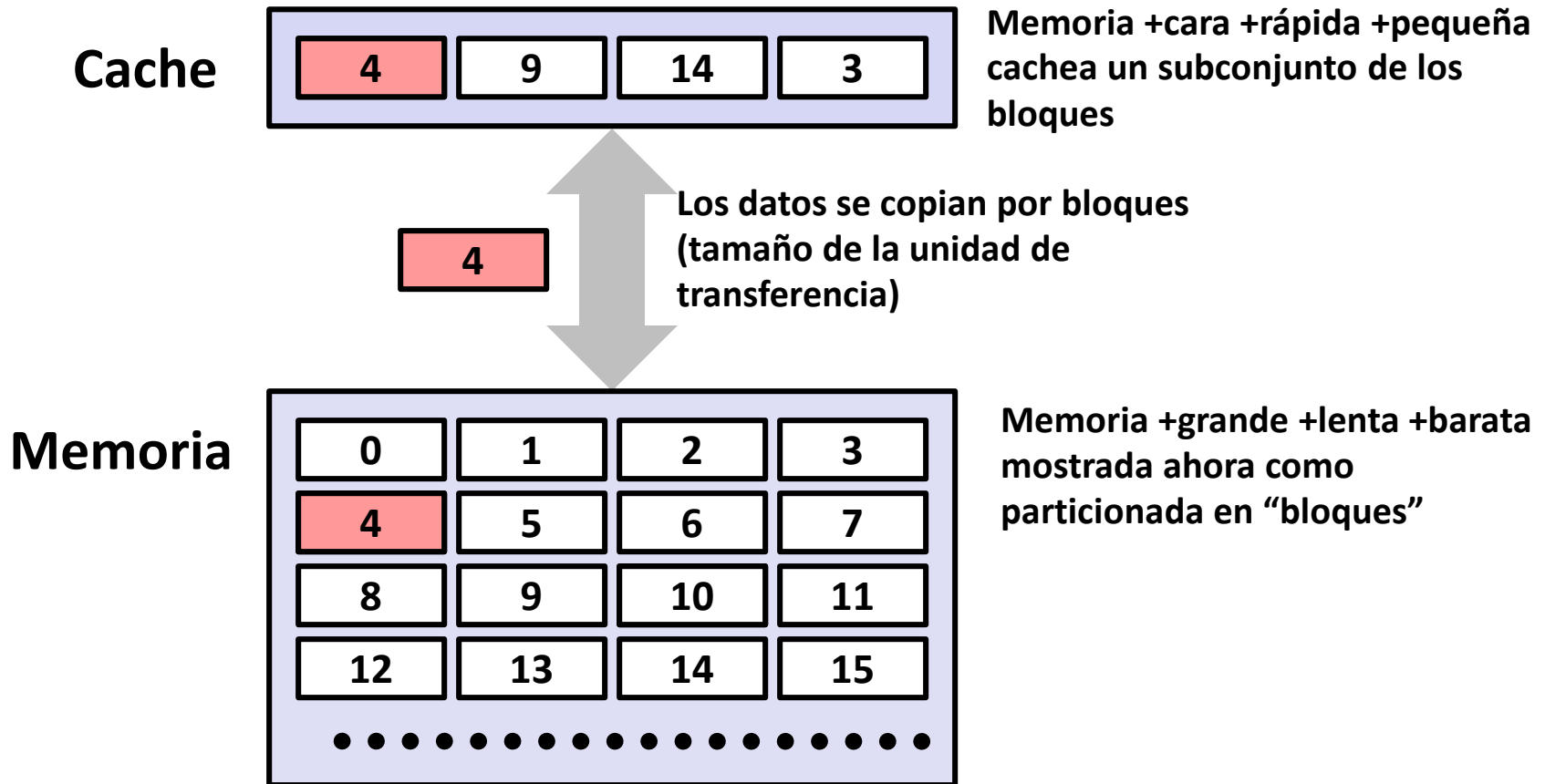
↓
dispositivos
almacnmto.
más lentos
más baratos
(por byte)
y + grandes
(de mayor
capacidad)



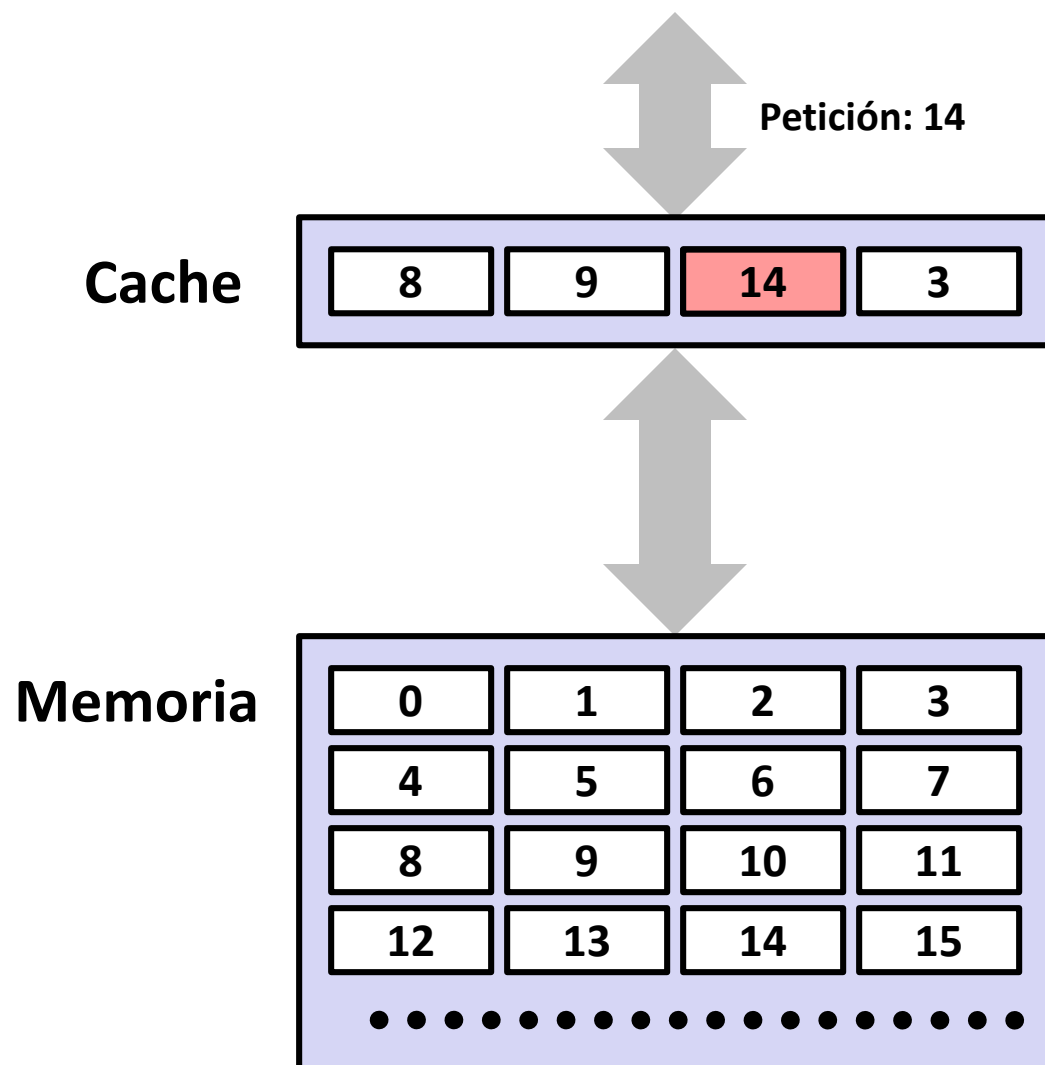
Caches

- **Cache:** Un dispositivo de almacenamiento más rápido y pequeño que funciona como zona de trabajo temporal para un subconjunto de los datos de otro dispositivo mayor y más lento
- **Idea fundamental de una jerarquía de memoria:**
 - $\forall k$, el dispositivo a nivel k (+rápido, +pequeño) sirve de cache para el dispositivo a nivel $k+1$ (+lento, +grande)
- **¿Por qué funcionan bien las jerarquías de memoria?**
 - Debido a la localidad, los programas suelen acceder a los datos a nivel k más a menudo que a los datos a nivel $k+1$
 - Así, el almacenamiento a nivel $k+1$ puede ser más lento, y por tanto más barato (por bit) y más grande
- **Idea Brillante (ideal):** La jerarquía de memoria conforma un gran conjunto de almacenamiento que cuesta como el más barato pero que proporciona datos a los programas a la velocidad del más rápido

Conceptos Generales de Cache



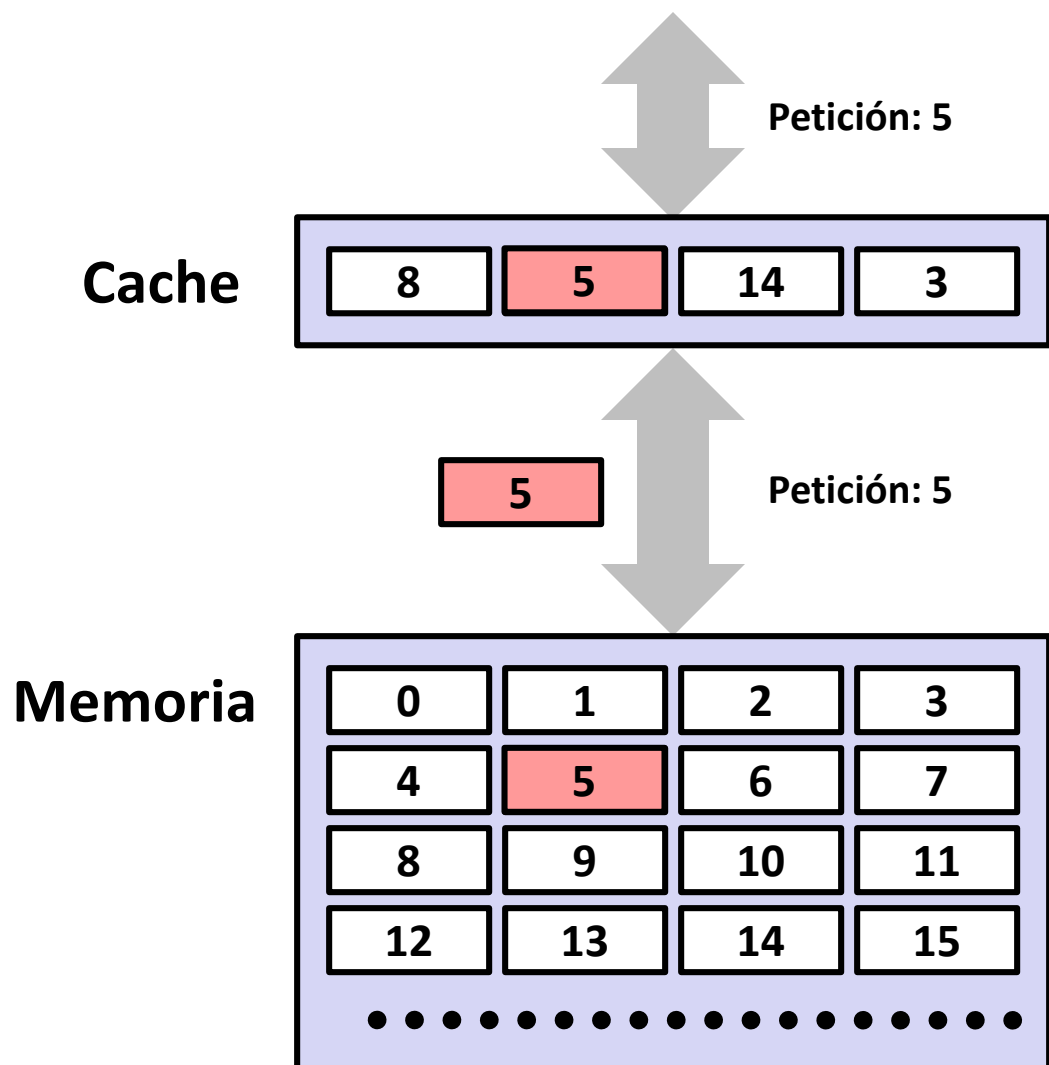
Conceptos Generales de Cache: Acierto[†]



*Se necesitan datos
del bloque b*

*El bloque b está en cache:
¡Acierto!*

Conceptos Generales de Cache: Fallo[†]



*Se necesitan datos
del bloque b*

*El bloque b no está en cache:
¡Fallo!*

*El bloque b se capta de
memoria*

*El bloque b se almacena
en cache*

- **Política de colocación[†]:**
determina dónde va b
- **Política de reemplazo[†]:**
determina qué bloque es
desalojado[†] (víctima)

[†] cache miss
[re]placement policy,
evicted/victim block

Conceptos Generales de Cache:

3 Tipos de Fallo de Cache

■ Fallos en frío (obligados)

- Los fallos en frío ocurren porque la cache empieza vacía y esta es la primera referencia al bloque

■ Fallos por capacidad

- Ocurren cuando el conjunto de bloques activos (**conjunto de trabajo**) es más grande que la cache

■ Fallos por conflicto

- Mayoría caches limitan que los bloques a nivel $k+1$ puedan ir a pequeño subconjunto (a veces unitario) de las posiciones de bloque a nivel k
 - P.ej. Bloque i a nivel $k+1$ debe ir a bloque $(i \bmod 4)$ a nivel k (corr. directa)
- Fallos por conflicto ocurren cuando cache nivel k suficientemente grande pero a varios datos les corresponde ir al mismo bloque a nivel k
 - P.ej. Referenciar bloques 0, 8, 0, 8, 0, 8, ... fallaría continuamente (ejemplo anterior con correspondencia directa)

Ejemplos de cacheado en Jerarquía Memoria

Tipo de Cache	Qué se cachea?	Dónde se cachea?	Latencia (ciclos)	Gestionado por
Registros	Palabras de 4-8 B	CPU (on-chip)	0	Compilador
TLB [†]	Trad. de direcciones	TLB (on-chip)	0	MMU [†] (hardw)
cache L1	Bloques de 64 bytes	L1 (on-chip)	4	Hardware
cache L2	Bloques de 64 B	L2 (on-chip)	10	Hardware
cache L3	Bloques de 64 B	L3 (on-chip)	50	Hardware
Memoria Virtual	Páginas de 4 KB	Memoria principal	200	Hardware + SO
Buffer de disco	Partes de ficheros	Memoria principal	200	SO
cache de disco	Sectores de disco	Controladora de disco	100,000	Firmware disco
Buffer disco red	Partes de ficheros	Disco local	10,000,000	Cliente NFS [†]
cache Navegador	Páginas Web	Disco local	10,000,000	Navegador web
cache Web	Páginas Web	Discos de servidores remotos	1,000,000,000	Servidor web proxy [†]

[†] Translation Lookaside Buffer, Memory Management Unit,

Network File System, proxy=procurador, apoderado (intermediario)

¡Hora de juego!

Conectarse a:

<https://swad.ugr.es> > EC > Evaluación > Juegos

Memoria I: Jerarquía de memoria

- La abstracción de memoria (concepto de Lectura y Escritura)
- RAM: bloque constructivo de memoria principal
- Configuración y diseño de memorias utilizando varios chips
- Localidad de las referencias
- Jerarquía de memoria
- **Tecnologías de almacenamiento, y tendencias**

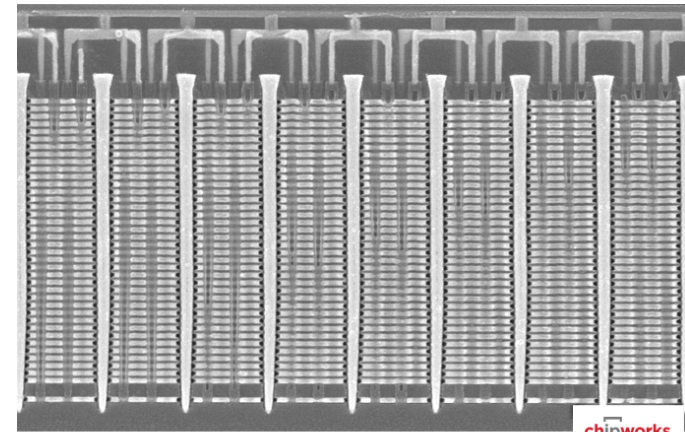
Tecnologías de almacenamiento

■ Discos Magnéticos



- Almacenamiento en medio magnético
- Acceso electromecánico

■ Memoria No-volátil (Flash)



Close-up image of V-NAND flash array

chipworks

- Almacenamiento como carga persistente
- Implementado con estructura 3-D[†]
 - +100 niveles de celdas
 - 3 bits de datos por celda

[†] Ver https://en.wikipedia.org/wiki/Flash_memory#Vertical_NAND

¿Qué hay dentro de una unidad de disco?

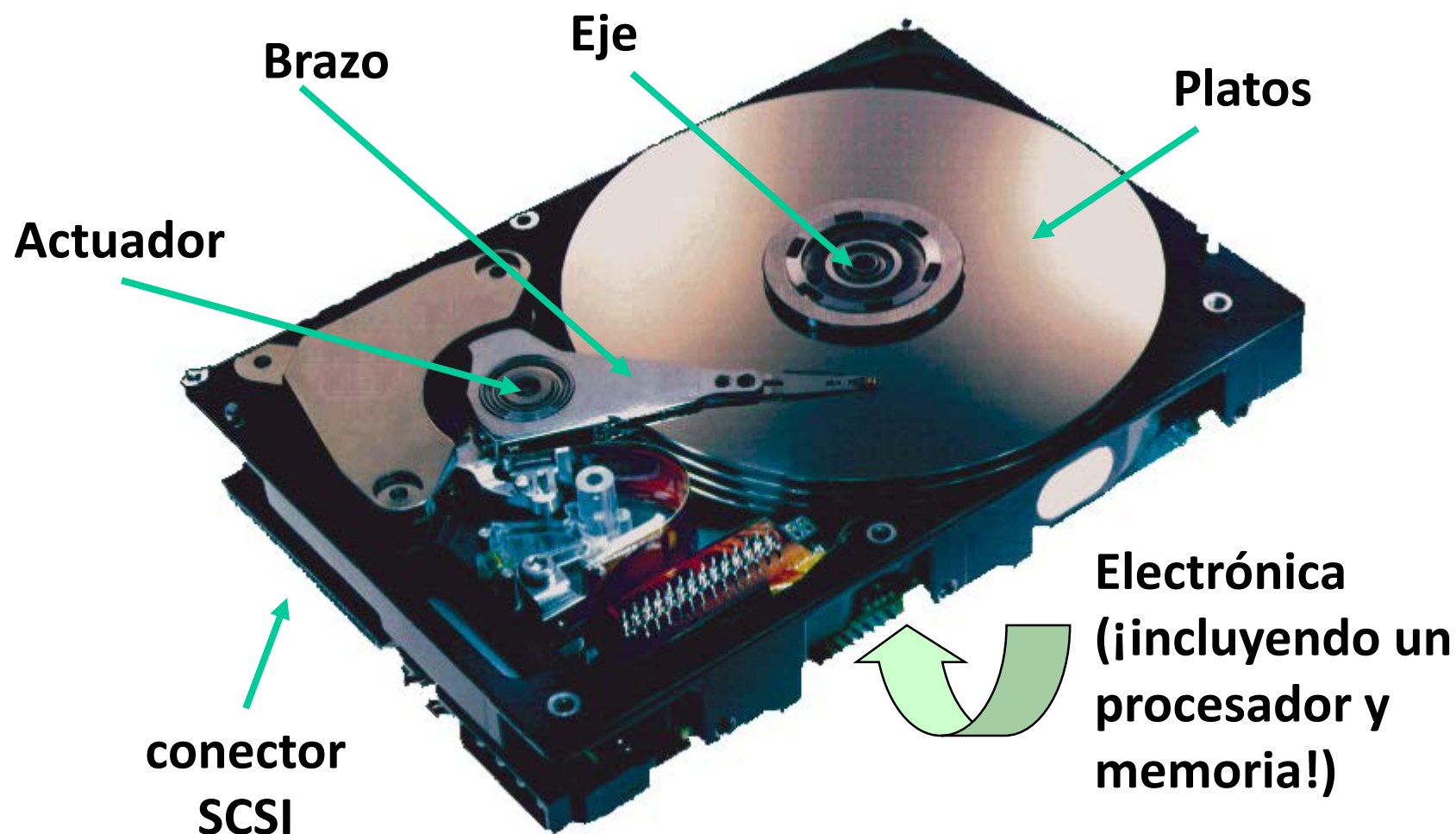
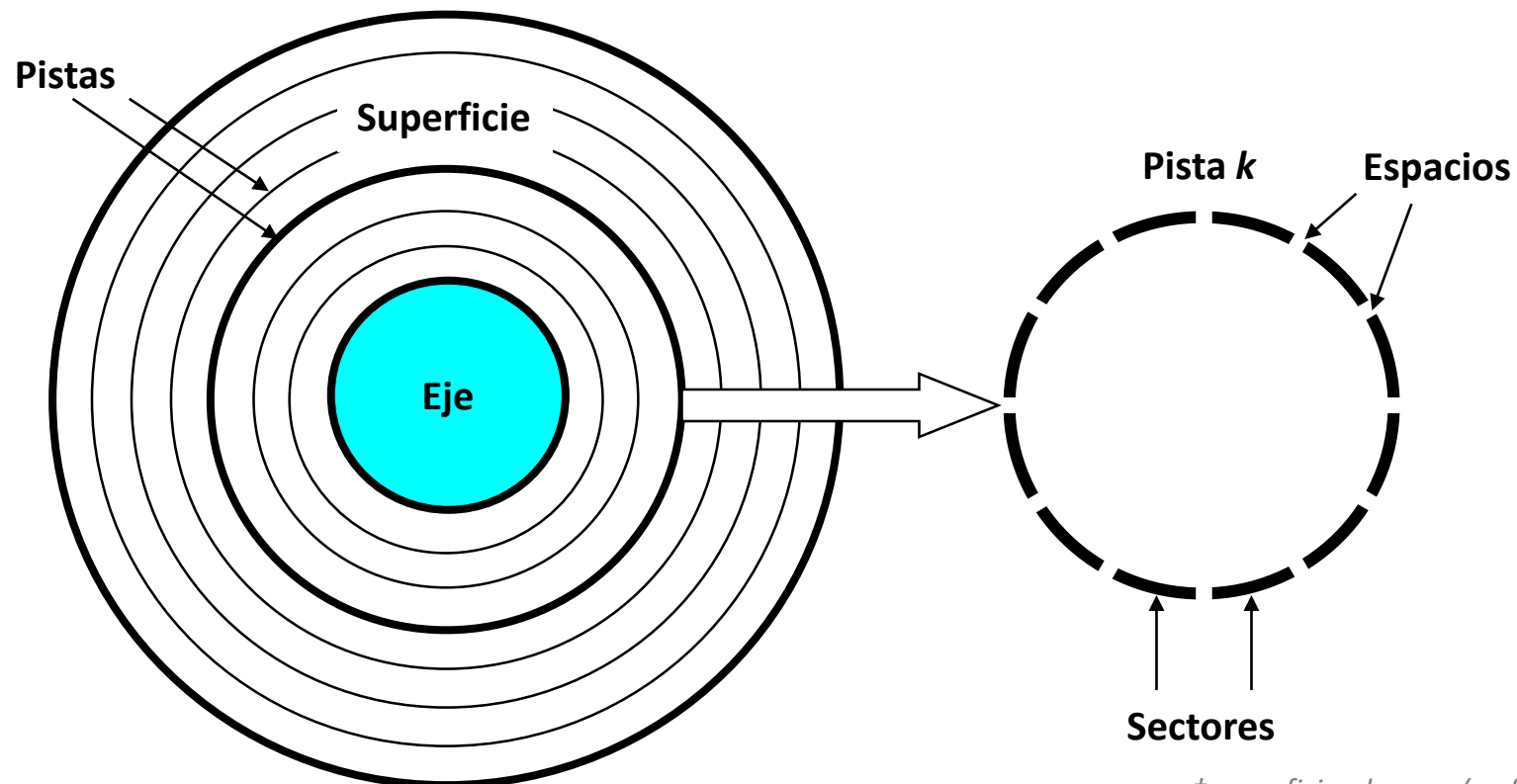


Imagen cortesía de Seagate Technology

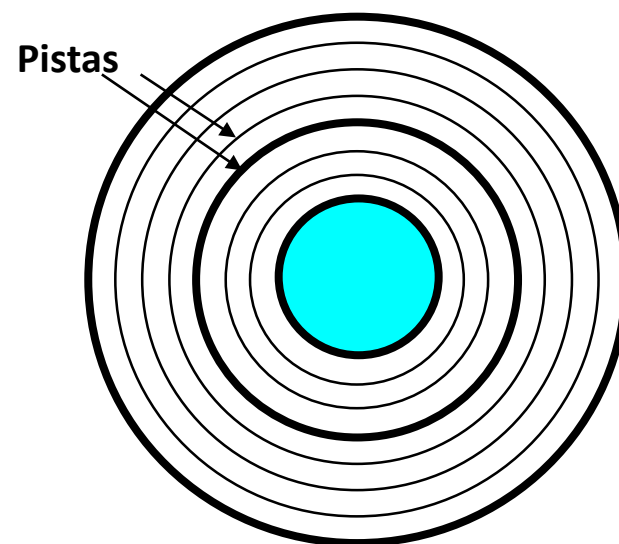
Geometría de un disco

- Los discos consisten en **platos** con dos **superficies (caras)**[†]
- Cada cara consiste en anillos concéntricos llamados **pistas**
- Cada pista consiste en **sectores** separados por **espacios**[†]



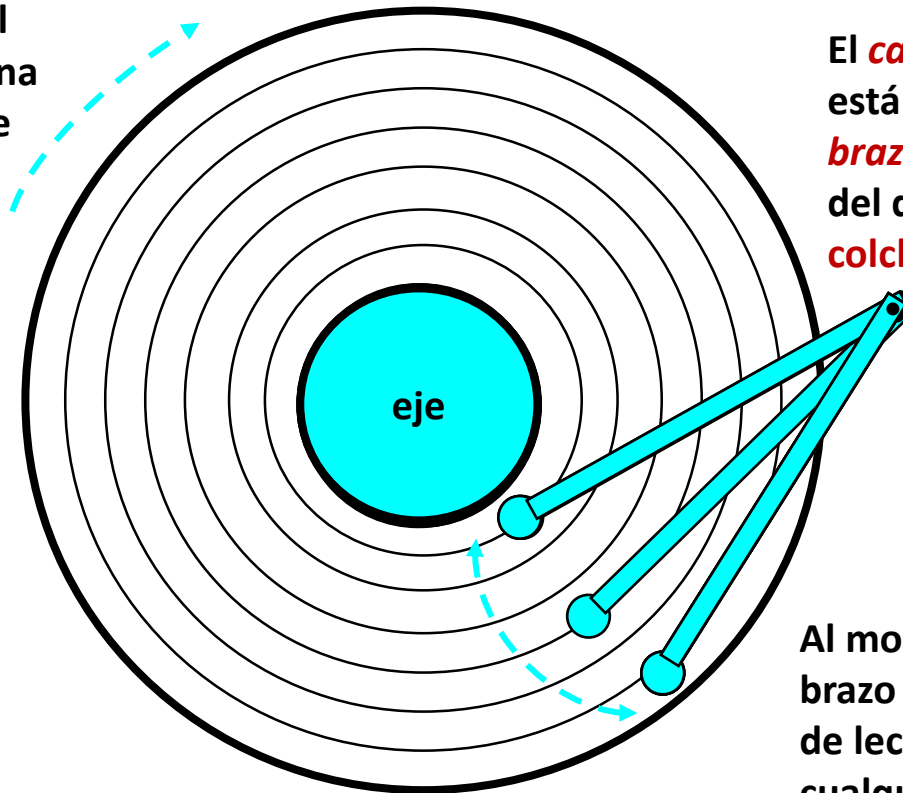
Capacidad de un disco

- **Capacidad:** máximo número de bits que se pueden almacenar
 - Los proveedores expresan la capacidad en unidades de gigabytes (GB) o terabytes (TB), donde $1 \text{ GB} = 10^9 \text{ Bytes}$ y $1 \text{ TB} = 10^{12} \text{ Bytes}$
- **La capacidad queda determinada por estos factores tecnológicos:**
 - **Densidad de grabación** (bits/pulgada[†]): nº de bits que se pueden comprimir en un tramo de pista de 1 pulgada
 - **Densidad de pistas (radial)** (pistas/pulgada): nº de pistas que se pueden comprimir en un tramo radial de 1 pulgada
 - **Densidad superficial** (bits/pulg²): producto de ambas densidades (grabación y radial)



Funcionamiento de un disco (Visto en un solo plato)

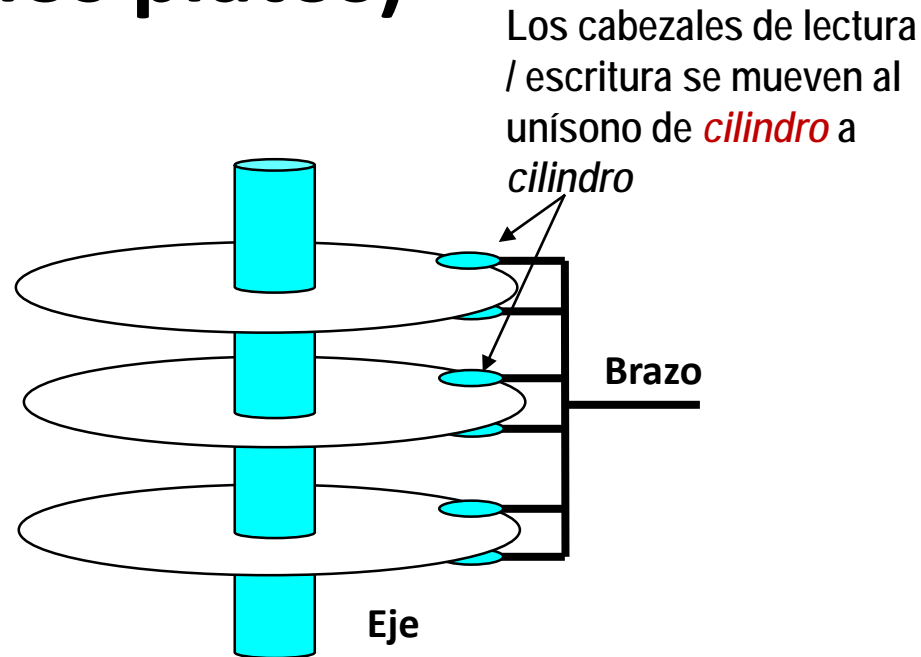
La superficie del disco gira con una velocidad fija de rotación



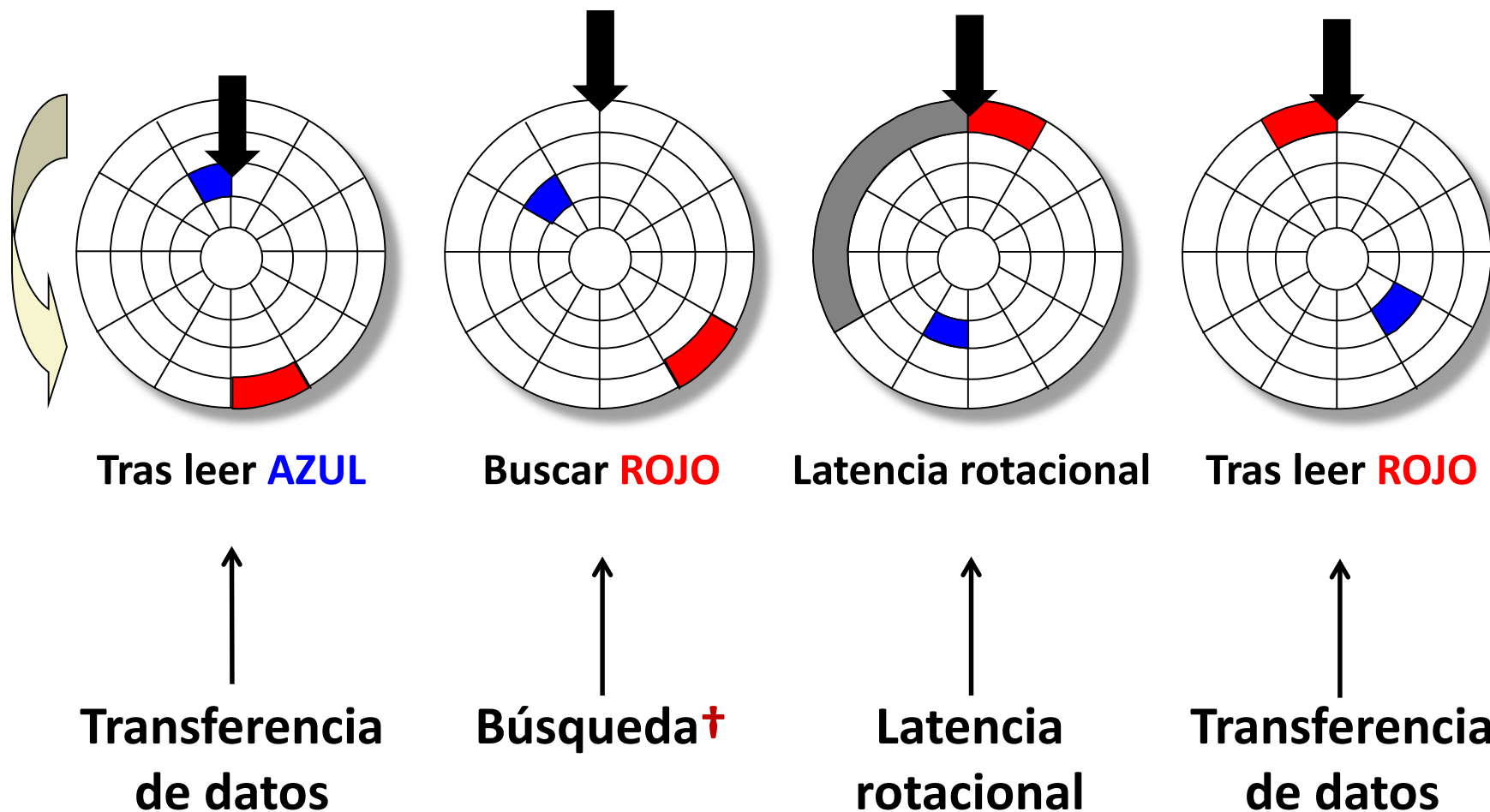
El **cabezal** de lectura / escritura está unido al extremo del **brazo** y vuela por la superficie del disco sobre un delgado **colchón** de aire

Al moverse radialmente, el brazo puede colocar el cabezal de lectura / escritura sobre cualquier pista.

Funcionamiento de un disco (Visto en varios platos)



Acceso a disco: componentes del tiempo de servicio



Tiempo de acceso a disco

■ Tiempo promedio acceso algún sector determinado, aprox:

- $T_{\text{acceso}} = T_{\text{prom búsqueda}} + T_{\text{prom rotación}} + T_{\text{prom transferencia}}$

■ Tiempo de búsqueda ($T_{\text{prom búsqueda}}$)

- Tiempo para colocar cabezales sobre el cilindro que contiene el sector
- Valores típicos $T_{\text{prom búsqueda}} = 3\text{—}9\text{ ms}$

■ Latencia rotacional ($T_{\text{prom rotación}}$)

- Tiempo esperando a que pase bajo cabezales el primer bit del sector
- $T_{\text{prom rotación}} = 1/2 \times 1/\text{RPMs} \times 60\text{ s}/1\text{ min}$
- Velocidad rotacional típica = 7,200 RPMs ($\Rightarrow 4.17\text{ ms}$)

■ Tiempo de transferencia ($T_{\text{prom transferencia}}$)

- Tiempo para leer los bits del sector
- $T_{\text{prom transferencia}} = 1/\text{RPMs} \times 1/(\# \text{ sectores/pista prom}) \times 60\text{ s}/1\text{ min}$

Tiempo para una rotación (minutos)  fracción de rotación a leer 

Ejemplo de tiempo de acceso a disco

■ Datos:

- Velocidad rotacional = 7,200 RPM
- Tiempo búsqueda promedio = 9 ms
- # sectores/pista promedio = 400

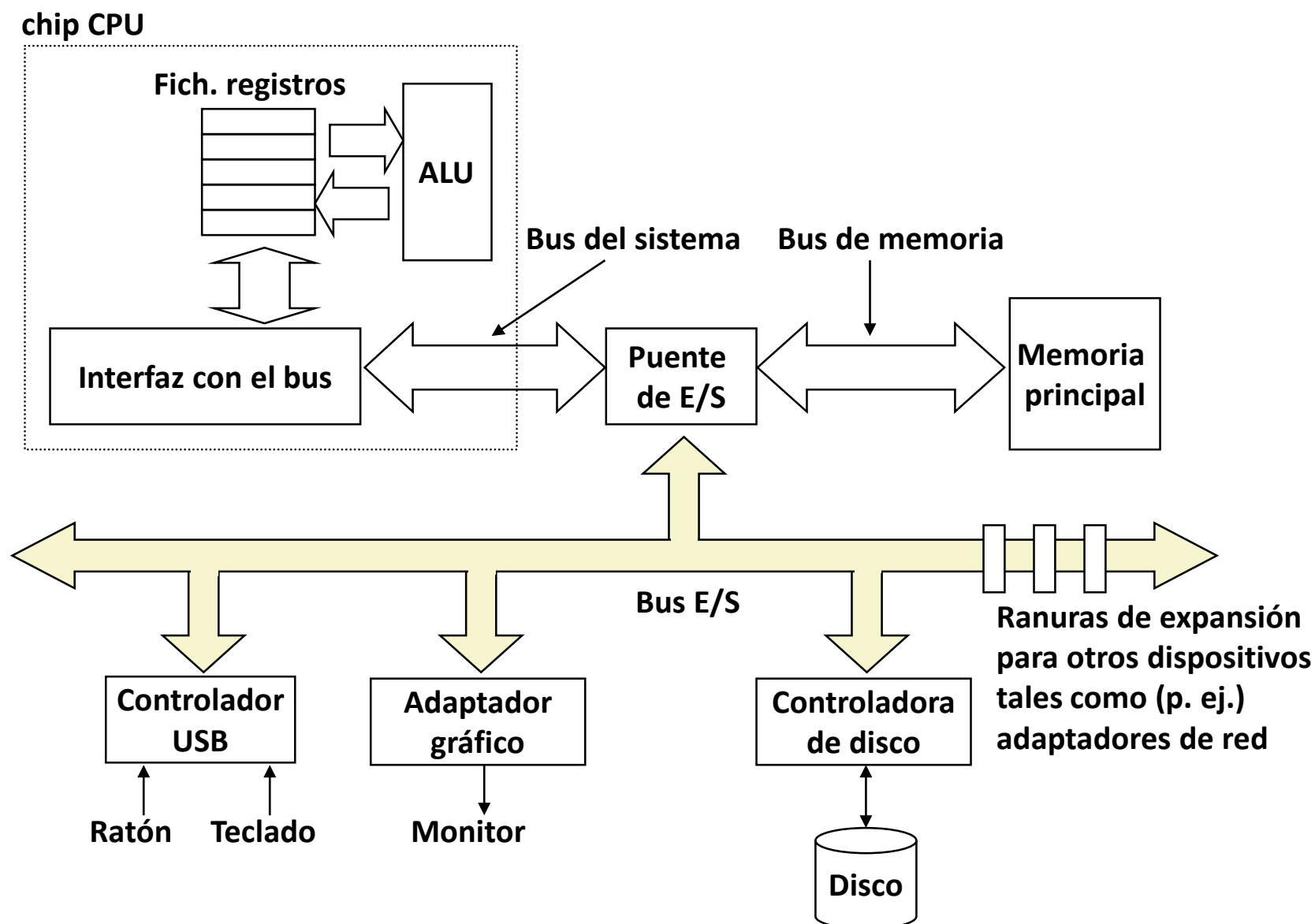
■ Calcular:

- $T_{\text{prom rotación}} = 1/2 \times (60 \text{ s}/7200 \text{ RPM}) \times 1000 \text{ ms/s} = 4.17 \text{ ms}$
- $T_{\text{prom transferencia}} = 60/7200 \times 1/400 \times 1000 \text{ ms/s} = 0.02 \text{ ms}$
- $T_{\text{acceso}} = 9 \text{ ms} + 4.17 \text{ ms} + 0.02 \text{ ms} = 13.19 \text{ ms}$

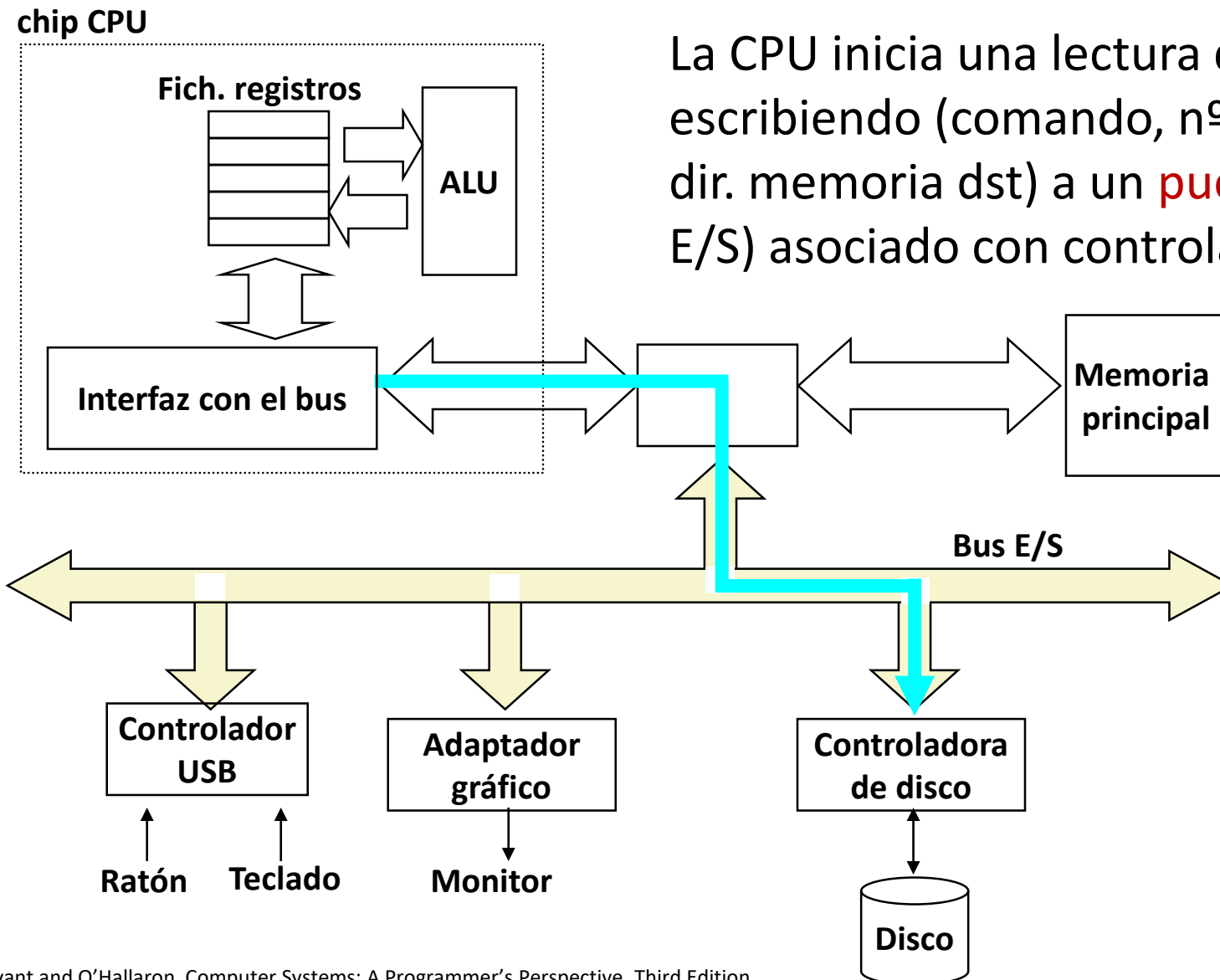
■ Puntos importantes:

- Tiempo acceso dominado por tiempo búsqueda y latencia rotacional
- El primer bit en un sector es el más caro (lento), el resto sale gratis.
- *Tiempo acceso SRAM aprox. 4ns/palabra 64b, DRAM aprox. 60ns*
 - *Disco es aprox. 40.000 veces más lento que SRAM,*
 - *2.500 veces más lento que DRAM[†]*

Bus de E/S

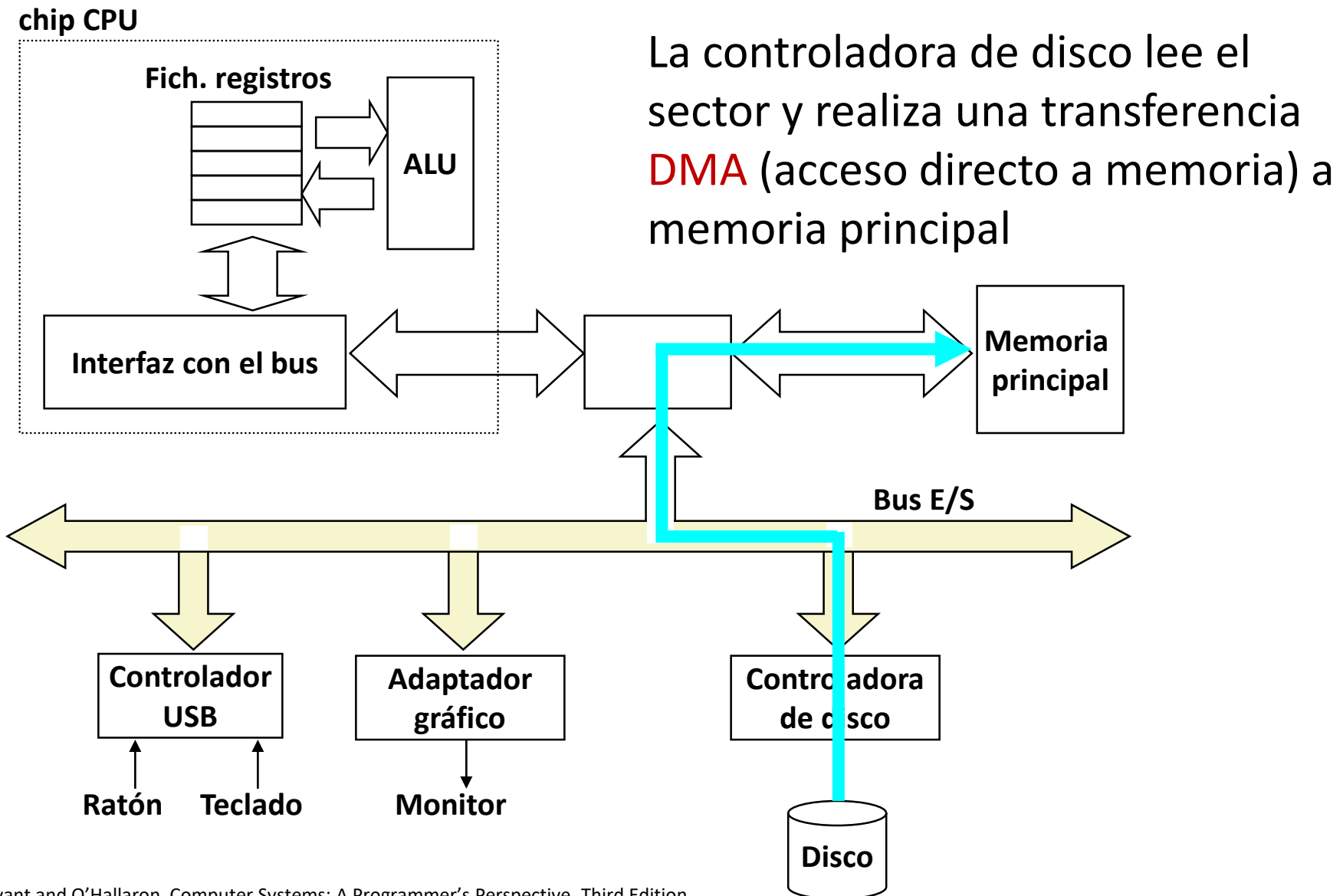


Lectura de un sector de disco (1)

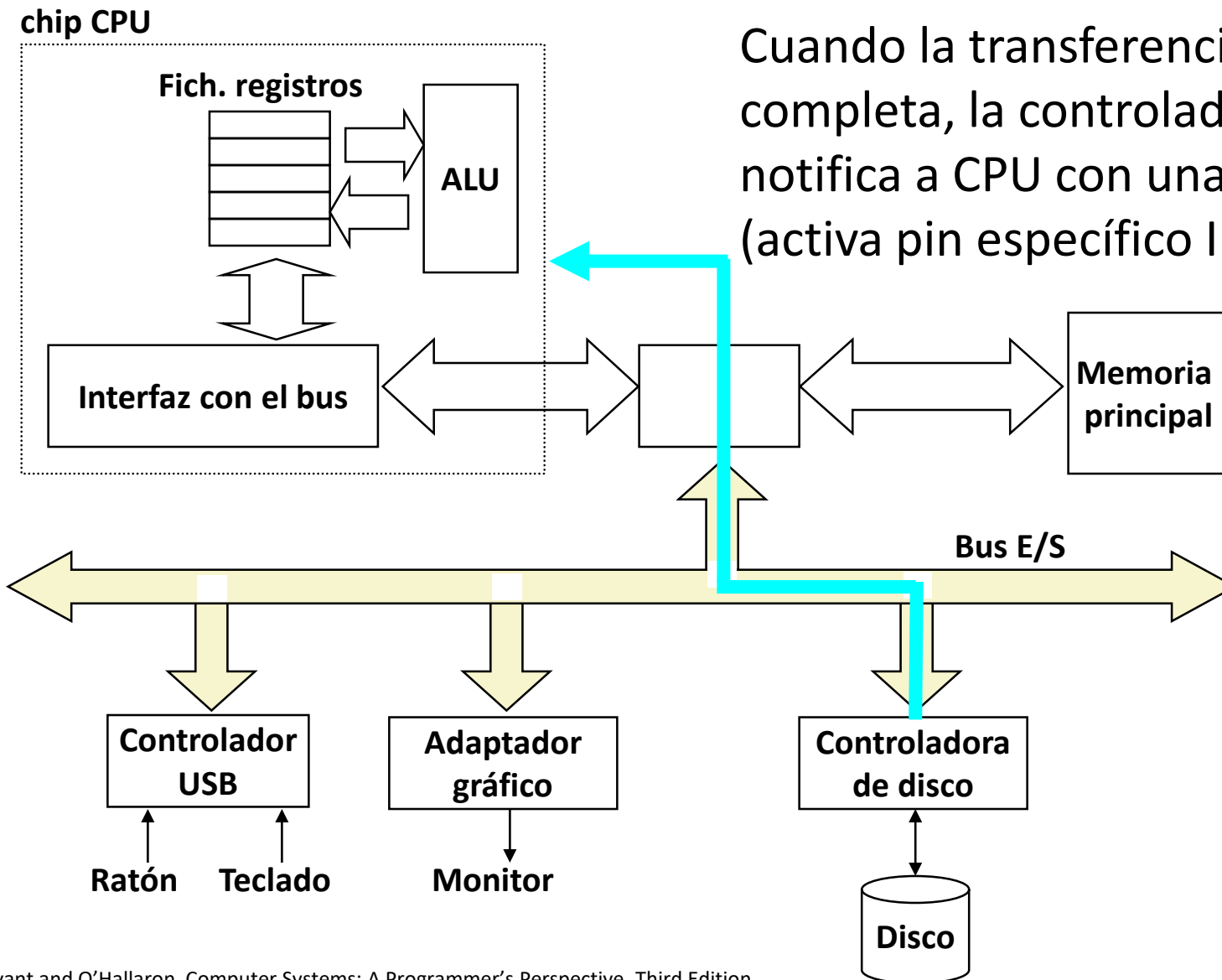


La CPU inicia una lectura de disco escribiendo (comando, nº bloque lógico, dir. memoria dst) a un **puerto** (dirección E/S) asociado con controladora de disco

Lectura de un sector de disco (2)



Lectura de un sector de disco (3)



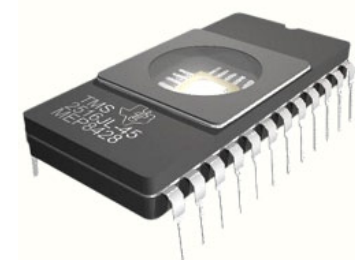
Memorias no volátiles

■ SRAM y DRAM son memorias volátiles

- Pierden información si se apagan

■ Las memorias no volátiles retienen valores incluso si se apagan

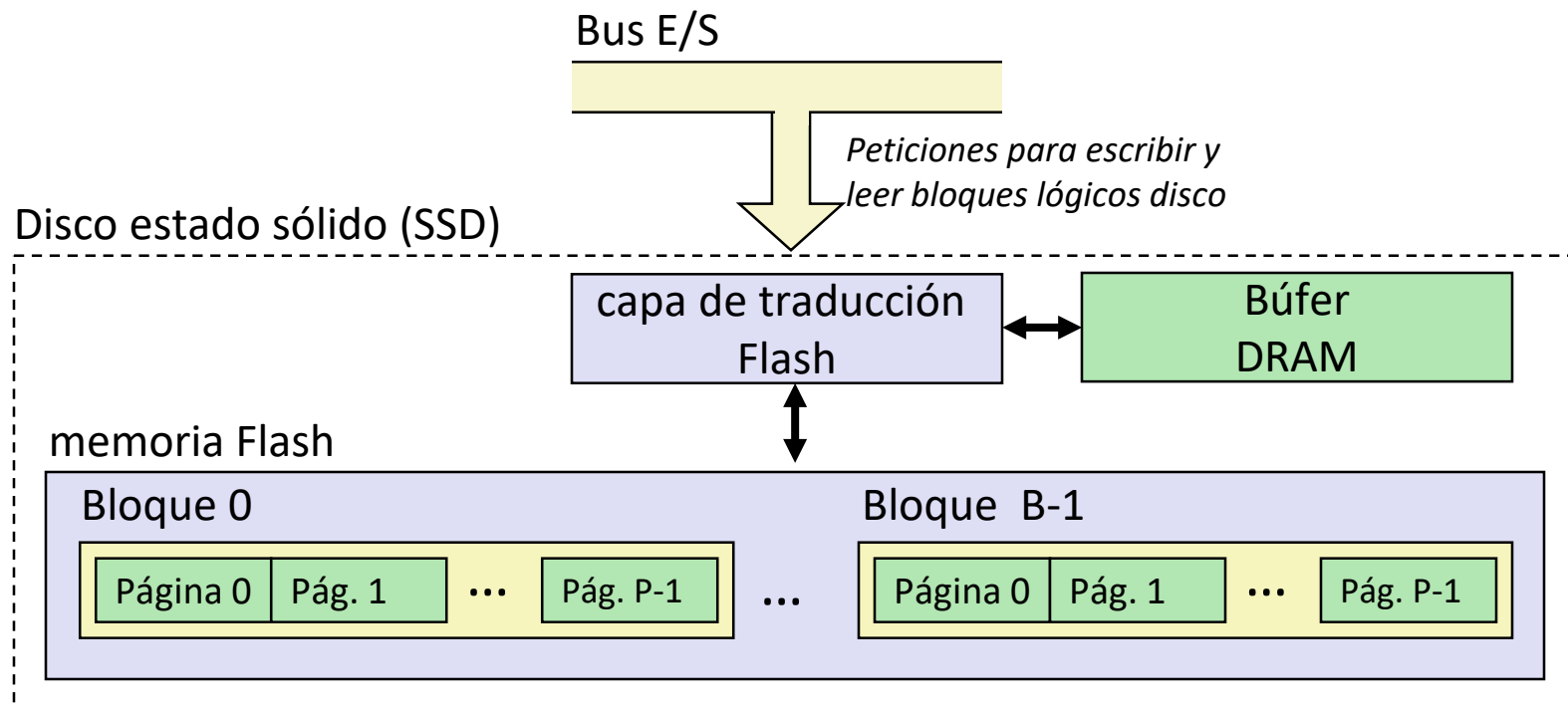
- Memoria de sólo lectura (**ROM**):
 - programada durante su fabricación
- **PROM** progr. usr. irreversible, **EPROM** borrable (luz UV)→
- **EEPROM**: PROM borrable eléctricamente
- Memorias **Flash**: EEPROMs con capacidad borrado parcial (por bloques)
 - se desgastan tras aprox. 100.000 borrados
- 3D XPoint[†] (Intel Optane) & NVMs emergentes
 - nuevos materiales



■ Aplicaciones de las memorias no volátiles

- Programas firmware almacenados en una ROM (BIOS, controladoras de disco, tarjetas de red, aceleradores gráficos, subsistemas seguridad...)
- Discos de estado sólido (reemplazando discos giratorios)
- Caches de disco

Discos de estado sólido (SSDs)



- **Páginas:** de 4KB a 512KB, **Bloques:** de 32 a 128 páginas
- **Datos escritos/leídos** en unidades de páginas
- Una página se puede escribir sólo tras borrar su bloque
- Un bloque se desgasta tras unas 100.000 escrituras

Prestaciones características SSD

■ Benchmark[‡] de un Samsung 970 EVO Plus

<https://ssd.userbenchmark.com/SpeedTest/711305/Samsung-SSD-970-EVO-Plus-250GB>

Rendimiento lectura secuencial	2.126 MB/s	Rendimto. escritura sec.	1.880 MB/s
Rendimiento [†] lectura aleatoria	140 MB/s	Rendimto. escritura aleat.	59 MB/s

■ Acceso secuencial mucho más rápido que aleatorio

- Tema omnipresente en jerarquías de memoria

■ Escrituras aleatorias son especialmente lentas

- Borrar un bloque lleva mucho tiempo (~1ms)
- Modificar una página de un bloque requiere que todas las otras se copien a un nuevo bloque
- La capa de traducción Flash permite acumular una serie de pequeñas escrituras antes de realizar una escritura de bloque

[‡] test, prueba de referencia

[†] throughput

SSD frente a Discos giratorios

■ Ventajas

- No partes móviles → más rápidos, menor consumo, más resistentes

■ Desventajas

- Eventual desgaste
 - mitigado por “lógica nivelado desgaste” en capa traducción flash
 - p.ej. Samsung 970 EVO Plus garantiza que se pueda escribir 600x la capacidad del disco antes de desgastarse[†]
 - controladora migra datos para repartir/minimizar nivel desgaste
- En 2019, aprox. 4x más caro por byte (que giratorios)
 - y seguirá cayendo ese coste relativo

■ Aplicaciones

- reproductores MP3, smartphones, portátiles
- cada vez más común en servidores y PC sobremesa

[†] 600x << 100.000 borrados bloque

ver folleto en <https://www.samsung.com/semiconductor/minisite/ssd/product/consumer/970evoplus/>

disponible en 0.25, 0.5, 1, 2TB con TBW = 150, 300, 600, 1200 TB **81.**

Memoria I: Jerarquía de memoria

- La abstracción de memoria (concepto de Lectura y Escritura)
- RAM: bloque constructivo de memoria principal
- Configuración y diseño de memorias utilizando varios chips
- Localidad de las referencias
- Jerarquía de memoria
- Tecnologías de almacenamiento, y tendencias

Resumen

- **La brecha de velocidad entre CPU, memoria y almacenamiento masivo continúa ampliándose.**
- **Los programas bien escritos exhiben una propiedad llamada localidad.**
- **Las jerarquías de memoria, basadas en cacheado, cierran la brecha al explotar la localidad.**
- **El progreso en memoria flash está sobrepasando a todas las demás tecnologías de memoria y almacenamiento (DRAM, SRAM, disco magnético)**
 - Capaz de apilar celdas en tres dimensiones

Guía de trabajo autónomo (4h/s)

■ **Estudio:** del Cap.6 CS:APP (Bryant/O'Hallaron)

- Accessing Main Memory, Random Access Memory, *Enhanced DRAMs*
 - § 6.1.1 pp.623-625, 615-621, 621-622
- Locality, The Memory Hierarchy
 - § 6.2 – 6.3 pp.640-650
- Disk Storage, *Nonvolatile Memory*, Solid State Disks, *Storage Technology Trends*
 - § 6.1.2 – .1.3 pp.625-636, 622-623, 636-638-640

■ **Ejercicios:** del Cap.6 CS:APP (Bryant/O'Hallaron)

- Probl. 6.1 § 6.1.1, p.620
- Probl. 6.7 – 6.8 § 6.2.3, pp.644, 645
- Probl. 6.2 – 6.6 § 6.1.2, pp.628, 631, 632, 637, 640

Bibliografía:

[BRY16] Cap.6

Computer Systems: A Programmer's Perspective 3rd ed. Bryant, O'Hallaron. Pearson, 2016

Signatura ESIIT/C.1 BRY com

Diapositivas suplementarias

Tendencias en almacenamiento

SRAM

Métrica	1985	1990	1995	2000	2005	2010	2015	2015:1985
\$/MB	2,900	320	256	100	75	60	25	116
acceso (ns)	150	35	15	3	2	1.5	1.3	115

DRAM

Métrica	1985	1990	1995	2000	2005	2010	2015	2015:1985
\$/MB	880	100	30	1	0.1	0.06	0.02	44,000
acceso (ns)	200	100	70	60	50	40	20	10
tam. típico (MB)	0.256	4	16	64	2,000	8,000	16.000	62,500

Disco

Métrica	1985	1990	1995	2000	2005	2010	2015	2015:1985
\$/GB	100,000	8,000	300	10	5	0.3	0.03	3,333,333
búsqueda (ms)	75	28	10	8	5	3	3	25
tam. típico (GB)	0.01	0.16	1	20	160	1,500	3,000	300,000

Frecuencia reloj CPU

Punto inflexión en historia computadores
al chocar diseñadores con “Muro Potencia”

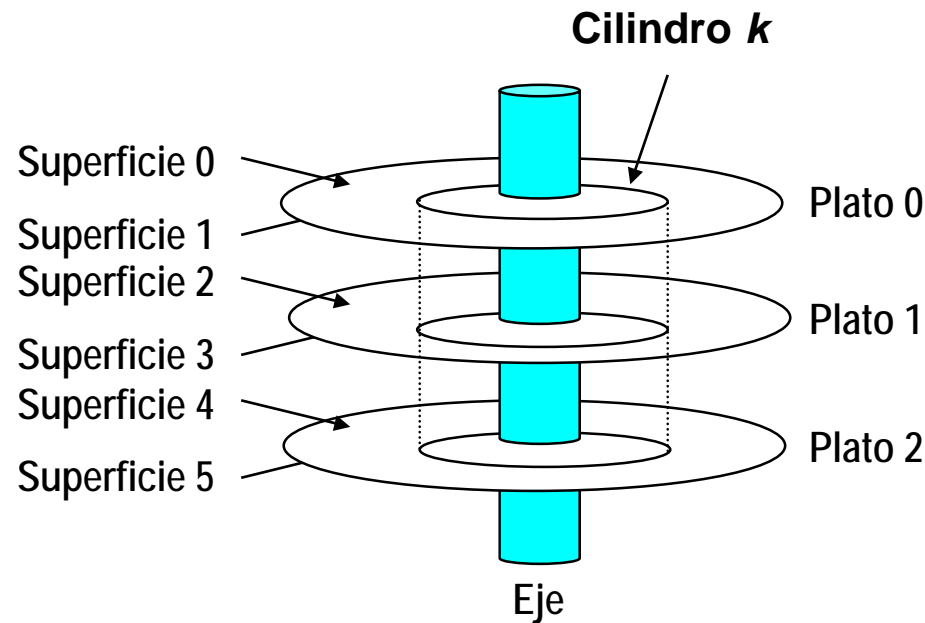
	1985	1990	1995	2003	2005	2010	2015	2015:1985
CPU	80286	80386	Pentium	P-4	Core 2	Core i7(n)	Core i7(h)	
Frecuencia reloj (MHz)	6	20	150	3,300	2,000	2,500	3,000	500
Tiempo ciclo (ns)	166	50	6	0.30	0.50	0.4	0.33	500
Cores	1	1	1	1	2	4	4	4
Tiempo efectivo ciclo (ns)	166	50	6	0.30	0.25	0.10	0.08	2,075

(n) procesador Nehalem
(h) procesador Haswell

Geometría de un disco

(Vista en varios platos)

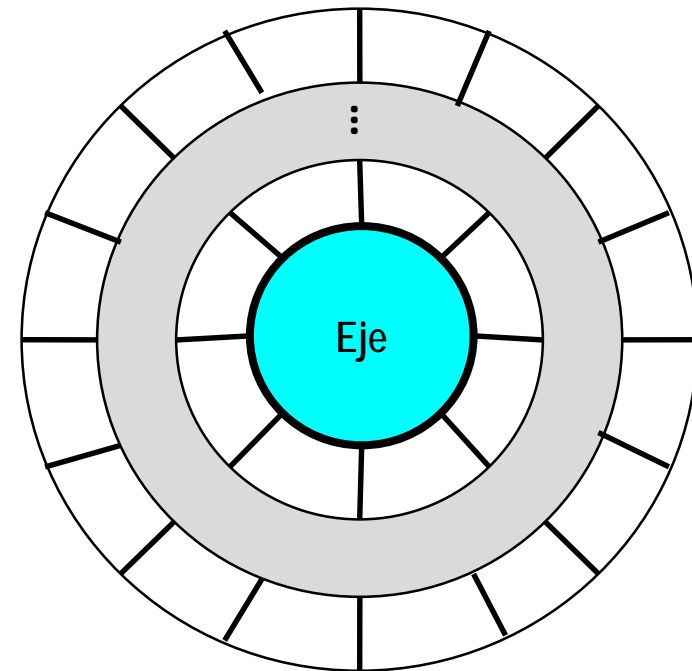
- Pistas alineadas forman un cilindro



Zonas de grabación

■ Los discos modernos particionan las pistas en subconjuntos disjuntos llamados **zonas de grabación**

- Las pistas de una zona tienen el mismo número de sectores, determinado por la circunferencia de la pista más interna.
- Cada zona tiene una cantidad diferente de sectores/pista, las zonas externas tienen más que las zonas internas.
- Así que usamos el promedio de sectores/pista cuando calculamos la capacidad.



Cálculo de la capacidad de un disco

**Capacidad = (# bytes/sector) x (# sectores/pista prom.) x
(# pistas/superficie) x (# superficies/plato) x
(# platos/disco)**

Ejemplo:

- 512 bytes/sector
- 300 sectores/pista (en promedio)
- 20,000 pistas/superficie
- 2 superficies/plato
- 5 platos/disco

**Capacidad = 512 x 300 x 20000 x 2 x 5
= 30,720,000,000
= 30.72 GB**

Bloques (de disco) Lógicos[†]

- Los discos “modernos” presentan una visión más simple y abstracta de la compleja geometría de sectores en disco:
 - El conjunto de sectores disponibles se modela como una secuencia (0, 1, 2, ...) de bloques lógicos de tamaño b
- Correspondencia entre bloques lógicos y sectores reales (físicos)
 - Mantenido en hardware/firmware (controladora de disco)
 - Convierte peticiones de bloques lógicos en tripletes (superficie, pista, sector).
- Permite al controlador reservar cilindros de repuesto para cada zona.
 - Contribuye a la diferencia entre "capacidad formateada" y "capacidad máxima"