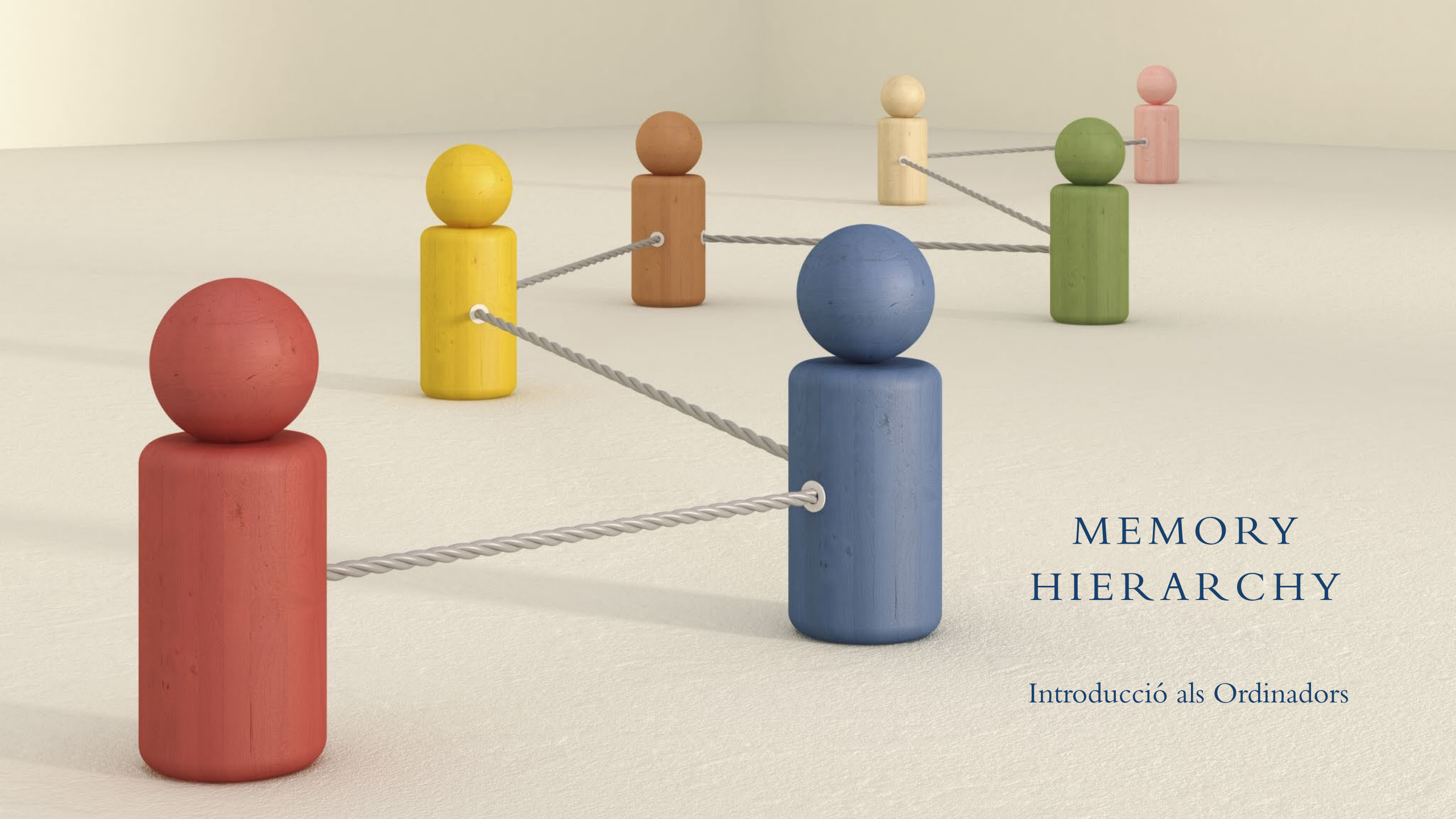


MEMORY HIERARCHY

Introducció als Ordinadors

Memory hierarchy

What Memory hierarchy means?

Memory levels & Memory Performance

Locality:

- **Temporal Locality:** If an item is referenced, it Will tend to be referenced again soon. If you recently brought a book to your desk to look at, you Will probably need to look at it again soon.
- **Spatial Locality:** If an item is referenced, items whose addresses are close by Will tend to be referenced soon. Libraries put books on the same topic together on the same shelves to increse spatial locality.

We take advantage of the principle of locality by implementing the memory of a computer as **memory hierarchy**. Multiple levels of memory with different speeds and sizes.

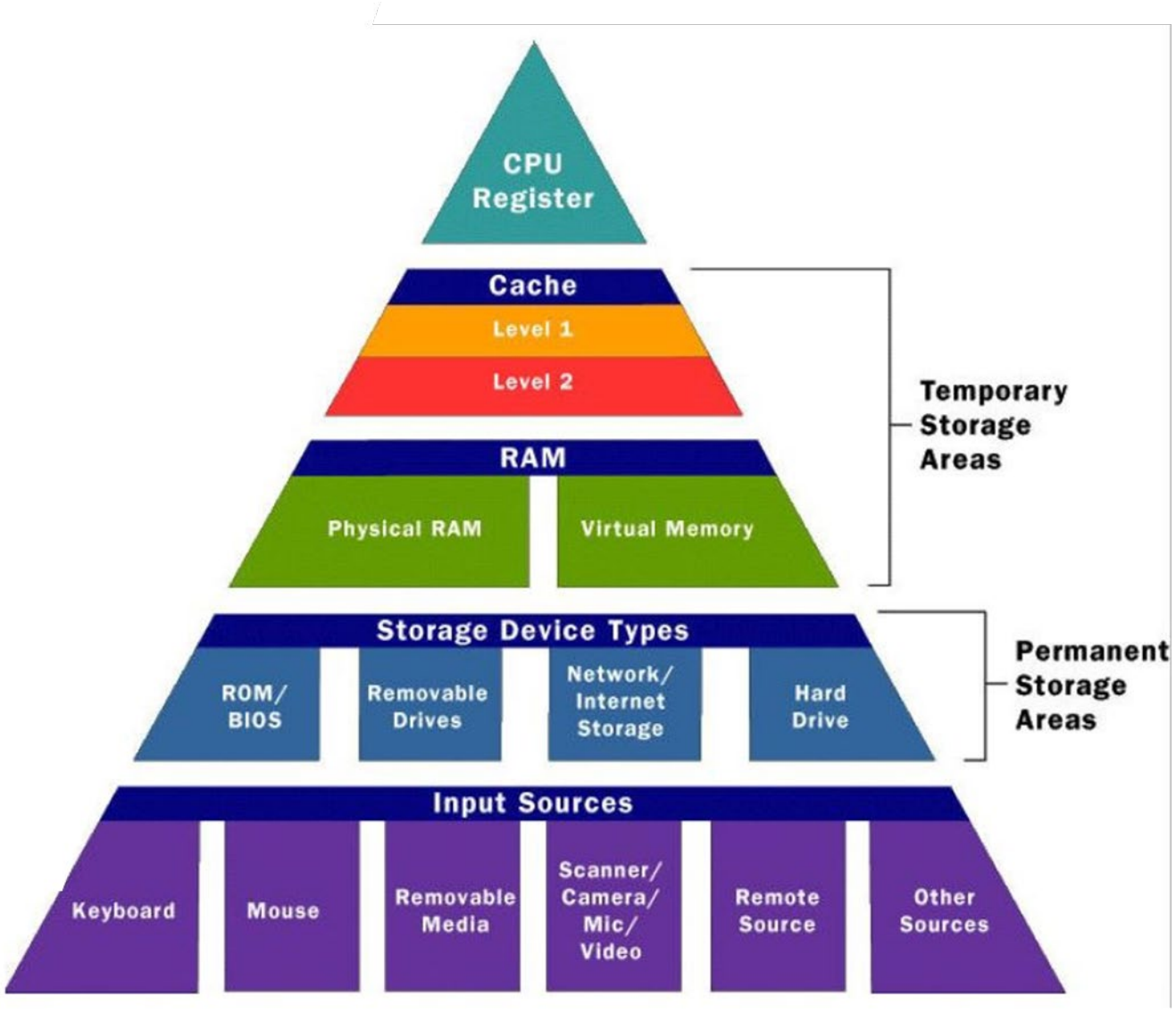
A memory hierarchy can consist of multiple levels, but data are copied between only two adjacent levels at a time, so we can focus our attention on just two levels.

JERARQUIA I MEMÒRIA: TERMINOLOGIA I CONCEPTES

Cada nivell és
+petit
+ràpid
+car
que l'inferior

Objectiu:
Cost proper al nivell més barat
Velocitat propera al més ràpid

Les dades guardades en un nivell també ho estan en el nivell inferior

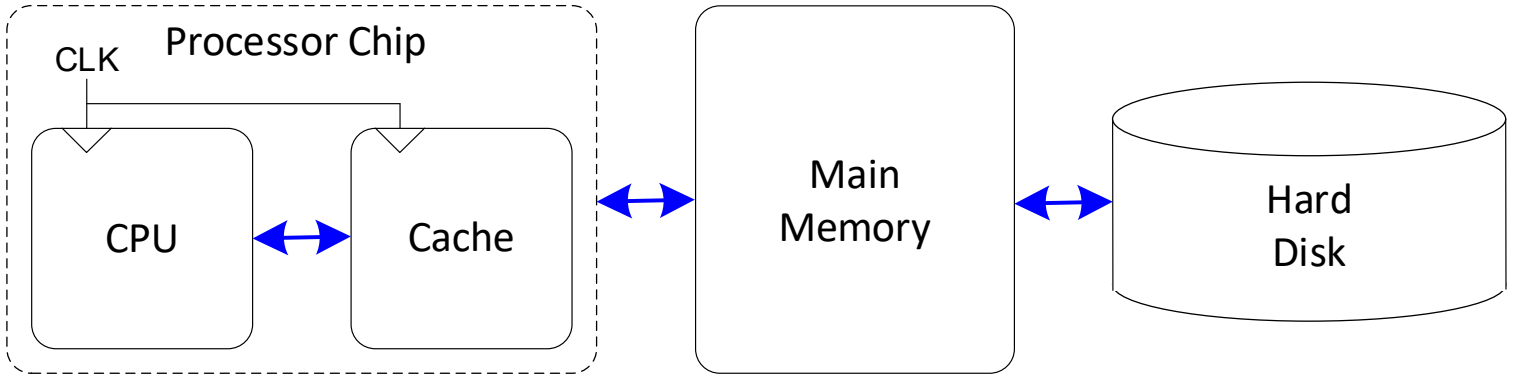


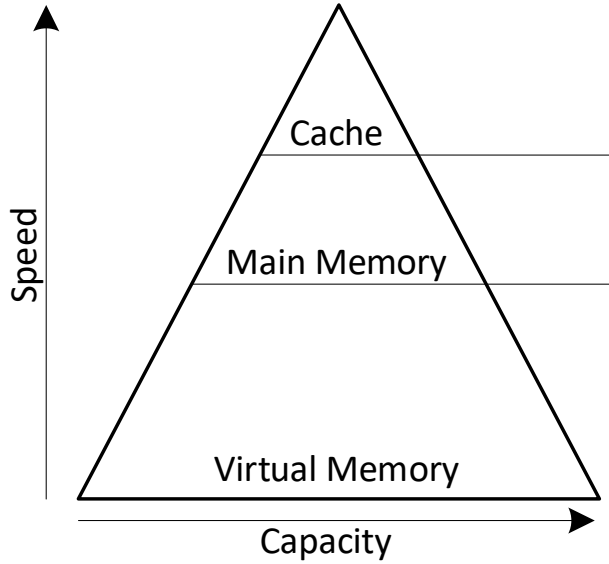
Locality:

Exploit locality to make memory accesses fast:

- **Temporal Locality:**
  - Locality in time
  - If data used recently, likely to use it again soon
  - **How to exploit:** keep recently accessed data in higher levels of memory hierarchy
- **Spatial Locality:**
  - Locality in space
  - If data used recently, likely to use nearby data soon
  - **How to exploit:** when access data, bring nearby data into higher levels of memory hierarchy too

Memory Hierarchy

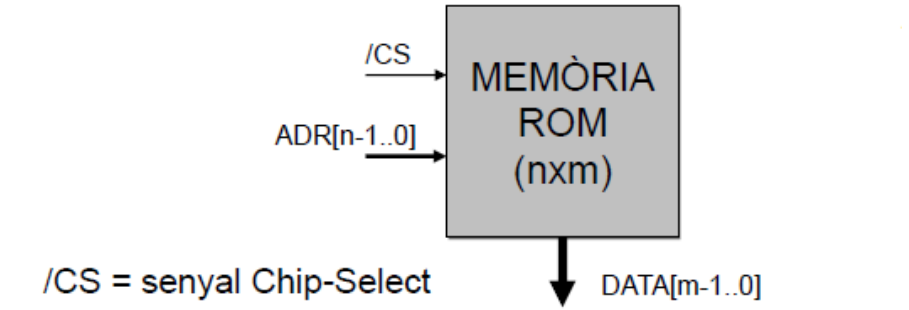


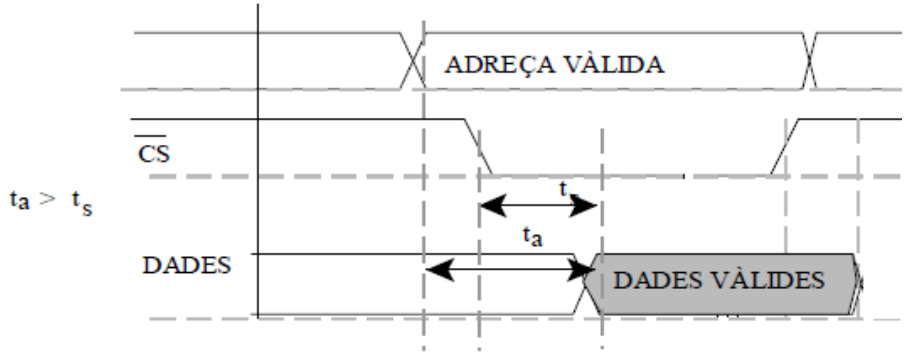


Technology	Price / GB	Access Time (ns)	Bandwidth (GB/s)
SRAM	$100	0.2 - 3	100+
DRAM	$3	10 - 50	30
SSD	$0.10	20,000	0.05 - 3
HDD	$0.03	5,000,000	0.001 - 0.1

Memory Hierarchy

Temps d'accés o latència: Es defineix el temps d'accés a una memòria com l'interval de temps que passa entre que l'adreça surt pel bus de adreces i la dada es col·loca en el bus de dades





Memory Hierarchy

- **Hit:** data found in that level of memory hierarchy
- **Miss:** data not found (must go to next level)

Hit Rate (HR) = # hits / # memory accesses
= 1 – Miss Rate

Miss Rate (MR) = # misses / # memory accesses
= 1 – Hit Rate

- **Average memory access time (AMAT):** average time for processor to access data

$$\text{AMAT} = t_{cache} + MR_{cache}[t_{MM} + MR_{MM}(t_{VM})]$$

Memory Hierarchy

- Exemple:

  Tenim un subsistema de memòria organitzat en dos nivells de jerarquia. En el nivell superior tenim una cache de 16kB i Ta = 20 ns. Al nivell inferior tenim una memòria principal de 8MB i Ta2 = 100 ns. Si la tassa d'encerts és del 90% quin és el temps d'accés mitjà a una paraula del subsistema de memòria??

$$\textbf{AMAT} = t_{\text{cache}} + MR_{\text{cache}}[t_{MM} + MR_{MM}(t_{VM})]$$

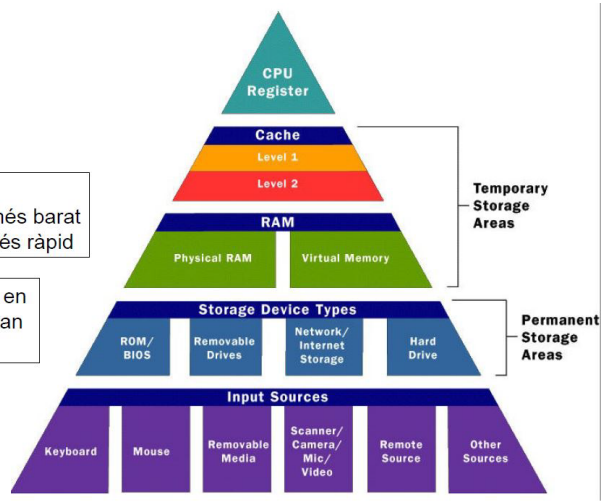
$$\textbf{AMAT} = t_{\text{cache}} + MR_{\text{cache}}[t_{MM}] = 20\ ns + (1-0.9)[100\ ns] = 30\ \text{ns}$$

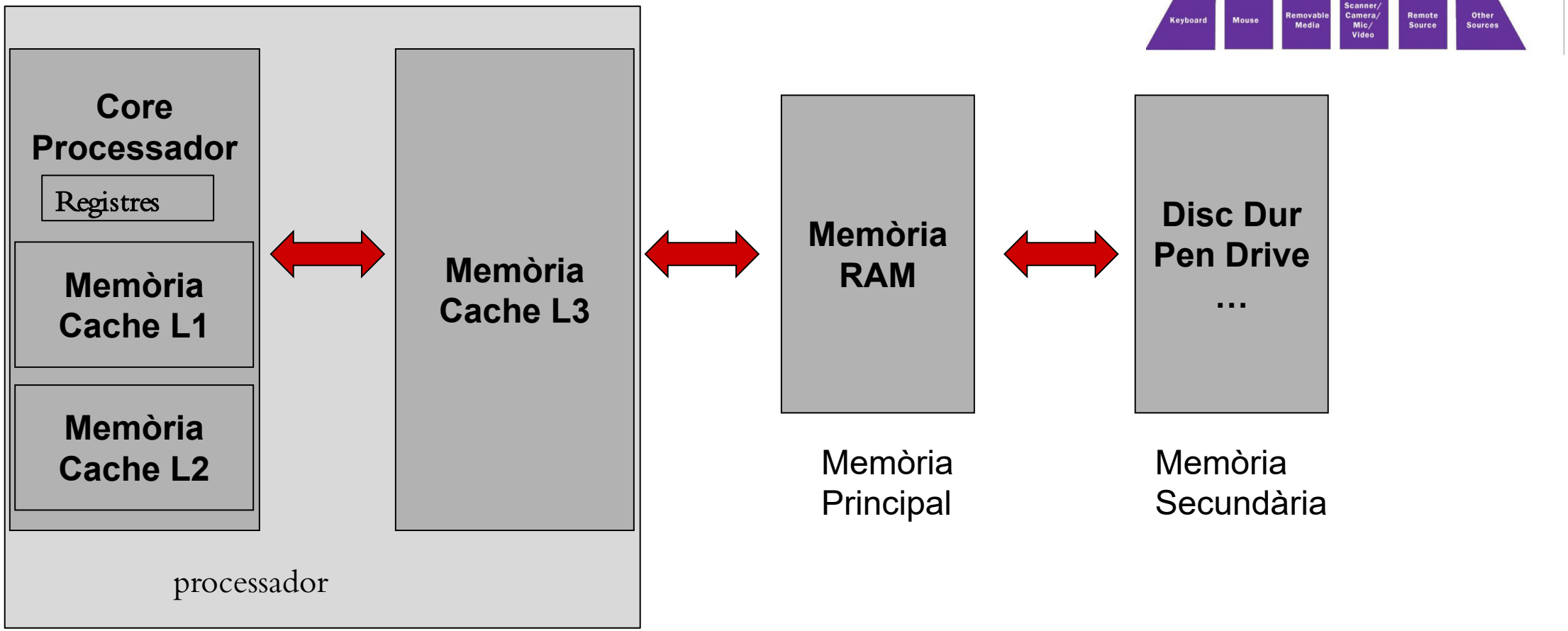
Memory Hierarchy

Cada nivell és
+petit
+ràpid
+car
que l'inferior

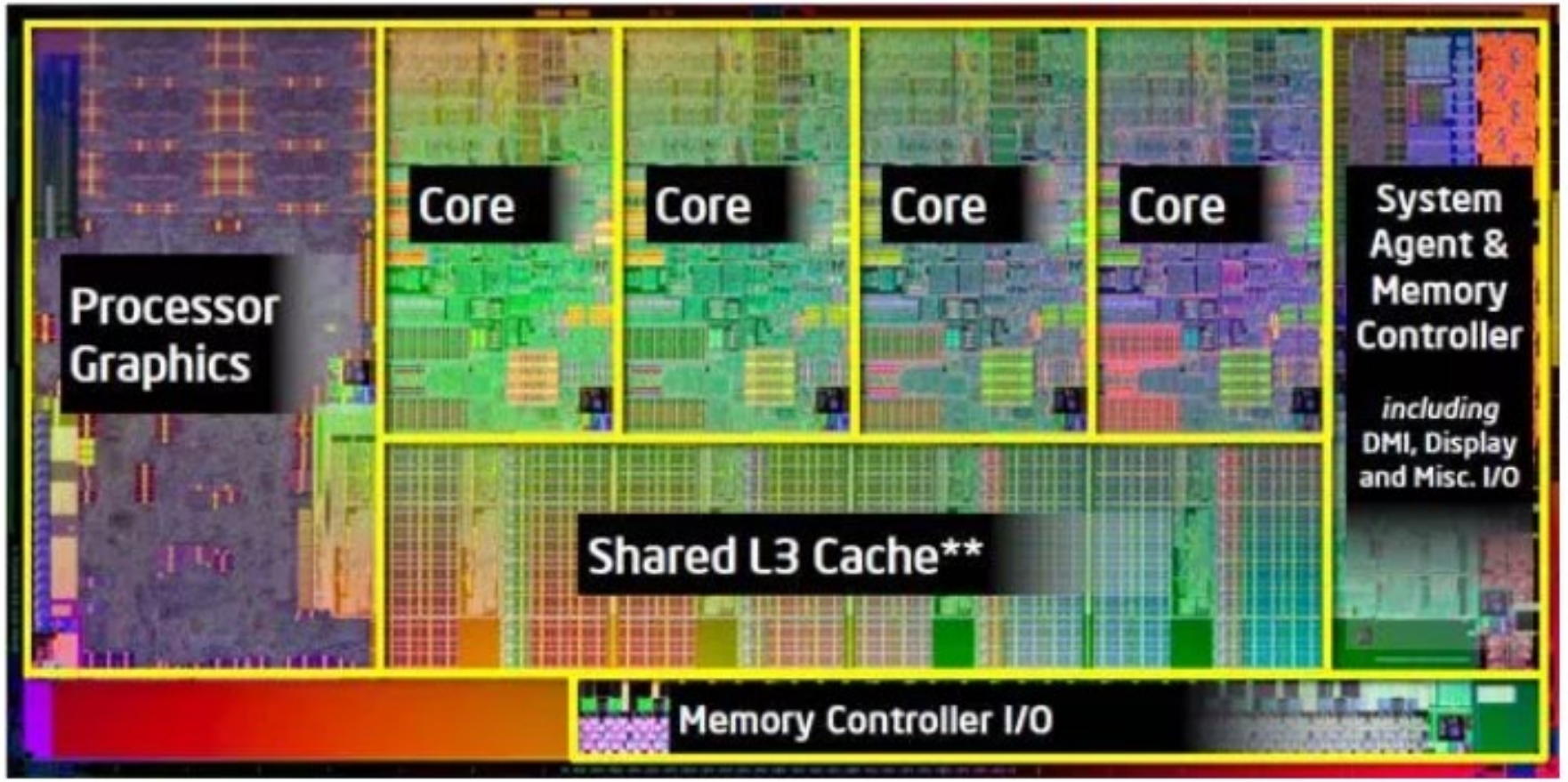
Objectiu:
Cost proper al nivell més barat
Velocitat propera al més ràpid

Les dades guardades en un nivell també ho estan en el nivell inferior





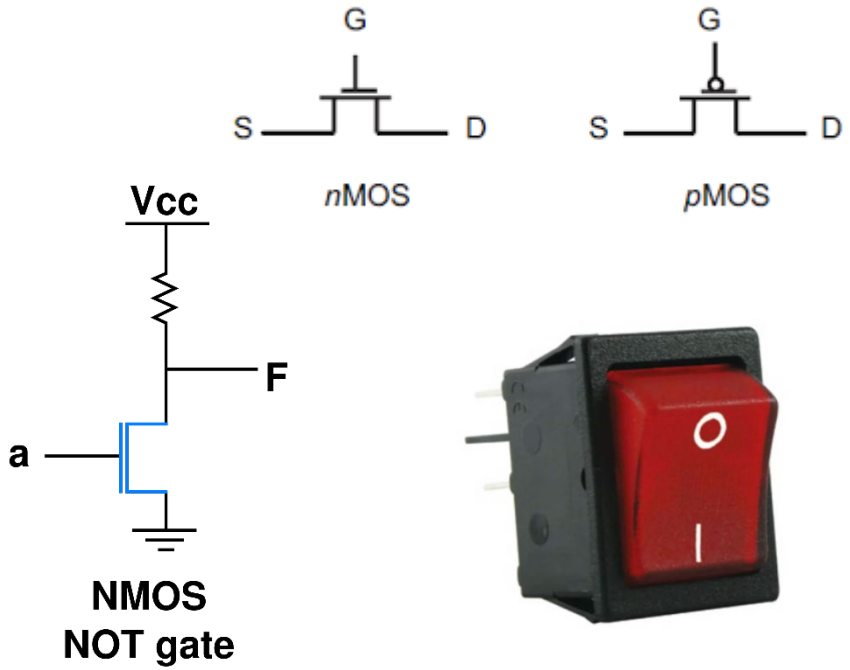
Memory Hierarchy

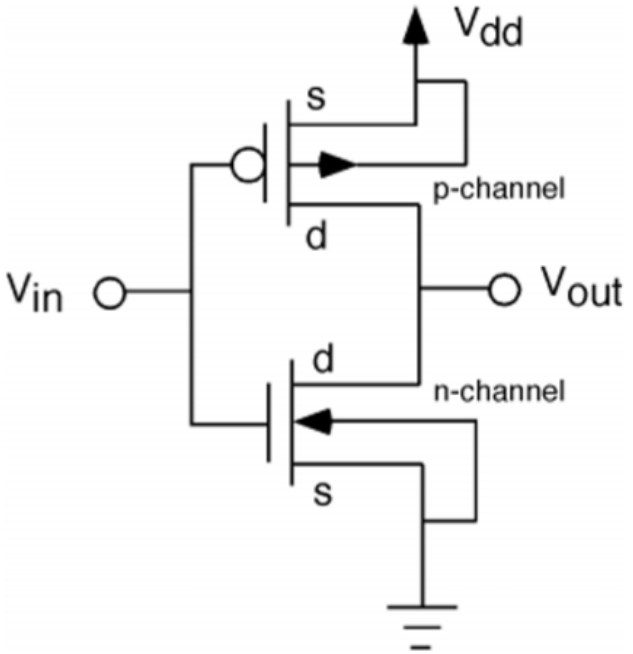


Die procesador Intel Core i7-4790K

Fonaments de Memòries

- La base en el disseny d'una memòria és el transistor MOS (Metall – Òxid – Semiconductor)
- Actua com un interruptor en funció de la tensió que tingui a l'entrada (porta G)
- Normalment es fa servir tecnologia CMOS





Memory Technologies

Permanència d'informació	Accés		Escriptura	
	Accés aleatori	Accés especial	Programació	Esborrat
Permanents (no volàtils)	ROM		Màscara	No
	PROM - OTP		Elèctrica	No
	EPROM		Elèctrica	Llum UV
	EEPROM		Elèctrica	Elèctric
	Flash-EEPROM	EEPROM sèrie	Elèctrica	Elèctric - blocs
			Elèctrica	Elèctric
	NVRAM		Elèctrica	Elèctric (bat)
	FRAM		magnètica	magnètica
Volàtils (quan Vcc=>0)	RAM	LIFO	Elèctrica	Elèctric
	SRAM	FIFO	Elèctrica	Elèctric
	DRAM	CAM	Elèctrica	Elèctric

MEMÒRIES RAM (Random Access Memory)

Memòries d'accés aleatori: adreça – dada

S'utilitzen com a primària dels ordinadors (Cache i memòria principal "RAM"),

Nom més adient seria memòries de *lectura/escriptura* **RWM** (***Read-Write Memory***)

De fet, el més correcte seria *Memòries de lectura-escriptura* ***intensiva*** (a diferència de les EEPROM que estan pensades per un operació "Alguna escriptura/Moltes lectures")

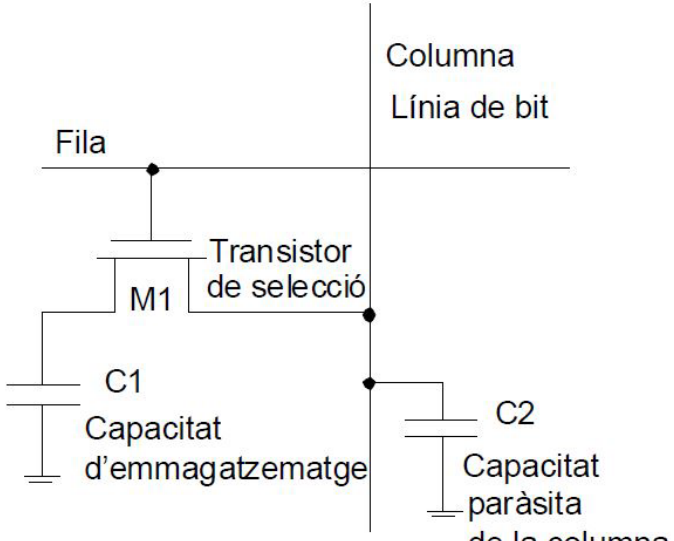
Es divideixen en dos tipus principals segons constitució de la cel·la d'emmagatzematge:

Memòries RAM estàtiques (SRAM) => No necessita refresc (inversors)

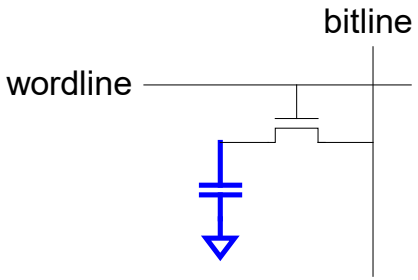
Memòries RAM dinàmiques (DRAM) => Necessita refresc (capacitat)

Memòries DRAM: Estructura bàsica

- **Dinàmiques** perquè necessiten de senyal de rellotge per mantenir les dades.
- La cèl·la de memòria està constituïda per un transistor i un condensador



CEL·LA D'EMMAGATZEMATGE



Procés de lectura (escriptura) és transferència de càrrega de C1 a C2 (de C2 a C1)

El canvi en el valor de la columna controlat per la relació entre les capacitats (C2 =10·C1)

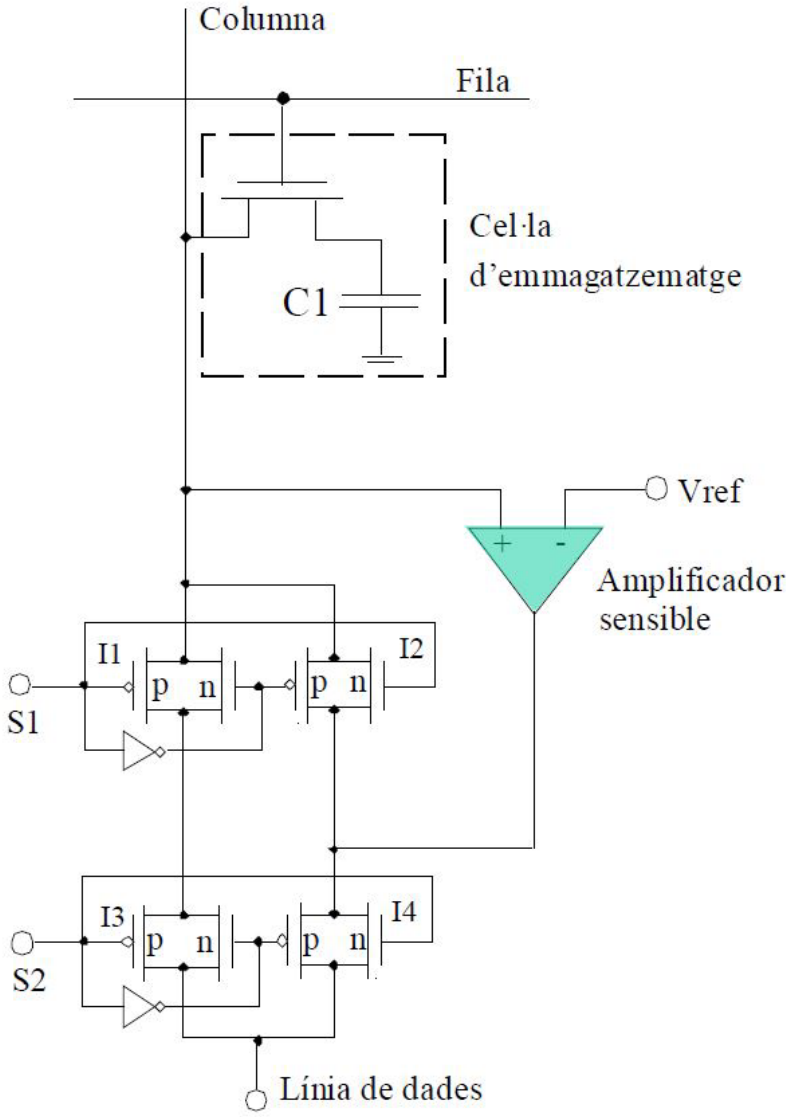
La transferència de càrrega és C2/(C1+C2)

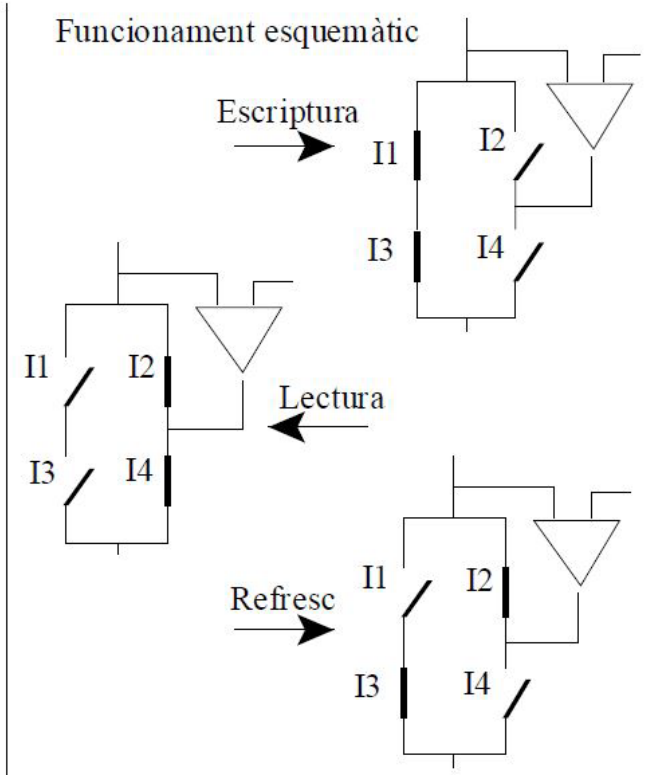
Excursió de tensió petita entre nivells L-H

A cada lectura (transferència de càrrega) es perd càrrega, cal reescriure.

La càrrega es perd amb el temps per fuites. Cal refrescar la dada.

Memòries DRAM: Sistema de refresc





Senyals de control d'operacions

	Escriptura	Lectura	Refresc
S1	0	1	1
S2	0	1	0

Memòries DRAM

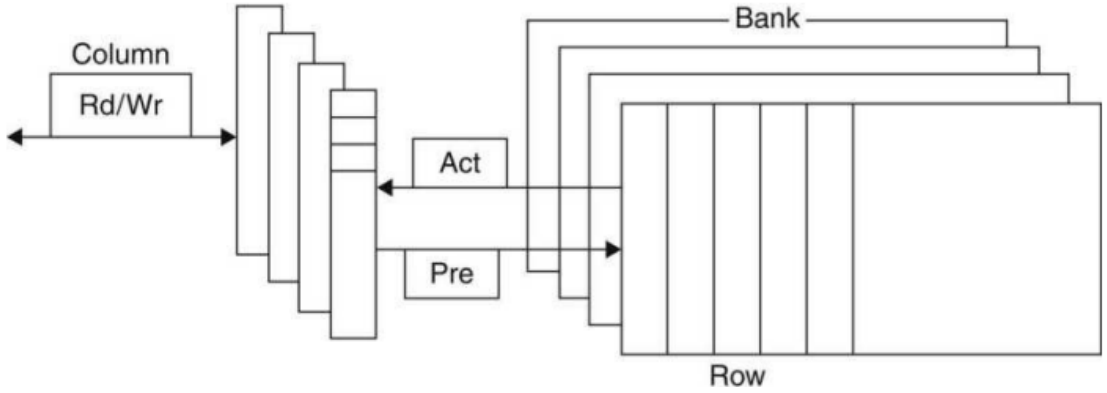
To refresh the cell, we merely read its contents and write it back.

The change can be kept for several milliseconds.

If every bit had to be read out of the DRAM and then written back individually, we would constantly be refreshing the DRAM, leaving no time for accessing it.

DRAM uses a two-level decoding structure and this allows us to refresh an entire row (which shares a Word line) with a read cycle followed immediately by a write cycle.

Memòries DRAM



Modern DRAMs are organized in Banks. Each banck consist of a series of rows.

Sending a precharge command, open or closes a banck.

A row address is sent with an ACT (activate), which causes the row to transfer to a buffer.

When the row is in the buffer, it can be transfered by successive columna addresses at whatever the width of the DRAM is or by specifying a block transfer and the starting address.
Each command as well as blocks transfer is synchronized with a clock

Memòries DRAM

To improve the interface to processors further, DRAMs added clocks and are propertly called synchronous DRAM (SDRAM) – Eliminates the time for the memory and processor to synchronize.

The speed advantage of SDRAMs comes from the ability to transfer the bits in the burst without having to specify additional address bits.

The fastest versión is called Double Data Rate (DDR) SDRAM. The name means data transfers on both rising and falling Edge of the clock ➔ Twice as much bandwidth

DDR4 – 3200 DRAM can do 3200 million transfer per second.

Memòries DRAM

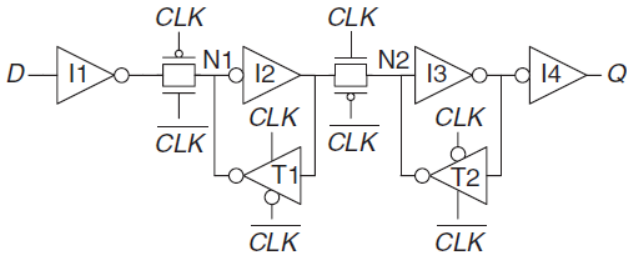
Additional Info

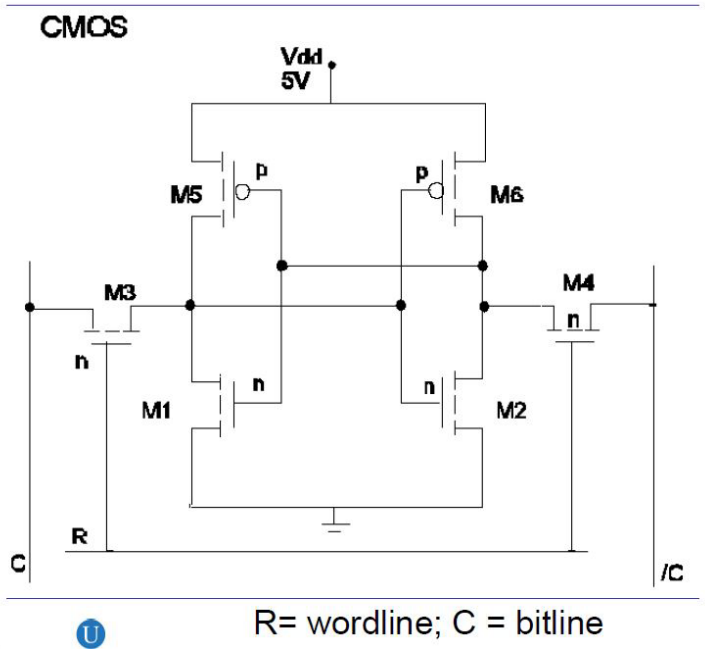
Clever organization inside the DRAM is needed. Instead of just a faster row buffer, the DRAM can be internally organized to read or write from multiple Banks, with each having is own row buffer.

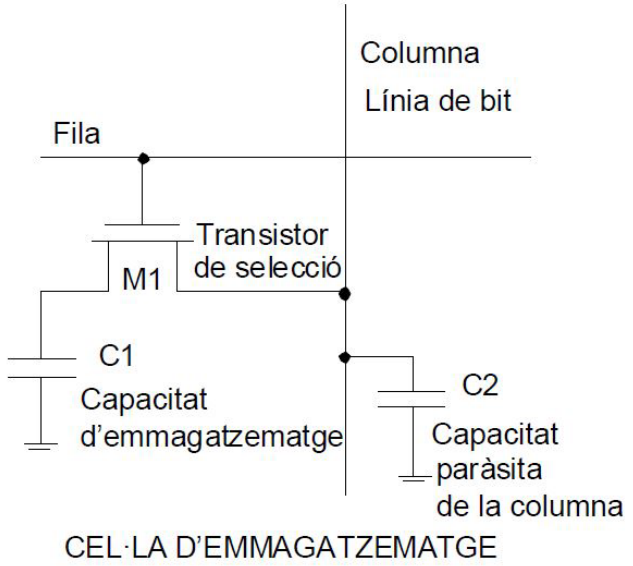
Sending an address to several Banks, there is just one Access time and then accesses rotate between the four Banks to supply four times the bandwidth. This rotating Access scheme is called *address interleaving*.

SRAM vs. DRAM

SRAM	Manté dades mentre hi hagi Vcc Cel·la gran (6 – 8 trans.) Baixa capacitat Ràpida Cost elevat	Mem. Dades baixa Capac. Mem. Caches (L1, L2 i L3)
DRAM	Necessita refresc periòdic (Controlador) Cel·la petita (1-3 trans) Alta Capacitat (densitat) Lenta Cost baix	Mem. Dades alta Capac. Mem. Principal







Bit size Registers

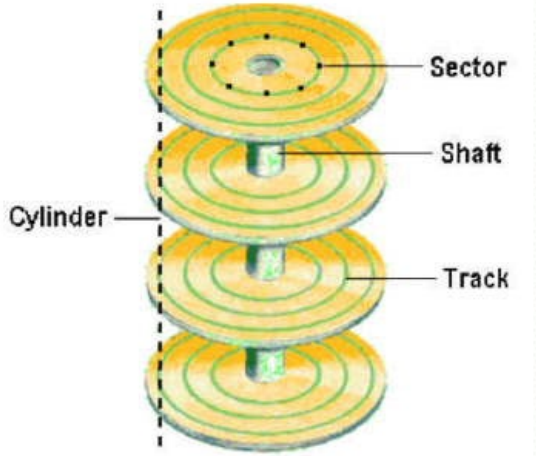
Bit size SRAM

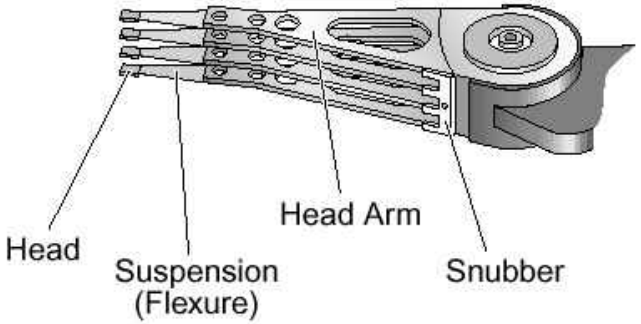
Bit size DRAM

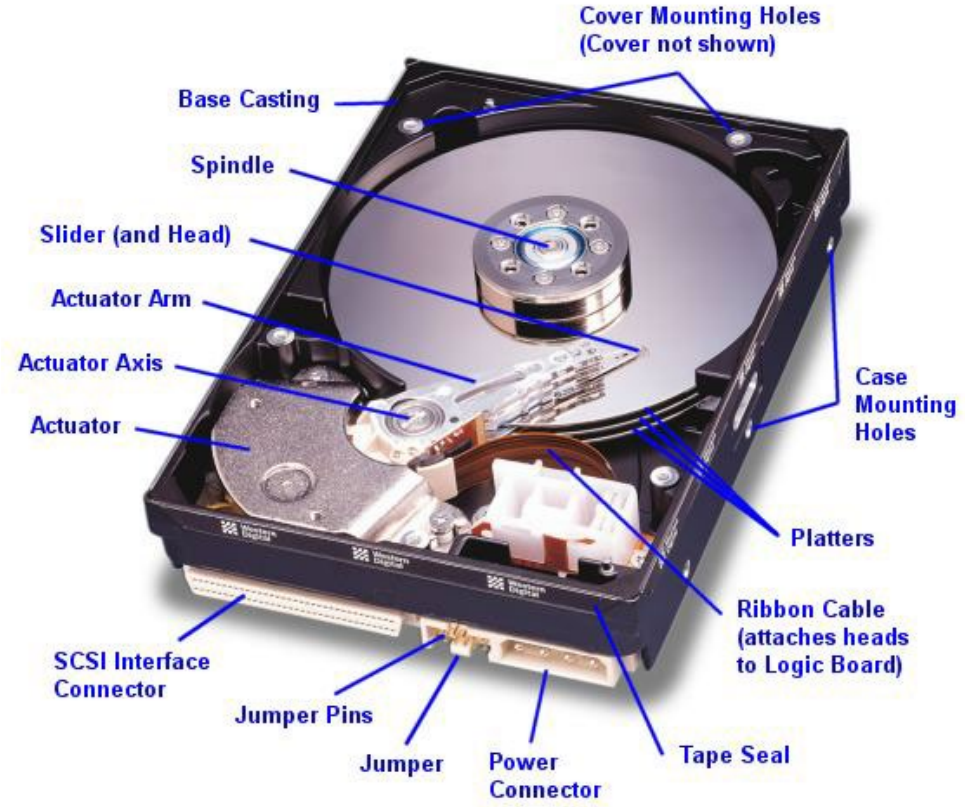
Memory Type	Transistors per Bit Cell	Latency	
Flip-flop	~20	Fast	Registres
SRAM	6	Medium	Cache
DRAM	1	Slow	Memoria principal

Disc dur HDD (Hard disk drive)

Dispositiu d'emmagatzematge electro-mecànic







https://www.youtube.com/watch?v=S1JtsstYcLs&t=54s

Cluster: És la unitat mínima de informació utilitzada pel S.O.
La seva grandària oscil.la entre 1 i 64 sectors (32KB). Els sectors tenen una mida de entre 512 – 4096 bytes

Un fitxer és una seqüència de clústers. Ocupa un nº sencer de clusters encara que algun estigui parcialment ple.

Flash and Disk Memories

Disc dur SSD (Solid-State Drive) és un tipus de dispositiu d'emmagatzematge de dades que fa servir memòria del tipus FLASH (no volàtil) per guardar les dades.

Les memòries FLASH fonamenten el seu funcionament en les memòries EEPROM

Què és una memòria EEPROM?

(Electrical Erasable Programable ROM) és una memòria que pot ser programada, esborrada i tornada a re-programar "tantes vegades" com sigui necessari.

AVANTATGES

- Més ràpides que el disc dur magnètic
- Més silencioses
- Més lleugeres

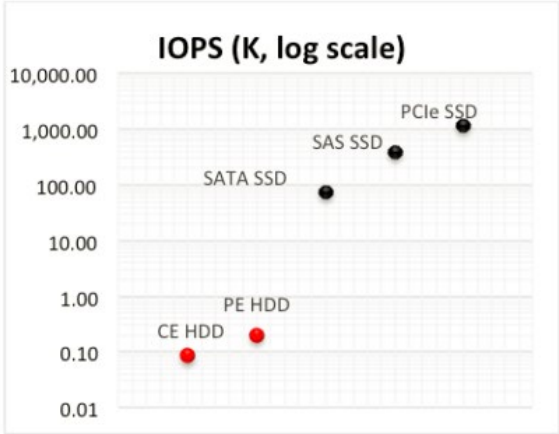
DESAVANTATGES

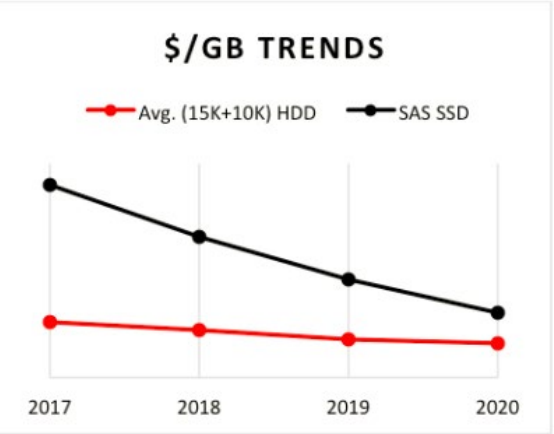
- Més cares
- Menys capacitat

El disc dur SSD vs HDD

Attribute	SSD (Solid State Drive)	HDD (Hard Disk Drive)
Power Draw / Battery Life	Less power draw, averages 2 – 3 watts, resulting in 30+ minute battery boost ✓	More power draw, averages 6 – 7 watts and therefore uses more battery
Cost	Expensive, roughly $0.20 per gigabyte (based on buying a 1TB drive)	Only around $0.03 per gigabyte, very cheap (buying a 4TB model) ✓
Capacity	Typically not larger than 1TB for notebook size drives; 4TB max for desktops	Typically around 500GB and 2TB maximum for notebook size drives; 10TB max for desktops ✓
Operating System Boot Time	Around 10-13 seconds average bootup time ✓	Around 30-40 seconds average bootup time
Noise	There are no moving parts and as such no sound ✓	Audible clicks and spinning can be heard
Vibration	No vibration as there are no moving parts ✓	The spinning of the platters can sometimes result in vibration
Heat Produced	Lower power draw and no moving parts so little heat is produced ✓	HDD doesn't produce much heat, but it will have a measurable amount more heat than an SSD due to moving parts and higher power draw

Attribute	SSD (Solid State Drive)	HDD (Hard Disk Drive)
Failure Rate	Mean time between failure rate of 2.0 million hours ✓	Mean time between failure rate of 1.5 million hours
File Copy / Write Speed	Generally above 200 MB/s and up to 550 MB/s for cutting edge drives ✓	The range can be anywhere from 50 – 120MB / s
Encryption	Full Disk Encryption (FDE) Supported on some models ✓	Full Disk Encryption (FDE) Supported on some models ✓
File Opening Speed	Up to 30% faster than HDD ✓	Slower than SSD
Magnetism Affected?	An SSD is safe from any effects of magnetism ✓	Magnets can erase data





MEMÒRIA CACHE (CAU)

MEMÒRIA CACHE

-Nivell superior de les memòries
-Recomanable en aquelles aplicacions que precisen d'un accés molt ràpid de la informació
-Capacitat petita (16-512KB-8MB). Del tipus SRAM
-Tecnologia de fabricació bipolar
-Mode d'accés associatiu (camp d'etiqueta i camp de dades)
-Quan la paraula donada no coincideix amb el valor de cap etiqueta es produeix

Miss ➔ La cache no disposa de la dada demanada. S'ha de buscar a la memòria principal

Cache Terminologia

- **Capacitat (C):**
  - Nombre de bytes de la memòria cache
- **Mida de bloc (*b*):**
  - bytes de dades introduïts a la memòria cache alhora (estructurats en *b words*)
- **Nombre de blocs ($B = C/b$):**
  - Nombre total de blocs a la cache
- **Grau d'associabilitat (N):**
  - Nombre de blocs associats a un set (índex)
- **Nombre de set (índex) of sets ($S = B/N$):**
  - Cada direcció de memòria mapeja a un set (índex) de la cache

La memòria CACHE es troba actualment integrada en la CPU.

Necessitem trobar resposta a dues preguntes:

1.- La dada que busquem està a la cache?

2.- Si està, com la trobem?

Tenim tres tipus d'estructuració de memòries cache:

a) **Mapeig directe:** 1 bloc a cada set

b) **Totalment associatives**: només té un set on van tots els blocs

c) **Associatives per conjunts (ways)**: N blocs a cada set

Organization	Number of Ways (N)	Number of Sets ($S = B/N$)
Direct Mapped	**1**	B
N-Way Set Associative	$1 < N < B$	B / N
Fully Associative	B	**1**

El mètode més simple d'entendre és el mapeig directe. Els altres mètodes s'expliquen en EC

Mapeig directe: 1 bloc a cada set

La memòria cache té tres parts importants:

a) **Etiqueta (tag)**
b) **Bloc de memòria.**
   Un bloc de memòria pot estat constituït per b *words*
c) **Índex (set)**

Un exemple senzill seria el següent: Tenim una cache que té **8 blocs**. Cada bloc està format per **1 word**. El bus d'adreces és de **32 bits**. Com serà la nostra cache?

B= 8 blocs
b= 1 word

C= *B* ·*b*= 8 words

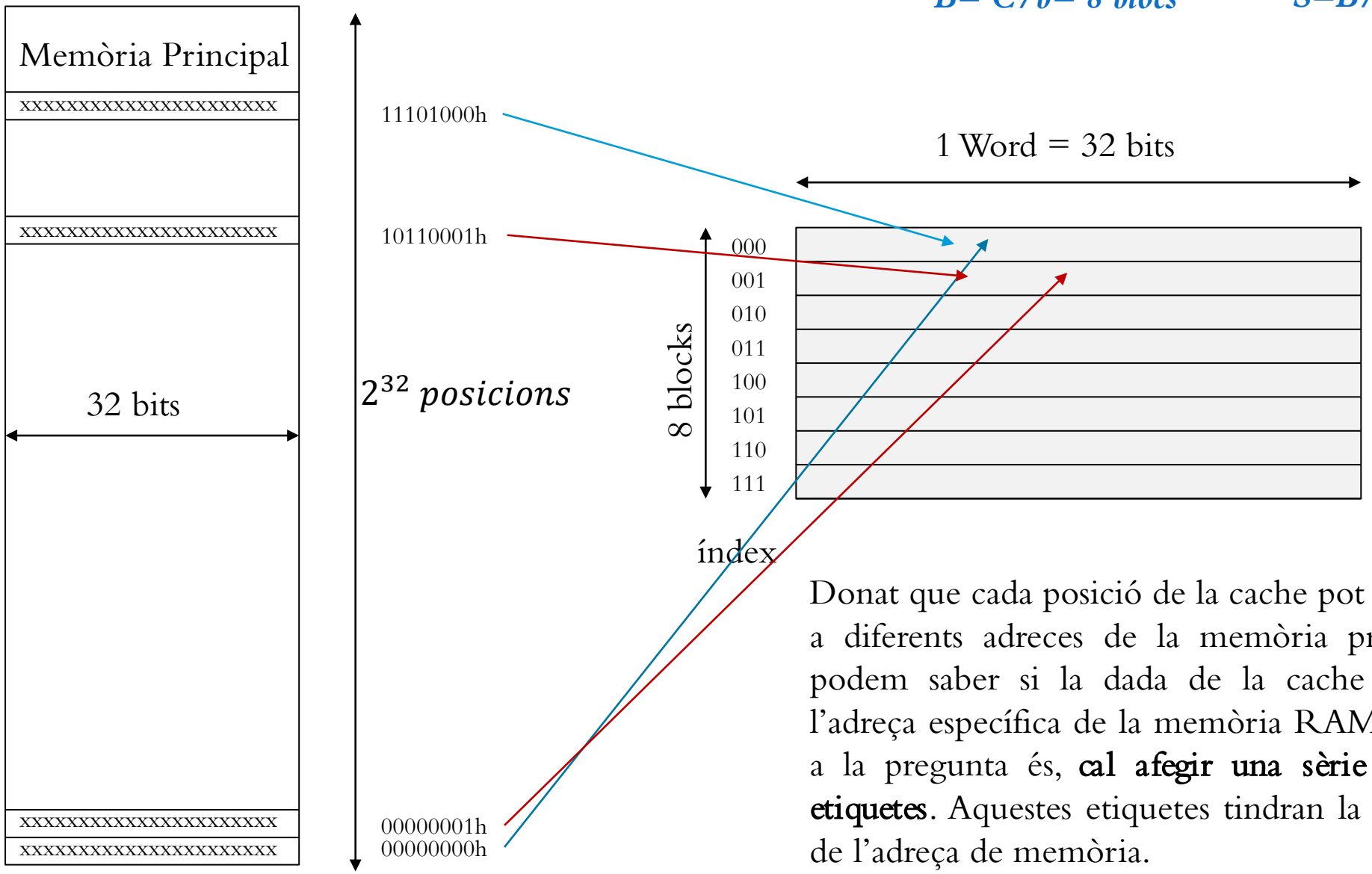
N=1 way (mapeig directe)

S=B/N= 8 sets

MEMÒRIA CACHE

Mapeig directe C= 8 words *b= 1 word* *N=1*

B= C/b= 8 blocs *S=B/N= 8 sets*



Donat que cada posició de la cache pot correspondre a diferents adreces de la memòria principal, com podem saber si la dada de la cache correspon a l'adreça específica de la memòria RAM? La resposta a la pregunta és, **cal afegir una sèrie de TAGS o etiquetes**. Aquestes etiquetes tindran la part superior de l'adreça de memòria.

MEMÒRIA CACHE

Exemple de Mapeig directe:

C= 8 words *b= 1 word* *N=1*

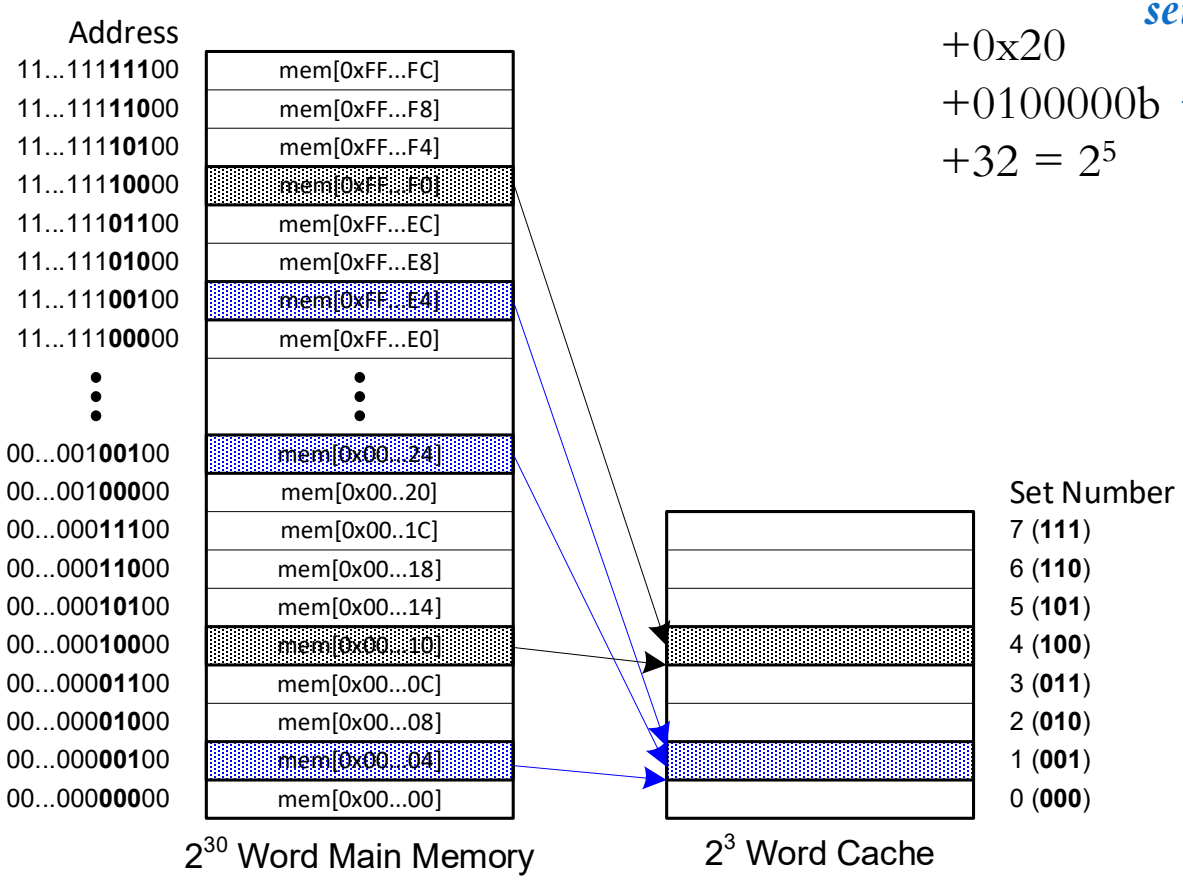
B= C/b= 8 blocs *S=B/N= 8 sets*

+0x04

set

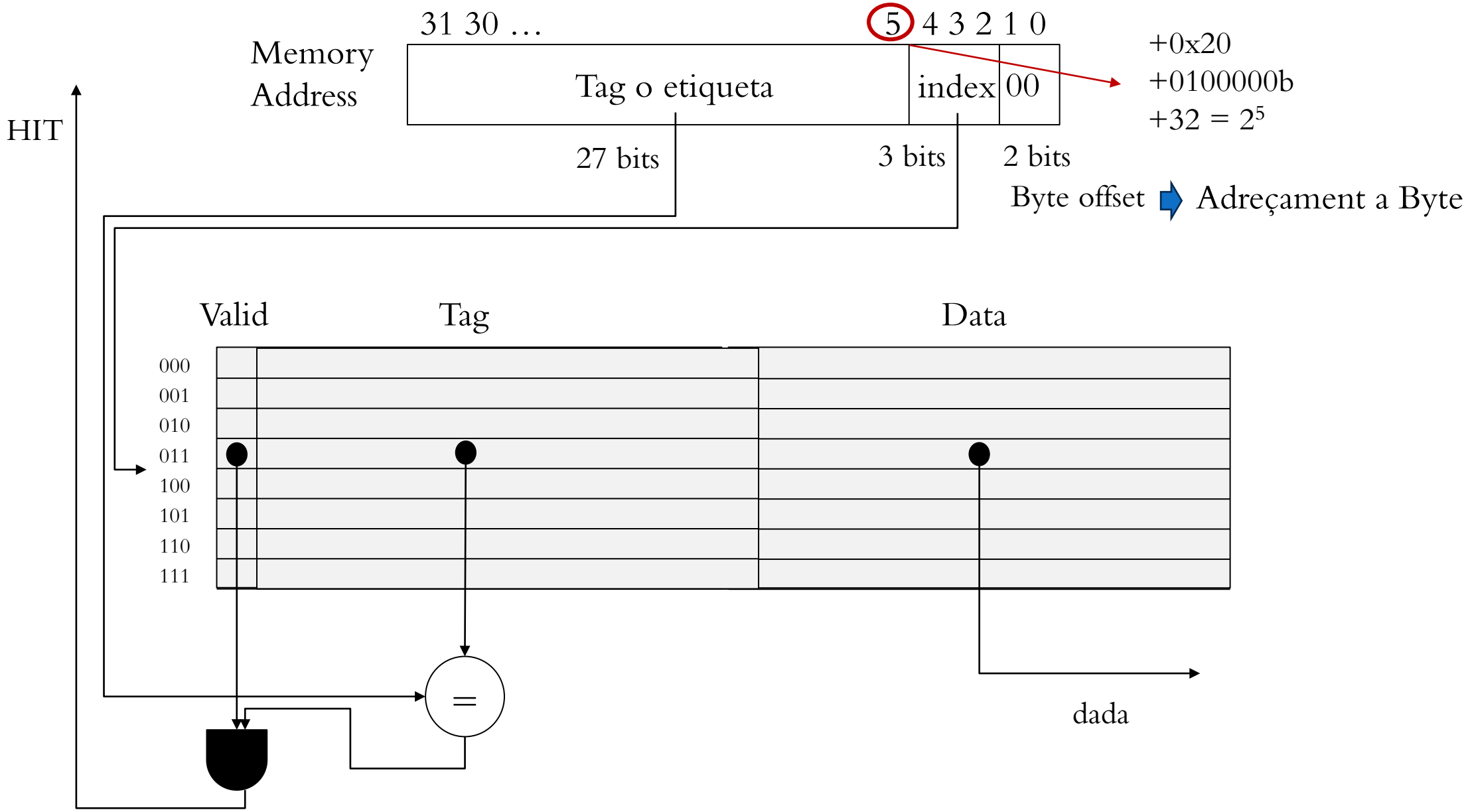
+0x20
+0100000b
+32 = 2^5

0 (000)	1 (001)	2 (010)	3 (011)	4 (100)	5 (101)	6 (110)	7 (111)
[0x..00]	[0x..04]	[0x..08]	[0x..0c]	[0x..10]	[0x..14]	[0x..18]	[0x..1c]
[0x..20]	[0x..24]			[0x..30]	[0x..34]		
[0x..40]	[0x..44]			[0x..50]	[0x..54]		
[0x..60]	[0x..64]			[0x..70]	[0x..74]		
[0x..80]	[0x..84]			[0x..90]	[0x..94]		
[0x..A0]	[0x..A4]			[0x..B0]	[0x..B4]		
[0x..C0]	[0x..C4]			[0x..D0]	[0x..D4]		



MEMÒRIA CACHE

Mapeig directe C= 8 words *b= 1 word* *B= C/b= 8 blocs* *N=1* *S=B/N= 8 sets*



MEMÒRIA CACHE

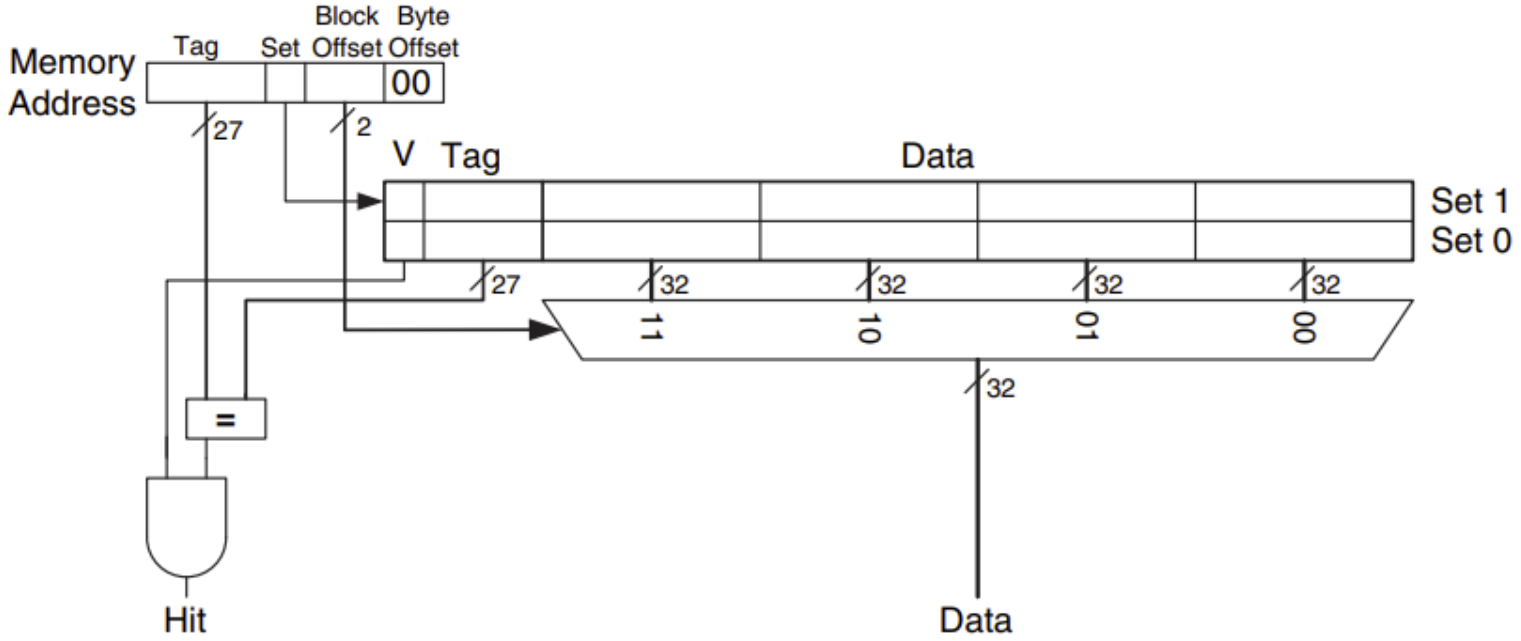
Mapeig directe *b > 1*

C= 8 words

b= 4 word Spatial Locality

B= C/b= 2 blocs

N=1 *S=B/N= 2 sets*



set	0				1			
Block offset	0	1	2	3	0	1	2	3
	[0x..00]	[0x..04]	[0x..08]	[0x..0c]	[0x..10]	[0x..14]	[0x..18]	[0x..1c]
	[0x..20]	[0x..24]			[0x..30]	[0x..34]		
	[0x..40]	[0x..44]			[0x..50]	[0x..54]		
	[0x..60]	[0x..64]			[0x..70]	[0x..74]		
	[0x..80]	[0x..84]			[0x..90]	[0x..94]		
	[0x..A0]	[0x..A4]			[0x..B0]	[0x..B4]		
	[0x..C0]	[0x..C4]			[0x..D0]	[0x..D4]		

N-Way Set Associative Cache

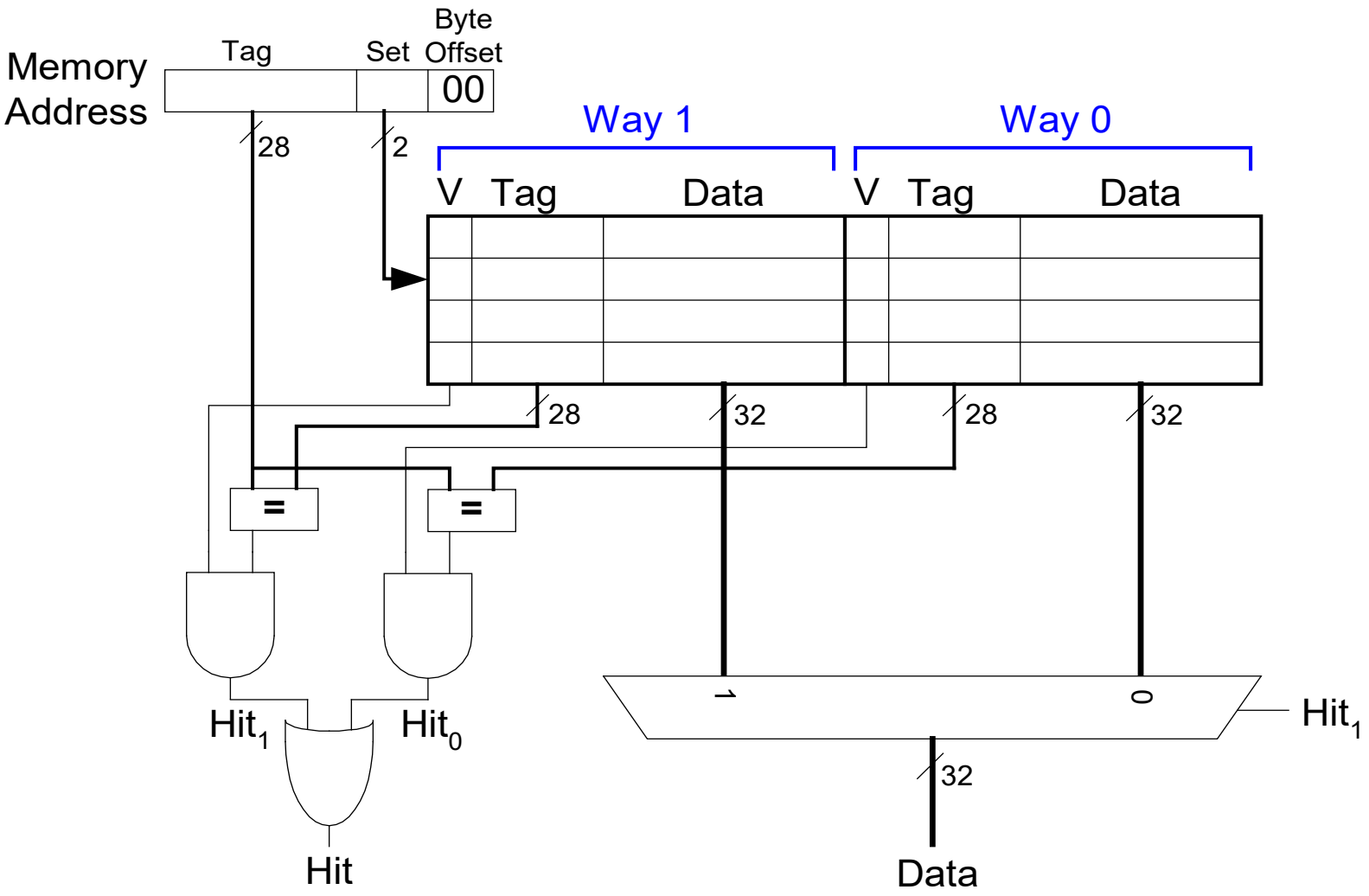
Mapeig 2-Way

C= 8 words

b= 1 word

B= C/b= 8 blocs

N=2 *S=B/N= 4 sets*



Fully Associative Cache

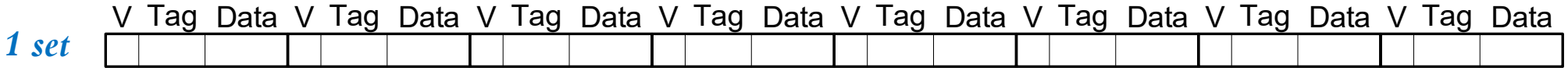
Mapeig totalment associatiu

C= 8 words

b= 1 word

B= C/b= 8 blocs

N=B *S=B/N= 1 set*



EFICIÈNCIA DE LA CACHÉ

És la relació entre el temps d'accés i el temps d'accés mig. Aquesta eficiència dependrà d'uns algoritmes que s'utilitzen per carregar-la amb la informació necessària per la UCP.

Tenim dos paràmetres que influeixen en la eficiència:

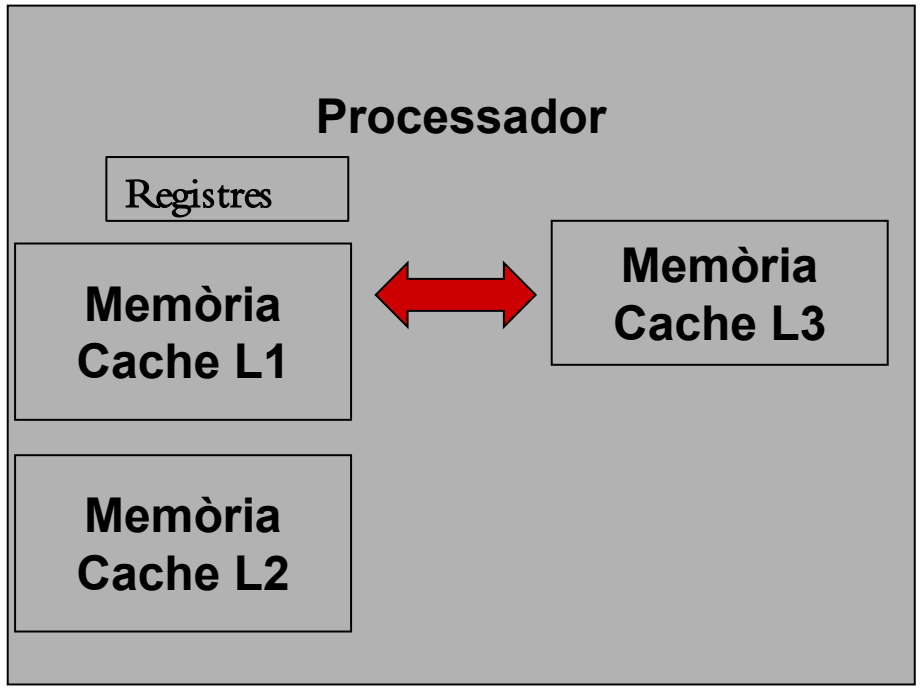
La probabilitat de presència Hit Rate (HR)= # hits / # memory accesses= 1 – Miss Rate

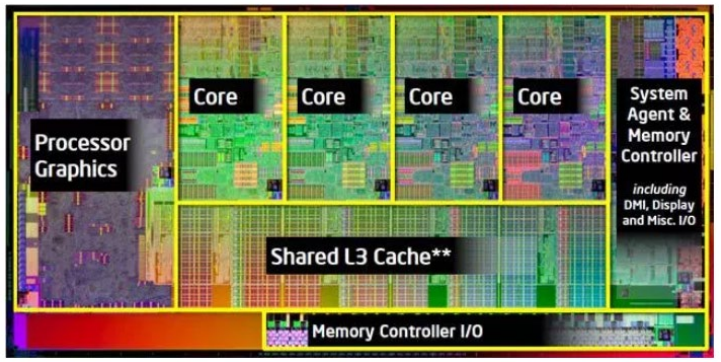
La probabilitat de absència Miss Rate (MR) = # misses / # memory accesses= 1 – Hit Rate

Average memory access time: AMAT = $t_{cache} + MR_{cache}[t_{MM} + MR_{MM}(t_{VM})]$

Un altre paràmetre que sol donar-se és el ***factor de velocitat***, que és la relació entre les velocitat d'accés a la cache i a la RAM, que és l'invers de la relació entre els temps d'accés

MEMÒRIA CACHE





Caché L1. Integrada en la CPU. Es fa servir per accedir a dades importants i de ús freqüent. Totes les instruccions es busquen primer aquí

Caché L2. Utilitzada per guardar informació utilitzada recentment. També coneguda com caché secundaria, està dissenyada per reduir el temps d'accés a les dades utilitzades prèviament. Es fa servir també per fer pipeline temporal d'instruccions.

Caché L3. Abans la Memòria L3 es trobava integrada en la placa base. Es fa servir per alimentar la memòria caché L2

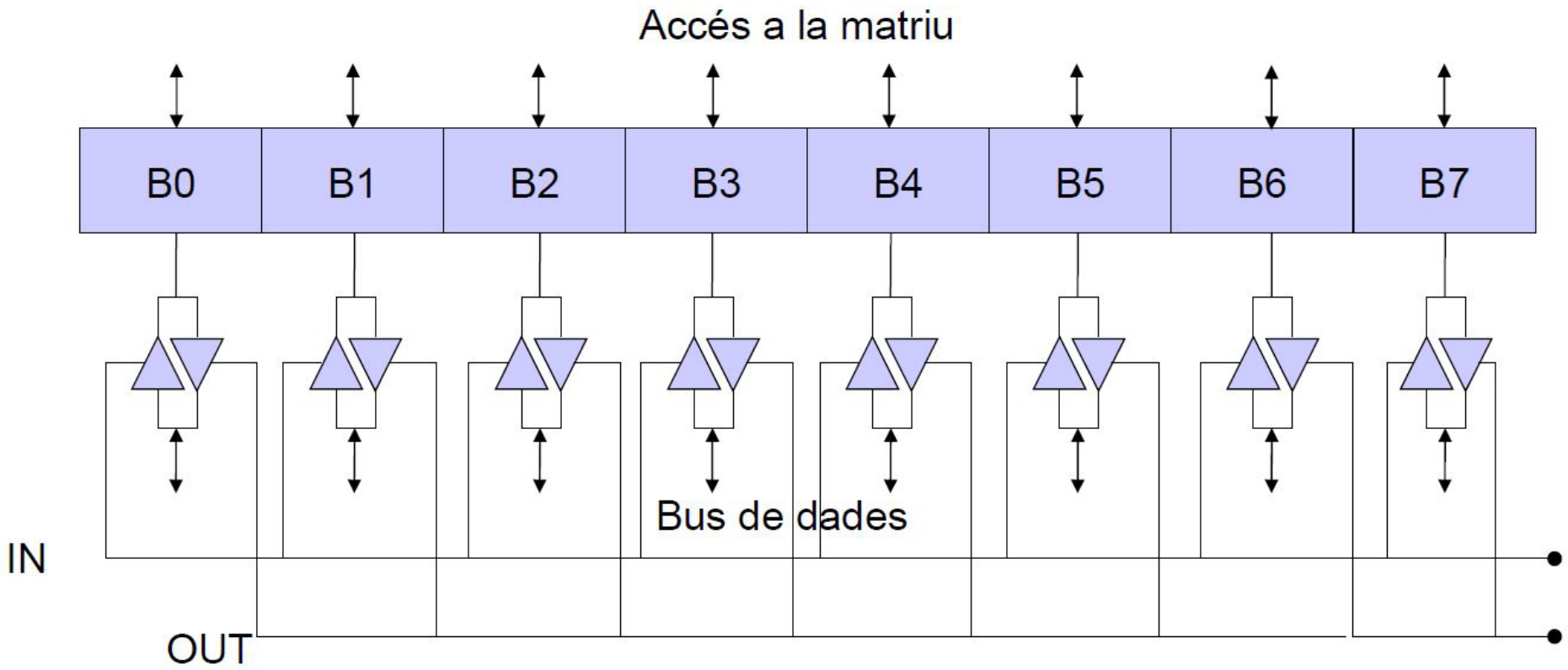
La memòria Principal ("RAM")

ESTRUCTURA DE LA MEMÒRIA

- Organització interna de la memòria
  - Matriu de memòria: s'organitza en files i columnes
  - Un o dos descodificadors per seleccionar les paraules guardades a la matriu de cel·les
  - Un registre d'entrada/sortida format per un nº de biestables igual a la longitud de paraula o dada guardat a la posició de memòria (Opcional. Depen de la memòria). Els buffers tri-state que té integrats determinen si la paraula és de lectura o escriptura
  - Lògica de control. A partir dels senyals externs del bus (CS, R/W, etc) genera els senyals de govern intern

Entrades		Sortides	
CS	\R/W	IN	OUT
0	X	0	0
1	0	0	1
1	1	1	0

ESTRUCTURA DE LA MEMÒRIA



ESTRUCTURA DE LA MEMÒRIA

La distribució vista fins ara de la memòria és INEFICIENT Detalls a considerar:

-S'intenta crear una estructura matricial el més quadrat i petit possible

-Dividim les línies del decodificador en dos tipus:

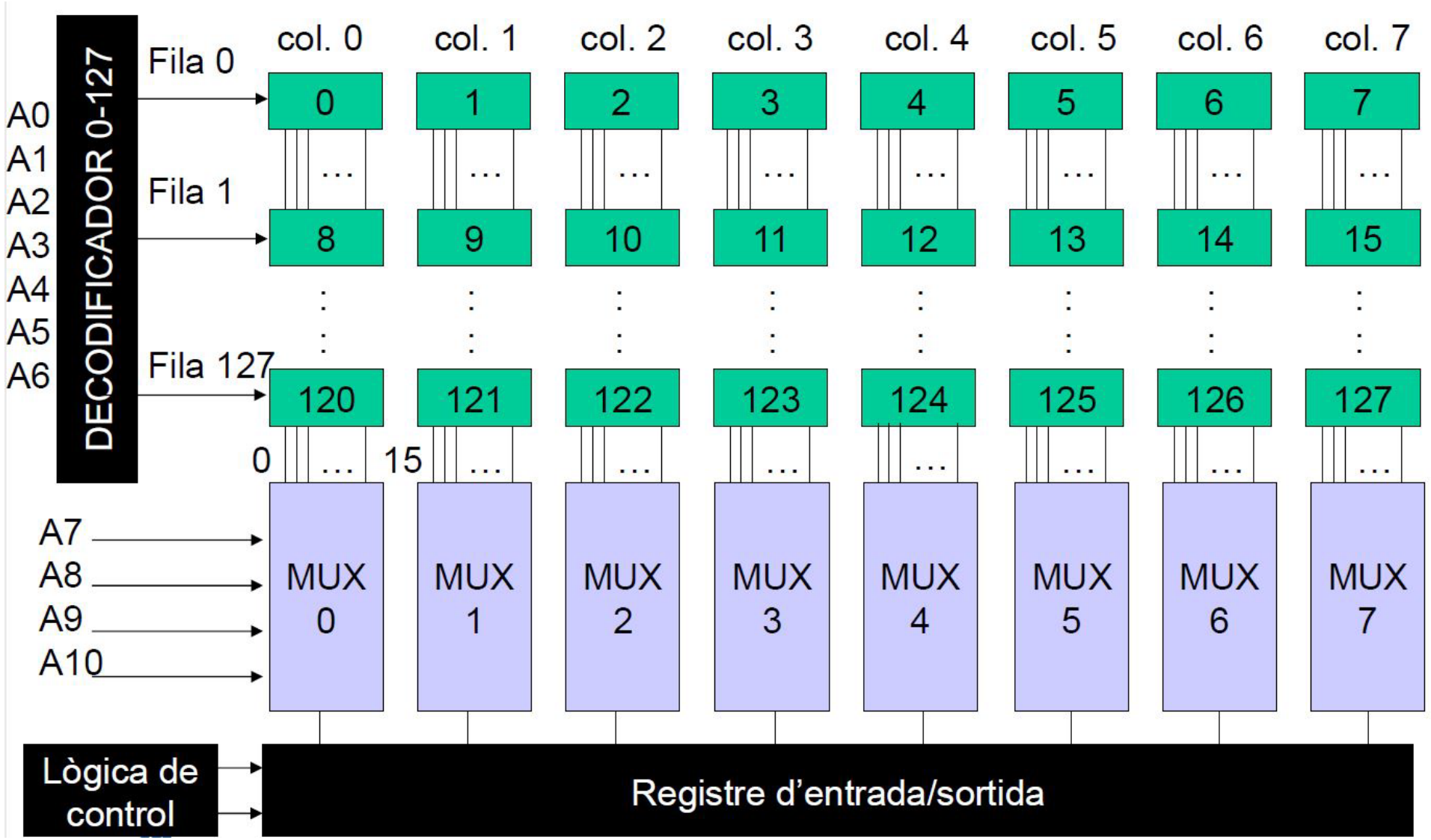
a) Decodificador de files, format per les línies baixes del bus de direccions

b) Decodificador de columnes, format per les línies altes del bus de direccions

EXEMPLE: MEMÒRIA RAM DE 2KBytes

ESTRUCTURA DE LA MEMÒRIA

EXEMPLE: MEMÒRIA RAM DE 2KBytes

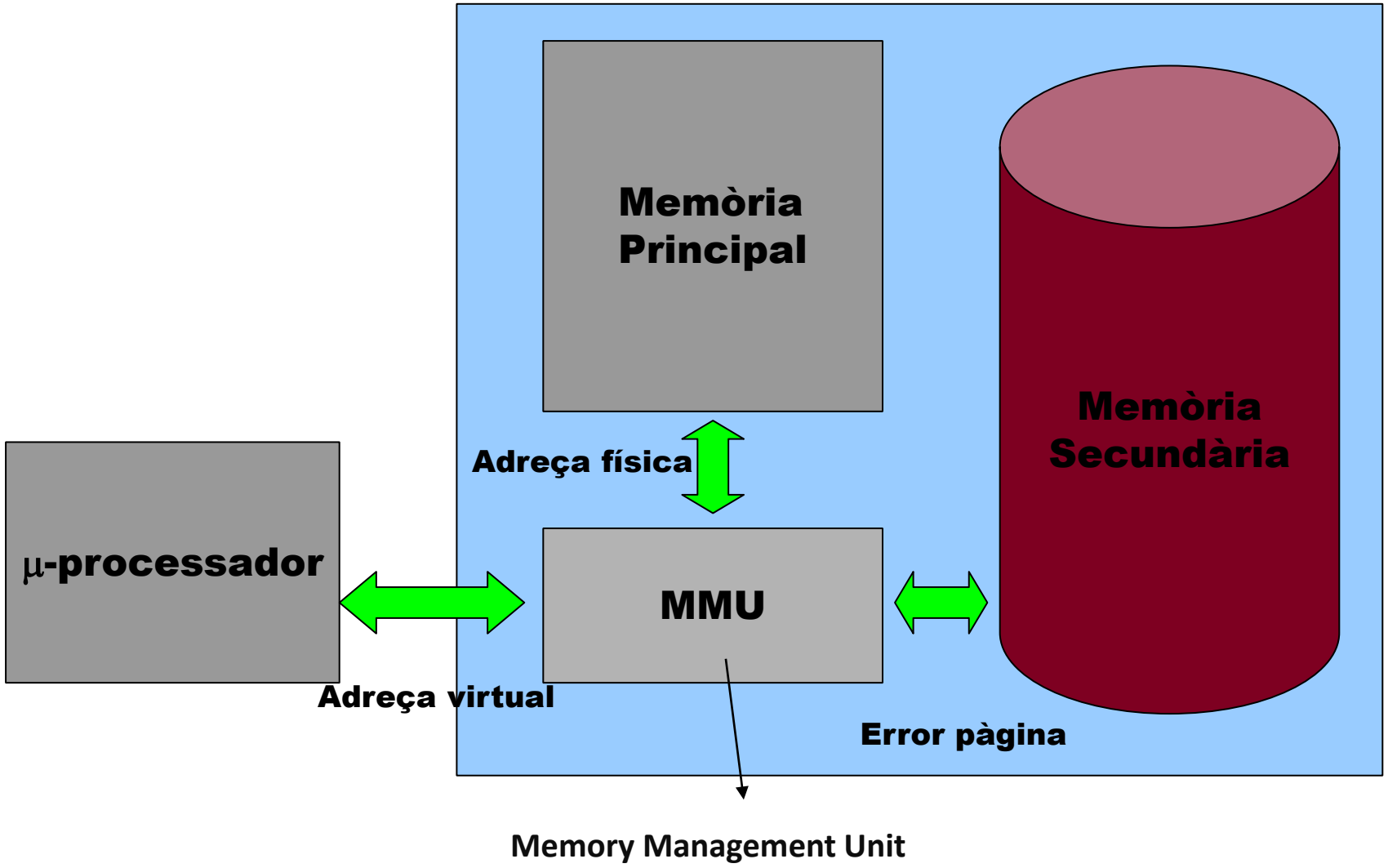


ESTRUCTURA DE LA MEMÒRIA

- Les 7 línies de menys pes del bus de direccions es decodifiquen amb el decodificador de files, seleccionant una de les 128 files possibles
- Les 4 línies restants, de A7 a A10 es decodifiquen amb el decodificador de columnes format per 8 MUX de 16 entrades de les que seleccionem 8, 1 per cada MUX.
- Amb aquest tipus d'estructura, les línies del bus de direccions es poden repartir de diferents formes en funció del fabricant.

MEMÒRIA SECUNDARIA
I
MEMÒRIA VIRTUAL

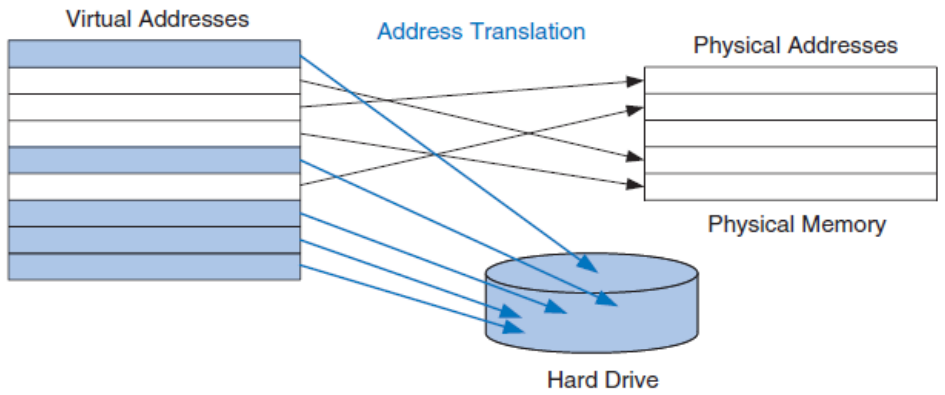
Jerarquia de memòria: Memòria Virtual



Memòria Virtual

Physical memory acts as cache for virtual memory

Cache	Virtual Memory
Block	Page
Block Size	Page Size
Block Offset	Page Offset
Miss	Page Fault
Tag	Virtual Page Number



- **Page size:** amount of memory transferred from hard disk to DRAM at once
- **Address translation:** determining physical address from virtual address
- **Page table:** lookup table used to translate virtual addresses to physical addresses

Jerarquia de memòria: Memòria Virtual

- La memòria virtual és una tècnica d'administració de la memòria real que permet al SO proporcionar al sw d'usuari un espai d'adreces superior a la memòria física que tenim.
- La memòria virtual fa servir dos nivells de jerarquia de memòria: La memòria principal i la memòria de recolzament (memòria secundaria: Disc Dur)
- La gestió de la memòria virtual demanda una gestió automàtica que es fa per HW.
- Les fallades de pàgines són ateses pel SO. Per tant es fa per SW. El procés de migració per atendre les fallades de pàgines es denomina **paginació**

Jerarquia de memòria: Memòria Virtual

- El mapa virtual associat a un programa en execució esta suportat físicament per una zona de memòria principal i una zona del disc anomenada zona d'intercanvi o swap
- El microprocessador genera adreces virtuals
- Tot i que el programa genera adreces virtuals, per poder executar-se ha de residir a la MP
- Si una dada no es troba a la MP, s'haurà de importar de la zona de swap per tal que el programa pugui executar-se correctament
- L'espai virtual i físic es divideix en pàgines. Pàgines virtuals (memòria virtual), pàgines d'intercanvi (swap) i marc de pàgina (MP)

Prestacions dels Ordinadors

Prestacions dels Ordinadors

Els paràmetres característics necessaris per caracteritzar les prestacions dels ordinadors són:

1. Ample de paraula: nº bits que fa servir el uP en paral·lel.
2. Memòria: Indica la mida de la memòria principal del uP
3. Memòria auxiliar: Expressa en Mbytes o Gbytes la mida dels perifèrics tipus disc que contingui el uP.
4. Ample de banda: Cabal d'informació capaç de transmetre un bus a una unitat
5. MIPS: (milions de instruccions per segon).Velocitat d'execució de les instruccions de la màquina
6. MFLOPS (milions d'operacions en coma flotant per segon).Velocitat de càlcul científic d'un computador

Exemple prestacions

	Clarkdale - 32nm		Lynnfield - 45nm		Bloomfield - 45nm
Modelos	Core i3	Core i5 6xx	Core i5 7xx	Core i7 8xx	Core i7 9xx
Velocidad	2.93 – 3.06 Ghz	3.2 – 3.46 Ghz	2.66 Ghz	2.8 Ghz	2.66 – 3.33 Ghz
Núcleos / Procesos Memoria Caché	2/4 4MB	4/4 4MB	4/4 8MB	4/8 8MB	4/8 8MB

Bandwidth Comparison					
	Bus Clock	Internal Rate	Prefetch	Transfer Rate	Channel Bandwidth
DDR	100-200 MHz	100-200 MHz	2n	0.20-0.40 GT/s	1.60-3.20 GBps
DDR2	200-533 MHz	100-266 MHz	4n	0.40-1.06 GT/s	3.20-8.50 GBps
DDR3	400-1066 MHz	100-266 MHz	8n	0.80-2.13 GT/s	6.40-17.0 GBps
DDR4	1066-2133 MHz	100-266 MHz	8n	2.13-4.26 GT/s	12.80-25.60 GBps

DRAM memory

MIPS i MFLOPS

Calculem els MIPS a partir del nombre total d'instruccions i del temps que triguen en executar-se

$$MIPS = \frac{N^{o}\ total\ de\ instruccions}{temps\ que\ triga} \times 10^{-6}$$

Donat que el temps depèn de la freqüència del sistema, tindrem

$$temps\ que\ triga = N^{o}\ cicles\ clk \times temps\ de\ cicle = \frac{\langle N^{o}\ cicles\ clk \rangle}{Freqüència}$$

$$\Rightarrow \quad MIPS = \frac{N^{o}\ total\ de\ instruccions \times Freqüència}{\langle N^{o}\ cicles\ de\ clk \rangle}$$

MIPS i MFLOPS

De les expressions anteriors podem extreure un paràmetre clau: Els Cicles Per Instrucció

$$MIPS = \frac{Freqüència}{CPI} \times 10^{-6} \qquad CPI = \frac{Nºcycles}{Nºinstructions}$$

Els MIPS permeten calcular el temps que triga en executar-se un determinat programa amb un determinat nombre d'instruccions. El temps d'execució pot calcular-se com:

$$t_{exec} = \frac{Nº\ total\ d'instruccions \times CPI}{Freqüència} = \frac{Nº\ total\ d'instruccions}{MIPS} \times 10^{-6}$$

Exemple

Un programa consta de 140 instruccions, de les quals 70 triguen en executar-se 4 cicles, 35 triguen en executar-se 5 cicles, 20 triguen en executar-se 3 cicles i les 15 restants triguen 7 cicles

1.- Calculeu el CPI promig per aquest programa

2.- Si l'ordinador funciona a una freqüència de 20 MHz, calculeu el temps que triga en executar el programa

Exemple

$$t_{exec} = \frac{N^o\ total\ d'instruccions \times CPI}{Freqüència} = \frac{N^o\ total\ d'instruccions}{MIPS} \times 10^{-6}$$

Un programa consta de 140 instruccions, de les quals 70 triguen en executar-se 4 cicles, 35 triguen en executar-se 5 cicles, 20 triguen en executar-se 3 cicles i les 15 restants triguen 7 cicles

1.- Calculeu el CPI promig per aquest programa

2.- Si l'ordinador funciona a una freqüència de 20 MHz, calculeu el temps que triga en executar el programa

$$CPI = \frac{Nºcycles}{Nºinstructions} = \frac{4 \cdot 70 + 5 \cdot 35 + 3 \cdot 20 + 7 \cdot 15}{70 + 35 + 20 + 15} = \frac{620}{140} = 4.4\ Cycles/I$$

$$t_{exec} = \frac{Nºinstruccions\ x\ CPI}{freqüència} = \frac{140\ x\ 4.4}{20 \cdot 10^6} = \frac{620\ cycles}{20 \cdot 10^6} = 31\ ns$$

Rendiment d'un Ordinador

Definim el rendiment que té un determinat uP per executar un programa com:

$$\eta = \frac{1}{t_{exec}} = \frac{Freqüència}{Nº\ instruccions \times CPI}$$

On el temps d'execució no és més que el Nº instruc. x Temps que triga en executar-se una instrucció.

Rendiment d'un Ordinador

El rendiment d'un uP és directament proporcional a la freqüència de treball

El rendiment d'un uP és inversament proporcional al valor del CPI

La potència de processament és inversament proporcional al nº total d'instruccions que s'han d'executar.

Exemple

Es disposa de les següents dades per dos ordinadors i una determinada aplicació:

1.- Power PC 601 de 80 MHz de freqüència i 70 Megainstruccions per segon (MIPS) de potència

2.- Pentium 120 MHz i 85 MIPS

Calculeu els CPI promig de cada processador

Si es fa servir un programa de 70 línies de codi, determina el temps d'execució, el rendiment i valora quin dels dos ordinadors presenta millor rendiment.

Augment del rendiment: Lleis de Amdahl

Les lleis d'Amdahl serveixen per evaluar l'augment de rendiment en un sistema al introduïr una millora.

Es recomana sempre millorar aquells elements que es fan servir més freqüentment, ja que són els que més influeixen en el rendiment.

Primera llei de Amdahl:

"L'augment del rendiment per una millora està limitat per el temps que s'utilitza aquesta millora"

$$t_{millorat} = t_{antic} \times \left(\frac{Fracció\ de\ temps\ millorada}{Guany\ de\ velocitat} + Fracció\ de\ temps\ no\ millorada \right)$$

Augment del rendiment: Lleis de Amdahl

Exemple:

Els dissenyadors de Intel decideixen canviar la ALU d'un uP dedicat a una tasca a la qual hi dedica el 50% del temps. La nova ALU és dos vegades més ràpida que l'anterior. Quin és el temps de millora?

$$T_{millora} = T_{anterior} \times \left\{ \left(0.5/2 \right) + 0.5 \right\} = T_{anterior} \times 0.75$$

Augment del rendiment: Lleis de Amdahl

Definim l'acceleració com:

$$Acceleració = \frac{temps\ inicial\ d'execució}{temps\ d'execució\ després\ de\ millora}$$

Imaginem que tenim una aplicació en la que el 25% de les instruccions pot millorar-se amb un factor 10 i la resta de les instruccions no es poden millorar. Quina speed-up (acceleració) es podria assolir segons la **Llei de Amdhal** ?

$$A = \frac{1}{\frac{0{,}25}{10} + 0{,}75} = \frac{1}{0{,}775} = 1{,}2903$$

Podem interpretar que la millora és del 29,03%

Augment del rendiment: Lleis de Amdahl

Segona Llei de Ambdhal

“ Quan s’introdueix una millora a un computador previament millorat, l’increment del rendiment és menor que si aquesta millora s’haguès introduït sobre el uP sense millorar”

Aquesta llei va enfocada a la variació del rendiment quan es realitzen millores succesives

Augment del rendiment: Lleis de Amdahl

Exemple

A un uP se li fa una millora a la ALU, sent 30% més ràpida que l'anterior, i el programa fa un us d'aquesta millora el 40% del temps. Posteriorment es canvia la caché substituint-la per una 4 vegades més ràpida, amb una tasa d'encerts del 80% i tenint en compte que el programa presenta un 15% del temps en accessos a memòria.

Percentatge de millora del temps:

a) Si només cambiem la caché

b) Si només substituïm la ALU

c) Si primer es canvia la ALU i després la caché

Augment del rendiment: Lleis de Amdahl

Calculem la millora de la caché (considerem un temps inicial de 100 segons):

$$t_{caché} = 100 \times \left(\frac{0.15 \times 0.8}{4} + 0.88\right) = 91\ segons$$

Calculem la millora de la ALU:

$$t_{ALU} = 100 \times \left(\frac{0.4}{1.3} + 0.6\right) = 90.77\ segons$$

Aplicant les dues millores tenim:

$$t_{ALU+caché} = 90.77 \times \left(\frac{0.15 \times 0.8}{4} + 0.88\right) = 82.6\ segons$$

Exemple

Un ordinador té 512 Mbytes de memòria RAM i un processador pentium 4 a 1 GHz de freqüència. Usualment fa servir programes que utilitzen el 35% del temps d'execució en accedir a la memòria. L'accés al disc dur ocupa el 12% del temps. Ens pregunten que és millor:

Canviar la memòria per una altra que dobla la velocitat d'accés o

Canviar el disc dur per un altre amb el quàdruple de la velocitat d'accés al disc

Què passa si fem les dues millores?