

Organización de computadoras

Clase 11

Universidad Nacional de Quilmes

Lic. Martínez Federico

¿Qué vimos?

¿Qué vimos?

- Mascaras

¿Qué vimos?

- Mascaras
- Repeticiones controladas

¿Qué vimos?

- Mascaras
- Repeticiones controladas
- Arreglos

¿Qué vimos?

- Mascaras
- Repeticiones controladas
- Arreglos
- Modo indirecto

¿Qué vimos?

- Mascaras
- Repeticiones controladas
- Arreglos
- Modo indirecto
- Q5

Y ahora?

Y ahora?

- Memorias:

Y ahora?

- Memorias:
 - Características

Y ahora?

- Memorias:
 - Características
 - Memorias ROM

Y ahora?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias

Y ahora?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:

Y ahora?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación

Y ahora?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación
 - ¿Qué?

Y ahora?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación
 - ¿Qué?
 - Organización

Y ahora?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación
 - ¿Qué?
 - Organización
 - Decisiones de implementación

Memorias

Memorias

Características

Ubicación: Internas vs externas



Volatilidad



Capacidad de Escritura



Metodo de acceso



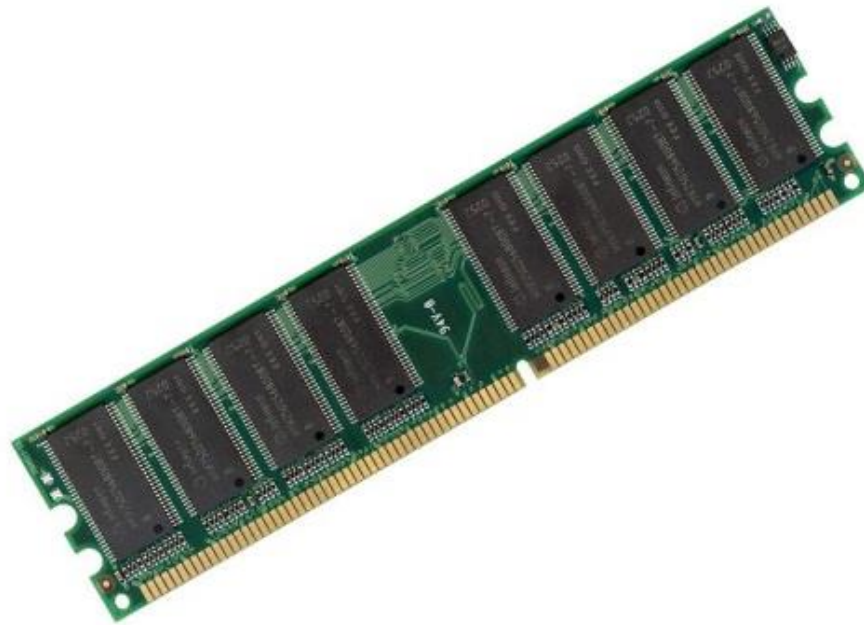
Acceso secuencial



Acceso directo



Accesso aleatorio



Acceso asociativo



Memorias de solo lectura

ROM

ROM

- Programas estáticos
- Útiles también para dispositivos como microondas, calculadoras, etc

PROM

- P es por *Programmable*
- ROMs que pueden ser grabadas por el usuario
- Se graban utilizando pulsos de altos voltajes

EPROM

- E de *Erasable*
- PROMs borrables
- Pueden ser borradas con luz ultravioleta
- EEPROM pueden ser borradas electrónicamente

Performance

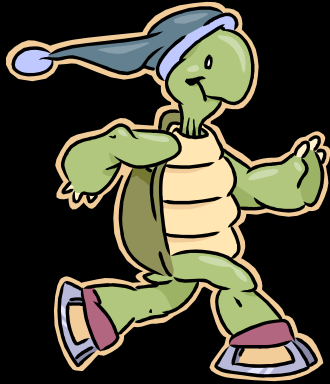
Performance

- La CPU necesita sí o sí ir a memoria para obtener la próxima instrucción

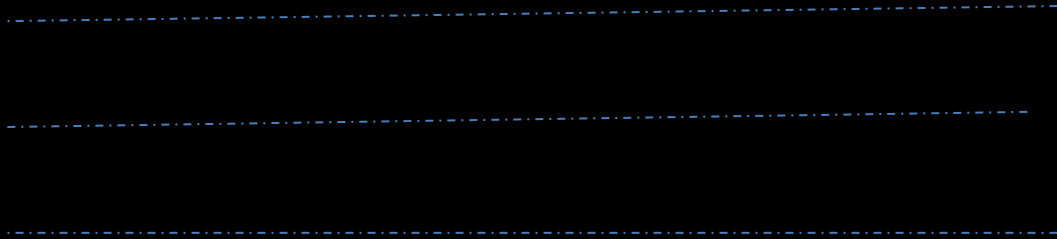
Performance

- La CPU necesita sí o sí ir a memoria para obtener la próxima instrucción
- Pero.....

Performance



Memoria



CPU

Performance

- ¿Se podría hacer algo con una memoria mas rápida pero mas chica que la principal?

YES

WWW.ENTERTAINMENTWALLPAPER.COM



Performance

- ¿Se podría hacer algo con una memoria mas rápida pero mas chica que la principal? Sí
- Los accesos en general no son al azar, sino que tienen cierto patrón

Performance

- ¿Se podría hacer algo con una memoria mas rápida pero mas chica que la principal? Sí
- Los accesos en general no son al azar, sino que tienen cierto patrón:
 - El PC suele ejecutar la instrucción que está en la celda siguiente
 - Al recorrer un arreglo se recorren sus posiciones
 - El código de una subrutina se suele ejecutar mas de una vez

Localidad
temporal





F. Bernal

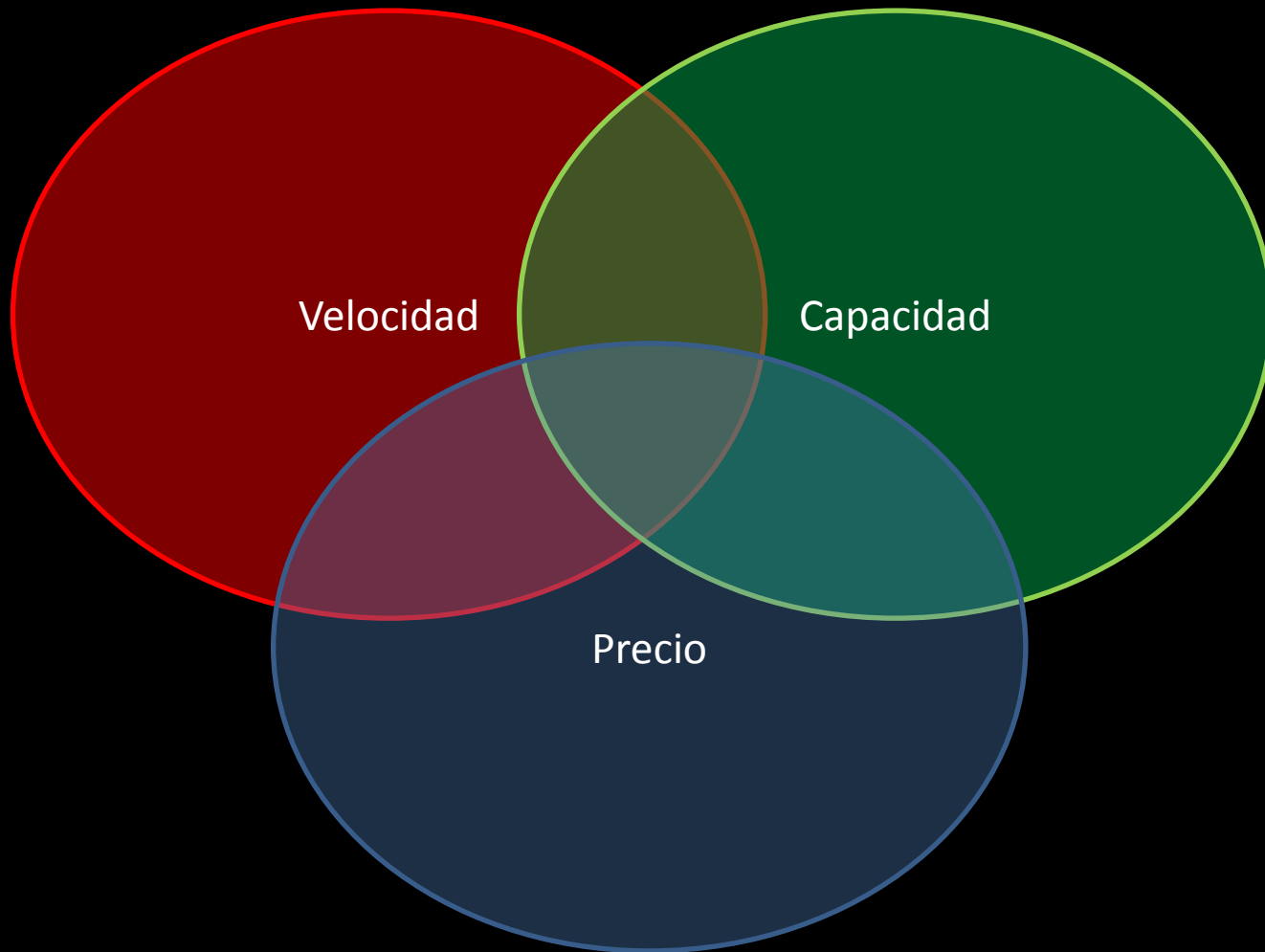
Localidad
espacial

Performance

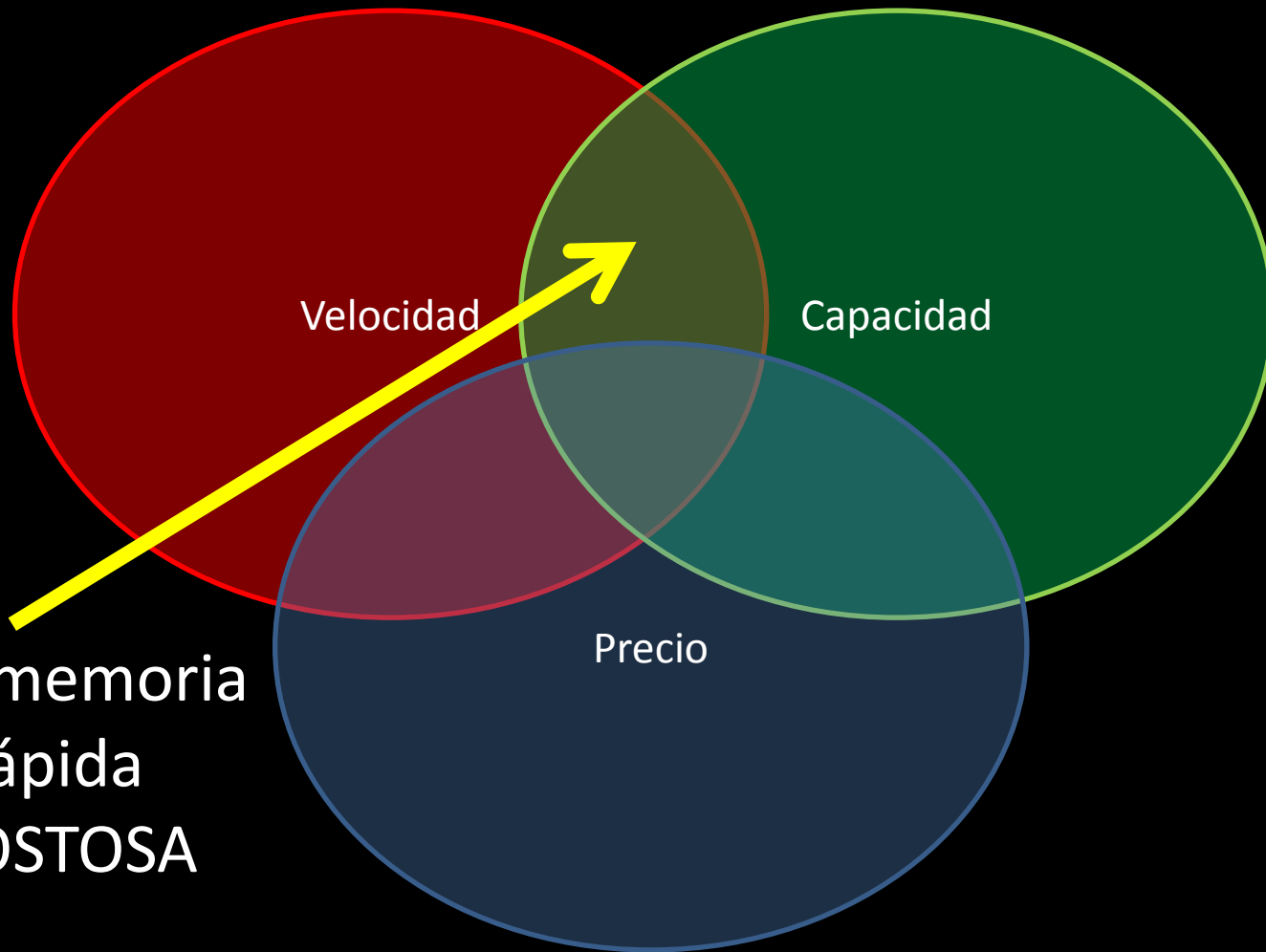
- En base a estos principios se ve que tiene sentido “tener a mano” a las posiciones recientemente usadas

Jerarquía de memorias

Jerarquía de memorias

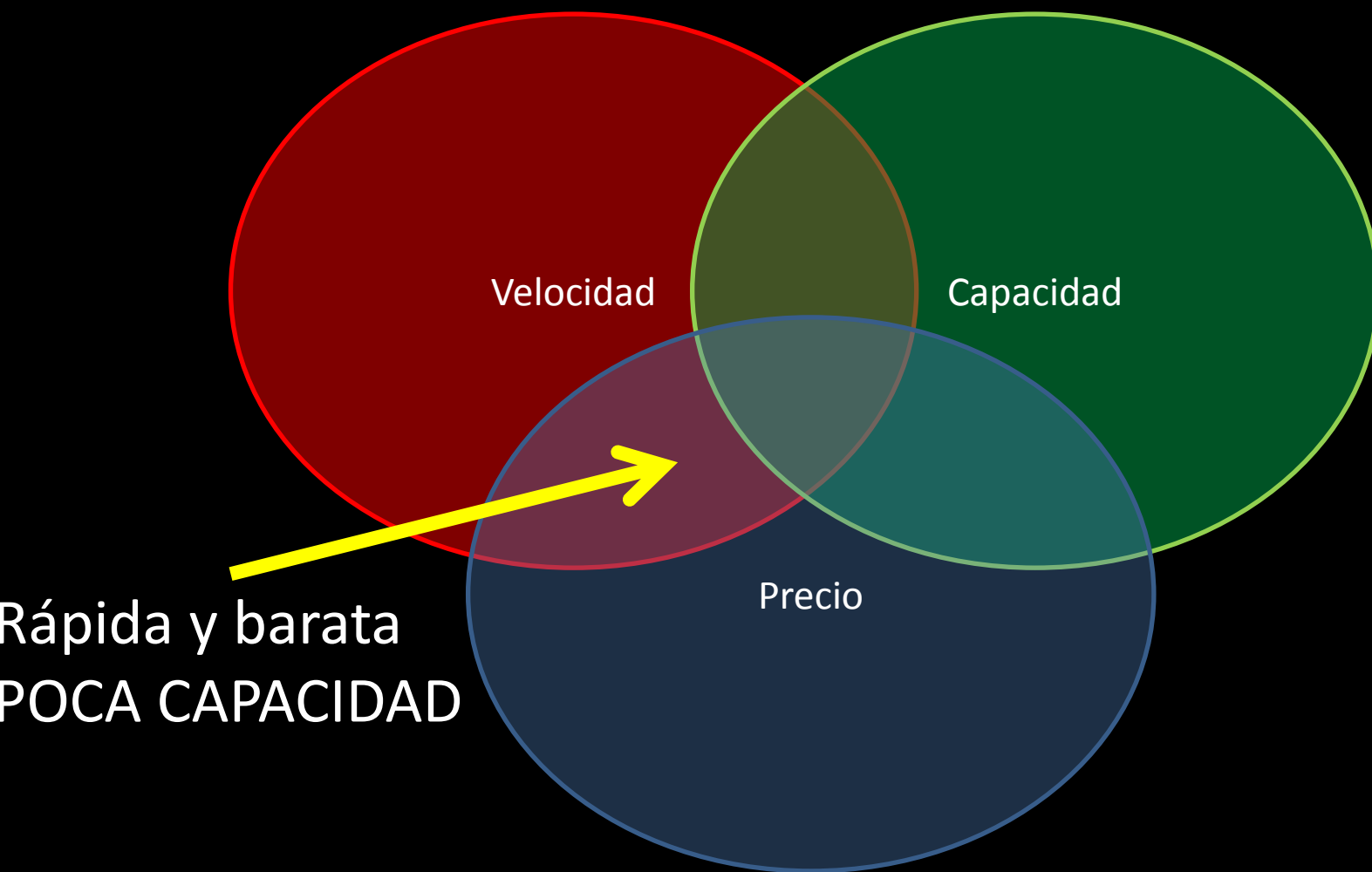


Jerarquía de memorias

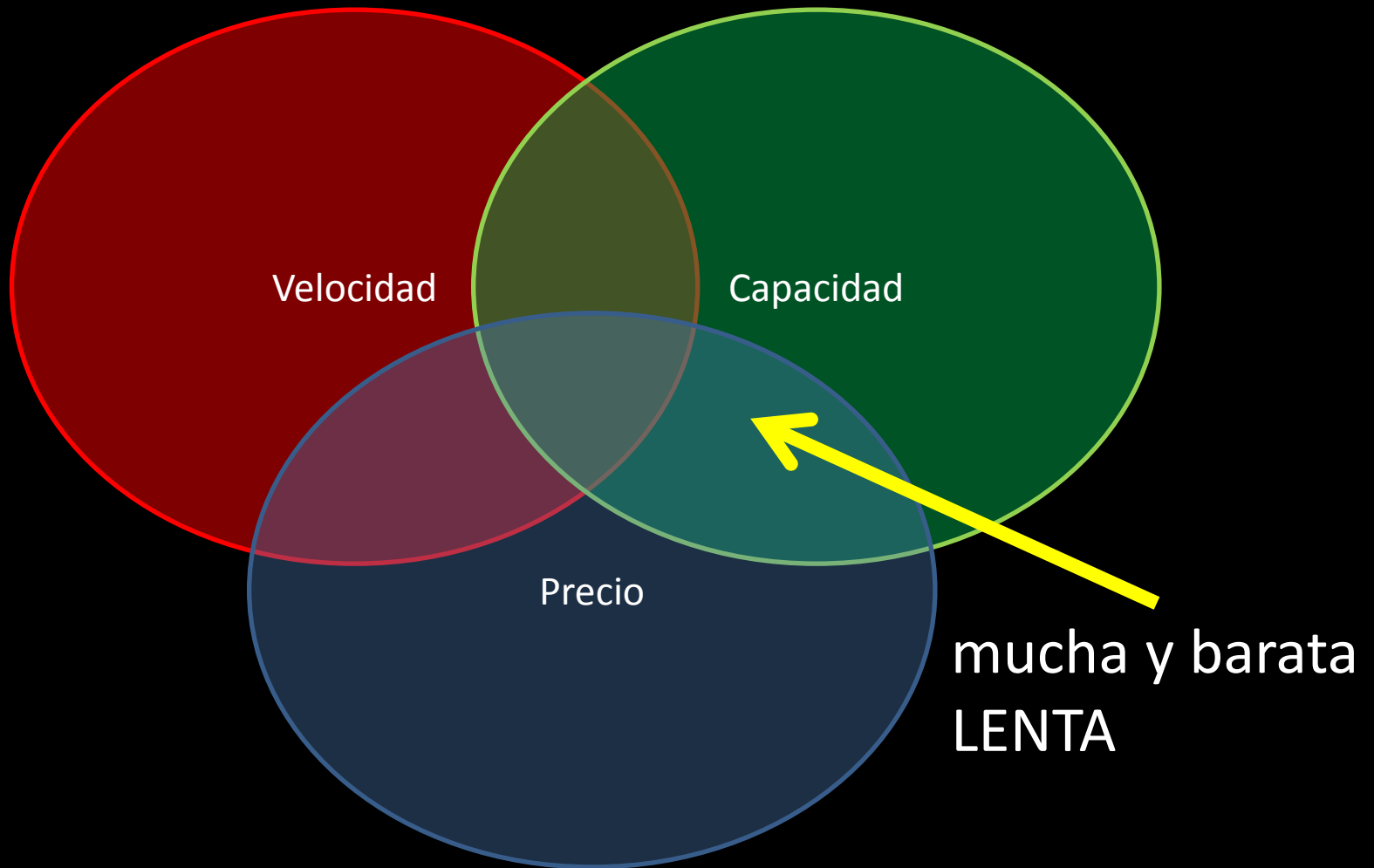


Mucha memoria
y muy rápida
MUY COSTOSA

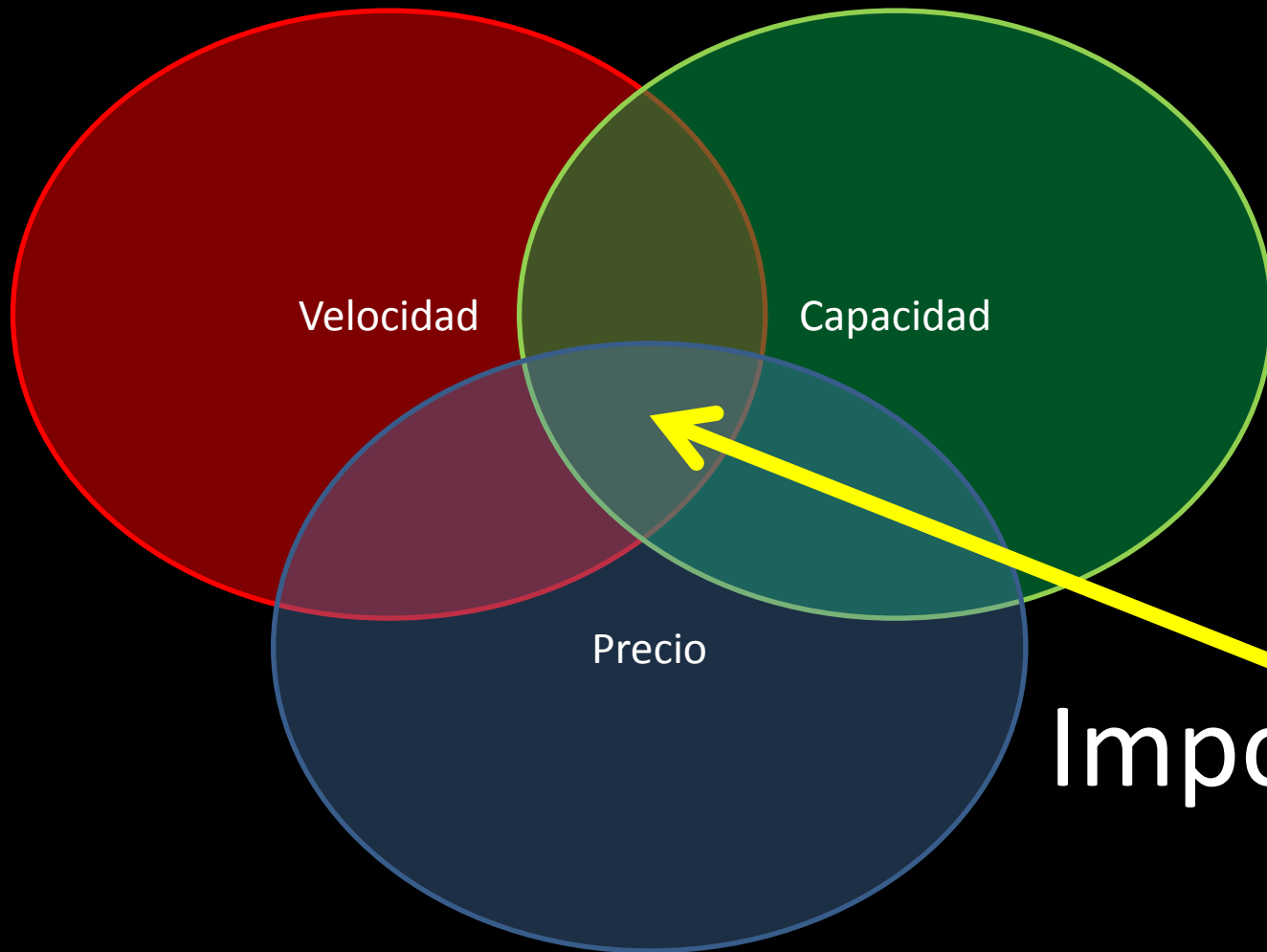
Jerarquía de memorias



Jerarquía de memorias



Jerarquía de memorias



Imposible!!

Precio por bit



+

Velocidad



+

Capacidad



+

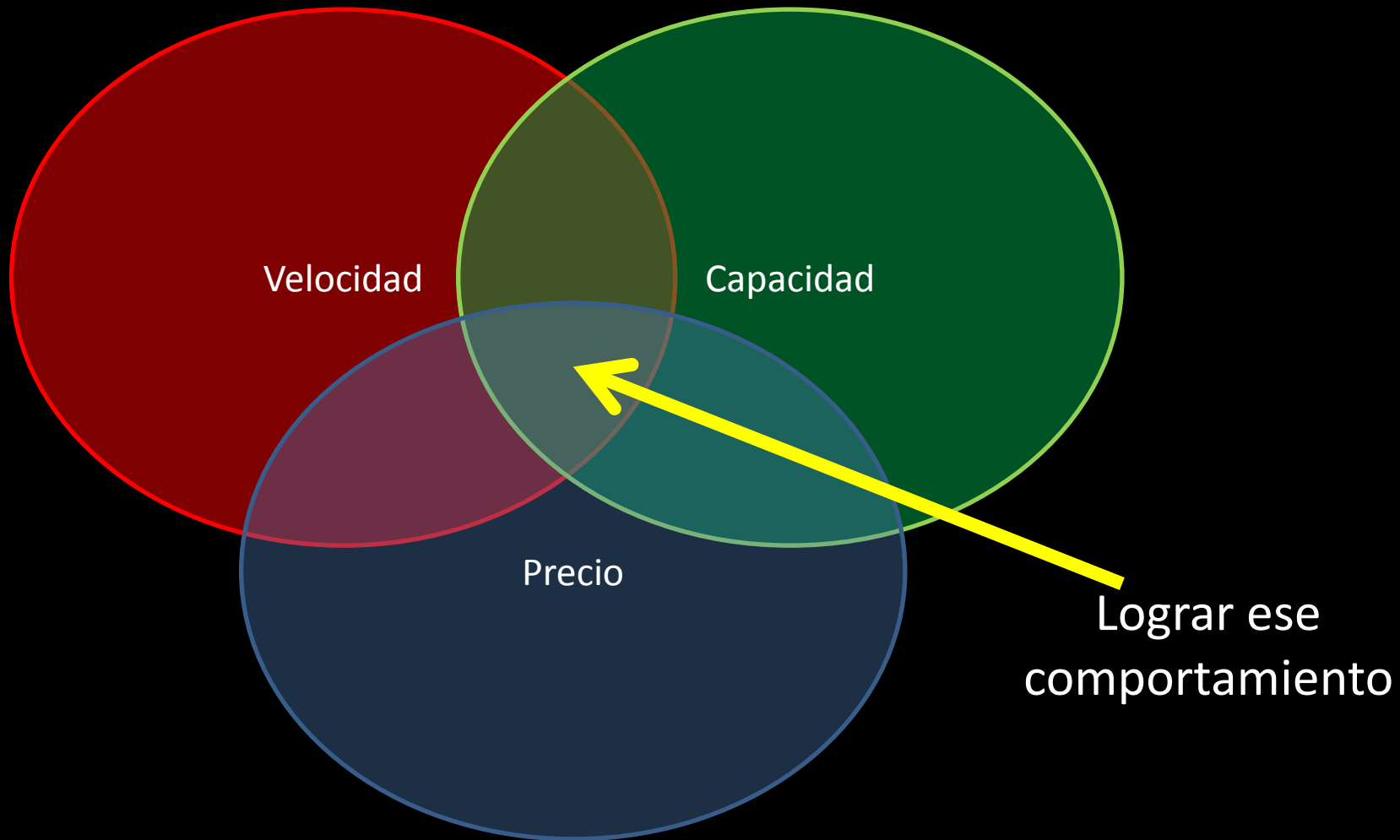
Registros

Memoria
principal

Memoria secundaria



Jerarquía de memorias



Jerarquía de memorias

- La **jerarquía de memoria** es la organización piramidal de la memoria en niveles.
- El objetivo es conseguir el rendimiento de una memoria de gran velocidad al coste de una memoria de baja velocidad, basándose en los principios anteriores.

Performance

- En base a estos principios se ve que tiene sentido “tener a mano” a las posiciones recientemente usadas
- Se puede agregar una memoria rápida pero de poca capacidad para ir guardando estas posiciones

Precio por bit



+

Velocidad

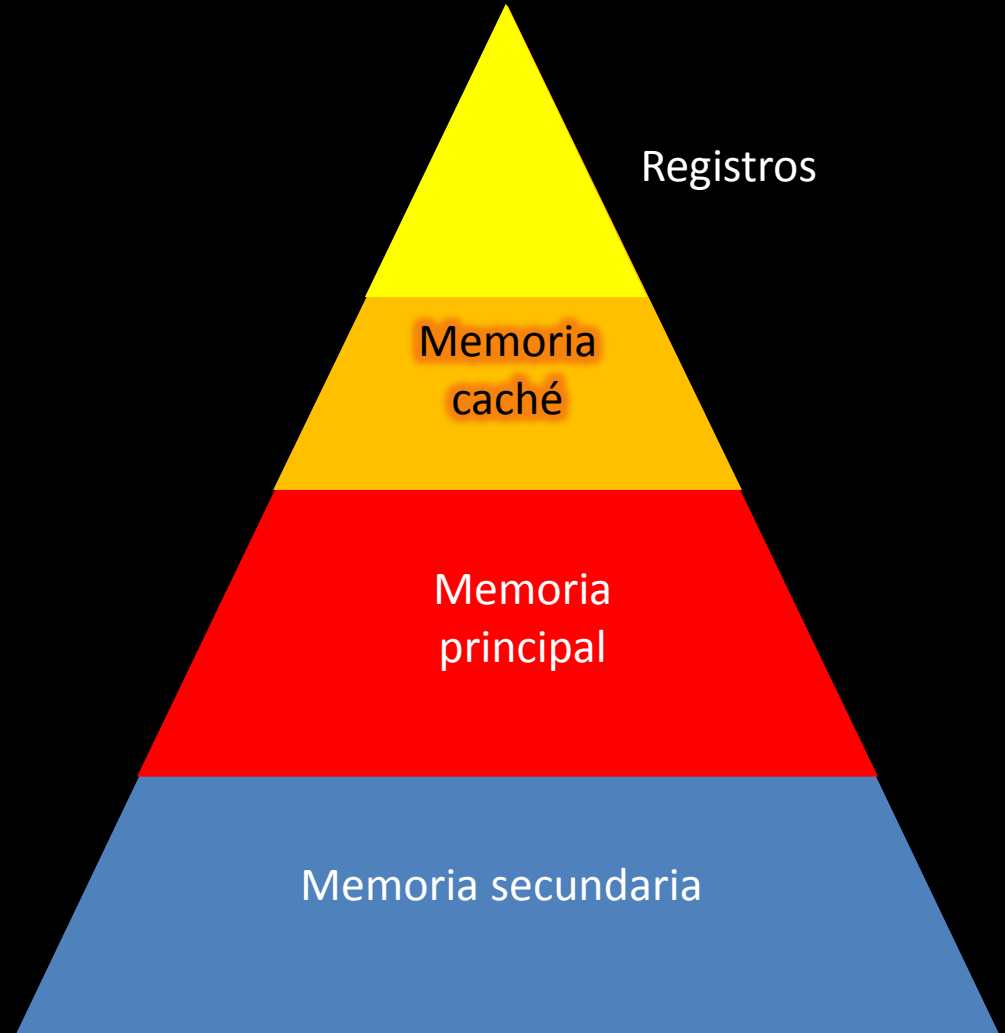


+

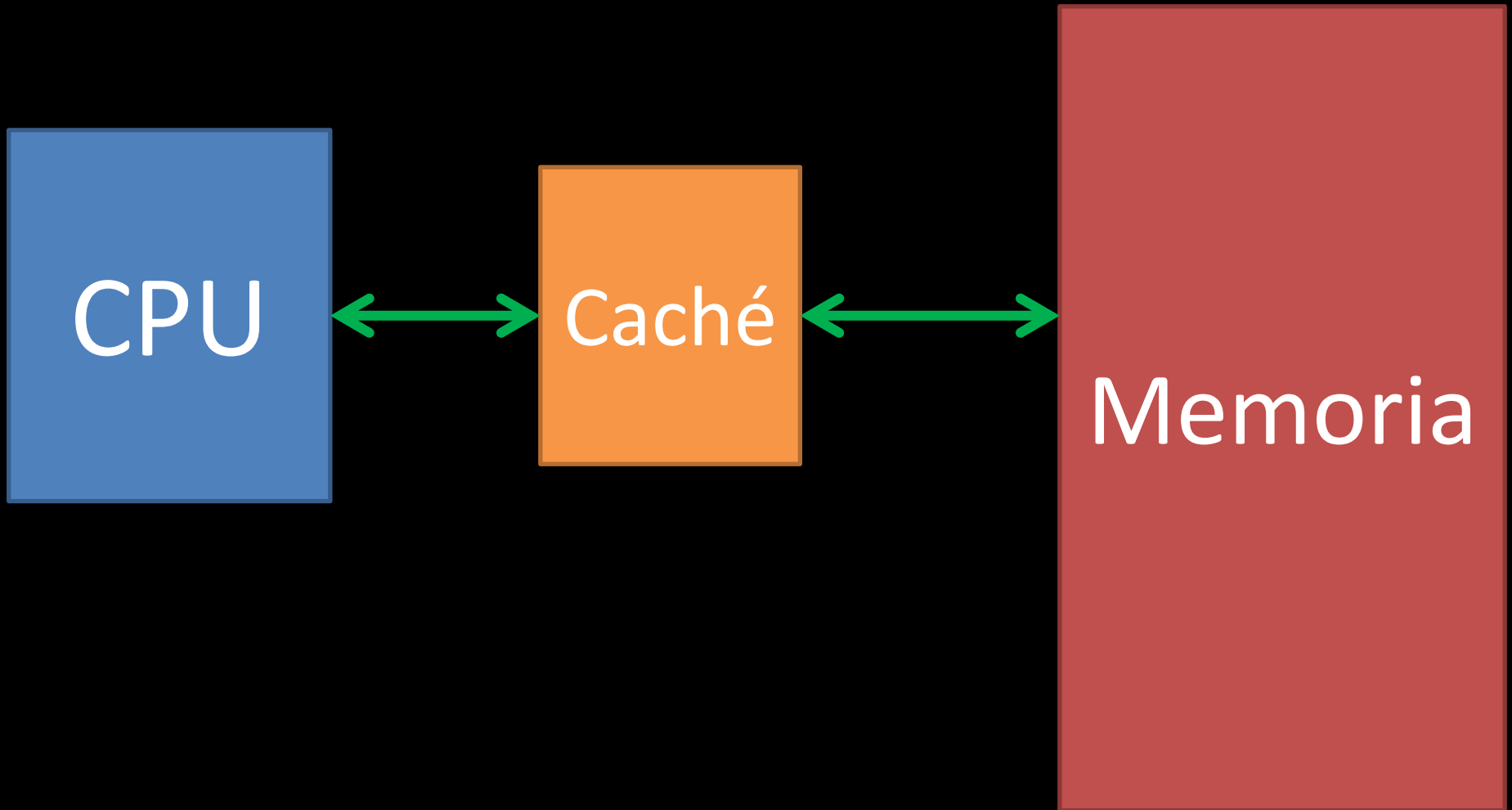
Capacidad



+



Memoria caché



Memoria caché

- ¿Cómo funciona?

Memoria caché

- ¿Cómo funciona?
 - La CPU pide una posición de memoria

Memoria caché

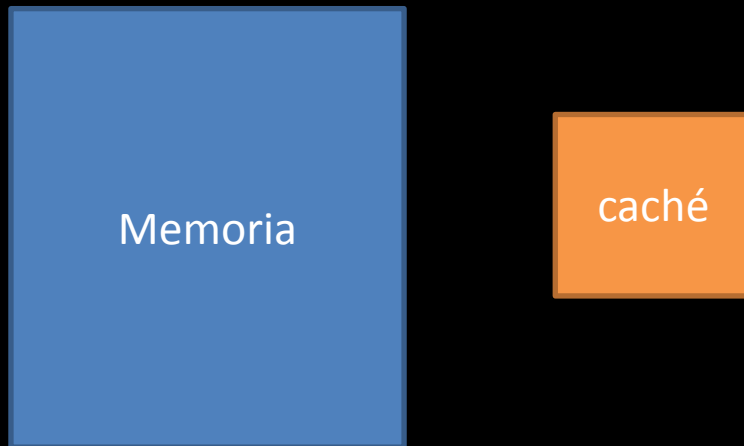
- ¿Cómo funciona?
 - La CPU pide una posición de memoria
 - La caché busca esta posición
 - Si está: la devuelve y no se accede a memoria
 - Si no: Se pide la posición a la memoria, se guarda en el cache y se la devuelve al CPU

Memoria caché

- ¿Cómo funciona?
 - La CPU pide una posición de memoria
 - La caché busca esta posición
 - Si está: la devuelve y no se accede a memoria
 - Si no: Se pide la posición a la memoria, se guarda en el cache y se la devuelve al CPU
- ¿Cómo saber si algo esta en la caché o no?

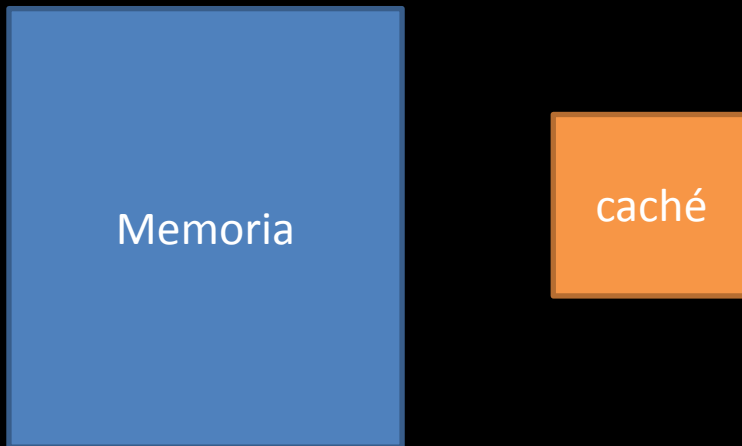
Memoria caché

- ¿Cómo saber si algo esta en la caché o no?
 - ¿Puede estar todo el contenido de la memoria en caché?



Memoria caché

- ¿Cómo saber si algo esta en la caché o no?
 - ¿Puede estar todo el contenido de la memoria en caché?



- Hay que decidir cómo y dónde se guarda lo que entra en la caché

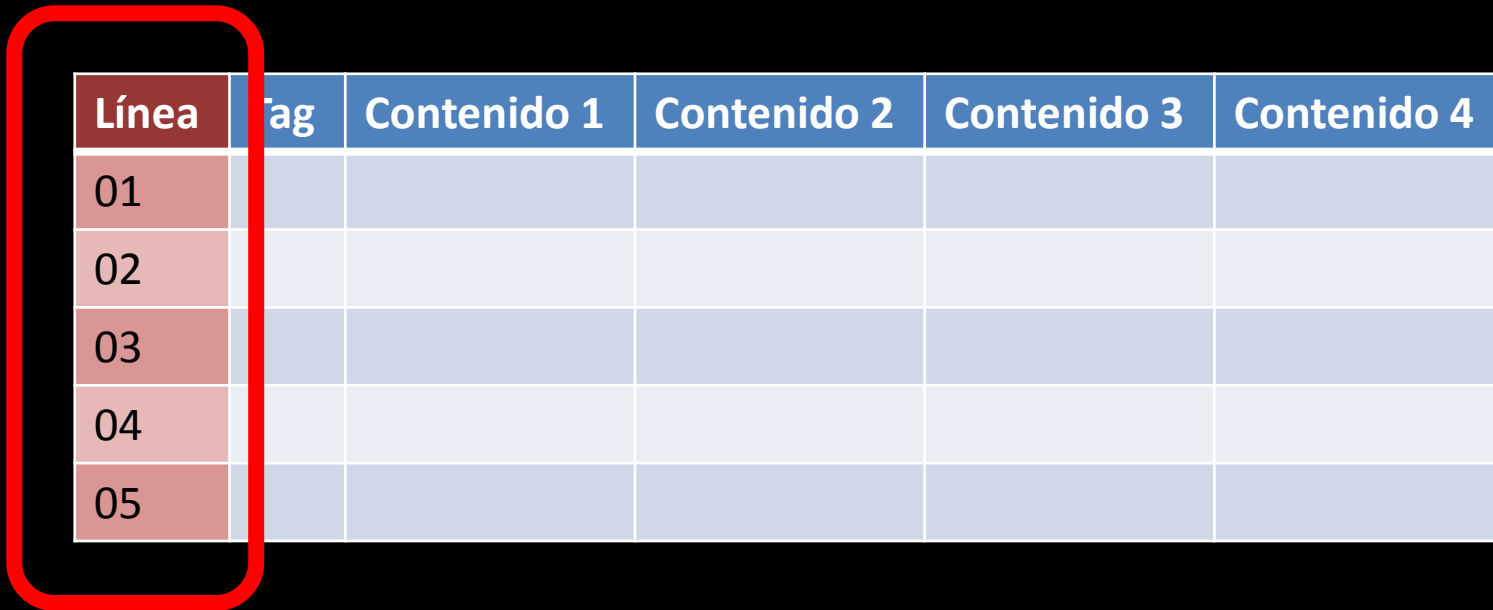
Memoria caché

- Organización:

Línea	Tag	Contenido 1	Contenido 2	Contenido 3	Contenido 4
01					
02					
03					
04					
05					

Memoria caché

- Organización:



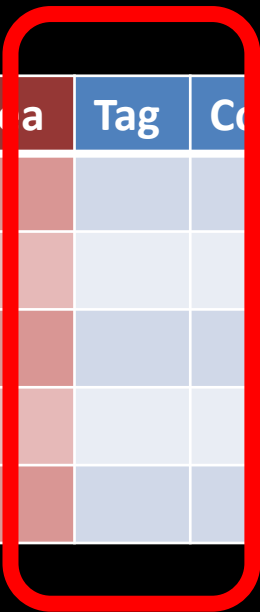
A diagram illustrating the organization of a cache. It features a table with six columns: 'Línea', 'Tag', 'Contenido 1', 'Contenido 2', 'Contenido 3', and 'Contenido 4'. The 'Línea' column is highlighted with a red rounded rectangle, indicating that the cache is organized by lines rather than individual cells. The table contains five rows of data, with the first row being the header and the subsequent four rows representing the cache lines.

Línea	Tag	Contenido 1	Contenido 2	Contenido 3	Contenido 4
01					
02					
03					
04					
05					

En vez de celdas la memoria caché tiene líneas
(esto no se guarda en la caché)

Memoria caché

- Organización:



Línea	Tag	Contenido 1	Contenido 2	Contenido 3	Contenido 4
01					
02					
03					
04					
05					

Indica que posición de memoria está guardada en cada línea (Memoria asociativa)

Tag

- Dada una posición de memoria, se le computa un tag.
- Una celda de memoria tiene un único tag y se calcula en base a su dirección.
- Cuando se pide una celda de memoria a la cache, esta calcula el tag y se fija si este está

Memoria caché

- Organización:



Línea	Tag	Contenido 1	Contenido 2	Contenido 3	Contenido 4
01					
02					
03					
04					
05					

Contenido de la celda de memoria. A veces se guardan varias celdas consecutivas (bloque) por línea. En ese caso se cargan todas juntas al cargarse alguna.

Memoria caché

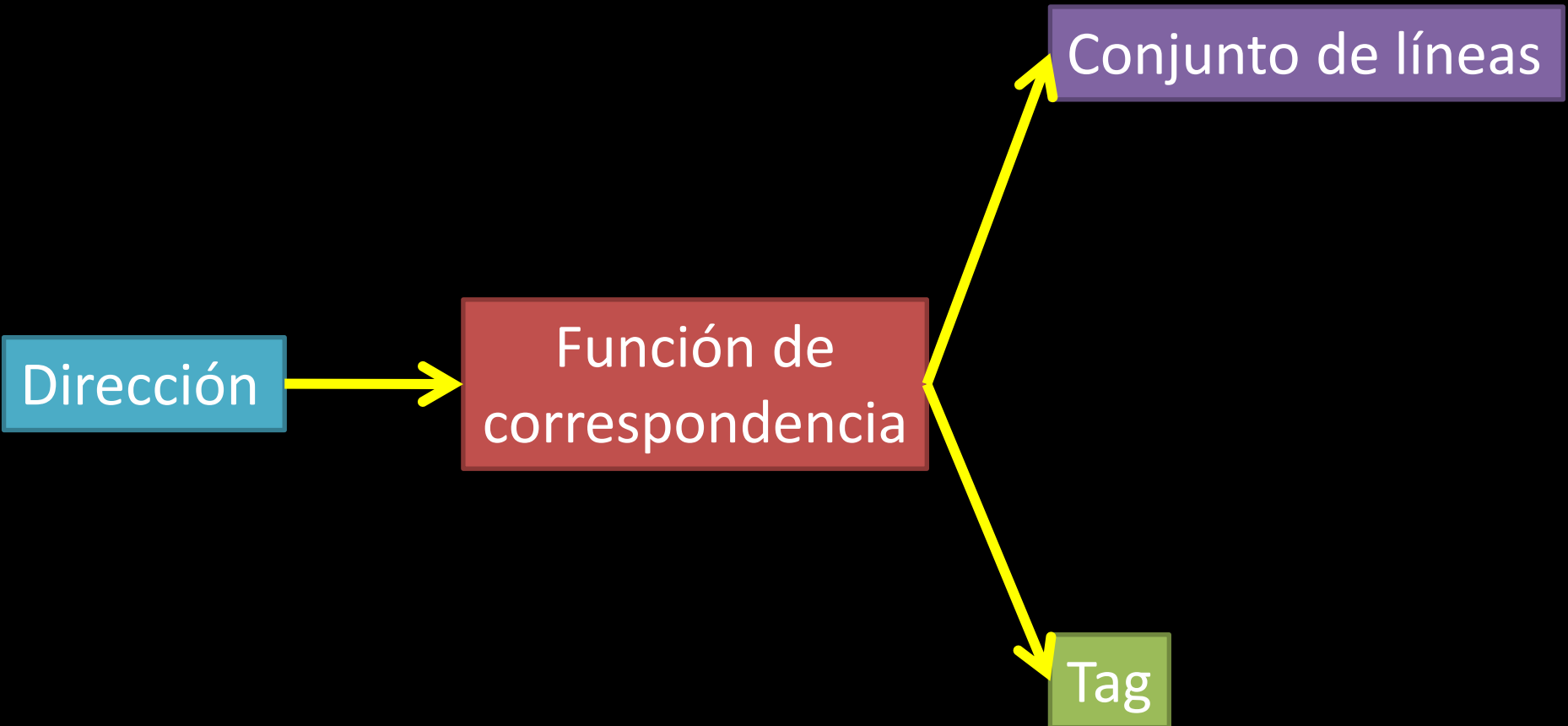
- ¿Cómo decidir en que línea guardar una celda de memoria?

Memoria caché

- ¿Cómo decidir en que línea guardar una celda de memoria?

Función de correspondencia

Función de correspondencia



Función de correspondencia

Dirección

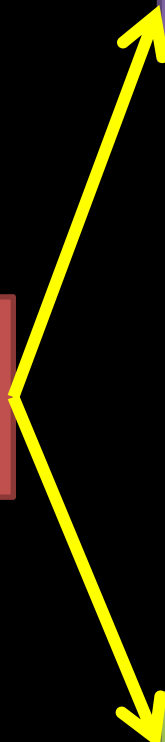
Función de
correspondencia

Conjunto de líneas

¿Dónde los puedo
guardar?

Tag

¿A quien/es guardo?



Función de correspondencia

- Mapeo directo:
 - Cada celda puede ir a un único lugar

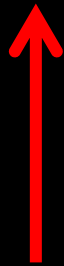
Función de correspondencia

- Mapeo directo:
 - Cada celda puede ir a un único lugar
 - La dirección se divide en 3 partes:
 - Tag: Indica el bloque de memoria que está guardada (hay que almacenarla en la caché)
 - Línea: Se usa para decidir en que línea se guarda el bloque (Esta parte no se guarda)
 - Índice: Indica dentro de la línea donde está una celda en particular

5FA1 → Mapeo Directo

Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0	FE	0000	2234	2312	1234
1	DE	345F	AAA2	BE32	212A
2	5F	EA34	123F	DDDD	CCCC
3	6E	A689	0985	4359	1237
4	BC	1223	1123	2132	5753
5	D0	2334	2553	2132	213E
6	56	DDB5	DB65	A235	EEEF
7	23	4569	2386	1234	0532
8	2E	2543	4128	8654	003A
9	12	1250	2354	4467	0990
A	5F	0086	FEDE	5056	2A10
B	03	3221	7652	4000	FE23
C	00	4652	1254	01A7	4567
D	AA	7904	0054	4322	8754
E	AA	56A3	0000	6789	4329
F	23	FC56	34AA	8754	8875

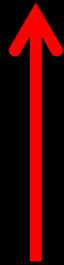
Tag: 5F
Línea: A
Índice: 1



5FA1 → Mapeo
Directo

Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0	FE	0000	2234	2312	1234
1	DE	345F	AAA2	BE32	212A
2	5F	EA34	123F	DDDD	CCCC
3	6E	A689	0985	4359	1237
4	BC	1223	1123	2132	5753
5	D0	2334	2553	2132	213E
6	56	DDB5	DB65	A235	EEEF
7	23	4569	2386	1234	0532
8	2E	2543	4128	8654	003A
9	12	1250	2354	4467	0990
A	5F	0086	FEDE	5056	2A10
B	03	3221	7652	4000	FE23
C	00	4652	1254	01A7	4567
D	AA	7904	0054	4322	8754
E	AA	56A3	0000	6789	4329
F	23	FC56	34AA	8754	8875

Tag: 5F
Línea: A
Índice: 1

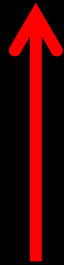


5FA1 → Mapeo
Directo

Línea A

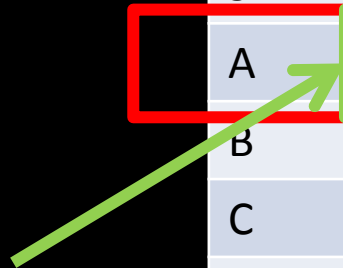
Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0	FE	0000	2234	2312	1234
1	DE	345F	AAA2	BE32	212A
2	5F	EA34	123F	DDDD	CCCC
3	6E	A689	0985	4359	1237
4	BC	1223	1123	2132	5753
5	D0	2334	2553	2132	213E
6	56	DDB5	DB65	A235	EEEF
7	23	4569	2386	1234	0532
8	2E	2543	4128	8654	003A
9	12	1250	2354	4467	0990
A	5F	0086	FEDE	5056	2A10
B	03	3221	7652	4000	FE23
C	00	4652	1254	01A7	4567
D	AA	7904	0054	4322	8754
E	AA	56A3	0000	6789	4329
F	23	FC56	34AA	8754	8875

Tag: 5F
Línea: A
Índice: 1



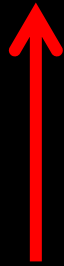
5FA1 → Mapeo
Directo

Hay que
buscar 5F



Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0	FE	0000	2234	2312	1234
1	DE	345F	AAA2	BE32	212A
2	5F	EA34	123F	DDDD	CCCC
3	6E	A689	0985	4359	1237
4	BC	1223	1123	2132	5753
5	D0	2334	2553	2132	213E
6	56	DDB5	DB65	A235	EEEF
7	23	4569	2386	1234	0532
8	2E	2543	4128	8654	003A
9	12	1250	2354	4467	0990
A	5F	0086	FEDE	5056	2A10
B	03	3221	7652	4000	FE23
C	00	4652	1254	01A7	4567
D	AA	7904	0054	4322	8754
E	AA	56A3	0000	6789	4329
F	23	FC56	34AA	8754	8875

Tag: 5F
Línea: A
Índice: 1



5FA1 → Mapeo
Directo

En 1 está el contenido
de esa celda

Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0	FE	0000	2234	2312	1234
1	DE	345F	AAA2	BE32	212A
2	5F	EA34	123F	DDDD	CCCC
3	6E	A689	0985	4359	1237
4	BC	1223	1123	2132	5753
5	D0	2334	2553	2132	213E
6	56	DDB5	DB65	A235	EEEF
7	23	4569	2386	1234	0532
8	2E	2543	4128	8654	003A
9	12	1250	2354	4467	0990
A	5F	0086	FEDE	5056	2A10
B	03	3221	7652	4000	FE23
C	00	4652	1254	01A7	4567
D	AA	7904	0054	4322	8754
E	AA	56A3	0000	6789	4329
F	23	FC56	34AA	8754	8875

Tag: 5F

Línea: A

Índice: 1

Para cargar no importa
el índice ya que se carga
toda la linea junta.

Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0					
1					
2					
3					
4					
5					
6					
7					
8					
9					
A					
B					
C					
D					
E					
F					



Número de línea
Número de piso





Índice

Número de depto



Tag
¿Quién es el inquilino?



Función de correspondencia

- Mapeo directo:
 - Ejemplo:
 - AFF1 con 8 bits de tag, 4 de línea y 4 de índice

Función de correspondencia

- Mapeo directo:
 - Ejemplo:
 - AFF1 con 8 bits de tag, 4 de línea y 4 de índice
 - Tag = AF, línea = F, índice = 1

Función de correspondencia

- Mapeo directo:
 - Ejemplo:
 - AFF1 con 8 bits de tag, 4 de línea y 4 de índice
 - Tag = AF, línea = F, índice = 1
 - A cada bloque le toca una única línea

Función de correspondencia

- Mapeo directo:
 - Ejemplo:
 - AFF1 con 8 bits de tag, 4 de línea y 4 de índice
 - Tag = AF, línea = F, índice = 1
 - A cada bloque le toca una única línea
 - Hay varios bloques que “compiten” por la misma línea

Función de correspondencia

- Mapeo directo:
 - Ejemplo:
 - AFF1 con 8 bits de tag, 4 de línea y 4 de índice
 - Tag = AF, línea = F, índice = 1
 - A cada bloque le toca una única línea
 - Hay varios bloques que “compiten” por la misma línea
 - Si hay que guardar un bloque y la línea esta ocupada, se saca a su ocupante actual

Ejemplo

- Calcular como se divide una dirección de memoria de 16 bits si se tiene una cache con mapeo directo de 4 celdas por bloque y 256 líneas
- ¿Cómo se decide si la dirección FA32 está en caché?

Función de correspondencia

- Mapeo directo:
 - Ventajas:
 - Sencillo de implementar
 - Desventajas:
 - Poco flexible: Si varias celdas con el mismo número de línea pero distinto tag se acceden todo el tiempo la caché no funciona muy bien.
 - Ejemplo: 8 bits de tag, 4 de línea y 4 de índice, y tengo 2 arreglos en AA00 y BB00

Función de correspondencia

- Asociativa:

Función de correspondencia

- Asociativa:
 - Cada bloque puede ir a cualquier línea de cache

Función de correspondencia

- Asociativa:
 - Cada bloque puede ir a cualquier línea de cache
 - Parte la dirección en tag e índice

Función de correspondencia

- Asociativa:
 - Cada bloque puede ir a cualquier línea de cache
 - Parte la dirección en tag e índice
 - Ejemplo: F145 en una cacha asociativa con 4 celdas por bloque

Función de correspondencia

- Asociativa:
 - Cada bloque puede ir a cualquier línea de cache
 - Parte la dirección en tag e índice
 - Ejemplo: F145 en una cacha asociativa con 16 celdas por bloque
 - Índice = 5
 - Tag = F14

Función de correspondencia

- Asociativa:
 - Cada bloque puede ir a cualquier línea de caché
 - Parte la dirección en tag e índice
 - Ejemplo: F145 en una caché asociativa con 16 celdas por bloque
 - Índice = 5
 - Tag = F14
 - ¿Qué pasa cuando la caché se llena? Algoritmos de reemplazo, mas adelante

Ejemplo

- Calcular como se divide una dirección de memoria de 16 bits si se tiene una cache asociativa de 4 celdas por bloque y 256 líneas
- ¿Cómo se decide si la dirección FA32 está en caché?

Función de correspondencia

- Asociativa:
 - Ventajas:
 - Muy flexible. Solo se saca algo de la caché cuando no hay mas lugar
 - Desventajas:
 - Muy compleja de implementar, hay que revisar toda la caché para saber si algo está o no.

Función de correspondencia

- Asociativa por conjuntos:

Función de correspondencia

- Asociativa por conjuntos:
 - Compromiso entre las anteriores.
 - Divide la caché en conjuntos de varias líneas.
 - Cada celda puede ir a un único conjunto
 - Dentro del conjunto puede ir a cualquier línea
 - Si se llena el conjunto hay que elegir a alguna para desalojar
 - La dirección se parte en tag, conjunto e índice

Conjuntos de 2 vías

Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0					
1					
2					
3					
4					
5					
6					
7					
8					
9					
A					
B					
C					
D					
E					
F					

Conjuntos de 2 vías

Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0					
1					
2					
3					
4					
5					
6					
7					
8					
9					
A					
B					
C					
D					
E					
F					

Conjuntos de 4 líneas

Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0					
1					
2					
3					
4					
5					
6					
7					
8					
9					
A					
B					
C					
D					
E					
F					

Conjuntos de 4 líneas

Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0					
1					
2					
3					
4					
5					
6					
7					
8					
9					
A					
B					
C					
D					
E					
F					

Conjuntos de 8 líneas

Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0					
1					
2					
3					
4					
5					
6					
7					
8					
9					
A					
B					
C					
D					
E					
F					

Conjuntos de 8 líneas

Línea	Tag	Cont 0	Cont 1	Cont 2	Cont 3
0					
1					
2					
3					
4					
5					
6					
7					
8					
9					
A					
B					
C					
D					
E					
F					

Ejemplo

- Calcular como se divide una dirección de memoria de 16 bits si se tiene una cache con asociativa por conjuntos de 4 vías, de 4 celdas por bloque y 256 líneas.
- ¿Cómo se decide si la dirección FA32 está en caché?

Políticas de desalojo



Políticas de desalojo

- En las cachés asociativas o asociativas por conjuntos puede ser necesario elegir que línea reemplazar.
- Hay distintos algoritmos para elegir a quien sacar

Políticas de desalojo

- Algoritmo LRU:

Políticas de desalojo

- Algoritmo LRU:
 - Least Recently Used = Menos recientemente usado

Políticas de desalojo

- Algoritmo LRU:
 - Least Recently Used = Menos recientemente usado
 - Se reemplaza el bloque que ha estado mas tiempo en la cache sin ser usado.

Políticas de desalojo

- Algoritmo LRU:
 - Least Recently Used = Menos recientemente usado
 - Se reemplaza el bloque que ha estado mas tiempo en la cache sin ser usado.
 - Se necesita almacenar información extra para saber cuando se la utilizó por ultima vez:
 - En asociativa por conjunto de dos vías es muy fácil, con un solo bit alcanza. ¿Cómo?

Políticas de desalojo

- Algoritmo FIFO:

Políticas de desalojo

- Algoritmo FIFO:
 - First In-First Out = El primero que llega es el primero que se va.

Políticas de desalojo

- Algoritmo FIFO:
 - First In-First Out = El primero que llega es el primero que se va.
 - Se reemplaza el bloque que ha estado en la cache por mas tiempo.

Políticas de desalojo

- Algoritmo LFU:

Políticas de desalojo

- Algoritmo LFU:
 - Least Frequently Used = Menos frecuentemente usado

Políticas de desalojo

- Algoritmo LFU:
 - Least Frequently Used = Menos frecuentemente usado
 - Se reemplaza el bloque que ha sido usado menos veces

Políticas de desalojo

- Algoritmo LFU:
 - Least Frequently Used = Menos frecuentemente usado
 - Se reemplaza el bloque que ha sido usado menos veces
 - Puede implementarse con un contador de accesos en cada ranura

Políticas de desalojo

- Algoritmo Random:
 - Al azar
 - Suele ser menos performante

Políticas de escritura



Políticas de escritura

- Cuando se escribe una celda que está en caché, ¿Qué pasa?

Políticas de escritura

- Cuando se escribe una celda que está en caché, ¿Qué pasa?
 - Write through: Se escribe inmediatamente a memoria. Las escrituras no se benefician de la caché

Políticas de escritura

- Cuando se escribe una celda que está en caché, ¿Qué pasa?
 - Write through: Se escribe inmediatamente a memoria. Las escrituras no se benefician de la caché
 - Write back: Se marca a la celda como sucia. Cuando se la desaloja, ahí se escribe a memoria. Hay que agregar un bit para indicar que está sucia.

Tasa de aciertos



Tasa de aciertos

- Nos va a interesar saber cuantos de los accesos a memoria encuentran el dato en cache, ya que esto es un claro indicio del desempeño de la misma

Tasa de aciertos

- Nos va a interesar saber cuantos de los accesos a memoria encuentran el dato en cache, ya que esto es un claro indicio del desempeño de la misma
- Una sistema con una cache con baja tasa de aciertos puede ser mas lento que uno sin cache. (¿Por qué?)

HIT



MISS



Tasa de aciertos

- Si tenemos n accesos a memoria, y de esos k fueron hit, la tasa de aciertos se calcula como:

$$Ta = k/n$$

- La tasa de fallos es:

$$Tf = 1 - Ta$$

Tiempo de acceso total

- Si tenemos n accesos, una tasa de aciertos T_a y conocemos el tiempo de cache t_c y el tiempo de respuesta de la memoria t_m :

$$t_p = n * T_a * t_c + n * (1 - T_a) * (t_c + t_m)$$

Ejercicio

- Se tiene un sistema con una memoria principal con un tiempo de acceso de 3s, y una memoria caché cuyo tiempo de acceso es de 0,3 y cuya tasa de aciertos es del 90 %.

¿Cuánto tiempo se tarda en promedio para leer 2000 celdas?

Ejercicio

Considerar una máquina con una memoria cache de mapeo directo de 1024 líneas, un tamaño de bloque de 4 bytes y una memoria principal de 64 KBytes con celdas de un byte.

a) Dada una dirección de memoria calcular la cantidad de bits que se destinan a: tag, línea e índice.

b) Suponer que la cache esta vacía, y que se realizan lecturas de datos cuyas direcciones están en el siguiente orden:

**0xEA00, 0xEA01, 0x9A03, 0xEA02, 0xEA04, 0xEA05, 0xEA07,
0xEA04**

Determinar para cada lectura si esta produjo un fallo o un acierto.

c) Sabiendo que el tiempo de acceso a la memoria ram es de 0,5 micro-segundos, calcular el tiempo máximo de acceso a la cache para que el tiempo total de los accesos sea inferior a los 2,4 microsegundos

¿Qué nos llevamos hoy?

¿Qué nos llevamos hoy?

- Memorias:

¿Qué nos llevamos hoy?

- Memorias:
 - Características

¿Qué nos llevamos hoy?

- Memorias:
 - Características
 - Memorias ROM

¿Qué nos llevamos hoy?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias

¿Qué nos llevamos hoy?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:

¿Qué nos llevamos hoy?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación

¿Qué nos llevamos hoy?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación
 - ¿Qué?

¿Qué nos llevamos hoy?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación
 - ¿Qué?
 - Organización

¿Qué nos llevamos hoy?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación
 - ¿Qué?
 - Organización
 - Función de mapeo

¿Qué nos llevamos hoy?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación
 - ¿Qué?
 - Organización
 - Función de mapeo
 - Política de reemplazo

¿Qué nos llevamos hoy?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación
 - ¿Qué?
 - Organización
 - Función de mapeo
 - Política de reemplazo
 - Política de escritura

¿Qué nos llevamos hoy?

- Memorias:
 - Características
 - Memorias ROM
 - Jerarquía de memorias
- Caché:
 - Motivación
 - ¿Qué?
 - Organización
 - Función de mapeo
 - Política de reemplazo
 - Política de escritura
 - Tasa de aciertos



