



MEMORIA CACHE

MEMORIAS

UBICACION:

- INTERNAS
 - EJ.RAM
- EXTERNAS
 - EJ.PENDRIVE,

VOLATILIDAD:

CAPACIDAD DE ESCRITURA

METODO DE ACCESO:

SECUENCIAL

DIRECTO

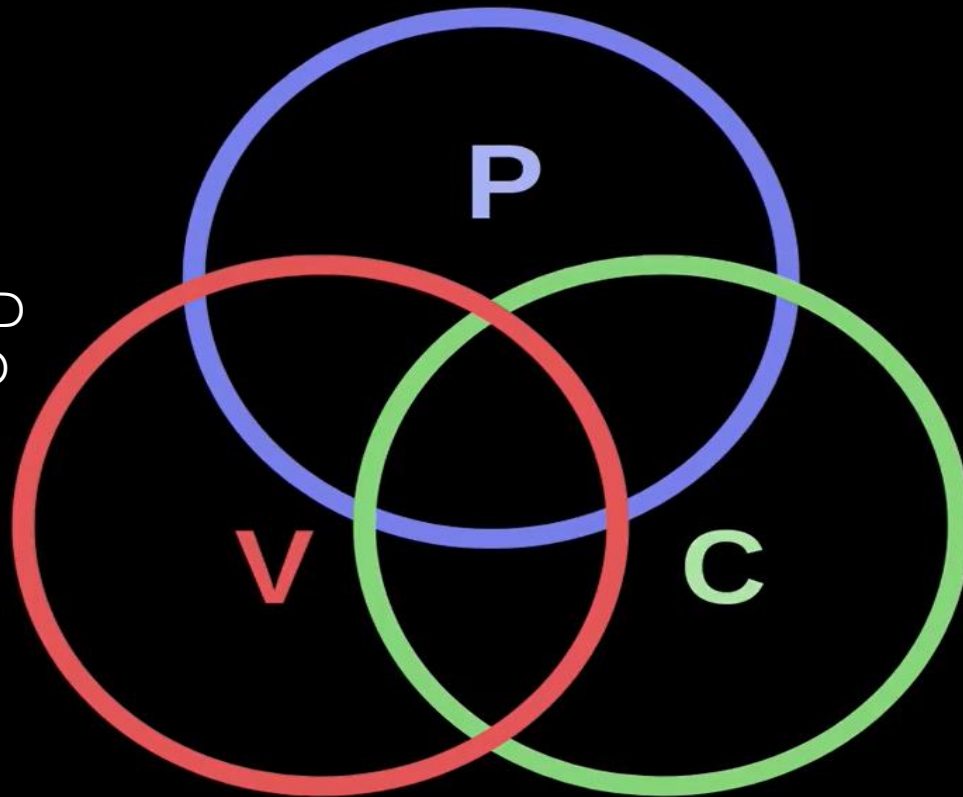
ALEATORIO

MEMORIAS DE SOLO LECTURA:

ROM, EPROM...SON MEMORIAS QUE NO VAN A SER ESCRITAS Y BORRADAS TODO EL TIEMPO (EJ MICROONDAS)

Memoria

PRECIO
CAPACIDAD
VELOCIDAD



VELOCIDAD Y BUENA CAPACIDAD= ALTO PRECIO
PRECIO BAJO Y CAPACIDAD= MEMO LENTA
PRECIO BAJO Y VELOZ= POCA CAP DE ALMACENAMIENTO

Memoria

Velocidad

Precio x bit

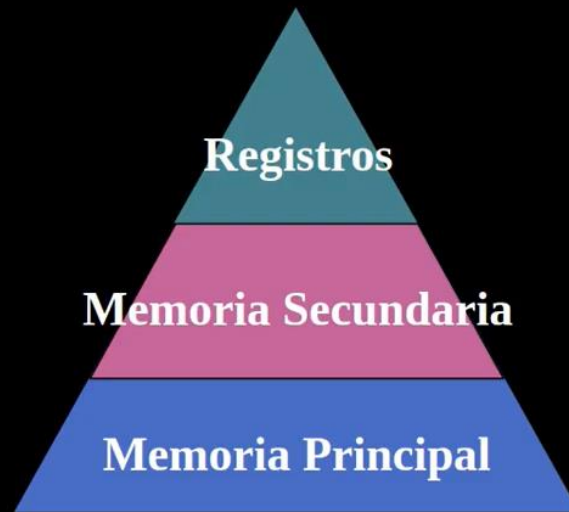
Capacidad



Registros

Memoria Secundaria

Memoria Principal



MEMORIA CACHE

Memoria

- CPU va a memoria siempre para conseguir la siguiente instrucción
- El problema es que es mucho más rápida y la memoria es mucho más lenta

PATRONES EN LOS ACCESOS A MEMORIA

PROCESAMOS DATOS DE UN ARREGLO: LOS DATOS SE GUARDAN DE MANERA CONTINUA EN LA MEMORIA

PC BUSCA CELDA SIGUIENTE SALVO QUE HAYA UN SALTO

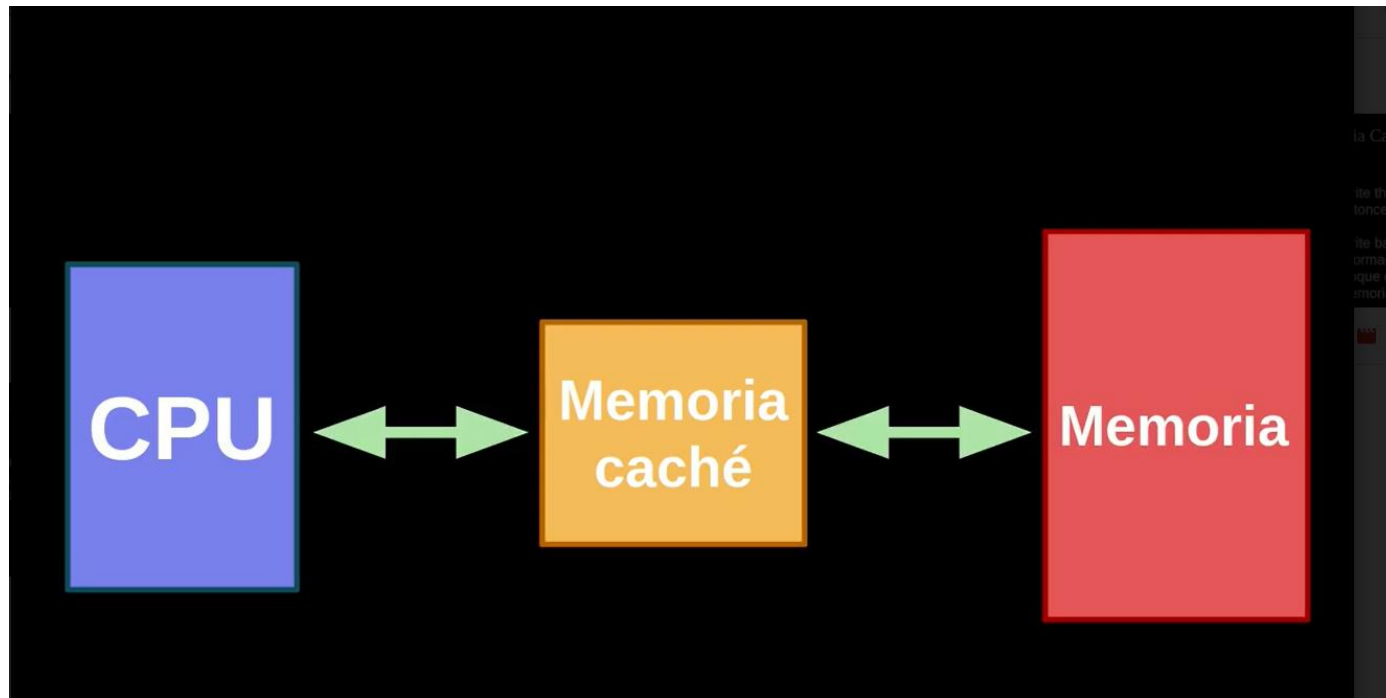
DURANTE LA EJECUCION EJECUTAMOS UN BUCLE, EN ESE CASO PODEMOS EJECUTAR VARIAS VECES LAS MISMAS CELDAS.

PRINCIPIOS DE LOCALIDAD

ESPACIAL Y TEMPORAL

ES PROBABLE UTILIZAR CELDAS ALREDEDOR DE LA ACTUAL

SE UTILIZA UNA MEMO MAS PEQUEÑA QUE CONTENGA ESTOS
DATOS A MANO, SIGUIENDO ESTOS PRINCIPIOS



Algoritmo de lectura

- (1) La CPU pide una celda X
- (2) La cache controla si X está *cacheada*
 - ✓ Si está cacheada (hit o acierto):
se otorga X a la CPU
 - ✗ Si NO está cacheada (fault o fallo):
 - ① Se lee un bloque de memoria principal que contiene X
 - ② Se cachea el bloque
 - ③ se otorga X a la CPU

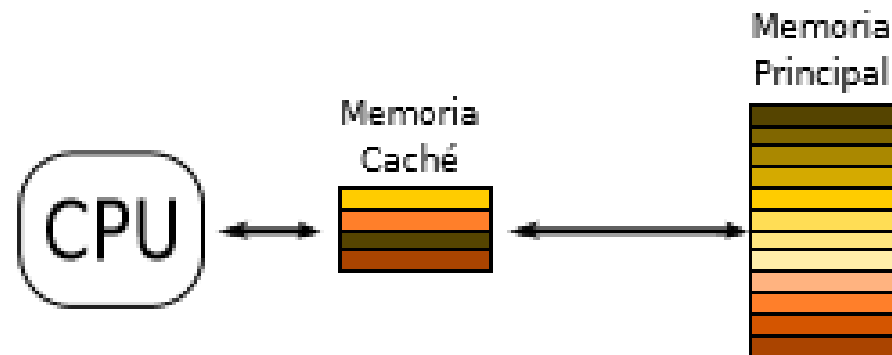
Algoritmo de escritura

- (1) La CPU pide la escritura del valor V en la celda X
- (2) La cache controla si X está *cacheada*
 - ✓ Si está cacheada (hit o acierto):
se escribe V en la copia de X **en la cache**
 - ✗ Si NO está cacheada (fault o fallo):
 - ① Se lee un bloque de memoria principal que contiene X
 - ② Se cachea el bloque
 - ③ se escribe V en la copia de X **en la cache**

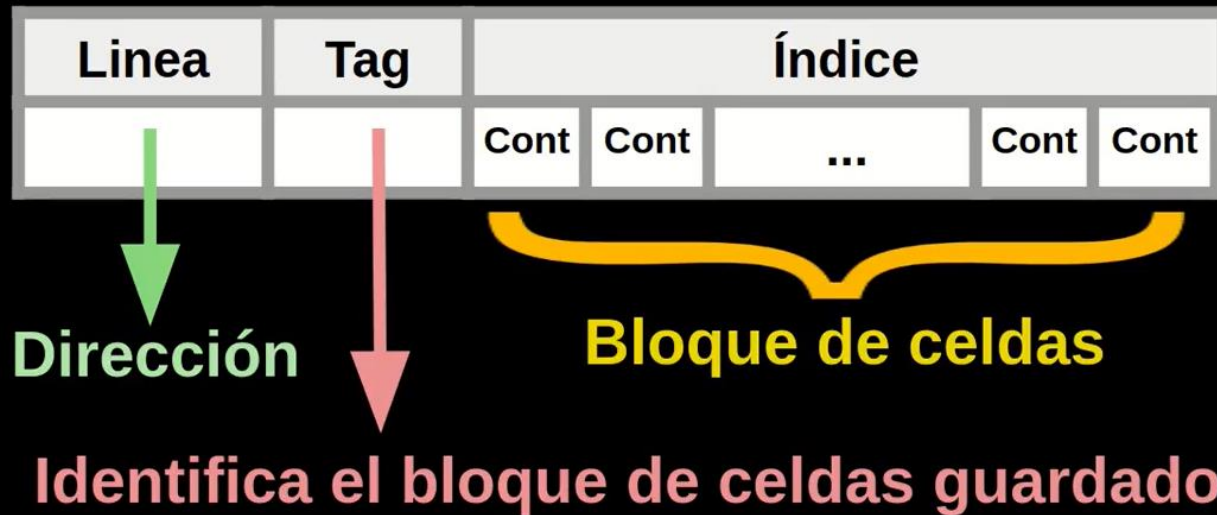
La memoria caché tiene como objetivo

- proveer una velocidad de acceso cercana a la de los registros
- pero al mismo tiempo proveer un mayor tamaño de memoria

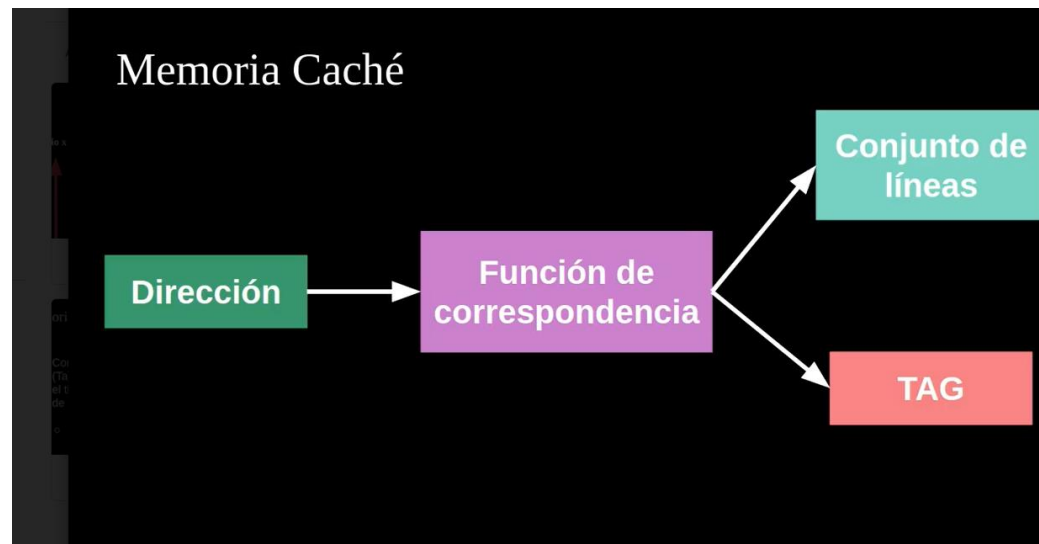
Para esto la caché contiene una copia de porciones de la memoria principal



Memoria Caché



COMO SABER GUARDAR O BUSCAR CONTENIDO EN CACHE?



Memoria Caché - Mapeo directo

Dirección de memoria



Línea de caché

Celda dentro del bloque

Identifica el bloque de celdas guardado

Bloque de celdas (índice)

Línea	Tag	Cont 0	Cont 1	...	Cont 15
0	FE	0xFFFF	0x1123	...	0xDE03
1	AD	0xFA34	0x234A	...	0xFAEE
2	02	0x0F14	0xC344	...	0xCD43
...
E	8D	0xCF02	0xD459	...	0x0003
F	95	0xABCD	0x0003	...	0xCDF3

Bloque de celdas (índice)

0x0220



Tag: 02
Linea: 2
Indice: 0



Línea	Tag	Cont 0	Cont 1	...	Cont 15
0	FE	0xFFFF	0x1123	...	0xDE03
1	AD	0xFA34	0x234A	...	0xFAEE
2	02	0x0F14	0xC344	...	0xCD43
...
E	8D	0xCF02	0xD459	...	0x0003
F	95	0xABCD	0x0003	...	0xCDF3

FALLO

Bloque de celdas (índice)

0x0220



Tag: 02

Línea: 2

Índice: 0



Línea	Tag	Cont 0	Cont 1	...	Cont 15
0	FE	0xFFFF	0x1123	...	0xDE03
1	AD	0xFA34	0x234A	...	0xFAEE
2	AB	0x0015	0x5501	...	0xAA11
...
E	8D	0xCF02	0xD459	...	0x0003
F	95	0xABCD	0x0003	...	0xCDF3

QUÉ HAGO ENTONCES?

Buscar en memoria
contenidos de: 0x0220X

Bloque de celdas (índice)

0x0220



Tag: 02
Linea: 2
Indice: 0



Línea	Tag	Cont 0	Cont 1	...	Cont 15
0	FE	0xFFFF	0x1123	...	0xDE03
1	AD	0xFA34	0x234A	...	0xFAEE
2	AB	0x0015	0x5501	...	0xAA11
...
E	8D	0xCF02	0xD459	...	0x0003
F	95	0xABCD	0x0003	...	0xCDF3

Buscar en memoria
contenidos de: 0x022X

Bloque de celdas (índice)

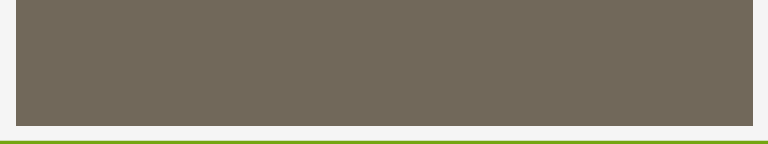
0x0220



Tag: 02
Linea: 2
Indice: 0



Línea	Tag	Cont 0	Cont 1	...	Cont 15
0	FE	0xFFFF	0x1123	...	0xDE03
1	AD	0xFA34	0x234A	...	0xFAEE
2	02	0x0F14	0xC344	...	0xCD43
...
E	8D	0xCF02	0xD459	...	0x0003
F	95	0xABCD	0x0003	...	0xCDF3



Cada bloque de memoria principal esté asignado a una línea determinada. Como la cantidad de líneas es mucho menor a la cantidad de bloques (porque en otro caso la memoria principal y la cache tendrían la misma capacidad), hay un conjunto de bloques candidatos para una misma línea que “compiten” entre si.

Para verificar si una celda esta cacheada, la memoria caché compara solamente en una línea (la que corresponde al bloque) para determinar si contiene el tag buscado, simplificando la tecnología que se necesita para implementar la cache.

EJERCICIO 2 CORRESPONDENCIA DIRECTA :

2. Considerar una computadora con una memoria de 64 celdas de un byte y una memoria cache con 4 líneas y bloques de 8 celdas por línea. Dada una dirección de memoria calcular la cantidad de bits que se destinan a: **tag, línea y palabra.**

- MEMORIA PRINCIPAL:

64 CELDAS DE 1 BYTE EN MEMORIA

CANTIDAD DE BITS QUE TIENEN LAS DIRECCIONES DE MEMORIA:

$$2^X = 64$$

$$X = 6 \text{ BITS}$$

- MEMORIA CACHÉ:

- 8 CELDAS POR BLOQUE, 4 LÍNEAS.**

LÍNEA: 4 LINEAS O SEA $2^Z = 4$

$$Z = 2 \text{ BITS}$$

ÍNDICE: 8 CELDAS POR BLOQUE O SEA $2^Y = 8$

$$Y = 3 \text{ BITS}$$

TAG: $X - (Y + Z)$

$$6 - (3 + 2)$$

TAG 1 LÍNEA 2 ÍNDICE 3

Podemos concluir:

- La correspondencia directa es mas económica en su construcción pero la correspondencia asociativa es mas flexible y maximiza el porcentaje de aciertos.
- Para hacerlo evidente, suponer una secuencia de accesos que requiera repetidamente cachear bloques que corresponden a una misma línea, causando repetidos fallos. En una correspondencia asociativa, esos bloques no compiten y no se darían mas fallos que los que producen bloques nuevos.

Performance

- HIT: Pedido a memoria que está en Caché
- MISS: Pedido a Memoria que no está en Caché

● 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

Un sistema con una cache con **baja tasa de aciertos** puede ser mas lento que uno sin cache.

Tasa de aciertos:

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

$$T_a = k/n$$

- La tasa de fallos es:

$$T_f = 1 - T_a$$

- 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 .

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

4) Se tiene un sistema con una memoria principal con un tiempo de acceso de 35s, y una memoria caché cuyo tiempo de acceso es de 0,25 s y cuya tasa de aciertos es del 95%. ¿Cuánto tiempo se tarda en leer 8000 celdas?

$$TP = 8000 * 0,95 * 0,25 \text{ SEG} + 8000 * 0,05 * (0,25 \text{ seg} + 35 \text{ seg}) = 16000 \text{ seg}$$

$$\underline{tp} = n * Ta * \underline{tc} + n * (1 - Ta) * (\underline{tc} + \underline{tm})$$

3) Dado el siguiente programa ensamblado a partir de la celda 0x0B05

MOV R2, 0x0003

ciclo: CMP R2, 0x0000

JE FIN

ADD R4, [0x0001]

SUB R2, 0x0001

JMP ciclo

Fin: RET

4. Considerando una computadora con las siguientes características:

Bus de direcciones de 16 bits.

Memoria caché con 16 líneas.

Bloques de 16 celdas.

Mapeo directo

Caché inicialmente vacía

- a) Completar la siguiente tabla sobre los accesos a memoria caché durante su ejecución:

Celda | Tag | Línea | Palabra | Fallo o Acierto

- b) ¿Cuál es la tasa de aciertos y de fallos?
-

	0B05	1880	MOV R2,0x0003
	0B06	0003	
ciclo	0B07	6880	CMP R2,0x0000
	0B08	0000	
	0B09	F106	JE fin
	0B0A	2910	ADD R4,[0x0001]
	0B0B	0001	
	0B0C	3880	SUB R2,0x0001
	0B0D	0001	
	0B0E	A000	JMP ciclo
	0B0F	0B07	
Fin	0B10	C000	RET

3) Dado el siguiente programa ensamblado a partir de la celda 0x0B05

MOV R2, 0x0003

ciclo: CMP R2, 0x0000

JE FIN

ADD R4, [0x0001]

SUB R2, 0x0001

JMP ciclo

Fin: RET

Direccion = 16 Bits

Linea = 4 bits

Palabra = 4 bits

Direccion = Tag + Linea + Palabra

16 Bits = Tag + 4 Bits + 4 Bits

Tag = Direccion - Linea - Palabra

8 Bits = Tag

Celda	Tag(8 Bits)	Linea(4 Bits)	Palabra (4 Bits)	F/A
0B05	0B	0	5	F
0B06	0B	0	6	A
0B07	0B	0	7	A/A/A/A
0B08	0B	0	8	A/A/A/A
0B09	0B	0	9	A/A/A/A
0B0A	0B	0	A	A/A/A
0B0B	0B	0	B	A/A/A
0001	00	0	1	F/A/A
0B0C	0B	0	C	A/A/A
0B0D	0B	0	D	A/A/A
0B0E	0B	0	E	A/A/A
0B0F	0B	0	F	A/A/A
0B10	0B	1	0	F

Direccion = 16 Bits

Linea = 4 bits

Palabra = 4 bits

Direccion = Tag + Linea + Palabra

16 Bits = Tag + 4 Bits + 4 Bits

Tag = Direccion - Linea - Palabra

8 Bits = Tag

ACCESOS A MEMORIA = 36

FALLOS = 3

ACIERTOS = 33

TA = 33/36

TF = 3/36

	0B05	1880	MOV R2,0x0003
	0B06	0003	
ciclo	0B07	6880	CMP R2,0x0000
	0B08	0000	
	0B09	F106	JE fin
	0B0A	2910	ADD R4,[0x0001]
	0B0B	0001	
	0B0C	3880	SUB R2,0x0001
	0B0D	0001	
	0B0E	A000	JMP ciclo
	0B0F	0B07	
Fin	0B10	C000	RET

3) Dado el siguiente programa ensamblado a partir de la celda 0x0B05

MOV R2, 0x0003

ciclo: CMP R2, 0x0000

JE FIN

ADD R4, [0x0001]

SUB R2, 0x0001

JMP ciclo

Fin: RET

