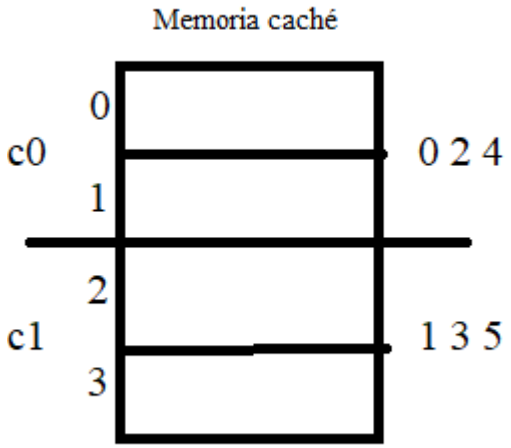
Posicionamiento de bloques

- Mapeo directo
  Un bloque concreto de la memoria principal se sitúa (cuando tengamos que llamarlo o si ya está en caché) Siempre en la misma posición de la caché.
  - ➔ Pros
    Es más rápido al buscar si el bloque necesitado está o no está en la caché porque sé dónde debería estar. Esta posición se calcula como i%nb
  - ➔ Contra no permite cualquier combinación de los bloques de la memoria principal.

- Totalmente asociativo
  - ➔ Pros
    Permite cualquier combinación de los bloques de la memoria principal.

- Asociativa por conjuntos
  Se dividen los bloques en conjuntos de tantos bloques como nos indique las vías (x-ways, vías de x, x es un número).
  Un bloque (i)de memoria principal está asociado a un conjunto (c)en concreto pero wn cualquiera de sus vías, el conjunto se calcula como en el directo c=i%x



* LA directa y la totalmente asociativa son tipos específicos de la de conjuntos (la directa tiene un solo conjunto y la asociativa tiene una sola vía **curiosidad* *

Ejemplo

LOOP
100
17
79
50
END LOOP

	*1 dirección MP	*2 bloques MP	*3 posi MC	A/F
	100	25	1	F A
	17	4	4	F F
	79	19	3	F A
	50	12	4	F F

*1 se divide entre el número de instrucciones de cada bloque(i%capacidad bloque)

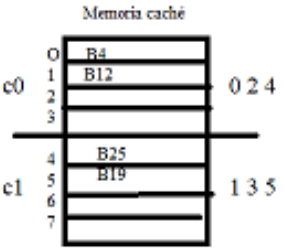
*2 depende del tipo de cache porque en directa depende del número de bloque (i%num de bloques de cache)

*3 Si no está en el caché o hay que sustituir se considera fallo sino acierto

*4 Para saber a que conjunto i%num conjuntos

		*4 Conjunto	
100	25	1	F A
17	4	0	F A
79	19	1	F A
50	12	0	F A

Cache directa



Asociatica por conjunto 4-ways(--> 2 conjuntos)

En binario

Para dividir en binario entre 4 se cortan los dos ultimos bits

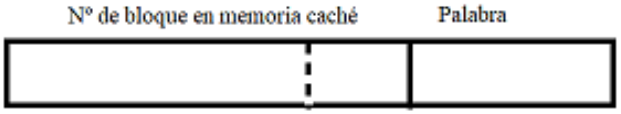
100/4 =100/2^2-->Por eso se cortan 2 bits, porque está elevado a eso el 2 por el que hay que dividir

Al hacer el corte queda
A la izquierda el cociente (el bloque)
A la derecha el resto(la palabra, lo que en realidad nos pide el procesador)

**** La palabra no es necesaria pero está mejor

25%2 (es 25 porque ya se usó el bloque para calcular y el dos por que son los conjuntos que hay)
Para esto se corta un bit (porque es % 2^1) y miramos el bit cortado.

Direc MP	bloque (palabra) *****	Posi MC
100 -->01100100	25(0)	1
17 --> 00010001	4(1)	0
79 --> 01001111	19(3)	1
50 --> 00110010	12(2)	0



** En la totalmente asociativa no hace falta hacer el corte para la pos C porque se puede poner en cualquier posición disponible

Pos en C

EL Tag(que es el identificador de bloque en la caché) puede llegar a ser muy largo por lo que alguna veces se puede acortar dependiendo del tipo de caché.
En la totalmente asociativa necesitamos guardar el número completo de bloque. No puede haber TAGs repetidas.
En la de conjunto no necesitamos guardar el bits/bits que cortamos para identificar el conjunto ya que sé que el número de conjunto donde lo pusimos sería siempre el último o últimos del número de bloque original. Se pueden repetir tantas veces los TAGs como conjuntos haya.
En la directa tampoco necesitamos guardar el número que nos indique en qué posición guardaremos el bloque ya que será siempre el mismo y corresponderá a los último/s números del bloque. Se pueden repetir tantas veces los TAGs como espacios en la caché haya.

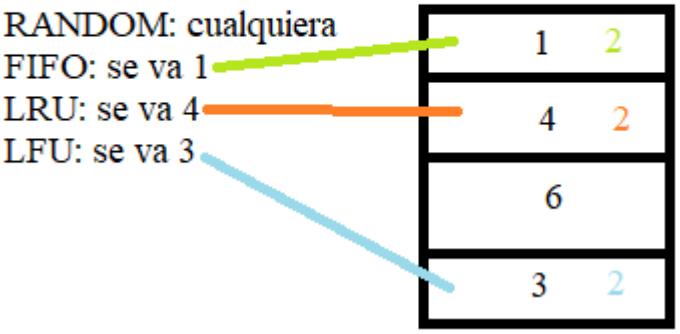
Algoritmos de reemplazos

No todas las cachés necesitan de estos algoritmos, las directas porque sabemos donde tiene que ir cada bloque donde se tendría que reemplazar el bloque que esté en el hueco de otro bloque que vaya a usar.

Tipos de algoritmos:

- FIFO(first input, first output): Se va la más antigua.
- LRU(least recently used): Se va la que hace más tiempo que no se usa
- RANDOM: aleatorio (no se preguntan en exámenes porque siempre se acierta)
- LFU(least frequently used): Se va la que menos veces se usa

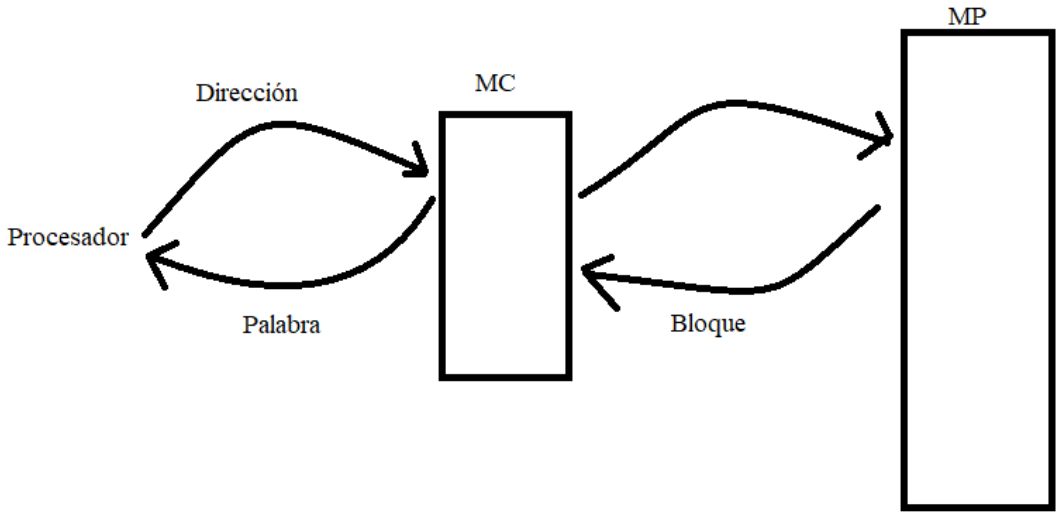
Bloques: 1, 4, 6, 3, 3, 3, 6, 1, 1, 6, 1, 2
F F F F A A A A A A A F



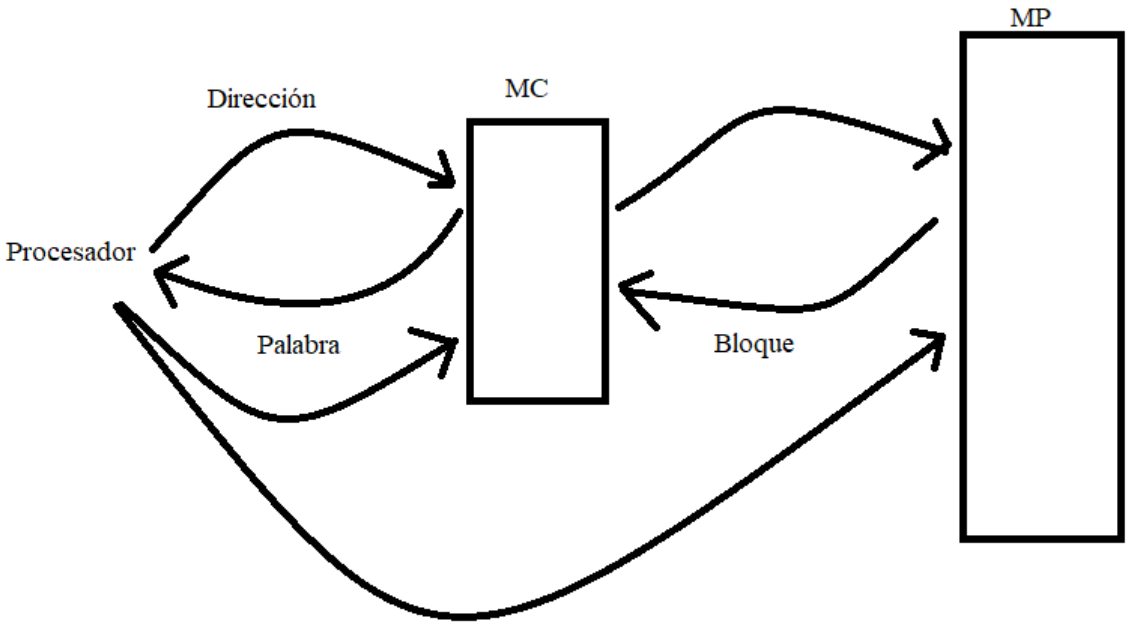
Tipos de acceso a memorias desde procesador.

Los procesadores al acceder a memoria solo pueden hacer Lectura o Escritura.

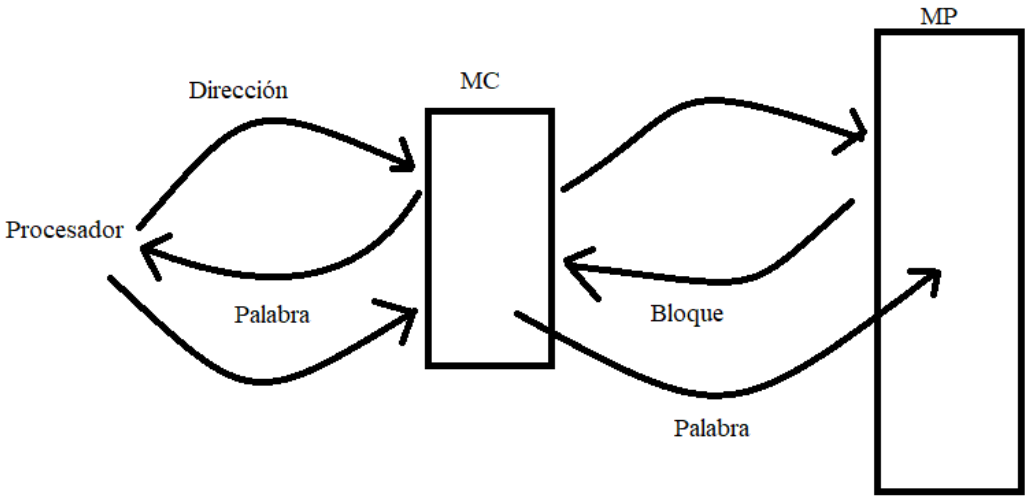
- ➔ Lectura: Tiene prioridad ante las escrituras



- ➔ Escritura:
  (WT) Escritura directa: Al devolver la palabra a escribir se reescribe lo que sea necesario en el mismo paso
  Más coherencia entre datos
  Más pesado el tráfico entre el procesador y MC,MP.



(WB)Post-escritura: Se reescribe en caché y ya cuando se el bloque reemplazado en caché se reescribirá en La MP
Menos pesado el tráfico entre el procesador y MC,MP
Hay menos coherencia ya que si otro procesador (no el que estemos usando porque ese accedería a MC)accede a la MP antes de corregirse con la nueva escritura.



LOOP
100
17
79
50
END LOOP

	*1	*2	*3
	dirección MP -->	bloques MP -->	posi MC --> A/F
100	25	1	F A
17	4	4	F F
79	19	3	F A
50	12	4	F F

		*4 Conjunto	
100	25	1	F A
17	4	0	F A
79	19	1	F A
50	12	0	F A

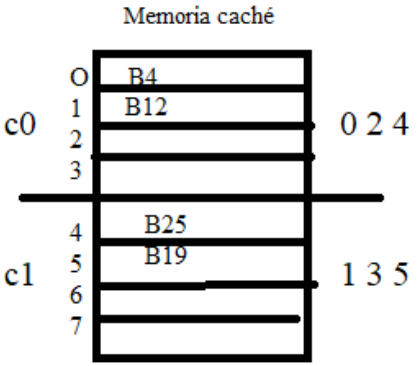
*1 se divide entre el número de instrucciones de cada bloque(i%capacidad bloque)

*2 depende del tipo de cache porque en directa depende del número de bloque (i%num de bloques de cache)

*3 Si no está en el caché o hay que sustituir se considera fallo sino acierto

*4 Para saber a que conjunto i%num conjuntos

Cache directa



Asociatica por conjunto
4-ways(--> 2 conjuntos)

En binario

Para dividir en binario entre 4 se cortan los dos ultimos bits

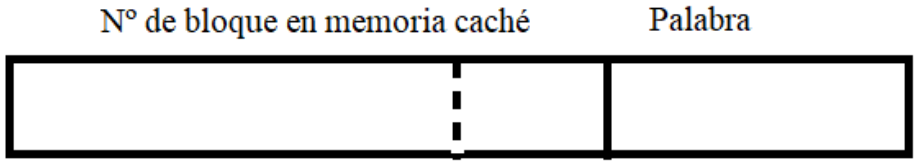
100/4 =100/2^2-->Por eso se cortan 2 bits, porque está elevado a eso el 2 por el que hay que dividir

Al hacer el corte queda
A la izquierda el cociente (el bloque)
A la derecha el resto(la palabra, lo que en realidad nos pide el procesador)

**** La palabra no es necesaria pero está mejor

25%2 (es 25 porque ya se usa el bloque para calcular y el dos por que son los conjuntos que hay)
Para esto se corta un bit (porque es % 2^1) y miramos el bit cortado.

Direc MP	bloque (palabra) *****	Posi MC
100 -->01100100	25(0)	1
17 --> 00010001	4(1)	0
79 --> 01001111	19(3)	1
50 --> 00110010	12(2)	0



** En la totalmente asociatiba no hace falta hacer el corte para la pos C porque se puede poner en cualquier posición disponible