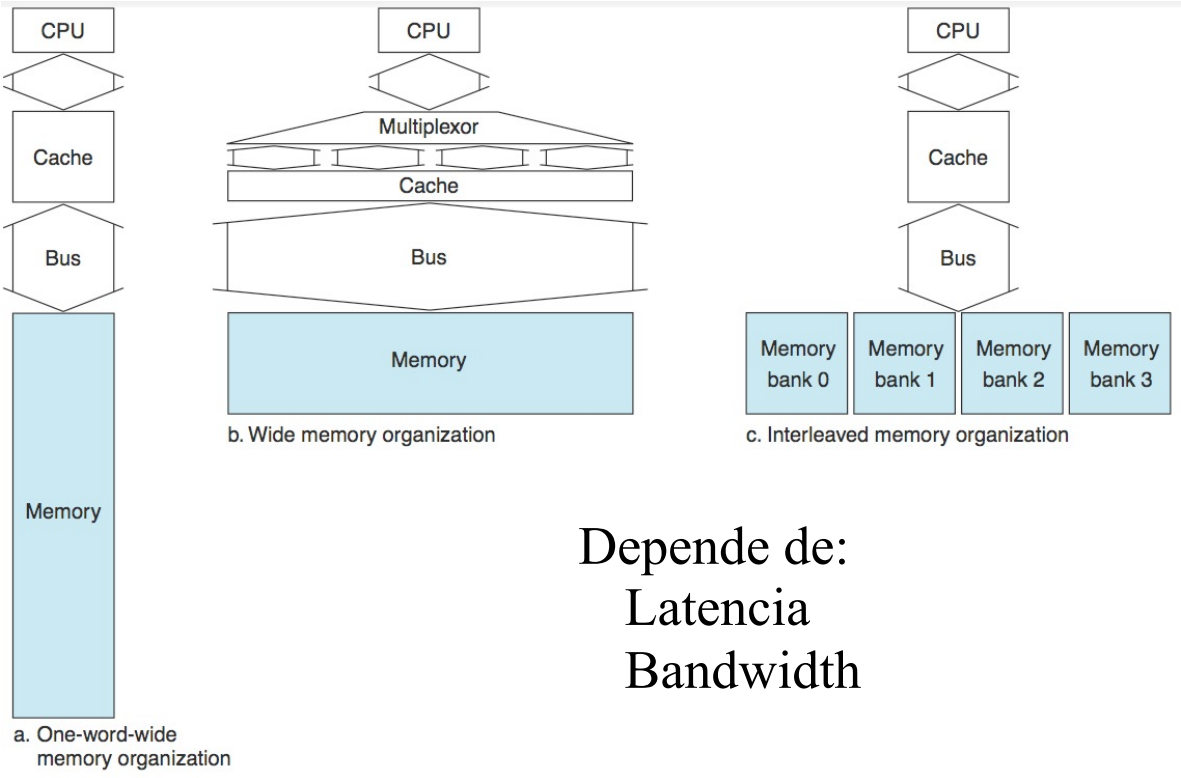
Miss rate vs. Tamaño de bloque

- En general el MR baja cuando se aumenta el tamaño del bloque.
- Ejemplo, bloque de una palabra vs bloque de cuatro palabras.
- Existe una relación casi directa entre el aumento del bloque y la reducción de miss. Especialmente en el código (localidad).
- Pero el MR aumenta cuando el tamaño de bloque llega a ser una fracción significativa de la cache completa.
- Un problema más serio es que si el TB aumenta, también aumenta la penalidad por miss. (recuperar bloque y cargarlo)

Latencia de la primer palabra + transferencia del resto del bloque

Demora por miss de cache



Depende de:
Latencia
Bandwidth

Desempeño con cache

$$\text{Tiempo CPU}=(\text{Ciclos CPU} + \text{Ciclos stall Memoria}) \text{ X Ciclo}$$

$$\text{Ciclos stall Memoria}=(\text{Ciclos stall lectura} + \text{Ciclos stall escritura})$$

$$Ciclos\ stall\ lectura = \frac{Lecturas}{Programa} \times Tasa\ fallos\ lectura \times Penalidad\ fallo\ lectura$$

$$Ciclos\ stall\ escritura = (\frac{Escrituras}{Programa} \times Tasa\ fallos\ escritura \times Penalidad\ fallo\ escritura) + Stall\ buffer\ escritura$$

Combinando lecturas y escrituras

$$Ciclos\ stall\ memoria = \frac{Accesos\ a\ memoria}{Programa} \times Tasa\ fallos \times Penalidad\ fallos$$

$$Ciclos\ stall\ memoria = \frac{Instrucciones}{Programa} \times \frac{Fallos}{Instrucciones} \times Penalidad\ fallos$$

Estimando el desempeño con fallos de cache

CPI=2, ld+st=36%, m.penalty=100,
m.r inst=2%, m.r datos=4%

¿Cuánto más rápido es el procesador si no hay miss?

Ciclos Miss Instrucciones= I X 2% X 100 = 2 X I

Ciclos Miss Datos = I X 36% X 4% X 100 = 1,44 X I

$$\frac{Tiempo\ CPU\ con\ stall}{Tiempo\ CPU\ con\ cache\ perfecta} = \frac{I \times CPI_{stall} \times Ciclo\ Reloj}{I \times CPI_{perfecta} \times Ciclo\ Reloj}$$

$$\frac{CPI_{stall}}{CPI_{perfecta}} = \frac{5,44}{2} = 2,72$$

Tiempo promedio de acceso a memoria

- El desempeño del procesador es afectado por el tiempo de acceso a los datos, tanto en hit como en miss.
- Average Memory Access Time, es el tiempo promedio para acceder a la memoria considerando hits y misses y la frecuencia de los diferentes accesos:

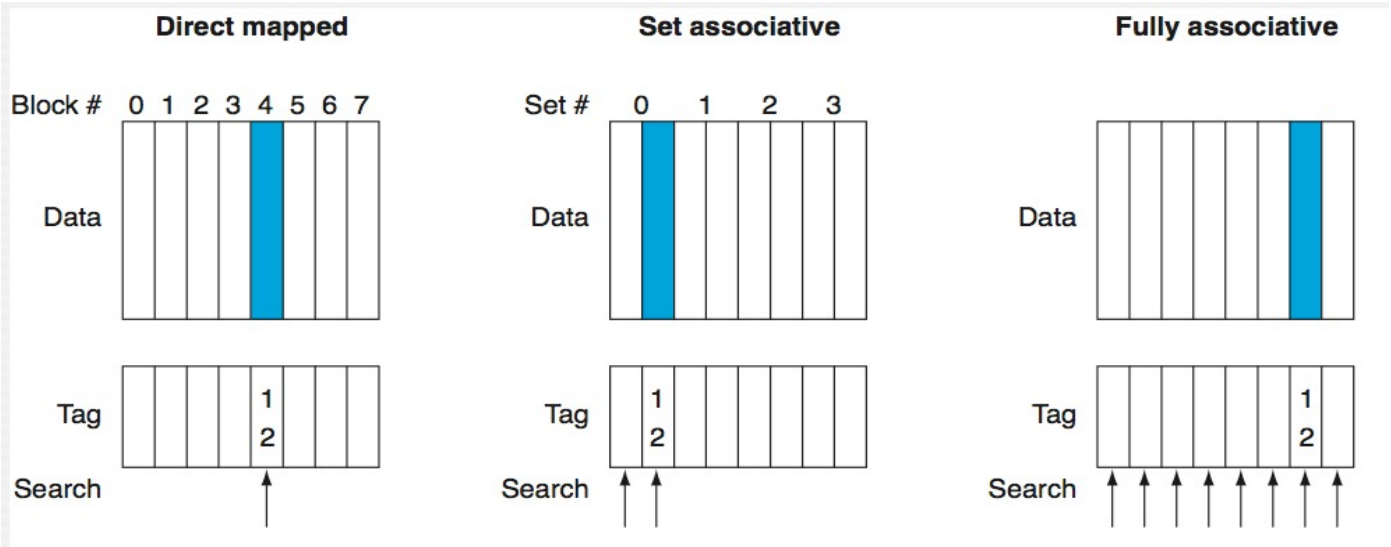
AMAT=Tiempo de hit + Tasa de fallo X Penalidad por miss

Ejemplo: Cálculo de AMAT

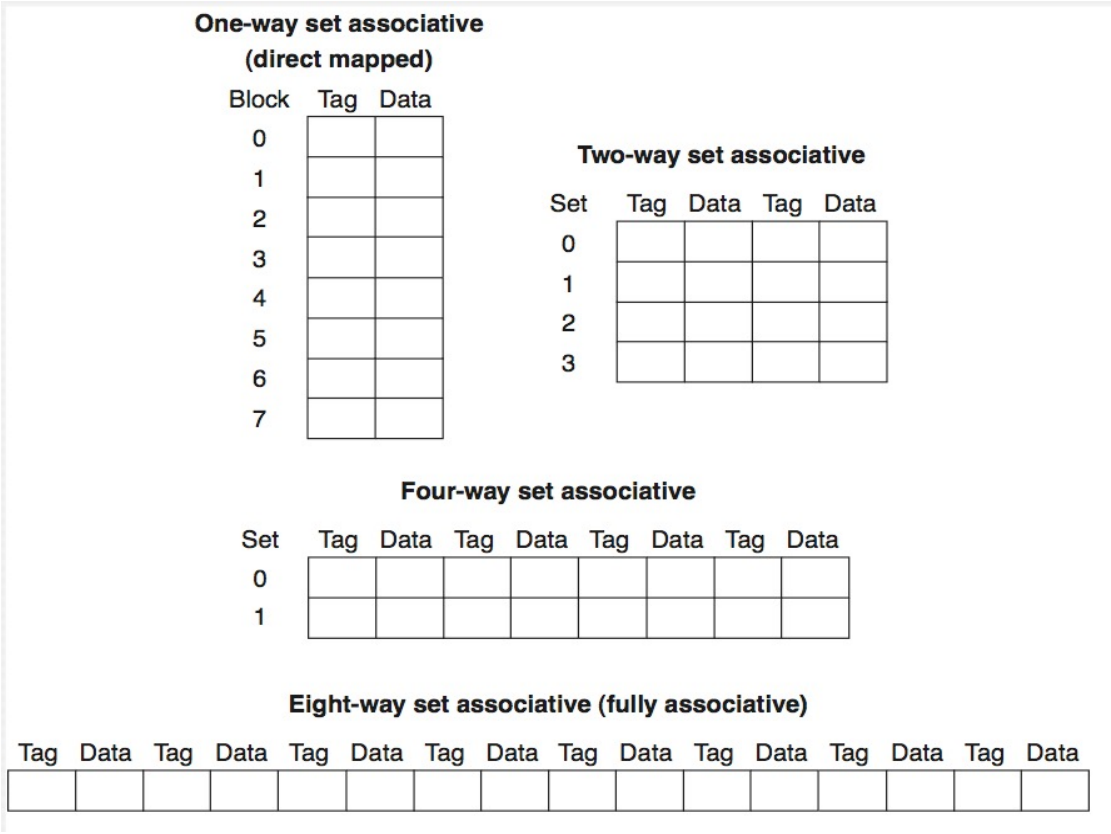
- Tiempo de ciclo de reloj = 1ns
- Penalidad por fallo = 20 ciclos
- Tasa de fallo 0,05 misses por instrucción
- Tiempo de acceso a cache = 1ns
- La penalidad por lectura o escritura es la misma.

$$\text{AMAT} = \text{Tiempo de hit} + \text{Tasa de fallo X Penalidad por miss}$$
$$= 1 + 0,05 \times 20$$
$$= 2 \text{ ciclos de reloj}$$

Ubicación flexible de los bloques



Asociatividad para 8 bloques



Ejemplo:misses y asociatividad

- Tres caches con cuatro bloques de una palabra.
- Caso Uno: Asociativa
- Caso Dos: Dos vías
- Caso Tres: Correspondencia directa
- Secuencia de direcciones de bloques: 0, 8, 0, 6, 8
- Para cada caso ¿Cuántos misses?

Ejemplo:misses y asociatividad. Correspondencia directa

Dirección del bloque	Bloque de cache
0	(0 modulo 4)=0
6	(6 modulo 4)=2
8	(8 modulo 4)=0

Dir. de mem. del bloque	Hit o miss	Contenidos de bloques de cache después de la referencia			
		0	1	2	3
0	miss	memoria[0]			
8	miss	memoria[8]			
0	miss	memoria[0]			
6	miss	memoria[0]		memoria[6]	
8	miss	memoria[8]		memoria[6]	

Ejemplo:misses y asociatividad. Conjuntos asociativos de 2 vías

Dirección del bloque	Conjunto en la cache
0	(0 modulo 2)=0
6	(6 modulo 2)=0
8	(8 modulo 2)=0

Dir. de mem. del bloque	Hit o miss	Contenidos de bloques de cache después de la referencia			
		Conjunto 0	Conjunto 0	Conjunto 1	Conjunto 1
0	miss	memoria[0]			
8	miss	memoria[0]	memoria[8]		
0	hit	memoria[0]	memoria[8]		
6	miss	memoria[0]	memoria[6]		
8	miss	memoria[8]	memoria[6]		

Ejemplo:misses y asociatividad. Totalmente Asociativo

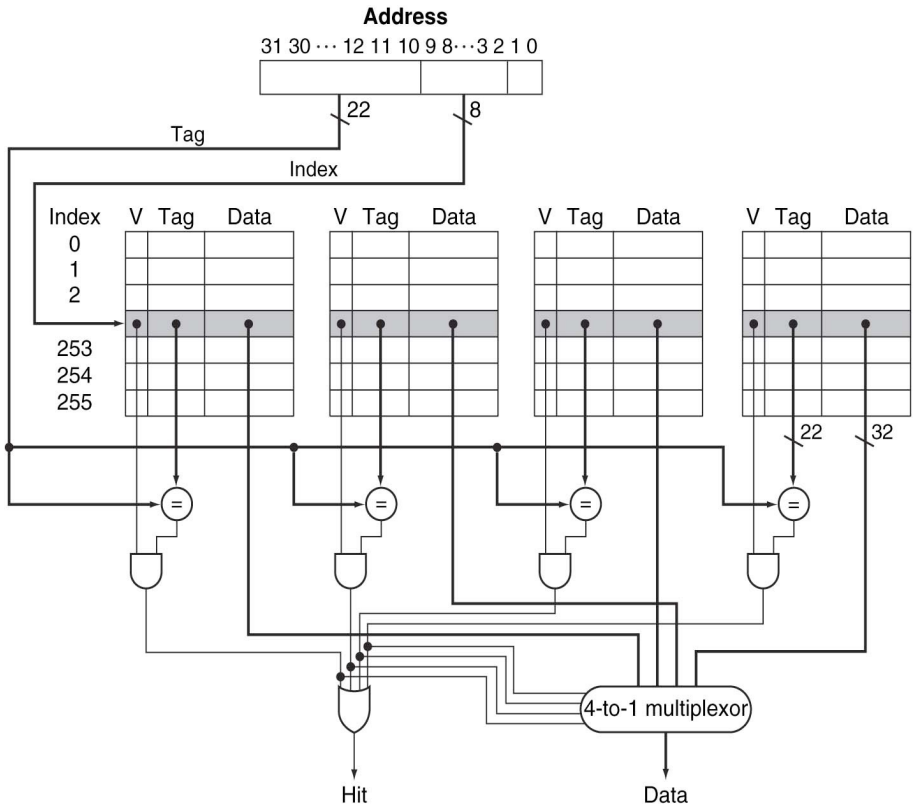
Los bloques de memoria se pueden ubicar en cualquier bloque de cache

Dir. de mem. del bloque	Hit o miss	Contenidos de bloques de cache después de la referencia			
		0	1	2	3
0	miss	memoria[0]			
8	miss	memoria[0]	memoria[8]		
0	hit	memoria[0]	memoria[8]		
6	miss	memoria[0]	memoria[8]	memoria[6]	
8	hit	memoria[0]	memoria[8]	memoria[6]	

Organización asociativa: características

- La búsqueda del tag en el conjunto se hace en paralelo.
- A mayor asociatividad mayor costo para buscar.
- Totalmente Asociativa es un único conjunto.
- En correspondencia directa hay un único comparador.

Conjunto asociativo de cuatro vías



Bloque a reemplazar

- La cache asociativa debe elegir donde ubicar el bloque requerido
- Totalmente asociativa se puede reemplazar cualquier bloque.
- Conjunto Asociativo se puede reemplazar dentro del conjunto.
- LRU (least recently used) es comúnmente usado.
  - reemplaza el que no ha sido usado por más tiempo.
  - se registra cuando se usa cada elemento.
  - con dos vías se puede usar un bit por conjunto.
  - mayor asociatividad => mayor complejidad

Ej: Tamaño de tag vs Asociatividad

- Cache de 4096 bloques. Bloques de 4 palabras.
- Dirección de 32 bits.
- ¿Cuántos conjuntos?¿Cuántos bits de tag?

Correspondencia directa	tag	índice	Desplazamiento en el bloque
	16	12	4
Conjunto asociativo de dos vías	tag	conjunto	Desplazamiento en el bloque
	17	11	4
Conjunto asociativo de cuatro vías	tag	conjunto	Desplazamiento en el bloque
	18	10	4
Totalmente asociativa	tag		Desplazamiento en el bloque
	28		4

Ej: Tamaño de tag vs Asociatividad Correspondencia directa

- Son 4096 conjuntos
- Tiene 12 bits de índice, (2^{12}=4096).
- Tag de 16 bits. Total 16 X 4096 = 2^{16} bits = 64 Kib tags

Correspondencia directa	tag	índice	Desplazamiento en el bloque
	16	12	4

Ej: Tamaño de tag vs Asociatividad Conjunto asociativo de dos vías

- Son 2048 conjuntos
- Tiene 11 bits de índice, (2^{11}=2048).
- Tag de 17 bits. Total 17 X 2 X 2048 = 68 Kib para tags

Conjunto asociativo de dos vías

tag	conjunto	Desplazamiento en el bloque
17	11	4

Ej: Tamaño de tag vs Asociatividad Conjunto asociativo de cuatro vías

- Son 1024 conjuntos
- Tiene 10 bits de índice, (2^{10}=1024).
- Tag de 18 bits. Total 18 X 4 X 1024 = 72 Kib para tags

Conjunto asociativo de cuatro vías

tag	conjunto	Desplazamiento en el bloque
18	10	4

Ej: Tamaño de tag vs Asociatividad Totalmente asociativa

- Es 1 conjunto de 4096 elementos.
- Tiene 0 bits de índice.
- Tag de 28 bits. Total 28 X 1 X 4096 = 112 Kib para tags

Totalmente asociativa	tag	Desplazamiento en el bloque
	28	4

Cache multinivel

- Muchos microprocesadores cuentan con un nivel adicional de cache.
- El segundo nivel es accedido si el primer nivel falla.
- La penalidad del primer nivel es el tiempo de acceso al segundo nivel, siempre que el dato esté allí.
- Si el segundo nivel también falla, la penalidad será el acceso a la memoria principal.

Ej: Desempeño de cache multinivel

- Un procesador con CPI = 1, todos hits en primer nivel, reloj de 4 GHz.
- Penalidad de acceso a memoria principal 100ns.
- Tasa de fallo por instrucción 2% en cache primaria.
- Calcular la mejora agregando una cache secundaria de 5ns con una tasa de fallo de 0,5%

Ej: Desempeño de cache multinivel

- Penalidad por acceder a la memoria principal:

$$\frac{100\ ns}{0{,}25\frac{ns}{ciclos\ \ de\ \ reloj}} = 400\ ciclos\ de\ reloj$$

- CPI con un solo nivel de cache:
  - CPI total = CPI base + ciclos stall instrucción.
  - CPI total = 1 + 2% X 400 = 9

Ej: Desempeño de cache multinivel

- Penalidad por acceder al segundo nivel:

$$\frac{5\ ns}{0,25\frac{ns}{ciclos\ de\ reloj}} = 20\ ciclos\ de\ reloj$$

- CPI con dos niveles de cache:
  - CPI total=1+stall primaria+stall secundaria
  - =1 + 2% X 20 + 0,5% X 400 = 1 + 0,4 + 2,0 = 3,4

- Ganancia con dos niveles $= \frac{9,0}{3,4} = 2,6$

Diseño de cache de dos niveles

- Las consideraciones son diferentes para cada nivel.
- La cache primaria busca reducir el tiempo de hit.
- La cache secundaria reduce la tasa de miss.
- La Primaria es más chica:
  - tamaño de bloque pequeño.
  - reduce la penalidad por miss.
- La Secundaria es mucho más grande:
  - tiempo de acceso no crítico.
  - tamaño de bloque grande.
  - mayor asociatividad.

¿Qué vimos?

- Desempeño con fallas de cache. AMAT
- Ubicación flexible de los bloques.
- Ubicación de un bloque en cache.
- Reemplazo de bloques.
- Cache multinivel.