



# **Memoria Cache**

## Optimizaciones

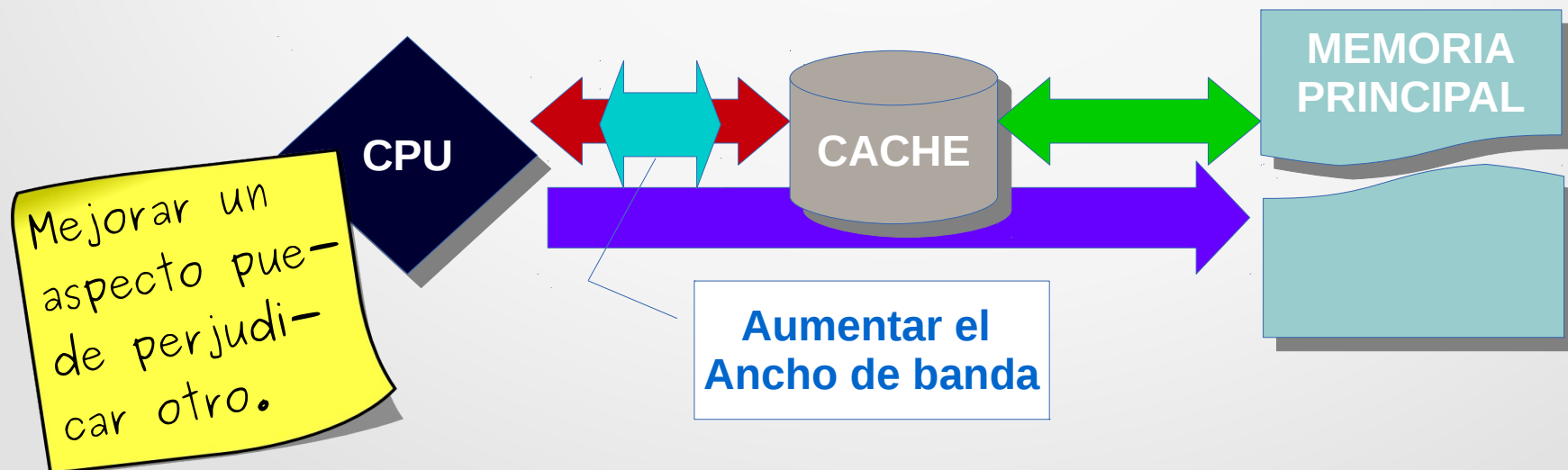
# Tipos de misses

- **Compulsivos:** son los que se ocasionan en la primera vez que un bloque es referenciado.
- **Capacitivos:** son los que se producen si la cache no puede contener a todos los bloques que se necesitan para la ejecución de un programa.
- **Conflictivos:** si se utiliza mapeo directo o set asociativo, se producen (además) misses debido a que muchos bloques mapean al mismo set, lo que genera que se descarten bloques que son requeridos posteriormente.

# Optimizaciones

¿Cómo podemos reducir el tiempo promedio de acceso a memoria caché?

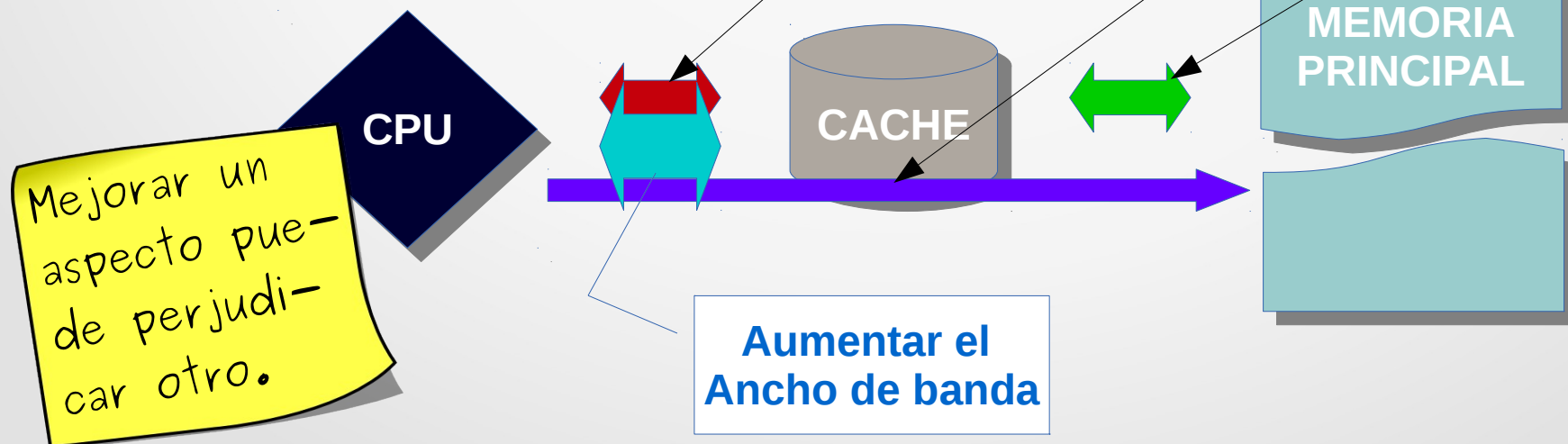
$$\text{Average Memory Access time} = \text{Hit time} + \text{Miss rate} * \text{Miss penalty}$$



# Optimizaciones

¿Cómo podemos reducir el tiempo promedio de acceso a memoria caché?

$$\text{Average Memory Access time} = \text{Hit time} + \text{Miss rate} * \text{Miss penalty}$$



# Optimizaciones

## A) Reducir el *Miss Rate*

- Mayor Tamaño de Cache
- Mayor Tamaño de Bloque
- Mayor Asociatividad
- Victim Caches

## B) Reducir el *Miss Penalty*

- Caches Multinivel
- Prioridad a los reads miss
- ER y CWF
- Prefetching de información

## C) Reducir el *Hit Time*

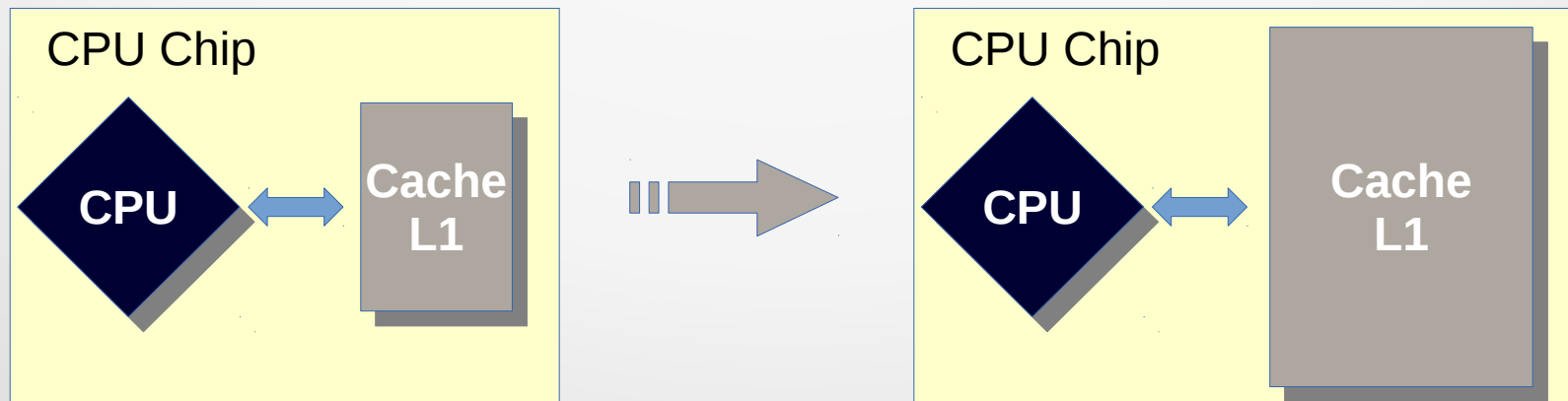
- Caches Simples
- Way Prediction

## D) Aumentar el *Ancho de Banda*

- Non Blocking Caches
- Multibanked Caches
- Pipelined Caches

# Optimizaciones: Reduciendo el Miss Rate

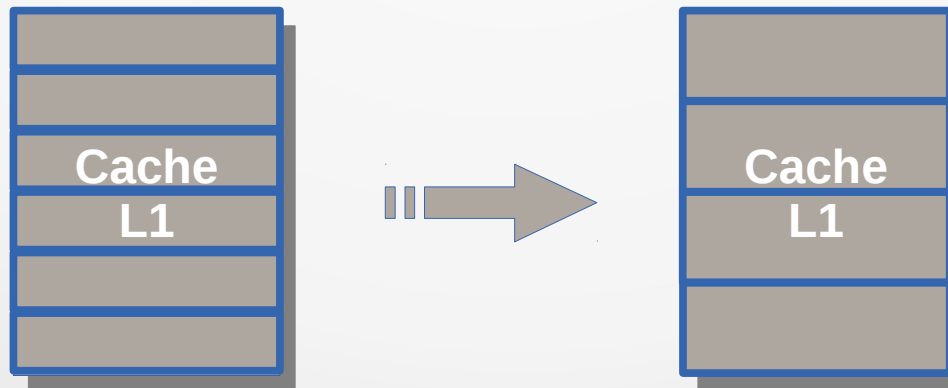
- **Mayor tamaño de cache** para reducir el miss rate
  - ¿Qué tipos de misses estamos reduciendo?
    - Capacitivos
  - ¿Qué desventaja presenta?
    - Mayor hit time
    - Mayor costo y consumo



# Optimizaciones: Reduciendo el Miss Rate

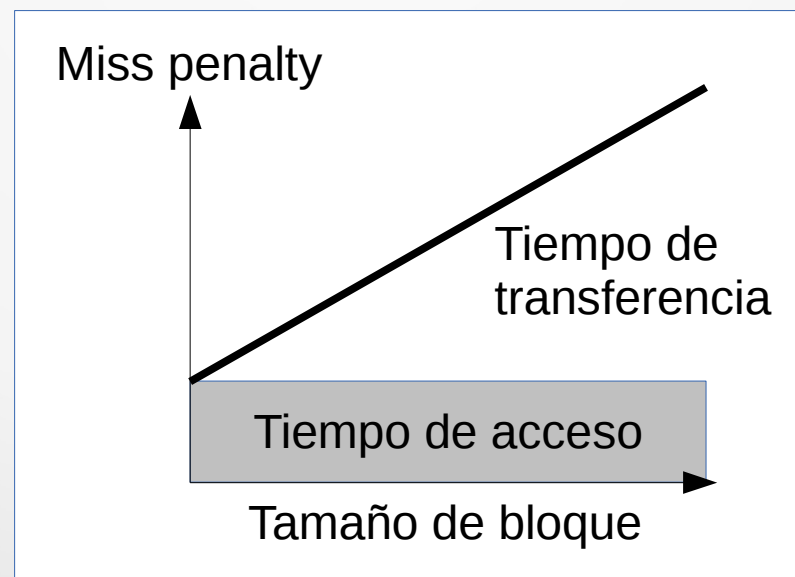
- **Bloques más grandes** para reducir el miss rate
  - ¿Qué tipos de misses reducimos y que localidad favorecemos?
    - ↓ Misses compulsivos (localidad espacial)

Manteniendo  
el tamaño  
de cache



# Optimizaciones: Reduciendo el Miss Rate

- **Bloques más grandes** para reducir el miss rate
  - ¿Qué tipos de misses reducimos y que localidad favorecemos?
    - ↓ Misses compulsivos (localidad espacial)
  - ¿Qué sucede con el *Miss Penalty*?

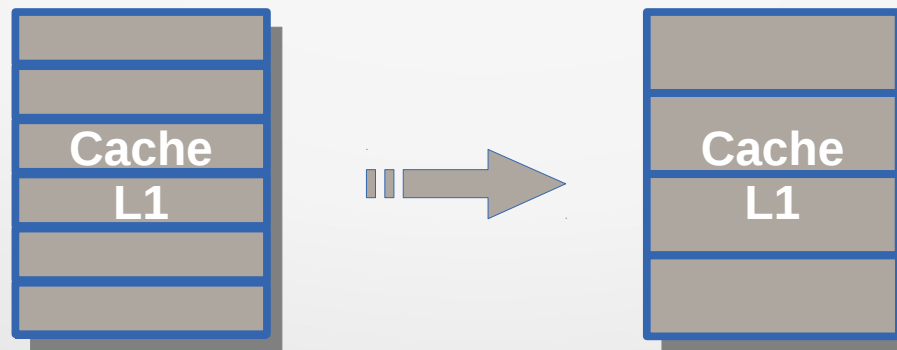




# Optimizaciones: Reduciendo el Miss Rate

- **Bloques más grandes** para reducir el miss rate
  - ¿Qué tipos de misses reducimos y que localidad favorecemos?
    - ↓ Misses compulsivos (localidad espacial)
  - ¿Que sucede con el *Miss Penalty*?
    - ↑ Miss penalty
  - ¿Qué tipo de misses puede empeorar y porqué?
    - ↑ Conflict misses

Manteniendo  
el tamaño  
de cache

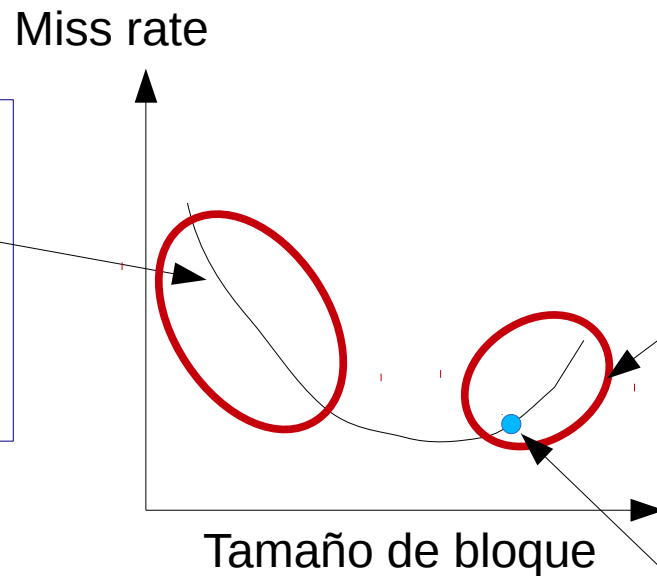


# Optimizaciones: Reduciendo el Miss Rate

- **Tamaño de bloque** y Miss rate

Manteniendo  
el tamaño  
de cache

A medida que pasamos de bloques pequeños a bloques grandes aprovechamos la **localidad espacial**, el miss rate disminuye.

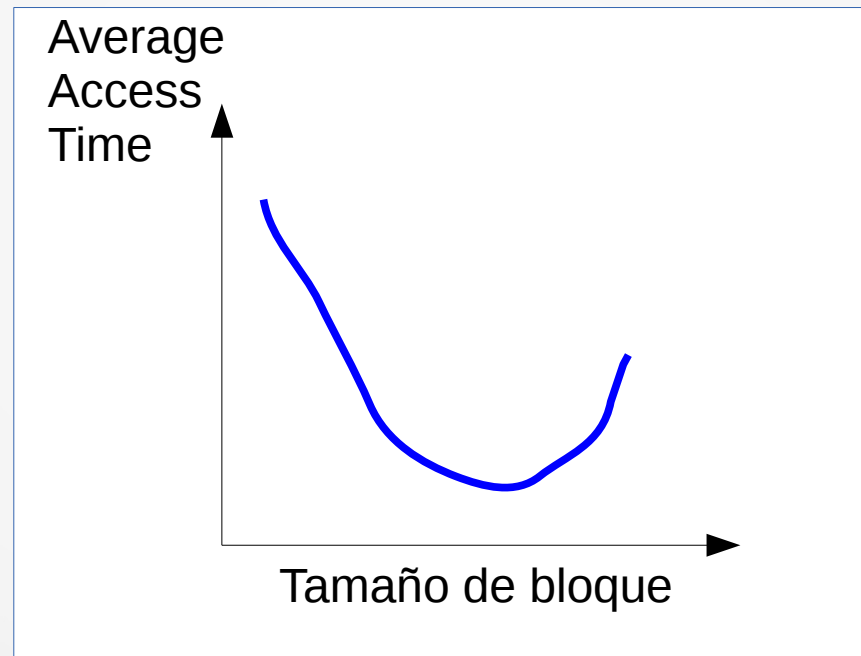


Cuanto más grande es el bloque, menos bloques entran en cache. Aumentarán los conflictos (los bloques serán reemplazados con más frecuencia) comprometiendo la **localidad temporal**.

**Polution point**

# Optimizaciones: Reduciendo el Miss Rate

- **Tamaño de bloque** y Average access time



$$\text{Average Memory Access time} = \text{Hit time} + \text{Miss rate} * \text{Miss penalty}$$

# Optimizaciones: Reduciendo el Miss Rate

- **Mayor asociatividad** para reducir el miss rate
  - ¿Qué tipos de Misses estamos reduciendo?
    - Conflictivos
  - Dos reglas empíricas:
    - 8-way ~ full asociativo
    - Regla 2:1
  - ¿Qué desventajas podemos observar?
    - ↑ Hit Time

# Optimizaciones: Reduciendo el Miss Rate

Cache size (KB)	Degree associative	Total miss rate	Miss rate components (relative percent) (sum = 100% of total miss rate)					
			Compulsory		Capacity		Conflict	
4	1-way	0.098	0.0001	0.1%	0.070	72%	0.027	28%
4	2-way	0.076	0.0001	0.1%	0.070	93%	0.005	7%
4	4-way	0.071	0.0001	0.1%	0.070	99%	0.001	1%
4	8-way	0.071	0.0001	0.1%	0.070	100%	0.000	0%
8	1-way	0.068	0.0001	0.1%	0.044	65%	0.024	35%
8	2-way	0.049	0.0001	0.1%	0.044	90%	0.005	10%
8	4-way	0.044	0.0001	0.1%	0.044	99%	0.000	1%
8	8-way	0.044	0.0001	0.1%	0.044	100%	0.000	0%
16	1-way	0.049	0.0001	0.1%	0.040	82%	0.009	17%
16	2-way	0.041	0.0001	0.2%	0.040	98%	0.001	2%
16	4-way	0.041	0.0001	0.2%	0.040	99%	0.000	0%
16	8-way	0.041	0.0001	0.2%	0.040	100%	0.000	0%
32	1-way	0.042	0.0001	0.2%	0.037	89%	0.005	11%
32	2-way	0.038	0.0001	0.2%	0.037	99%	0.000	0%
32	4-way	0.037	0.0001	0.2%	0.037	100%	0.000	0%
32	8-way	0.037	0.0001	0.2%	0.037	100%	0.000	0%
64	1-way	0.037	0.0001	0.2%	0.028	77%	0.008	23%

# Optimizaciones: Reduciendo el Miss Rate

Cache size (KB)	Degree associative	Total miss rate	Miss rate components (relative percent) (sum = 100% of total miss rate)					
			Compulsory		Capacity		Conflict	
4	1-way	0.098	0.0001	0.1%	0.070	72%	0.027	28%
4	2-way	0.076	0.0001	0.1%	0.070	93%	0.005	7%
4	4-way	0.071	0.0001	0.1%	0.070	99%	0.001	1%
4	8-way	0.071	0.0001	0.1%	0.070	100%	0.000	0%
8	1-way	0.068	0.0001	0.1%	0.044	65%	0.024	35%
8	2-way	0.049	0.0001	0.1%	0.044	90%	0.005	10%
8	4-way	0.044	0.0001	0.1%	0.044	99%	0.000	1%
8	8-way	0.044	0.0001	0.1%	0.044	100%	0.000	0%
16	1-way	0.049	0.0001	0.1%	0.040	82%	0.009	17%
16	2-way	0.041	0.0001	0.2%	0.040	98%	0.001	2%
16	4-way	0.041	0.0001	0.2%	0.040	99%	0.000	0%
16	8-way	0.041	0.0001	0.2%	0.040	100%	0.000	0%
32	1-way	0.042	0.0001	0.2%	0.037	89%	0.005	11%
32	2-way	0.038	0.0001	0.2%	0.037	99%	0.000	0%
32	4-way	0.037	0.0001	0.2%	0.037	100%	0.000	0%
32	8-way	0.037	0.0001	0.2%	0.037	100%	0.000	0%
64	1-way	0.037	0.0001	0.2%	0.028	77%	0.008	23%

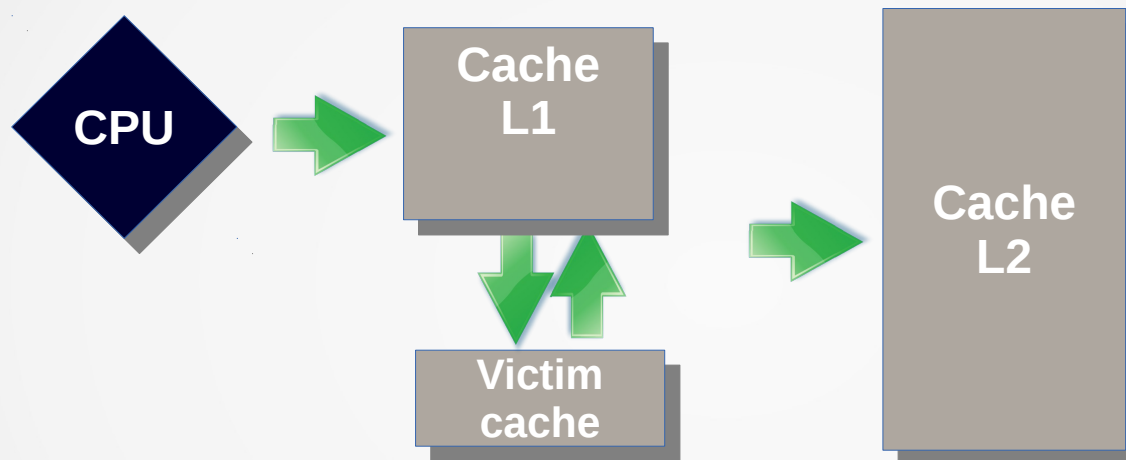
# Optimizaciones: Reduciendo el Miss Rate

Cache size (KB)	Degree associative	Total miss rate	Miss rate components (relative percent) (sum = 100% of total miss rate)					
			Compulsory		Capacity		Conflict	
4	1-way	0.098	0.0001	0.1%	0.070	72%	0.027	28%
4	2-way	0.076	0.0001	0.1%	0.070	93%	0.005	7%
4	4-way	0.071	0.0001	0.1%	0.070	99%	0.001	1%
4	8-way	0.071	0.0001	0.1%	0.070	100%	0.000	0%
8	1-way	0.068	0.0001	0.1%	0.044	65%	0.024	35%
8	2-way	0.049	0.0001	0.1%	0.044	90%	0.005	10%
8	4-way	0.044	0.0001	0.1%	0.044	99%	0.000	1%
8	8-way	0.044	0.0001	0.1%	0.044	100%	0.000	0%
16	1-way	0.049	0.0001	0.1%	0.040	82%	0.009	17%
16	2-way	0.041	0.0001	0.2%	0.040	98%	0.001	2%
16	4-way	0.041	0.0001	0.2%	0.040	99%	0.000	0%
16	8-way	0.041	0.0001	0.2%	0.040	100%	0.000	0%
32	1-way	0.042	0.0001	0.2%	0.037	89%	0.005	11%
32	2-way	0.038	0.0001	0.2%	0.037	99%	0.000	0%
32	4-way	0.037	0.0001	0.2%	0.037	100%	0.000	0%
32	8-way	0.037	0.0001	0.2%	0.037	100%	0.000	0%
64	1-way	0.037	0.0001	0.2%	0.028	77%	0.008	23%



# Optimizaciones: Reduciendo el Miss Rate

- Empleo de ***Victim cache*** para reducir el Miss Rate

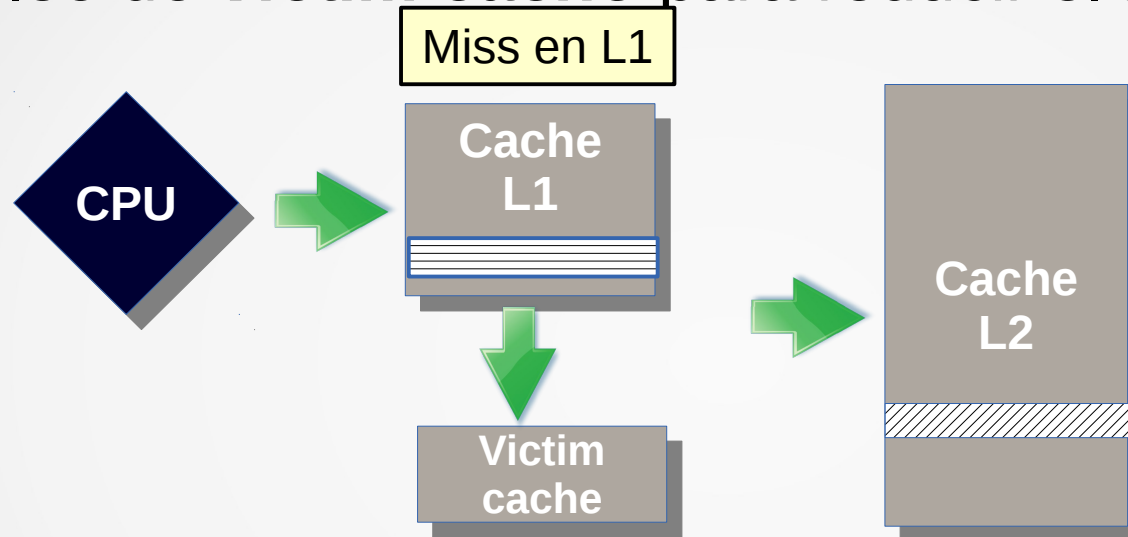


- L1 con MD + Victim cache Full asociativa



# Optimizaciones: Reduciendo el Miss Rate

- Empleo de ***Victim cache*** para reducir el Miss Rate



- L1 con MD + Victim cache Full asociativa

# Optimizaciones: Reduciendo el Miss Rate

- Empleo de ***Victim cache*** para reducir el Miss Rate



- L1 con MD + Victim cache Full asociativa

# Optimizaciones: Reduciendo el Miss Rate

- Empleo de ***Victim cache*** para reducir el Miss Rate



- L1 con MD + Victim cache Full asociativa

# Optimizaciones: Reduciendo el Miss Rate

- Empleo de ***Victim cache*** para reducir el Miss Rate



- L1 con MD + Victim cache Full asociativa
- ↓ Misses por conflicto

# Optimizaciones

## A) Reducir el *Miss Rate*

- Mayor Tamaño de Cache
- Mayor Tamaño de Bloque
- Mayor Asociatividad
- Victim Caches

## B) Reducir el *Miss Penalty*

- Caches Multinivel
- Prioridad a los reads miss
- ER y CWF
- Prefetching de información

## C) Reducir el *Hit Time*

- Caches Simples
- Way Prediction

## D) Aumentar el *Ancho de Banda*

- Non Blocking Caches
- Multibanked Caches
- Pipelined Caches

# Optimizaciones: Reduciendo el Miss Penalty

- **Caches Multi-nivel** para reducir el Miss Penalty

$$\begin{aligned} AMAT &= Hit\ time_{L1} + Miss\ rate_{L1} * Miss\ penalty_{L1} \\ &= Hit\ time_{L1} + Miss\ rate_{L1} * (Hit\ time_{L2} + Miss\ rate_{L2} * Miss\ penalty_{L2}) \end{aligned}$$

- **Miss Rate Local** a  $L_i$ : número de misses en  $L_i$  dividido por el número de accesos a  $L_i$ . Equivale al Miss Rate de  $L_i$ .
- **Miss Rate Global** para  $L_i$ : número de misses en  $L_i$  dividido por el total de accesos. Para dos niveles:  $Miss\ Rate\ L_1 \times Miss\ Rate\ L_2$ .

# Optimizaciones: Reduciendo el Miss Penalty

- ¿Cuál será el **Average Memory Access Time**? Si:
  - Cada **1000 referencias a memoria** hay **40 misses en L1** y **20 misses en L2**
  - $Hit\ Time_{L1} = 1$  ciclo
  - $Hit\ Time_{L2} = 10$  ciclos
  - $Miss\ Penalty_{L2} = 200$  ciclos

$$\begin{aligned} AMAT_{2niveles} &= Hit\ time_{L1} + Miss\ rate_{L1} * Miss\ penalty_{L1} \\ &= 1 + (40/1000) * (Hit\ time_{L2} + Miss\ rate_{L2} * Miss\ penalty_{L2}) \\ &= 1 + 0.04 * (10 + (20/40) * 200) \\ &= 1 + 0.04 * 110 = 5.4\ ciclos \end{aligned}$$

$$\begin{aligned} AMAT_{sin\ L2} &= 1 + (40/1000) * 200 \\ &= 1 + 0.04 * 200 = 9\ ciclos \end{aligned}$$

# Optimizaciones: Reduciendo el Miss Penalty

- ¿Cuál será el **Average Memory Access Time**? Si:

- Cada **1000 referencias a memoria** hay **40 misses en L1** y **20 misses en L2**
- $Hit\ Time_{L1} = 1$  ciclo
- $Hit\ Time_{L2} = 10$  ciclos
- $Miss\ Penalty_{L2} = 200$  ciclos

$$Miss\ Rate_{L1} = 4\%$$

$$Miss\ Rate\ Local_{L2} = 50\%$$

$$Miss\ Rate\ Global_{L2} = 2\%$$

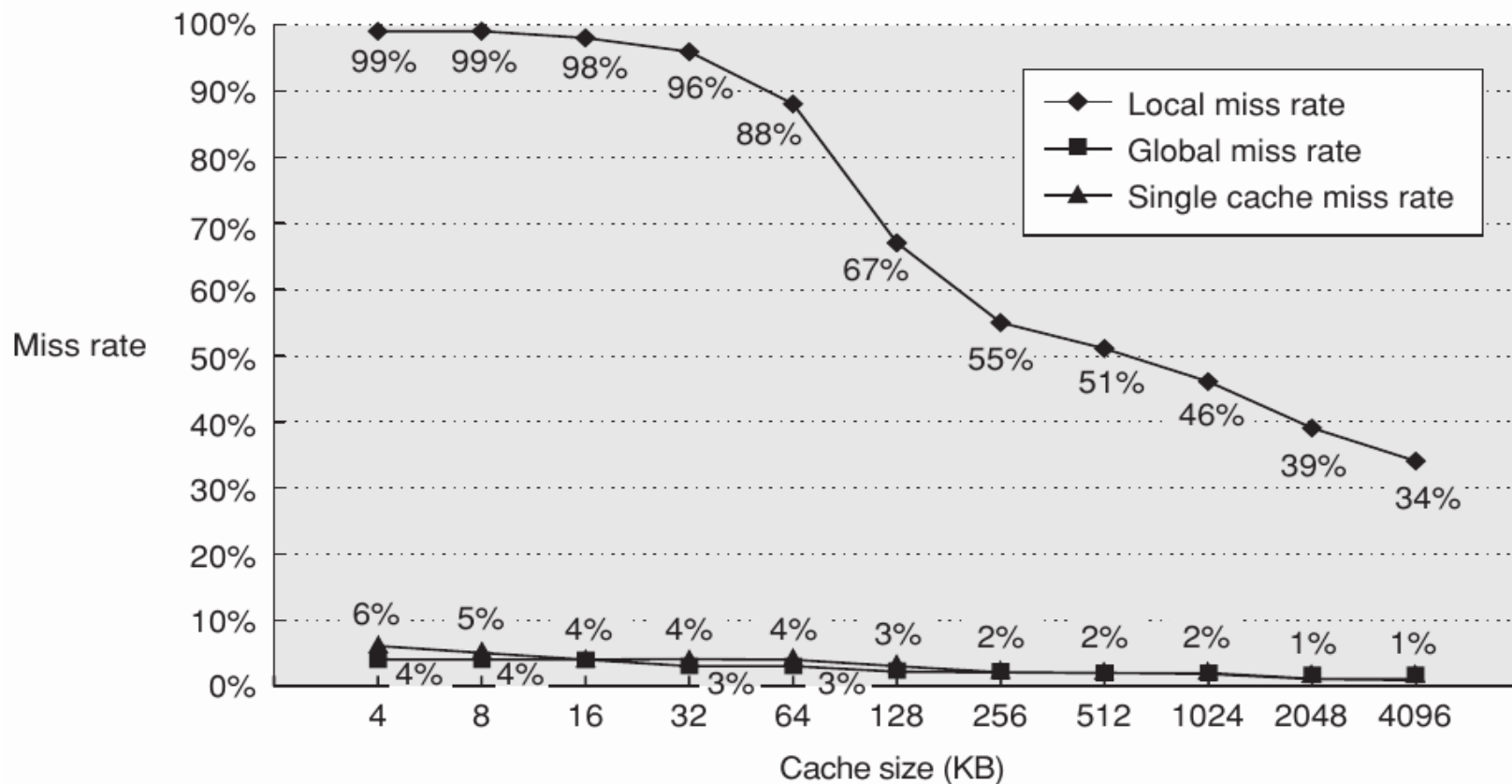
$$\begin{aligned} AMAT_{2niveles} &= Hit\ time_{L1} + Miss\ rate_{L1} * Miss\ penalty_{L1} \\ &= 1 + (40/1000) * (Hit\ time_{L2} + Miss\ rate_{L2} * Miss\ penalty_{L2}) \\ &= 1 + 0.04 * (10 + (20/40) * 200) \\ &= 1 + 0.04 * 110 = 5.4\ ciclos \end{aligned}$$

$$\begin{aligned} AMAT_{sin\ L2} &= 1 + (40/1000) * 200 \\ &= 1 + 0.04 * 200 = 9\ ciclos \end{aligned}$$



# Optimizaciones: Reduciendo el Miss Penalty

- **Caches Multi-nivel** para reducir el Miss Penalty



# Optimizaciones: Reduciendo el Miss Penalty

- **Caches Multi-nivel** para reducir el Miss Penalty
  - *Inclusión/Exclusión Multinivel:*
    - *Inclusión Multinivel:* Todo dato en  $L_1$  siempre se encuentra en  $L_2$ .
      - Reemplazo de un bloque de  $L_1$  por uno de  $L_2$ .
    - *Exclusión Multinivel:* Todo dato en  $L_1$  nunca se encuentra en  $L_2$ .
      - Swap de bloques entre  $L_1$  y  $L_2$ .
      - AMD Opteron


# Optimizaciones: Reduciendo el Miss Penalty

**CPU-Z**

**CPU** | Caches | Mainboard | Memory | SPD | Graphics | About

**Processor**

Name	Intel Core i7 3667U		
Code Name	Ivy Bridge	Max TDP	17 W
Package	Socket 988B rPGA		
Technology	22 nm	Core VID	0.866 V
Specification	Intel(R) Core(TM) i7-3667U CPU @ 2.00GHz		
Family	6	Model	A
Ext. Family	6	Ext. Model	3A
Instructions	MMX, SSE (1, 2, 3, 3S, 4.1, 4.2), EM64T, VT-x, AES, AVX		



**Clocks (Core #0)**

Core Speed	3195.72 MHz
Multiplier	x 32.0
Bus Speed	99.9 MHz
Rated FSB	

**Cache**

L1 Data	2 x 32 KBytes	8-way
L1 Inst.	2 x 32 KBytes	8-way
Level 2	2 x 256 KBytes	8-way
Level 3	4 MBytes	16-way

Selection: Processor #1 | Cores: 2 | Threads: 4

**CPU-Z** Version 1.61.5.x64 | Validate | OK


# Optimizaciones: Reduciendo el Miss Penalty

CPU-Z - ID : 2641492

CPU | Caches | Mainboard | Memory | SPD | Graphics | About

Processor

Name	AMD Opteron 6168		
Code Name	Magny-Cours	Brand ID	2
Package	Socket G34 (1974)		
Technology	45 nm	Core Voltage	0.744 V



Specification

AMD Opteron(tm) Processor 6168			
Family	F	Model	9
Ext. Family	10	Ext. Model	9
Stepping	1	Revision	HY-D1

Instructions: MMX(+), 3DNow!(+), SSE (1, 2, 3, 4A), x86-64, AMD-V

Clocks (Core #0)

Core Speed	1901.7 MHz
Multiplier	x 9.5
Bus Speed	200.17 MHz
Rated FSB	2602.26 MHz

Cache

L1 Data	12 x 64 KBytes	2-way
L1 Inst.	12 x 64 KBytes	2-way
Level 2	12 x 512 KBytes	16-way
Level 3	10240 KBytes	96-way

Selection: Processor #1

Cores: 12 Threads: 12

CPU-Z Version 1.62

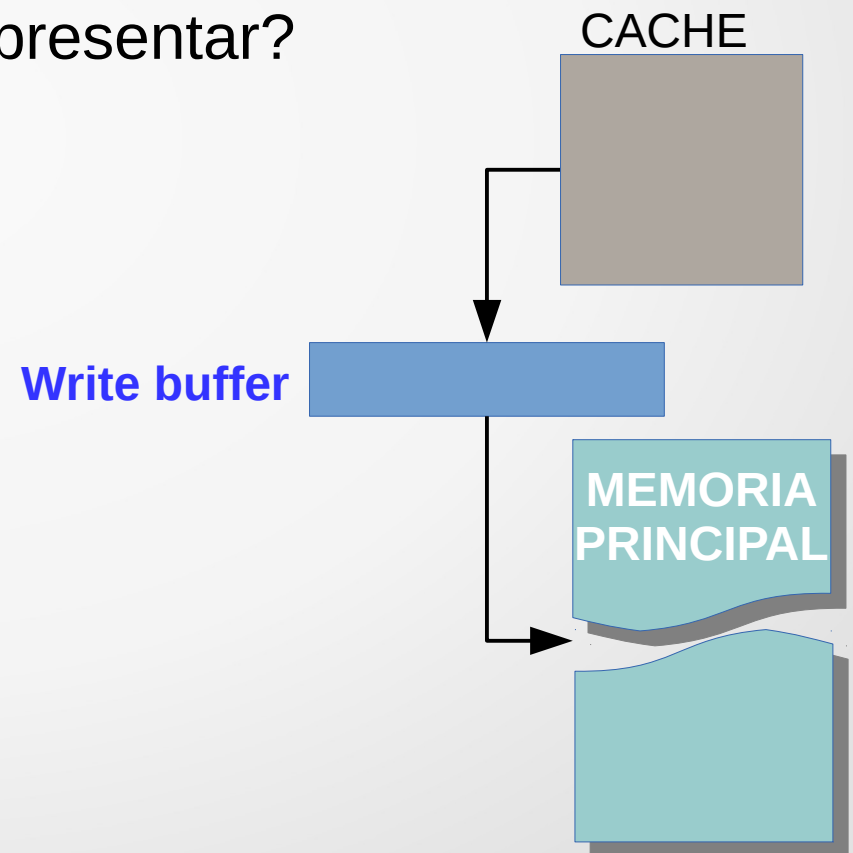
Validate OK

# Optimizaciones: Reduciendo el Miss Penalty

- Dar Prioridad a los **Read Miss** frente a los **Writes**
  - Mejora para writes con write-through → write buffer
  - ¿Qué inconveniente se puede presentar?

Si 512 y 1024 mapean al 0:

```
M[512] ← R3  
R1 ← M[1024]  
R2 ← M[512]
```



# Optimizaciones: Reduciendo el Miss Penalty

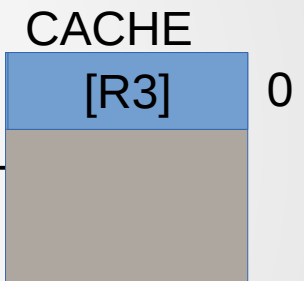
- Dar Prioridad a los **Read Miss** frente a los **Writes**
  - Mejora para writes con write-through → write buffer
  - ¿Qué inconveniente se puede presentar?

Si 512 y 1024 mapean al 0:

```
M[512] ← R3  
R1 ← M[1024]  
R2 ← M[512]
```

Write buffer

M[512] ← R3



MEMORIA PRINCIPAL

512

X

1024

Z

# Optimizaciones: Reduciendo el Miss Penalty

- Dar Prioridad a los **Read Miss** frente a los **Writes**
  - Mejora para writes con write-through → write buffer
  - ¿Qué inconveniente se puede presentar?

Si 512 y 1024 mapean al 0:

```
M[512] ← R3  
R1 ← M[1024]  
R2 ← M[512]
```

Read Miss

Write buffer

M[512] ← R3

CACHE  
Z 0

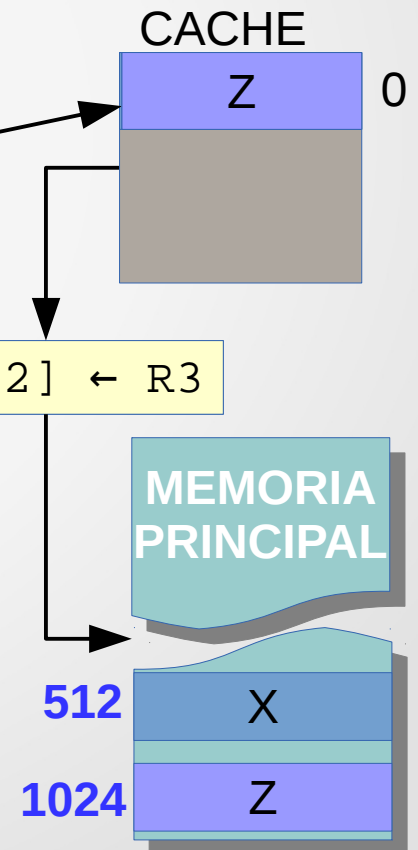
MEMORIA  
PRINCIPAL

512

X

1024

Z



# Optimizaciones: Reduciendo el Miss Penalty

- Dar Prioridad a los **Read Miss** frente a los **Writes**
  - Mejora para writes con write-through → write buffer
  - ¿Qué inconveniente se puede presentar?

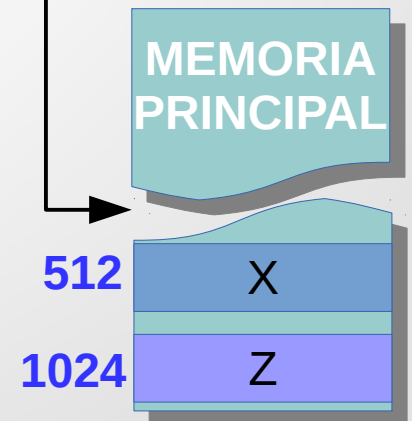
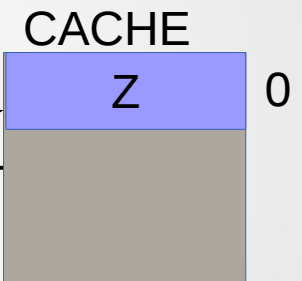
Si 512 y 1024 mapean al 0:

```
M[512] ← R3  
R1 ← M[1024]  
R2 ← M[512]
```

Read Miss

Write buffer

M[512] ← R3





# Optimizaciones: Reduciendo el Miss Penalty

- Dar Prioridad a los **Read Miss** frente a los **Writes**
  - Mejora para writes con write-through → write buffer
  - ¿Qué inconveniente se puede presentar?

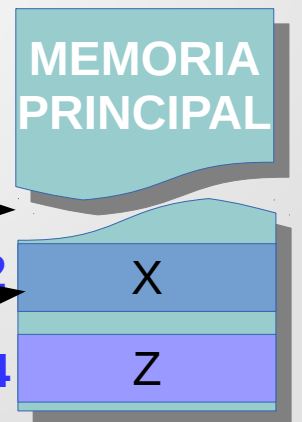
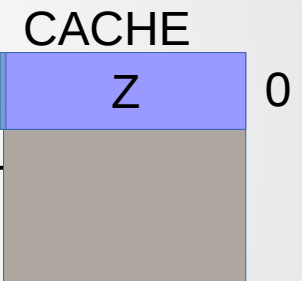
Si 512 y 1024 mapean al 0:

```
M[512] ← R3  
R1 ← M[1024]  
R2 ← M[512]
```

Read Miss

Write buffer

M[512] ← R3



Dato viejo

# Optimizaciones: Reduciendo el Miss Penalty

- Dar Prioridad a los **Read Miss** frente a los **Writes**
  - Mejora para writes con write-through → write buffer
  - ¿Qué inconveniente se puede presentar?

Si 512 y 1024 mapean al 0:

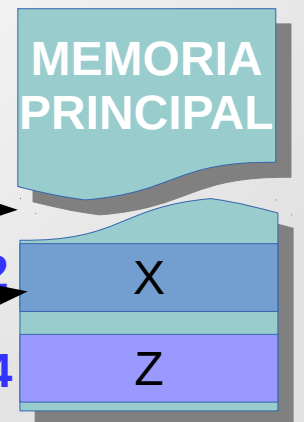
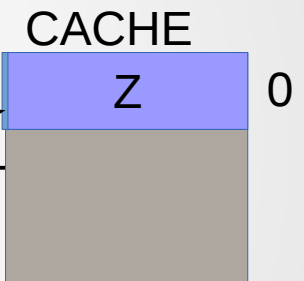
```
M[512] ← R3  
R1 ← M[1024]  
R2 ← M[512]
```

- Solución 1: asegurar buffer vacío
  - Puede incrementar todos los read miss penalties
- Solución 2: chequear el buffer

Read Miss

Write buffer

M[512] ← R3

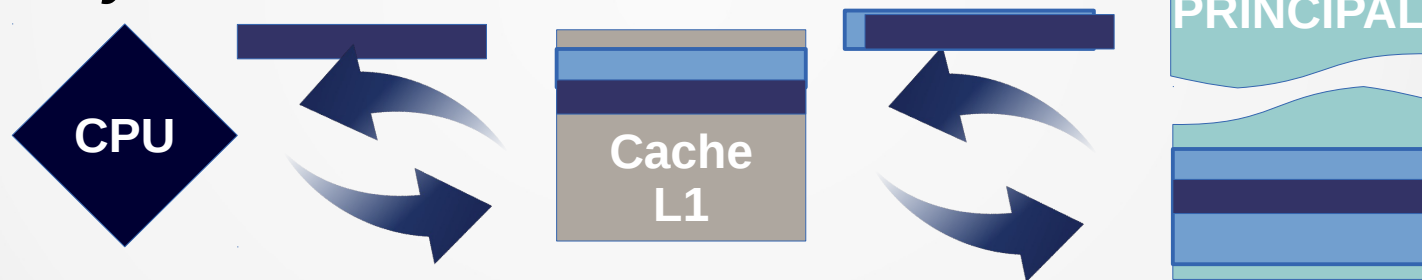


Dato viejo

# Optimizaciones: Reduciendo el Miss Penalty

- Ante un miss: ¿Debo esperar a todo el bloque para entregar la palabra al procesador?
  - Hay dos posibles esquemas:

- ***Early Restart***

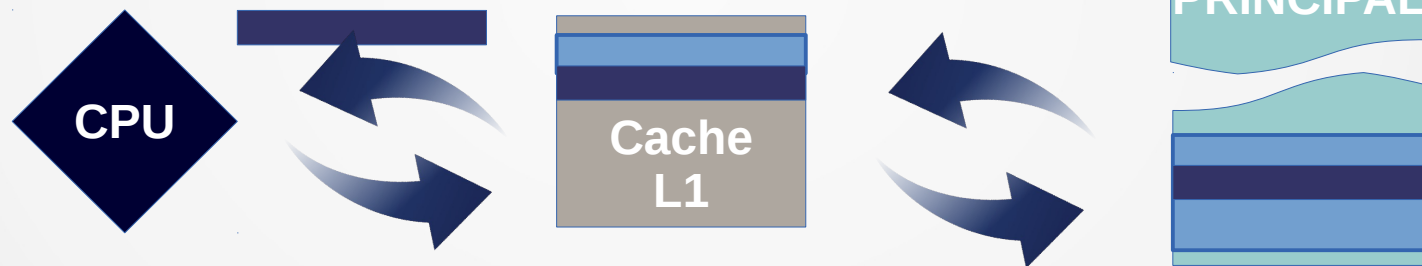


- Gano dependiendo de dónde se encuentra la palabra solicitada.

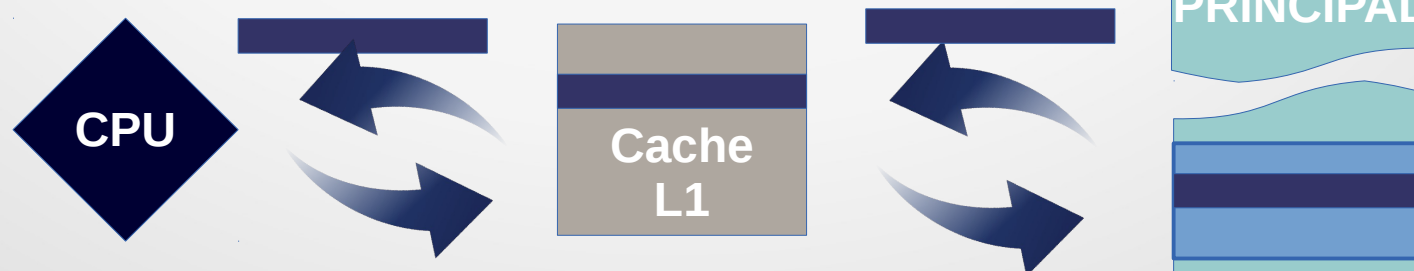
# Optimizaciones: Reduciendo el Miss Penalty

- Ante un miss: ¿Debo esperar a todo el bloque para entregar la palabra al procesador?
  - Hay dos posibles esquemas:

- ***Early Restart***



- ***Critical word first***

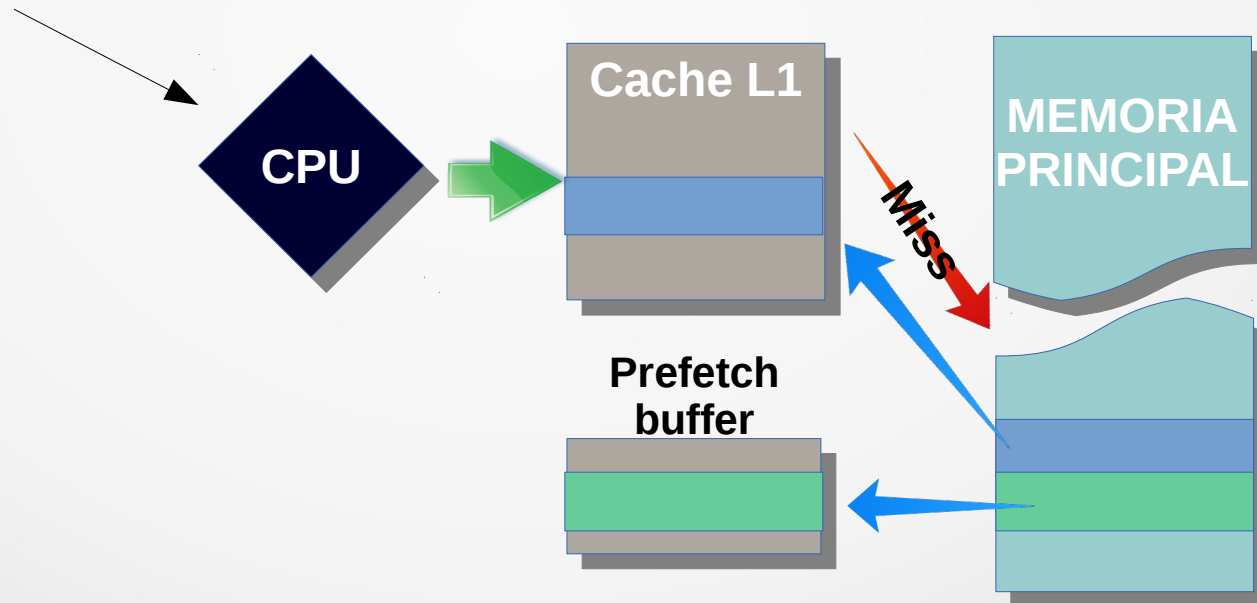


# Optimizaciones: Reduciendo el Miss Penalty

- Ante un miss: ¿Debo esperar a todo el bloque para entregar la palabra al procesador?
  - Hay dos posibles esquemas:
    - **Early Restart**: Entregar la palabra a penas esté disponible.
    - **Critical word first**: Entregar la palabra requerida al instante.
  - Mejor si bloques grandes.
  - La cache atiende accesos sucesivos a otros bloques.
  - Localidad espacial requerirá al resto del bloque.
  - Miss Penalty no es trivial.

# Optimizaciones: Reduciendo el Miss Penalty

- ***Prefetching*** de información
  - Disponer de la información con antelación.
  - Por HW.
  - Por SW.



# Optimizaciones

## A) Reducir el *Miss Rate*

- Mayor Tamaño de Cache
- Mayor Tamaño de Bloque
- Mayor Asociatividad
- Victim Caches

## B) Reducir el *Miss Penalty*

- Caches Multinivel
- Prioridad a los reads miss
- ER y CWF
- Prefetching de información

## C) Reducir el *Hit Time*

- Caches Simples
- Way Prediction

## D) Aumentar el *Ancho de Banda*

- Non Blocking Caches
- Multibanked Caches
- Pipelined Caches

# Optimizaciones: Reduciendo el Hit Time

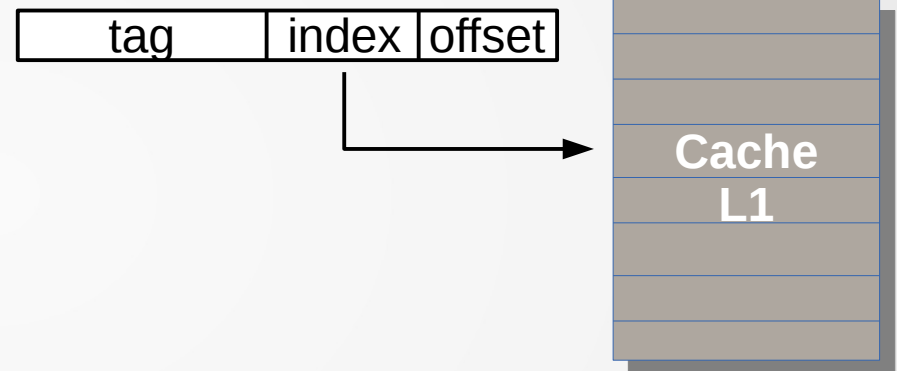
- Caches pequeñas y simples para reducir el Hit time

- *Pequeñas:*

- ↓ Tiempo de indexado
- L2 en chip

- *Simples:*

- Ej: usar mapeo directo
- (-) ↑ Misses capacitivos y conflictivos
  - Caches multi-nivel
  - Victim Caches





# Optimizaciones: Reduciendo el Hit Time

- **Way Prediction** para reducir el Hit time
  - ¿Cómo se puede combinar un Hit time veloz como el de mapeo directo con menos cantidad de conflictos como en set-asociativo?
  - **Way Prediction:**
    - Bits extras para predecir
    - Una comparación de TAG // lectura del dato
    - ¿Qué pasa si miss?
    - 85% de acierto

# Optimizaciones

## A) Reducir el *Miss Rate*

- Mayor Tamaño de Cache
- Mayor Tamaño de Bloque
- Mayor Asociatividad
- Victim Caches

## B) Reducir el *Miss Penalty*

- Caches Multinivel
- Prioridad a los reads miss
- ER y CWF
- Prefetching de información

## C) Reducir el *Hit Time*

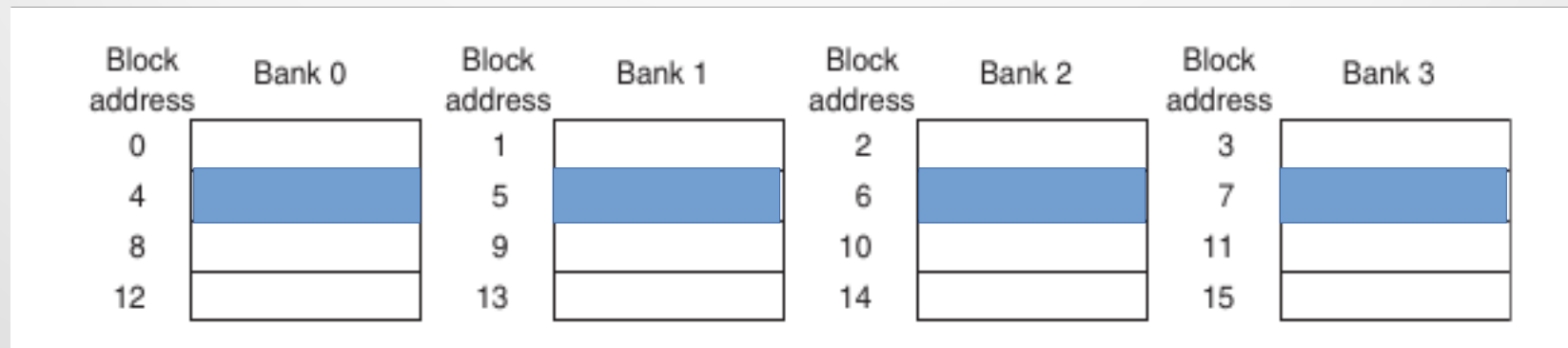
- Caches Simples
- Way Prediction

## D) Aumentar el *Ancho de Banda*

- Non Blocking Caches
- Multibanked Caches
- Pipelined Caches

# Optimizaciones: aumentando el **Ancho de Banda**

- Cache **No bloqueante**:
  - Solapar trabajo de CPU con resolución de misses
  - Atender Read Hits luego de un Miss
  - Mejora el Miss Penalty efectivo
  - “Miss under Miss” o “Hit under multiple Miss”
- **Multibanked caches**



# Optimizaciones: aumentando el **Ancho de Banda**

- **Cache No bloqueante:**
  - Solapar trabajo de CPU con resolución de misses
  - Atender Read Hits luego de un Miss
  - Mejora el Miss Penalty efectivo
  - “Miss under Miss” o “Hit under multiple Miss”
- **Multibanked caches**
- **Pipelined Caches**
  - + mayor throughput
  - - latencia individual para un hit
  - + Hit Time

# Referencias

Hennessy, J., and Patterson, D. Computer Architecture, second ed. Morgan Kaufmann, 1996. (Capítulo 5)

Hennessy, J., and Patterson, D. Computer Architecture, fourth ed. Morgan Kaufmann, 2006. (Capítulo 5 y Apéndice C)