Arquitectura de Computadoras para Ingeniería

(Cód. 7526)
1° Cuatrimestre 2018

Dra. Dana K. Urribarri
DCIC - UNS

Clasificación de Flynn

Clasificación de Flynn

- Clasificación de 1966
- En función del flujo de instrucciones y datos en un solo procesador:

{instrucción única, múltiples instrucciones} x {dato único, múltiples datos}

	Single Data	*Multiple Data*
Single Instruction	SISD	SIMD
Multiple Instruction	MISD	MIMD

Clasificación de Flynn

Flujo de instrucciones	Flujo de datos	Nombre	Ejemplos
1	1	SISD	Máquina Von Neumann Clásica
1	muchos	SIMD	Supercomputadora vectorial. Arreglo de procesadores.
muchos	1	MISD	?
muchos	muchos	MIMD	Multiprocesadores, multicomputadoras

- Está basada en 2 conceptos:
  - Flujo de instrucciones: Se corresponde con el *program counter*. Un sistema con *n* CPUs tiene *n program counters* y por lo tanto *n* flujos de instrucciones.
  - Flujo de datos: Se corresponde con el conjunto de operandos.

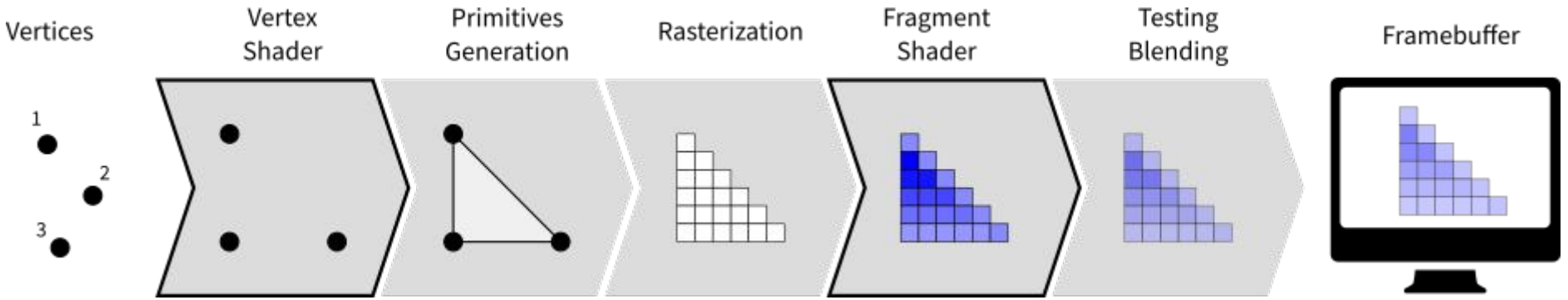
SISD

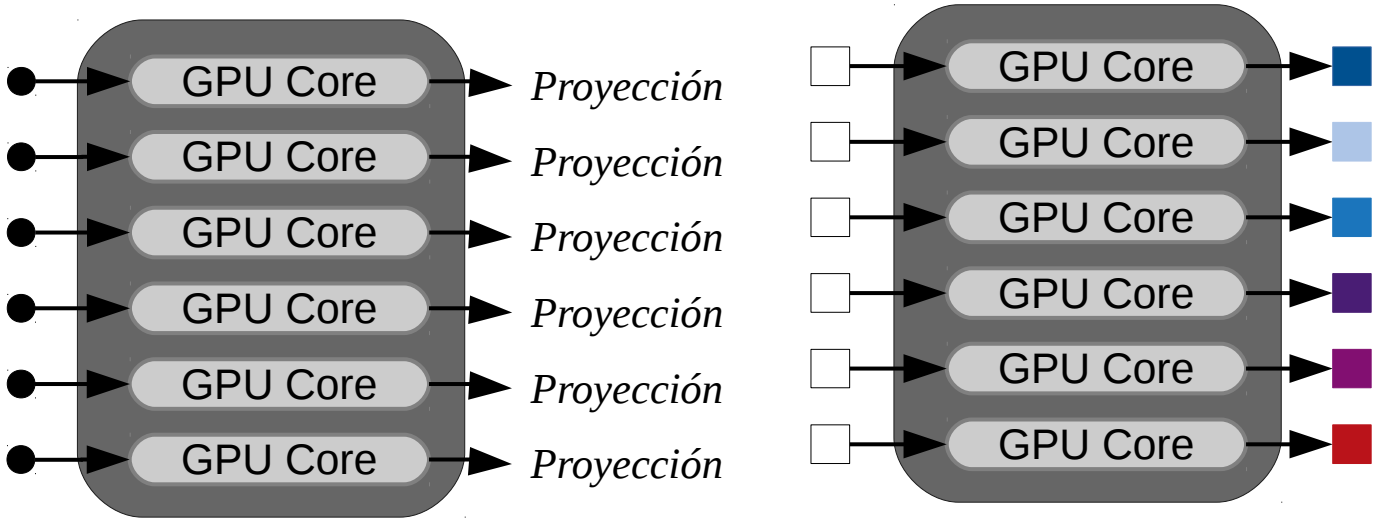
- Un único procesador Von Neumann clásico.
- Hay un único flujo de instrucciones y un único flujo de datos.
- Realiza una única operación a la vez.
  - Ejecución concurrente en pipeline
  - Procesadores superescalar con múltiples unidades funcionales.

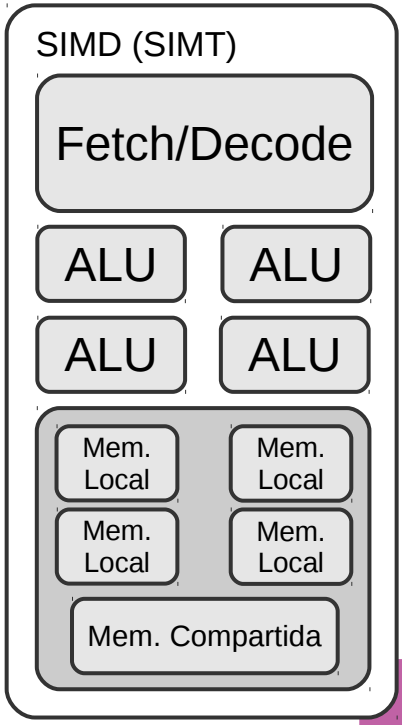
SIMD

- Tienen una única unidad de control que procesa una única instrucción a la vez.
- Tienen múltiples ALUs para operar sobre múltiples conjuntos de datos en simultáneo.
- Extensiones multimedia:
  - MMX (enteros, 1997 Pentium)
  - 3DNow! (pf, 1998 K6-2), SSE (1999, Pentium III), SSE2,3,4.x…, AVX (2010)
- GPU

SIMD: GPU


Vertices
Vertex Shader
Primitives Generation
Rasterization
Fragment Shader
Testing Blending
Framebuffer
1
2
3

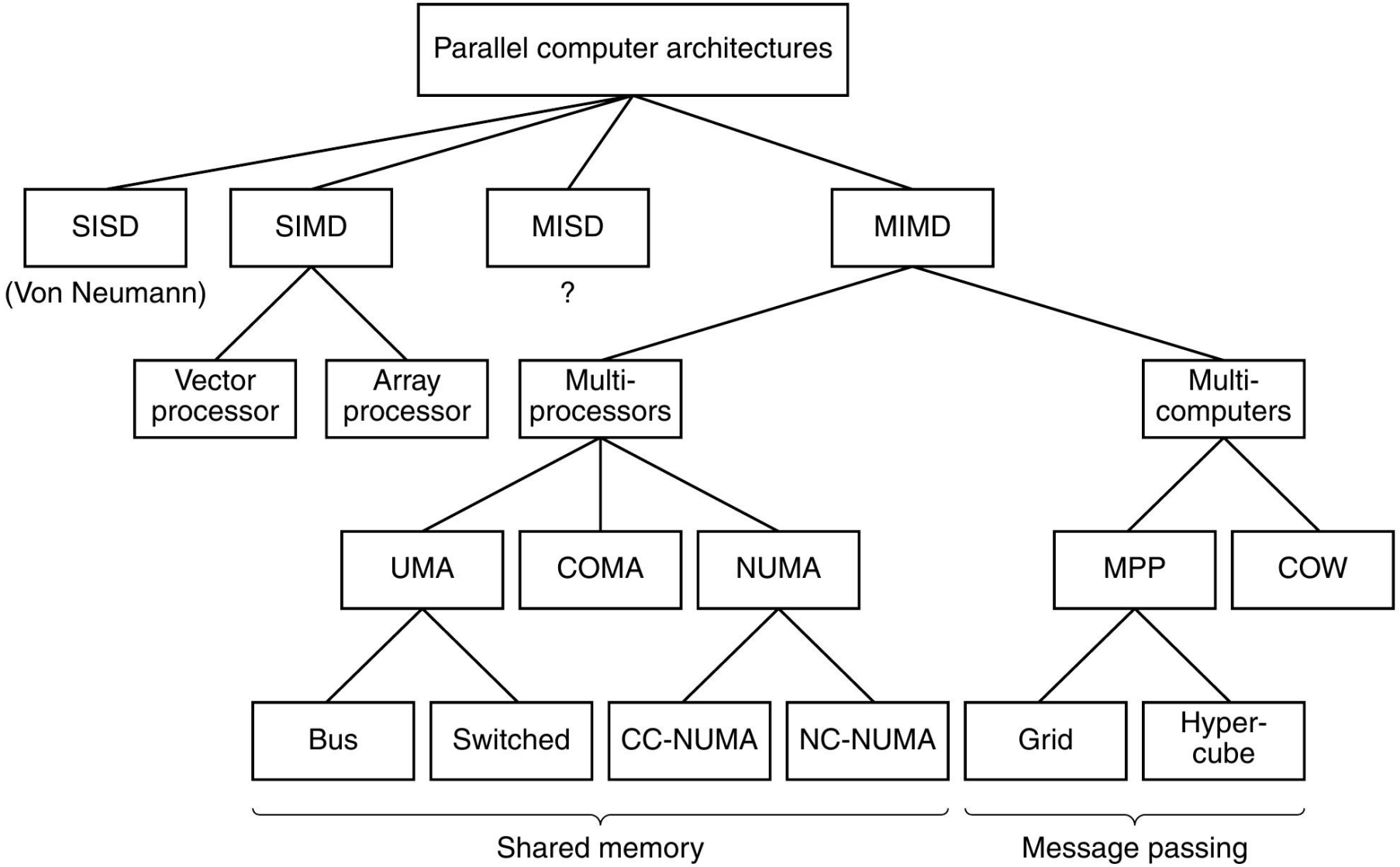

GPU Core
GPU Core
GPU Core
GPU Core
GPU Core
GPU Core
Proyección
Proyección
Proyección
Proyección
Proyección
Proyección
GPU Core
GPU Core
GPU Core
GPU Core
GPU Core
GPU Core


SIMD (SIMT)
Fetch/Decode
ALU
ALU
ALU
ALU
Mem. Local
Mem. Local
Mem. Local
Mem. Local
Mem. Compartida

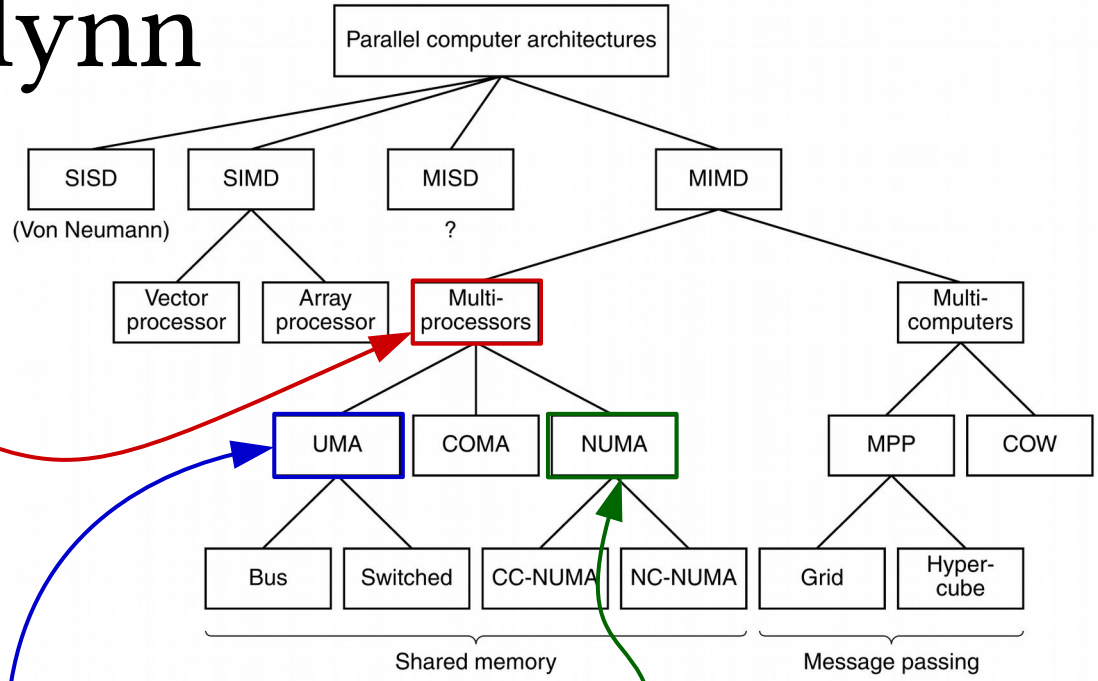
MIMD

- Varios procesadores operan en paralelo de manera asincrónica.
- Se distinguen en:
  - Cómo se comunican los procesadores
    - Memoria compartida
    - Pasaje de mensajes
  - El acceso a memoria principal
    - Uniforme
    - Distribuido entre los procesadores
  - Número de procesadores, homogeneidad, interconexión procesador-procesador y procesador-memoria, coherencia de caché, sincronización, etc...

Clasificación de Flynn



Clasificación de Flynn



- Los procesadores paralelos comparten un mismo espacio de direcciones de memoria.
- La comunicación es a través de instrucciones *load-store*.

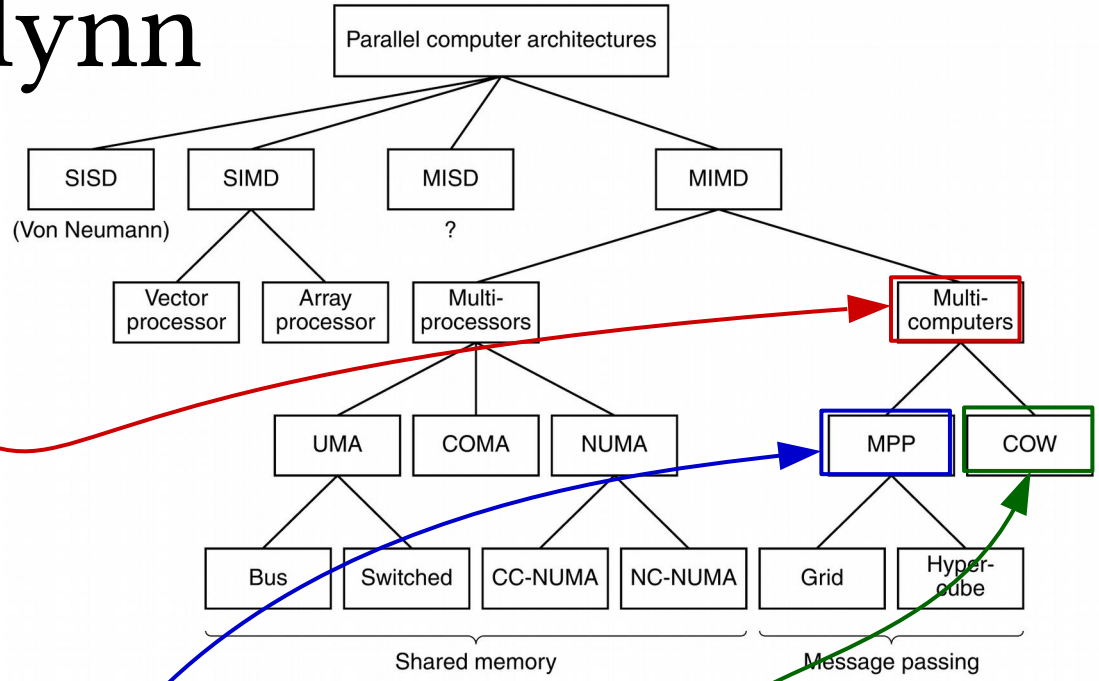
Acceso uniforme a memoria (UMA):

- Todos los procesadores tienen el mismo tiempo de acceso a cualquier módulo de memoria.
- Todas las palabras de memoria se acceden el mismo tiempo.
- Si técnicamente no se puede, se demora la referencia más rápida.
- Hace que el rendimiento sea predecible.

Acceso no uniforme a memoria (NUMA):

- Memoria compartida distribuida.
- Hay un módulo de memoria cercano a cada procesador (al cual es más rápido acceder).
- La ubicación del código y datos afecta el rendimiento.

Clasificación de Flynn



- Las multicomputadoras no tienen memoria compartida a nivel arquitectura (el SO en una CPU no puede acceder con load/store a memoria asociada a otra CPU).
- La comunicación es a través de mensajes.
- No tienen acceso directo a memoria remota.

Massively Parallel Processors (MPP)

- Supercomputadoras fuertemente acopladas por una red de interconexión rápida y propietaria.

Cluster, Cluster of Workstations (COW)

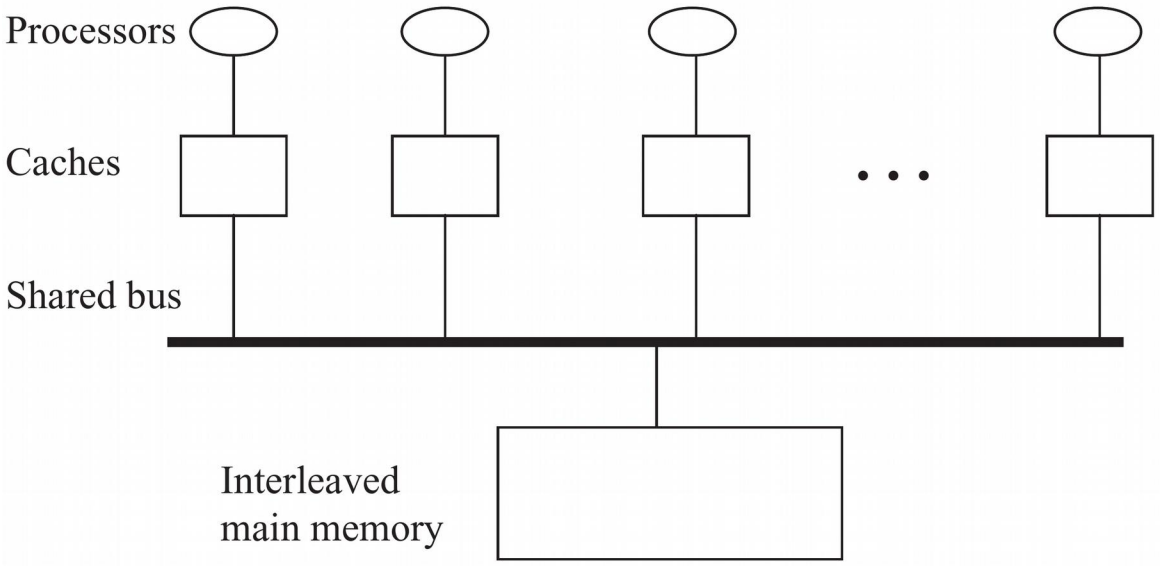
- PC regulares, workstations, servidores conectados en una LAN.
- Nodos más económicos.

Interconexión memoria/procesador (y procesador/procesador)

Interconexiones Memoria-Procesador

- Principales esquemas de conexionado:
  - Shared bus
  - Crossbar
  - Redes directas
  - Meshes

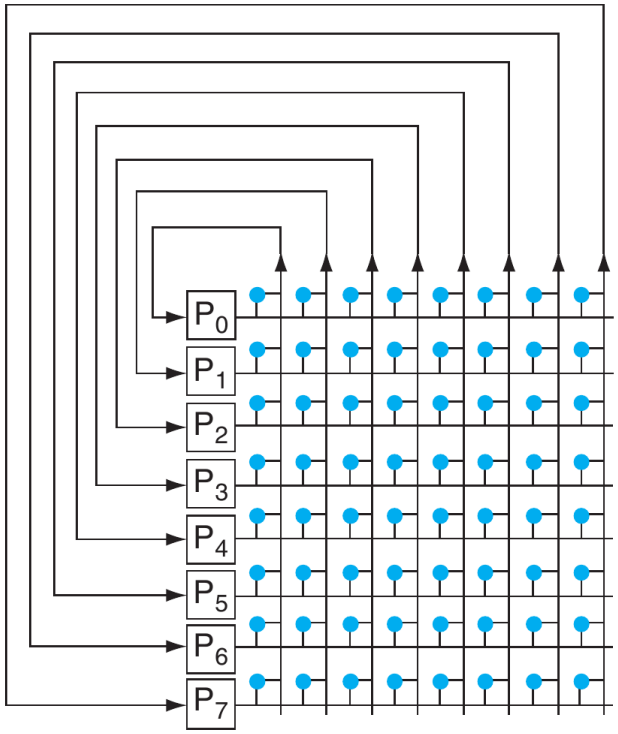
Shared bus

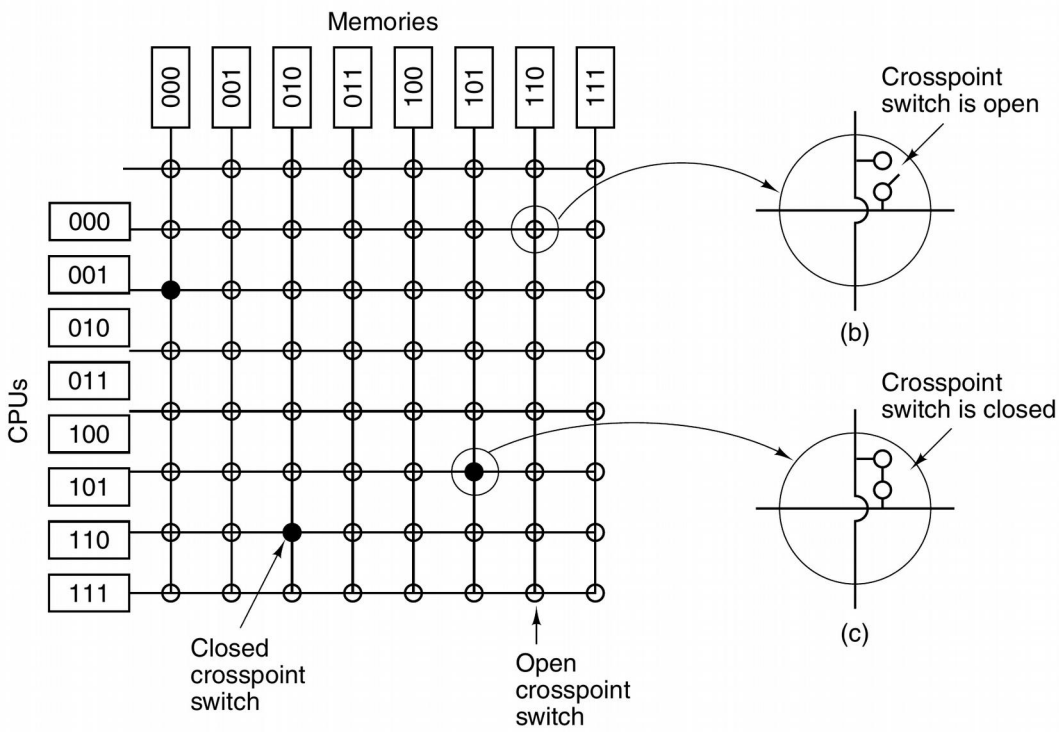


- ✔ Forma más simple de conexión
- ✔ Bajo costo
- ✔ Facilidad de uso
- ✔ Capacidad de hacer *broadcast*
- ✗ Restricciones físicas (longitud, carga eléctrica) que limitan el número de dispositivos que pueden conectarse.
- ✗ A mayor cantidad de dispositivos se incrementa la pelea por el bus.

Crossbar

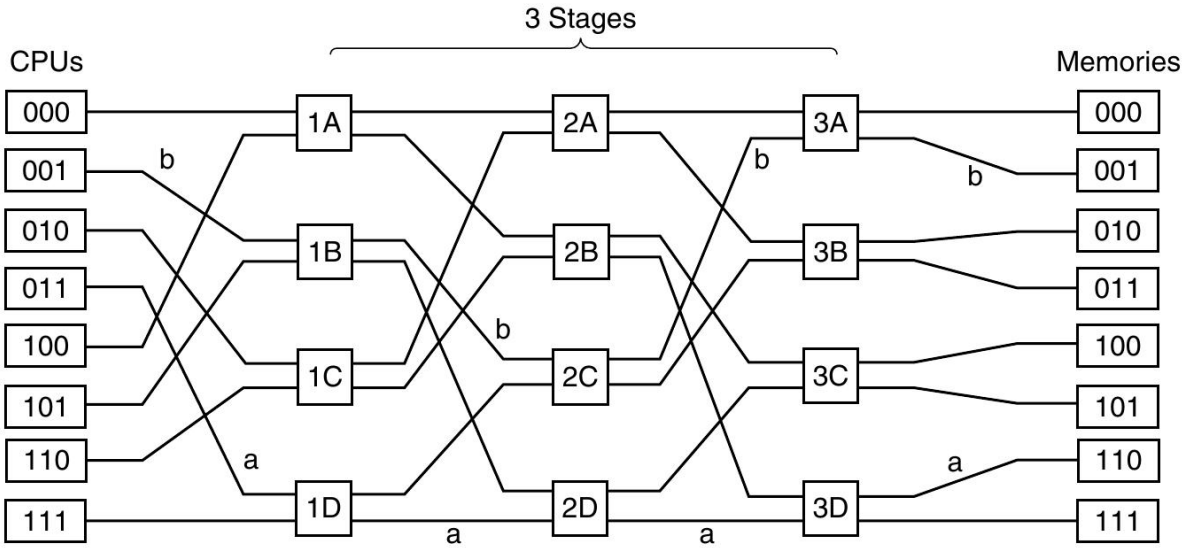
- ✔ Provee máxima concurrencia
  - Entre *n* procesadores y *m* memorias (UMA)
  - Entre *n* procesadores (NUMA)
- ✗ El costo crece con
  - $n \times m$
  - n^2



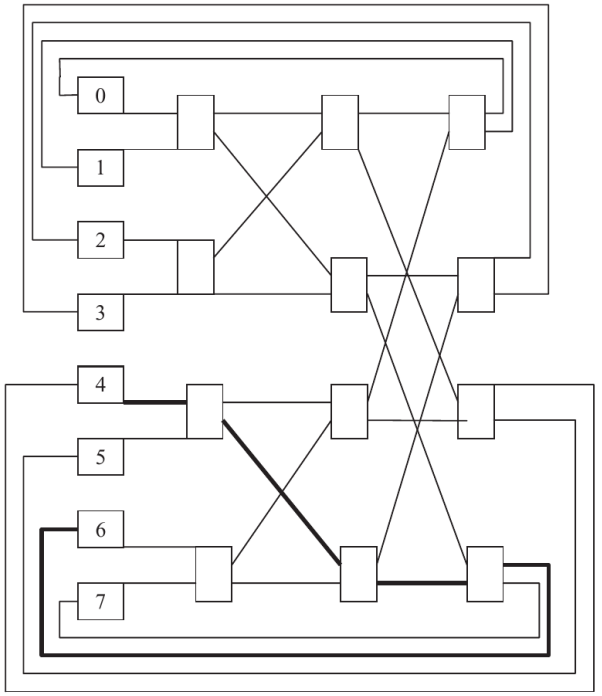


Direct interconnection networks

- Se basan en el *crossbar* de 2x2.
- Múltiples etapas de interconexión.
- Cada procesador está a la misma distancia de cada memoria o de los demás procesadores.



Omega Network

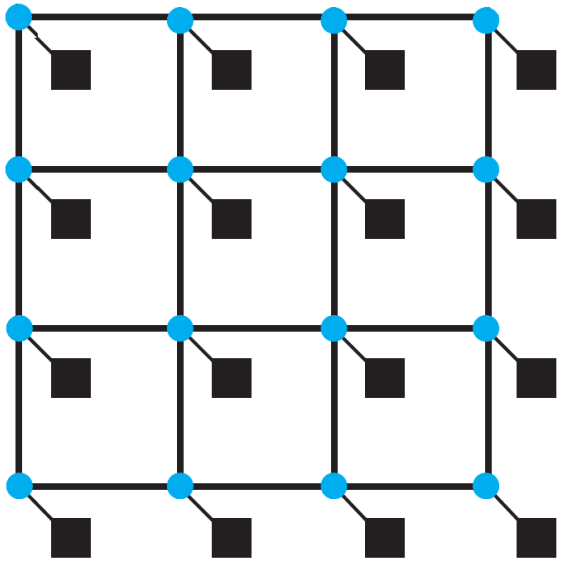


Butterfly Network

Meshes

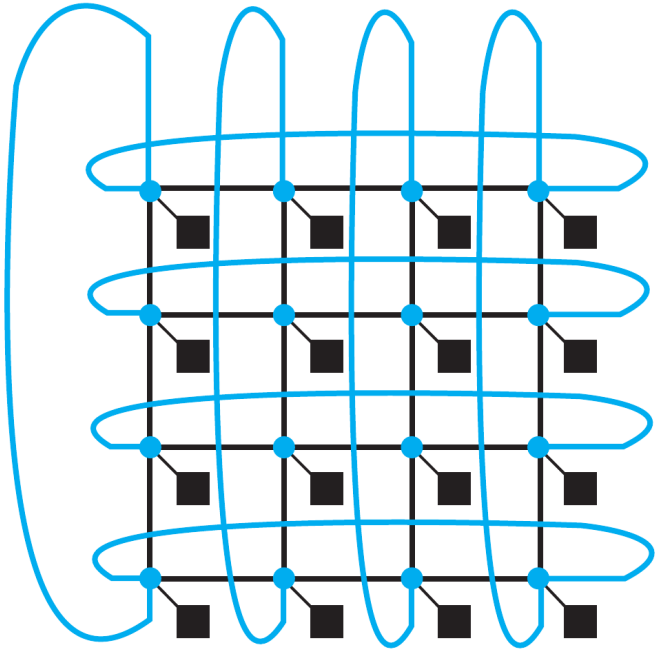
- Interconexión indirecta entre todos los nodos.
- Cada nodo está conectado con un enlace de longitud 1 con un subconjunto de nodos vecinos (dimensión).
- La distancia entre procesadores y memorias/procesadores varía.

Anillo
Cada nodo tiene conexión directa con 2 vecinos.
Conexión unidimensional.

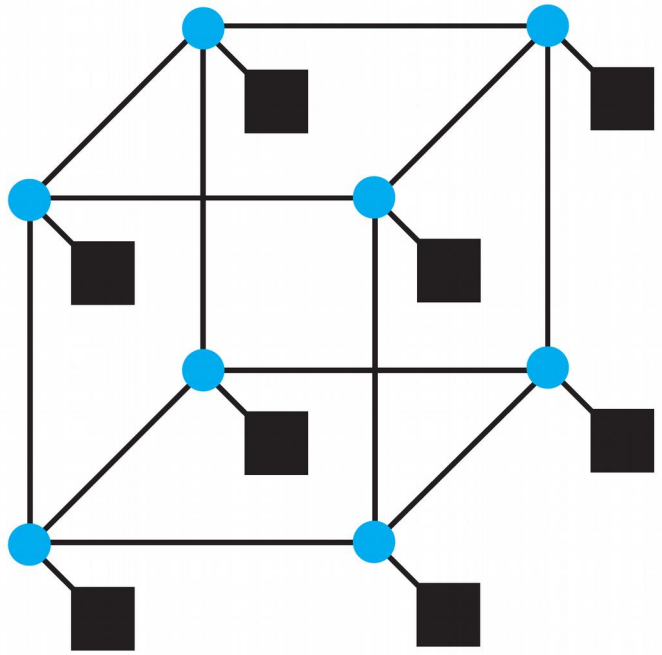


Mesh 2D
Cada nodo interior tiene conexión directa con sus 4 vecinos.

Meshes



Torus 2D
Todos los nodos están conectado con un enlace de longitud 1 con 4 vecinos.

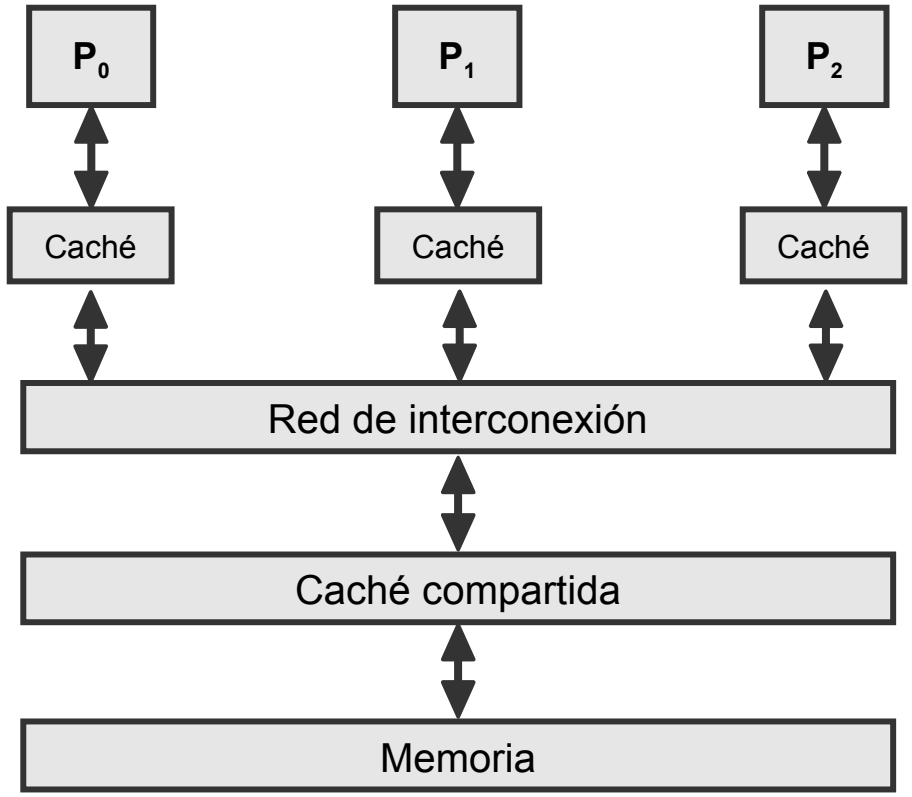


Hipercubo de *m* dimensiones
Cada nodo está conectado con un enlace de longitud 1 con *m* vecinos.

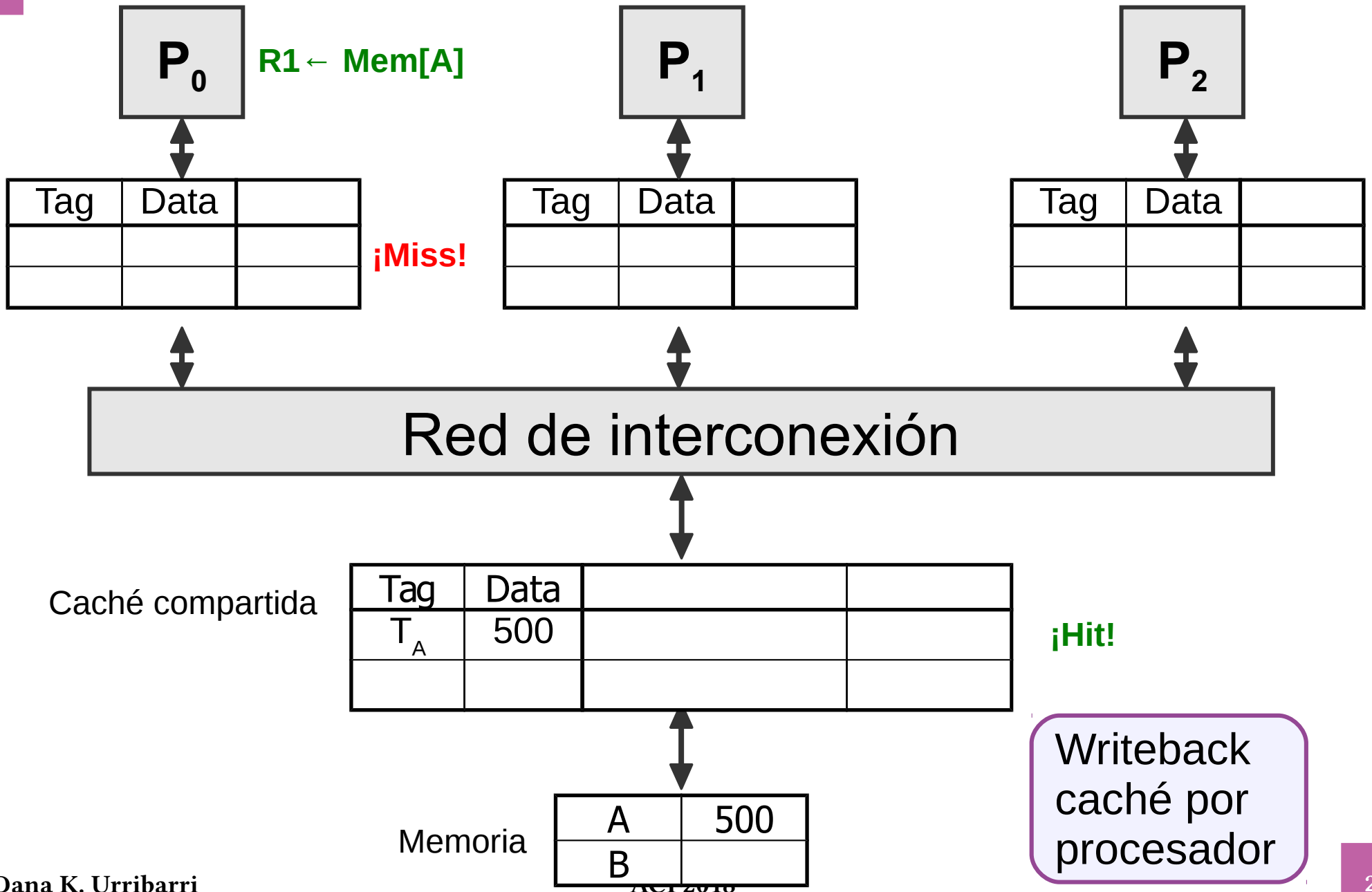
Multiprocesadores y la coherencia de caché

Memoria caché en multiprocesadores

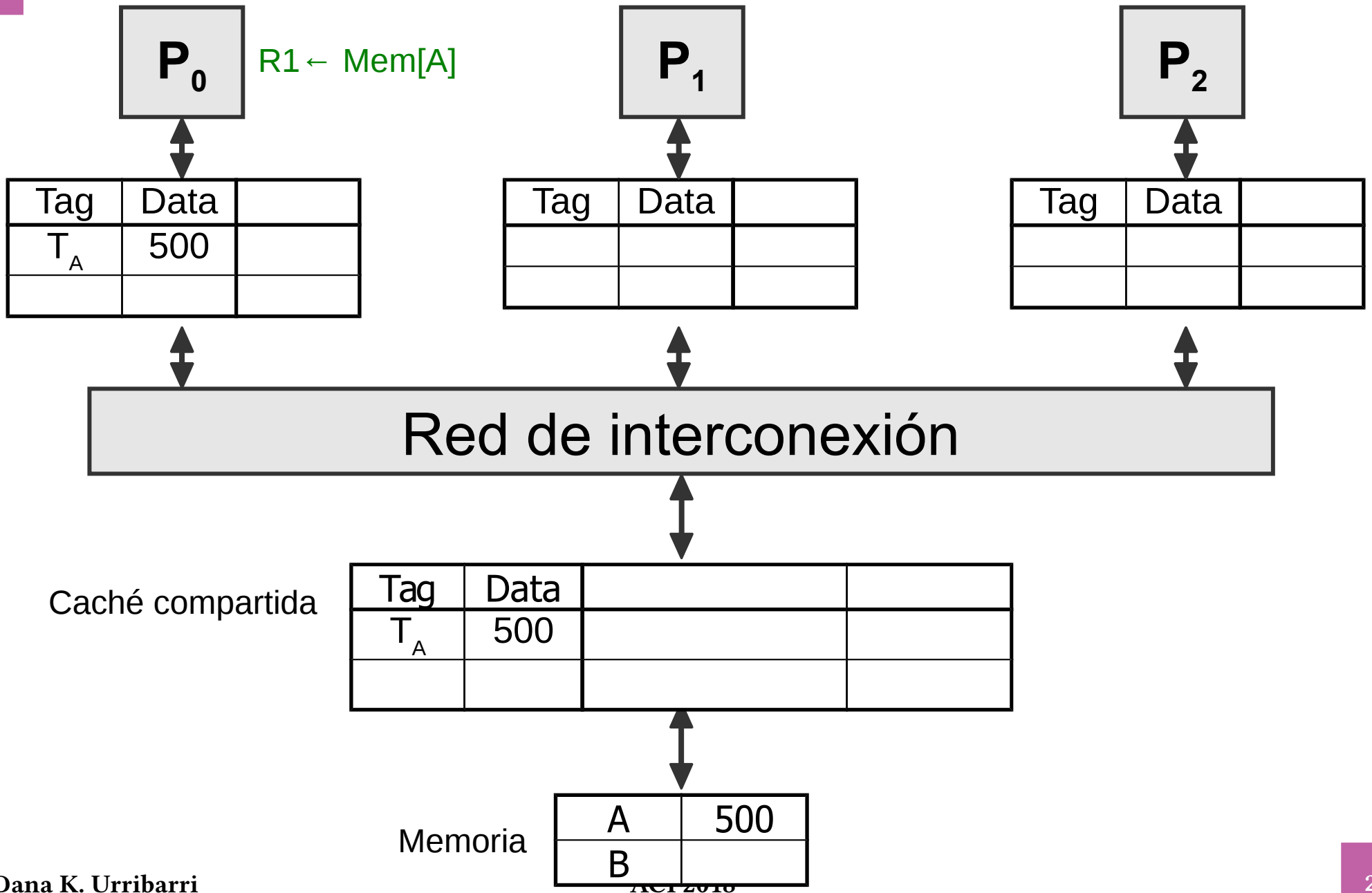
- Varios procesadores.
- Cada procesador con su propia memoria caché.
- Todos acceden al mismo espacio de direcciones.
- Se comunican por loads y store.
- Pueden tener compartir niveles de memoria caché.



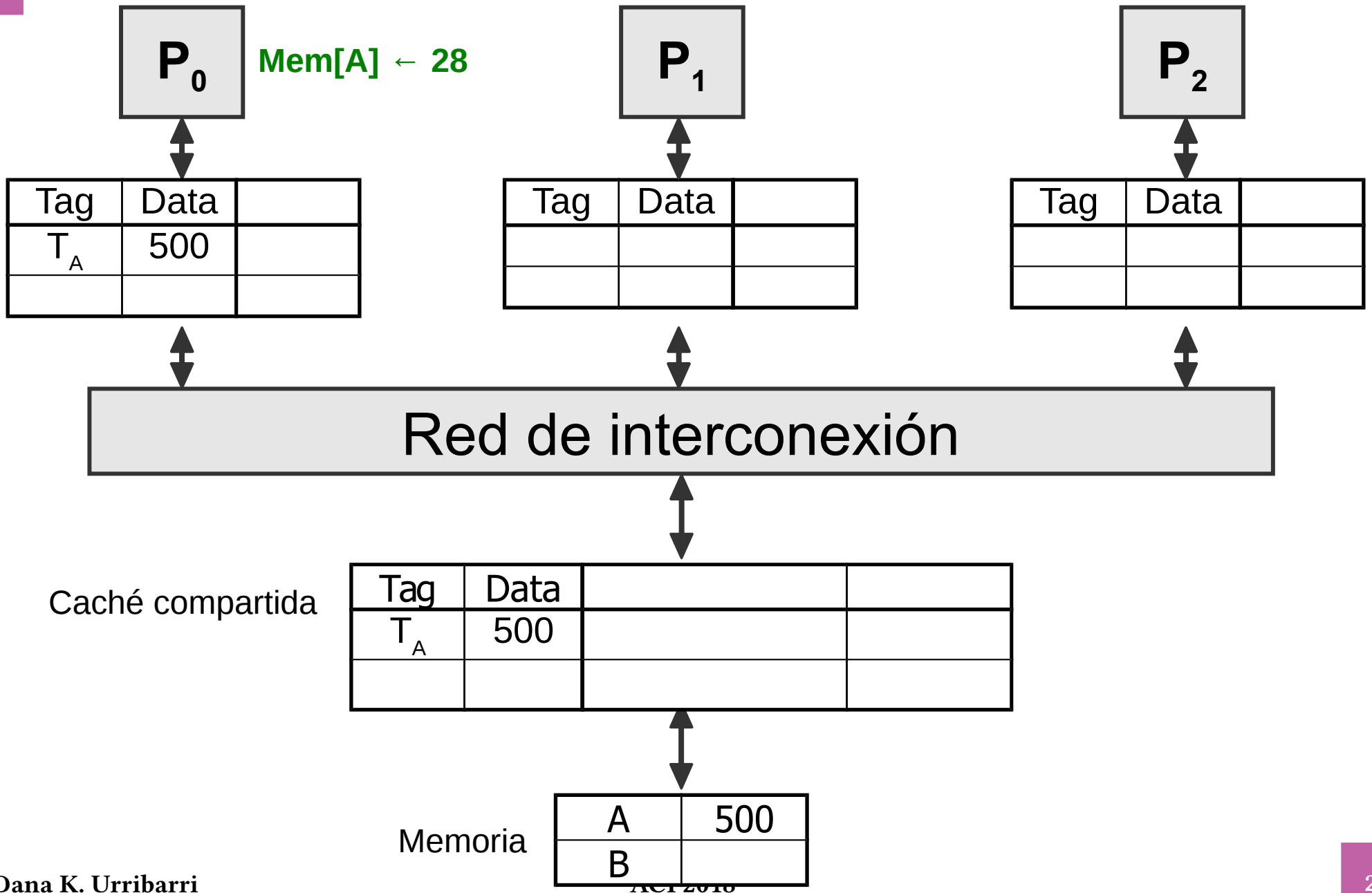
Ejemplo...


P0
R1 ← Mem[A]
P1
P2
Tag
Data
¡Miss!
Red de interconexión
Caché compartida
Tag
Data
TA
500
¡Hit!
Memoria
A
500
B
Writeback caché por procesador

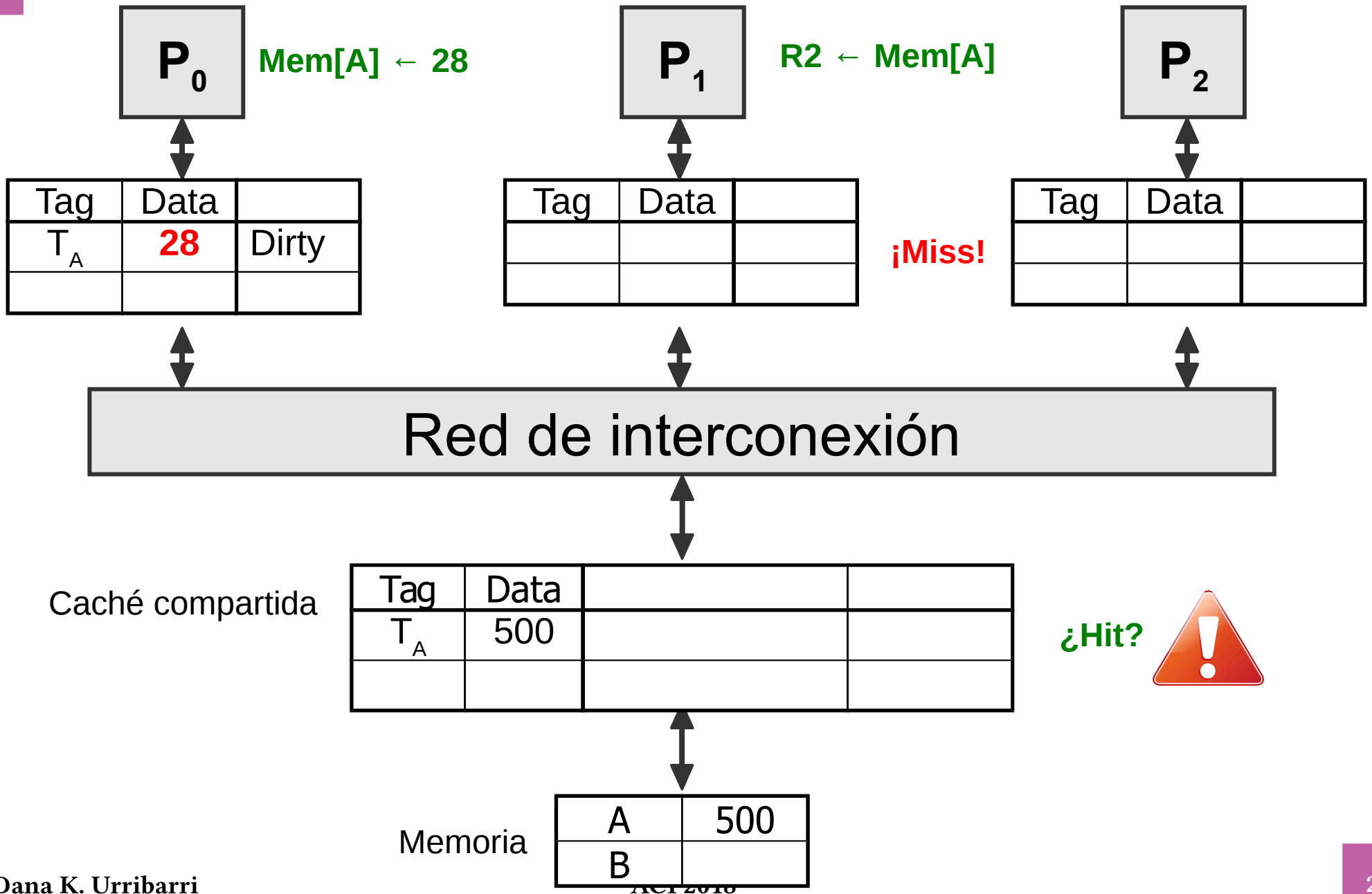
Ejemplo...


P0
R1 ← Mem[A]
P1
P2
Tag
Data
TA
500
Red de interconexión
Caché compartida
Memoria
A
500
B

Ejemplo...


P0
Mem[A] ← 28
P1
P2
Tag
Data
TA
500
Red de interconexión
Caché compartida
Memoria
A
500
B

Ejemplo...


P0
Mem[A] ← 28
P1
R2 ← Mem[A]
P2
Tag
Data
TA
28
Dirty
¡Miss!
Red de interconexión
Caché compartida
500
¿Hit?
Memoria
A
B

Cómo se sabe el estado de un bloque

- Fuertemente dependiente del tipo de interconexión
- *Snooping*
  - Cada caché mantiene el estado del bloque compartido.
  - Shared bus (fácil hacer broadcast).
  - Los controladores de caché controlan (o espían) el bus por si hay un requerimiento sobre un bloque que tengan.
- Basados en directorios
  - Se almacena un directorio con información del estado de cada bloque compartido de memoria física.
    - **UMA**: centralizado en algún punto común (por ej. último nivel de caché)
    - NUMA: directorios distribuidos.

Complejidad del protocolo

Write-invalidate protocol

- Cuando un procesador escribe en un bloque se invalidan todas las demás copias.
  - **MSI**: Un bloque tiene 3 estados posibles: *Modified*, *Shared*, *Invalid*
  - MESI (MSI + Exclusive)
  - *MESIF (MESI + Forward), MOESI (MESI + Owned)*

Write-update protocol

- Cuando un procesador escribe en un bloque se actualizan las demás copias.

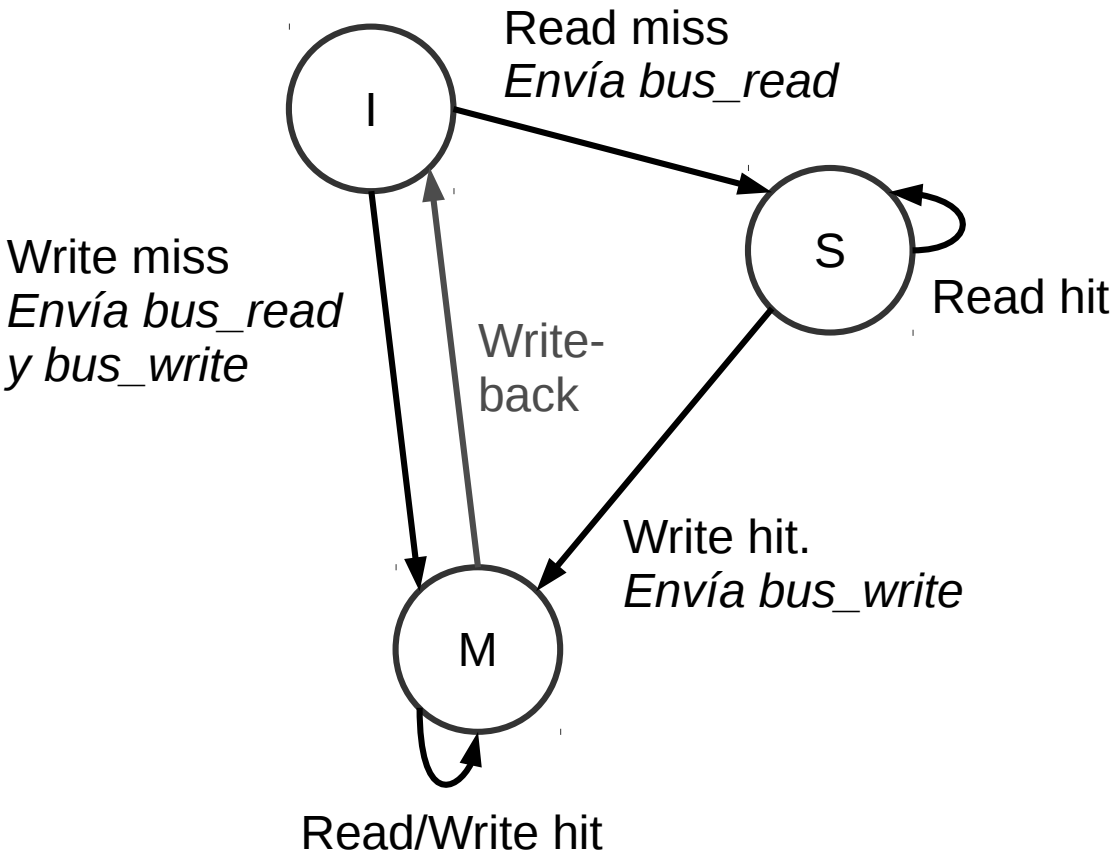
MSI

- Estados
  - Invalid (I): La línea estuvo en la caché pero fue invalidada.
  - Shared (S): En el sistema hay varias copias de la línea y son todas iguales.
  - Modified (M): La línea es la única copia en el sistema.

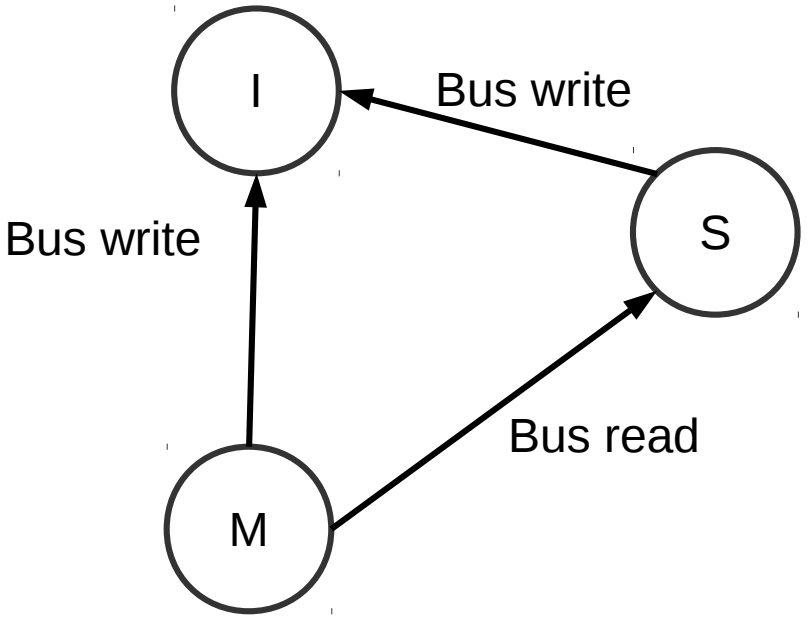
MSI + Snooping

Estado de una línea en la caché local

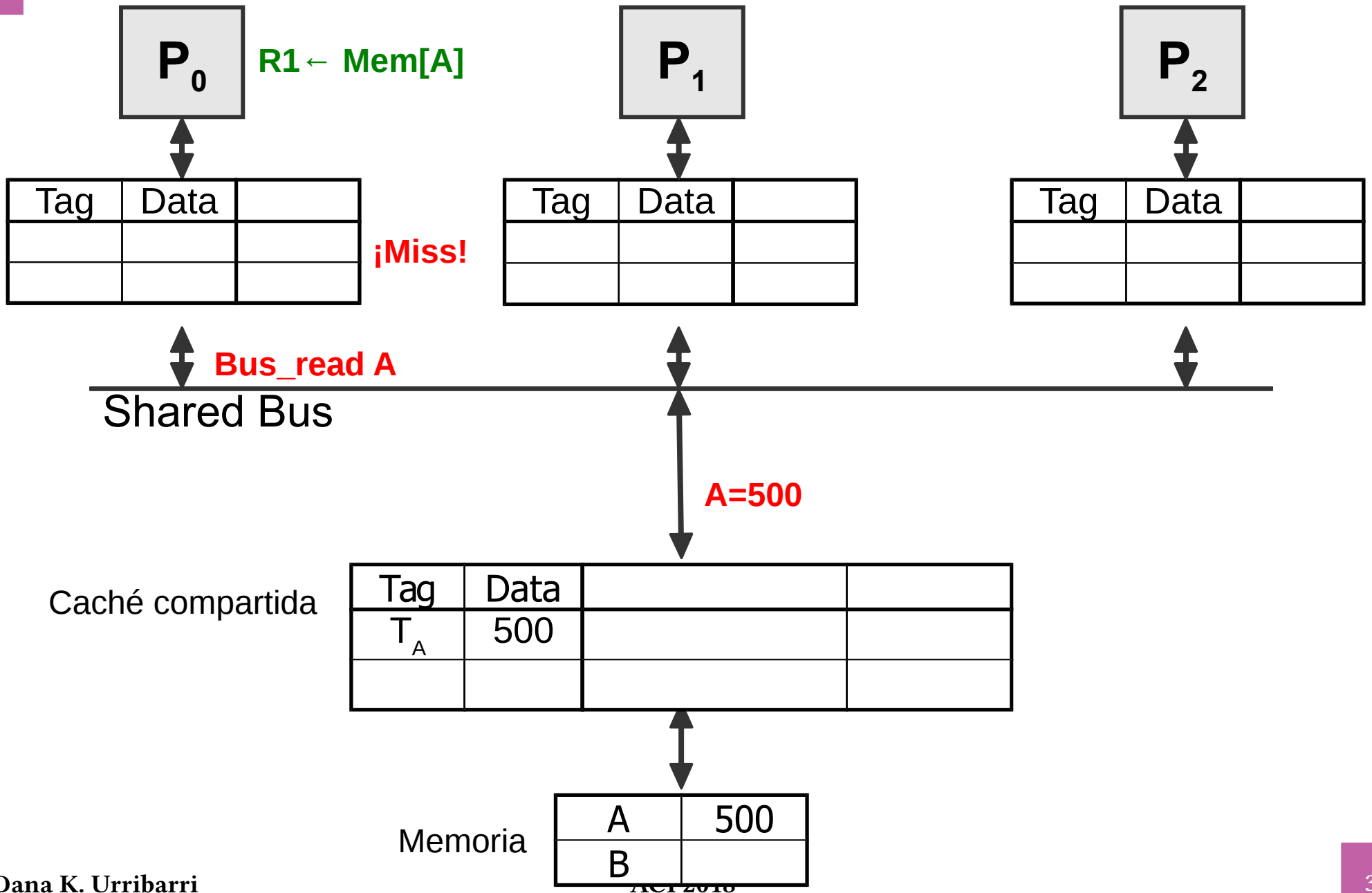
Según acciones del procesador



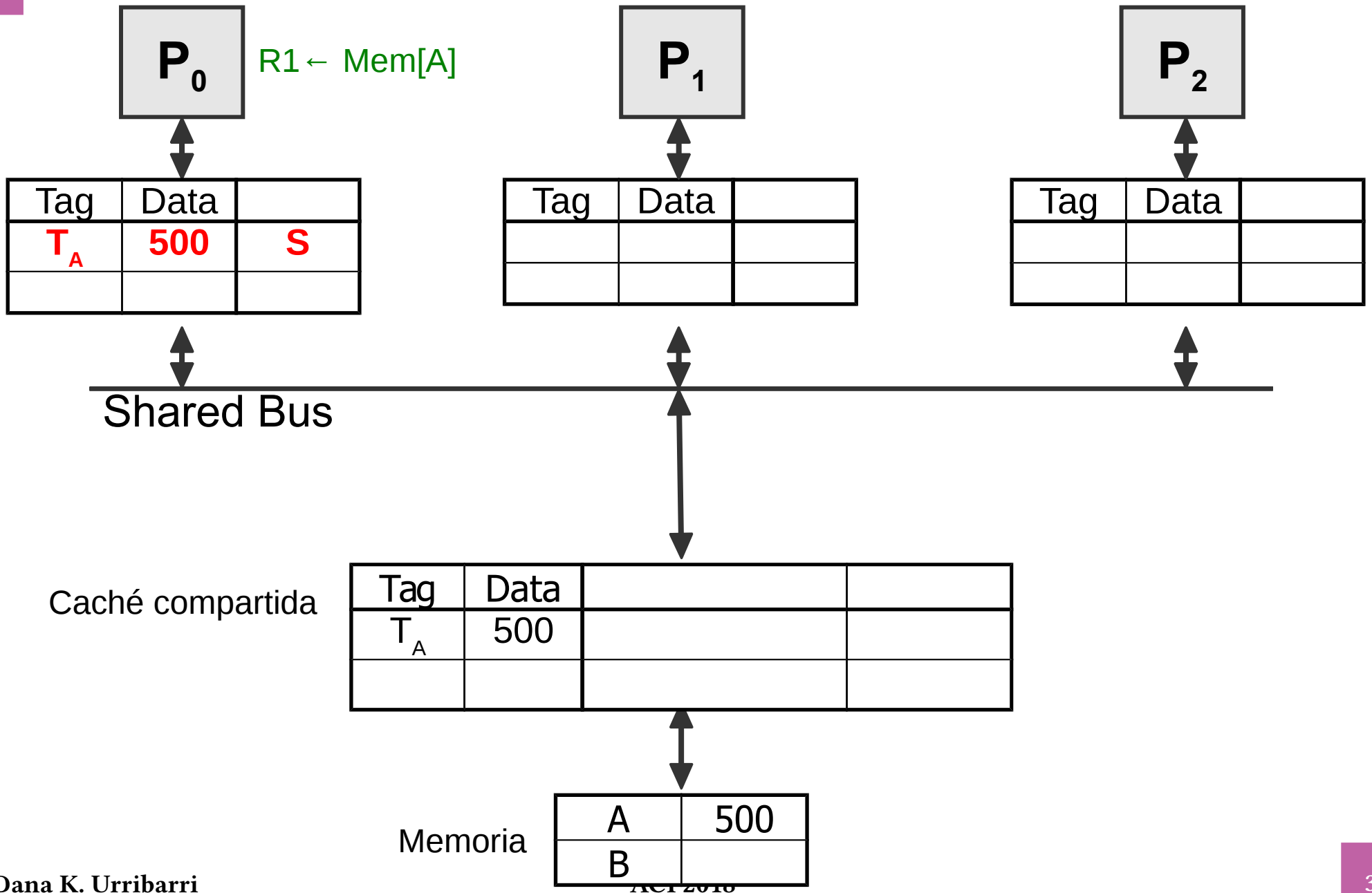
Según señales externas



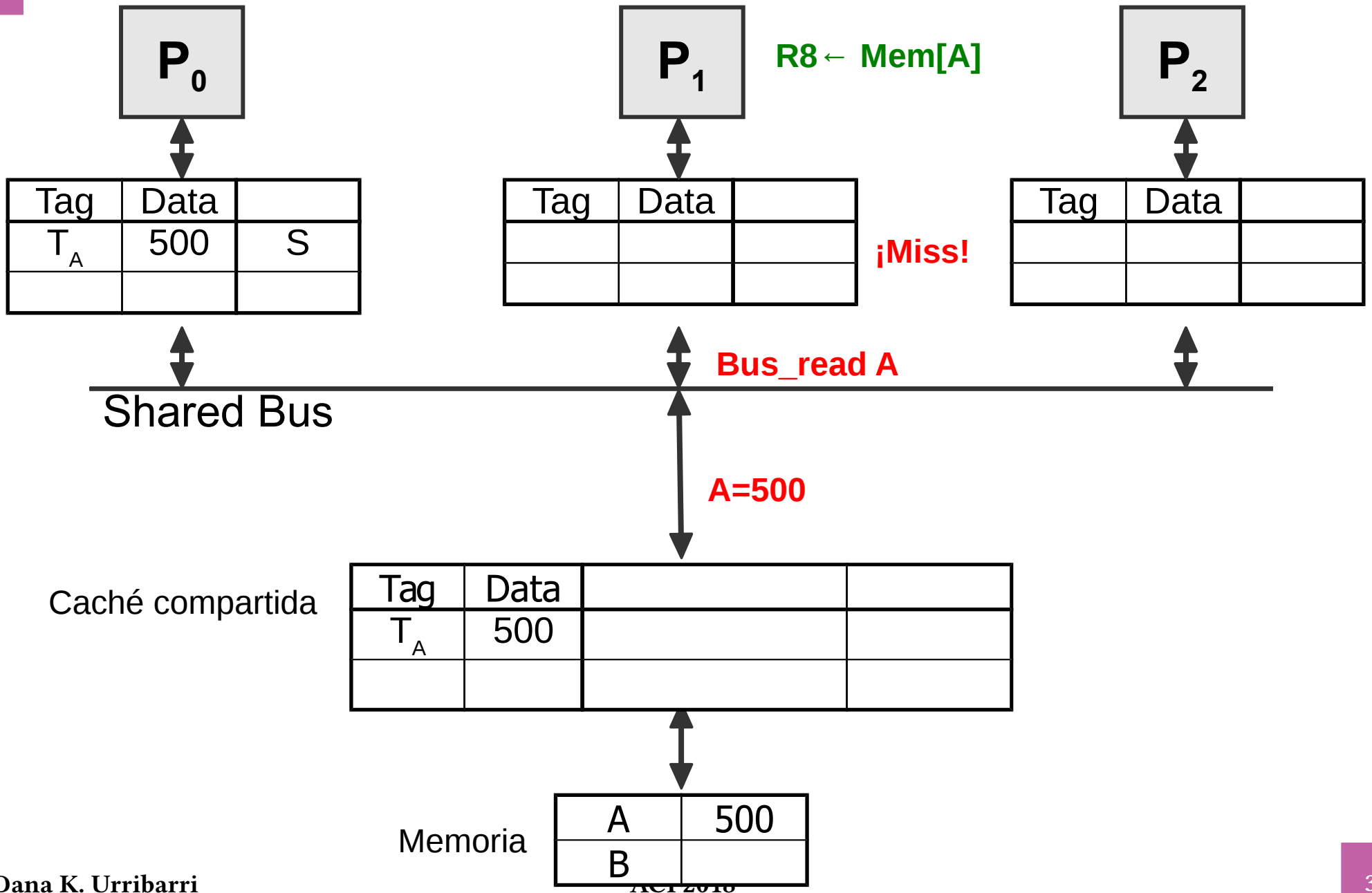
Ejemplo...


P0
R1 ← Mem[A]
P1
P2
Tag
Data
¡Miss!
Bus_read A
Shared Bus
A=500
Caché compartida
Tag
Data
TA
500
Memoria
A
500
B

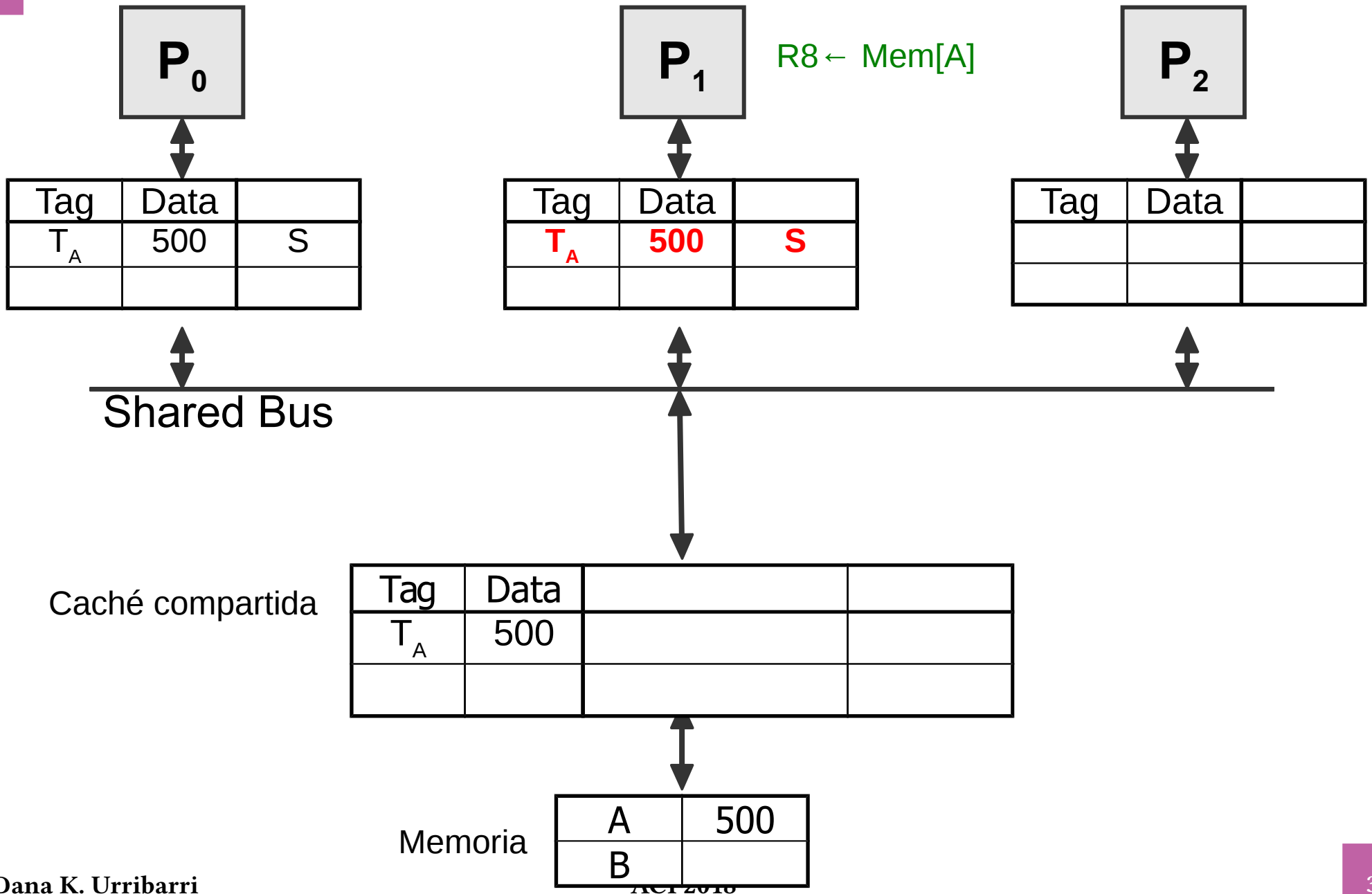
Ejemplo...


P0
R1 ← Mem[A]
P1
P2
Tag
Data
T A
500
S
Shared Bus
Caché compartida
Memoria
A
500
B

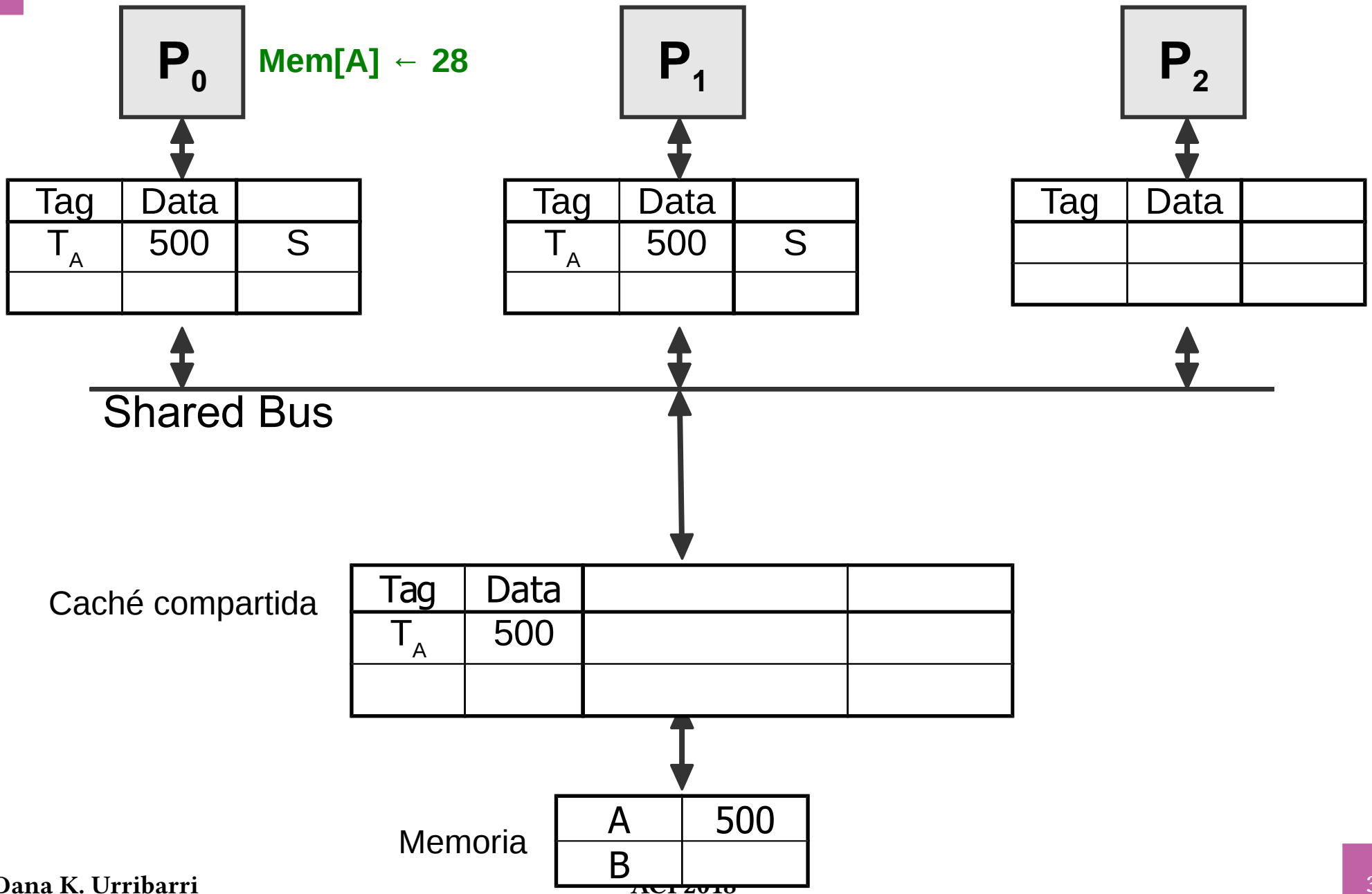
Ejemplo...


P0
P1
R8 ← Mem[A]
P2
Tag
Data
TA
500
S
¡Miss!
Bus_read A
Shared Bus
A=500
Caché compartida
Memoria
A
500
B

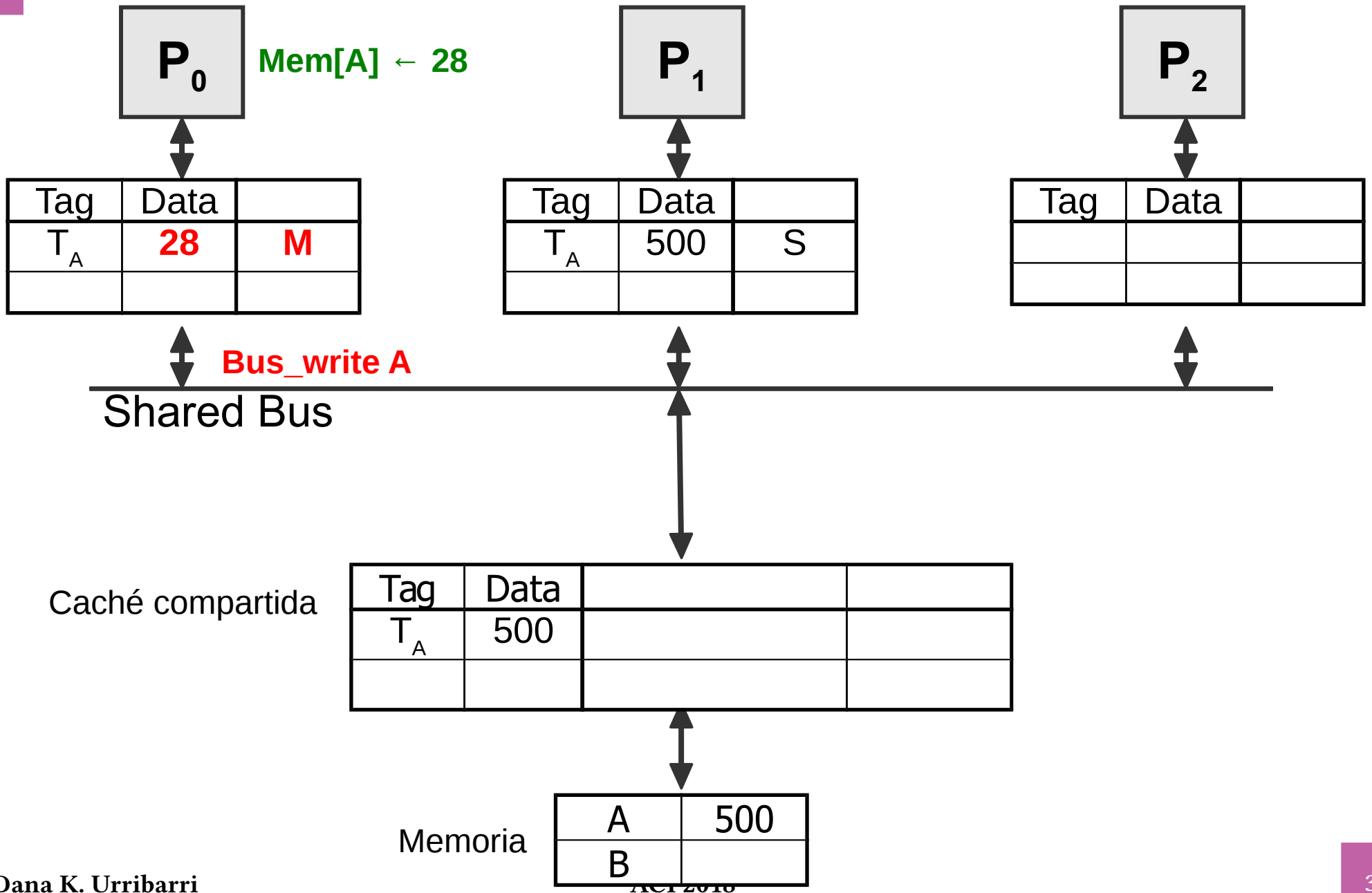
Ejemplo...


P0
P1
R8 ← Mem[A]
P2
Tag
Data
TA
500
S
Shared Bus
Caché compartida
Memoria
A
B

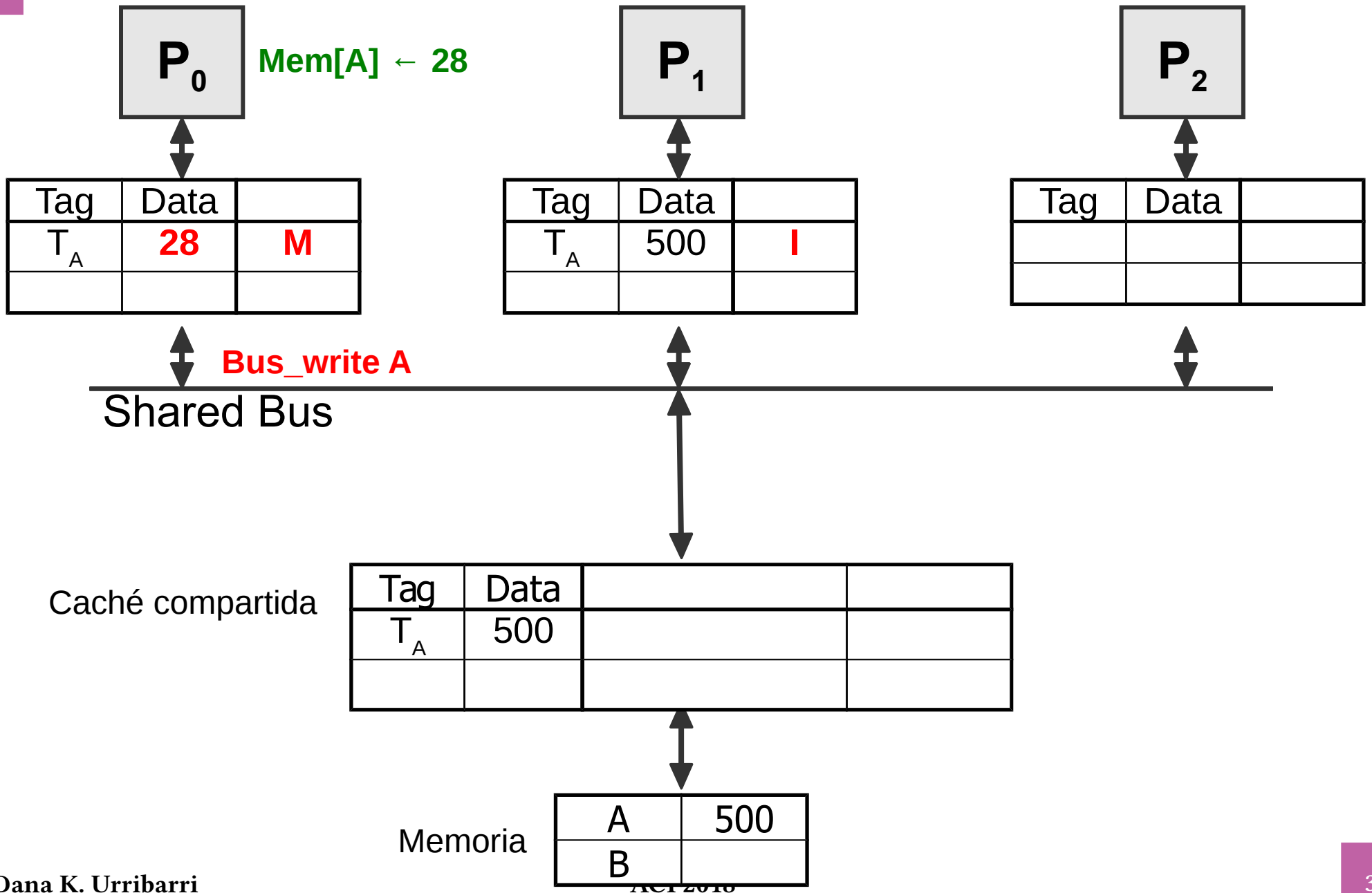
Ejemplo...


P0
Mem[A] ← 28
P1
P2
Tag
Data
TA
500
S
Shared Bus
Caché compartida
A
B
Memoria

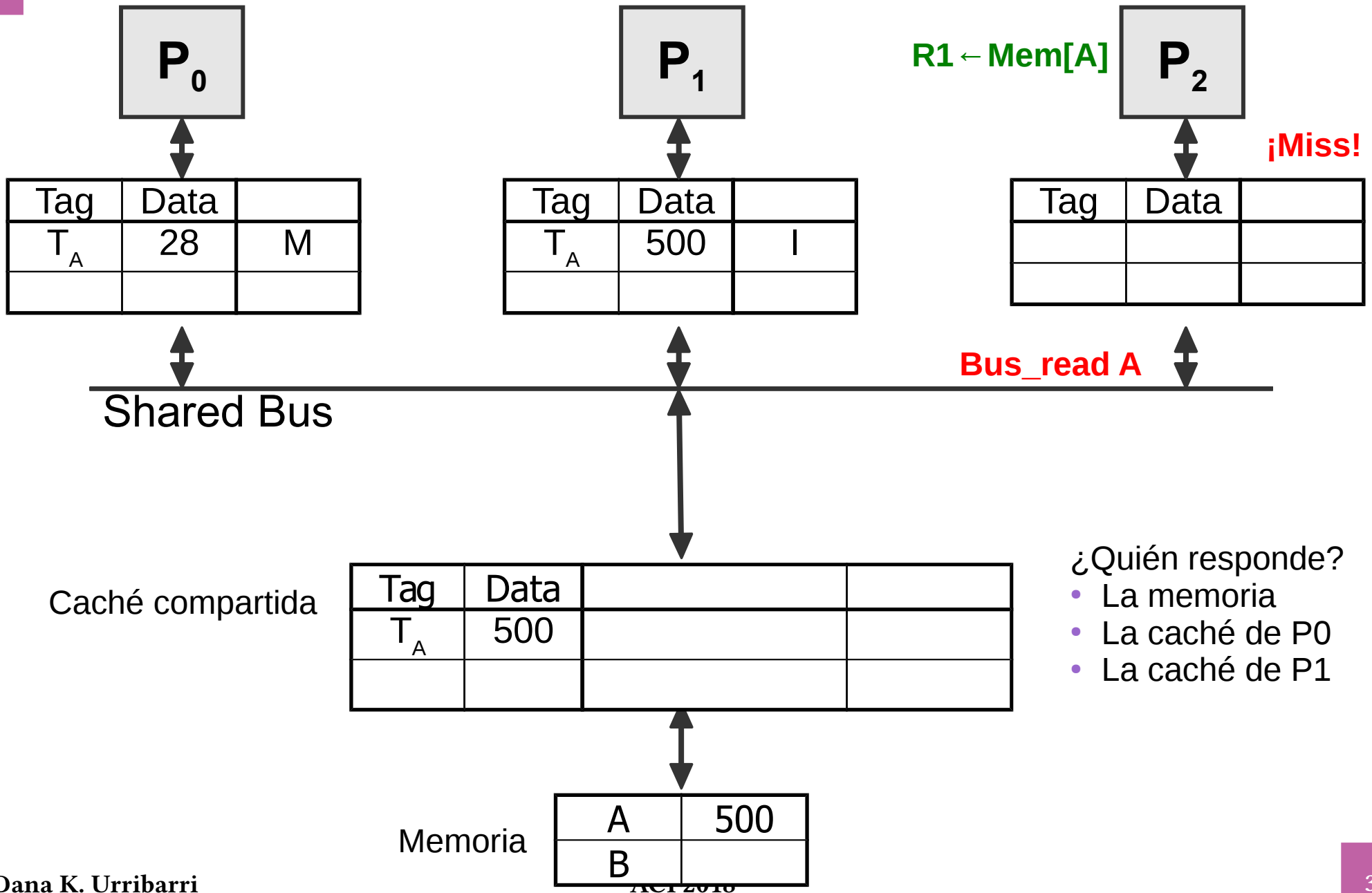
Ejemplo...


P0
Mem[A] ← 28
P1
P2
Tag
Data
TA
28
M
500
S
Bus_write A
Shared Bus
Caché compartida
Memoria
A
B

Ejemplo...


P0
Mem[A] ← 28
P1
P2
Tag
Data
TA
28
M
500
I
Bus_write A
Shared Bus
Caché compartida
Memoria
A
B

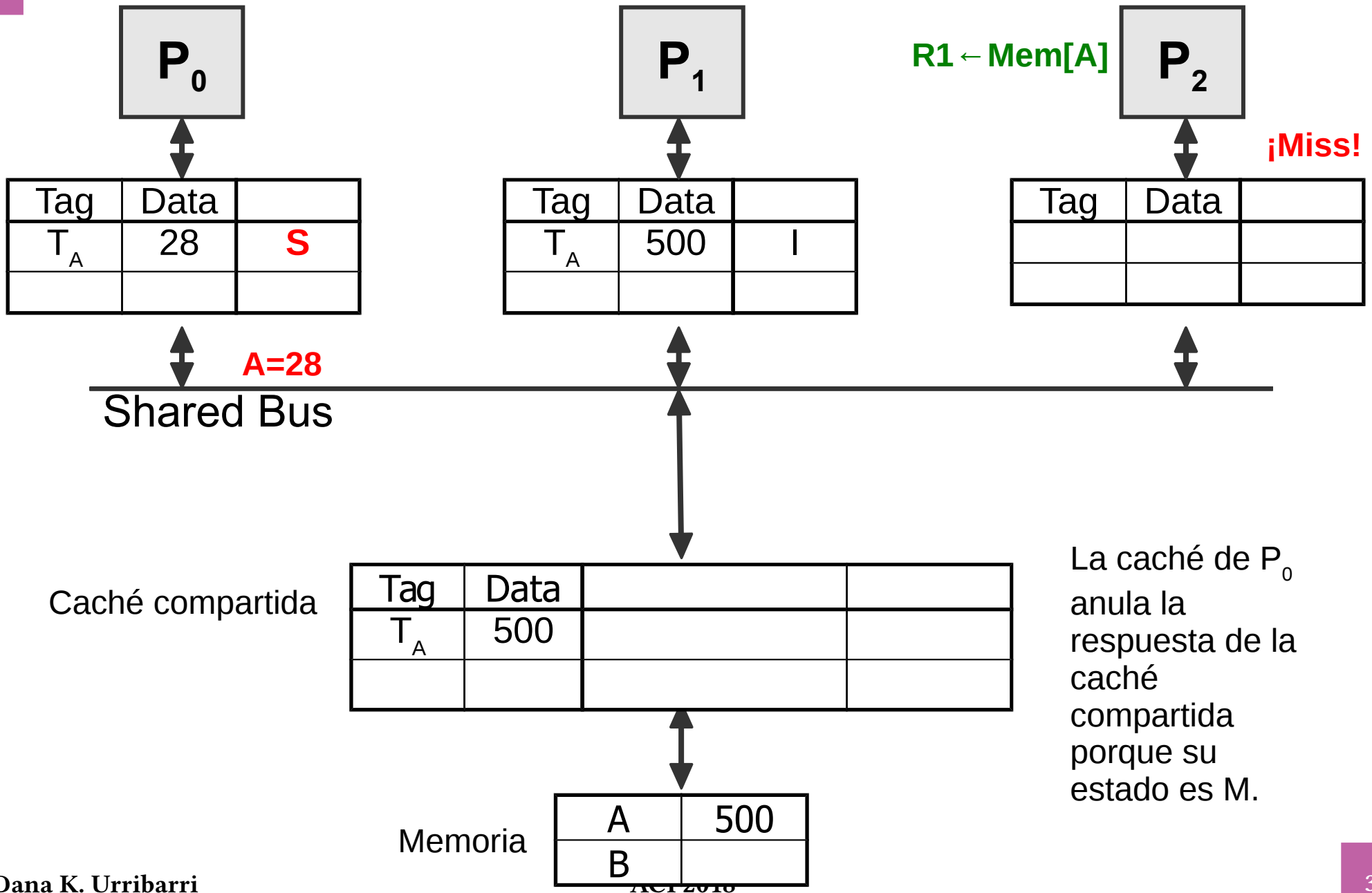
Ejemplo...



¿Quién responde?

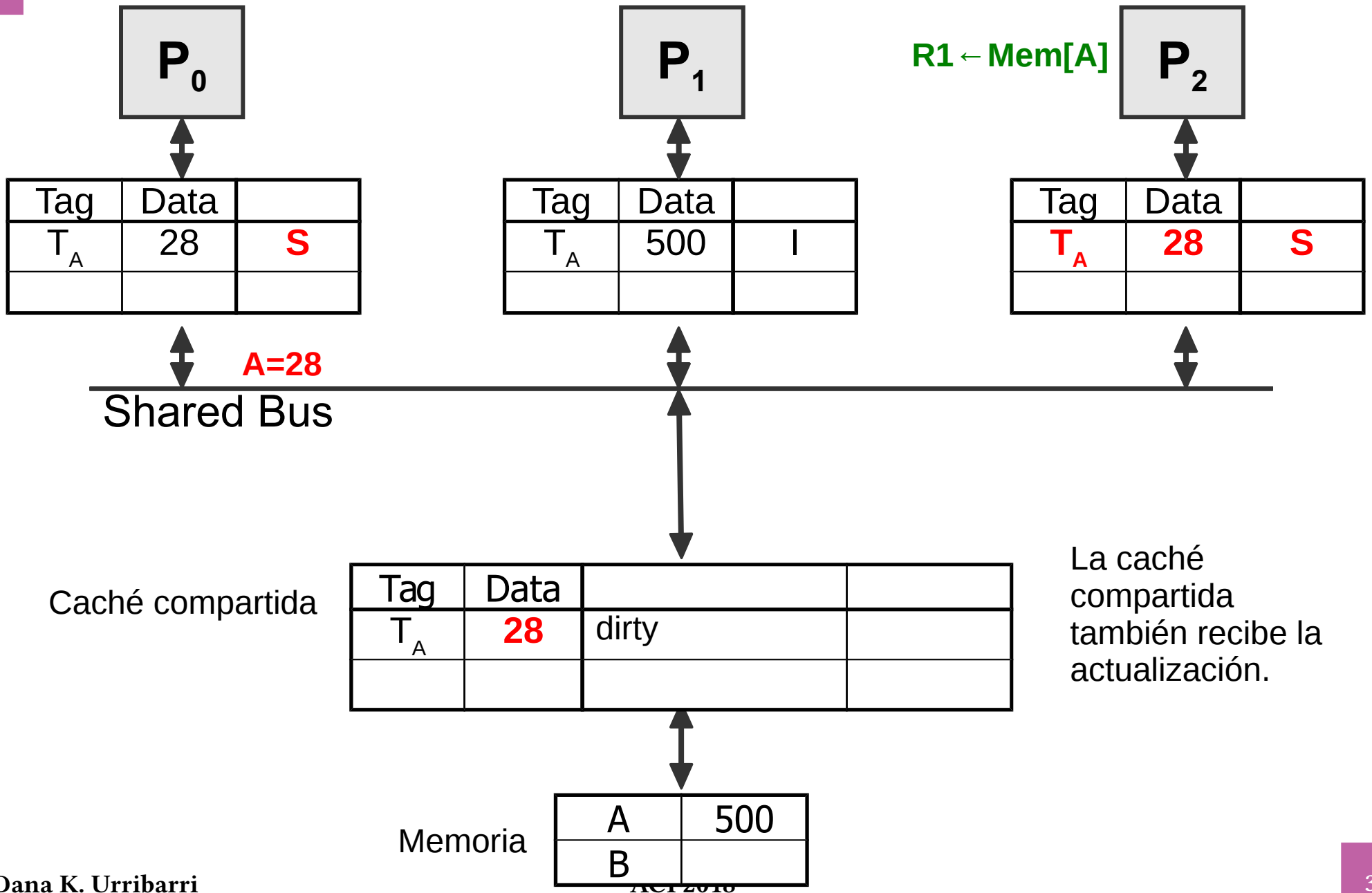
- La memoria
- La caché de P0
- La caché de P1

Ejemplo...



La caché de P_0 anula la respuesta de la caché compartida porque su estado es M.

Ejemplo

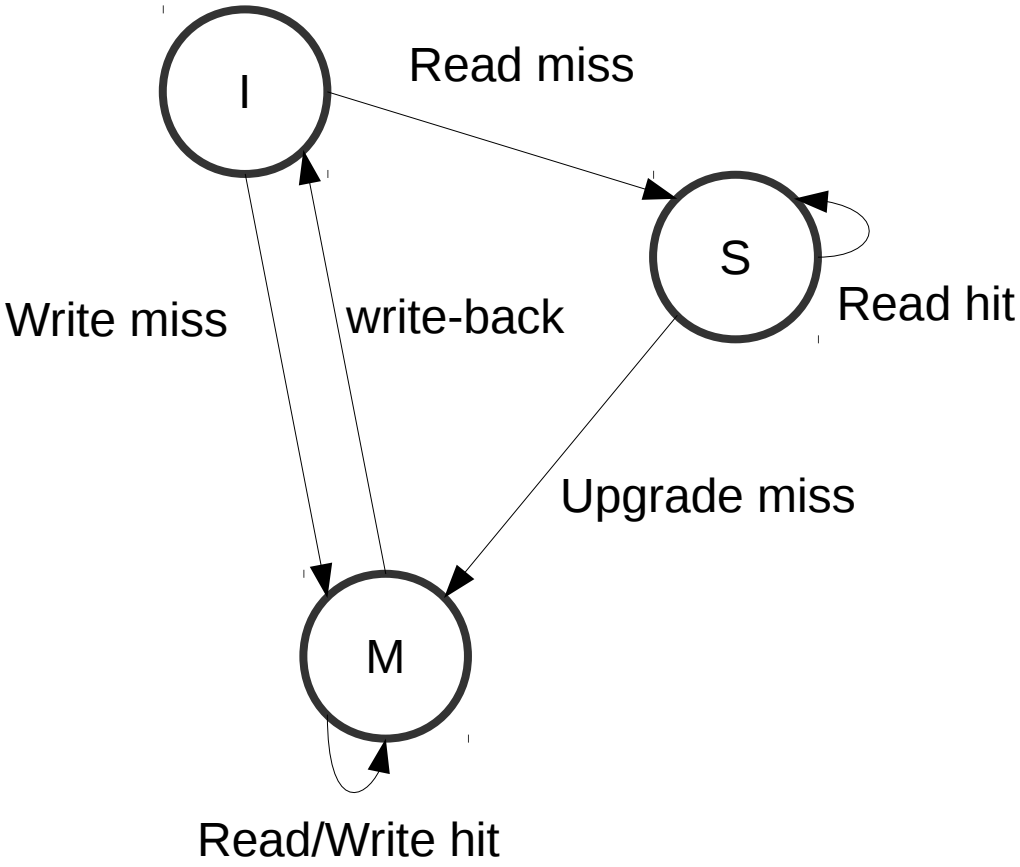


La caché compartida también recibe la actualización.

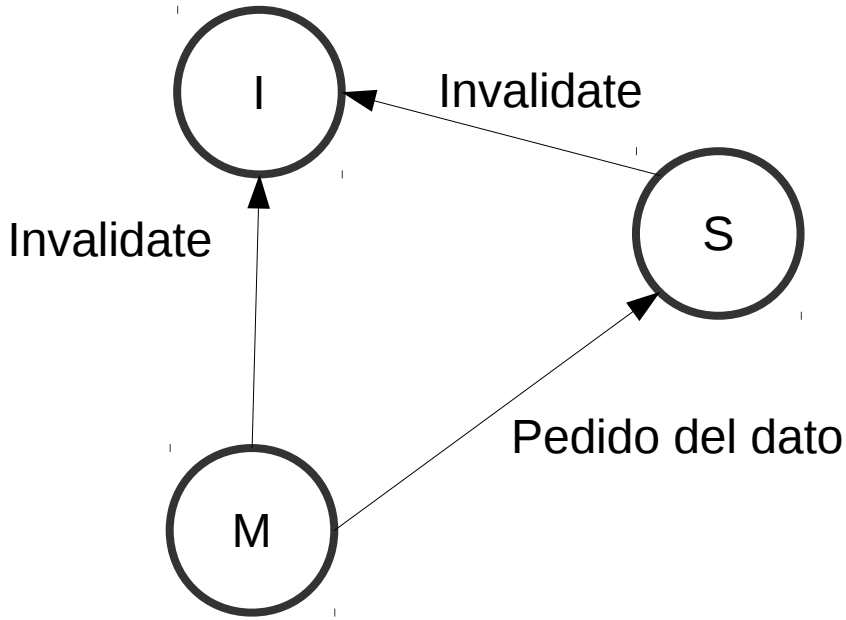
MSI + Directorio

Estado de una línea en la caché local

Según acciones del procesador local



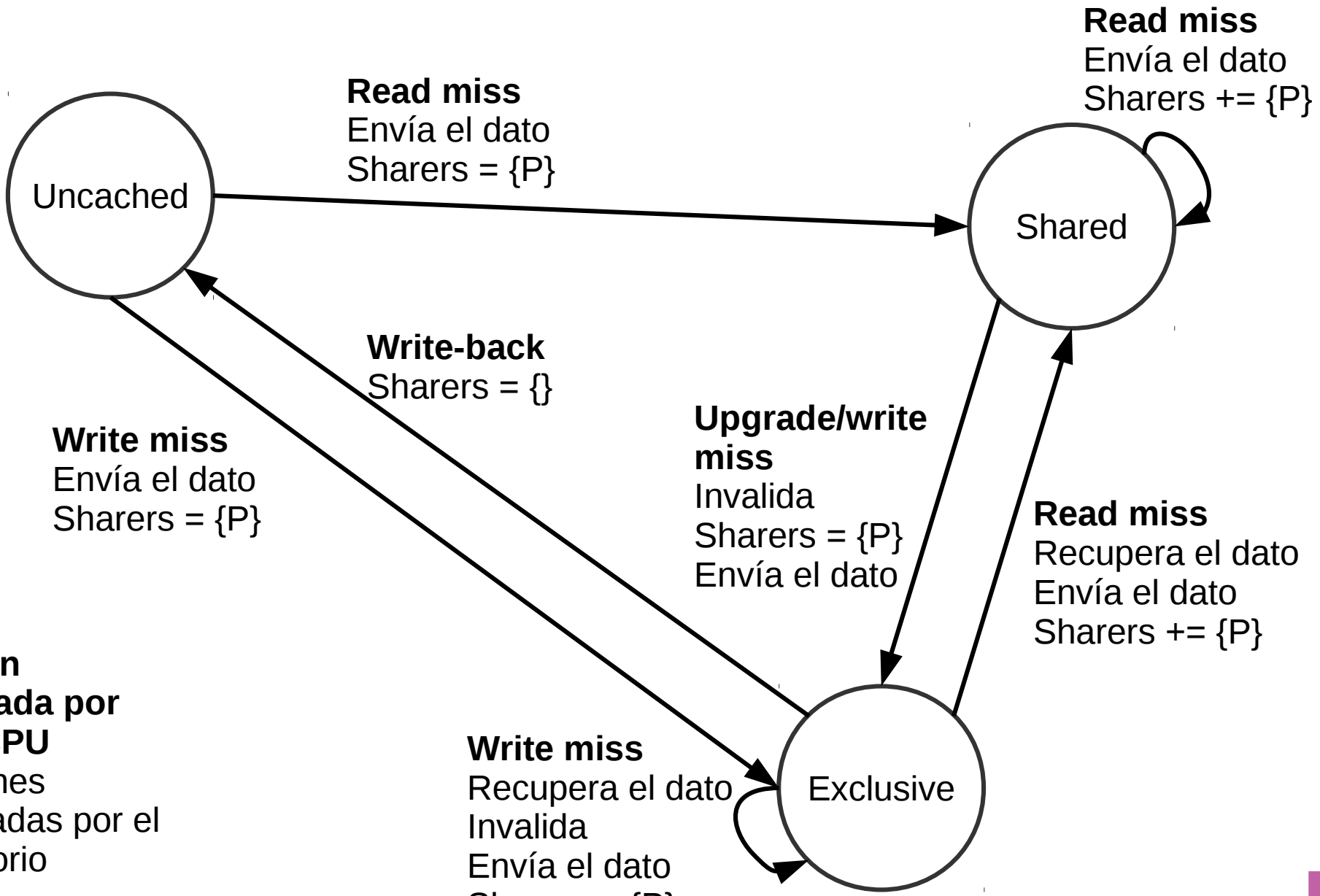
Según señales externas



Estados de una línea en el directorio

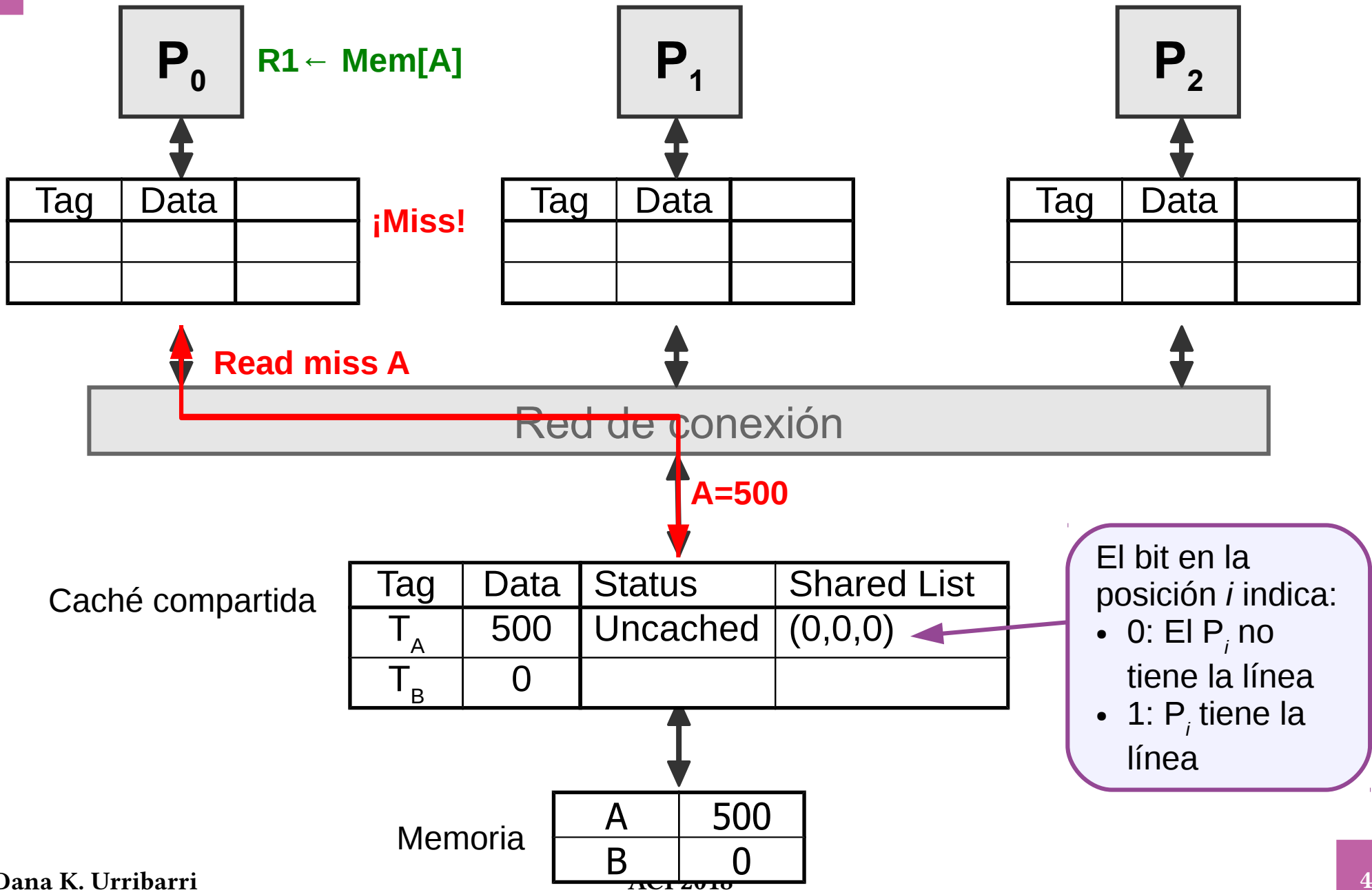
- Hay tres posibles estados:
  - *Uncached*: ninguna caché tiene la línea
  - *Shared*: la línea está en varias caché como sólo lectura
  - *Exclusive*: hay una caché que tiene la única copia válida de la línea.
- Pueden ocurrir cuatro eventos:
  - *Read/write miss*: un procesador quiere leer/escribir una línea que no tiene en caché.
  - *Upgrade miss*: un procesador quiere escribir una línea que tiene en caché pero como sólo lectura.
  - *Write-back*: un procesador escribe una línea en el nivel inferior antes de reemplazarla.
- ¿En qué estados puede ocurrir cada evento?

Estados de una línea en el directorio

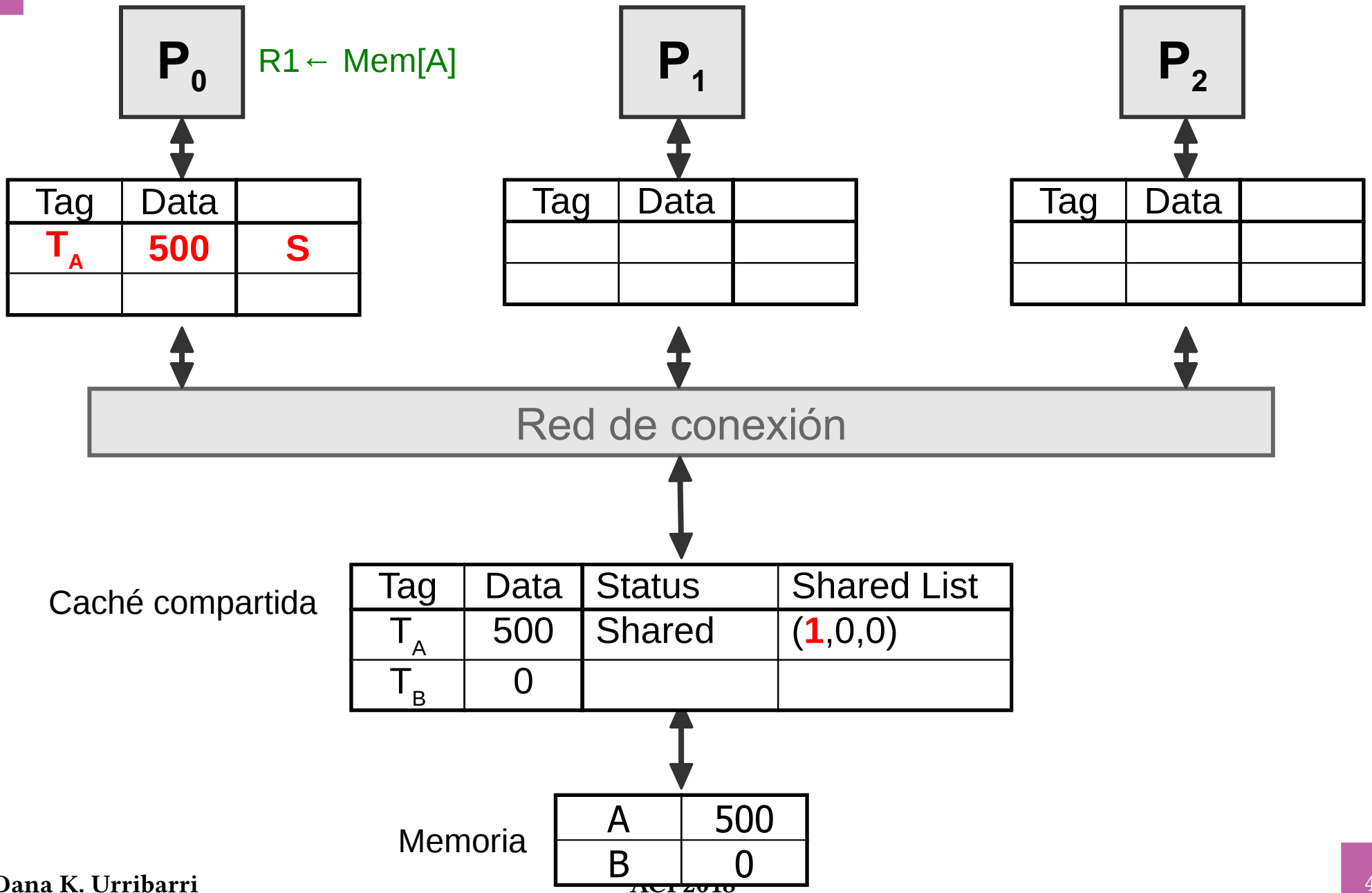


- **Acción realizada por una CPU**
- Acciones realizadas por el directorio

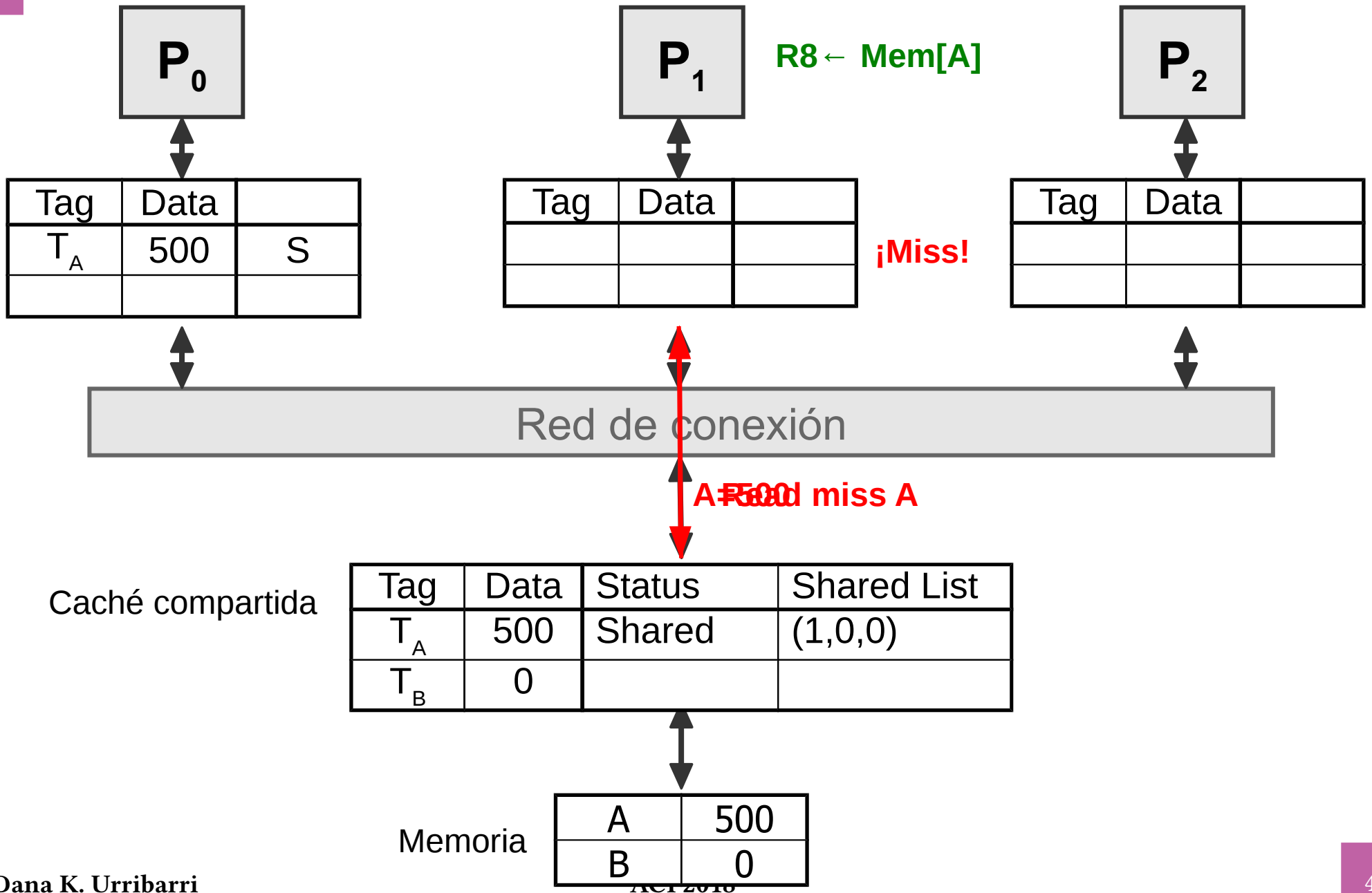
Ejemplo...


P0
R1 ← Mem[A]
P1
P2
Tag
Data
¡Miss!
Read miss A
Red de conexión
A=500
Caché compartida
Tag
Data
Status
Shared List
TA
500
Uncached
(0,0,0)
TB
0
El bit en la posición i indica:
0: El Pi no tiene la línea
1: Pi tiene la línea
Memoria
A
500
B
0

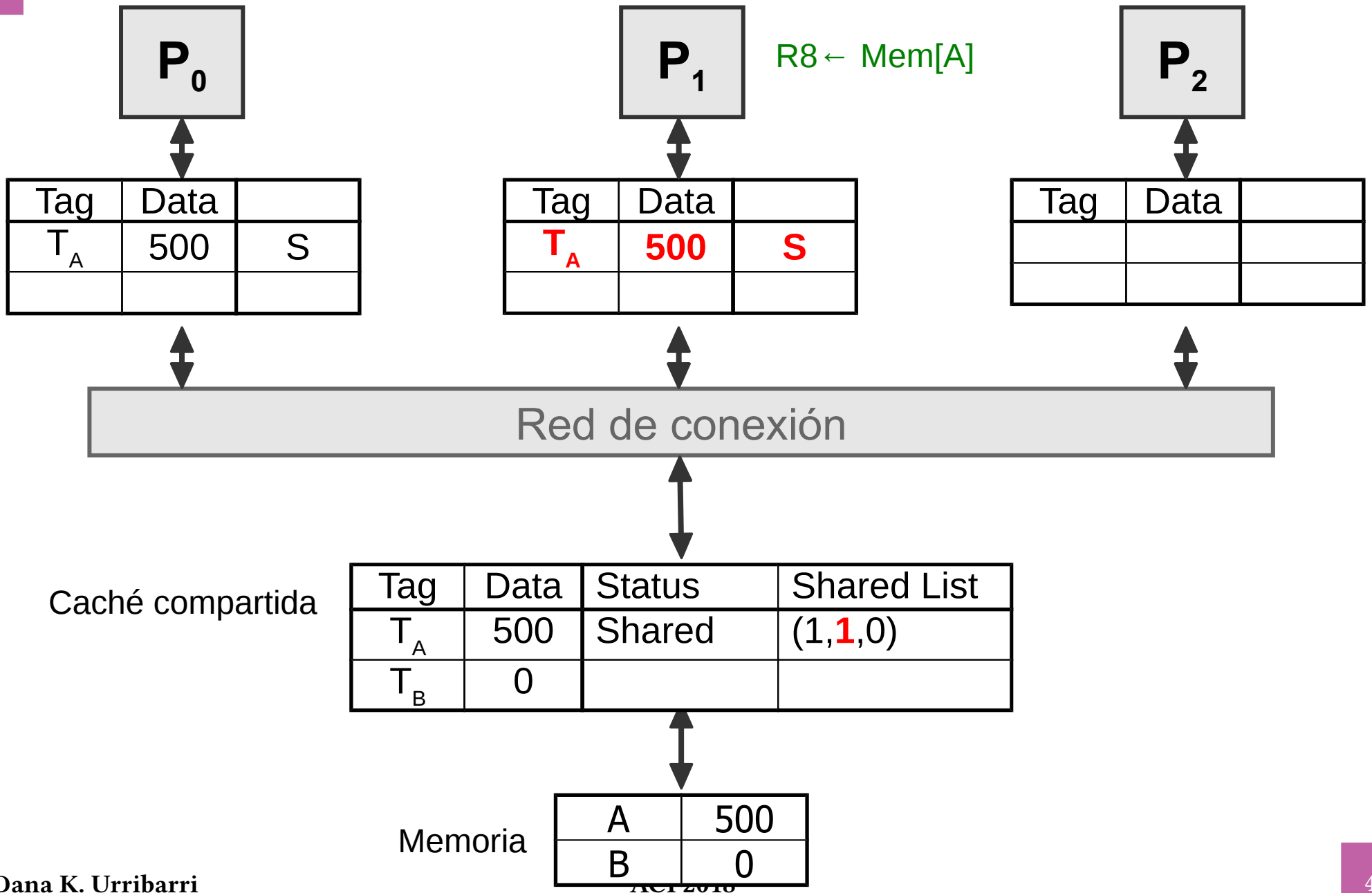
Ejemplo...


P0
R1 ← Mem[A]
P1
P2
Tag
Data
TA
500
S
Red de conexión
Caché compartida
Tag
Data
Status
Shared List
TA
500
Shared
(1,0,0)
TB
0
Memoria
A
500
B
0

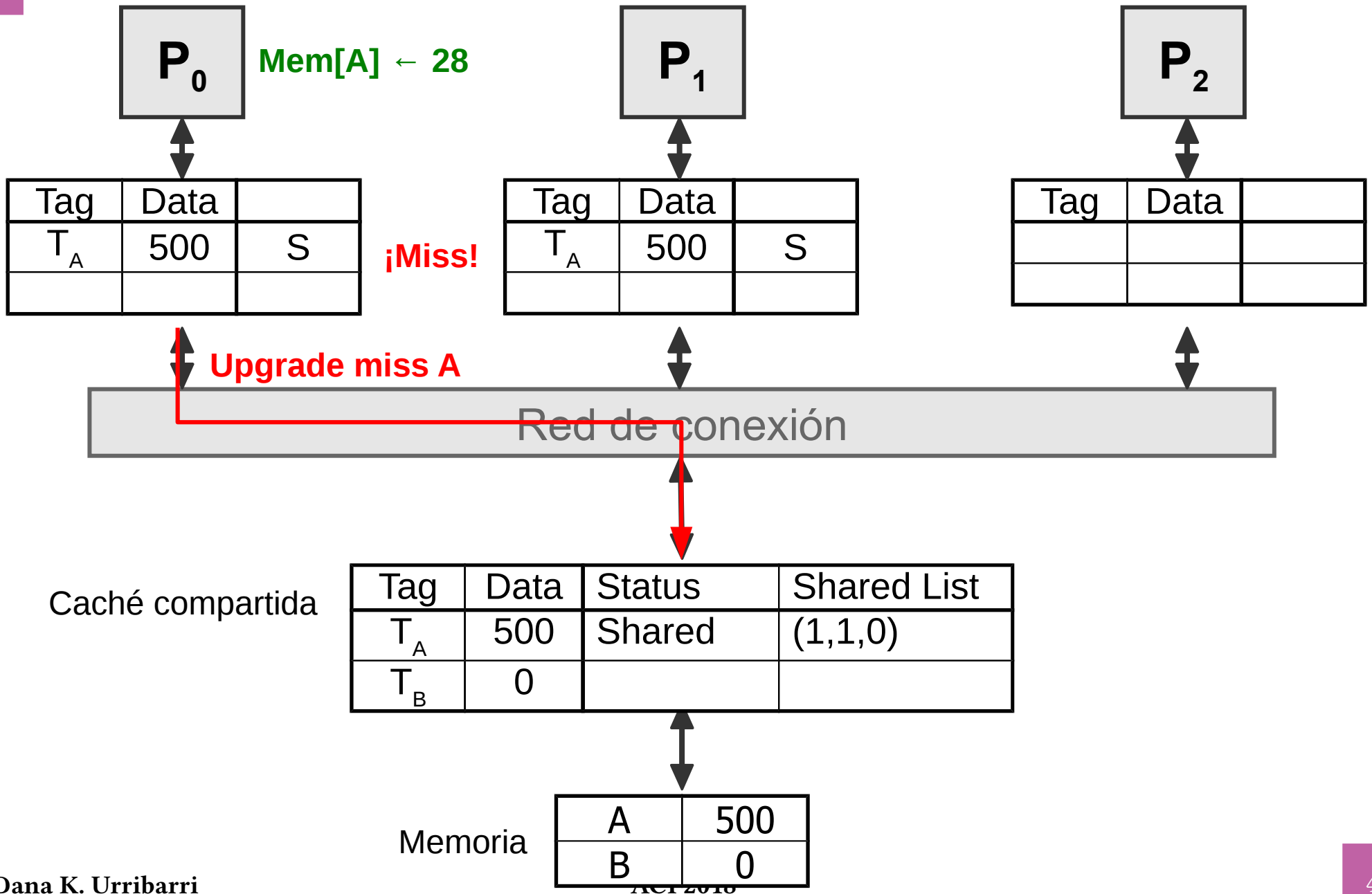
Ejemplo...


P0
P1
R8 ← Mem[A]
P2
Tag
Data
TA
500
S
¡Miss!
Red de conexión
Read miss A
Caché compartida
Status
Shared List
Shared
(1,0,0)
TB
0
Memoria
A
500
B
0

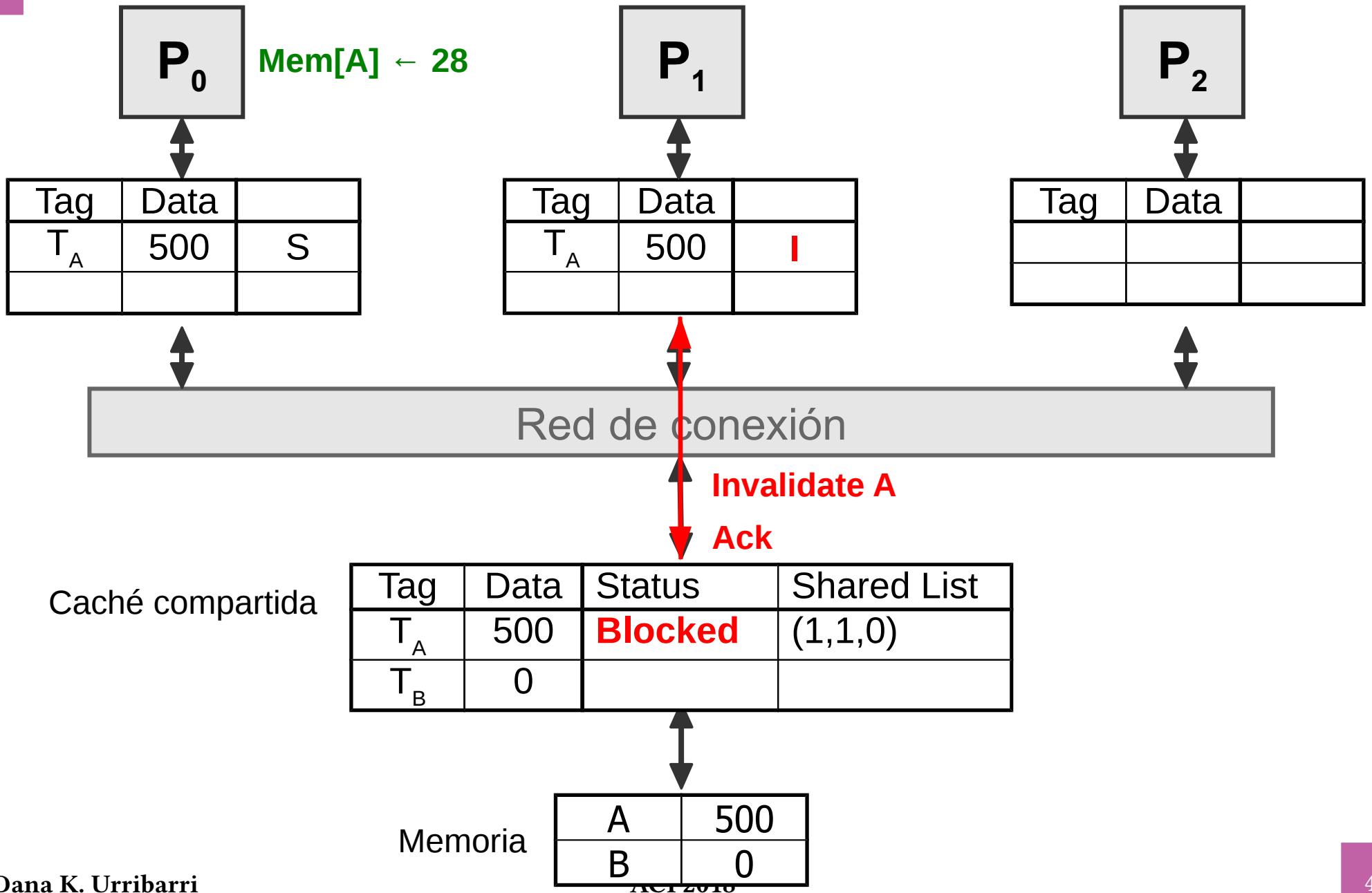
Ejemplo...


P0
P1
R8 ← Mem[A]
P2
Tag Data
TA 500 S
Tag Data
TA 500 S
Tag Data
Red de conexión
Caché compartida
Tag Data Status Shared List
TA 500 Shared (1,1,0)
TB 0
Memoria
A 500
B 0

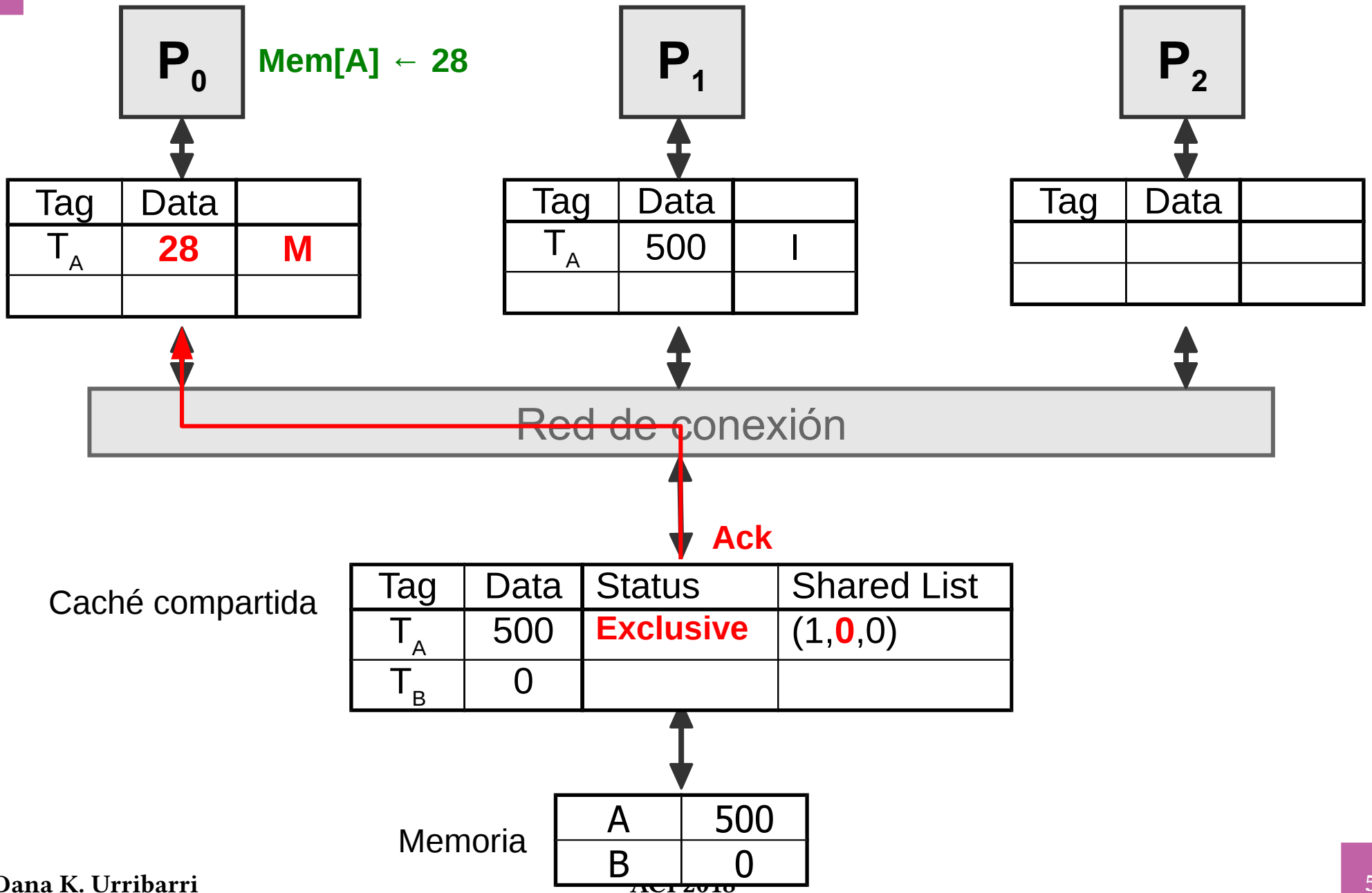
Ejemplo...


P0
Mem[A] ← 28
P1
P2
Tag
Data
TA
500
S
¡Miss!
Upgrade miss A
Red de conexión
Caché compartida
Tag
Data
Status
Shared List
TA
500
Shared
(1,1,0)
TB
0
Memoria
A
500
B
0

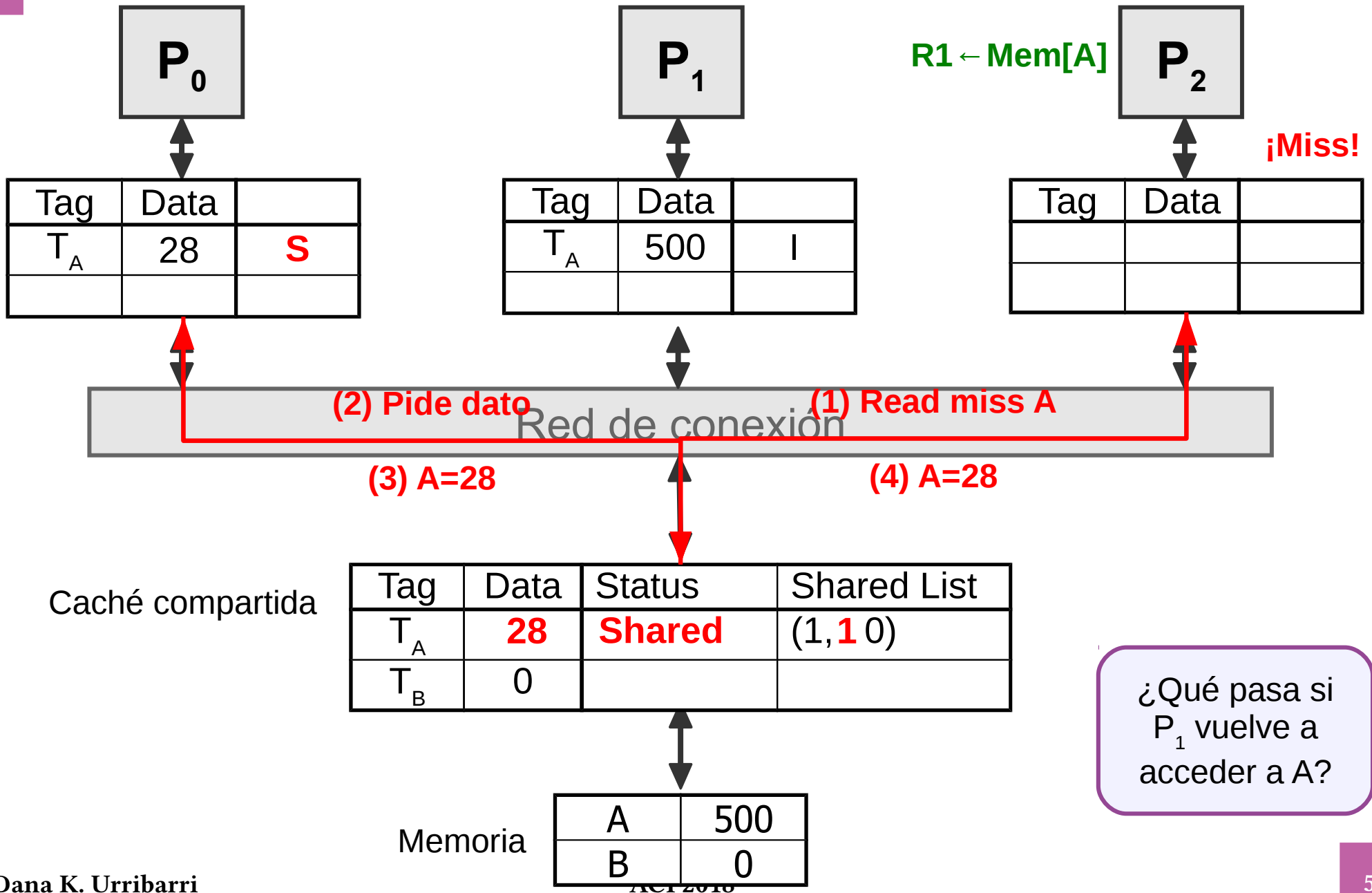
Ejemplo...


P0
Mem[A] ← 28
P1
P2
Tag
Data
TA
500
S
I
Red de conexión
Invalidate A
Ack
Caché compartida
Status
Shared List
Blocked
(1,1,0)
TB
0
Memoria
A
B

Ejemplo...


P0
Mem[A] ← 28
P1
P2
Tag
Data
TA
28
M
TA
500
I
Red de conexión
Ack
Caché compartida
Tag
Data
Status
Shared List
TA
500
Exclusive
(1,0,0)
TB
0
Memoria
A
500
B
0

Ejemplo



¿Qué pasa si P_1 vuelve a acceder a A?

Tipos de fallos en caché

- Capacitivos
- Compulsivos
- Conflictivos
- De coherencia (*Coherence miss*)

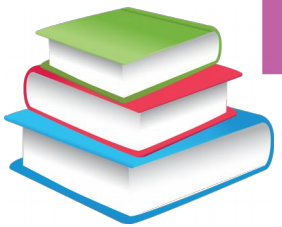
Fallos de coherencia

- Resultado de la comunicación entre procesadores.
- Motivos:
  - Real existencia de datos compartidos.
  - Falsos datos compartidos.

    Distintos procesadores acceden a distintos datos pero que están en el mismo bloque.

    Dependientes del tamaño del bloque.

Bibliografía



- *Capítulo 7*. *Multiprocessor Architecture. From simple pipelines to chip multiprocessor*. Jean-Loup Baer. Cambridge University Press. 2010.
- *Capítulo 5*. *Computer Architecture. A Quantitative Approach.* John L. Hennessy & David A. Patterson. Elsevier Inc. 2012, 5ta Ed.

Suplementaria

- *Capítulo 6*. *Computer Organization and Design. The Hardware/Software Interface.* David A. Patterson & John L. Hennessy. Elsevier Inc. 2014, 5ta Ed.