Escuela de Ingeniería y Arquitectura
Universidad Zaragoza

Departamento de Informática e Ingeniería de Sistemas
Universidad Zaragoza

Tema 9 – Descripción de Comportamiento

AOC2
Grado en Ing. Informática

P. Ibáñez, J.L. Briz, V. Viñals, J. Alastruey, J. Resano
Arquitectura y Tecnología de Computadores
Departamento de Informática e Ingeniería de Sistemas

Guión del tema

- Notación
- Descripción mediante Diagrama de estados
- Ejercicio: diagrama para *write once*
- Descripción mediante Algoritmo
- Calculo de Tiempo Efectivo de Acceso, ejemplos

Notación (1 de 3)

1. Datos y direcciones

- Palabra: minúscula con prima — *x'*

Bloque **x**

- Bloque: minúsculas
  - *que referencia el procesador* — *x*
  - *expulsado de Mc* — *u* — víctima
- Dirección de — *@*
- Dato y su dirección: mayúscula
  - *X* = <*@x*, *x*> — bloque
  - *U* = <*@u*, *u*> — bloque
  - *X'*= <*@x'*, *x'*> — palabra

Notación (2 de 3)

2. Enviar comando desde Mc a Mp

evento: comp_destino (comando, item)

rh: read hit
wh: write hit
rm: read miss
wm: write miss
rpl: replacement

Mp

rB: read block
wB: write block
rW: read word
wW: write word

parámetros del comando: @x, U, …

Comandos entre otras parejas de componentes;
por ejemplo *desde* Mc1 *hacia* Mc2

evento: Mc1>Mc2 (comando, item)

Notación (3 de 3)

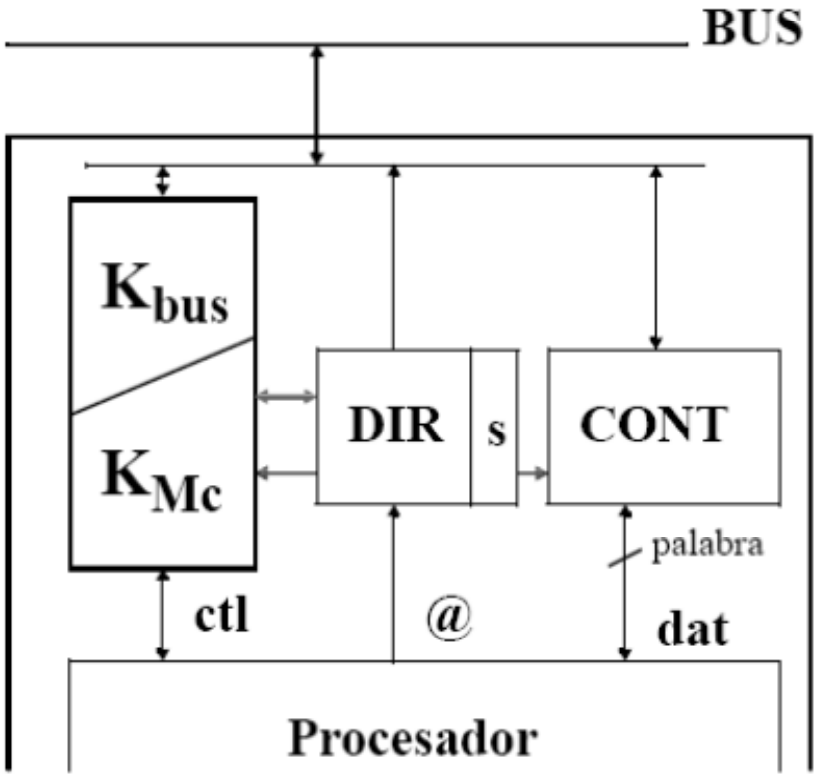
3. Cargar bloque o palabra en Mc
   - Operador "**+**"

     Mc + *x*; después de un fallo,
     el bloque *x* se carga en Mc

4. Invalidar bloque en Mc
   - Operador "**-**"

     Mc - *u*; invalida el bloque *u*
     el contenedor puede recibir un nuevo bloque

Papel del controlador de bus y Mc



- Estado S de cada bloque según las reglas de escritura y reemplazo
  - validez, sucio/limpio, …
  - Estado de reemplazo
- Primer nivel en chip normalmente separado: datos e instrucciones
- Siguientes niveles en chip normalmente unificados

- **K_{Mc}**: implementa comunicación con el procesador y cambio de estado
- **K_{bus}**: implementa el autómata de comunicación con el bus

K_{BUS} + K_{Mc}: una máquina de estados

- Describimos un contenedor cualquiera de Mc
- Sólo detallamos relación Mc – Bus
  - P.e. no ponemos **`CPU+x´`** o **`Mc+x`**

1. Localizar

- selección de contenedor
  → el que contiene el bloque ***x*** o el bloque víctima ***u***
- comprobación de acierto/fallo

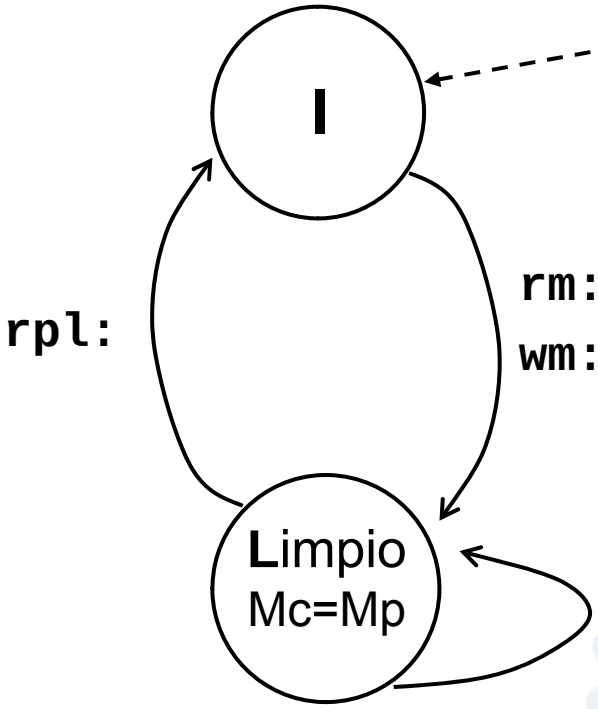
Esto no se ve en el diagrama de estados

2. Acierto: **1** transición (**`rh`** o **`wh`**)

Fallo: **1** transición (**`rpl`**) + **1** transición (**`rm`** o **`wm`**)

Escritura Inmediata – *Write Through* + *AF*

2 estados: Inválido y Limpio. Un solo bit por bloque (V bit)



rm: Mp(rB,@x)
wm: Mp(rB,@x); Mp(wW,X')

rh:
wh: Mp(wW, X')

Fallo en escritura:
Fetch on write-miss (AF)

COPIAS en Mc
COHERENTES con Mp

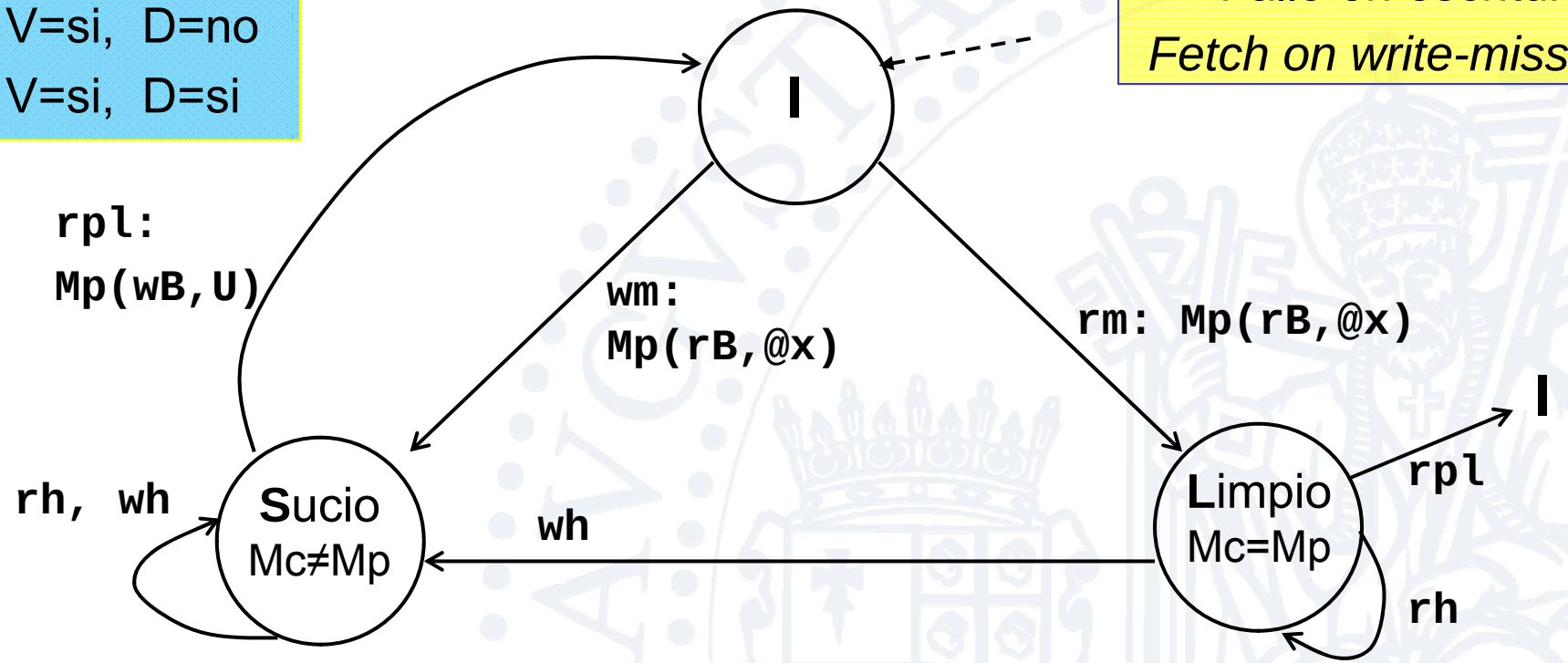
Comandos	¿cuándo?	comentario
Mp(rB,@x)	**m (r ó w)**	leemos bloque en fallo
Mp(wW,X')	**w (h ó m)**	escribimos palabra en Mp siempre

Escritura Retardada – *Copy-Back + AF*

3 estados: Inválido, Limpio y Sucio. Dos bits por bloque: V bit + D bit

I: V=no, D=¿?
L: V=si, D=no
S: V=si, D=si

Fallo en escritura:
Fetch on write-miss (AF)



Comandos	¿cuándo?	comentario
Mp(wW,	**nunca**	No es posible
Mp(wB,	**rpl ^ S**	El algoritmo de reemplazo selecciona un bloque víctima *u* y lo envía a memoria principal
Mp(rB,	**m**	Se carga el bloque *x*. En caso de escritura se cambia el valor de *x'* en el bloque recién traído

Ejercicio: Write Once

Aplica Write-Through en el primer acierto o fallo de escritura.

En el resto aplica Write-Back.

Política de carga de bloques: fetch on write-miss (AF)

Utilizado en el Motorola M88200

4 estados: Inválido, Limpio 0, Limpio 1 y Sucio.

Dos bits por bloque: V bit + D bit

M88200: chip con 16KB de SRAM cache + controlador pensado para funcionar con el micro Motorola 88000 (1990)

Escritura Inmediata Sin Buffering + AF

- Algoritmo

Especificación	Tiempos
`look-up(@x);`	1
`if miss(@x) {Mp(rB,@x); waitfor Mp; Mc+X;}`	CrB + 1
`MRU(@x);`	0
`switch (proc_r/w)`	
`case proc_r: ret x';`	0
`case proc_w: {Mc+x'; Mp(wW,X‘); waitfor Mp; ret;}`	CwW

- CrB lectura de bloque desde Mp (latencia + transferencia)
- CwW escritura de palabra en Mp

Escritura Inmediata Sin Buffering + AF

- Duración media de un acceso a datos = ciclos efectivos de acceso (ciclos/ref)

$$Ceff = 1 + \frac{\sum wh \times CwW}{\sum refs} + \frac{\sum rm \times (CrB + 1)}{\sum refs} + \frac{\sum wm \times (CrB + 1 + CwW)}{\sum refs}$$

- CrB + 1 penalización de fallo en lectura
- CrB + 1 + CwW penalización de fallo en escritura

Escritura Retardada sin Buffering + AF

- Algoritmo

Especificación	Tiempos
`look-up(@x);`	1
`if miss(@x) {` `  u=LRU(); Mc-u;`	0
`  if sucio (u)`	
`    {Mp(wB,U); waitfor Mp;}`	CwB
`  Mp(rB,@x); waitfor Mp; Mc+X;`	CrB+1
`  MRU(@x);`	0

`  switch (proc_r/w)`
`    case proc_r: ret x';`
`    case proc_w: {Mc+x'; sucio(x)=true; ret;}`
	0
	1

- CwB escritura del bloque hacia Mp (transferencia + latencia)
- CrB lectura de bloque desde Mp (latencia + transferencia)

Escritura Retardada Sin Buffering

- Cálculo del tiempo efectivo en ciclos (ciclos/ref)

$$Ceff = 1 + \frac{\sum wh \times 1}{\sum refs} + \frac{\sum rm \times (CrB + 1)}{\sum refs} + \frac{\sum wm \times (CrB + 2)}{\sum refs} + \frac{\sum (m \wedge sucio) \times CwB}{\sum refs}$$

- CrB + 1 penalización de fallo en lectura
- CrB + 2 penalización de fallo en escritura
- CwB penalización de fallo sucio