2º curso / 2º cuatr.

Grado en Ing. Informática

Arquitectura de Computadores
Tema 3

Arquitecturas con paralelismo a nivel de thread (TLP)

Material elaborado por los profesores responsables de la asignatura:
Mancia Anguita – Julio Ortega

ugr Universidad de Granada

ETSIIT
Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

ATC
Departamento de Arquitectura y Tecnología de Computadores
UNIVERSIDAD DE GRANADA

Lecciones

- Lección 7. Arquitecturas TLP
- Lección 8. Coherencia del sistema de memoria
  - Sistema de memoria en multiprocesadores
  - Concepto de coherencia en el sistema de memoria: situaciones de incoherencia y requisitos para evitar problemas en estos casos
  - Protocolos de mantenimiento de coherencia: clasificación y diseño
  - Protocolo MSI de espionaje
  - Protocolo MESI de espionaje
  - Protocolo MSI basado en directorios con o sin difusión
- Lección 9. Consistencia del sistema de memoria
- Lección 10. Sincronización

sincronización

jerarquía

Objetivos Lección 8

- Comparar los métodos de actualización de memoria principal implementados en cache.
- Comparar las alternativas para propagar un escritura en protocolos de coherencia de cache.
- Explicar qué debe garantizar el sistema de memoria para evitar problemas por incoherencias.
- Describir las partes en las que se puede dividir el análisis o el diseño de protocolos de coherencia.
- Distinguir entre protocolos basados en directorios y protocolos de espionaje (snoopy).
- Explicar el protocolo de mantenimiento de coherencia de espionaje MSI.
- Explicar el protocolo de mantenimiento de coherencia de espionaje MESI.
- Explicar el protocolo de mantenimiento de coherencia MSI basado en directorios con difusión y sin difusión.

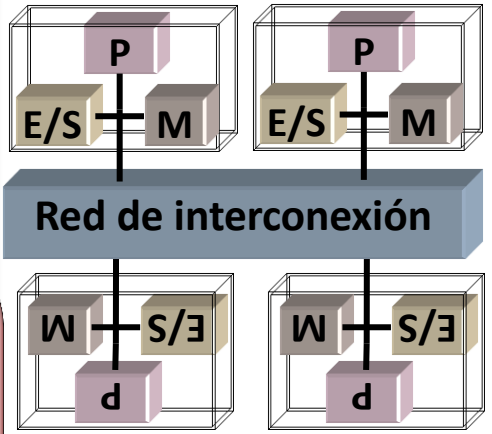
Bibliografía Lección 8

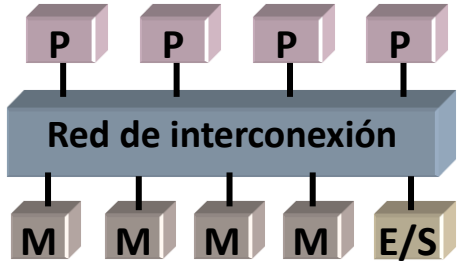
- Fundamental
  - Cap. 3, Secc. 3. M. Anguita, J. Ortega. *Fundamentos y Problemas de Arquitectura de Computadores.* Librería Fleming/Ed. Avicam, 2016.
  - Secc. 10.1. J. Ortega, M. Anguita, A. Prieto. *Arquitectura de Computadores*. Thomson, 2005. ESII/C.1 ORT arq
- Complementaria
  - T. Rauber, G. Ründer. *Parallel Programming: for Multicore and Cluster Systems*. Springer 2010. Disponible en línea (biblioteca UGR): http://dx.doi.org/10.1007/978-3-642-04818-0

Computadores que implementan en hardware mantenimiento de coherencia

Multi-computadores Memoria no compartida	**NORMA** *No Remote Memory Access*	nivel de sistema (*Cluster*), armario, chasis (*blade server*)	Memoria físicamente distribuida
Multi-procesadores Memoria compartida Un único espacio de direcciones	**NUMA** *Non-Uniform Memory Access*	**NUMA** (nivel de sistema,n. armario/chasis, n. placa)	
		CC-NUMA (nivel de armario: SGI Altix; nivel de placa)	
		COMA	
	UMA *Uniform Memory Access*	**SMP** *Symmetric MultiProcessor* (nivel de placa; nivel de chip: multicores como Intel Core i7, i5, i3)	Memoria físicamente centralizada

Coherencia por hardware





+ Escalabilidad –

Contenido Lección 8

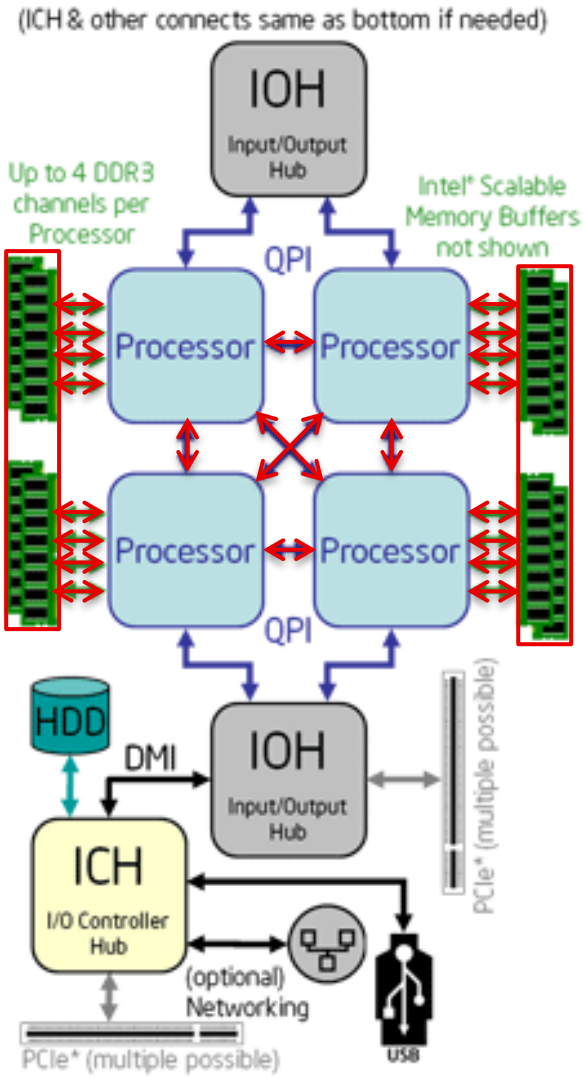
- Sistema de memoria en multiprocesadores
- Concepto de coherencia en el sistema de memoria: situaciones de incoherencia y requisitos para evitar problemas en estos casos
- Protocolos de mantenimiento de coherencia: clasificación y diseño
- Protocolo MSI de espionaje
- Protocolo MESI de espionaje
- Protocolo MSI basado en directorios con o sin difusión

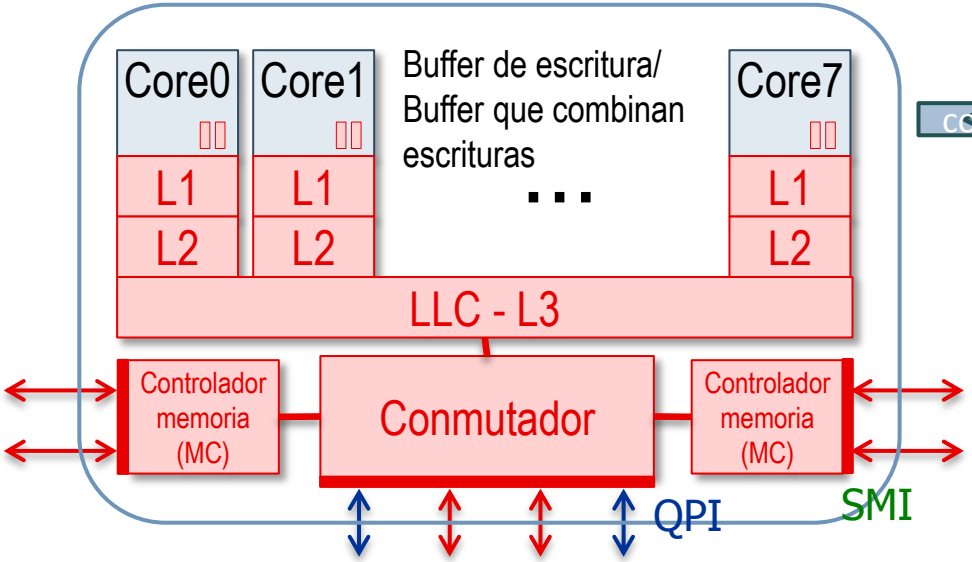
Sistema de memoria en multiprocesadores

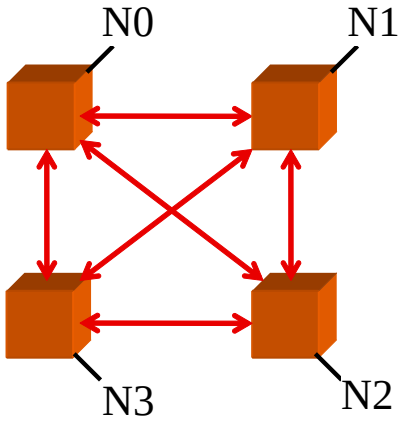
- ¿Qué incluye?
  - Caches de todos los nodos
  - Memoria principal
  - Controladores
  - Buffers:
    - Buffer de escritura/almacenamiento
    - Buffer que combinan escrituras/almacenamientos, etc.
  - Medio de comunicación de todos estos componentes (red de interconexión)
- La comunicación de datos entre procesadores la realiza el sistema de memoria

  - La lectura de una dirección debe devolver lo último que se ha escrito (desde el punto de vista de todos los componentes del sistema)

Sistema de memoria





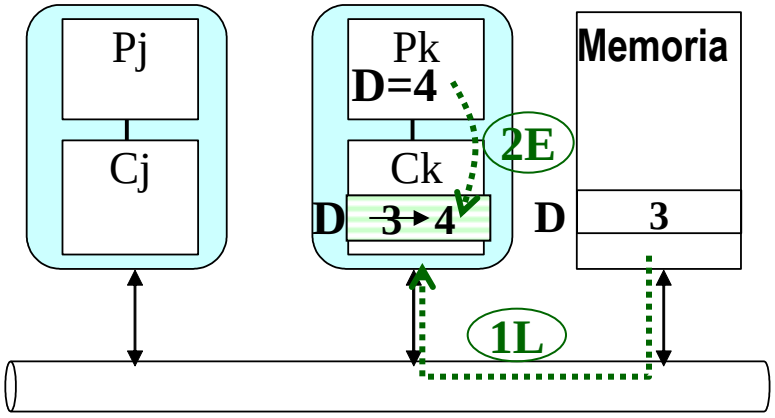


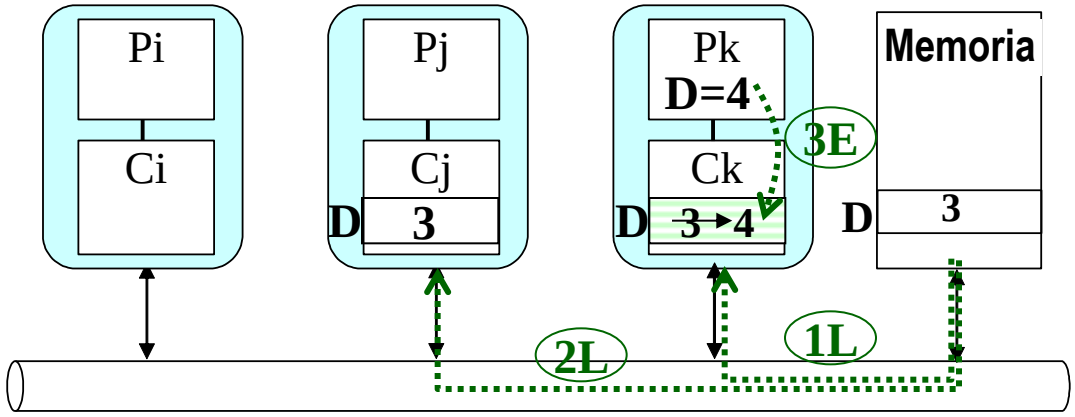
Contenido Lección 8

- Sistema de memoria en multiprocesadores
- Concepto de coherencia en el sistema de memoria: situaciones de incoherencia y requisitos para evitar problemas en estos casos
- Protocolos de mantenimiento de coherencia: clasificación y diseño
- Protocolo MSI de espionaje
- Protocolo MESI de espionaje
- Protocolo MSI basado en directorios con o sin difusión

Incoherencia en el sistema de memoria

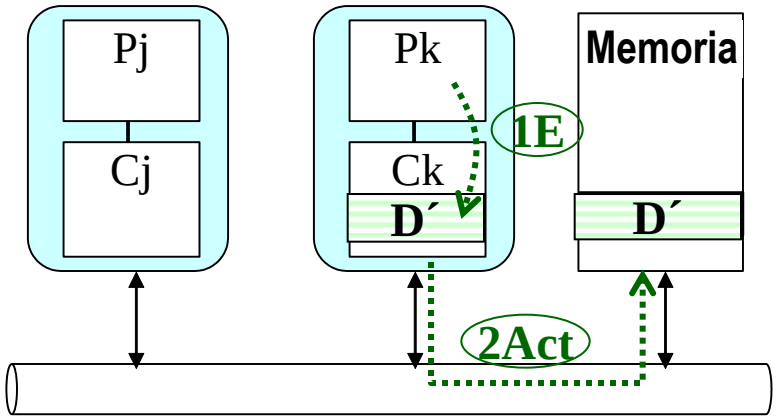
Clases de estructuras de datos	Eventos que ponen de manifiesto faltas de coherencia	Tipos de Falta de coherencia
Datos modificables	E/S	Cache-MP
Datos modificables compartidos	Fallo de cache	Cache-MP
Datos modificables privados	Emigra thread/proceso→Fallo cache	Cache-MP
Datos modificables compartidos	Lectura de cache no actualizada	Cache-Cache





Métodos de actualización de memoria principal implementados en caches

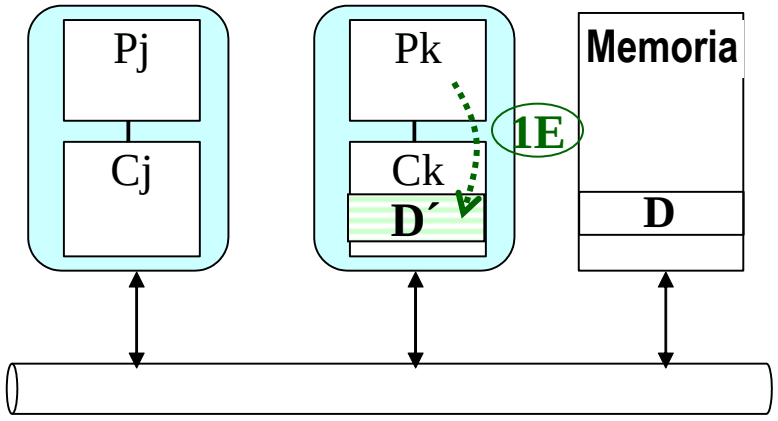
1. Escritura inmediata (*write-through*):



Cada vez que un procesador escribe en su cache escribe también en memoria principal

Por los principios de localidad temporal y espacial sería más rentable si se escribe todo el bloque una vez realizadas múltiples escrituras

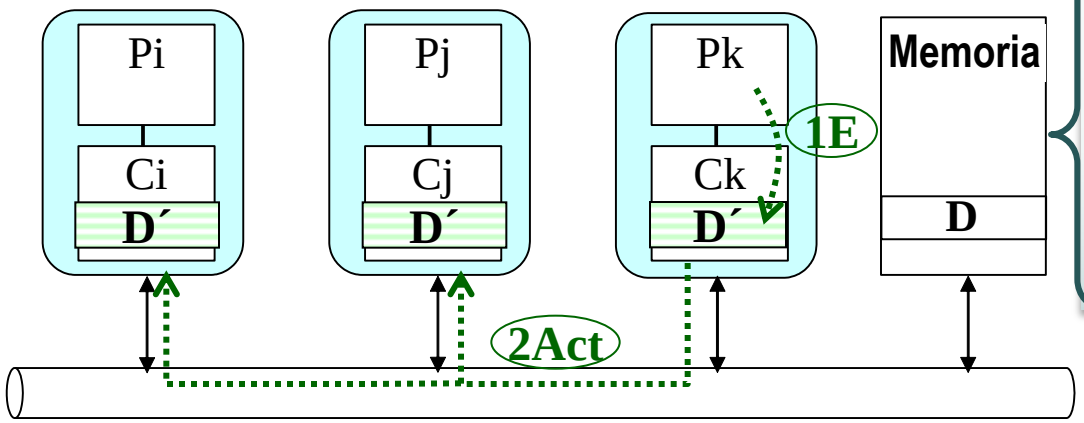
2. Posescritura (*write-back*):



Se actualiza memoria principal escribiendo todo el bloque cuando se **desaloja** de la cache

Alternativas para propagar una escritura en protocolos de coherencia de cache

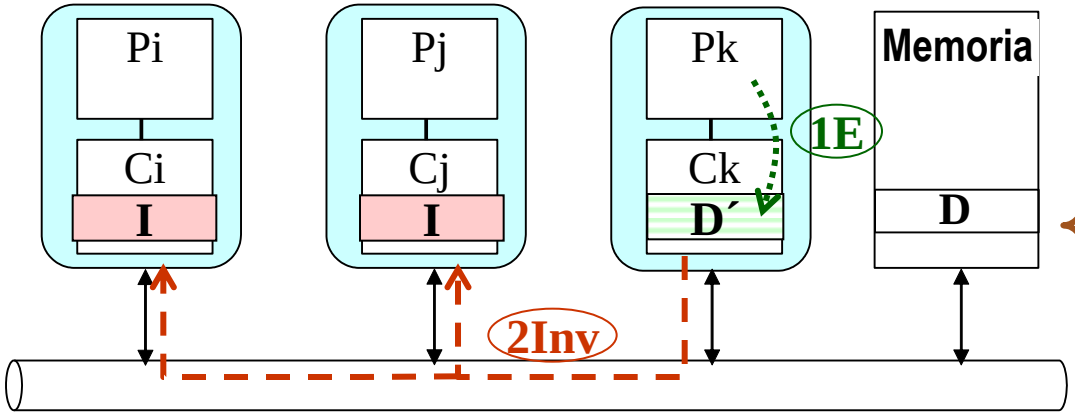
1. Escritura con actualización (*write-update*):



Cada vez que un procesador escribe en una dirección en su cache se escribe en las copias de esa dirección en otras caches

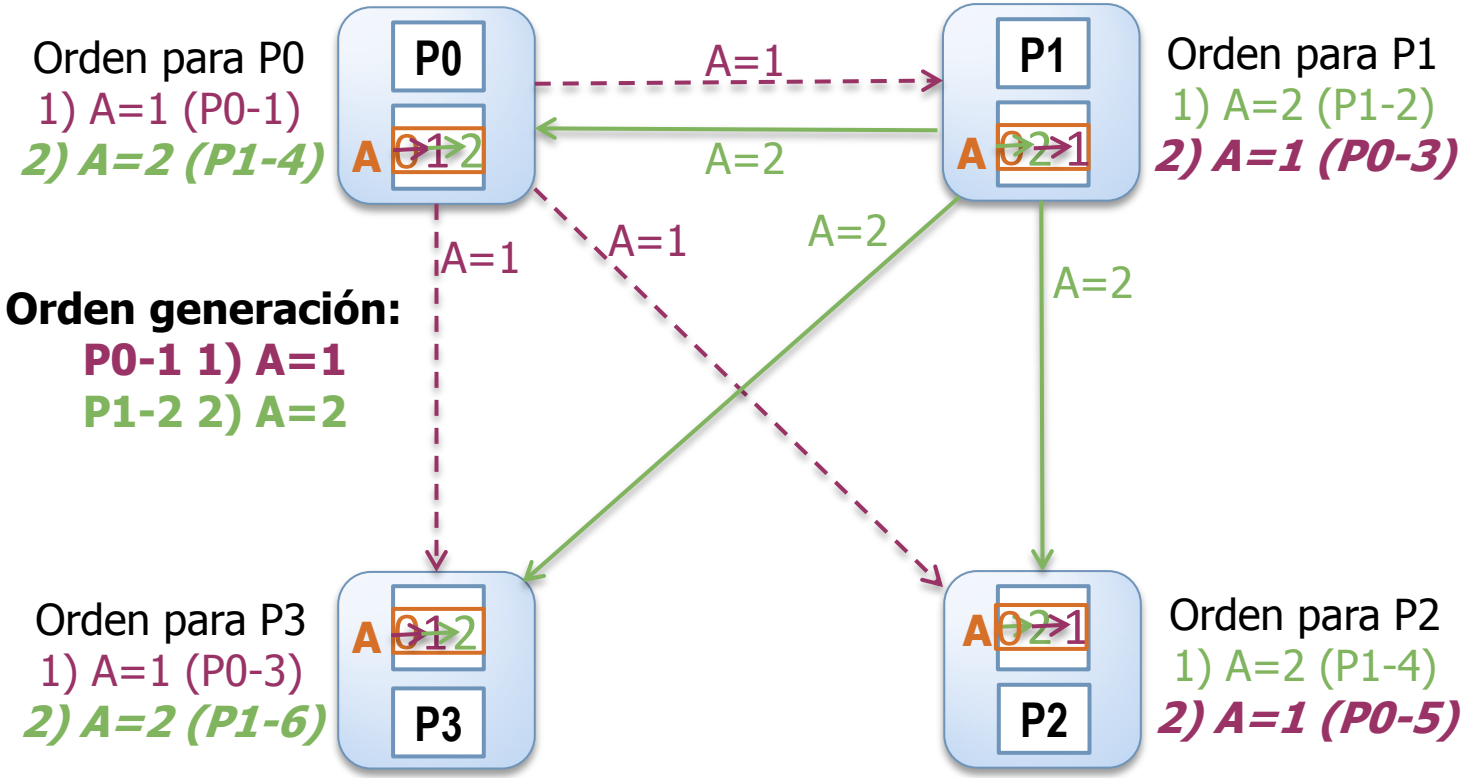
Para reducir tráfico, sobre todo si los datos están compartidos por pocos procesadores

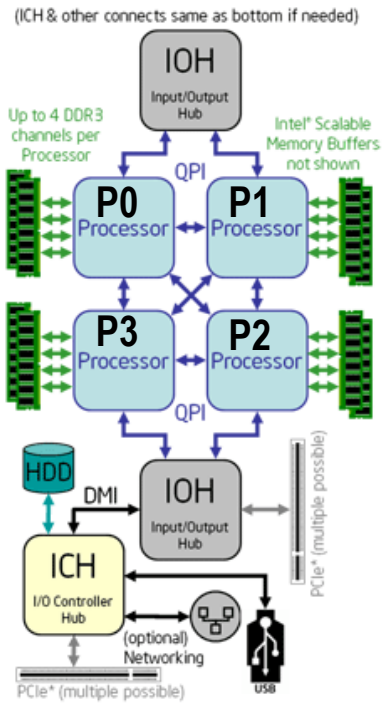
2. Escritura con invalidación (*write-invalidate*):



Antes que un procesador modifique una dirección en su cache se invalidan las copias del bloque de la dirección en otras caches

Situación de incoherencia aunque se propagan las escrituras (**usa difusión**)

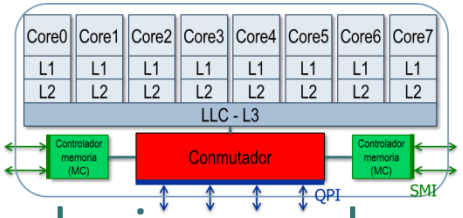




- Contenido inicial de las copias de la dirección A en **calabaza**. P0 escribe en A un 1 y, después, P1 escribe en A un 2.
- Se utiliza **actualización** para propagar las escrituras (las propagación se nota con flechas)
- Llegan en distinto orden las escrituras debido al distinto tiempo de propagación (**se está suponiendo que los Proc. están ubicados en la placa tal y como aparecen en el dibujo**). En ***cursiva*** se puede ver el contenido de las copias de la dirección A tras las dos escrituras.
  - Se da una situación de incoherencia aunque se propagan las escrituras: P0 y P3 acaban con 2 en A y P1 y P2 con 1.

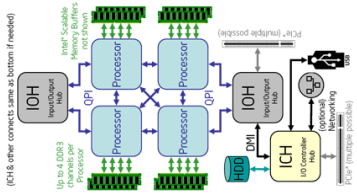
Requisitos del sistema de memoria para evitar problemas por incoherencia I

- Propagar las escrituras en una dirección
  - La escritura en una dirección debe hacerse visible en un tiempo finito a otros procesadores
  - Componentes conectados con un bus:
    - Los paquetes de actualización/invalidación son visibles a todos los nodos conectados al bus (controladores de cache)



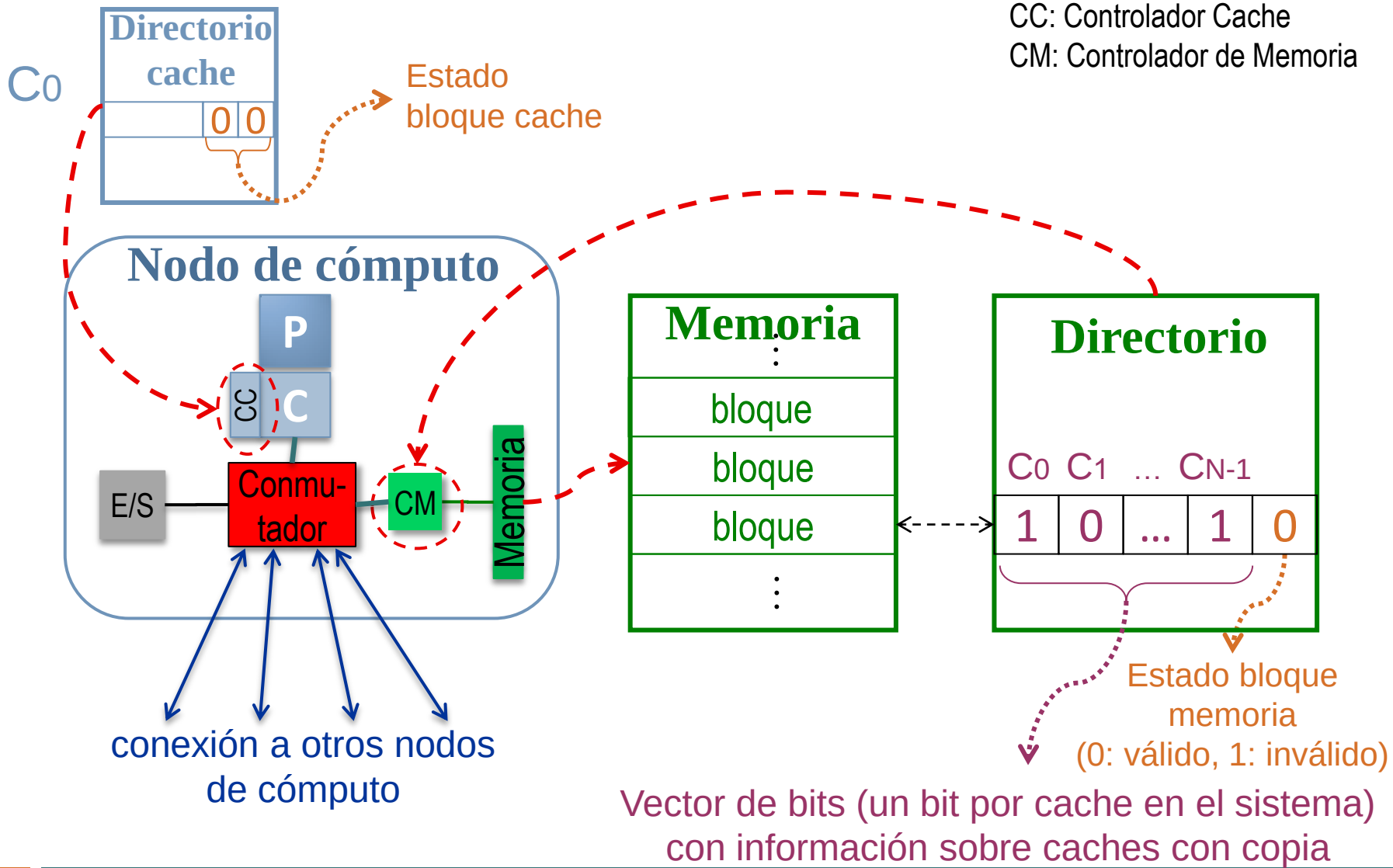
- Serializar las escrituras en una dirección
  - Las escrituras en una dirección deben verse en el mismo orden por todos los procesadores (el sistema de memoria debe **parecer** que realiza en serie las operaciones de escritura en la misma dirección)
  - Componentes conectados con un bus:
    - El orden en que los paquetes aparecen en el bus determina el orden en que se ven por todos los nodos.

Requisitos del SM para evitar problemas por incoherencia II: la red no es un bus



- Propagar escrituras en una dirección
  - Usando difusión:
    - Los paquetes de actualización/invalidación se envían a todas las caches
  - Para conseguir mayor escalabilidad:
    - Se debería enviar paquetes de actualización/invalidación sólo a caches (nodos) con copia del bloque
    - Mantener en un directorio, para cada bloque, los nodos con copia del mismo
- Serializar escrituras en una dirección
  - El orden en el que las peticiones de escritura llegan a su *home* (nodo que tiene en MP la dirección) o al directorio centralizado sirve para serializar en sistemas de comunicación que garantizan el orden en las trasferencias entre dos puntos

Directorio de memoria principal



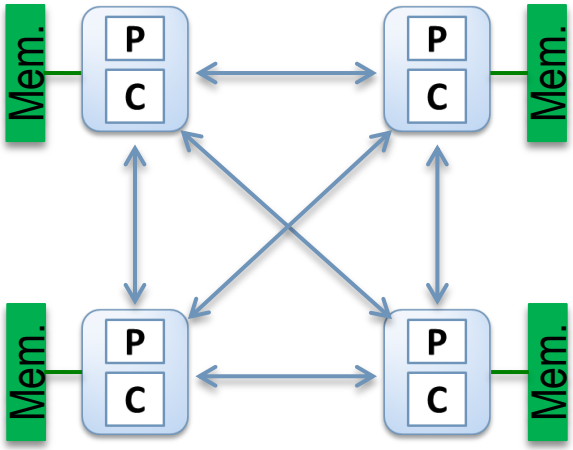
Alternativas para implementar el directorio

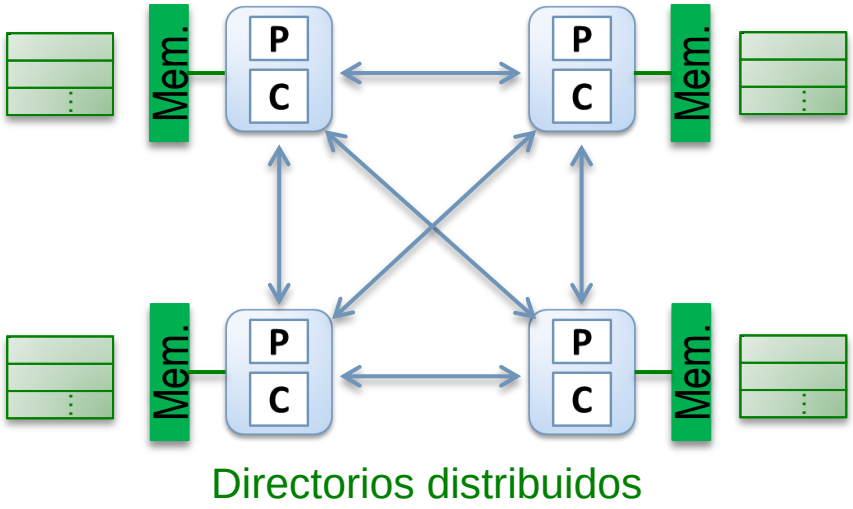
Centralizado

- Compartido por todos los nodos
- Contiene información de los bloques de todos los módulos de memoria

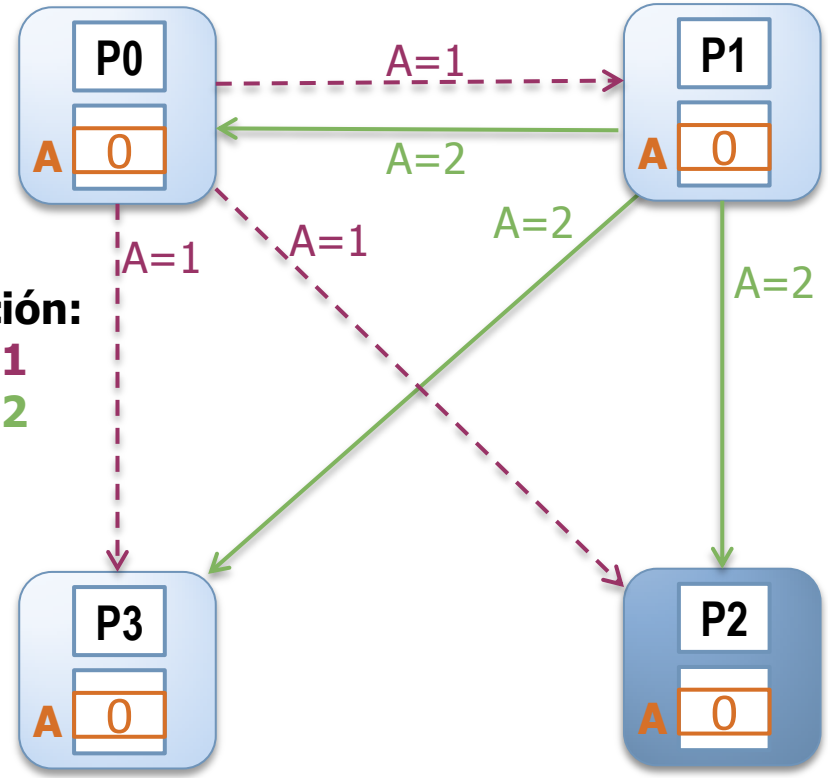
Distribuido

- Las filas se distribuyen entre los nodos
- Típicamente el directorio de un nodo contiene información de los bloques de sus módulos de memoria





Serialización de las escrituras por el home. Usando difusión I



Orden generación:

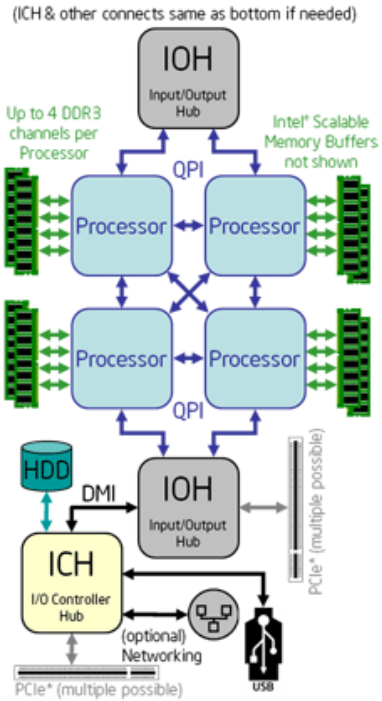
P0-1 1) A=1

P1-2 2) A=2

Orden llegada al home = real

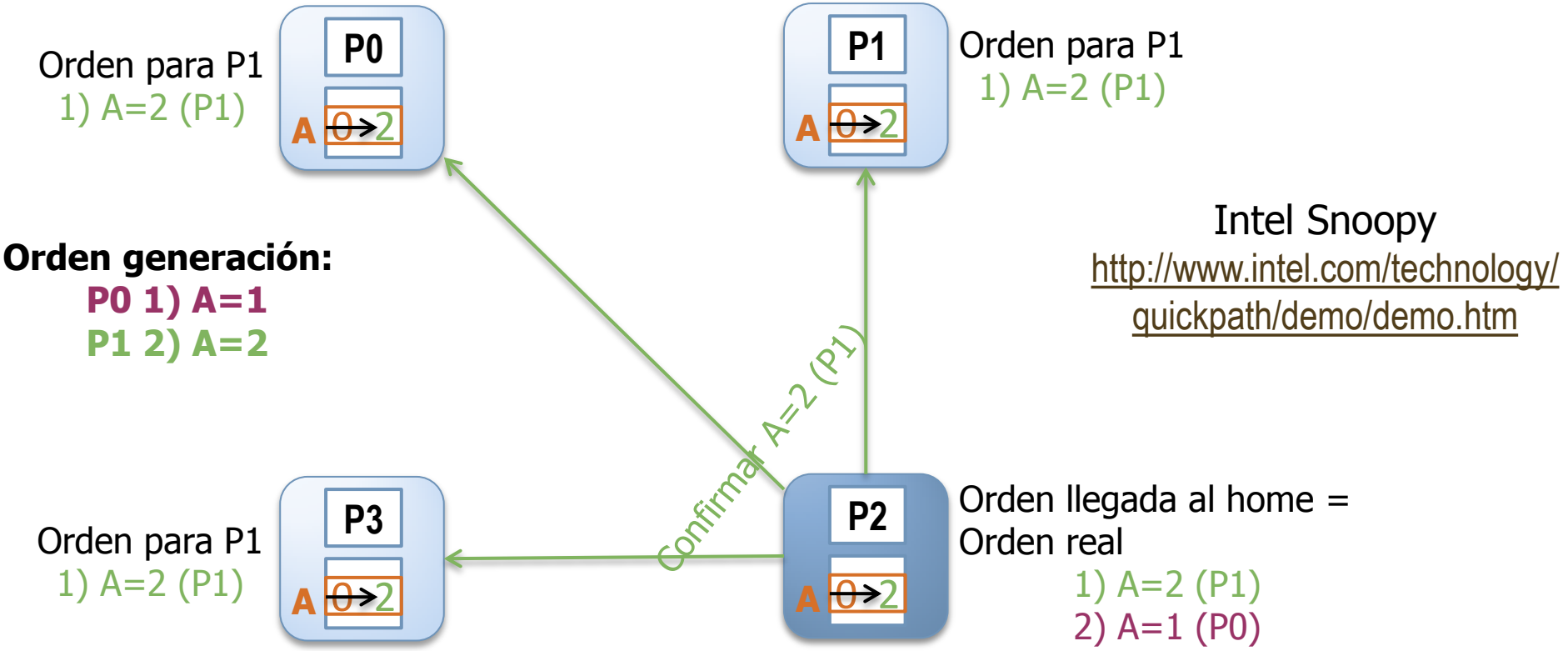
1) A=2 (P1-4)

2) A=1 (P0-5)



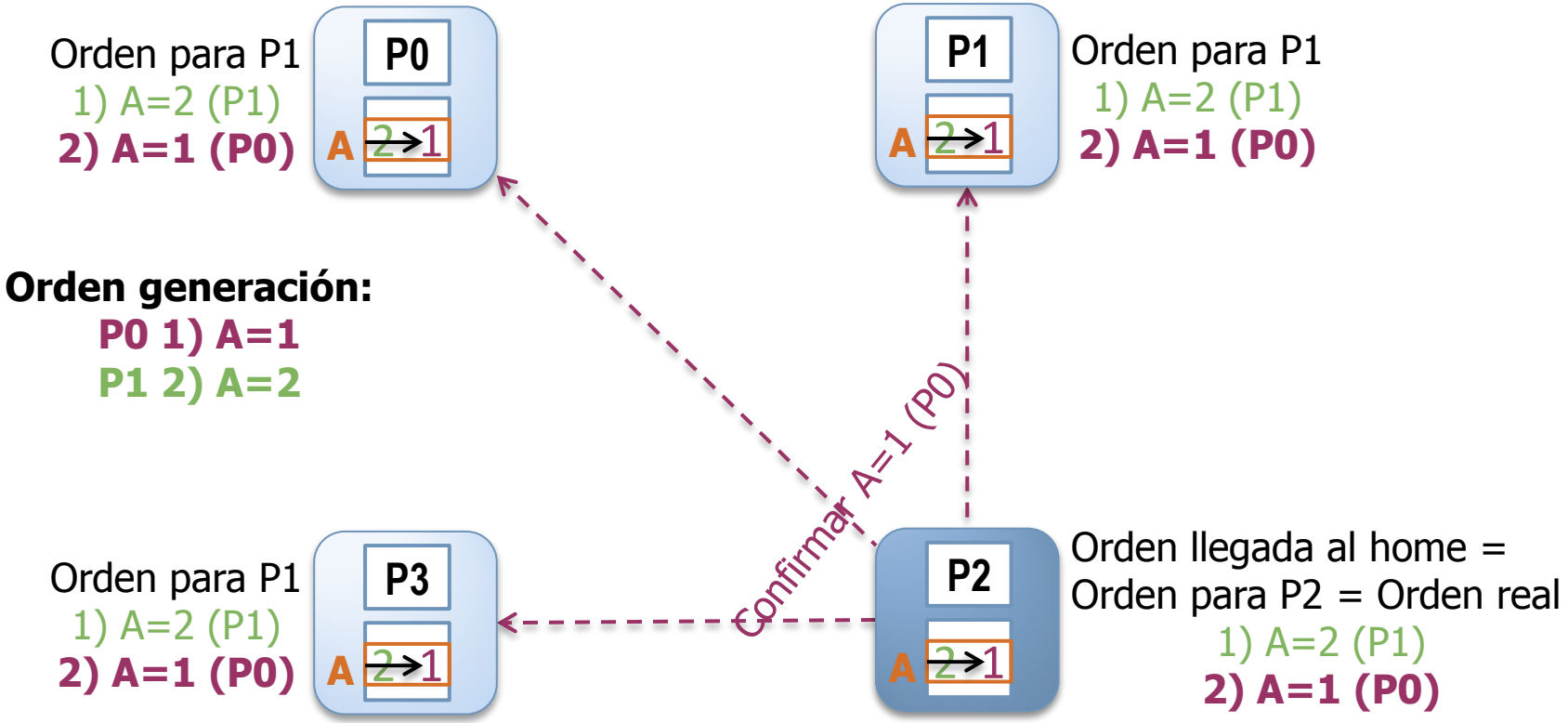
- Contenido inicial de las copias de la dirección A en **calabaza**. P0 escribe en A un 1 y, después, P1 escribe en A un 2.
- Se utiliza **actualización** para propagar las escrituras (las propagación se nota con flechas)
- El orden de llegada al home es el orden real para todos

Serialización de las escrituras por el home. Usando difusión II



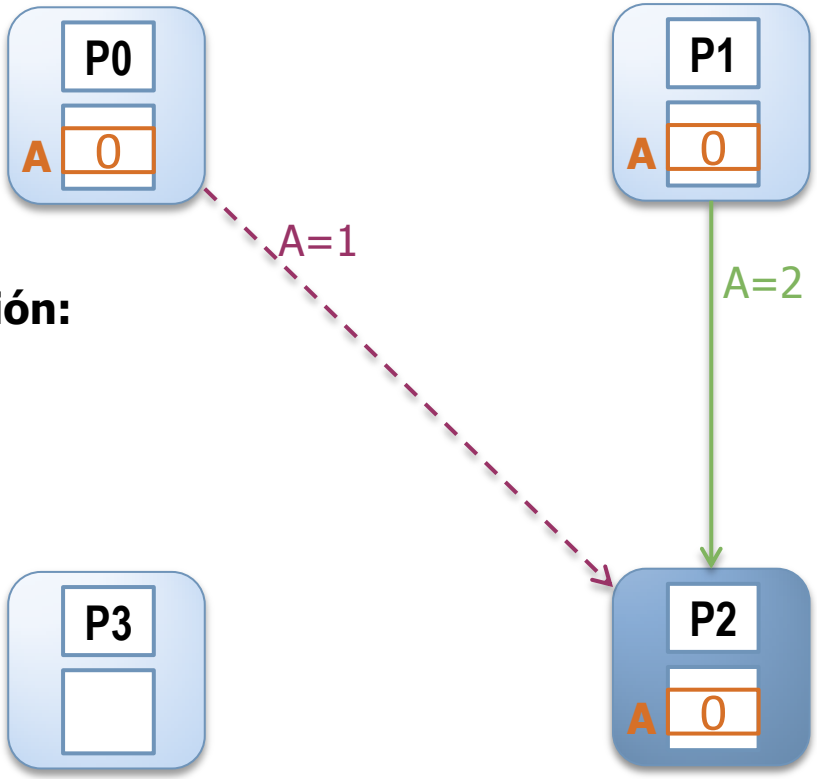
- Contenido inicial de las copias de la dirección A en **calabaza**
- Se utiliza actualización para propagar las escrituras (las propagación se nota con flechas)

Serialización de las escrituras por el home. Usando difusión III



- Se utiliza actualización para propagar las escrituras (las propagación se nota con flechas)

Serialización de las escrituras por el home. Sin difusión y con directorio distribuido I



direct.

Orden generación:

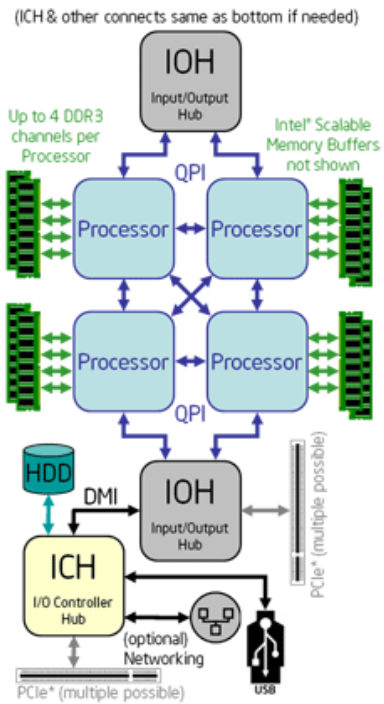
1) A=1

2) A=2

Orden llegada al home = real

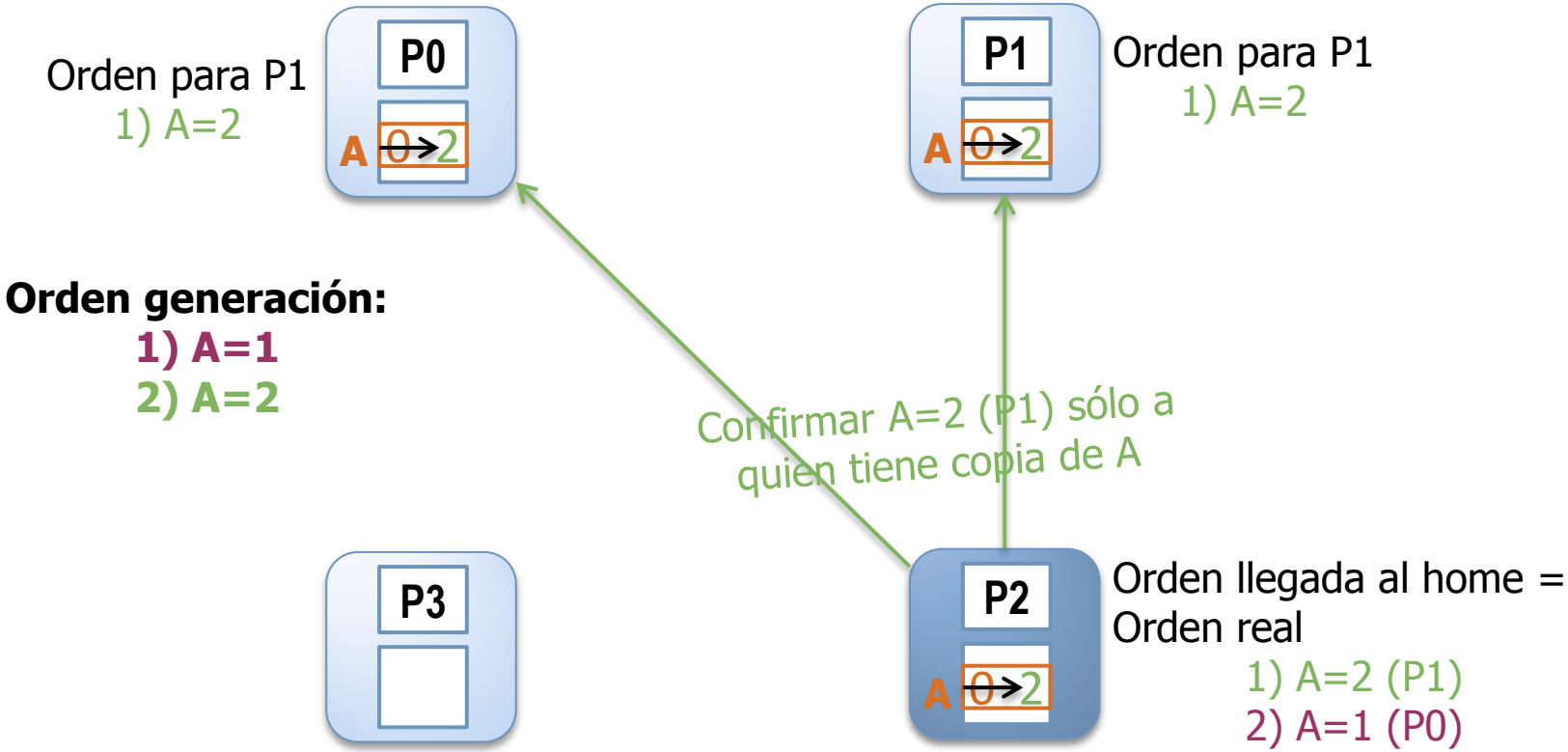
1) A=2 (P1-4)

2) A=1 (P0-5)



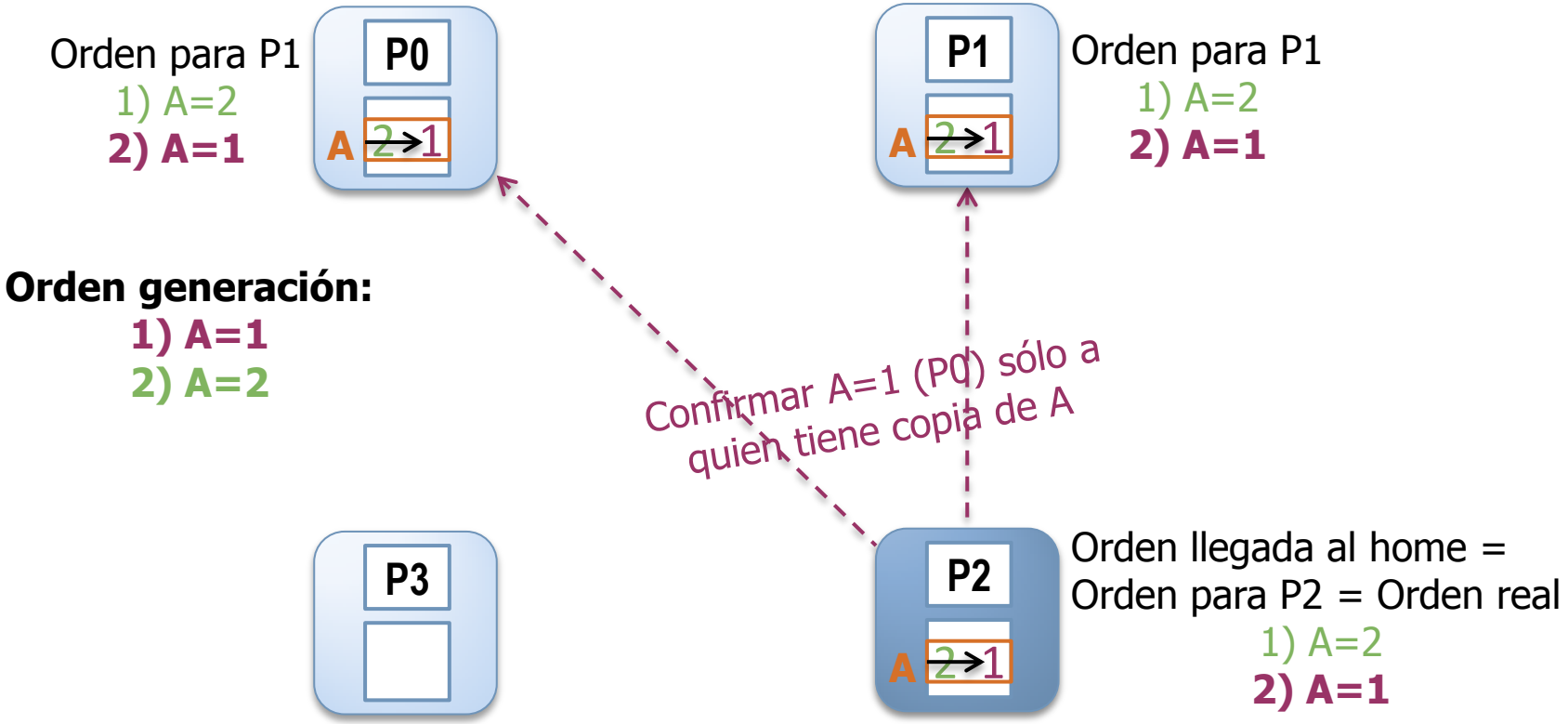
- Contenido inicial de las copias de la dirección A en **calabaza**. P0 escribe en A un 1 y, después, P1 escribe en A un 2.
- Se utiliza actualización para propagar las escrituras (las propagación se nota con flechas)
- El orden de llegada al home es el orden real para todos

Serialización de las escrituras por el home. Sin difusión y con directorio distribuido II



- Contenido inicial de las copias de la dirección A en **calabaza**
- Se utiliza actualización para propagar las escrituras (las propagación se nota con flechas)

Serialización de las escrituras por el home. Sin difusión y con directorio distribuido III



- Se utiliza actualización para propagar las escrituras (las propagación se nota con flechas)

Contenido Lección 8

- Sistema de memoria en multiprocesadores
- Concepto de coherencia en el sistema de memoria: situaciones de incoherencia y requisitos para evitar problemas en estos casos
- Protocolos de mantenimiento de coherencia: clasificación y diseño
- Protocolo MSI de espionaje
- Protocolo MESI de espionaje
- Protocolo MSI basado en directorios con o sin difusión

Clasificación de protocolos para mantener coherencia en el sistema de memoria

- Protocolos de espionaje (snoopy) **bus**
  - Para buses, y en general sistemas con una difusión eficiente (bien porque el número de nodos es pequeño o porque la red implementa difusión).
- Protocolos basados en directorios.
  - Para redes sin difusión o escalables (multietapa y estáticas).
- Esquemas jerárquicos.
  - Para redes jerárquicas: jerarquía de buses, jerarquía de redes escalables, redes escalables-buses.

Facetas de diseño lógico en protocolos para coherencia

- Política de actualización de MP:
  - escritura inmediata, posescritura, mixta
- Política de coherencia entre caches:
  - escritura con invalidación, escritura con actualización, mixta
- Describir comportamiento:
  - Definir posibles estados de los bloques en cache, y en memoria
  - Definir transferencias (indicando nodos que intervienen y orden entre ellas) a generar ante eventos:
    - lecturas/escrituras del procesador del nodo
    - como consecuencia de la llegada de paquetes de otros nodos.
  - Definir transiciones de estados para un bloque en cache, y en memoria

posescr

Ej. tras.

Contenido Lección 8

- Sistema de memoria en multiprocesadores
- Concepto de coherencia en el sistema de memoria: situaciones de incoherencia y requisitos para evitar problemas en estos casos
- Protocolos de mantenimiento de coherencia: clasificación y diseño
- Protocolo MSI de espionaje
- Protocolo MESI de espionaje
- Protocolo MSI basado en directorios con o sin difusión

Protocolo de espionaje de tres estados (MSI) – posescritura e invalidación

- Estados de un bloque en cache:
  - **Modificado** (M): es la única copia del bloque válida en todo el sistema
  - **Compartido** (C,S): está válido, también válido en memoria y puede que haya copia válida en otras caches
  - **Inválido** (I): se ha invalidado o no está físicamente
- Estados de un bloque en memoria (en realidad se evita almacenar esta información):
  - **Válido**: puede haber copia válida en una o varias caches
  - **Inválido**: habrá copia valida en una cache

direct.

posescr

invalid.

Protocolo de espionaje de tres estados (MSI) – posescritura e invalidación

- Transferencias generadas por un nodo con cache (tipos de paquetes):
  - Petición de lectura de un bloque (**PtLec**): por lectura con fallo de cache del procesador del nodo (**PrLec**)
  - Petición de acceso exclusivo (**PtLecEx**): por escritura del procesador (**PrEsc**) en bloque ***compartido*** o ***inválido***
  - Petición de posescritura (**PtPEsc**): por el reemplazo del bloque ***modificado*** (el procesador del nodo no espera respuesta)
  - Respuesta con bloque (**RpBloque**): al tener en estado ***modificado*** el bloque solicitado por una PtLec o PtLecEx recibida

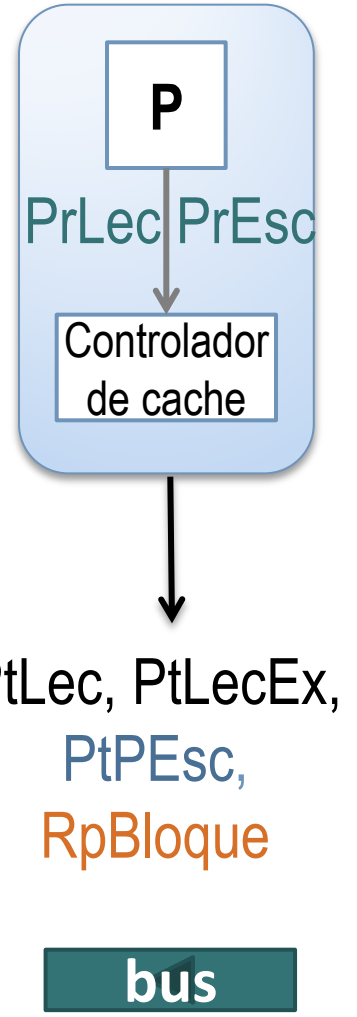
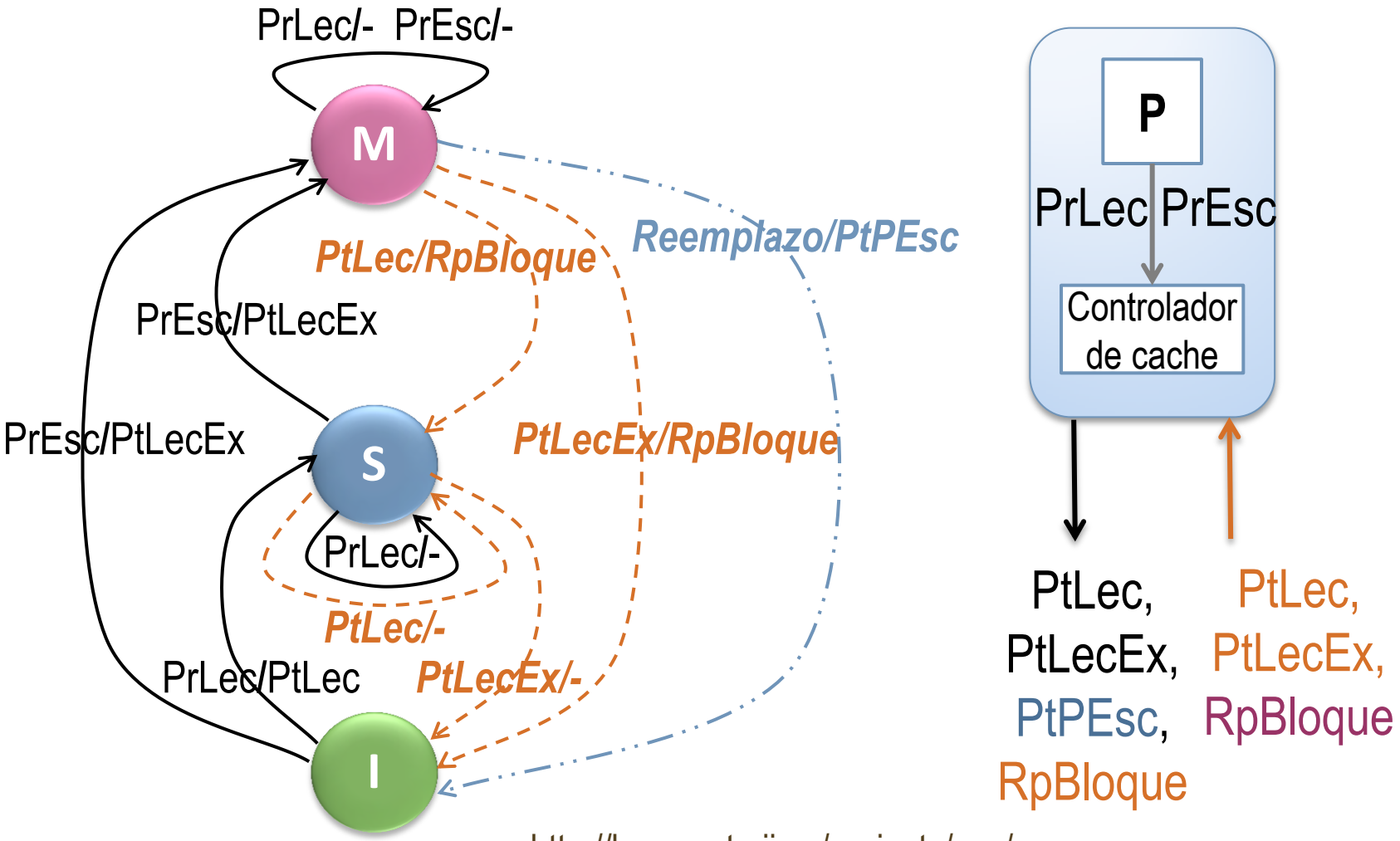


Diagrama MSI de transiciones de estados

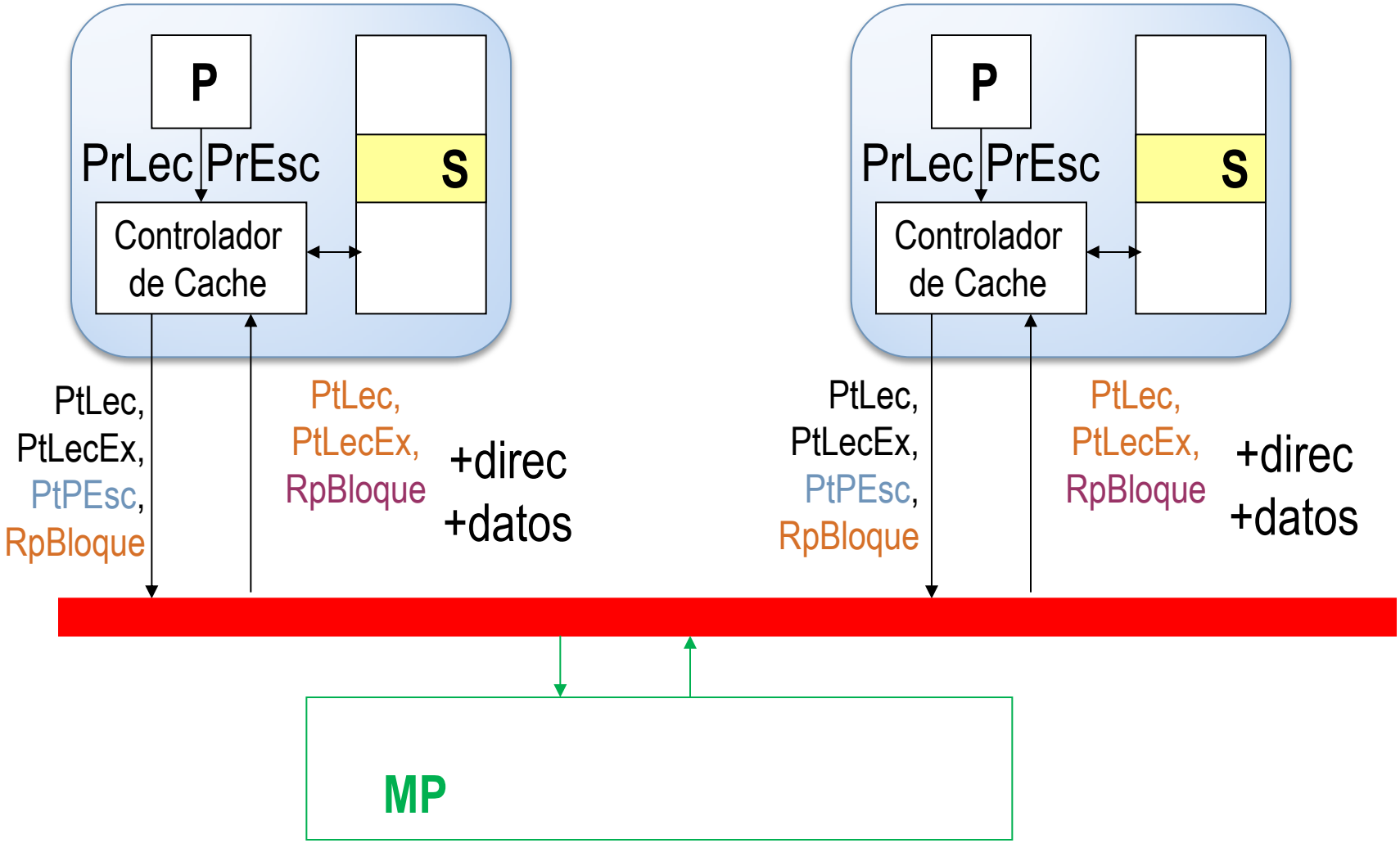


http://lorca.act.uji.es/projects/ccp/

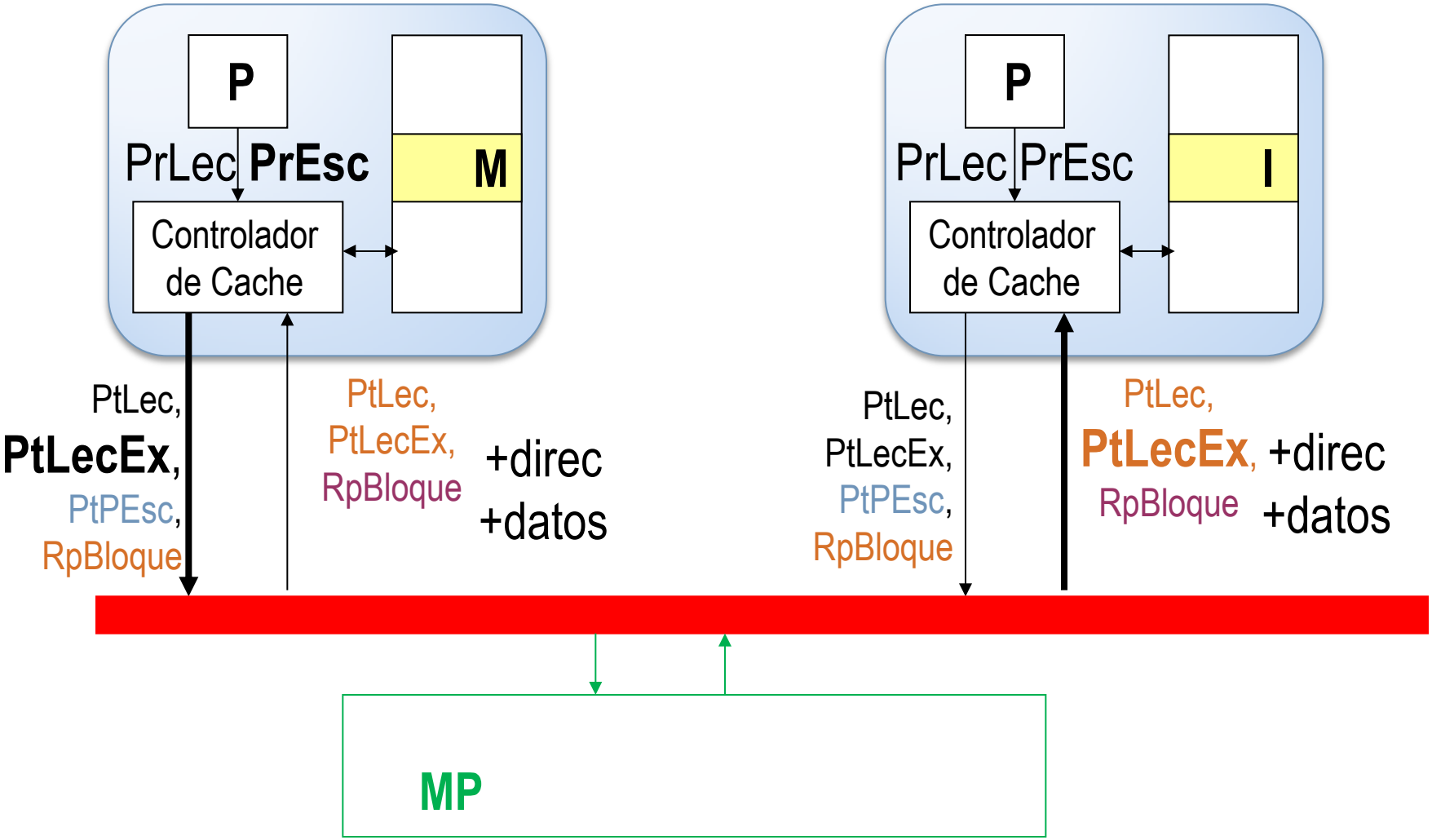
Tabla de descripción de MSI

EST. ACT.	EVENTO	ACCIÓN	SIGUIENTE
Modificado (M)	PrLec/PrEsc		Modificado
	PtLec	Genera paquete respuesta (RpBloque)	Compartido
	PtLecEx	Genera paquete respuesta (RpBloque) Invalida copia local	Inválido
	Reemplazo	*Genera paquete posescritura (PtPEsc)*	*Inválido*
Compart. (S)	PrLec		Compartido
	PrEsc posescr	Genera paquete PtLecEx (PtEx)	Modificado
	PtLec		Compartido
	PtLecEx invalido	Invalida copia local	Inválido
Inválido (I)	PrLec	Genera paquete PtLec	Compartido
	PrEsc	Genera paquete PtLecEx	Modificado
	PtLec/PtLecEx		Inválido

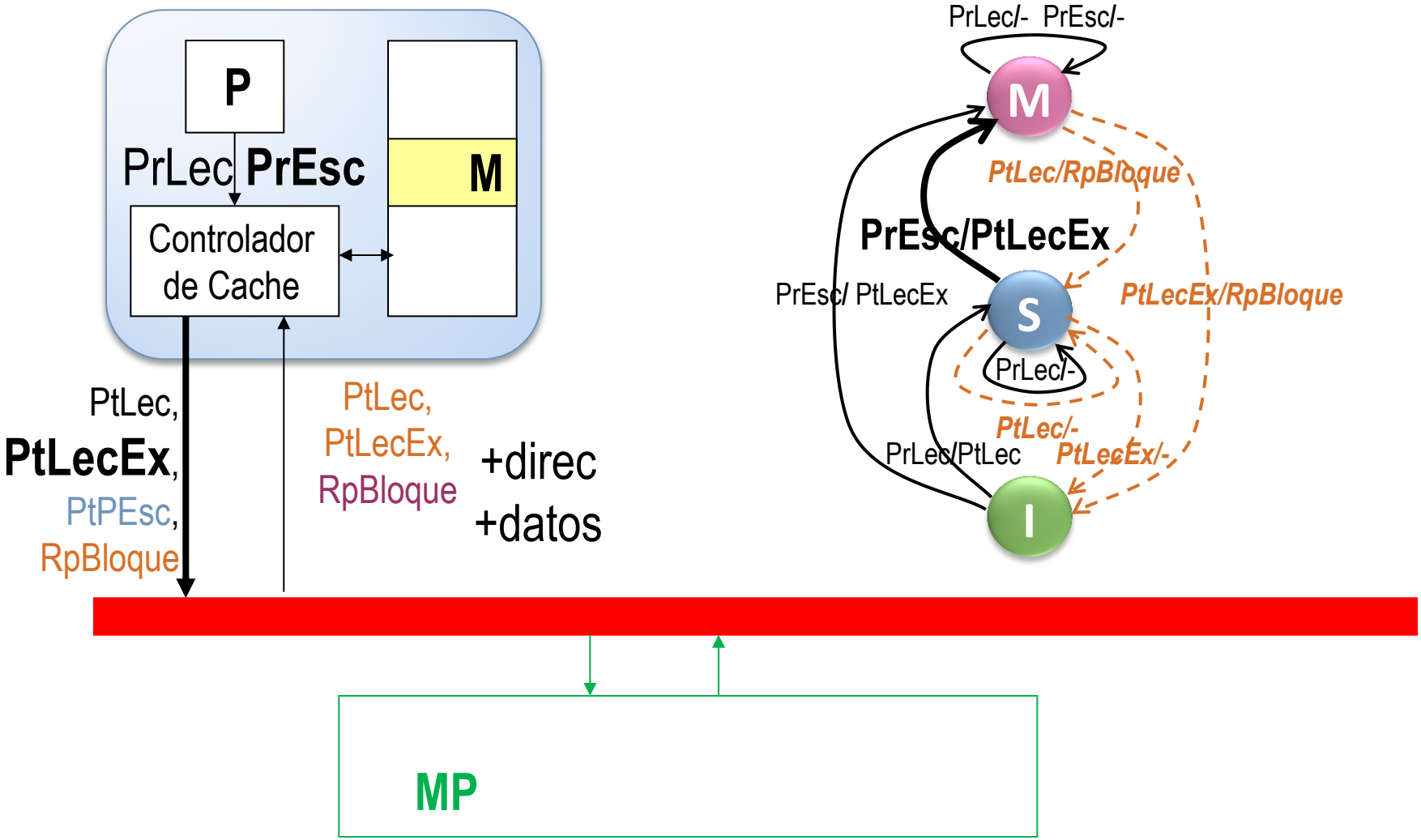
Ejemplo MSI I



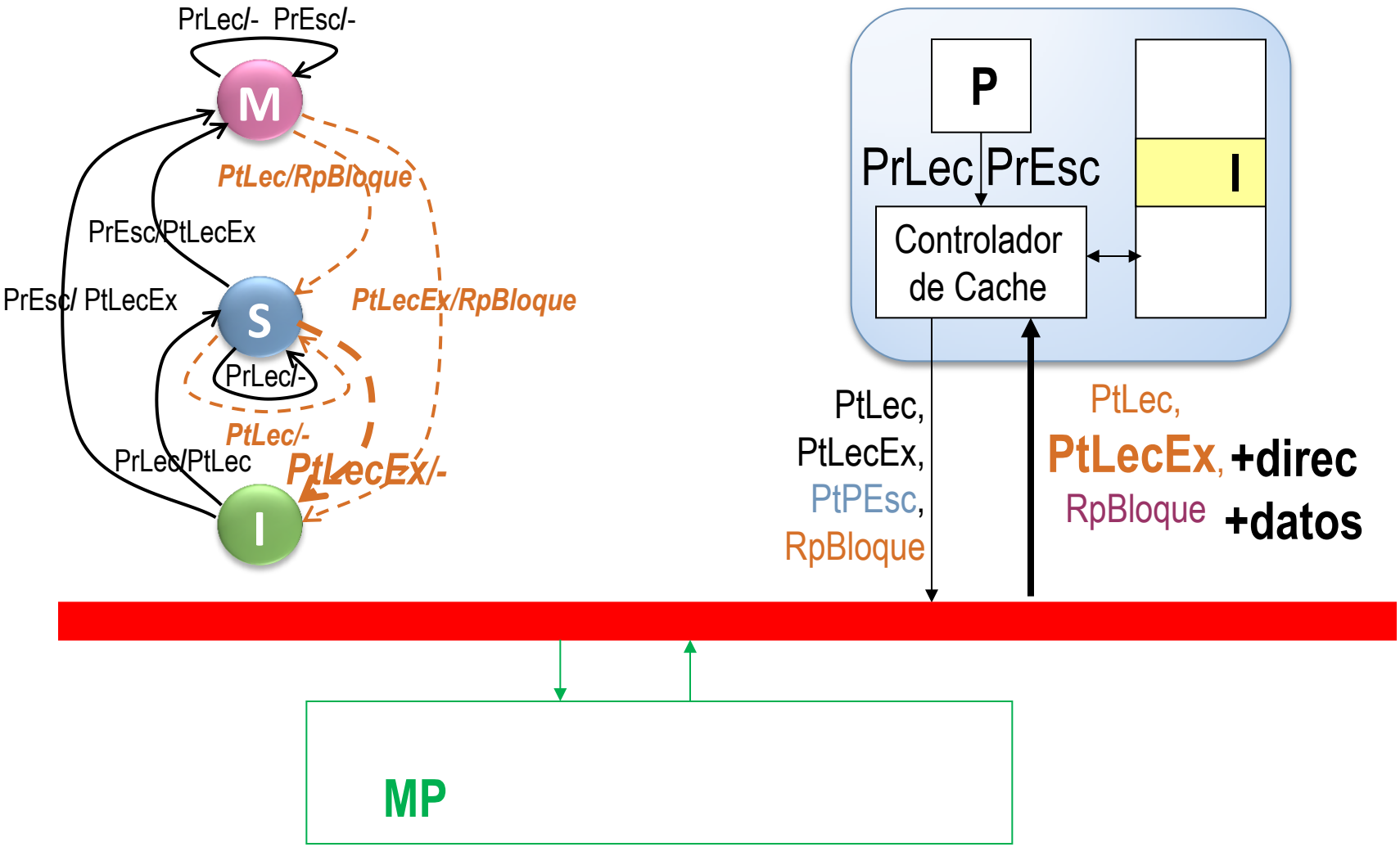
Ejemplo MSI II



Ejemplo MSI III



Ejemplo MSI IV



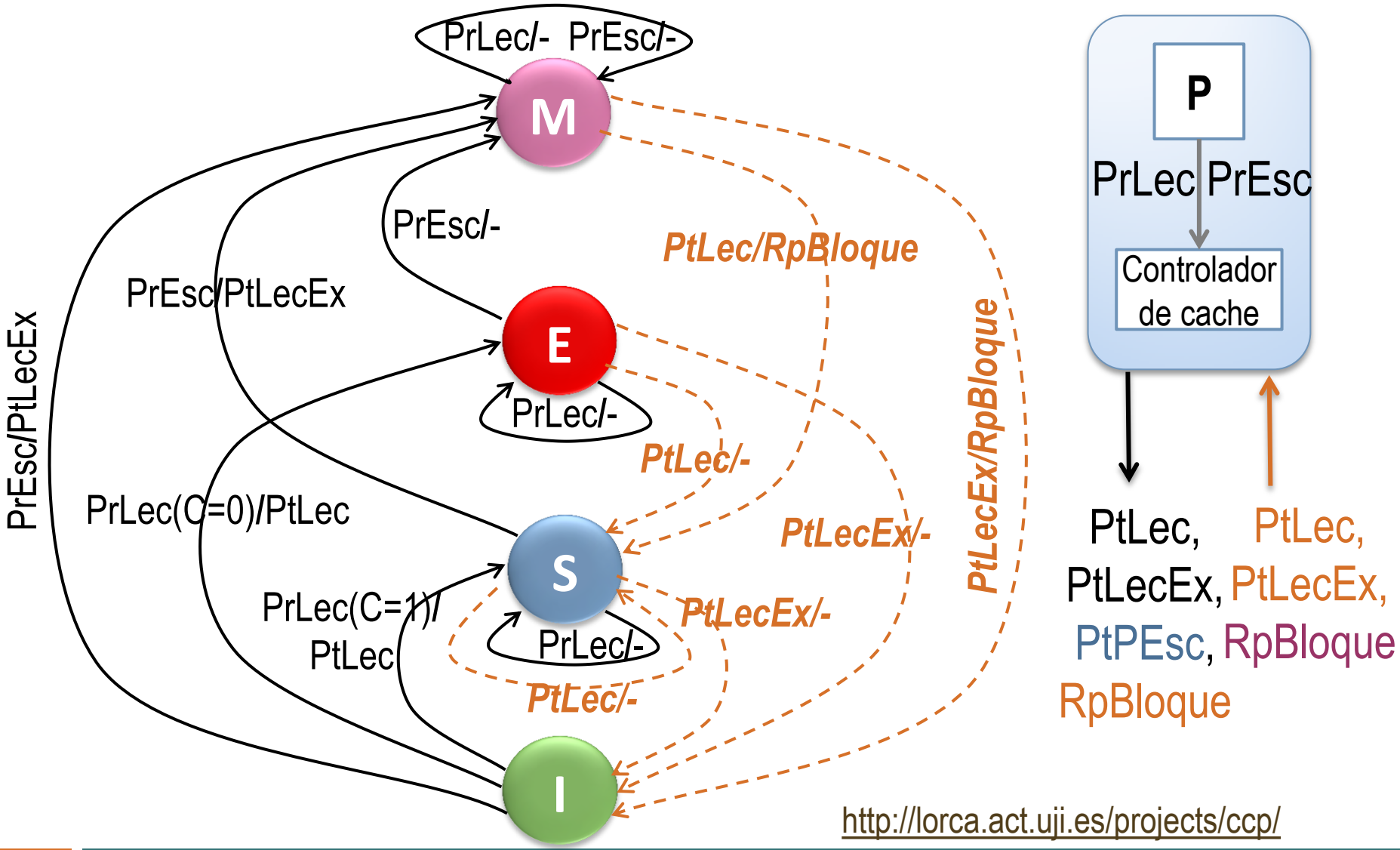
Contenido Lección 8

- Sistema de memoria en multiprocesadores
- Concepto de coherencia en el sistema de memoria: situaciones de incoherencia y requisitos para evitar problemas en estos casos
- Protocolos de mantenimiento de coherencia: clasificación y diseño
- Protocolo MSI de espionaje
- Protocolo MESI de espionaje **MSI**
- Protocolo MSI basado en directorios con o sin difusión

Protocolo de espionaje de cuatro estados (MESI) – posescritura e invalidación

- Estados de un bloque en cache:
  - **Modificado** (M):es la única copia del bloque válida en todo el sistema
  - **Exclusivo** (E): es la *única* copia del bloque válida en caches, la memoria también está actualizada
  - **Compartido** (C,Shared): es válido, también válido en memoria y en *al menos* otra cache
  - **Inválido** (I): se ha invalidado o no está físicamente
- Estados de un bloque en memoria(en realidad se evita almacenar esta información):
  - **Válido**: puede haber copia válida en una o varias caches
  - **Inválido**: habrá copia valida en una cache

Diagrama MESI de transiciones de estados



http://lorca.act.uji.es/projects/ccp/

Tabla de descripción de MESI

MSI

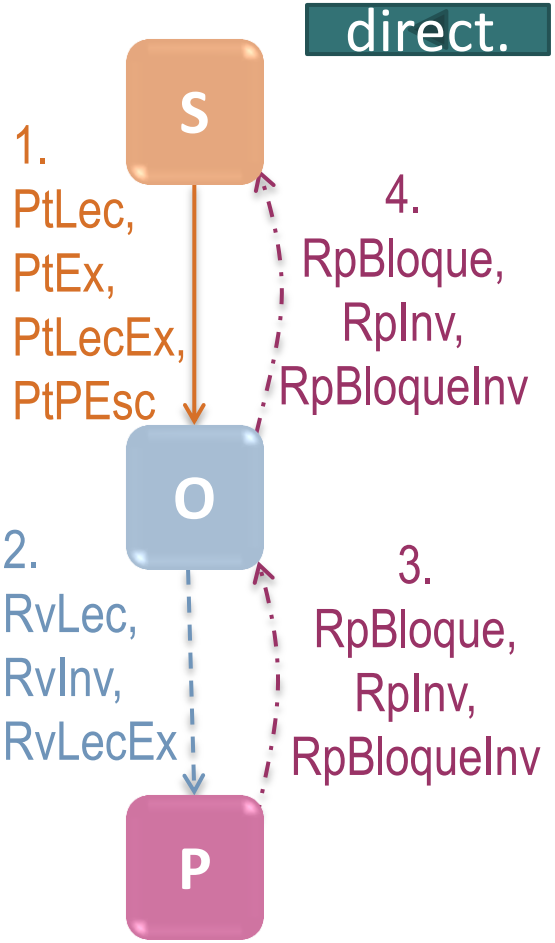
Modificado (M)	PrLec/PrEsc		Modificado
	PtLec	Genera RpBloque	Compartido
	PtLecEx	Genera RpBloque. Invalida copia local	Inválido
	Reemplazo	*Genera PtPEsc*	*Inválido*
Exclusivo (E)	PrLec		Exclusivo
	PrEsc		Modificado
	PtLec		Compartido
	PtLecEx	Invalida copia local	Inválido
Compartido (S)	PrLec/PtLec		Compartido
	PrEsc	Genera PtLecEx	Modificado
	PtLecEx	Invalida copia local	Inválido
Inválido (I)	PrLec (C=1)	Genera PtLec	Compartido
	PrLec (C=0)	Genera PtLec	Exclusivo
	PrEsc	Genera PtLecEx	Modificado
	PtLec/PtLecEx		Inválido

Contenido Lección 8

- Sistema de memoria en multiprocesadores
- Concepto de coherencia en el sistema de memoria: situaciones de incoherencia y requisitos para evitar problemas en estos casos
- Protocolos de mantenimiento de coherencia: clasificación y diseño
- Protocolo MSI de espionaje
- Protocolo MESI de espionaje
- Protocolo MSI basado en directorios con o sin difusión

MSI con directorios (sin difusión) I

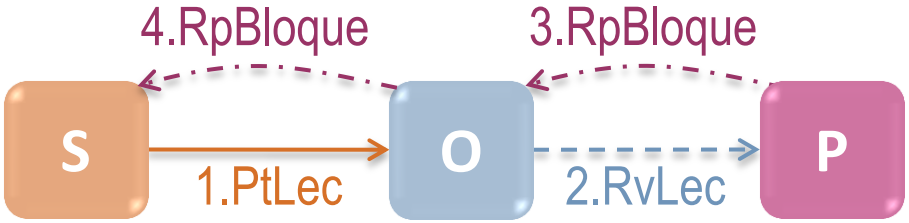
- Estados de un bloque en cache:
  - Modificado (M), Compartido (C), Inválido (I)
- Estados de un bloque en MP:
  - Válido e inválido
- Transferencias (tipos de paquetes) :
  - Tipos de nodos: solicitante (S), origen (O), modificado (M), propietario (P) y compartidor (C)
  - **Petición** de
    - nodo **S** a **O**: lectura de un bloque (**PtLec**), lectura con acceso exclusivo (**PtLecEx**), petición de acceso exclusivo sin lectura (**PtEx**), posescritura (**PtPEsc**)
  - **Reenvío** de petición de
    - nodo **O** a nodos con copia (**P**, **M**, **C**): invalidación (**RvInv**), lectura (**RvLec**, **RvLecEx**).
  - **Respuesta** de
    - nodo **P** a **O**: respuesta con bloque (**RpBloque**), resp. con o sin bloque confirmando inv. (**RpInv**, **RpBloqueInv**)
    - nodo **O** a **S**: resp. con bloque (**RpBloque**), resp. con o sin bloque confirmando fin inv. (**RpInv**, **RpBloqueInv**)

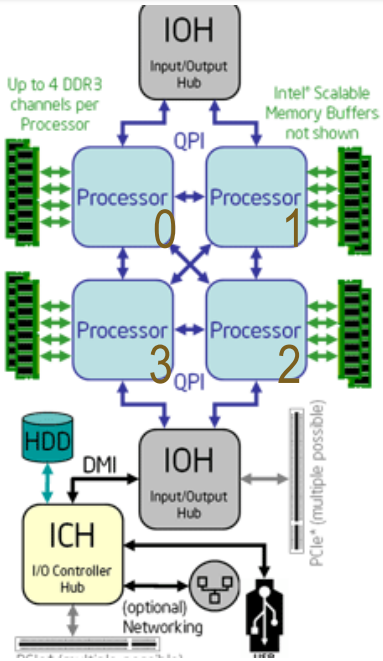


http://www.intel.com/content/www/us/en/performance/performance-quickpath-architecture-demo.html

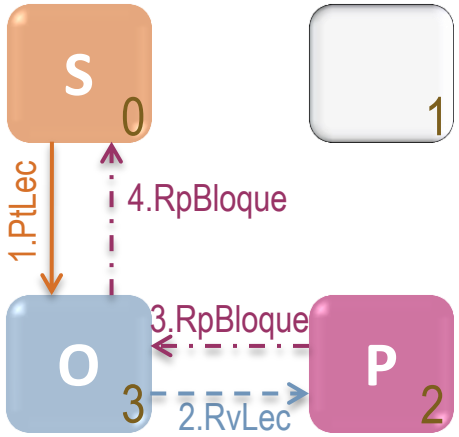
MSI con directorios (sin difusión) II

Estado inicial	Evento	Estado final
D) Inválido S) Inválido P) Modificado Acceso remoto	Fallo de lectura	D) Válido S) Compartido P) Compartido



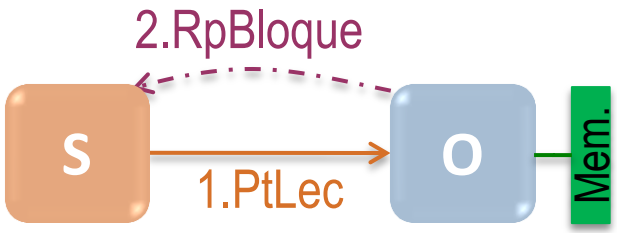


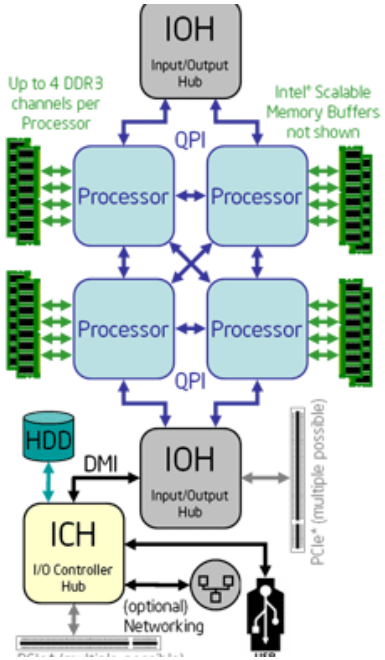
Ejemplo con 4 nodos:

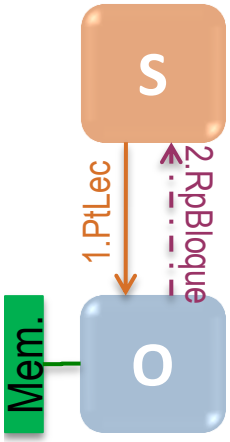


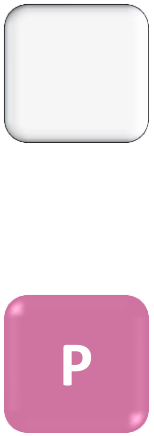
MSI con directorios (sin difusión) III

Estado inicial	Evento	Estado final
D) Válido S) Inválido P) Compartido Acceso remoto	Fallo de lectura	D) Válido S) Compartido P) Compartido



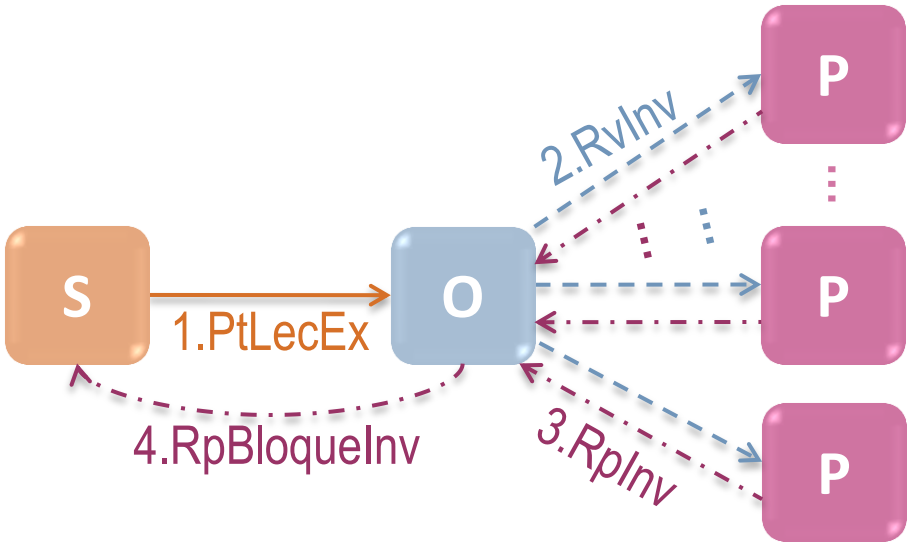


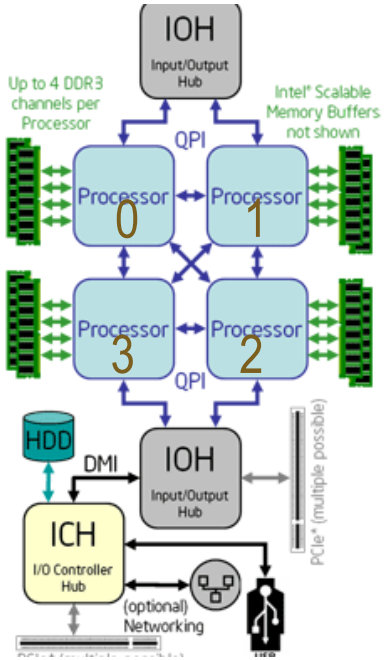


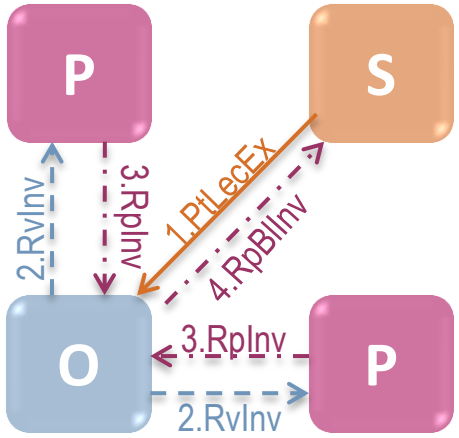


MSI con directorios (sin difusión) IV

Estado inicial	Evento	Estado final
D) Válido S) Inválido P) Compartido Acceso remoto	Fallo de escritura	D) Inválido S) Modificado P) Inválido

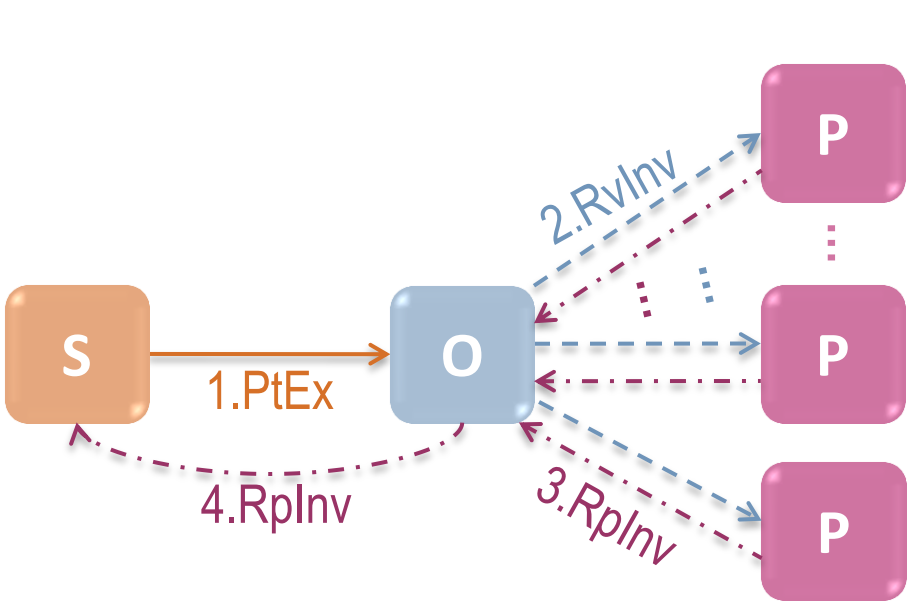


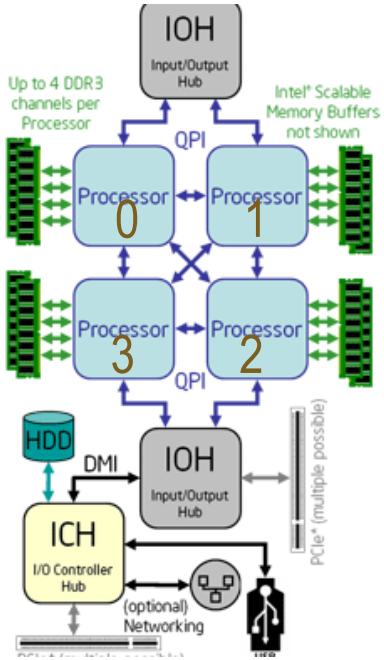


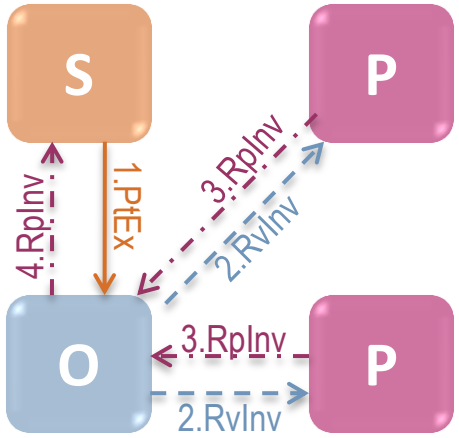


MSI con directorios (sin difusión) V

Estado inicial	Evento	Estado final
D) Válido S) Compartido P) Compartido Acceso remoto	Fallo de escritura	D) Inválido S) Modificado P) Inválido

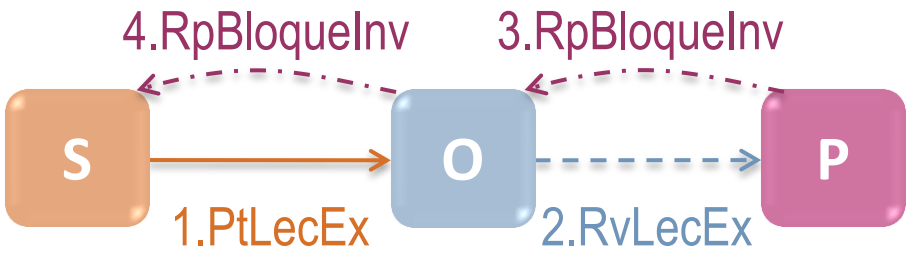


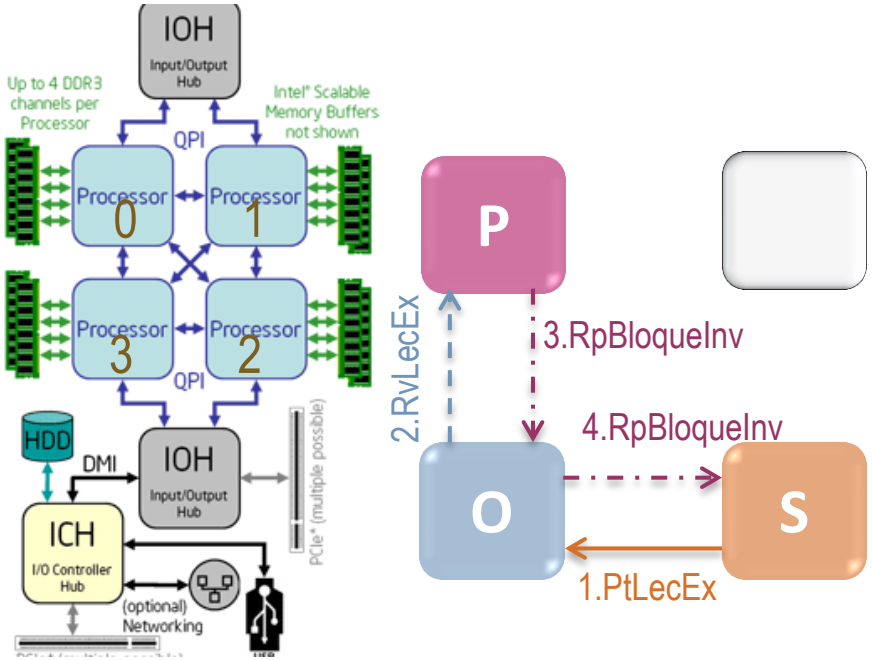




MSI con directorios (sin difusión) VI

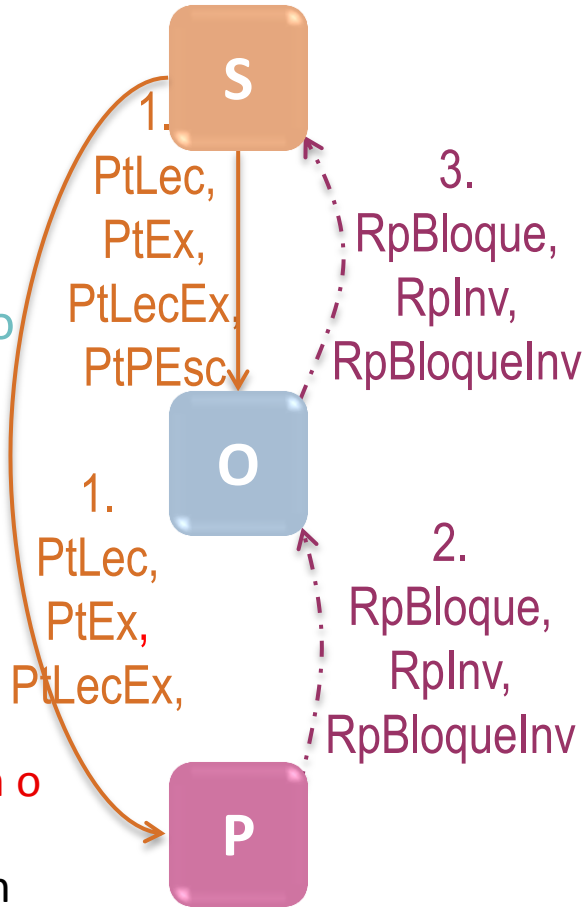
Estado inicial	Evento	Estado final
D) Inválido S) Inválido P) Modificado Acceso remoto	Fallo de escritura	D) Inválido S) Modificado P) Inválido





MSI con directorios (con difusión) I

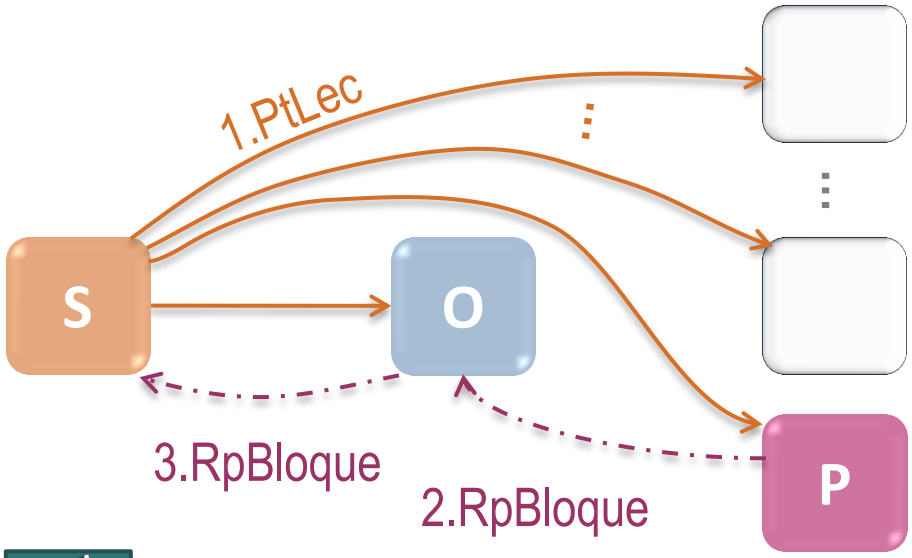
- Estados de un bloque en cache:
  - Modificado (M), Compartido (C), Inválido (I)
- Estados de un bloque en MP:
  - Válido e inválido
- Transferencias (tipos de paquetes) :
  - Tipos de nodos: solicitante (S), origen (O), modificado (M), propietario (P) y compartidor (C)
  - Difusión de **petición** del nodo **S** a
    - **O** y **P**: lectura de un bloque (**PtLec**), lectura con acceso exclusivo (**PtLecEx**), petición de acceso exclusivo sin lectura (**PtEx**)
    - **O**: posescritura (**PtPEsc**)
  - **Respuesta** de
    - nodo **P** a **O**: respuesta con bloque (**RpBloque**), resp. con o sin bloque confirmando inv. (**RpInv**, **RpBloqueInv**)
    - nodo **O** a **S**: resp. con bloque (**RpBloque**), resp. con o sin bloque confirmando fin inv. (**RpInv**, **RpBloqueInv**)



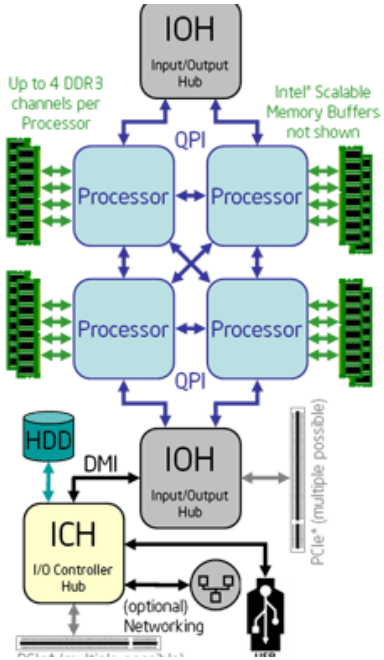
s.d.

MSI con directorios (con difusión) II

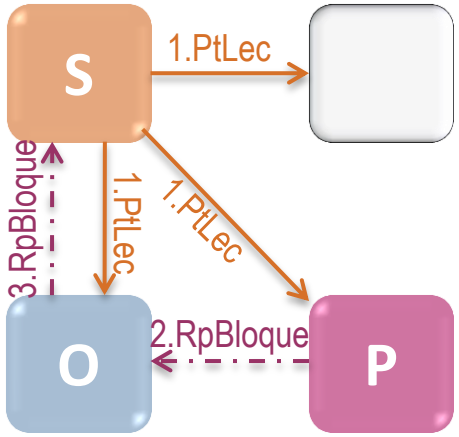
Estado inicial	Evento	Estado final
D) Inválido S) Inválido P) Modificado Acceso remoto	Fallo de lectura	D) Válido S) Compartido P) Compartido



s.d.

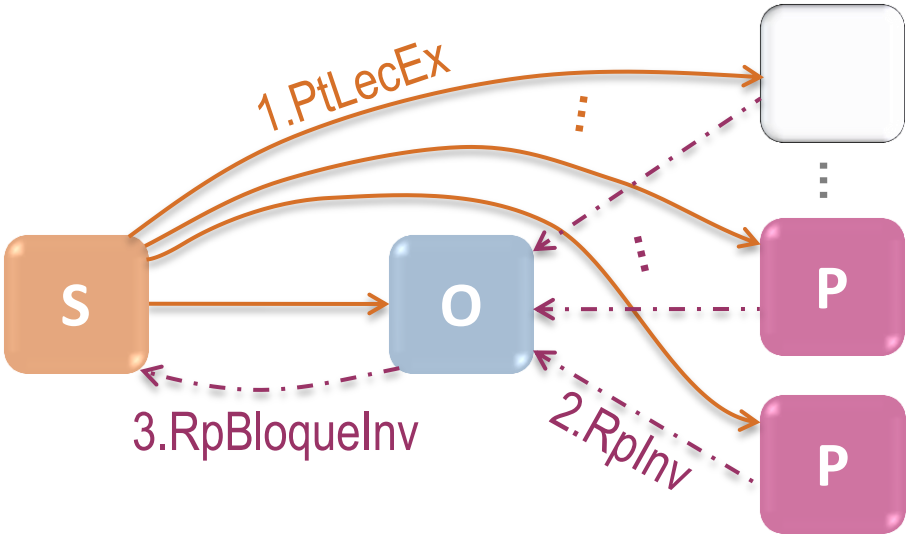


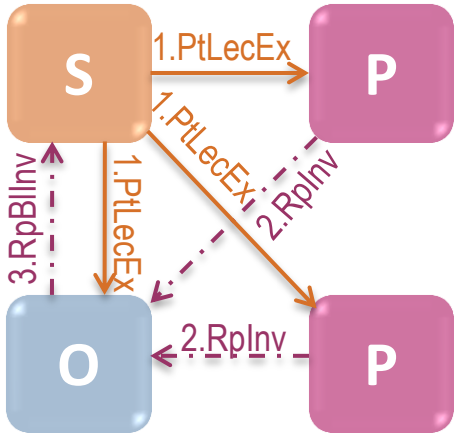
Ejemplo con 4 nodos:



MSI con directorios (con difusión) III

Estado inicial	Evento	Estado final
D) Válido S) Inválido P) Compartido Acceso remoto	Fallo de escritura	D) Inválido S) Modificado P) Inválido





s.d.

Para ampliar …

- Webs
  - An Introduction to the Intel® QuickPath Interconnect, http://www.intel.com/content/www/us/en/io/quickpath-technology/quick-path-interconnect-introduction-paper.html
  - Demo Intel® QuickPath Interconnect http://www.intel.com/content/www/us/en/performance/performance-quickpath-architecture-demo.html
  - Animaciones de protocolos de coherencia de cachés http://lorca.act.uji.es/projects/ccp/