Arquitectura de Computadoras

CURSO 2022

Turno:

Clase 9

Resumen clase 9

Procesamiento paralelo

- Procesamiento paralelo
- Taxonomía de Flynn
- Mutiprocesadores simétricos SMP
- Multiprocesadores NUMA
- Cluster
- Tipos de acceso a memoria en sistemas multiprocesadores
- Arquitecturas on-chip
- Procesamiento multihebra (Multithreading)

Procesamiento paralelo

Introducción

- La demanda de máquinas de mayor rendimiento es una exigencia que surgió desde la aparición de las primeras computadoras, y continúa en forma permanente.
- Existen 2 caminos para aumentar la capacidad de procesamiento:

1. Mejorar el rendimiento de una máquina con un solo procesador.
2. Disponer de sistemas con varios procesadores.

Procesamiento paralelo

Introducción

1. Mejorar el rendimiento de una máquina con un solo procesador requiere:
   - Explotar el paralelismo a nivel instrucción (ILP).
   - Optimizar la detección del paralelismo, a nivel de hardware (MPL).
2. Mejorar la capacidad de procesamiento con varios procesadores, requiere:
   - Explotar el paralelismo a nivel proceso.
   - Detección del paralelismo a nivel de sistema operativo, compilador, o programación.

Taxonomía de Computadoras

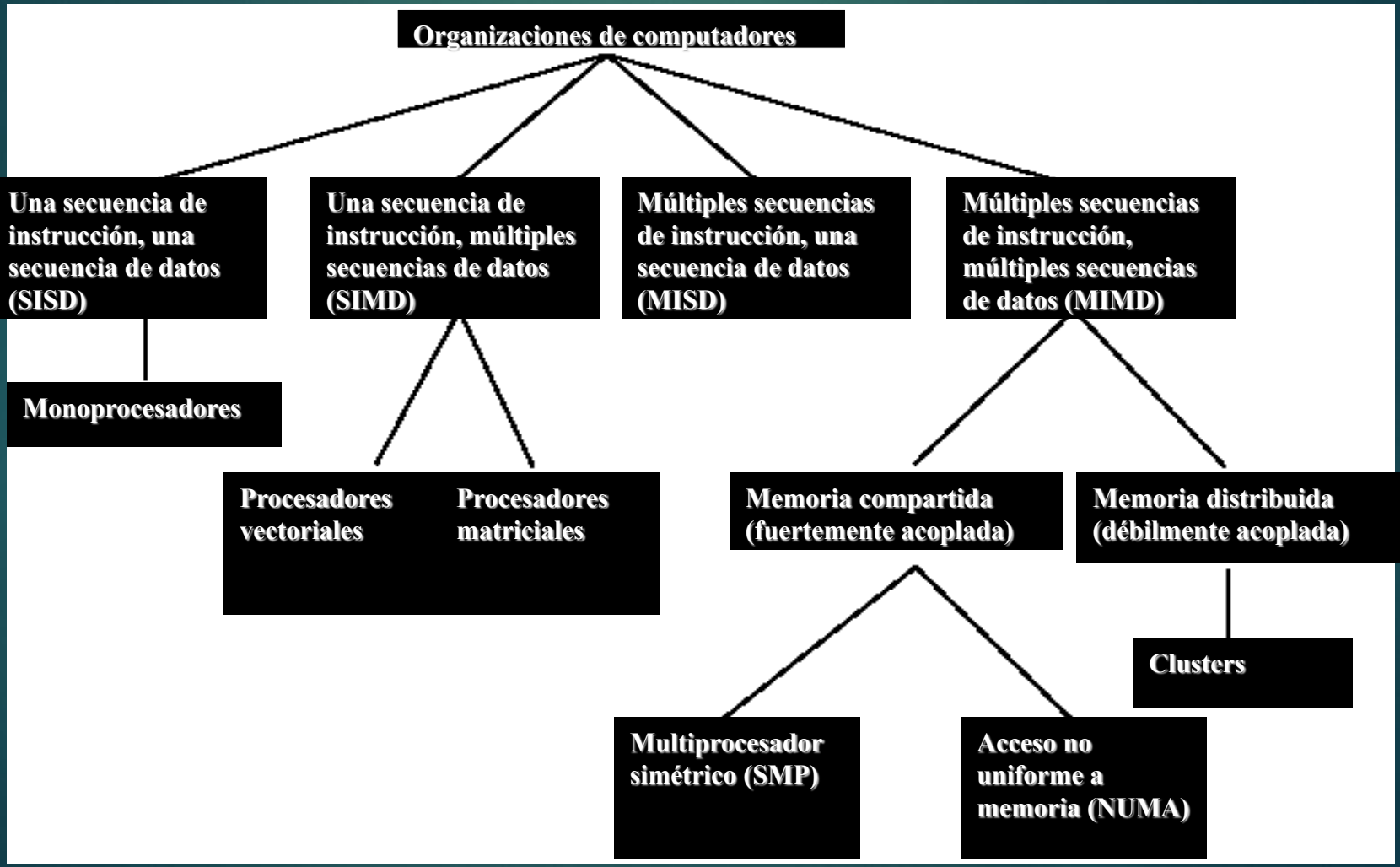
Taxonomía de Flynn

Existen varias formas distintas de clasificar los Sistemas de cómputo. Una de ellas se basa en determinar las cantidades de flujos de instrucciones y datos que se pueden transferir simultáneamente. En base a este concepto se tienen 4 tipos básicos de máquinas:

- SISD: una secuencia de instrucciones, una secuencia de datos.
- SIMD: una secuencia de instrucciones, múltiples secuencias de datos.
- MISD: múltiples secuencias de instrucciones, una secuencia de datos.
- MIMD: múltiples secuencias de instrucciones y múltiples secuencias de datos.

Taxonomía de Flynn

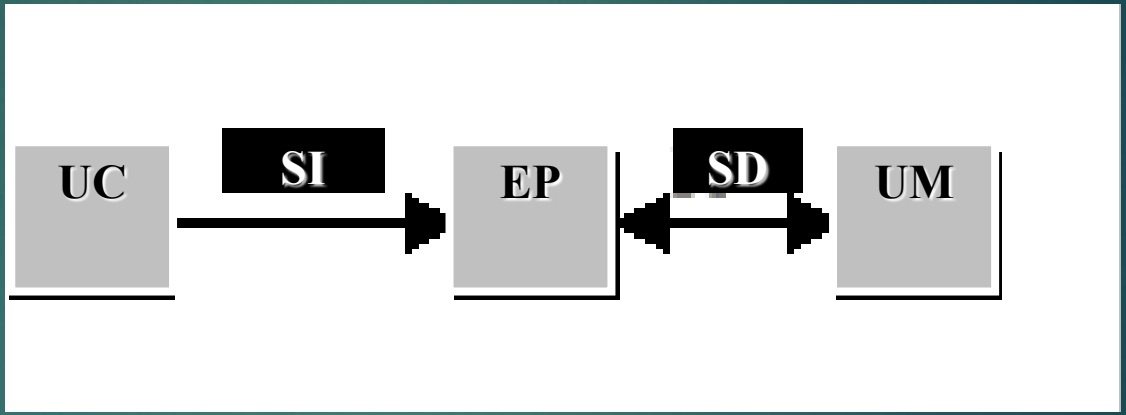
La imagen siguiente muestra los diferentes categorías de Computadoras de acuerdo a la taxonomía de Flynn.



Taxonomía de Flynn

Máquinas SISD

Una máquina tipo SISD (Single Instruction – Single Data), según Flynn, tiene la siguiente estructura funcional.



donde:

- UC: Unidad de control (captura de instrucciones)
- EP: Elemento de Proceso (ejecuta)
- UM: Unidad de Memoria (almacenamiento de datos)
- SI: secuencia de instrucciones
- SD: secuencia de datos

Taxonomía de Flynn

Máquinas SISD

Es un sistema básicamente compuesto por:

- una Unidad de Control (UC)
- un elemento de proceso (EP)
- una Unidad de Memoria (UM).

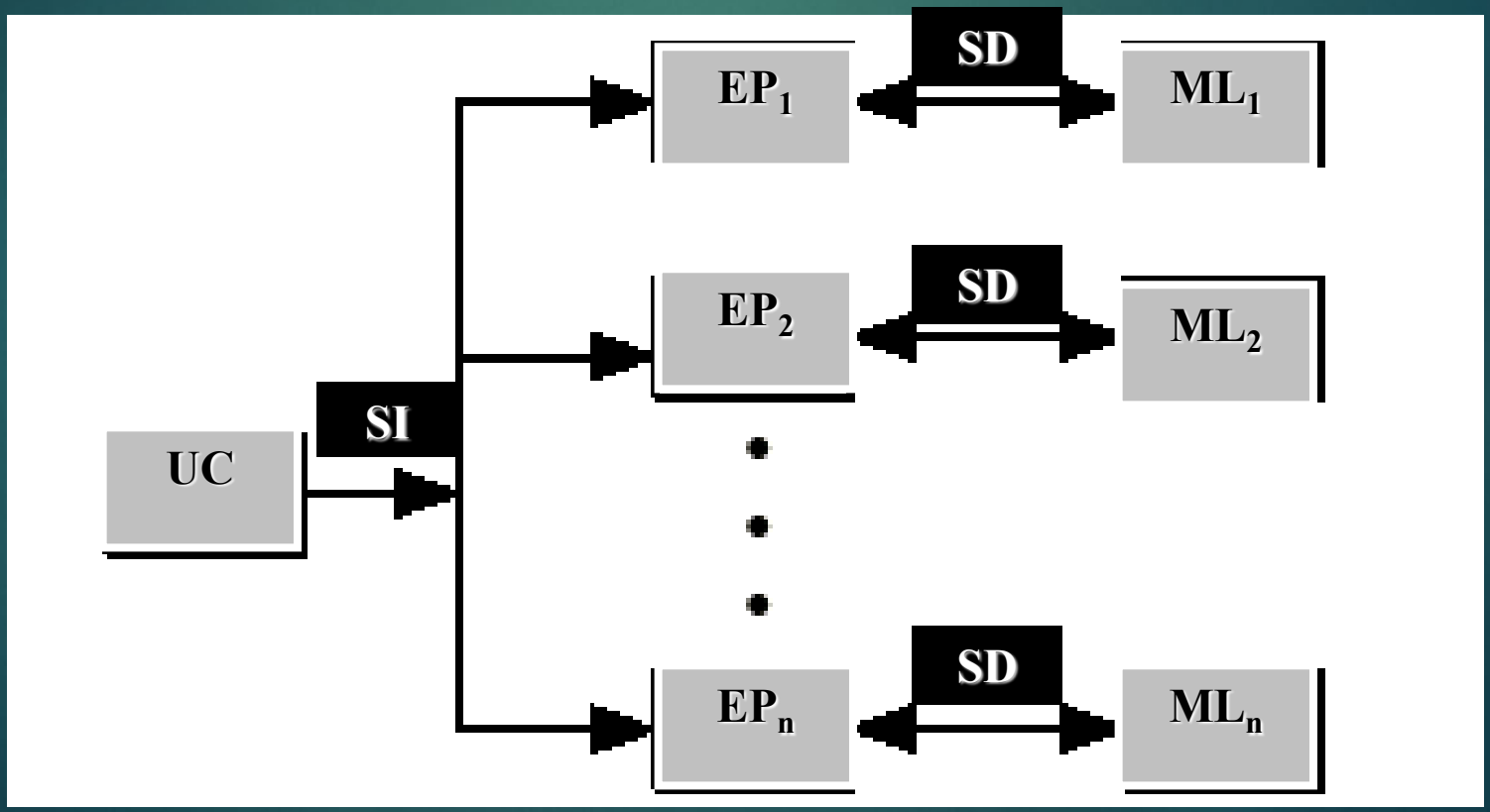
La UC interpreta (lee) una única secuencia de instrucciones (SI), que el EP resuelve sobre una única secuencia de datos (SD) proveniente de la UM.

- Ejemplo máquina SISD: Computadoras monoprocesador.

Taxonomía de Flynn

Máquinas SIMD

La imagen siguiente muestra el esquema de una máquina tipo SIMD (Single Instruction – Multiple Data).



Taxonomía de Flynn

SIMD

Es un sistema compuesto por:

- una Unidad de Control (UC).
- una matriz de elementos computacionales o elementos de proceso (EP).
- Una Unidad de memoria local (ML) por cada EP.

La UC interpreta una única secuencia de instrucciones (SI) que múltiples EP resuelven simultáneamente cada uno sobre una secuencia de datos (SD) proveniente de su propia ML.

Taxonomía de Flynn

Máquinas SIMD

Cada instrucción es ejecutada por varios elementos de proceso (EP), cada uno con sus propios datos (ML).

- Ejemplo máquina SIMD: Procesadores vectoriales y matriciales.

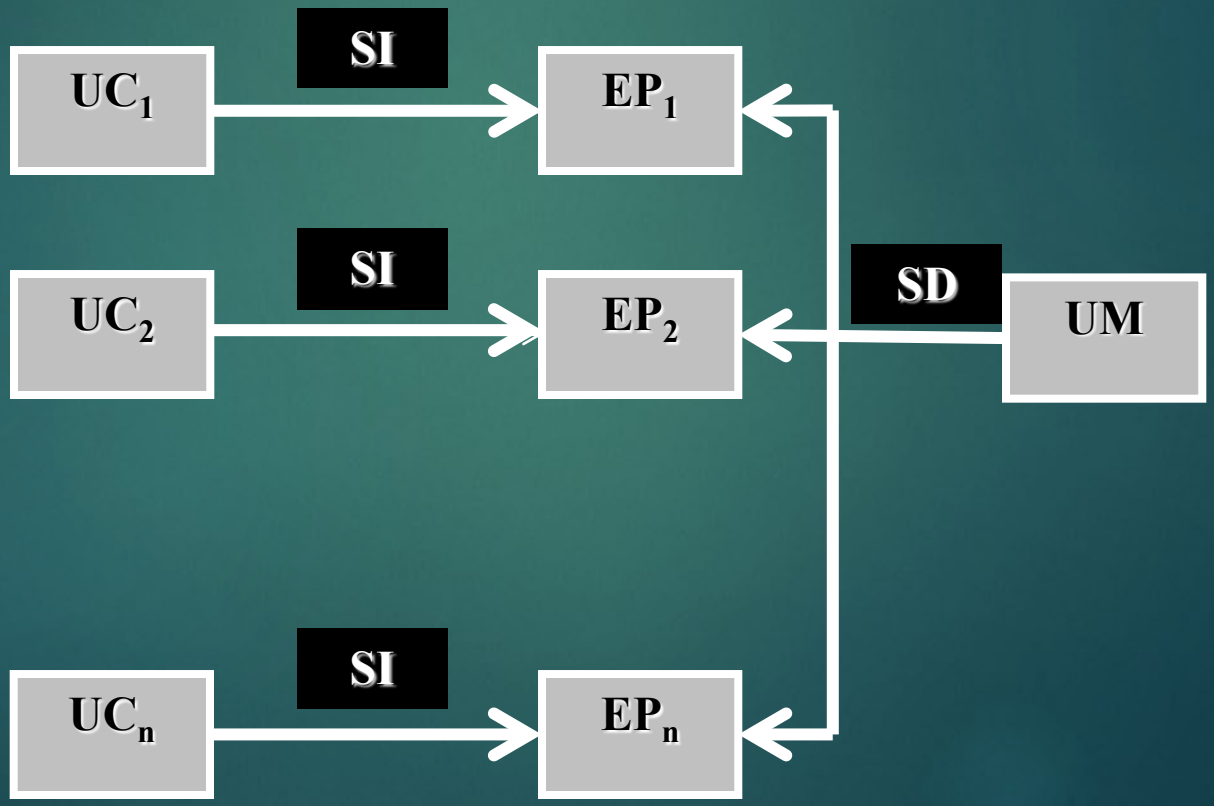
En estas máquinas se requiere disponer de:

- Un conjunto ampliado de instrucciones (además de las "convencionales" o escalares) para permitir manejar operaciones vectoriales. Es decir, el repertorio de instrucciones incluye operaciones de suma, almacenamiento, multiplicación, etc., de vectores, para poder asignar a los EPs.
- Se deben agregar instrucciones para transferir datos entre EPs, que es un atributo típico de un "lenguaje paralelo".

Taxonomía de Flynn

Máquinas MISD

La imagen siguiente muestra el esquema de una máquina tipo MISD (Multiple Instruction – Single Data).



Taxonomía de Flynn

Máquinas MISD

Es un sistema compuesto por:

- una matriz de elementos computacionales o elementos de proceso (EP) cada uno con su propia Unidad de Control (UC).
- Una Unidad de memoria (UM) para todos los EP.

Cada UC interpreta una secuencia de instrucciones (SI) que cada EP resuelve simultáneamente sobre una única secuencia de datos (SD) proveniente de una sola memoria.

Taxonomía de Flynn

Máquinas MISD

Se transmite una única secuencia de datos (SD) a un conjunto de procesadores cada uno ejecutando su propia secuencia de operaciones (MI).

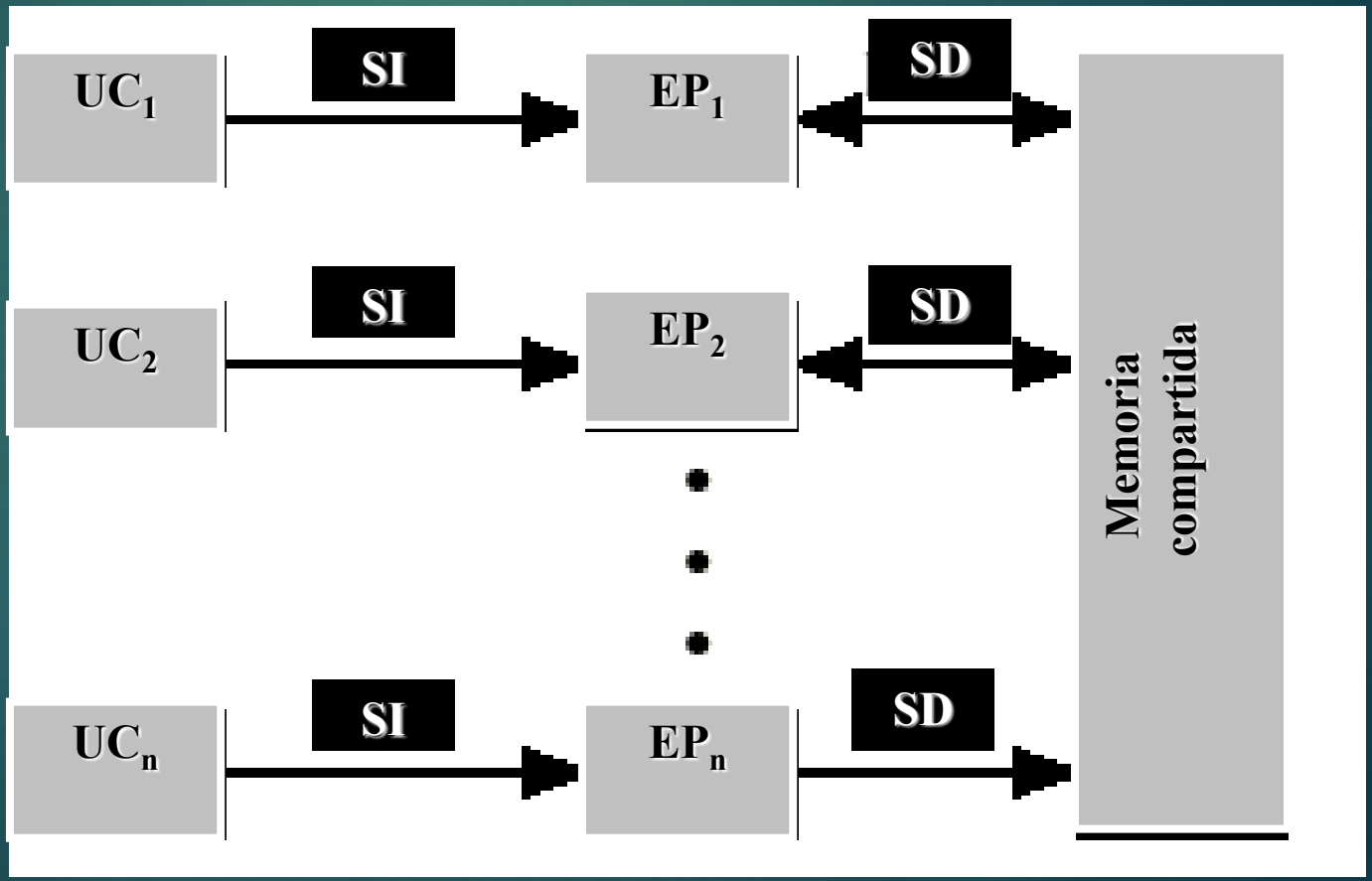
Cada procesador ejecuta una secuencia de instrucciones diferente.

- Ejemplo: es un modelo teórico, no hay registros de máquinas de este tipo (hay algunas de características similares pero no cumplen los requisitos para ser exactamente tipo SIMD).

Taxonomía de Flynn

Máquinas MIMD

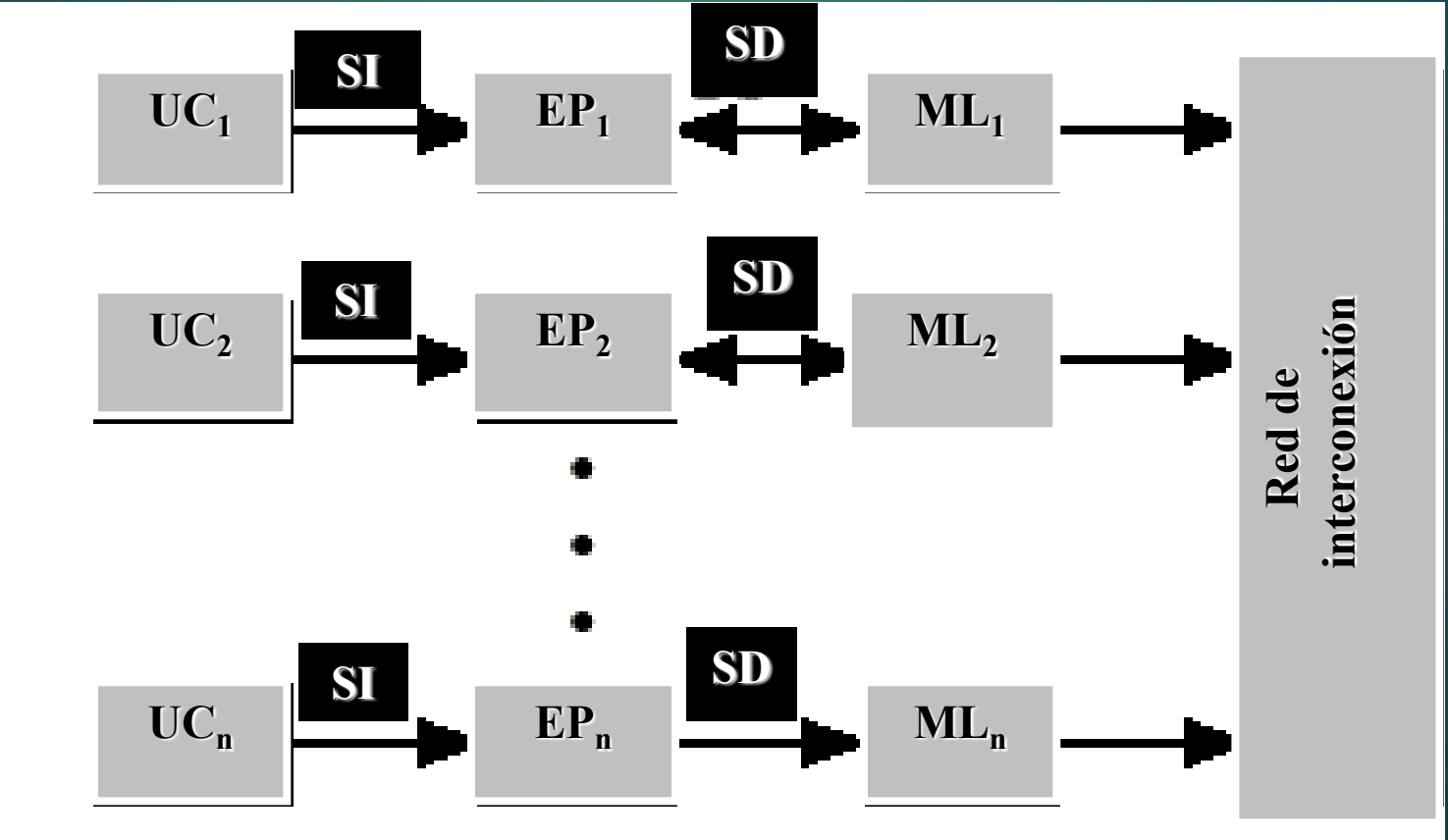
Existen 2 modelos de Computadoras tipo MIMD (Multiple Instruction – Multiple Data). Un modelo es MIMD de Memoria compartida.



Taxonomía de Flynn

Máquinas MIMD

El segundo tipo de Computadoras tipo MIMD es MIMD de Memoria distribuida.



Taxonomía de Flynn

MIMD

Es un sistema compuesto por:

- Múltiples procesadores (UC), ejecutando
- Múltiples secuencias de instrucciones (MI), sobre
- Múltiples secuencias de datos (MD).

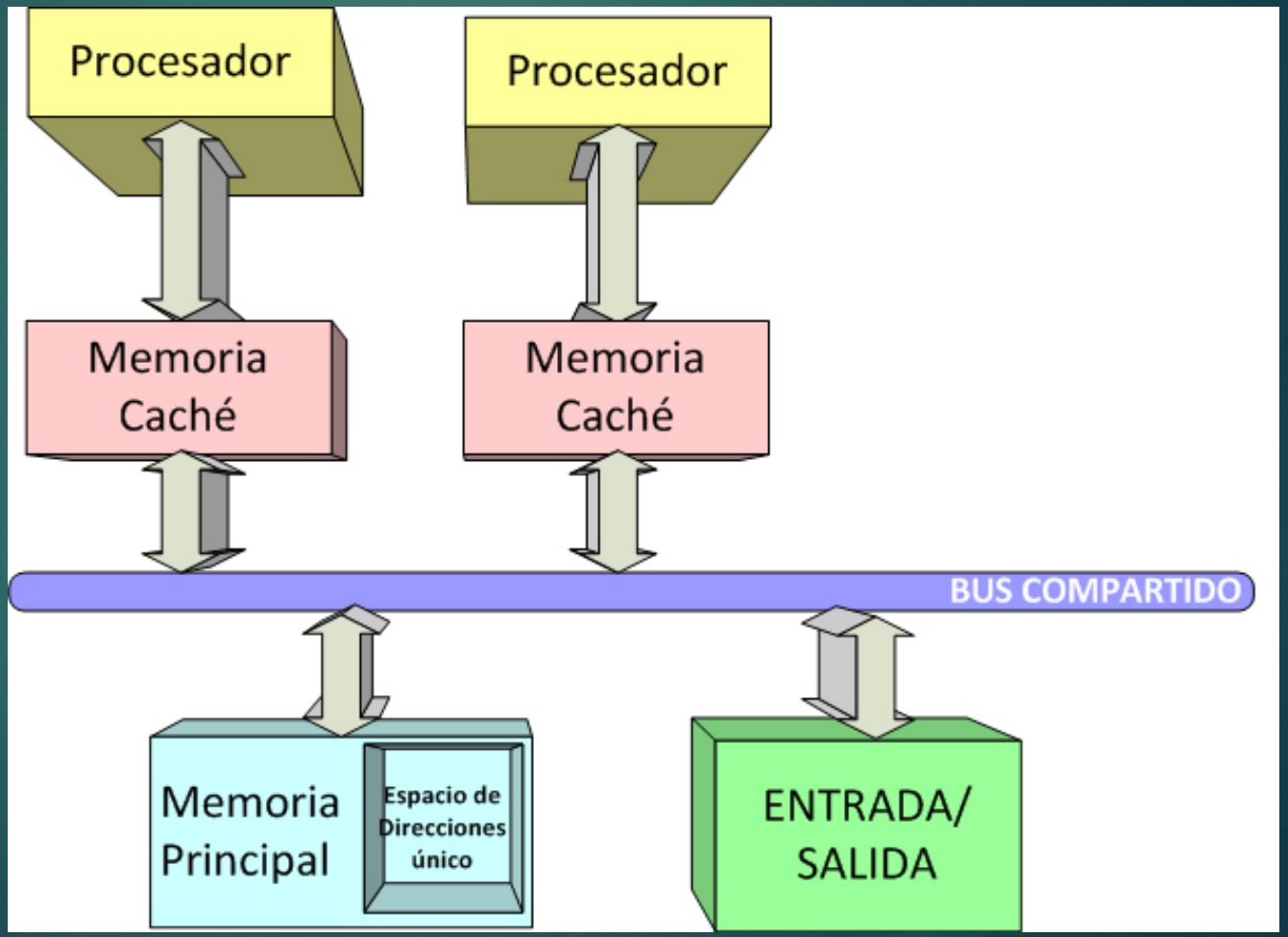
Se pueden dividir según la organización de la Memoria (que es también la forma de comunicarse) en:

- MIMD de Memoria compartida, con 2 variantes:
  - SMP (multiprocesadores simétricos)
  - Sistemas NUMA
- MIMD de Memoria distribuida, comúnmente llamados Clusters.

Multiprocesadores Simétricos SMP

Multiprocesadores SMP

El modelo de computadora tipo SMP se muestra en la figura siguiente.



Multiprocesadores Simétricos SMP

Es un sistema con las siguientes características principales:

- Dos o más procesadores idénticos (o muy similares) de capacidades comparables (homogéneos).
- Memoria principal y E/S única (compartida) por todos los procesadores.
- Interconexión mediante un bus u otro tipo de medio similar (“fuertemente acoplados”).
- Igual tiempo de acceso a la memoria para todos los procesadores. Por eso se los identifica como del tipo “UMA”: Uniform Memory Access.
- Todos los procesadores pueden desempeñar las mismas funciones.
- Sistema operativo integrado, que proporciona la interacción entre los procesadores y sus programas.

Multiprocesadores Simétricos SMP

Ventajas

- Mayores prestaciones: en general tienen buenos resultados, si las tareas pueden organizarse en paralelo.
- Buena disponibilidad: un fallo en un procesador no detiene la operación del sistema, dado que todos los procesadores pueden hacer las mismas tareas.
- Crecimiento: pueden aumentarse las prestaciones añadiendo más procesadores, pero hay restricciones a este mecanismo.
- Escalado: normalmente limitado, en función de la cantidad de procesadores.

Multiprocesadores Simétricos SMP

Desventajas

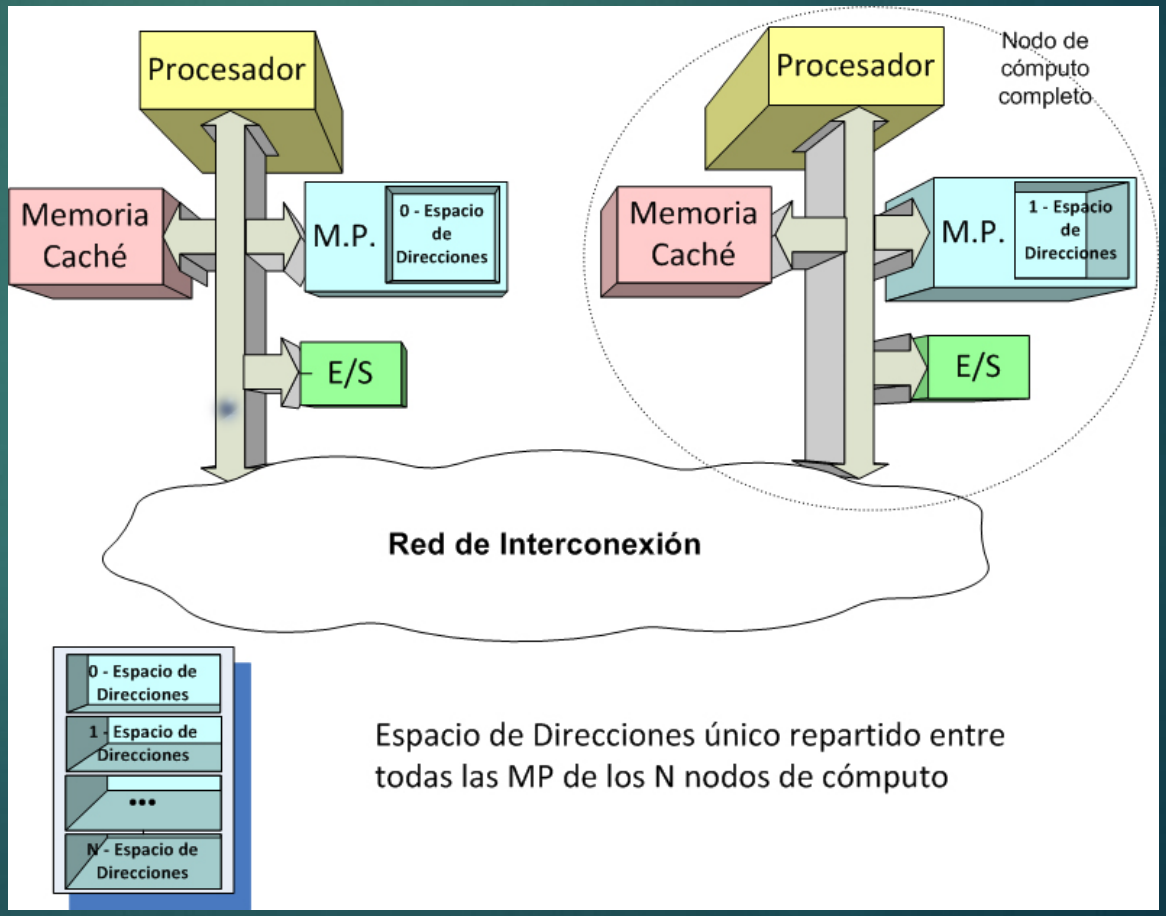
Los principales problemas se originan en los conflictos por el bus compartido, la coherencia y consistencia de los datos, y la sincronización de tareas entre procesadores.

- Las prestaciones están limitadas por el tiempo de ciclo del bus.
- Cada procesador está equipado con una memoria cache, que reduce los accesos a memoria y mejora las prestaciones. Pero eso trae un problema.
- Al disponer una caché en cada procesador, se pueden producir problemas de coherencia de cache (datos que pueden estar en más de una caché). Por razones de velocidad, este problema debe ser resuelto por el hardware. Se requieren usar protocolos para la administración de los datos en las caché (protocolos de coherencia tipo sondeo o snoopy).

Multiprocesadores NUMA

Multiprocesadores con memoria compartida y distribuida - NUMA

El modelo de computadora tipo NUMA se muestra en la figura siguiente.



Multiprocesadores NUMA

Es un sistema con las siguientes características principales:

- Dos o más procesadores idénticos (o muy similares) de capacidades comparables (homogéneos) formando “nodos”.
- Cada nodo es un procesador completo con su propia memoria local y E/S. Los nodos tienen características muy similares.
- Los nodos están interconectados mediante una red de interconexión.
- El espacio de direcciones (memoria lógica) es común a todos los procesadores, por eso es un sistema de memoria compartida. Pero el tiempo de acceso al espacio de memoria es distinto para el acceso a la memoria local que para el acceso a las memorias de los otros nodos.
- Por eso se los llama NUMA: Non Uniform Memory Access.

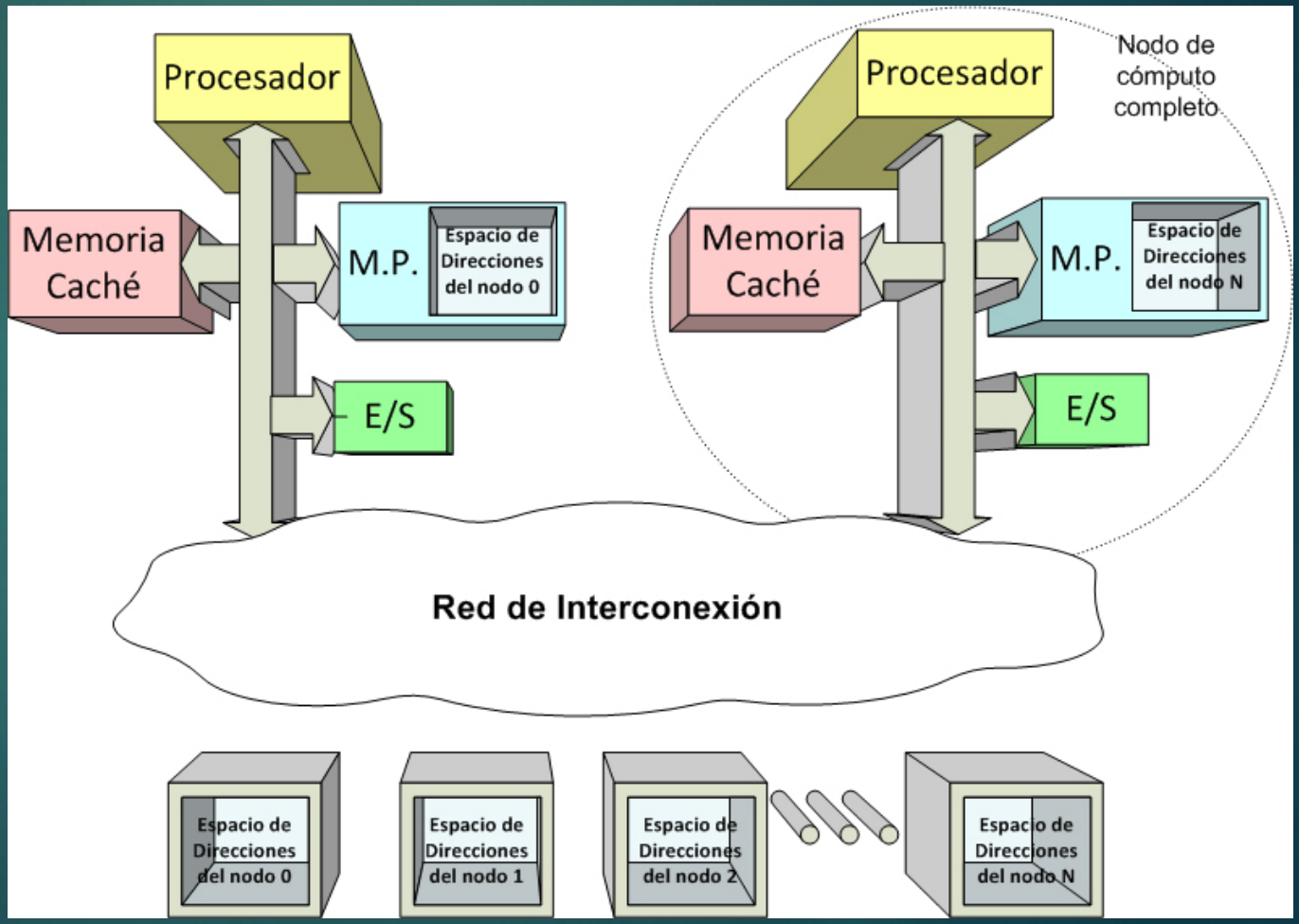
Multiprocesadores NUMA

- Todos los procesadores pueden desempeñar las mismas funciones.
- El Sistema operativo está integrado, y proporciona la interacción entre los procesadores y sus programas.

Cluster

Multiprocesadores de memoria distribuida - Cluster

El modelo de una arquitectura tipo Cluster se muestra en la figura siguiente.



Cluster

Es un sistema con las siguientes características principales:

- Compuesto por 2 o más nodos.
- Cada nodo es un procesador completo con su propia memoria local y E/S. Los nodos pueden tener características similares (“homogéneos”) o distintas (“heterogéneos”).
- Interconectados mediante una red de interconexión. Debido al tipo de red y protocolo de comunicación se los considera del tipo “levemente acoplados”.
- Como los espacios de direcciones son independientes es un sistema de Memoria distribuida.

Cluster

- Todos los procesadores pueden desempeñar las mismas funciones.
- El Sistema operativo está integrado, y proporciona la interacción entre los procesadores y sus programas.
- La comunicación entre procesos es en base a mensajes (a resolver por el programa).

Cluster

- Los Cluster, básicamente, son computadoras completas, interconectadas, que trabajan conjuntamente como un único recurso.
- Es decir, para las tareas en ejecución se comportan como si fuera una única máquina.
- Como ya se dijo, cada computadora se denomina “nodo”.
- En general, presentan prestaciones y disponibilidad elevadas.
- Las aplicaciones son propias de un servidor, y constituyen una alternativa a los SMP.

Cluster

Ventajas

- Escalabilidad absoluta: dependiente de la cantidad de nodos incorporados se puede disponer de mayores prestaciones .
- Escalabilidad incremental: posibilidad de agregar nuevos nodos fácilmente.
- Alta disponibilidad: capacidad de seguir operando con nodos en falla.
- Mejor relación precio/prestaciones: porque se usan equipos de cómputo estandar (y, posiblemente, de bajo costo).

Cluster vs SMP

SMP

- Permiten dar soporte a aplicaciones de alta demanda de recursos.
- Disponibles comercialmente (SMP es más antiguo).
- Más fácil de administrar y configurar.
- Cercano a los sistemas de un solo procesador.
- La planificación (scheduling) es la diferencia principal
- Menos espacio físico / Menor consumo de potencia
- SMP tiene límite práctico en su número de procesadores: entre 16 y 64 por degradación de prestaciones.

Cluster vs SMP

Cluster

- Permiten dar soporte a aplicaciones de alta demanda de recursos.
- Disponibles comercialmente (SMP es mas antiguo).
- Superior escalabilidad incremental y absoluta
- Superior disponibilidad.
- Mayor redundancia
- En Clusters cada nodo tiene su propia memoria principal. Así las aplicaciones no ‘ven’ la memoria global.
- La coherencia es mantenida por software y no por hardware.
- Alternativa a sistemas SMP para brindar multiprocesamiento a gran escala. Por ejemplo, el SGI Origin de Silicon Graphics es un NUMA con 1024 procesadores MIPS R10000.

TIPOS DE ACCESOS A MEMORIA: UMA, NUMA, CC-NUMA

Los sistemas multiprocesadores se pueden clasificar de acuerdo al tipo de acceso a memoria en:

1. UMA - Uniform memory access
   - Igual tiempo de acceso a todas las regiones de memoria.
   - Igual tiempo de acceso a memoria para los diferentes procesadores.

   Ej: SMP

TIPOS DE ACCESOS A MEMORIA: UMA, NUMA, CC-NUMA

2. NUMA - Non-uniform memory access
   - El tiempo de acceso de un procesador difiere, dependiendo de la región de memoria que accede.
   - Diferentes procesadores acceden a diferentes regiones de memoria a diferentes velocidades.

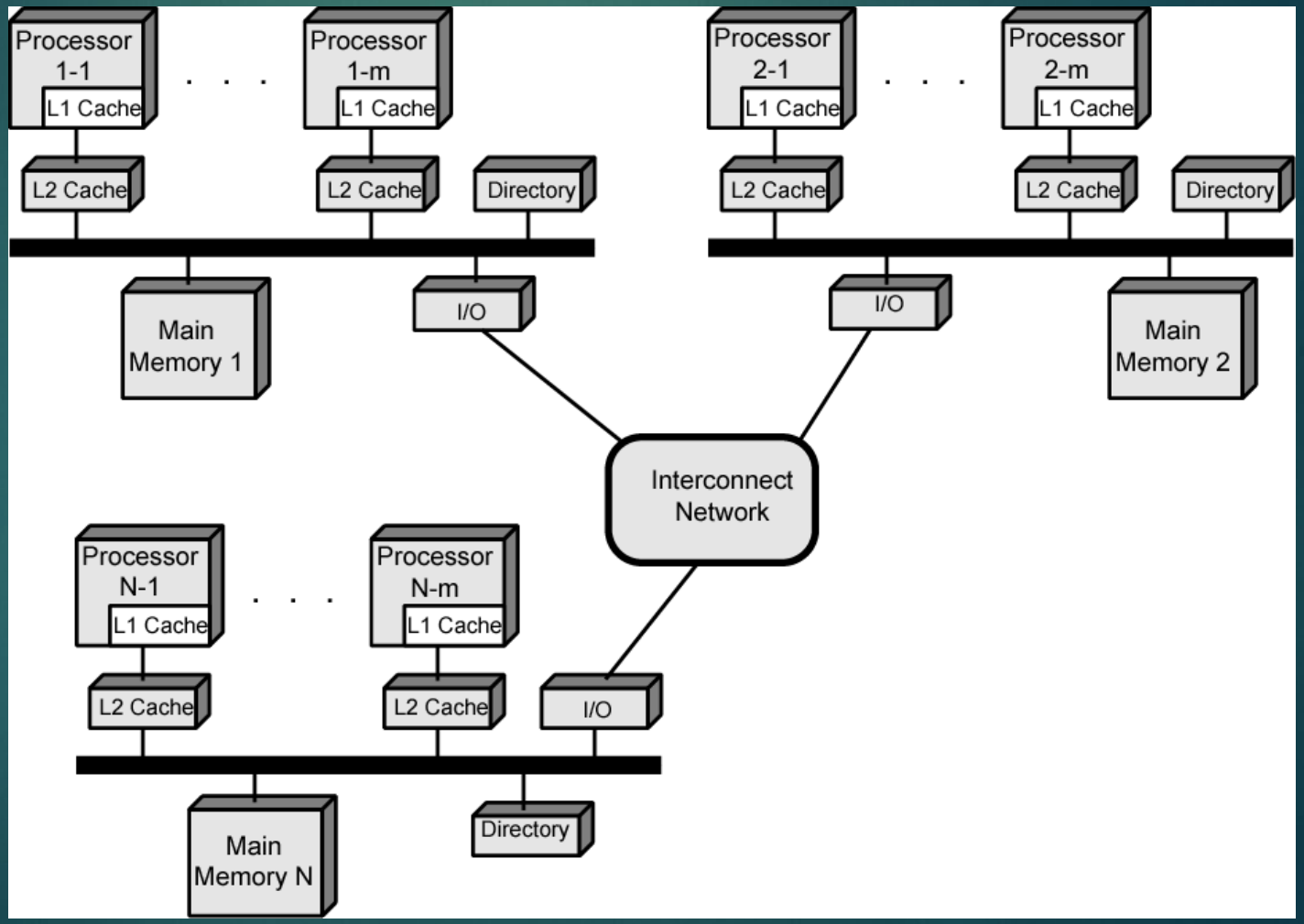
   Ej: Cluster

3. CC-NUMA - cache coherente NUMA
   - Es un NUMA que mantiene coherencia de cache entre las cache de los distintos procesadores.

   Ej: Sistemas con memoria compartida distribuida.

Sistemas CC-NUMA

En la imagen siguiente se muestra el modelo de organización de un sistema NUMA con coherencia en las caché (CC-NUMA).



Sistemas CC-NUMA

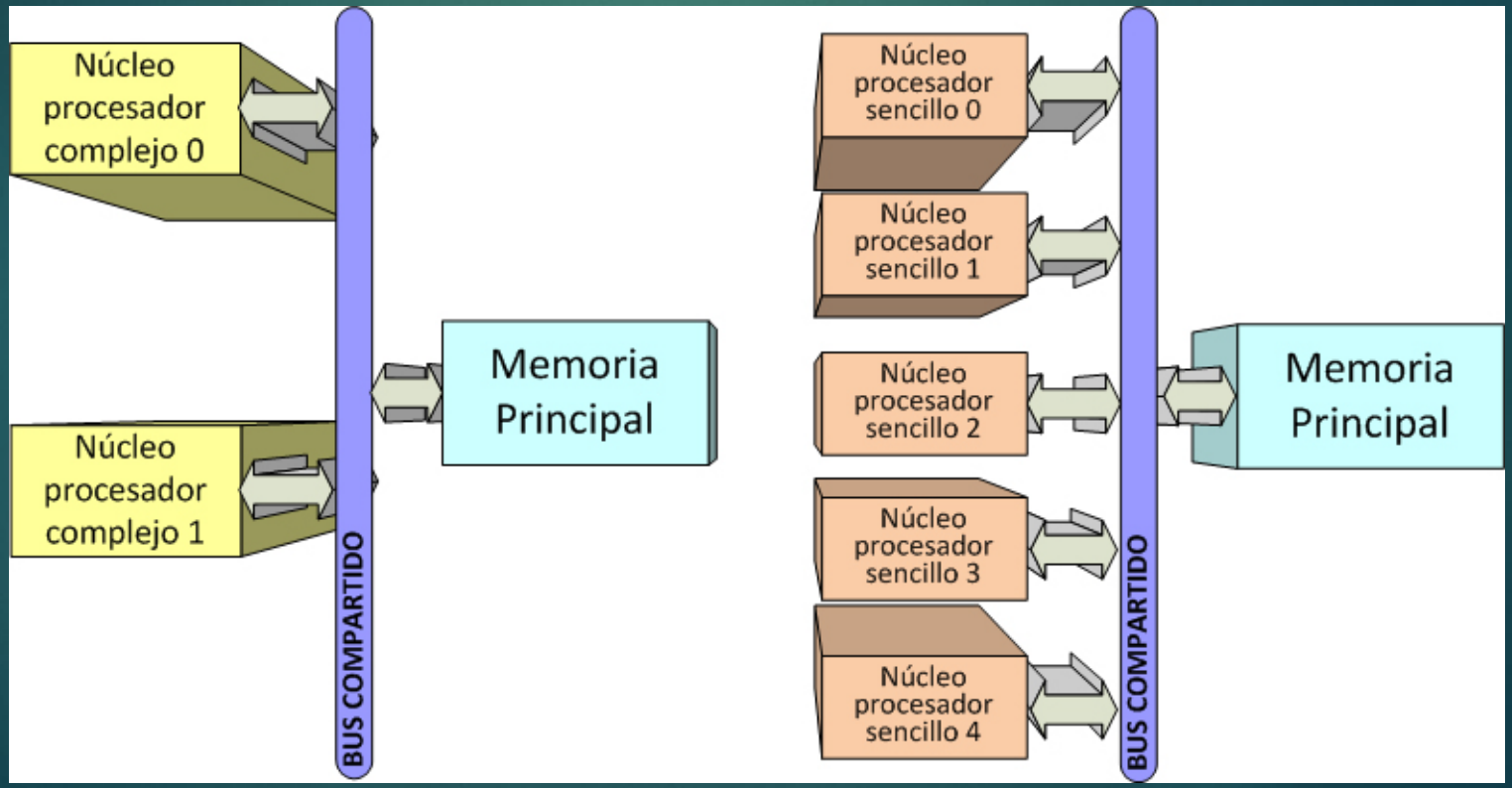
Los sistemas CC-NUMA presentan las siguientes características:

- Cada nodo tiene 2 o más procesadores (por ejemplo un SMP), cierta cantidad de memoria (principal) y E/S.
- Cada procesador tiene su cache (típicamente L1 y L2).
- Los nodos están interconectados por algún tipo de red.
- Existe un espacio de direcciones de memoria único para todos los procesadores de todos los nodos.
- Orden de acceso a memoria:
  - cache L1 (local al procesador).
  - cache L2 (local al procesador).
  - Memoria principal (local al nodo).
  - Memoria remota (petición por red).
- La coherencia se mantiene en forma automática y transparente

Arquitecturas on chip con Memoria compartida

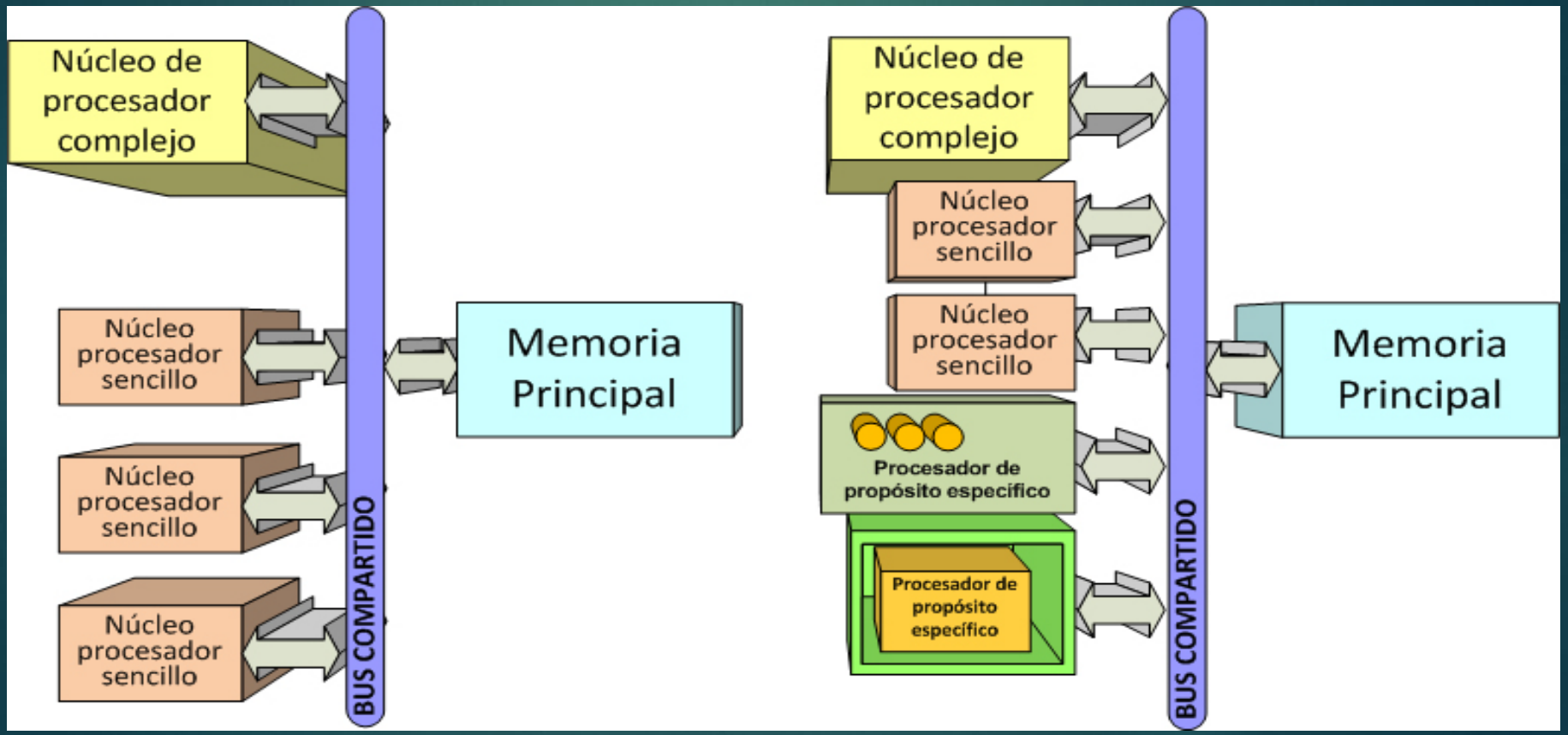
Las arquitecturas on-chip con memoria compartida pueden ser homogéneas o heterogéneas.

Son homogéneas si todos los procesadores son idénticos.



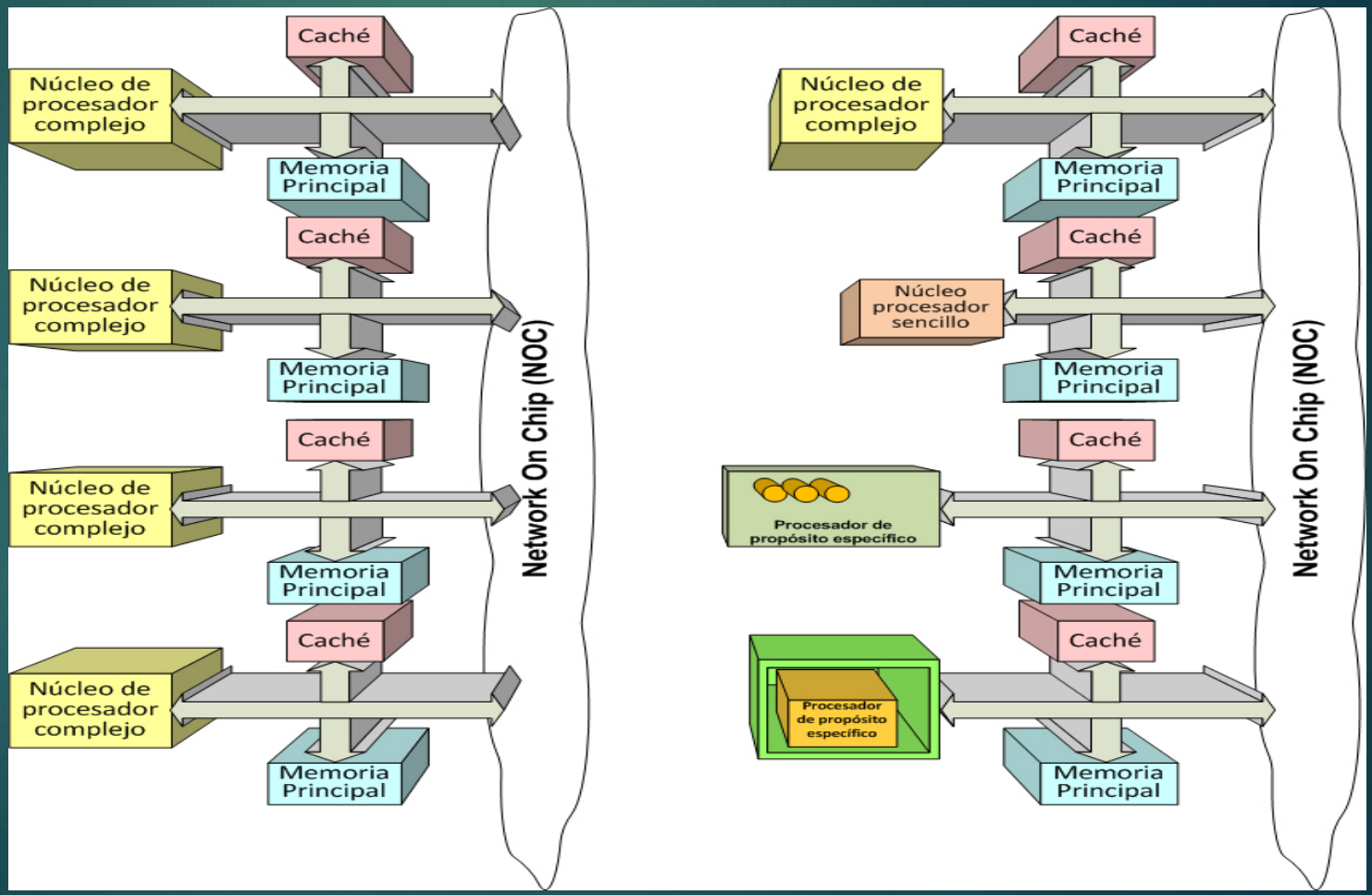
Arquitecturas on chip con Memoria compartida

Son heterogéneas si los procesadores tienen distintas prestaciones



Arquitecturas on chip con Memoria distribuida

Las arquitecturas on chip con memoria distribuida también pueden ser homogéneas o heterogénas.



Procesamiento Multihebra

Proceso

Un Proceso es un programa ejecutándose (“corriendo”) en un Sistema que:

- Es “propietario” de recursos propios.
- Maneja un espacio de direcciones virtuales para almacenar la imágen de un proceso (code, data, stack, etc) con:
  - Información propia del estado del proceso.
  - Planificación/ejecución por el sistema operativo.
- La comunicación entre procesos es a través de mecanismos específicos.
- La conmutación entre procesos (“Process switch”) requiere un cambio de los recursos asignados (mediante mecanismos precisos y complejos).

Procesamiento Multihebra (Multithreading)

Hebra (thread) o hilo

Es una unidad de trabajo que:

- Tiene su propio contexto de procesador (incluido PC y SP) y área de datos para su pila (stack).
- Comparte con otras hebras, código, variables globales (espacio de memoria), archivos abiertos por el proceso al que pertenecen, etc.
- Se ejecuta secuencialmente.
- Relativamente fácil de interrumpir, el procesador cambia a otra hebra rápidamente si lo necesita.
- La conmutación de hebra (“Thread switch”) es un cambio de control del procesador entre hebras de un mismo proceso.
- Un Proceso está compuesto de muchas Hebras o Hilos.

Procesamiento Multihebra (Multithreading)

Se pueden aumentar las prestaciones de un Sistema en 2 niveles:

- Nivel de instrucciones (IPL): aumentando la cantidad de instrucciones ejecutadas en paralelo, explotando el paralelismo de instrucciones secuenciales.
- Nivel de hebras (TPL): aumentando el nivel de hebras en paralelo, explotando el paralelismo entre instrucciones pertenecientes a distintos hilos de ejecución. Esto involucra el sistema operativo, el compilador y el hardware.

Procesamiento Multihebra (Multithreading)

Como ya se vió en clases anteriores, el IPL explota el paralelismo a nivel de instrucciones, pertenecientes a un hilo o hebra de ejecución. Si el programa no tiene mucho paralelismo, el resultado va a ser pobre.

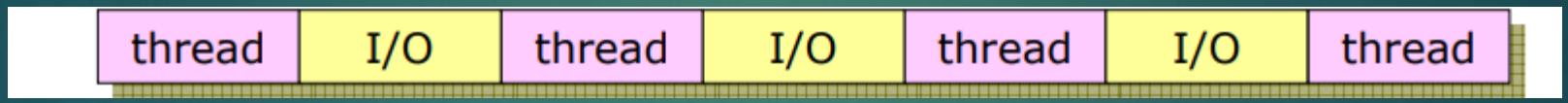
Un proceso puede tener muchos hilos o hebras. Entonces:

- Las instrucciones de diferentes hebras pueden ser paralelizadas.
- Se pueden mejorar las prestaciones generales si se ejecutan más de una hebra en paralelo (simultáneamente).
- La idea es que si hay recursos libres durante la ejecución de un hebra, ejecutando varias hebras se puede hacer un mejor aprovechamiento de los recursos.
- No se mejora el tiempo de ejecución de un hebra.

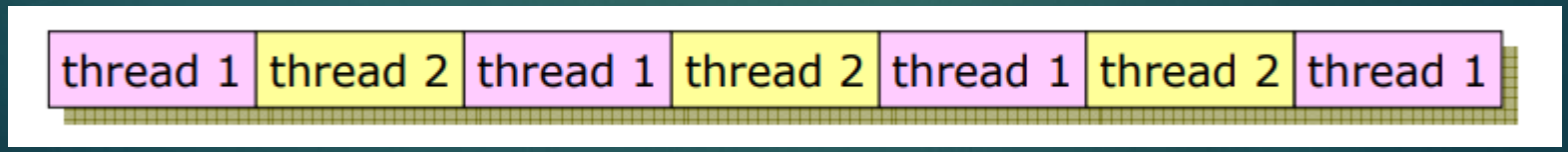
Procesamiento Multihebra (Multithreading)

El procesamiento de multiples hebras (multithreading) consiste en ejecutar 2 o más hebras en forma concurrente.

Un ejemplo claro es el siguiente. Se está ejecutanado 1 hebra únicamente, y el procesador queda inactivo durante los períodos de tiempo donde se ejecutan acciones de I/O.



Pero si se pueden ejecutar múltiples hebras, durante los períodos de I/O puede ejcutar otras hebras.



Procesamiento Multihebra (Multithreading)

La ejecución de múltiples hebras puede ser de 2 tipos:

- Multihebras explícitas: cuando son definidas en el programa, tanto a nivel de usuario (aplicaciones) como a nivel de Sistema operativo (núcleo).
- Multihebras implícitas: cuando son definidas por el compilador (estático) o por hardware (dinámico).

Para poder implementar procesamiento multihebra es necesario, al menos:

- Un PC (contador de programa) distinto para cada hebra que pueda ejecutarse concurrentemente.
- HW para ejecución concurrente.

Procesamiento Multihebra

Cambios de contexto en ejecución Multihebra

El contexto de una hebra está formado por:

- El contador de programa (PC)
- Registros de datos, de direcciones (punteros), de estado, control, de segmentos
- Datos propios en memoria (eventualmente)
- Datos en caché (eventualmente)

Cuando se cambia de una hebra a otra, se produce un cambio de contexto (de hebras).

Los cambios de contexto de hebras pueden ser de 2 tipos:

- tradicional
- rápido.

Procesamiento Multihebra

Cambios de contexto (de hebras) tradicional

- Lo produce el SO mediante una interrupción, típicamente asociada a un timer (asignación temporal de hebras).
- La interrupción detiene la hebra en ejecución.
- El SO salva el contexto de la hebra en ejecución.
- El SO recupera el contexto de la hebra detenida anteriormente.
- La hebra detenida anteriormente se reinicia mediante un retorno de interrupción.
- La hebra detenida no se entera que fue interrumpida.

Procesamiento Multihebra

Cambios de contexto (de hebras) rápido (por hardware)

- Se produce porque la hebra queda en espera por alguna razón, típicamente por un fallo de acceso a la caché (tener en cuenta que la PF de acceso a la caché es de varios órdenes de magnitud respecto de un acceso a la caché).
- Si el procesador tiene duplicados, al menos, el PC y los registros (todos o gran parte), entonces el procesador conmuta de hebra sin necesidad de salvar el contexto de la hebra detenida por el fallo (que consume tiempo).
- Este cambio de contexto es mucho más rápido que el tradicional (del SO).

Procesamiento Multihebra

Políticas de ejecución multihebra

Las hebras en ejecución usan recursos. Los recursos pueden ser asignados de 2 formas:

- Estática: el particionado de recursos es fijo.
- Dinámica: el particionado de recursos cambia por algún mecanismo.

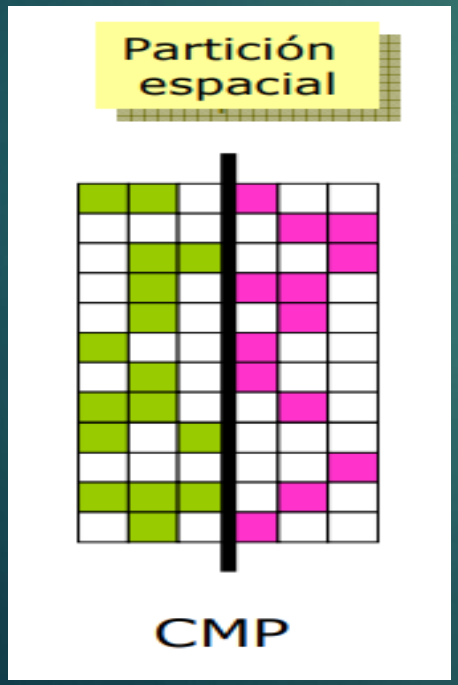
Existen 4 políticas de ejecución multihebra (MT):

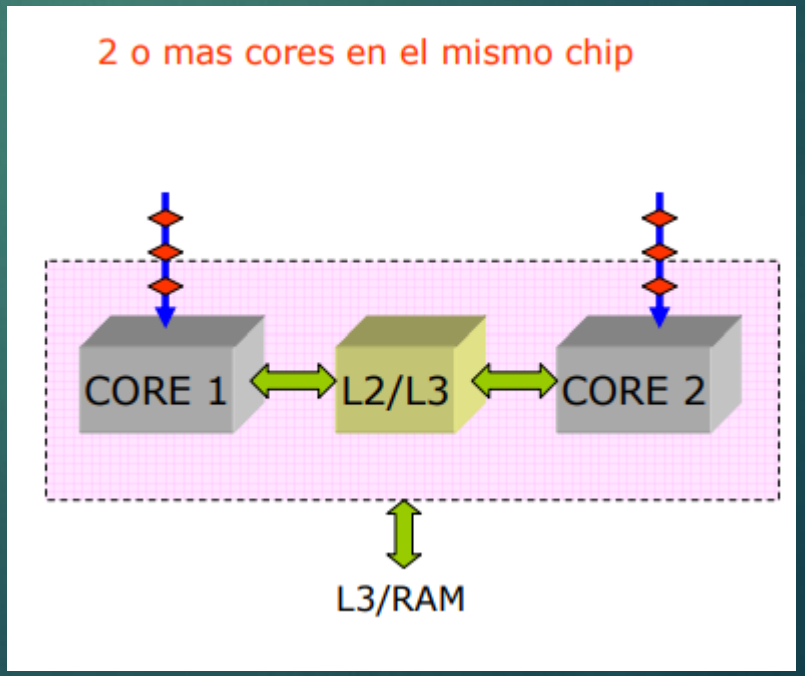
- CMP (Chip Multi Processor)
- FGMT (Fine Grained Multi Threading)
- CGMT (Coarse Grained Multi Threading)
- SMT (Simultaneous Multi Threading)

Procesamiento Multihebra

Procesamientos multihebra tipo CMP

- El procesador dispone de 2 o más núcleos. A cada núcleo se le asigna una hebra en forma fija.
- La política de partición CMP es de tipo estática espacial.
- Típico de procesadores SMP de 2 o más núcleos.

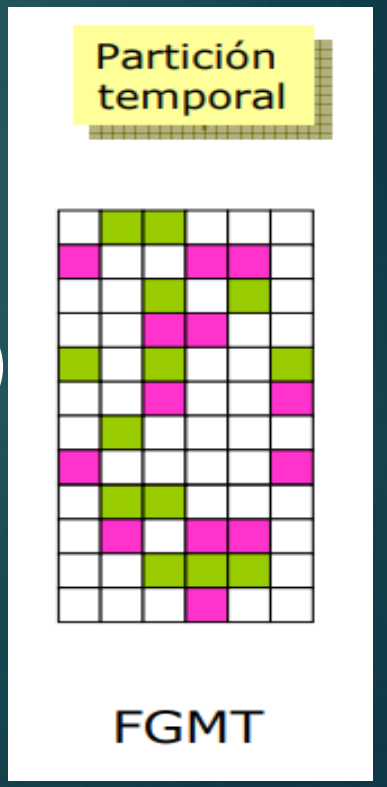




Procesamiento Multihebra

Procesamientos multihebra tipo FGMT

- El procesador conmuta de una hebra a otra en cada ciclo. La conmutación es rápida, 1 sola hebra por vez. El procesador dispone de 2 o mas contextos, para ejecutar 2 o más hebra concurrentemente (Ej: Cray CDC 6600 y Denelcor HEP).
- La política de partición FGMT es de tipo estática temporal.
- Al ser una conmutación por ciclo se dice que es de "grano fino" (FGMT : Fine grained Multi-threading)
- Reduce las latencias en memoria, y evita y resuelve dependencias de datos.
- Requiere que el compilador encuentre muchas hebras independientes.



Procesamiento Multihebra

Procesamientos multihebra tipo CGMT

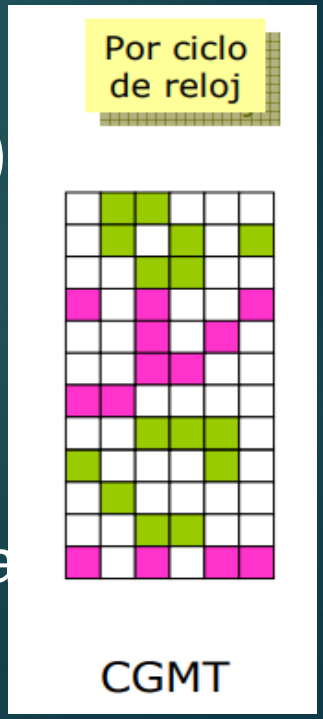
- El procesador conmuta de una hebra a otra cuando una hebra se detiene por algún evento de gran latencia (por ejemplo, fallo de acceso a la caché).
- La política de partición CGMT es de tipo dinámica temporal.
- La conmutación por algún fallo se dice que es de

"grano grueso" (CGMT : Coarse grained Multi-threading)

- En general requiere un manejo más complejo

porque la conmutación es asincrónica.

- Se deben replicar registros para conmutar rápido.
- Se debe hacer una distribución equitativa de las hebra



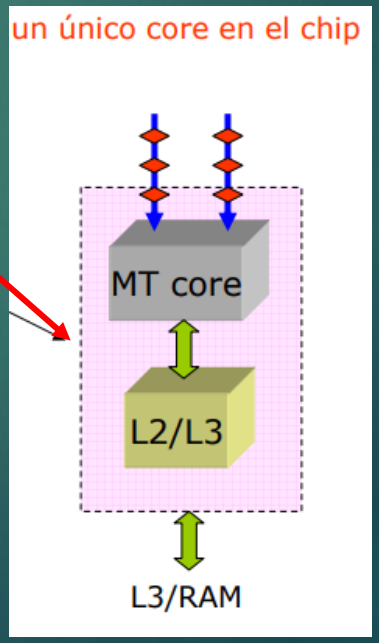
Procesamiento Multihebra

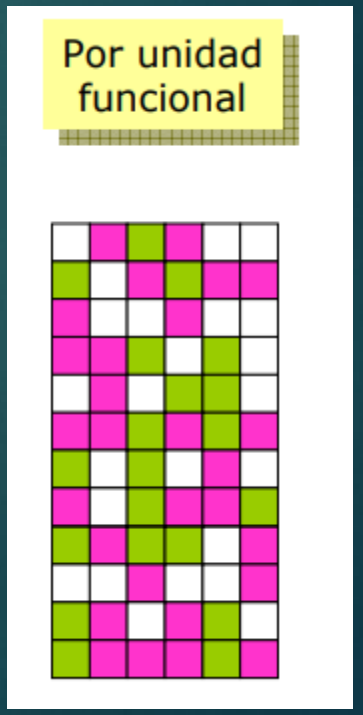
Procesamientos multihebra tipo SMT

- Hay varias hebras ejecutándose en varias unidades funcionales de un procesador superescalar (1 solo núcleo).
- La política de partición SMT es de tipo dinámica espacial.

Ej: Pentium 4, core i3, i5, i7

(Intel lo llama Hyper-threading)





Procesamiento Multihebra

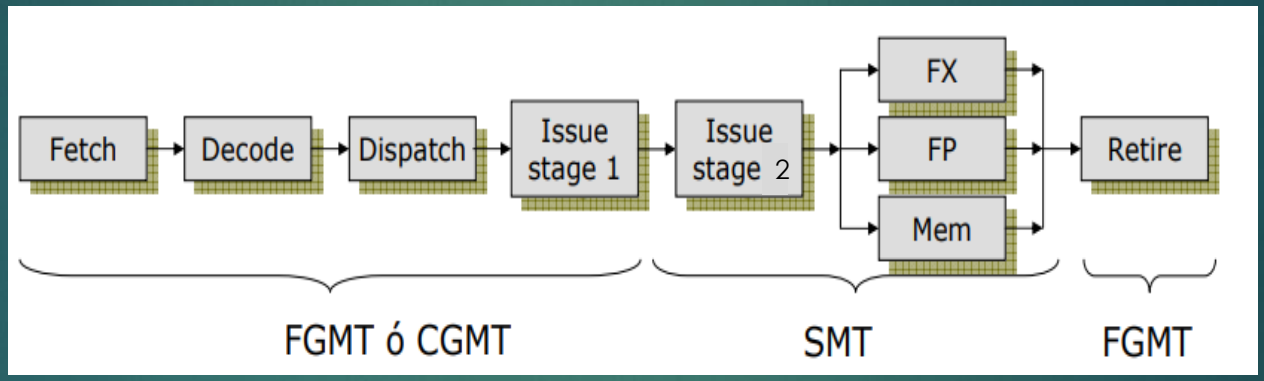
Pentium 4

- El Pentium 4 es un procesador con una arquitectura Multithreading (Hyper-threading o HT como la llama Intel) híbrida, que permite disponer de 2 procesadores lógicos compartiendo algunos de los recursos de ejecución del procesador físico (es decir, 1 solo núcleo).
- El SO reparte las tareas entre ellos.
- Cada procesador lógico puede actuar individualmente.
- La mejora depende fuertemente de grado de paralelización de las tareas. En algunos casos puede llegar a un 30% de mejora. En otros puede hacer la ejecución más lenta (por ello se puede deshabilitar el “modo HT”).

Procesamiento Multihebra

Pentium 4

El Pentium 4 tiene distintas políticas de tratamiento de las hebras dependiendo de las unidades funcionales involucradas (por eso es un híbrido). En la imagen siguiente se pueden ver las estrategias usadas en cada etapa.



- Etapas Fetch-Decode-Dispatch-Issue 1: FGMT, hasta que una hebra se frena, pasando a CGMT.
- Issue 2 – Ex – Mem: SMT, las hebras pueden mezclar instrucciones.
- Retire: FGMT

Referencias

- Organización y Arquitectura de Computadoras, William Stallings, Capítulo 16 de 5ta edición ó Capítulo 18 de 7ma edición.
- Diseño y evaluación de arquitecturas de computadoras, M. Beltrán y A. Guzmán, Capítulo 5 de 1ra edición.