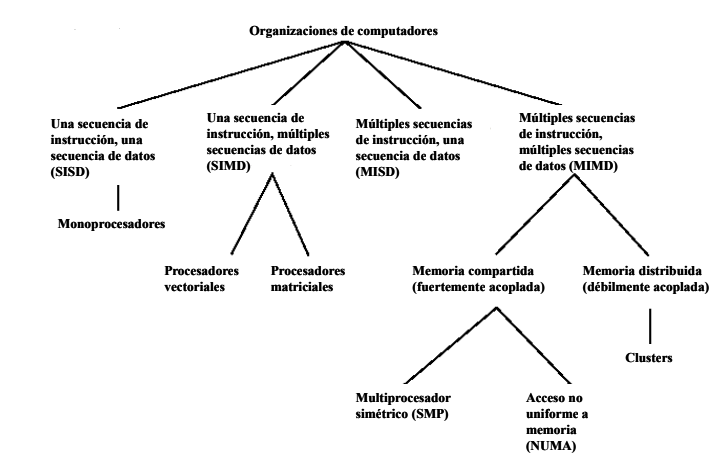
[Resumen clase 9](https://drive.google.com/file/d/1bZuKMa76frNRy_zkSfDny3fSHg96l3hB/view?usp=sharing)

# Procesamiento paralelo

Sea cual sea el nivel de prestaciones, la demanda de máquinas de mayor rendimiento seguirá existiendo

* Mejorar el rendimiento de una máquina con un solo procesador (Paralelismo a nivel instrucción -ILP)
* Arquitecturas de sistemas con varios procesadores (Paralelismos a nivel proceso)

# Taxonomía de las arquitecturas (Flynn)



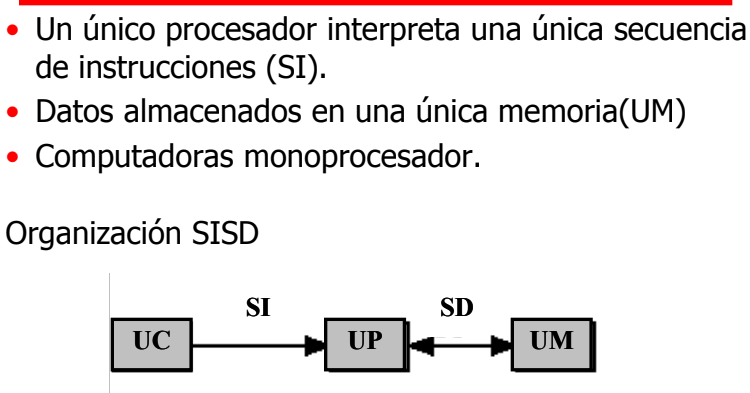
# Categoría de computadoras

## SISD

Una secuencia de instrucciones y una secuencia de datos

* Un único procesador interpreta una única secuencia de instrucciones (SI)
* Datos almacenados en una única memoria (UM)
* Computador monoprocesador

Organización SISD:

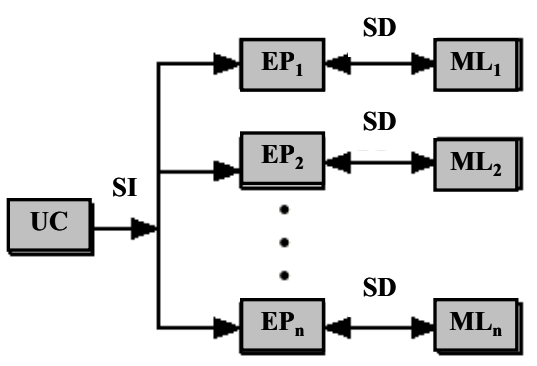


## SIMD

Una secuencia de instrucciones y múltiples secuencias de datos

* Una única instrucción máquina controla paso a paso la ejecución simultánea de un cierto número de elementos de proceso (EP)
* Cada elemento de proceso tiene una memoria dedicada (ML)
* Cada instrucción es ejecutada por cada procesador, con un conjunto de datos diferentes
* Procesadores vectoriales y matriciales

Organización SIMD:



### Matricial? Vectorial? Paralelo?

* Computadora con una única “unidad de control” y una matriz de elementos computacionales
* Tipos de instrucciones de procesador:
  + Extensiones de las instrucciones escalares: sumar almacenar, multiplicar, etc. Se convierten en operaciones vectoriales ejecutadas en todos los procesadores de modo simultáneo
  + Debe añadirse la capacidad de transferir al conjunto de instrucciones los datos escalares y vectoriales entre procesadores: atributos de un “lenguaje paralelo”

## MISD

Múltiples secuencias de instrucciones y una secuencia de datos

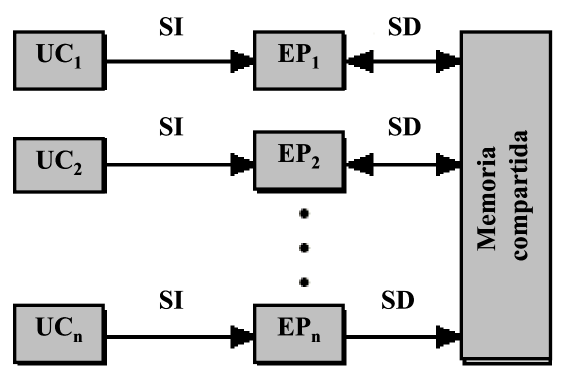
* Se transmite una secuencia de datos a un conjunto de procesadores
* Cada procesador ejecuta una secuencia de instrucciones diferente
* Esta estructura nunca ha sido implementada

## MIMD

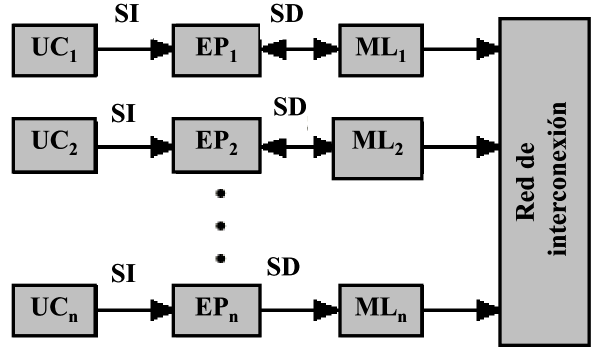
Múltiples secuencias de instrucciones y múltiples secuencias de datos

* Un conjunto de procesadores ejecuta secuencias de instrucciones diferentes en simultáneo
* Conjuntos de datos diferentes
* Se pueden dividir según la forma de comunicarse
  + Memoria compartida: SMP (multiprocesadores simétricos) y sistemas NUMA
  + Memoria distribuida: clusters

Organización MIMD de memoria compartida:



Organización MIMD de memoria distribuida:



### Multiprocesador simétrico - SMP

Computadora autónoma con las siguientes características:

* Dos o más procesadores similares de capacidades comparables
* Comparten la memoria principal y las E/S
* Interconectados mediante un bus u otro tipo de sistema de interconexión
* Tiempo de acceso a memoria similar para todos los procesadores (UMA)
* Todos los procesadores pueden desempeñar las mismas funciones
* Sistema operativo integrado, que proporciona la interacción entre los procesadores y sus programas

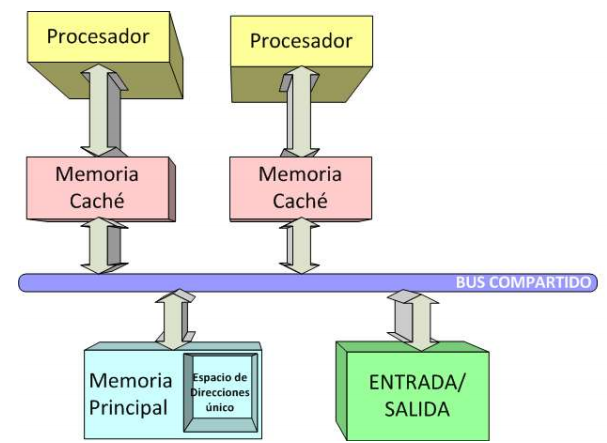
Ventajas potenciales de un SMP:

* Mayores prestaciones si el trabajo a realizar puede organizarse en paralelo
* Buena disponibilidad (un fallo en un procesador no detendrá la computadora)
* Crecimiento incremental (se pueden añadir mas procesadores)
* Escalado (en función de la cantidad de procesadores)
* Cuidado (bus compartido)

Bus de tiempo compartido (desventajas):

* La prestación está limitada por el tiempo de ciclo de bus
* Cada procesador debería estar equipado con una memoria caché para mejorar las prestaciones (se reduciría el número de accesos)
* Se pueden producir problemas de coherencia de cache
  + Este problema debe ser resuelto por el hardware (protocolos de sondeo y protocolos de directorio)

Arquitectura de un SMP:



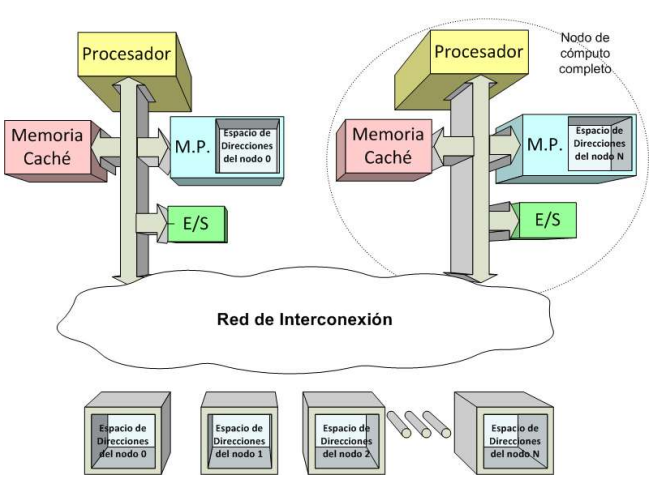
### Clusters

* Computadoras completas interconectadas que trabajan conjuntamente como un único recurso (ilusión de que se trata de una única máquina)
* Cada computadora se denomina “nodo”
* Prestaciones y disponibilidad elevadas
* Aplicaciones propias de un servidor
* Son la alternativa a los SMP

#### Beneficios del cluster

* Escalabilidad absoluta
* Escalabilidad incremental
* Alta disponibilidad
* Mejor relación precio/prestaciones

Arquitectura de un cluster



### Clusters vs SMP

* Ambos:
  + Dan soporte a aplicaciones de alta demanda de recursos
  + Disponibles comercialmente (SMP es más antiguo)
* SMP
  + 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
* Cluster:
  + Superior escalabilidad incremental y absoluta
  + Superior disponibilidad: redundancia

# Términos UMA, NUMA, CC-NUMA

Todos los procesadores tienen acceso a toda la memoria: usan “load” y “store”

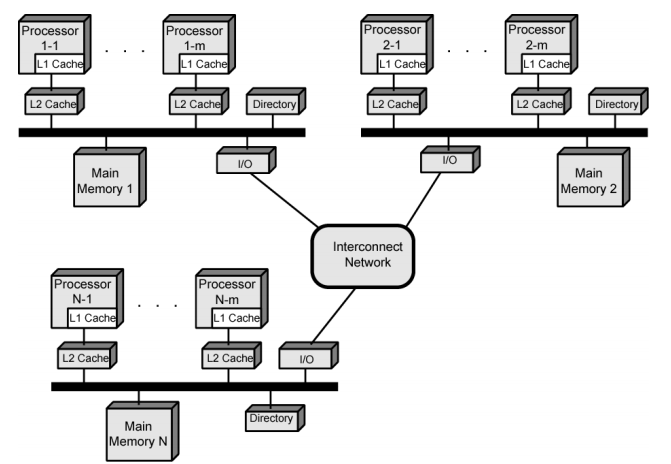
* UMA - Uniform memory access
  + Igual tiempo de acceso a todas las regiones de memoria
  + Igual tiempo de acceso a memoria para los diferentes procesadores
* NUMA - Nonuniform memory access
  + El tiempo de acceso de un procesador difiere dependiendo de la región de memoria que accede
  + Diferentes procesadores acceden a diferentes regiones de memoria a diferentes velocidades
* CC-NUMA - cache coherente NUMA
  + Es un NUMA que mantiene coherencia de cache entre las cache de los distintos procesadores

## Motivación NUMA

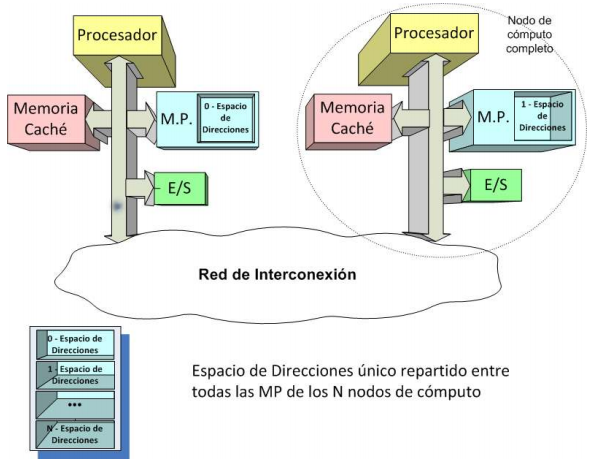
* SMP tiene límite práctico en su número de procesadores (entre 16 y 64 por degradación de prestaciones)
* En clusters cada nodo tiene su propia memoria principal
  + Aplicaciones no “ven” la memoria global
  + Coherencia de cache mantenida por software no por hardware
* NUMA retiene las características tipo SMP y brinda multiprocesamiento a gran escala (Ej: SGI Origin de Silicon Graphics es NUMA con 1024 MIPS R10000)

Objetivo NUMA: tener una memoria transparente del sistema y permitir nodos, cada uno con su propio bus o sistema de conexión interna

## Organización CC-NUMA



## Arquitectura de memoria compartida-distribuida



## Operación CC-NUMA

* Cada procesador tiene cache L1 y L2
* Cada nodo tiene su propia memoria principal
* Nodos conectados por algún tipo de red
* Cada procesador “ve” un único espacio de direcciones de memoria
* 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
* Automático y transparente

## Procesamiento multihebra (multithreading)

* Aumento de paralelismo de instrucciones: sin el aumento de complejidad y consumo de potencia de la segmentación de cauce y los superescalares
* La secuencia de instrucciones se divide en secuencias más pequeñas llamadas hebras (threads) que pueden ejecutarse en paralelo
* Amplia variedad de diseños multihebra

### Términos hebra y proceso

* Concepto de Hebra en procesadores multihebra puede no ser el de S.O. multiprogramados
* Proceso: un programa “corriendo” en una computadora
  + Propiedad de recursos: espacio de direcciones virtuales para almacenar la imagen de un proceso (code, data, stack, etc)
  + Planificación/ejecución: hay camino de ejecución (traza)
* Conmutación de Proceso (process switch)
* Hebra (thread): unidad de trabajo de un proceso que puede asignarse
  + Incluye un contexto de procesador (incluido PC y SP) y área de datos para su pila (stack)
  + Se ejecuta secuencialmente
  + Interrumpible: el procesador cambiaría a otra hebra
* Conmutación de hebra (thread switch): cambio de control del procesador entre hebras de un mismo proceso (usualmente menos costosa que la conmutación de proceso)

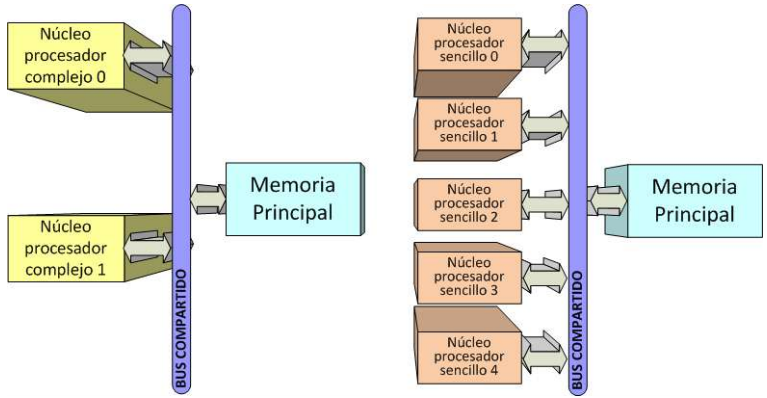
### Multihebra implícito y explícito

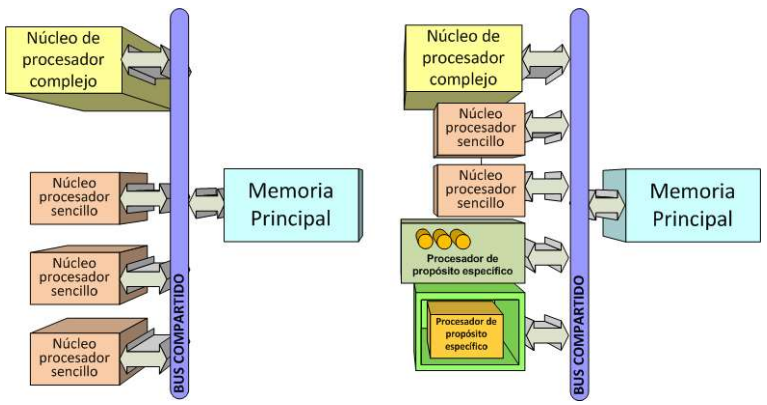
* Multihebra explícito: ejecución concurrente de instrucciones de diferentes hebras explícitas (todos los procesadores comerciales lo usan)
  + Mezcla de instrucciones de diferentes hebras en cauces compartidos
  + O por ejecución paralela en cauces paralelos
* Multihebra implícito: ejecución concurrente de varias hebras extraídas de un único programa secuencial
  + Definidas estáticamente por el compilador o dinámicamente por el hardware

## Procesador multihebra

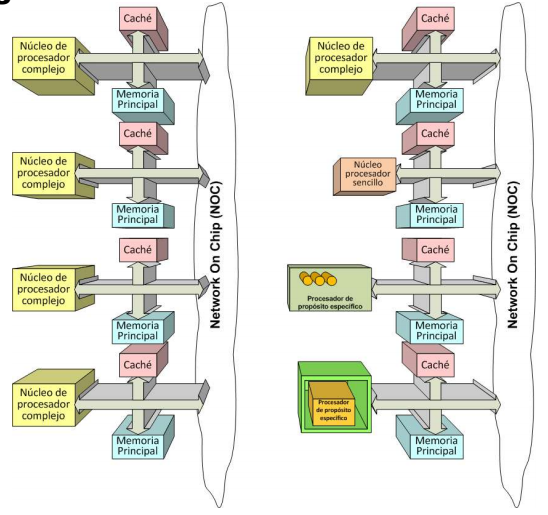
* PC (contador de programa) distinto para cada hebra que pueda ejecutarse concurrentemente
  + Cantidad y tipo de HW para ejecución concurrente
* Se trata cada hebra separadamente: predicción de saltos, renombre de registros y etcétera para optimizar ejecución
  + Paralelismo entre hebras
* Aproximaciones con ejecución simultánea real
  + Multihebra simultánea (SMT) - Pentium 4 HT
  + Multiprocesador monochip

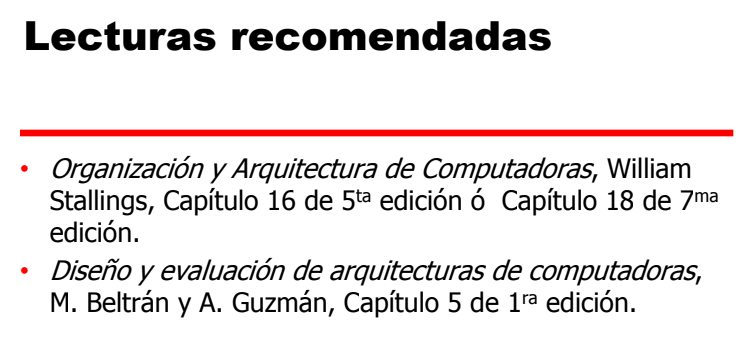
## Arquitecturas on chip (memoria compartida)





## Arquitecturas on chip (memoria distribuida)



[Anexo clase 9](https://drive.google.com/file/d/1aXdhlupp2Rvndl-UWKz65pHGqFXq6kkL/view?usp=sharing)