Arquitectura de Computadoras

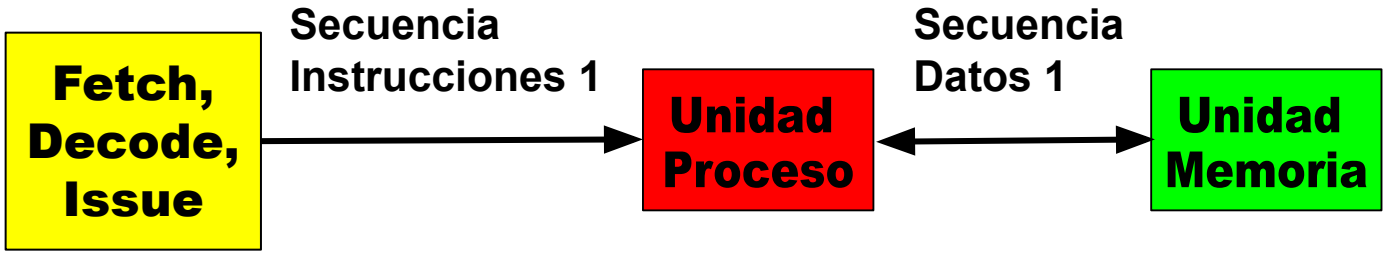
Unidad 6:
Arquitecturas paralelas

Tipos de sistemas paralelos
Clasificación de Flynn

- **SISD**: Single Instruction, Single Data
- **SIMD**: Single Instruction, Multiple Data
- **MISD**: Multiple Instruction, Single Data
- **MIMD**: Multiple Instruction, Multiple Data

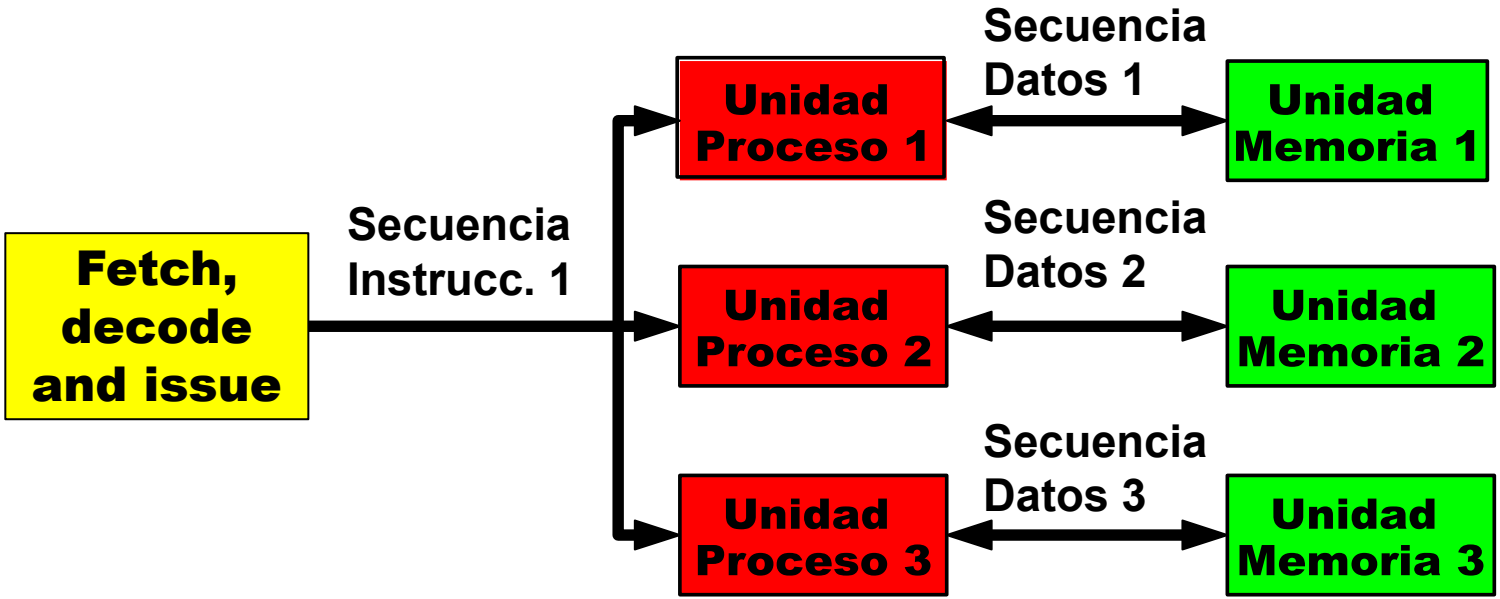
Tipos de sistemas paralelos
Clasificación de Flynn

- **Single Instruction, Single Data (SISD): Una única secuencia de instrucciones y una única fuente de datos.**
  - **Implementación: Una única unidad de ejecución operando sobre datos en una única memoria.**
  - **Ejemplos: Arquitectura Von Newman; procesador único**



Tipos de sistemas paralelos
Clasificación de Flynn

- **Single Instruction, Multiple Data (SIMD): Una única secuencia de instrucciones y múltiples fuentes de datos.**



Nota: Las unidades de memoria pueden ser una única memoria compartida, accediendo cada unidad de procesamiento a diferentes bloques de dicha memoria.
Figura basada en Computer Organization and Architecture 9th Edition William Stallings

Tipos de sistemas paralelos
Clasificación de Flynn

Single Instruction, Multiple Data (SIMD):

- **Distintas implementaciones:**
  - **Una única etapa de búsqueda (fetch), decodificación y envío de instrucciones (issue) enviando esas instrucciones a varias unidades de ejecución (UE). Cada UE ejecuta la instrucción pero sobre distintos datos.**
  - **Varias ALUs que ejecutan la misma instrucción sobre distintos datos (ALUs superescalar).**
  - **Pipeline sobre la ALU: Una ALU ejecuta la instrucción sobre diferentes datos “casi” al mismo tiempo (pipeline de datos).**
- **Ejemplos:**
  - **Procesadores matriciales o vectoriales**
  - **Extensiones MMX o SSE de x86**
  - **GPUs, GPGPUs**

Tipos de sistemas paralelos
Clasificación de Flynn

- **Multiple instruction, single data (MISD): múltiples secuencias de instrucciones y una única fuente de datos.**

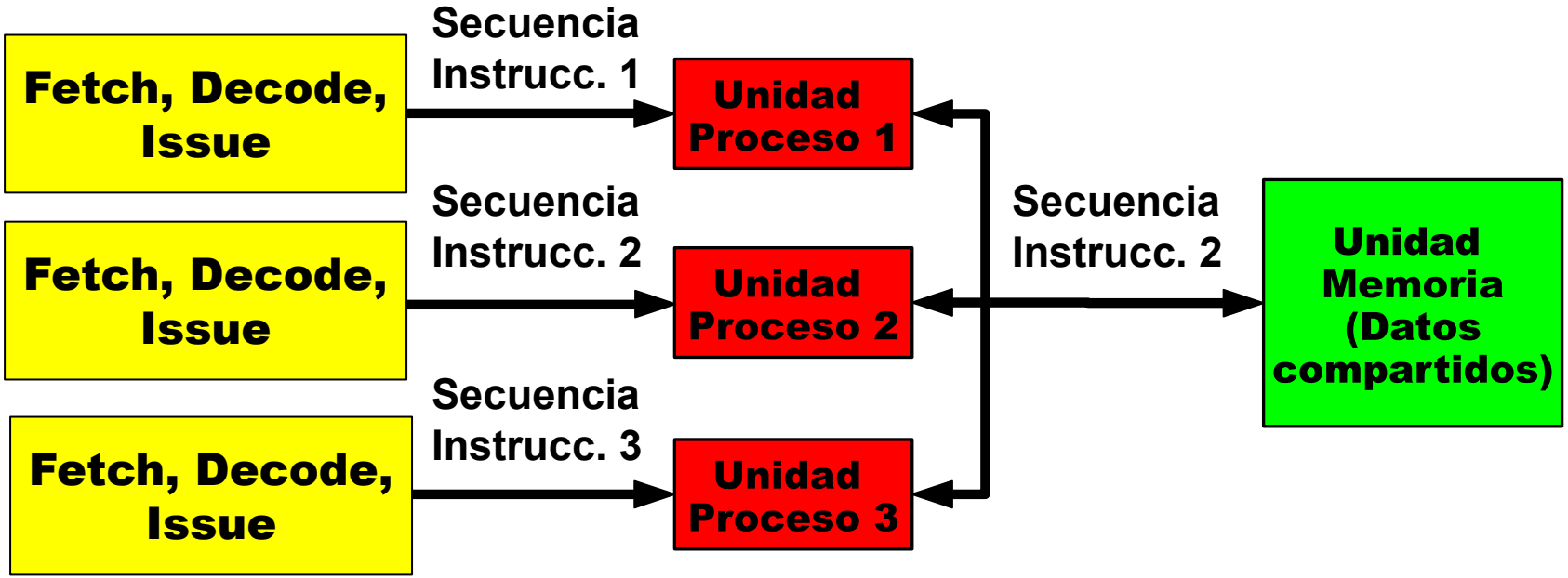


Figura basada en Computer Organization and Architecture 9th Edition William Stallings

Clasificación de Flynn

- **Multiple instruction, single data (MISD): múltiples secuencias de instrucciones y una única fuente de datos.**
- **Implementación: múltiples unidad de ejecución operando sobre datos en una única memoria**
- **Ejemplos Aplicación:**
  - **Computación redundante (sistemas que requieren alta tolerancia a fallas) [1].**
  - **Búsqueda de patrones [2, 3].**

[1] Spector, A.; Gifford, D. (September 1984). "The space shuttle primary computer system". Communications of the ACM. 27 (9): 872–900. doi:10.1145/358234.358246

[2] Arne Halaas, Børge Svingen, Magnar Nedland, Pål Sætrom, Ola Snøve, Jr., and Olaf René Birkeland. 2004. A recursive MISD architecture for pattern matching. IEEE Trans. Very Large Scale Integr. Syst. 12, 7 (July 2004), 727-734.

[3] Algunos autores afirman que éstas arquitecturas no se ajustan a ninguna de los tipos de Flynn

Tipos de sistemas paralelos
Clasificación de Flynn

- **Multiple instruction, multiple data (MIMD): múltiples secuencias de instrucciones y múltiples fuentes de datos.**

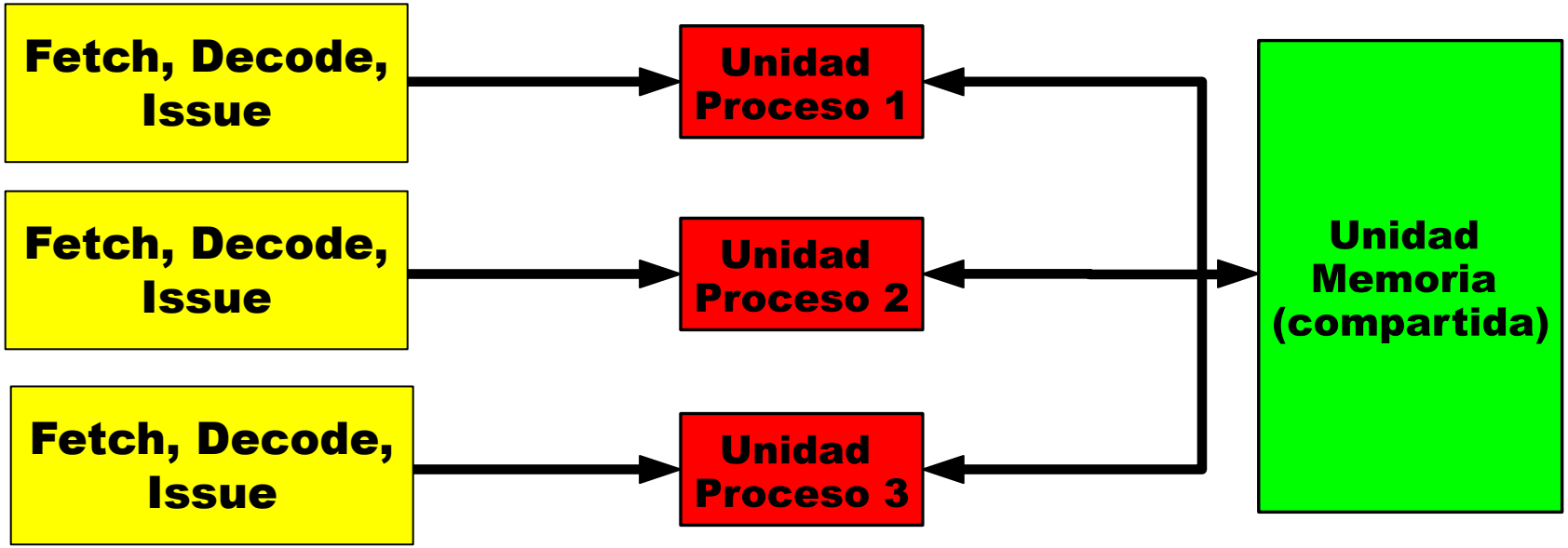


Figura basada en Computer Organization and Architecture 9th Edition William Stallings

Tipos de sistemas paralelos
Clasificación de Flynn

- **Multiple instruction, multiple data (MIMD):**

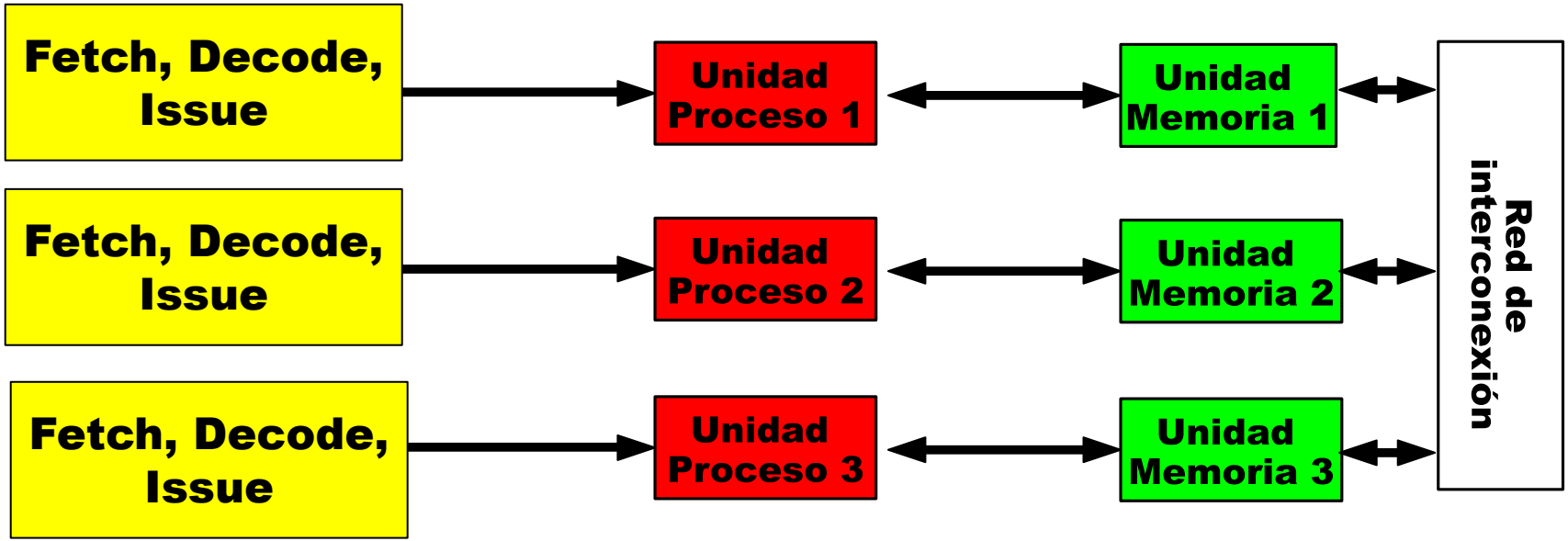


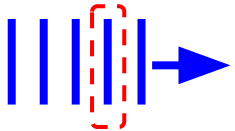
Figura basada en Computer Organization and Architecture 9th Edition William Stallings

Tipos de sistemas paralelos
Clasificación de Flynn

- **Multiple instruction, multiple data (MIMD):**
  - **Implementación: Múltiples unidades de ejecución ejecutando secuencias de instrucciones diferentes sobre diferentes datos.**
  - **Ejemplos: SMPs (Symmetric MultiProcessor), clusters, y sistemas NUMA.**

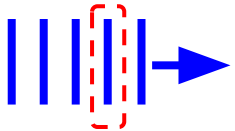
Nota 1: Un procesador superescalar NO es un ejemplo de MIMD, ya que, si bien puede ejecutar varias instrucciones al mismo tiempo, estas instrucciones forman parte de la misma secuencia de instrucciones. Puede ser SISD o SIMD (SIMD si soporta instrucciones vectoriales como MMX o SSE)
Nota 2: En los procesadores multithreading, todos los hilos que se ejecutan al mismo tiempo deben ser parte del mismo proceso, por lo que deben compartir recursos (como memoria caché, espacio de memoria, etc.). Por lo que puede suponerse que los hilos provienen del mismo "programa". Por lo tanto, estos procesadores son del tipo SISD o SIMD.

Multithreading (Multi hilo) - Motivación
Dependencia de datos reales



1	F	D	E	E	E	W					
2	F	D				E	W				
3		F	D				E	W			
4		F	D				E	W			

Retrasar el Pipeline

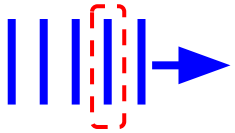


1	F	D	E	E	E	W					
3	F	D	E	E	W						
4		F	D	E	E	W					
2		F	D			E	E	W			

Fuera de orden

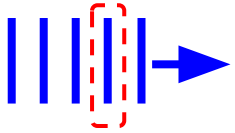
El procesador fuera de orden ayuda a mitigar el problema, pero no es una solución ideal

Multithreading (Multi hilo) - Motivación
Conflictos de recursos



1	F	D	E	E	W						
2	F	D			E	E	W				
3		F	D				E	W			
4		F	D				E	W			

Retrasar el Pipeline

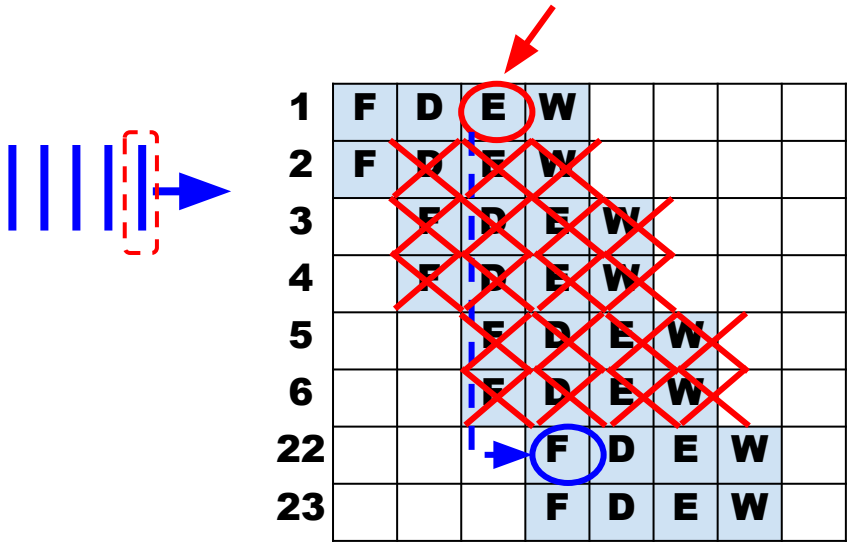


1	F	D	E	E	W						
3	F	D	E	W							
4		F	D	E	E	W					
2		F	D		E	E	W				

Fuera de orden

El procesador fuera de orden ayuda a mitigar el problema, pero no es una solución ideal

Multithreading (Multi hilo) - Motivación
Saltos condicionales

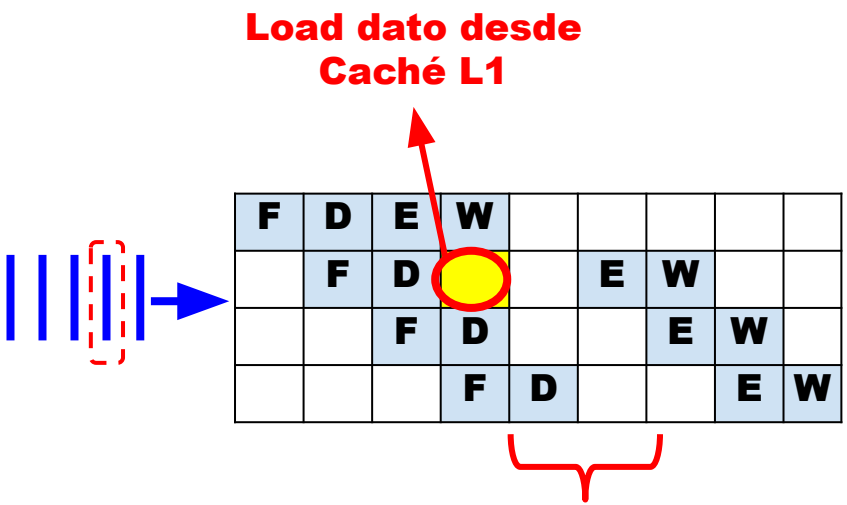


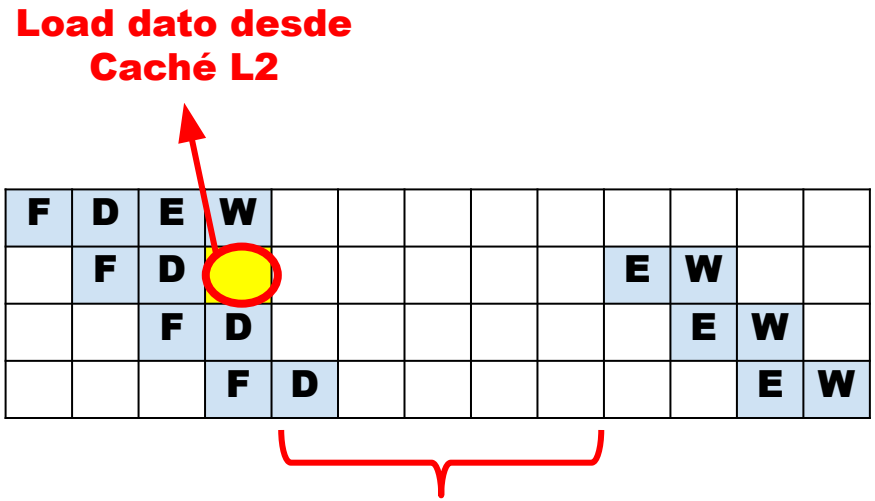
Posibles soluciones

- Branch history table
- Ejecución especulativa
- Salto demorado

Ninguna es una solución perfecta.

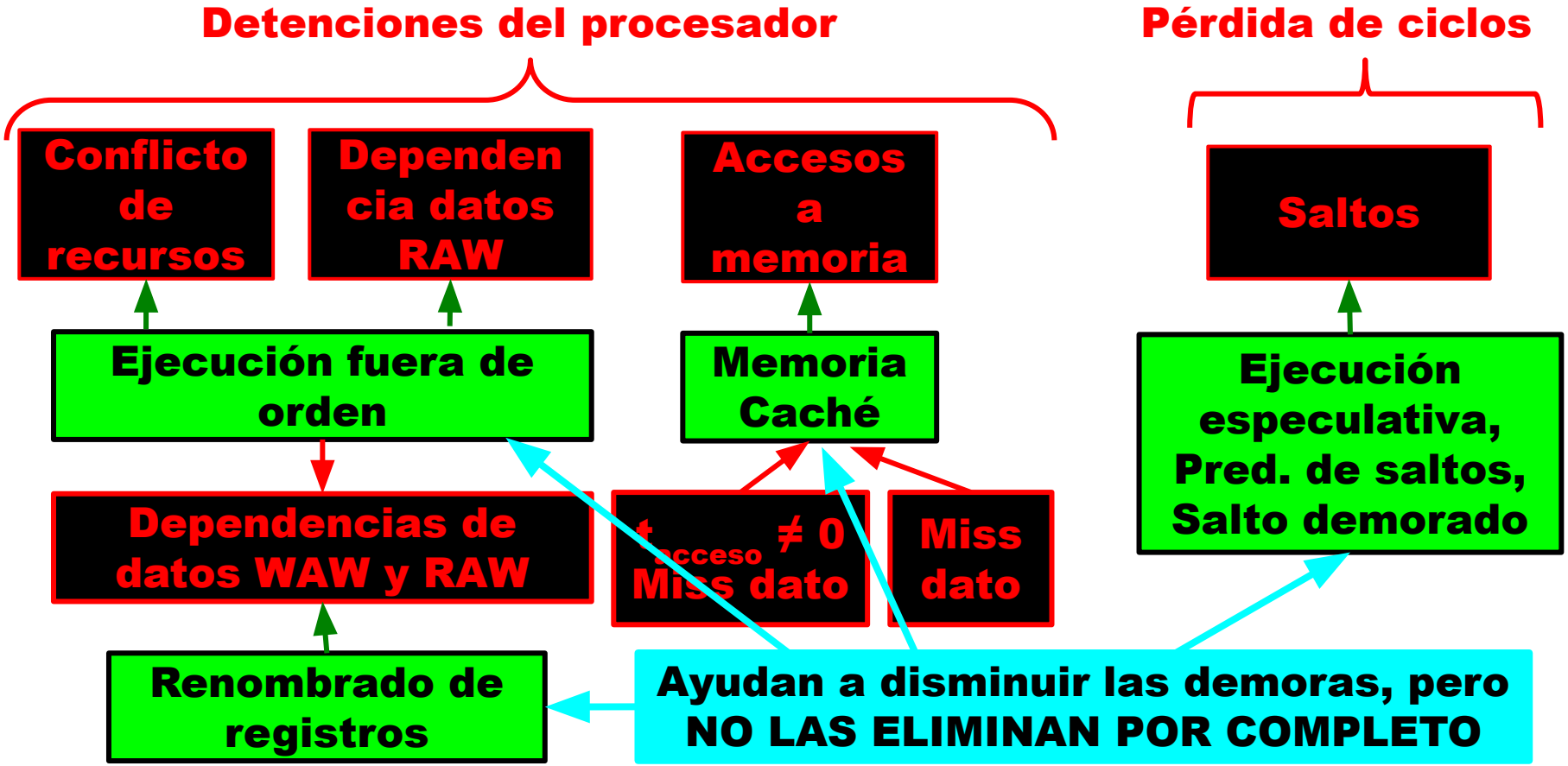
Multithreading (Multi hilo)
Acceso a memoria





Multithreading (Multi hilo)

Problemas pipeline y superescalar (speedup real<speedup ideal)



Multithreading (Multi hilo) - Motivación

Ley de Amdahl para procesadores de N vías

Ganancia en velocidad comparando un procesador de N vías (puede ejecutar N instrucciones al mismo tiempo) contra un procesador de una vía

$$\text{speedup} = \frac{\text{Tiempo de ejecución sin paralelismo}}{\text{Tiempo de ejecución con procesador de N vías}}$$

$$\text{speedup} = \frac{T}{T*f + \frac{T*(1-f)}{N}} = \frac{1}{f + \frac{(1-f)}{N}}$$

f: fracción de código NO paralelizable. 0<f<1
(1-f): fracción de código paralelizable
T: tiempo de ejecución procesador con N=1.
N=número de vías

Ejemplo basado en Computer Organization and Architecture 9th Edition William Stallings, pag. 56

Multithreading (Multi hilo) - Motivación

$$\text{Speedup} = \frac{1}{f + \frac{(1-f)}{N}}$$

$$\text{Lim Speedup} = 1/f$$

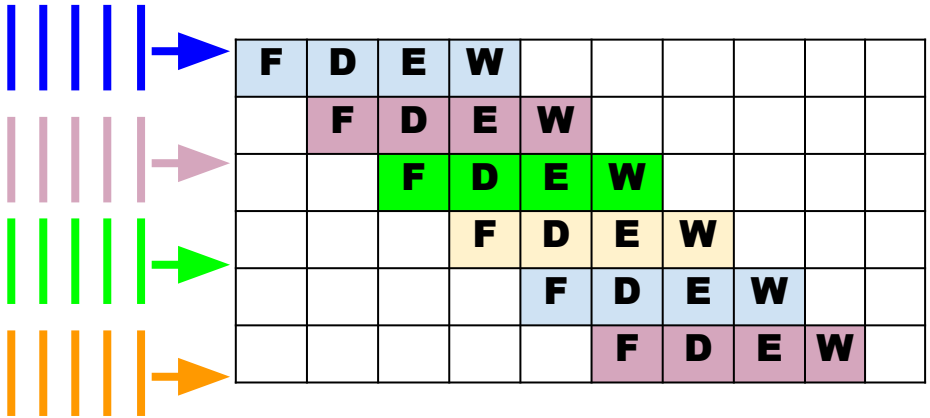
$N \to \infty$

N \ f	0.05	0.2	0,75
1	1	1	1
2	1,9	1,67	1,14
8	5,93	3,33	1,28
32	12,5	4,44	1,32

El speedup no solo depende de la arquitectura (N), sino también de la secuencia de instrucciones (f).

Multithreading (Multi hilo)

Multithreading: Dividir una secuencia de instrucciones en múltiples secuencias, llamadas threads (hilos), idealmente independientes entre si.

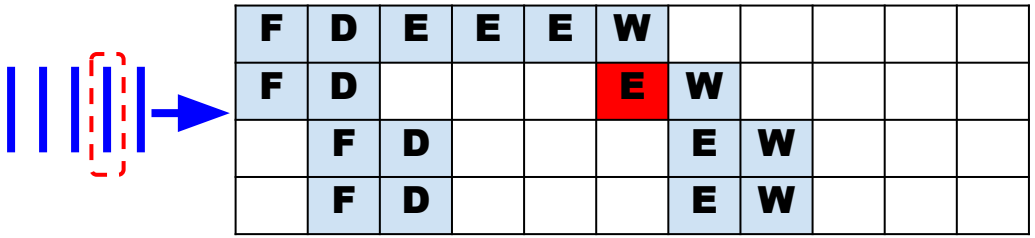


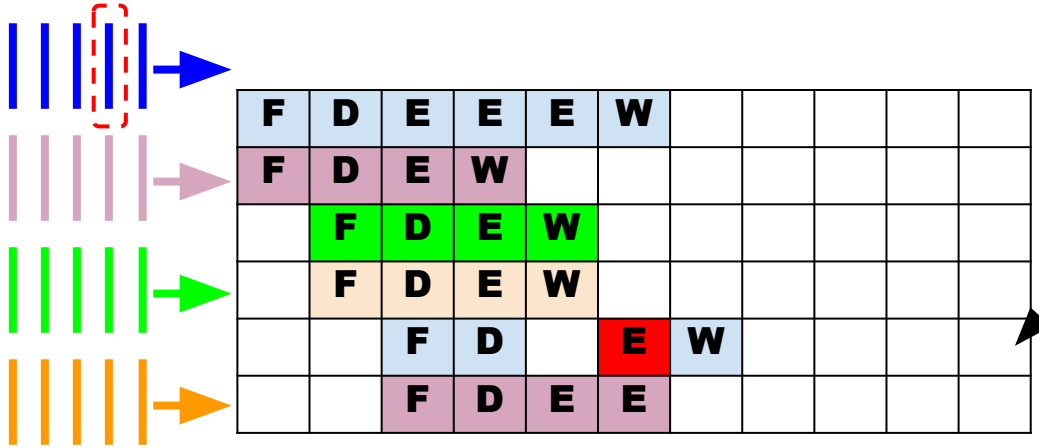
Observar que instrucciones de una misma secuencia están separadas. Esto ayuda a resolver varios de los problemas vistos

Los primeros procesadores en incluir la técnica Multithreading fueron los Intel Xeon MP (año 2002)

Multithreading (Multi hilo)

Ejemplo de funcionamiento con procesador superescalar de dos vías, con problema de dependencia de datos RAW

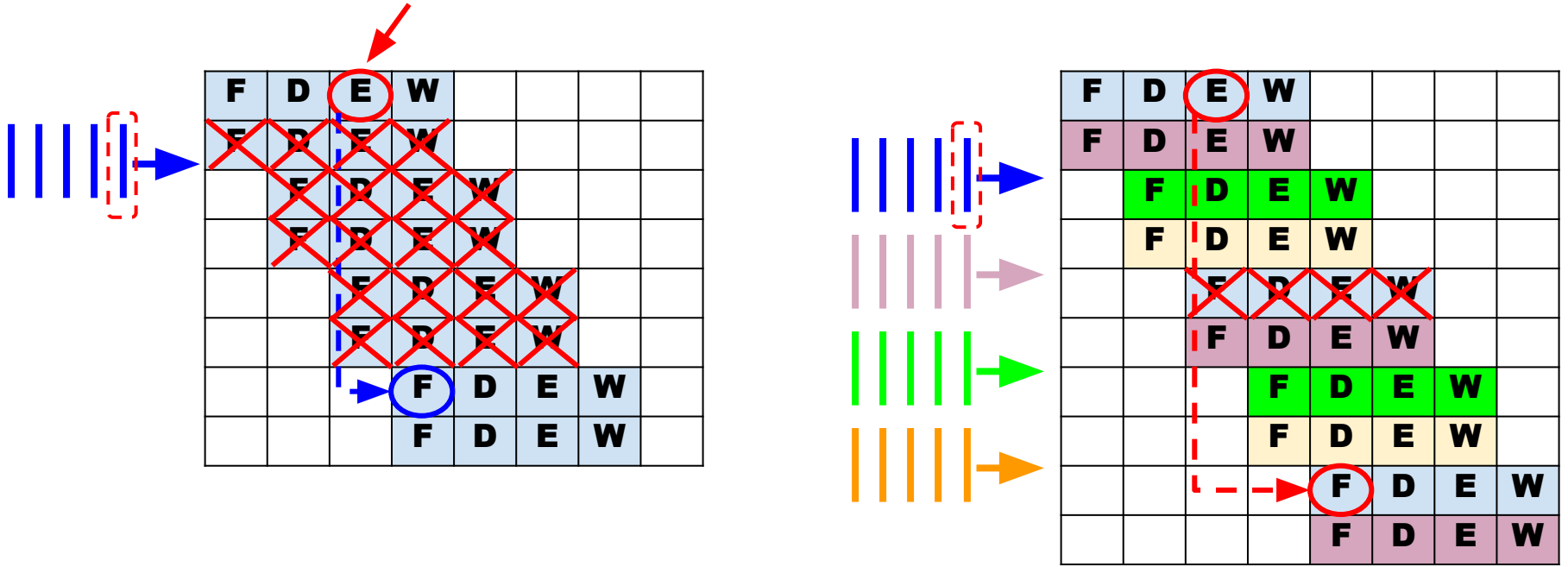




Si se combina con ejecución fuera de orden, podría no perderse ningún ciclo, ya que en lugar de ejecutar la segunda instrucción del primer hilo (que tiene problemas de dependencia), podría ejecutarse la tercera antes de la segunda.

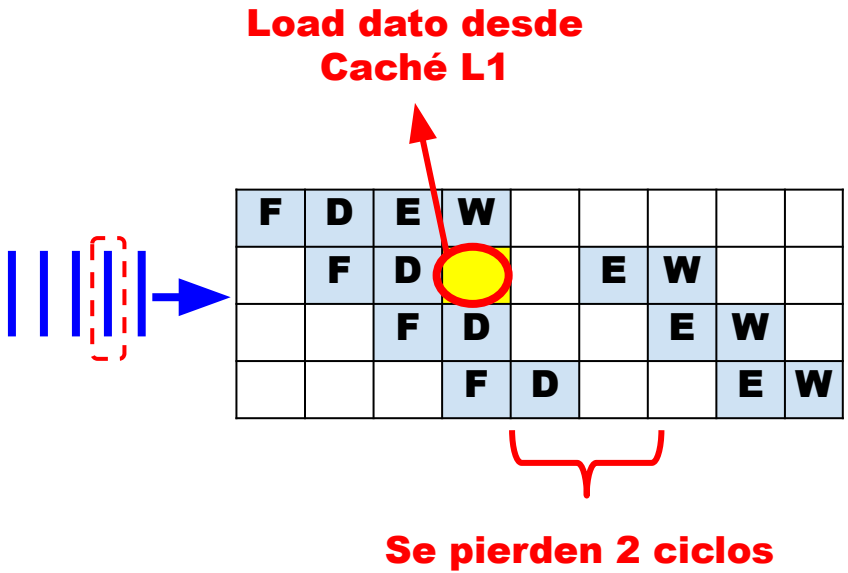
Multithreading (Multi hilo)

Ejemplo de funcionamiento con procesador superescalar de 2 vías, con problema de conflicto estructural (salto condicional)



Multithreading (Multi hilo)

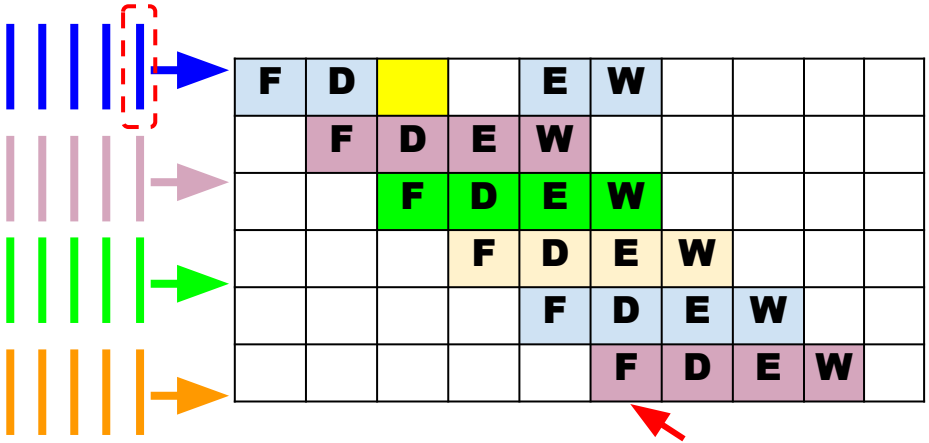
Problema: Detención del pipeline por lectura de datos en memoria
Solución 1: Separar las instrucciones con problemas deteniendo el pipeline hasta que se solucione el problema -> Pérdida de tiempo



Multithreading (Multi hilo)

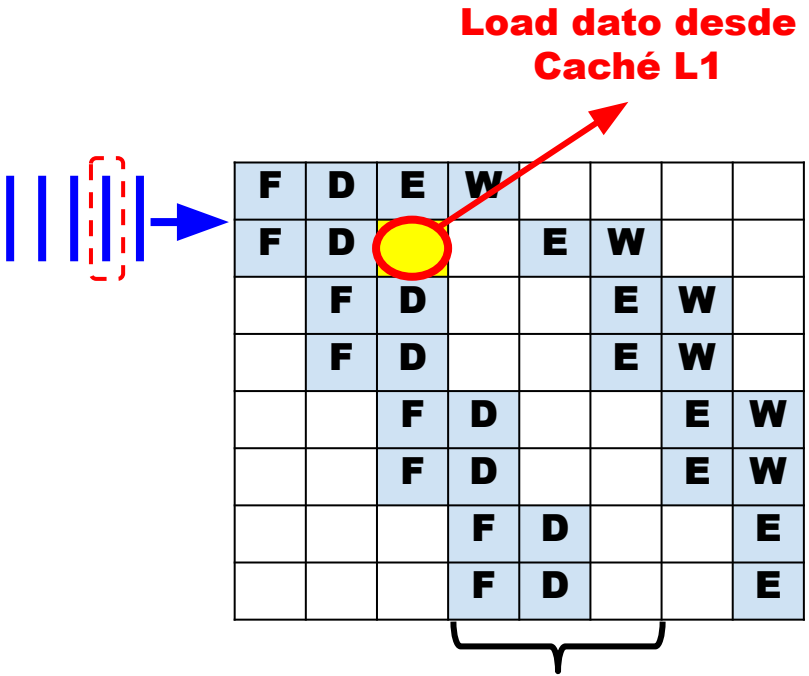
Solución 2: Separar las instrucciones con problemas insertando entre ellas instrucciones independientes -> Multithreading

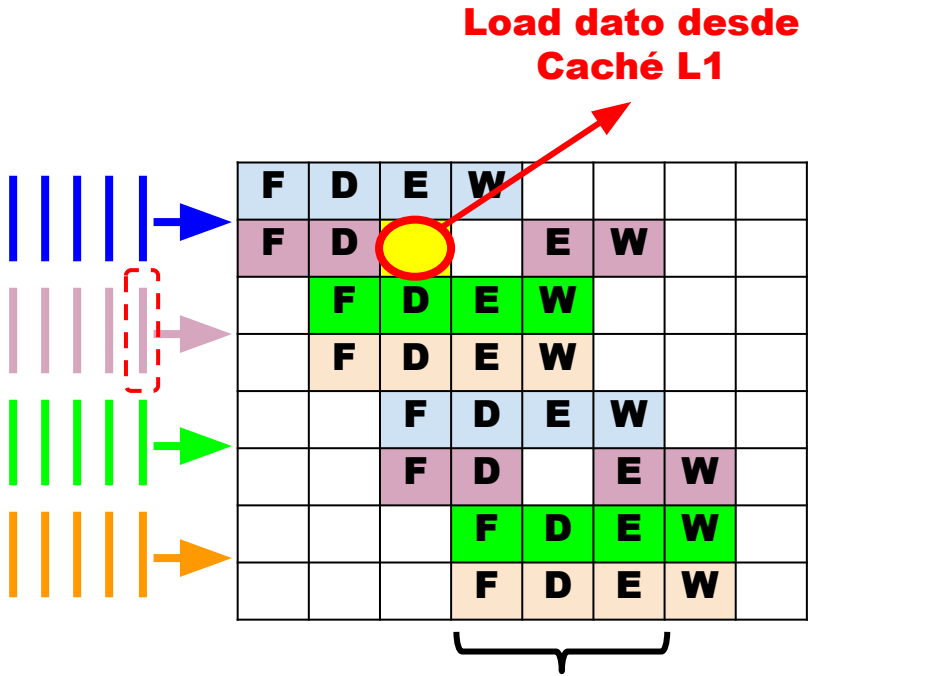
Multithreading: Dividir una secuencia de instrucciones en múltiples secuencias, llamadas threads (hilos), idealmente independientes entre si.



Multithreading (Multi hilo)

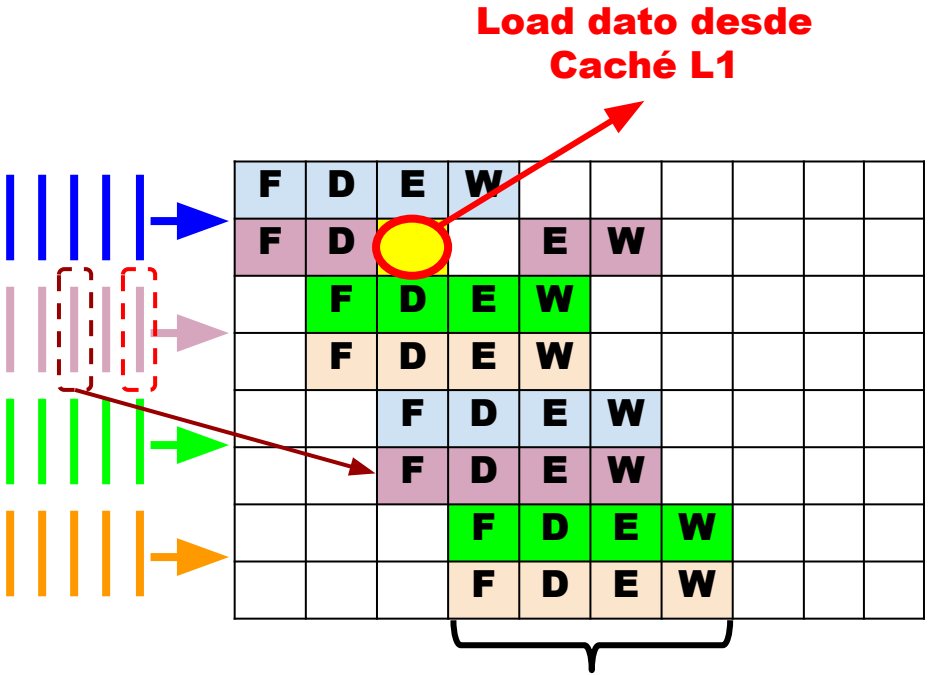
Ejemplo de funcionamiento con procesador superescalar de dos vías, acceso a memoria Caché L1 y 4 hilos





Multithreading (Multi hilo)

Ejemplo de funcionamiento con procesador superescalar de dos vías, acceso a memoria Caché L1, 4 hilos y ejecución fuera de orden.



Notas:

1) En el ciclo 5 tenemos 3 instrucciones en ejecución. Esto es posible si las instrucciones utilizan unidades de ejecución diferentes.

2) En el ciclo 6 hay tres instrucciones escribiendo resultados en memoria, esto es posible si al menos una no escribe en memoria (es decir, uno de los resultados queda en registros).

3) En un procesador fuera de orden, si la instrucción 2 del hilo 2 depende de la instrucción 1, el procesador podría ejecutar la instrucción 3 antes de la 2.

Multithreading (Multi hilo)

- **Proceso: Instancia de un programa ejecutándose en una computadora que posee:**
  - **Recursos propios asignados por el SO (información de estado):**
    - **Espacio de direcciones (variables, buffers, etc.).**
    - **Archivos abiertos.**
    - **Información de autentificación**
    - **Contenido de memoria caché.**
    - **Registros (incluido el contador de programa).**
  - **El proceso puede requerir la ejecución de varios hilos.**

Multithreading (Multi hilo)

- **Process Switching (cambio de proceso): Cambio de un proceso a otro (empleado en sistemas multitasking). Requiere almacenar información de un proceso y cargar la del otro, incluyendo:**
  - **Actualizar la memoria caché.**
  - **Archivos abiertos.**
  - **Información de autentificación**
  - **Registros (incluido el contador de programa).**
  - **Agenda (scheduling) de ejecución.**
  - **Etc.**
- **Objetivo**
  - **Multitasking.**
  - **Paralelismo a nivel de procesos.**

Multithreading (Multi hilo)

- **Thread (hilo): Secuencia de instrucciones contenido en un proceso (unidad de trabajo, subproceso, proceso liviano, etc.).**
  - **Los hilos de un proceso son idealmente independientes**
    - **Pueden ejecutarse en paralelo.**
  - **Recursos (información de estado) limitados al estado del procesador, lo cual incluye:**
    - **Contenido de los registros (incluyendo el contador de programa y puntero de pila).**
  - **Los threads de un proceso comparten los recursos del proceso, entre estos:**
    - **Espacio de direcciones (variables globales, buffers, etc.), por lo tanto, contenido de la caché.**
    - **Archivos abiertos**
    - **Autentificación**

Multithreading (Multi hilo)

- **Thread switching: Cambio de un thread a otro dentro del mismo proceso.**
  - **Incluye cambiar solo la información de un thread a otro, pero no la información del proceso padre de ambos thread. Lo cual incluye cambiar:**
    - **Contenido registros (incluyendo contador de programa y puntero de pila)**
  - **Un thread switching NO requiere cambiar:**
    - **Espacio de direcciones (Incluyendo el contenido de memoria caché).**
    - **Archivos abiertos**
    - **Información de autentificación**
  - **Menos costoso que un process switching**

Multithreading (Multi hilo)
Ventaja del uso de hilos

- **Facilidad para escribir un programa, ya que:**
  - **El problema puede dividirse en sub-problemas, resolviendo cada sub-problema con diferentes hilos.**
  - **Permiten crear aplicaciones paralelas con memoria compartida.**
  - **Pueden usarse llamadas al sistema bloqueantes.**
  - **Ejemplos de aplicaciones que son más eficientes gracias a los hilos: aplicaciones cliente-servidor, sistemas distribuidos, interfaces de usuario, aplicaciones productor-consumidor, etc.**

- **Hacer uso más eficiente de arquitecturas que permiten la ejecución de instrucciones en paralelo.**

Esta es la ventaja que nos interesa en esta materia

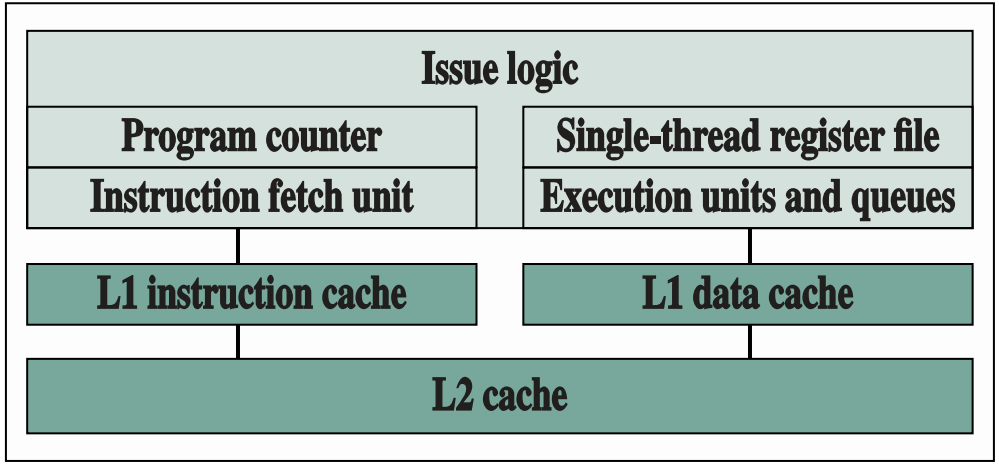
Multithreading (Multi hilo)

- **Procesador Multithreading: El procesador posee soporte de hardware para ejecutar dos o más hilos al mismo tiempo.**
  - **Duplicación de los registros de propósito general, contador de programa y puntero de pila**
  - **La etapa de fetching se realiza en base a thread**
  - **Aplicación de técnicas de paralelización y optimización en base a thread (branch prediction, register renaming, superscalar)**

Paralelismo a nivel de instrucción: Superescalar

F	D	E1	E2	W		
F	D	E1	E2	W		
	F	D	E1	E2	W	
	F	D	E1	E2	W	
		F	D	E1	E2	W
		F	D	E1	E2	W

Procesador 1 hilo

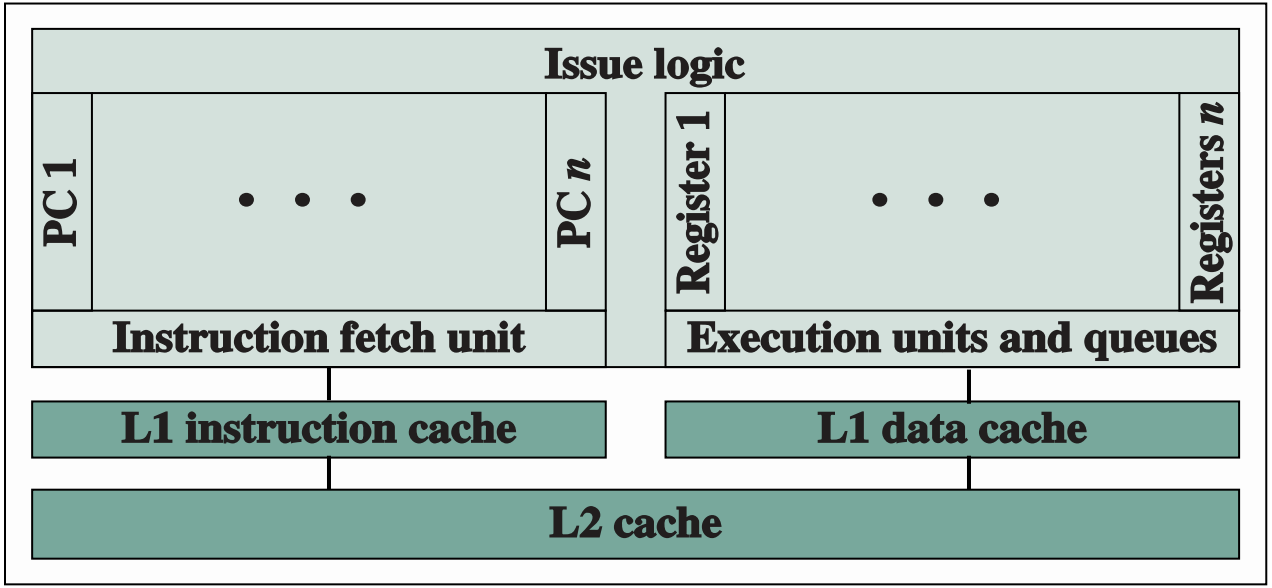


Speedup ideal = N*k

Figura obtenida de “Computer Organization and Architecture”, William Stallings, 10º edición, página 658

Multithreading sobre procesador de un núcleo

Procesador multihilos n hilos



Tipos de Multithreads a Nivel del Procesador

- **Multithreading intercalado [1]: Un procesador ejecuta varios thread, cambiando de uno a otro en cada ciclo de reloj.**
- **Blocked multithreading [2]: Cada thread se ejecuta hasta que se ocurre un evento que introduce una demora (dependencia de datos, acceso a memoria, saltos, etc.).**

[1] También llamado Interleaved multithreading o multithreading de grano fino
[2] También llamado multithreading de grano grueso

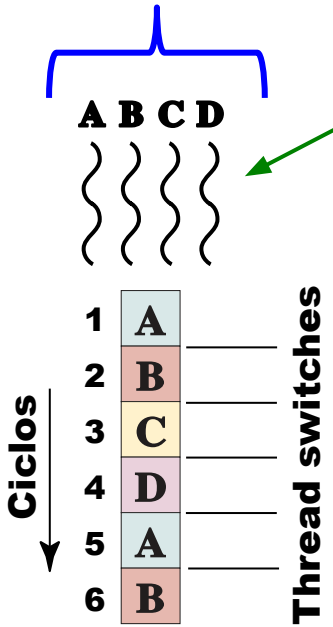
Tipos de Multithreads a Nivel del Procesador

- **Multithreading simultáneo (SMT): Las instrucciones se envían simultáneamente desde múltiples hilos a diferentes unidades de ejecución de un procesador superescalar. Los thread se ejecutan en paralelo en distintas unidades de ejecución**
  - **Ejemplo: Tecnología Intel Hyper-Threading**
- **Chip multiprocesadores (multinúcleos): Existen varios procesadores en un solo chip. Cada procesador maneja hilos separados.**
  - **Ejemplo: Procesadores ARM multinúcleo**

Multithreading intercalado (Interleaved multithreading)
Programa multihilo sobre Procesador escalar (un hilo)

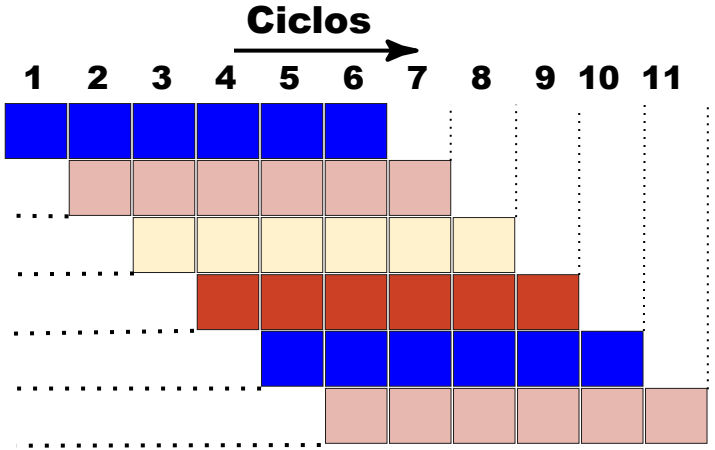
El SO ve 4 núcleos lógicos

Se necesitan 4 juegos de registros (uno por cada hilo). Un núcleo físico, 4 núcleos lógicos.
(https://www.intel.la/content/www/xl/es/products/processors/core/i7-processors/i7-9750h.html)



Secuencia entrada a Unidades de ejecución

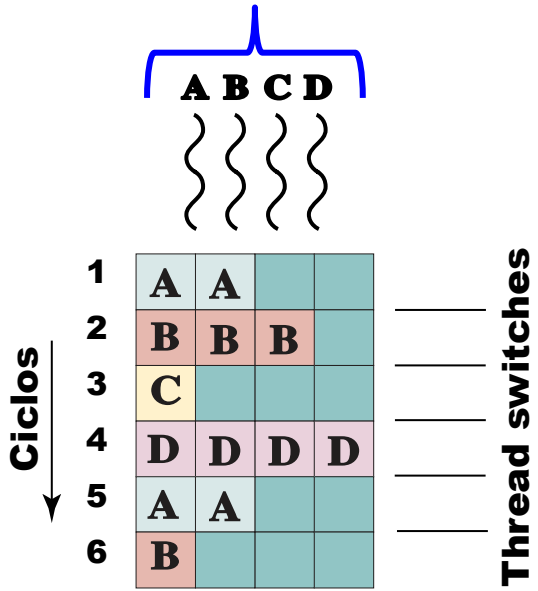
Instrucción Hilo A
Instrucción Hilo B
Instrucción Hilo C
Instrucción Hilo D
Instrucción Hilo A
Instrucción Hilo B



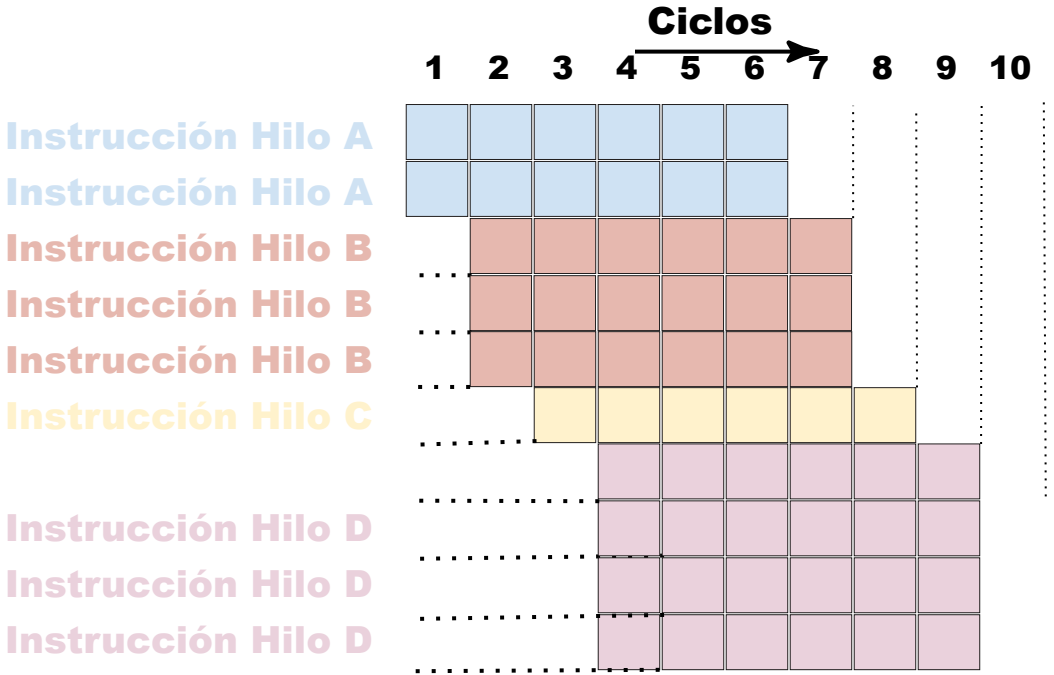
Pipeline

Multitheading intercalado (interleaved multithreading) sobre Procesador superescalar de 4 vías

El SO ve 4 núcleos lógicos



Secuencia entrada a Unidades de ejecución



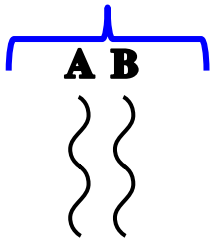
...
...

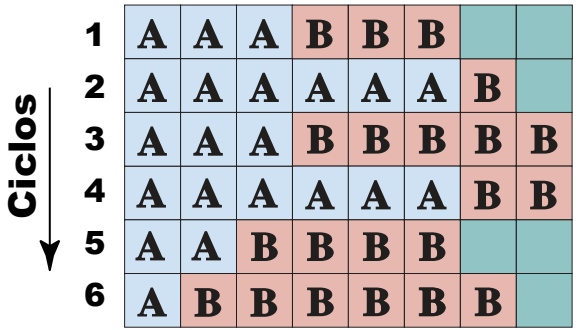
Pipeline

Ejemplo 1: Multithreading simultáneo de dos hilos sobre procesador superescalar de 8 vías fuera de orden

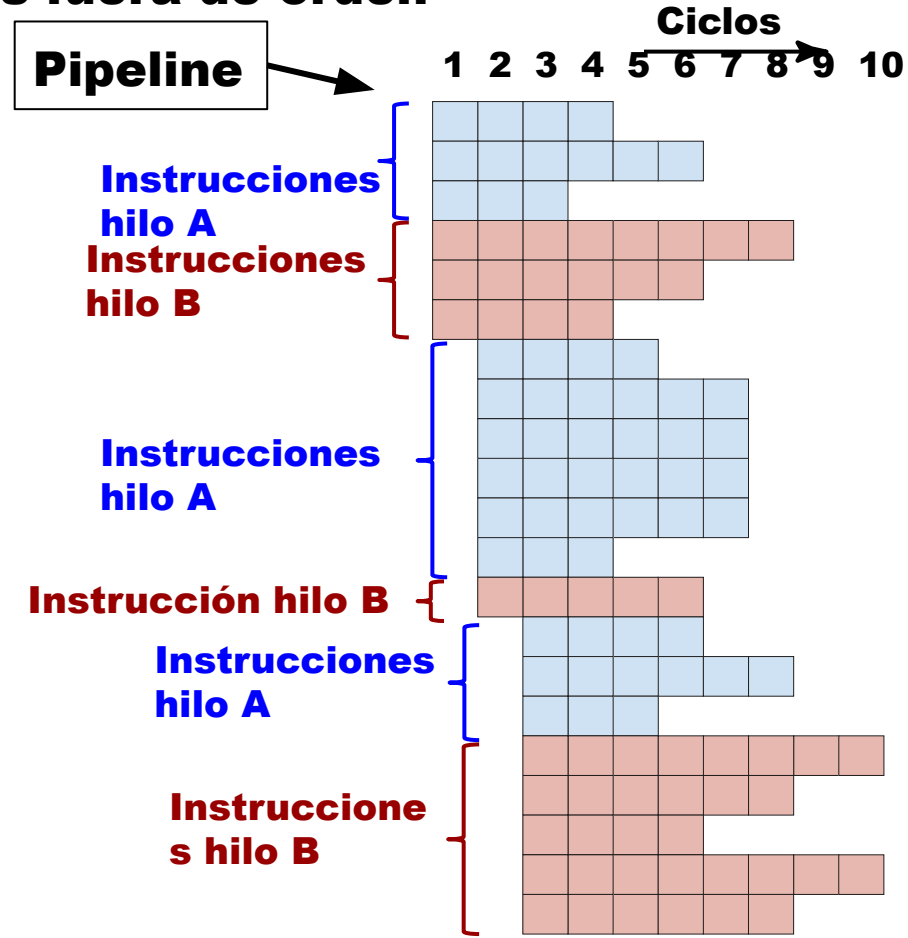
La microarquitectura Kaby Lake (i3, i5 e i7 año 2017) de Intel responde a este diseño (Hyperthreading)

El SO ve 2 núcleos lógicos





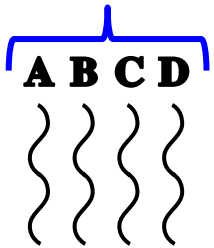
Secuencia de entrada a unidades de ejecución

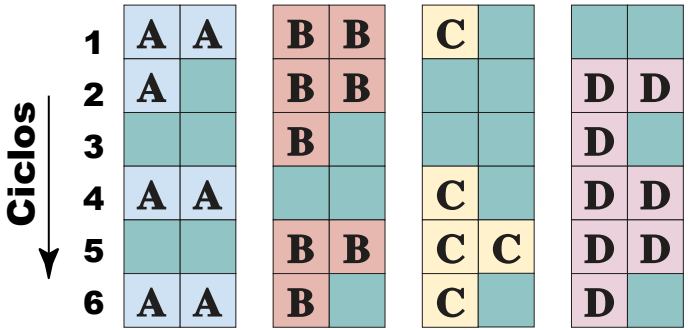


Ejemplo 2: Multithreading simultáneo sobre arquitectura multiprocesadores

La microarquitectura del ARM Cortex A75 (año 2017) responde a este diseño

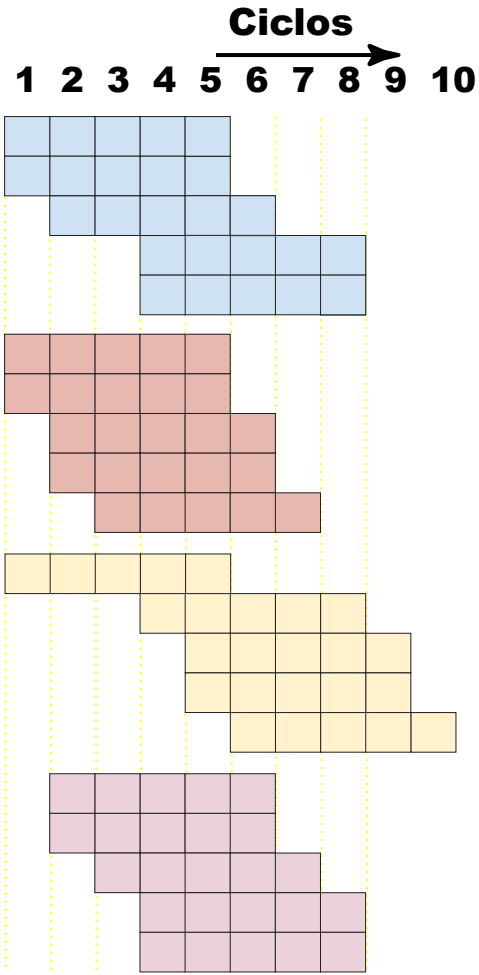
El SO ve 4 núcleos lógicos



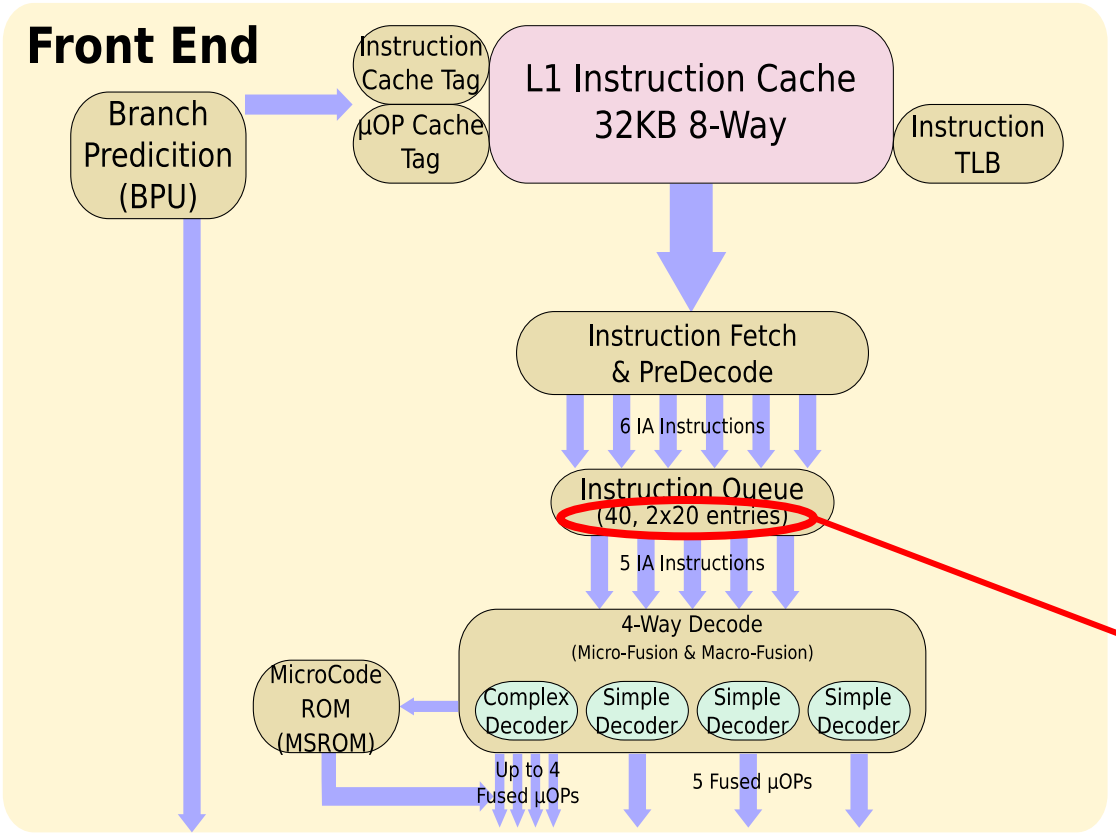


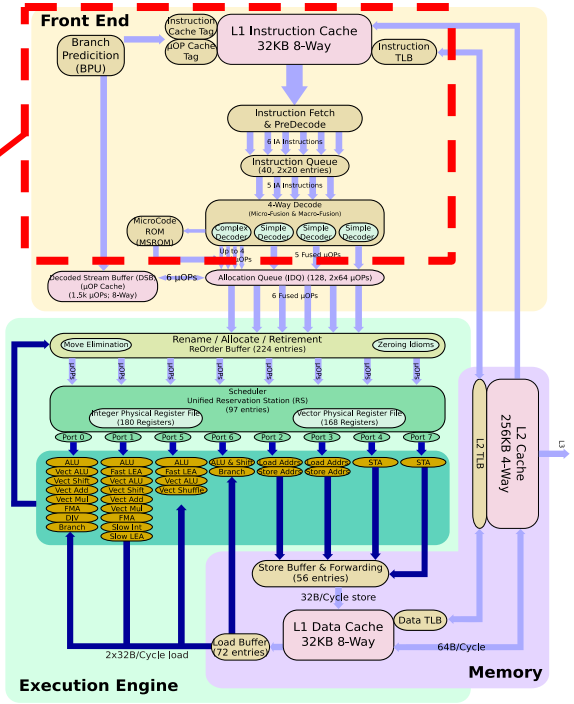
Secuencia de entrada a unidades de ejecución

Pipeline



Ejemplo1: microarquitectura Kaby Lake (Intel)

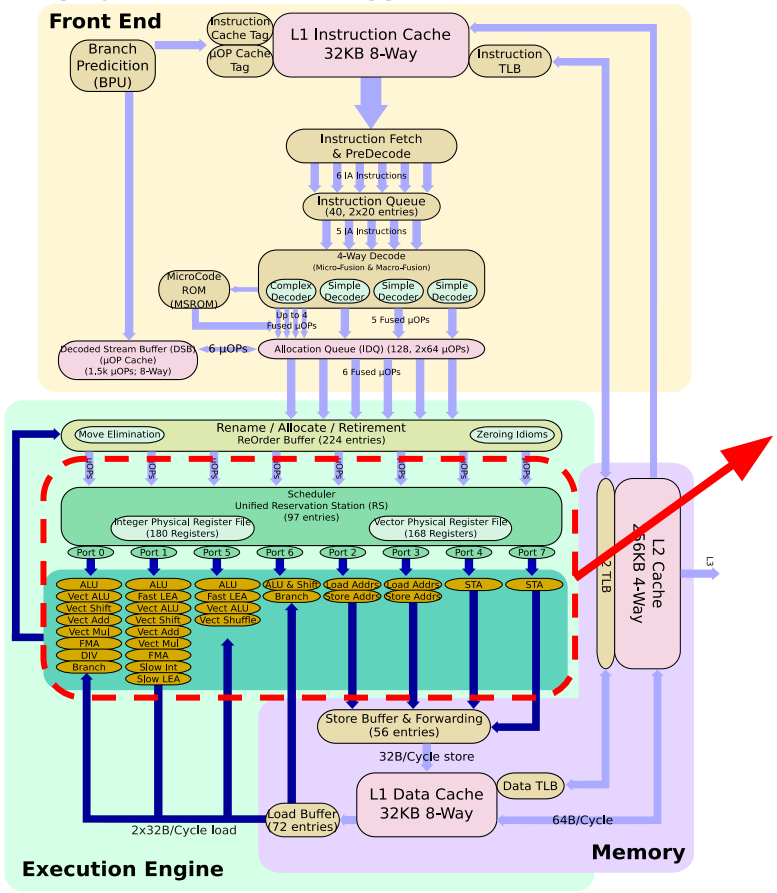




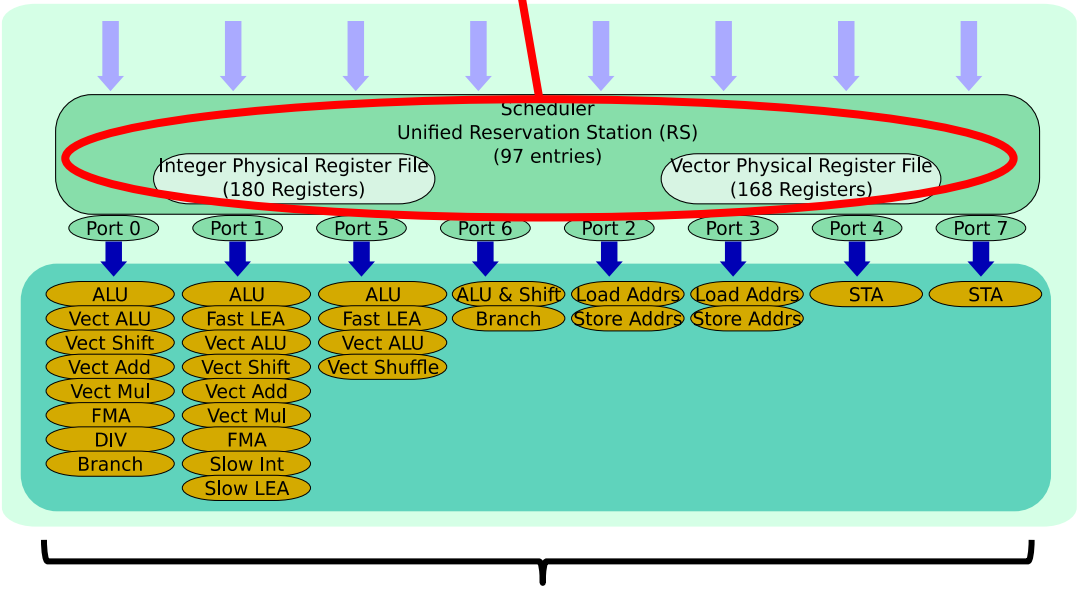
Buffer que almacena hasta 20 instrucciones por hilo. 2 hilos

Ejemplo Procesadores Multithreading: Kaby Lake

(Hyperthreading)



µoperaciones de ambos hilos en espera de que se liberen las unidades de ejecución que cada µoperación necesita



8 vías de ejecución

Multithreading - Desventajas

- **Se complica el manejo de señales, como excepciones o interrupciones.**
- **Se necesitan mecanismos de sincronización para evitar que dos hilos accedan de manera simultánea al mismo recurso, los cuales se verán en asignaturas posteriores.**

SMP (symmetric multiprocessor)

- Varios procesadores de capacidad comparable
  - Todos los procesadores pueden realizar las mismas tareas [1]
- Comparten la misma memoria, dispositivos de entrada-salida y SO.
  - El tiempo de acceso a memoria el mismo para todos los procesadores.
  - Conocidos como “sistema de procesadores estrechamente acoplados” [2].
- El acceso de los procesos a los núcleos es controlado por el sistema operativo:
  - Asigna la ejecución de los procesos o hilos sobre cada procesador (scheduling).
  - Debe hacer que estas tareas sean transparentes al usuario

[1] Esto significa que es indistinto ejecutar una tarea en uno u otro procesador.

[2] En contraste con los Clusters, denominados sistemas de procesadores débilmente acoplados, ya que NO comparten la memoria ni dispositivos de entrada salida, y ejecutan cada una su sistema operativo.

SMP (symmetric multiprocessor) - ventajas

- **Ejecución de tareas (que pueden estar formadas por varios procesos), procesos o hilos en paralelo.**
- **Disponibilidad: Si un procesador falla, la computadora puede seguir trabajando (con menor performance). Solo válido para procesadores simétricos.**

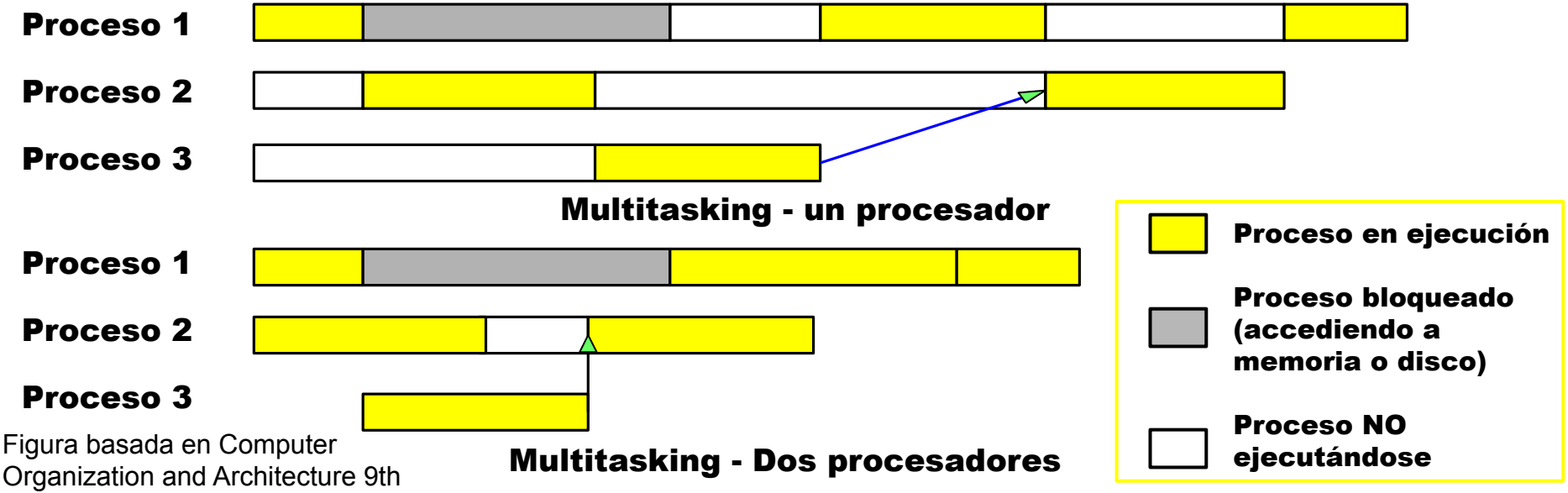


Figura basada en Computer Organization and Architecture 9th Edition William Stallings

SMP

- **Los procesadore se conectan a un bus**
- **Los procesadores pueden acceder a bloques separados de memoria (en algunas arquitecturas simultáneamente).**
- **Cada procesador posee su**
  - **ALU**
  - **Registros**
  - **Unidades de ejecución (si es superescalar).**
  - **Uno o más niveles de memoria Caché.**

SMP - bus compartido en tiempo

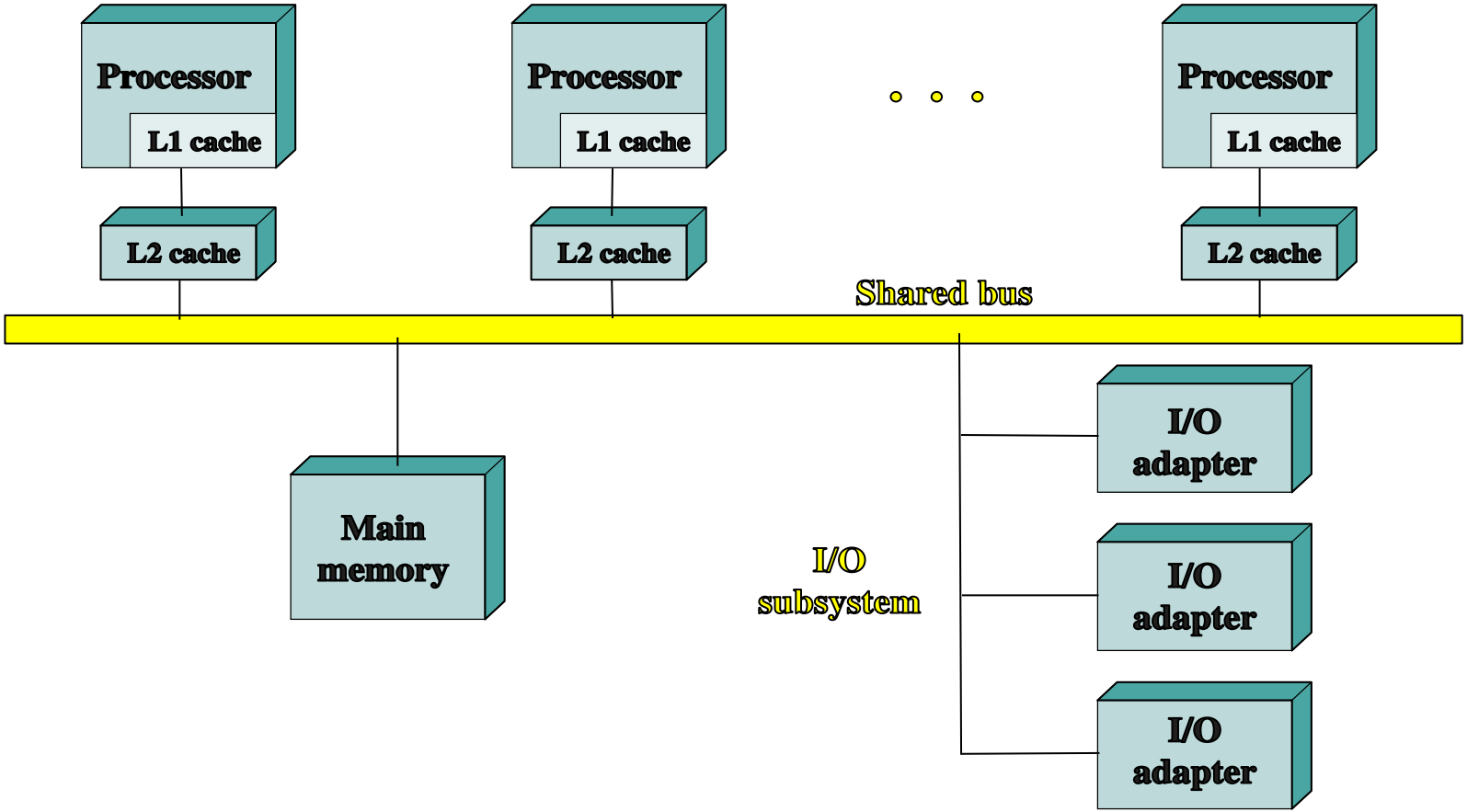
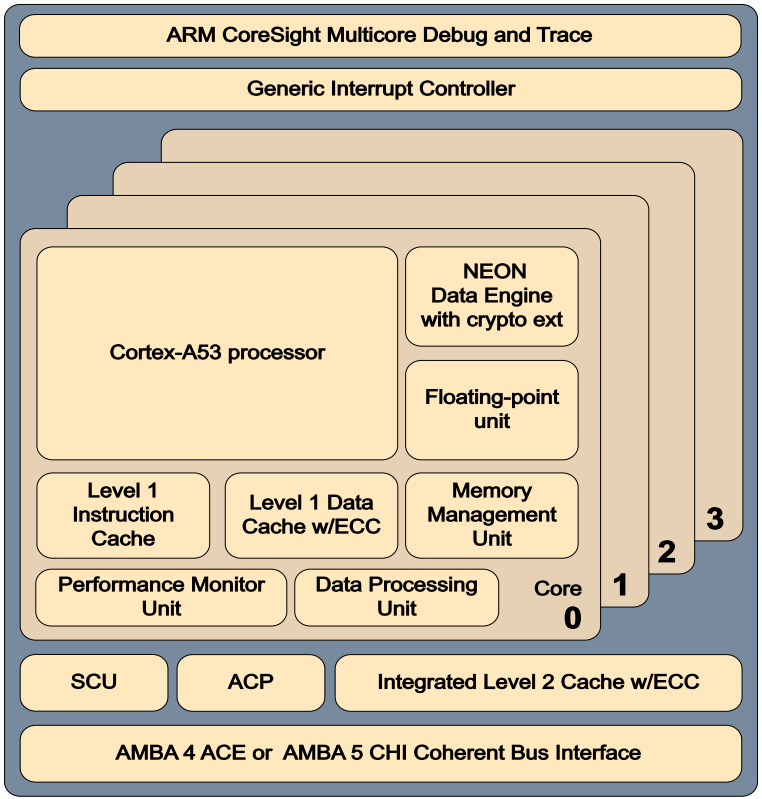


Figura basada en Computer Organization and Architecture 9th Edition William Stallings

SMP - bus compartido en tiempo

- **Estructura, buses (control, dirección y datos) e interfaces iguales a un sistema de procesador único.**
- **Desventajas:**
  - **Todas las comunicaciones con la memoria pasan por el bus: el bus es un cuello de botella.**
    - **Se agregan memorias cachés a cada procesador para disminuir los accesos a memoria.**
    - **Problema de coherencia con la memoria caché: Un dato puede estar en varias cachés, si un procesador lo modifica, debe actualizarse en todas las cachés y la memoria principal.**

SMP Ejemplos: ARM núcleos simétricos



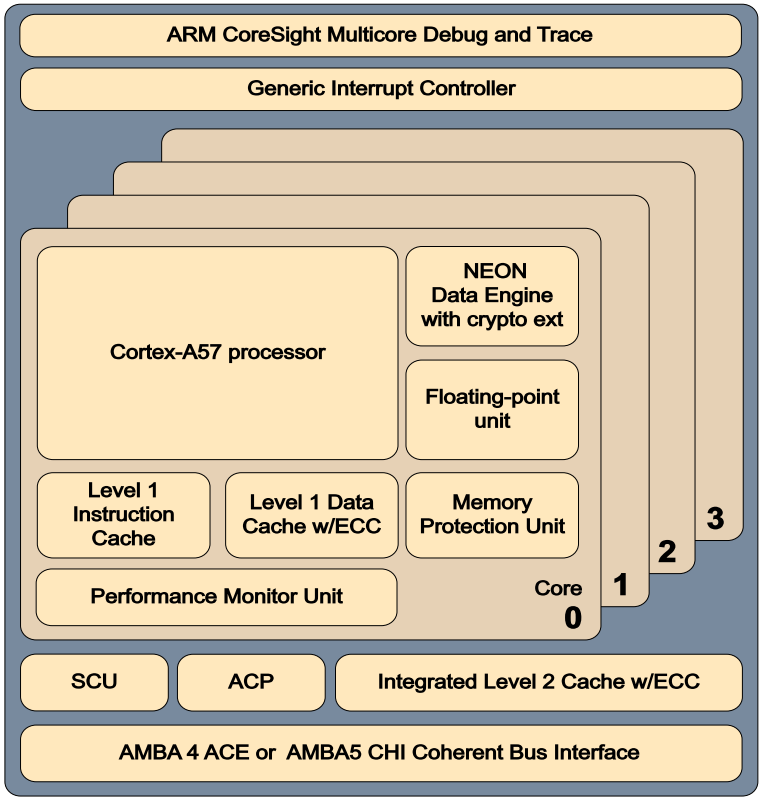


Figura obtenida de ARM Cortex -A Series, Programmer's Guide for ARMv8-A, Version: 1.0, pag 2-7 y 2-8

SMP Ejemplos: Intel i7 - 990x

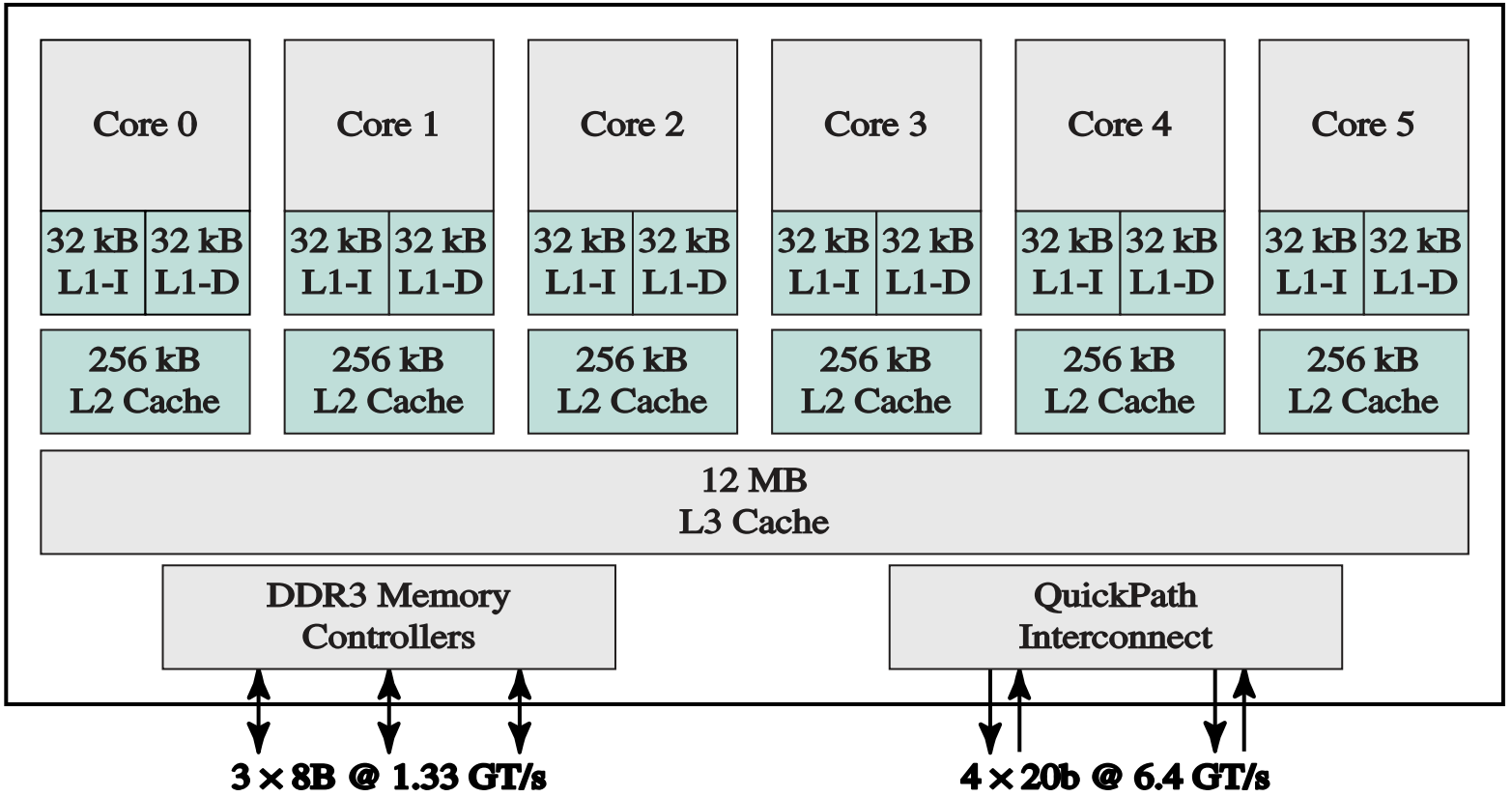
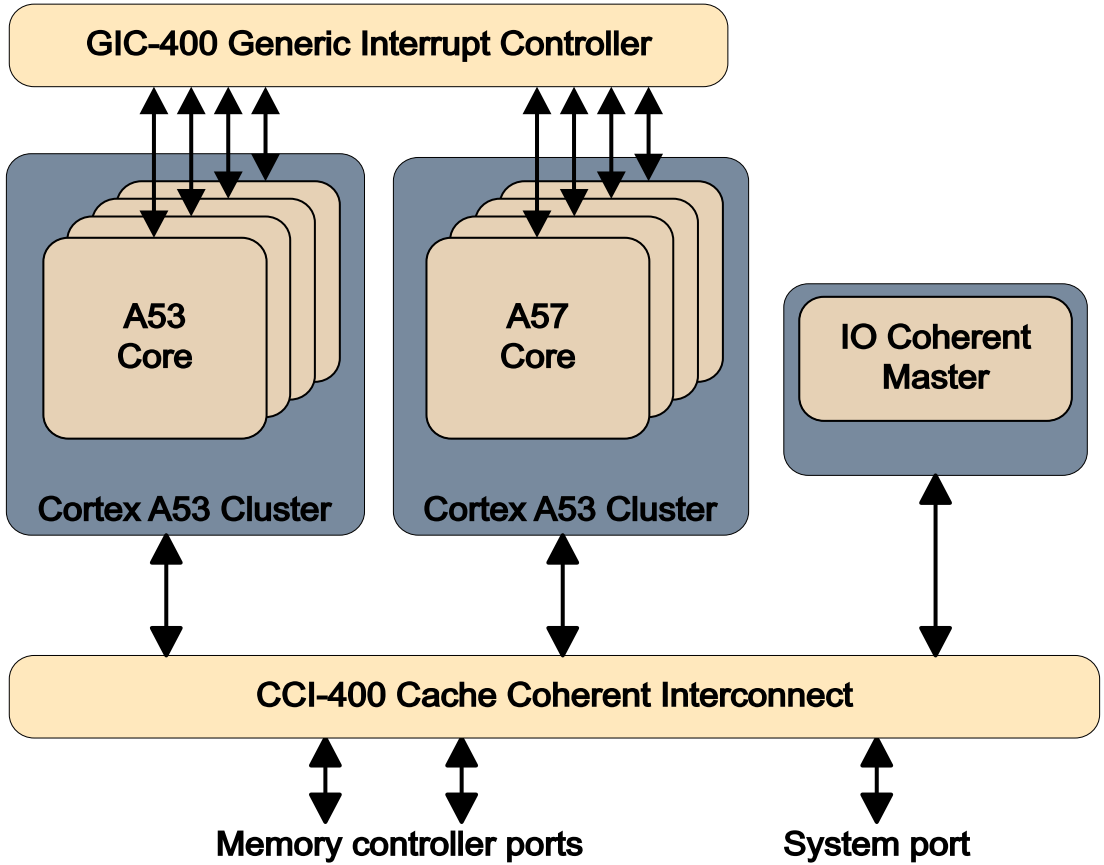


Figura obtenida de Operating Systems Internals and Design Principles - 8 ed - William Stallings

Heterogeneous multi-processing (HMP)

- **2 tipos:**
  - **Igual set de instrucciones.**
  - **Diferente set de instrucciones (los veremos en Arquitecturas Distribuidas).**



Fuente: ARM Cortex -A Series, Programmer's Guide for ARMv8-A, Version: 1.0. Pag 233

Heterogeneous multi-processing (HMP)

- **Procesadores heterogéneos con igual set de instrucciones: Los programas pueden ejecutarse indistintamente en un núcleo u otro, pero con diferente velocidad.**
- **Aplicación típica: Optimizar el consumo de energía.**
  - **Procesadores de elevado poder de procesamiento y por tanto con elevado consumo de energía (juegos, video, etc.).**
  - **Procesadores de bajo poder de procesamiento pero bajo consumo de energía (eventos del SO, tareas siempre conectadas, etc.).**
  - **Sistema que permite pasar aplicaciones de un núcleo a otro según la necesidad.**

Fuente: ARM Cortex -A Series, Programmer’s Guide for ARMv8-A, Version: 1.0

Heterogeneous multi-processing (HMP)

- **Ejemplo: Arquitectura big.LITTLE con procesadores Cortex A57 (big) y Cortex A53 (LITTLE).**
- **Dos modos de funcionamiento:**
  - **Migración: Las tareas migran entre procesadores (o clusters) según la carga de trabajo.**
    - **Migración de clusters: trabajan todos los procesadores big o todos los LITTLE.**
    - **Migración de CPU: Cada procesador big está emparejado con uno LITLEE. Solo uno puede usarse.**
  - **Programación global de tareas: El programador (parte del SO) conoce la capacidad de procesamiento necesaria de cada tarea (hilos), y las envía a los núcleos correspondientes.**

Fuente: ARM Cortex -A Series, Programmer's Guide for ARMv8-A, Version: 1.0

Cluster Computing

- **Límite de los sistemas multiprocesador:**
  - **Es difícil construir chip con muchos núcleos.**
  - **Un sistema multiprocesador no escala fácilmente.**
  - **más procesadores -> más problemas de coherencia de Caché.**
  - **Alternativa: Cluster Computing.**
- **Cluster Computing: Grupo de computadoras completas [1] interconectadas (no comparten memoria), trabajando juntas como una sola computadora.**
- **Ventajas:**
  - **Escalabilidad**
    - **Puede tener gran tamaño (cientos o miles de nodos)**
    - **Agregar nuevos nodos es fácil**
  - **Disponibilidad: Si un nodo falla, el cluster sigue funcionando**
  - **Precio: Poder de procesamiento similar a computadora de elevado poder a similar precio.**

[1] Computadora completa significa que poseen procesador, memoria, SO, etc., y todo lo necesario para funcionar como computadoras por si solas

Cluster Computing - Cluster Computer Architecture

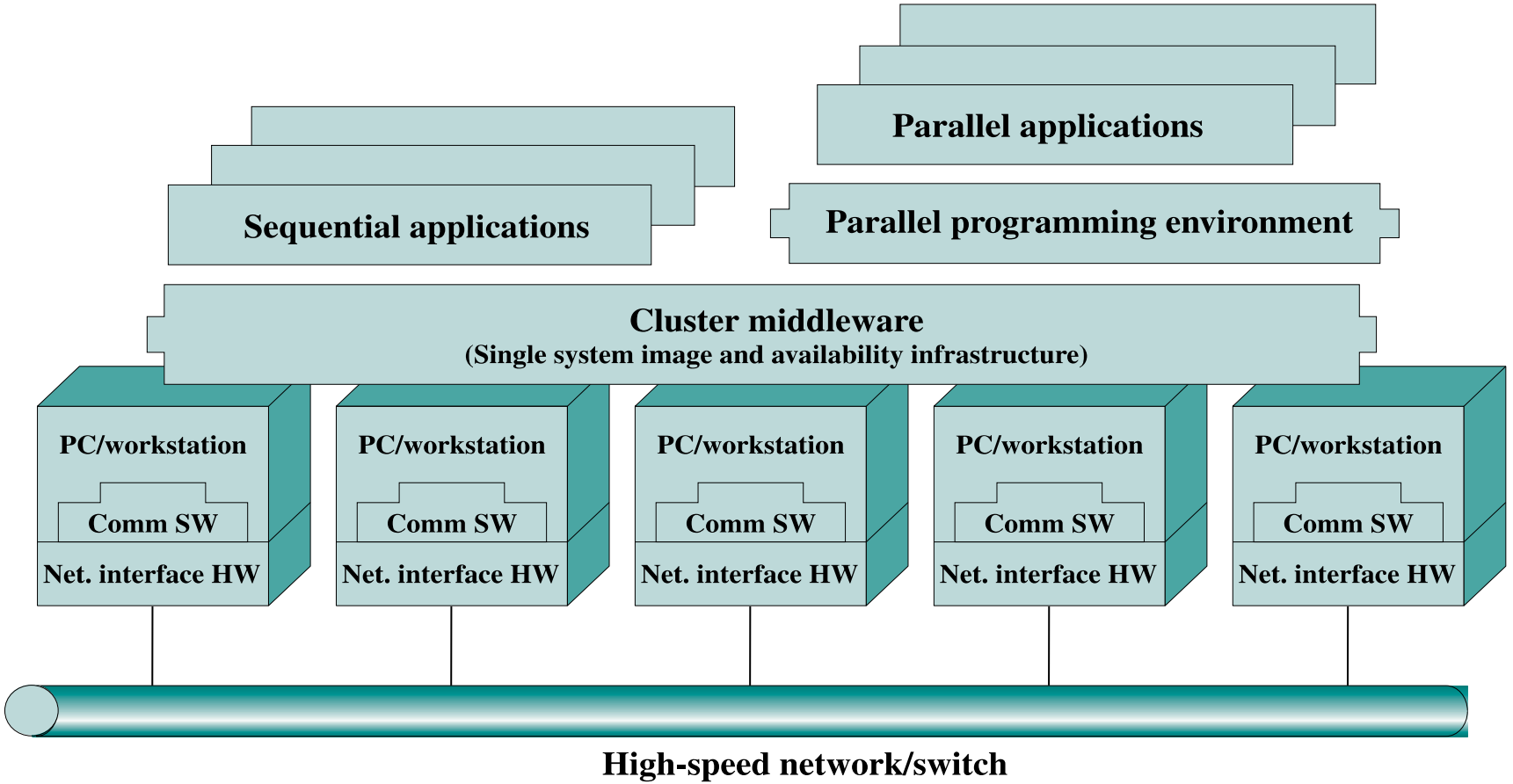
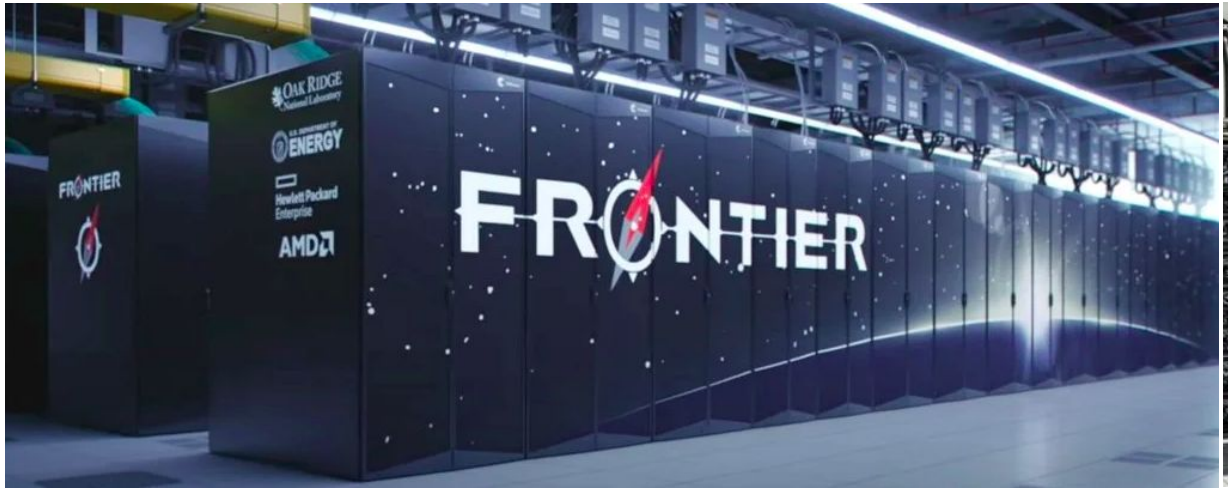


Figura basada en Computer Organization and Architecture 9th Edition William Stallings

Cluster Computing
Ejemplo: Frontier



- 1° cluster más potente del mundo.
- 1.1 HexaFLOPS.
- 21 Mw
- 8.7 millones de núcleos AMD 3rd Generation EPYC 2GHz
- Estados Unidos
- Red de interconexión: HPE Slingshot-11

[1] Datos que varían frecuentemente
Fuente: https://www.top500.org/

Cluster Computing
Ejemplo: Fugaku

- 2° cluster más potente del mundo.
- 442 PetaFLOPS.
- 29 Mw
- 7.6 millones de núcleos A64FX 48C 2.2GHz. ARMv8.2-A (primer ARM de 64 bits)
- Japón
- 5.09 PB de memoria.
- Red de interconexión: Infiniba

[1] Datos que varían frecuentemente
Fuente: https://www.top500.org/

Cluster Computing
Ejemplo: LUMI

- 3° cluster más potente del mundo.
- 151 PetaFLOPS[1].
- 2.9 Mw[1]
- 1.1 millones de núcleos[1]
  - AMD Optimized 3rd Generation EPYC 64C 2GHz.
- Red de interconexión: Slingshot-11
- Finlandia (EuroHPC)

Cluster Computing
Ejemplo: Serafin

- Cluster más potente de argentina
- 156 Teraflops
- Centro de Cómputo de Alto Desempeño (CCAD), Córdoba
- 60 nodos.
- 3840 núcleos (120 procesadores AMD EPYC 7532).
- https://ccad.unc.edu.ar/equipamiento/cluster-serafin/

Cluster Computing
Ejemplo: TUPAC

- Antiguo Cluster más potente de argentina
- 48 Teraflops
- 72 TB de memoria.
- Polo TIC de Buenos Aires
- 4096 núcleos AMD Opteron
- Aplicaciones:
  - Modelos meteorológicos
  - Modelados estructurales
  - Modelos fluidodinámicos de vehículos espaciales y aeronaves en general
  - Simulaciones dinámica molecular

Cluster Computing
Ejemplo: TOKO

- Cluster más potente del oeste argentino
- 1 Teraflops
- ITIC y FCEN, UNCuyo
- 390 núcleos (procesadores AMD de diferentes modelos y GPGPUs NVidia)
- 743 GB de Memoria.
- http://toko.uncu.edu.ar/

Tecnología	Descripción	Ejemplos	Speedup ideal
Paralelismo a nivel de instrucción	**Enviar a las UE varias instrucciones al mismo tiempo**	**Superescalar VLIW**	**10**
Múltiples procesadores con Memoria compartida	**Varios procesadores accediendo a la misma memoria compartida**	**Multi procesadores simétricos (SMP) Multi procesadores heterogeneos [1]**	**N**
Múltiples computadoras fuertemente acopladas	**Varias computadoras completas comunicadas mediante redes de muy alta velocidad**	**Cluster computing Cloud Computing**	**10^6**
Múltiples computadoras débilmente acopladas	**Varias computadoras completas comunicadas mediante redes de alta velocidad (no necesario)**	**Cloud Computing Grid Computing**	**Sin dato [2]**

[1] big.LITTLE
[2] El objetivo no es la velocidad

Comandos Linux relacionados

- **cat /proc/(PID)/status: Arroja información sobre el proceso cuyo ID es PID (process ID), incluyendo: número de hilos, cambios de contexto, uso de memoria (parámetros vmXX), proceso padre (ppid) etc.**
  - **| grep parámetro: el agregado de este modificador permite consultar un parámetro específico de todos los que brinda status, por ejemplo: cat /proc/(PID)/status | grep Threads: muestra el número de hilos.**
  - **Para consultar el PID de un proceso, puede utilizarse la aplicación "Monitor del sistema (System monitor)"**
- **top: muestra información sobre los procesos en ejecución.**

Bibliografía

Libros:

- Computer Organization and Architecture; William Stallings; 2010. (Muy completo)
- Arquitectura de Computadoras; Patricia Quiroga; 1º edición; 2010; (capítulo 10)
- ARM Cortex -A Series, Programmer's Guide for ARMv8-A, Version: 1.0
- Arquitectura de Computadoras; Morris Mano; 2006
- Organización de computadoras; Andrew Tanenbaum; 2008

Páginas web:

- ARM Developer: https://developer.arm.com/ (sección architecture)

Especificaciones:

- ARM® Cortex®-A75 Core Technical Reference Manual (Revision: r2p0), ARM.(2016)