Tema 1: Generalidades sobre Arquitecturas paralelas y distribuidas.

Universidad de Cádiz

ÍNDICE

ÍNDICE

Introducción

- Objetivos
- Limitaciones físicas de la computación secuencial.
- Limitaciones físicas de la computación Paralela.
- Niveles de Paralelismo
- Formas Básicas de Paralelismo
- Formas Básicas de Paralelismo
  - Segmentación y Replica

Conceptos Básicos

Computador paralelo

Computación paralela

Programación paralela

Paralelismo

Introducción

Principal objetivo: Aumento del RENDIMIENTO.
Aumento de la capacidad para resolver problemas computacionales grandes

¿Cómo?

- División del trabajo en tareas mas pequeñas e independientes
- Asignación de las tareas a distintas unidades de proceso
- Resolución de las tareas en simultaneo.

Problemas:

- Sincronización de las tareas.
- Control de ejecución simultanea
- Conflictos debidos a dependencias

Introducción

Procesadores secuenciales

Mayor velocidad de las puertas lógicas

Alto grado de integración y miniaturización

¿Hasta cuando?

Limitaciones físicas de la computación secuencial

Limitaciones físicas de la computación paralela

Limitaciones físicas de la computación secuencial

- Velocidad de la luz : supone 30 cm/s

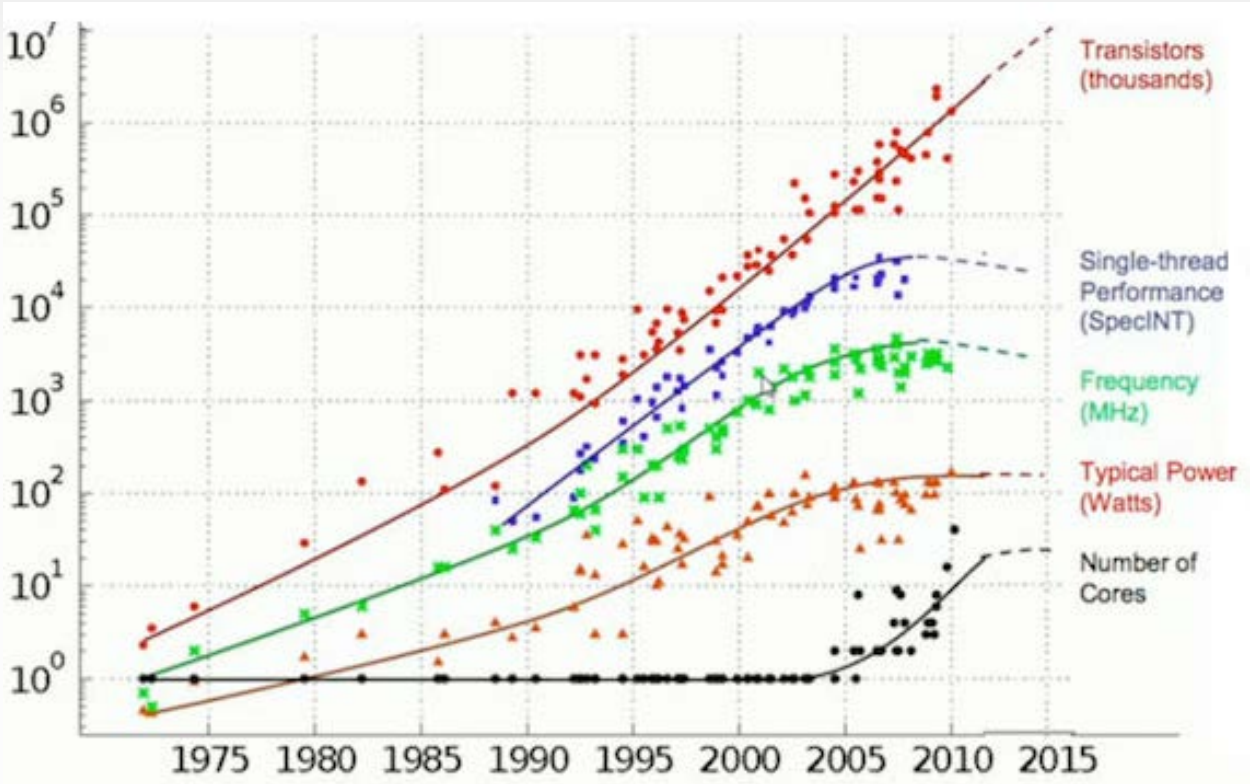
  En el orden de unos pocos ciclos de reloj esto quiere decir unos pocos nanosegundos

- Integración de más transistores: miniaturización

  Esto conlleva problemas

  - *Disipación de calor*
  - *Interferencia electromagnética*

Limitaciones físicas de la computación secuencial



- Ley de Moore:

Es imposible mantener la evolución del doble de transistores cada dieciocho meses.

Solución incrementar el numero de cores

Limitaciones físicas de la computación secuencial

SOLUCION

Paralelismo como forma de incrementar las prestaciones

Limitaciones físicas de la computación paralela

Mismas limitaciones que los secuenciales más…

- Acotación del grado:

  Es impensable el establecimiento de un número ilimitado de enlaces de comunicación entre los elementos de proceso.

- Organización lógica del circuito:

  Su susceptibilidad de división en problemas independientes

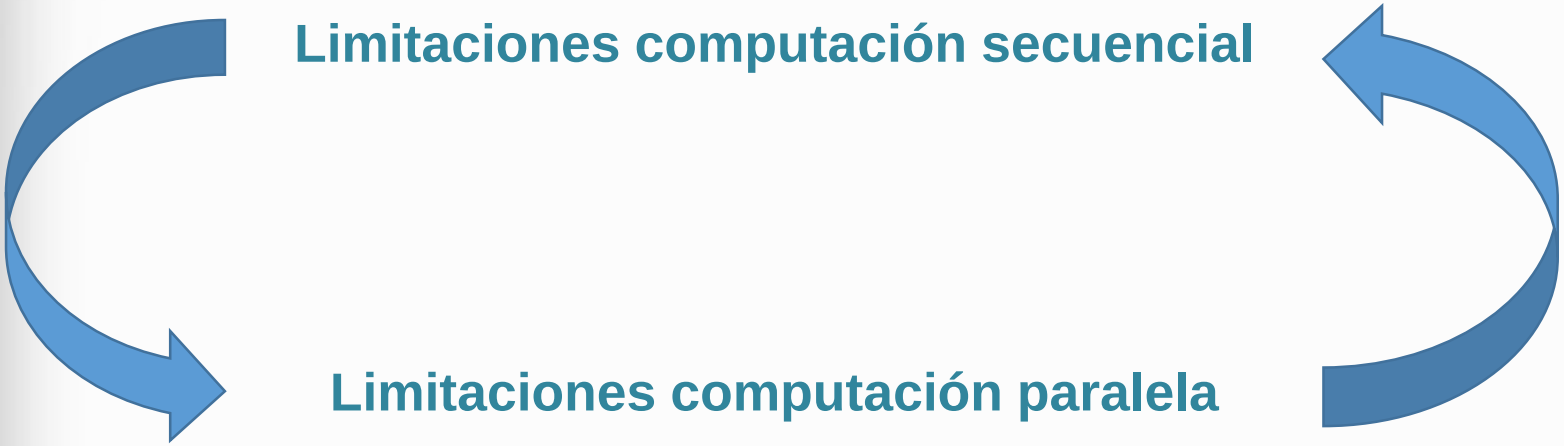
Limitaciones físicas de la computación paralela

El incremento del número de procesadores no basta para mejorar el rendimiento global, incluso puede llegar a empeorar la eficiencia del sistema

La eficiencia se mejora cuando:

- Se logra un balance de carga entre procesadores: igual número de tareas de igual tamaño
- Se minimiza la interacción entre tareas
  (minimizando la comunicación y/o mejorando los canales de comunicación)

Convergencia de Limitaciones



Niveles de Paralelismo

- Distintas necesidades que le surgen al programador que condicionan la programación paralela:

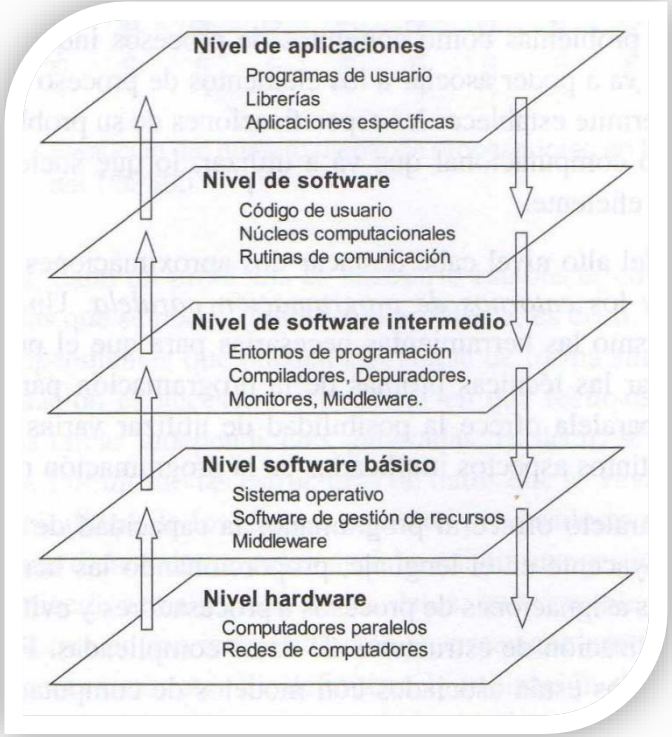
Nivel Hardware

Nivel de Software básico

Nivel de Software medio

Nivel de Software

Nivel de Aplicaciones



Niveles de Paralelismo

➢ Nivel Hardware:

La finalidad es de que los niveles superiores puedan abstraerse de las implementaciones hardware.

Dos modelos:

- *Computador paralelo con memoria compartida*

Distintos procesadores comparten una memoria organizada como un espacio único de direcciones.

- *Computador paralelo con memoria distribuida*

Cada procesador del sistema tiene su propia memoria local y se comunica con el resto de procesadores a través de un mecanismo de paso de mensajes.

Estos son simplemente modelos y pueden ocultar la verdadera naturaleza física

Niveles de Paralelismo

➢ Nivel de Software básico:

Elementos software que hacen posible la gestión de los elementos físicos en el nivel hardware y proporcionan al usuario un interfaz mínimo de interacción.

- *SO, Drivers, etc…*

➢ Nivel de Software medio:

Las herramientas que utiliza el programador para implementar programas paralelos de alto nivel:

- *Compiladores (Open MultiProcessing)*
- *Entornos de programación paralela*

Es frecuente que el uso de compiladores paralelos esté asociado a la utilización del modelo de computador paralelo con memoria compartida.

Niveles de Paralelismo

- **Nivel de Software:**

  Diseño y evaluación de algoritmos paralelos:

  - *Distintas estructuras de datos*
  - *Distribuciones de los datos en la memoria (o las memorias)*

- **Nivel de Aplicaciones:**

  Programas que específicamente abordan la problemática de un determinado campo, utilizando todas las herramientas que proporcionan los niveles anteriores:

  - *Librerías especializadas*

Formas Básicas de Paralelismo

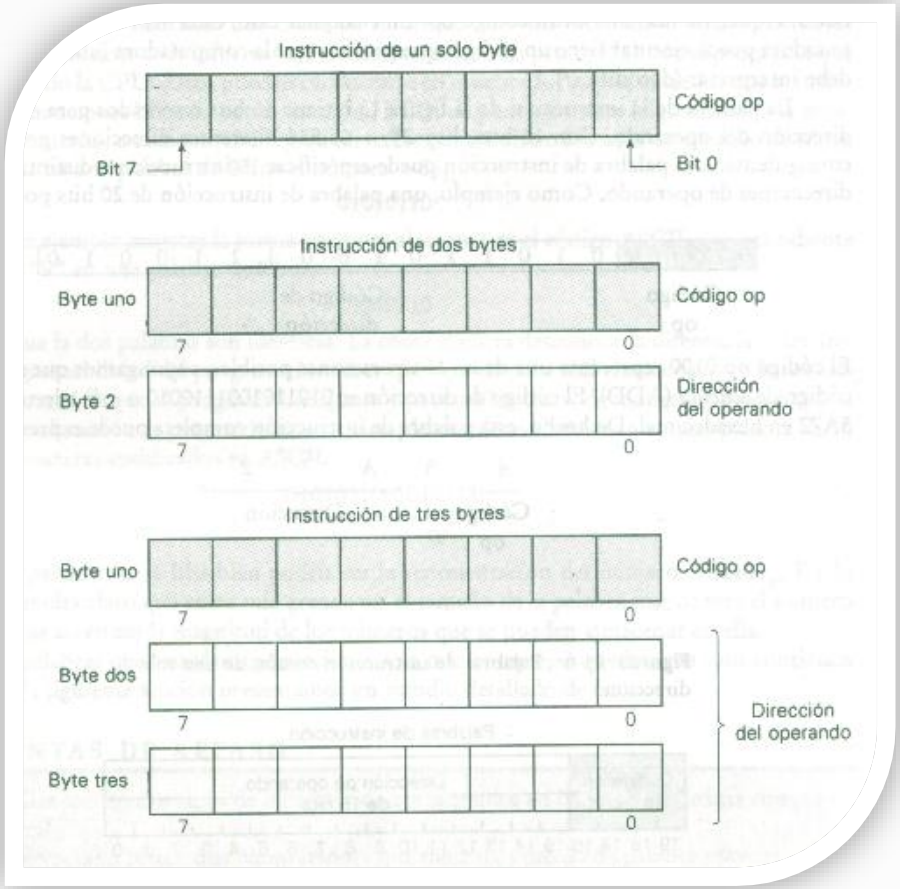
Tipos de Paralelismo

- *A nivel de Bit*
  (BLP – Bit Level Paralelism)
- *A nivel de Instrucción*
  (IPL – Instruction Level Paralelism)
- *A nivel de Datos*
  (DLP – Data Level Paralelism)
- *A nivel de Tarea*
  (TLP – Threads(Task) Level Paralelism)
- ***A nivel de De procesos***
  (PLP – Proccess level Paralelism)

Formas Básicas de Paralelismo

➢ A Nivel de Bit (*BLP – Bit level)*

El aumento del tamaño de la palabra reduce el número de instrucciones que el procesador debe ejecutar para realizar una operación en variables cuyos tamaños son mayores que la longitud de la palabra.



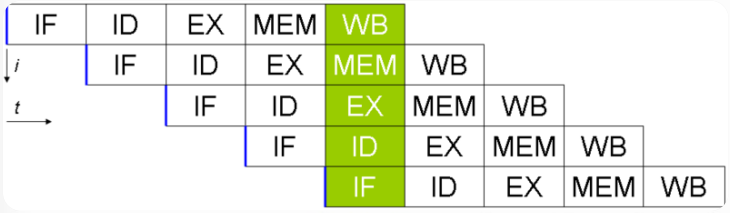
Formas Básicas de Paralelismo

➢ A nivel de Instrucción

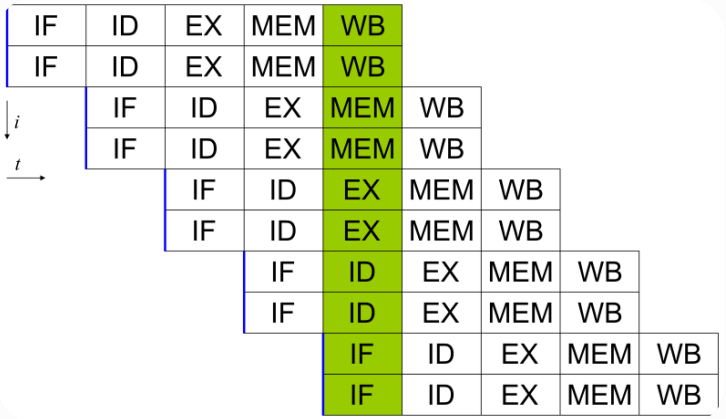
(IPL – Instruction Level Paralelism)

Las instrucciones independientes pueden reordenarse y combinarse en grupos que luego son ejecutadas en paralelo sin cambiar el resultado del programa.

Pipeline Procesador escalar:

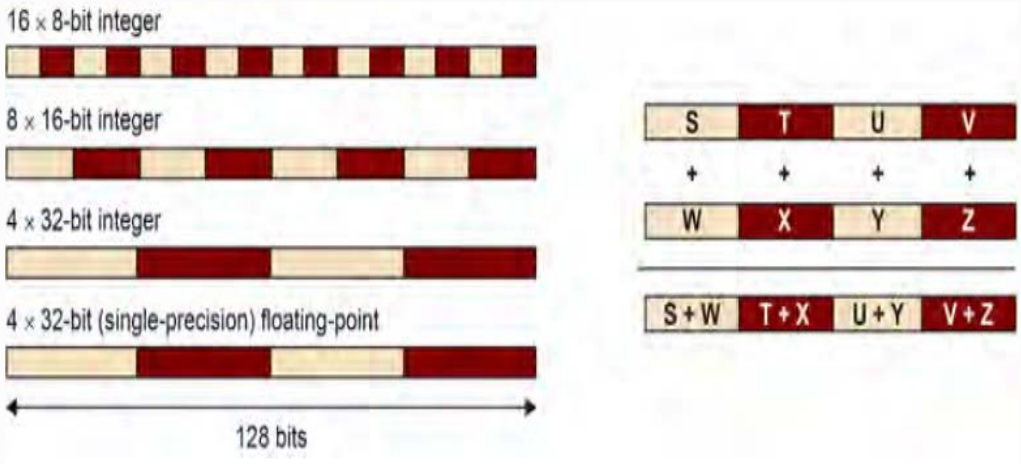


Pipeline Procesador Superescalar de grado 2:



Formas Básicas de Paralelismo

- **A nivel de Datos**
  (DLP – Data Level Paralelism)



Distribución de los datos independientes entre los diferentes nodos computacionales que deben tratarse en paralelo, es decir, se realiza el mismo cálculo en distintos o en los mismos grupos de datos.

Se refiere básicamente a la posibilidad de operar sobre dos o más datos con una única instrucción

Es necesaria la implicación del programador (y del compilador)

Formas Básicas de Paralelismo

- **A nivel de Tarea**
  (TLP – Threads*(Task)* Level Paralelism)

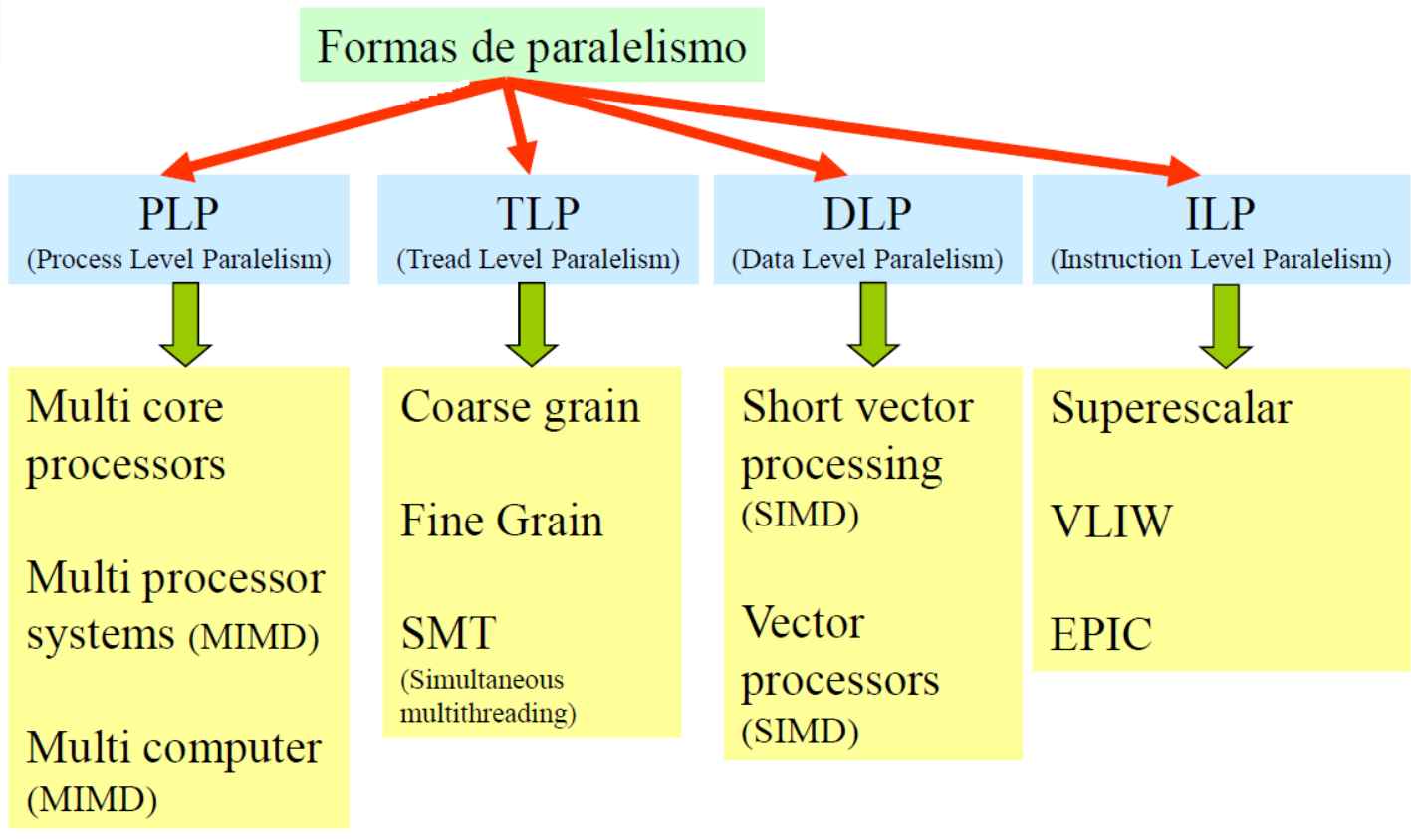
  Característica de un programa paralelo, donde distintas tareas son asignadas a cada uno de los procesadores, en la que «cálculos completamente diferentes se pueden realizar en cualquier conjunto igual o diferente de datos»

- ****A nivel de Procesos****
  (PLP – Proccess level Paralelism)

  Permiten ejecutar distintos procesos en diferentes procesadores paralelos o en diferentes cores de un mismo procesador

Formas Básicas de Paralelismo

Clasificación



Formas Básicas de Paralelismo (y II)

Segmentación (Pipelining) y Replica

Dos formas básicas para conseguir que un conjunto de procesos se ejecuten en paralelo en un único procesador.

Segmentación

Dividir una unidad funcional en etapas independientes para disminuir el tiempo entre instrucciones.

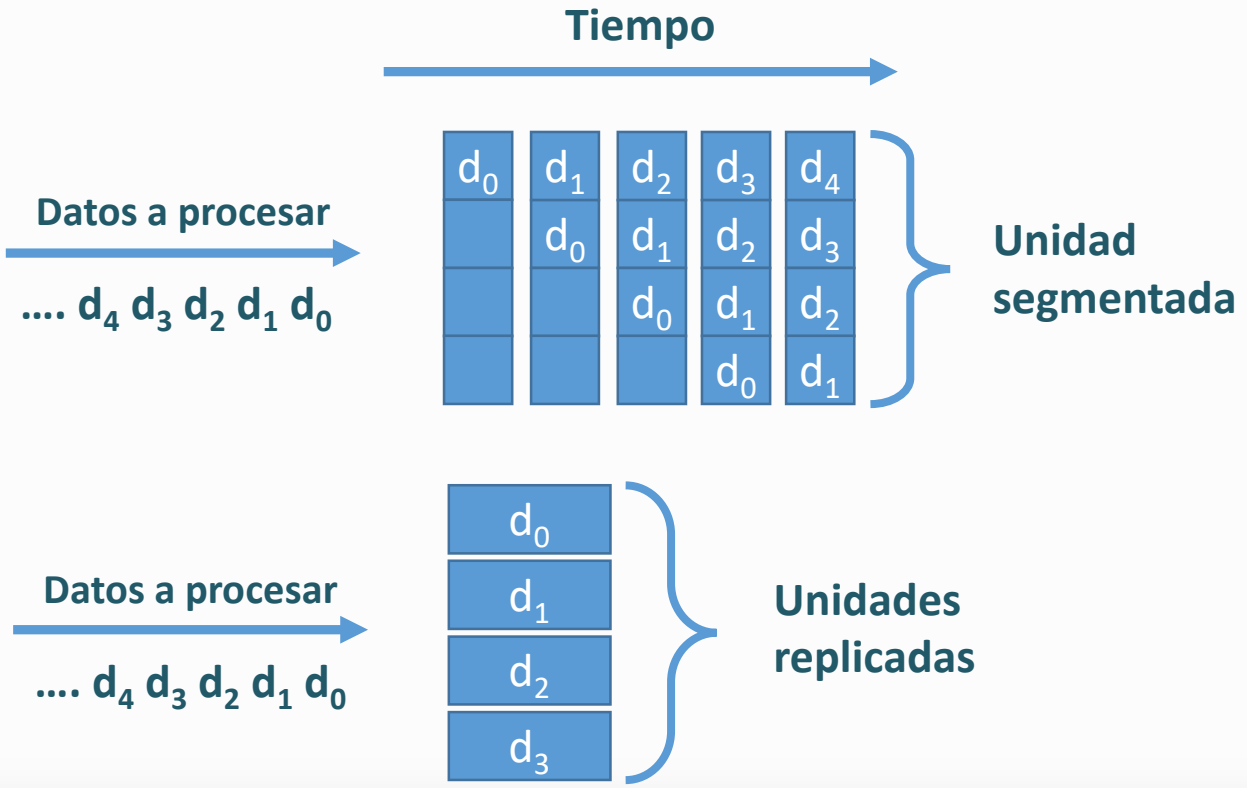
Réplica

Varias unidades funcionales para aumentar el numero de operaciones por unidad de tiempo.

Formas Básicas de Paralelismo (y II)

Segmentación (Pipelining) y Replica

Dos formas básicas para conseguir que un conjunto de procesos se ejecuten en paralelo en un único procesador.



Formas Básicas de Paralelismo (y II)

Segmentación (Pipelining)

- **Problemas:**
  - La necesidad de retardos diferentes en cada etapa obliga a usar el mayor de ellos, ralentizando el proceso.
  - Diferentes tipos de organizaciones segmentadas para cada juego de instrucciones.
  - Problemas de dependencias de datos.
  - Conflictos de acceso cuando un operando debe buscarse en el mismo módulo de memoria que una instrucción.
  - Instrucciones de salto puede invalidar toda la búsqueda anticipada de instrucciones.

Formas Básicas de Paralelismo (y II)

Segmentación (Pipelining)

- **Soluciones**:
  - Reordenar las instrucciones para evitar dependencias de datos.
  - Diseñar dos tipos de memorias cache, una para datos y otra para instrucciones (arquitectura Harvard),evitando conflictos de acceso.
  - Técnicas *pipelining* para construir una unidad segmentada de dos etapas evitando accesos innecesarios a memoria.
  - Desenrollado de bucles para aprovechar la unidad segmentada.

Formas Básicas de Paralelismo (y II)

Segmentación (Pipelining)

- **Soluciones:**
  - Técnica pipelining; Ej.Chaining

***Algoritmo**: Producto escalar de dos vectores.*

```
Para i = 1 hasta n hacer
        S = S + a[i] * b[i];
fin_para
```

Formas Básicas de Paralelismo (y II)

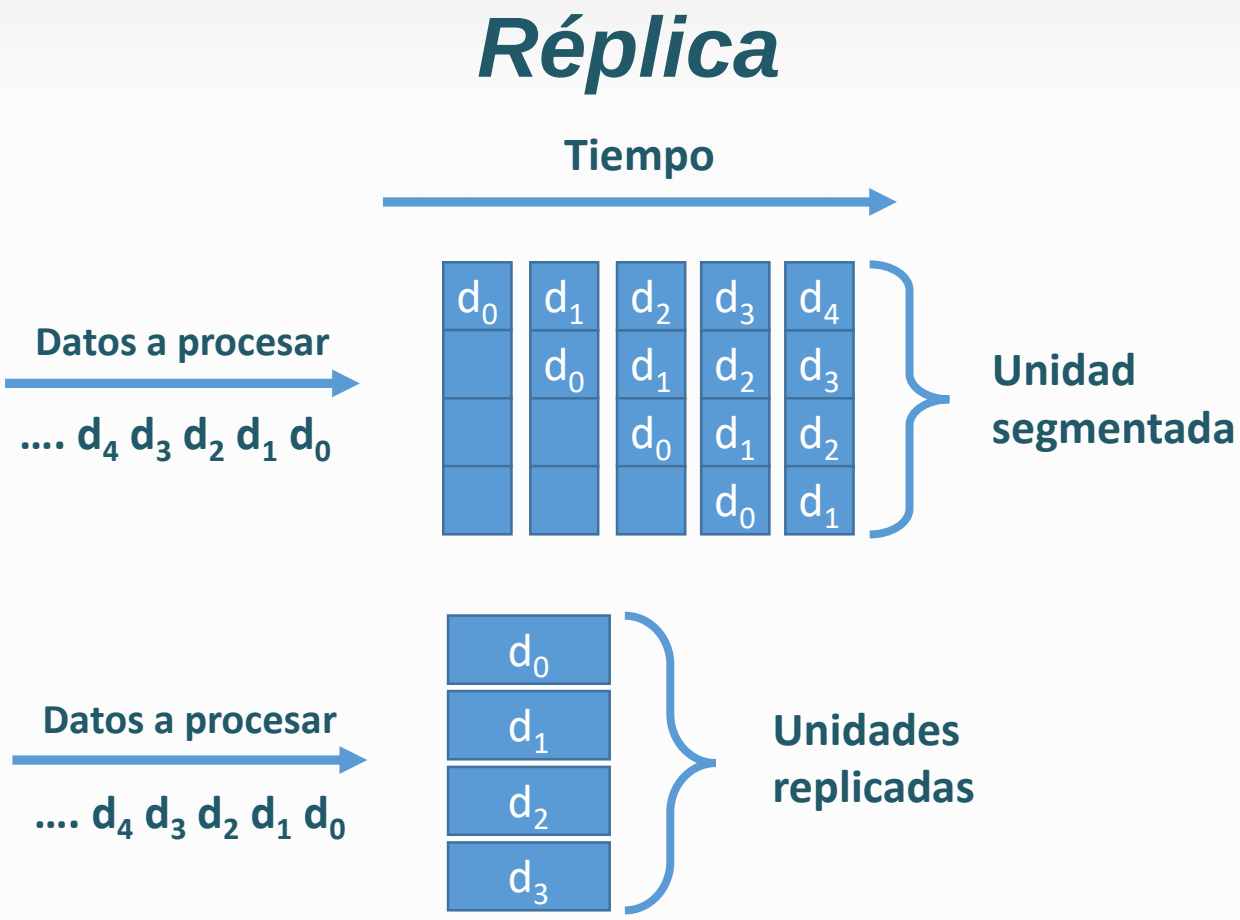
Segmentación (Pipelining)

- **Soluciones:**
  - Desenrollado de bucles.

***Algoritmo** : Producto escalar de dos vectores con desenrollado.*

```
// i=1,3,5, ... ,n
Para i = 1 hasta n paso 2 hacer
        S = S + a[i] * b[i];
        S = S + a[i+1] *b[i+1];
fin_para
```

Formas Básicas de Paralelismo (y II)



Duplicamos unidades funcionales

ÍNDICE

ÍNDICE

Paralelismo en Monoprocesadores

- Vectoriales
- Escalares
  - Procesadores segmentados
  - Procesadores supersegmentados
  - Procesadores superescalares
- Técnicas Multithread
- Procesadores VLIW

Paralelismo en monoprocesador

En computadores secuenciales con un único procesador se utilizan conceptos de paralelismo para incrementar las prestaciones que están relacionadas con distintos niveles:

- *Incremento de la longitud de palabra*
- *Multiplicidad de unidades funcionales*
- *Solapamiento entre las operaciones de la CPU e I/O*
- *Uso de los recursos a tiempo compartido*

Paralelismo en monoprocesador

- El tiempo de ejecución (t_E) ha de tener un compromiso entre tres factores:
  - N: Número de instrucciones del programa.
  - C: Número de ciclos de reloj por instrucción.
  - t_C. Tiempo en segundos del ciclo de reloj

$$t_E = NCt_C$$

- El producto ha de ser mínimo para favorecer la eficiencia.

Procesadores Vectoriales

Los conjuntos de datos homogéneos más utilizados son los vectores y matrices, por eso se diseñó los procesadores vectoriales.

Básicamente un procesador vectorial es un procesador que dispone de instrucciones vectoriales en su juego de instrucciones.

Implica cambio de hábitos al programar

Procesadores Vectoriales

Es importante la organización de la memoria, ya que los vectores suelen almacenarse en posiciones consecutivas en el mismo módulo, es más eficiente en este tipo de procesadores organizar la memoria de forma entrelazada, es decir, almacenar posiciones del vector en distintos módulos consecutivos así no existe penalización en el tiempo de acceso ala mismo módulo.

Procesadores Vectoriales

Arquitecturas asociadas:

- ***Procesadores de Array***
  Se basan en la replicación de las ALU's
  Muy poca salida comercial, en desuso
- ***Procesadores Vectoriales***

- ***Extensiones SIMD***

Procesadores Escalares

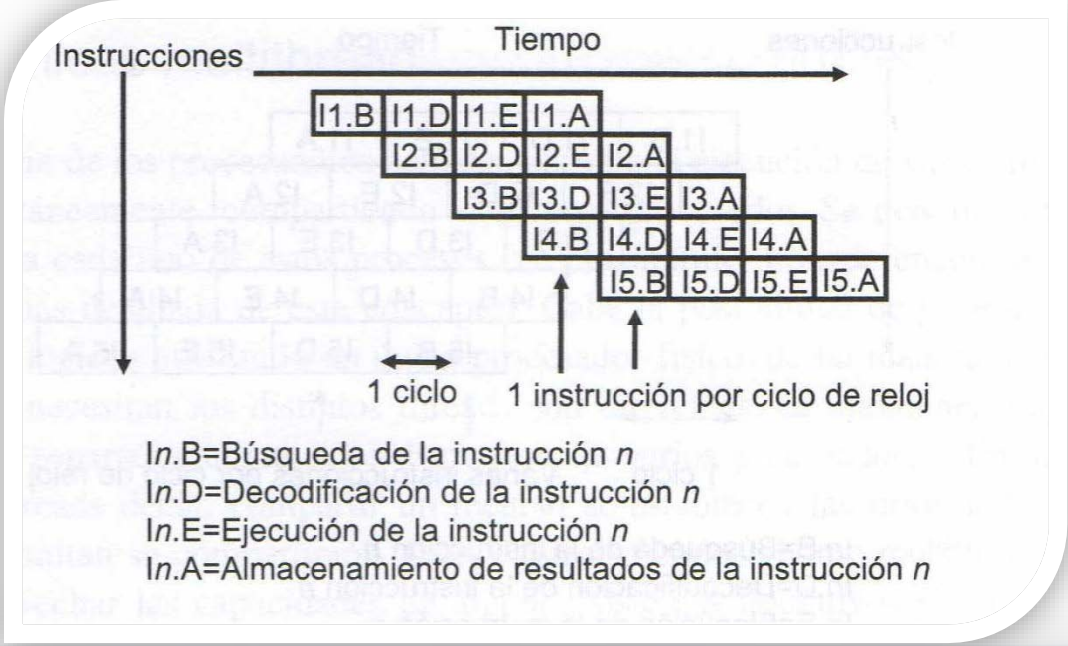
- Son procesadores ILP
- Explotan el paralelismo usando técnicas de replica y/o segmentación
- Dividen la ejecución de una instrucción en varias etapas físicas, permitiendo que cada etapa ejecute una parte de una instrucción diferente
  - Decodificación, lectura de operandos, ejecución, almacenamiento …
- Reducen el numero de ciclos necesarios para una instrucción
- Es transparente al programador
- Tres tipos principales (No mutuamente excluyentes)
  - **Procesadores segmentados**
  - **Procesadores supersegmentados**
  - **Procesadores superescalares**

Procesadores Escalares

Procesadores segmentados

Disminuir el número de ciclos de reloj por instrucción utilizando segmentación sobre todo el proceso de ejecución de la instrucción.

Suponiendo que cada etapa tenga una duración de un ciclo de reloj, se puede conseguir, en la práctica, ejecutar una instrucción por ciclo.

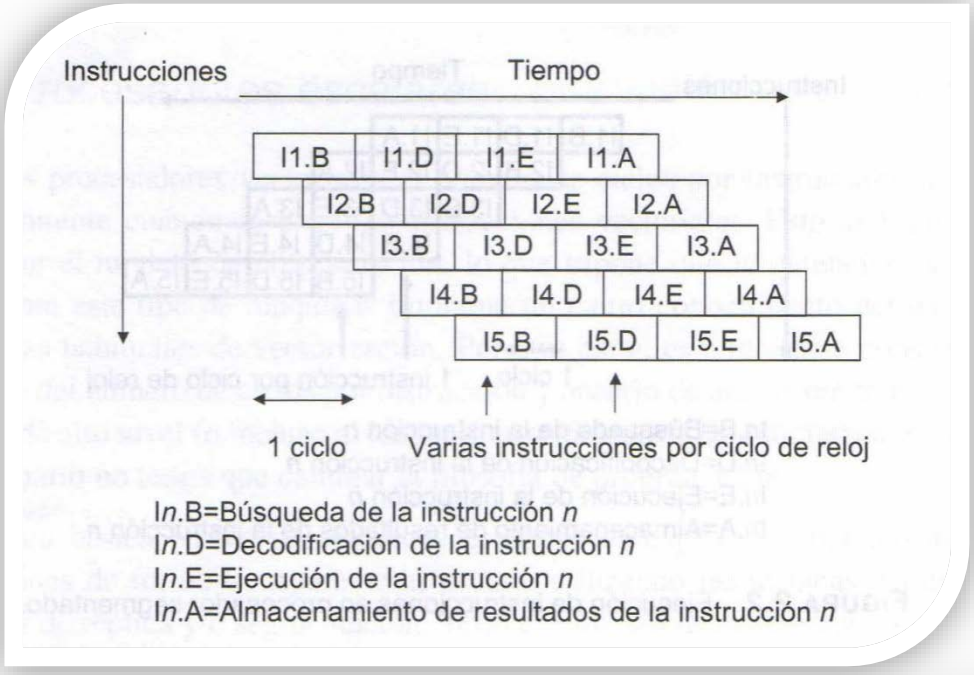


Procesadores Escalares

Procesadores supersegmentados

Cada etapa, no debe considerarse como una unidad atómica de ejecución, sino que puede dividirse en subetapas.

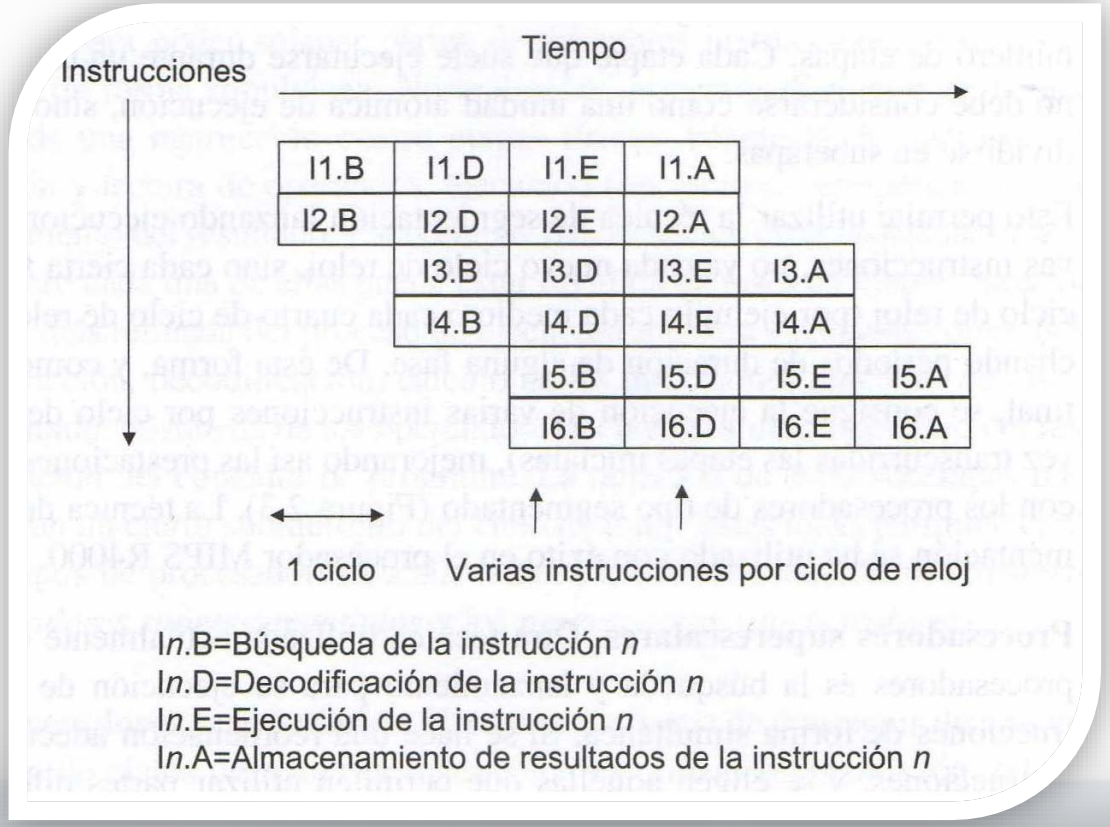
Esto permite lanzar ejecuciones de nuevas instrucciones cada cierta fracción del ciclo de reloj.



Procesadores Escalares

Procesadores superescalares

Ejecución de varias instrucciones de forma simultánea haciendo una reordenación de las instrucciones, y se eligen las que utilizan partes diferentes del microprocesador. Se puede completar usando replica de UF



Procesadores Escalares

Confusión común:

ES FALSO

Supersegmentado = Superescalar + Segmentado

APROXIMACIÓN CORRECTA

Supersegmentado = Segmentado + Segmentado

Técnicas Multithread

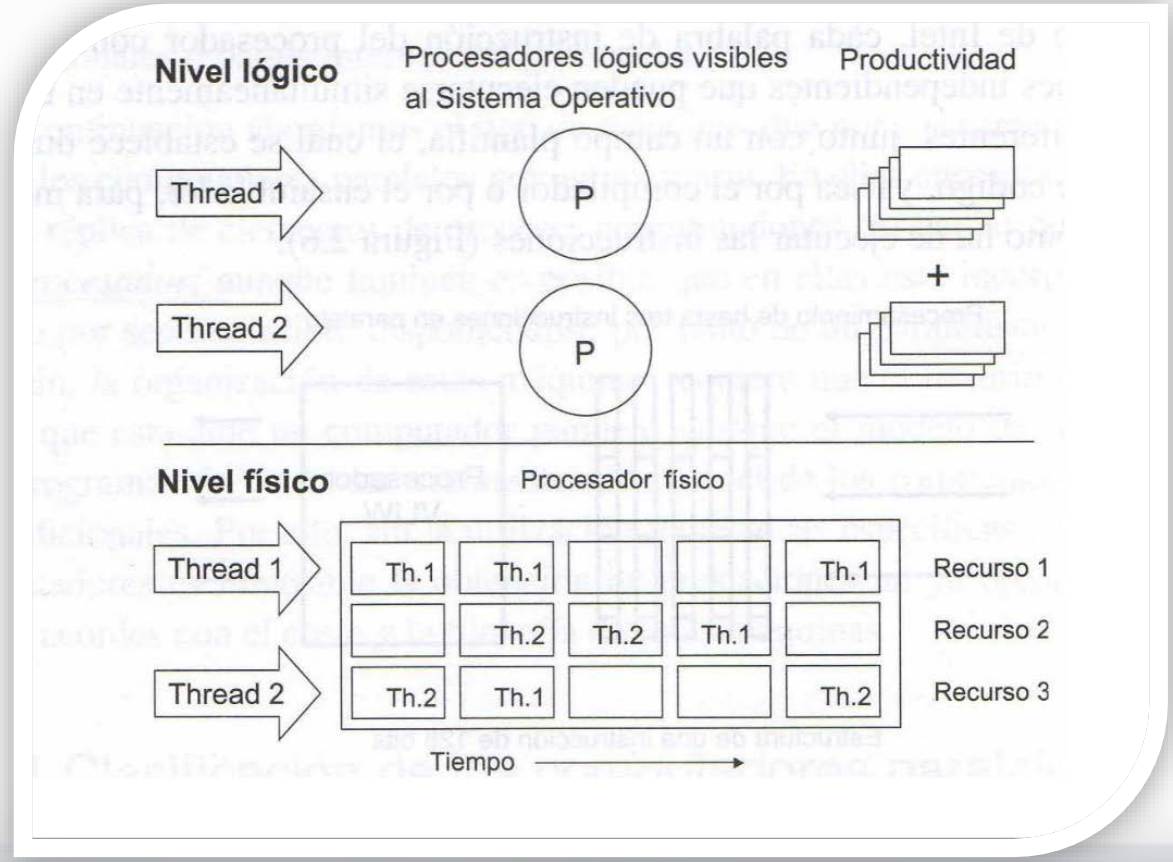
- Intenta generar varios procesadores lógicos con un procesador físico.
- Motivado para evitar los tiempos de espera al usar recursos mas lentos. (Como latencias de E/S)
- Los recursos que necesitan los distintos threads han de ser diferentes, manteniendo así flujos de instrucciones separados
- Se mantienen ocupados los recursos del procesador aun con fallos de cache o de predicción de salto

- ***Thread (hilo, hebra):*** procesos ligeros que comparten el procesador físico.
- ***Multithreading:*** Técnica hardware que permite compartir un recurso con varios *Threads* estableciendo prioridades.

  Hyperthreading. Implementación de Intel del multithread (Xeon, Core Ix,…)

Técnicas Multithread

El software puede dividir su carga de trabajo en procesos cuya ejecución se puede planificar de forma independiente, teniendo varios elementos de proceso ejecutándose simultáneamente utilizando la réplica.



Técnicas Multithread **(aproximaciones)**

- **Multithreading de grano grueso.**

Los threads son desalojados del procesador con baja frecuencia, usualmente cuando el thread realiza alguna I/O, o ante un fallo de cache.

- **Multithreading de grano fino.**

El thread en ejecución es cambiado (thread swaping) en cada ciclo de reloj

- **Simultaneous Multithreading (SMT).**

Similar a grano fino, pero permite ejecutar múltiples threads en cada ciclo de reloj. SMT permite concurrencia física, a diferencia de los anteriores que solo manejan Concurrencia virtual

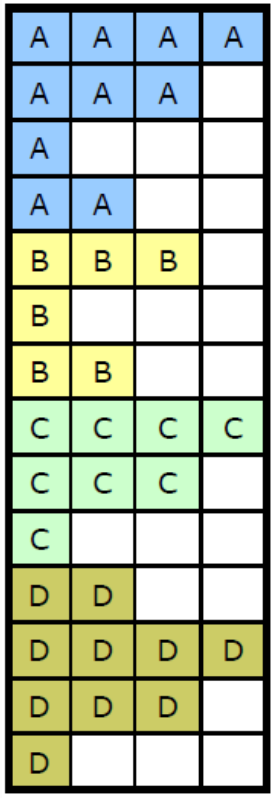
Hyperthreading. Implementación de Intel del multithread (Xeon, Core Ix,…)

- **Chip Multi Processor (CMP).**

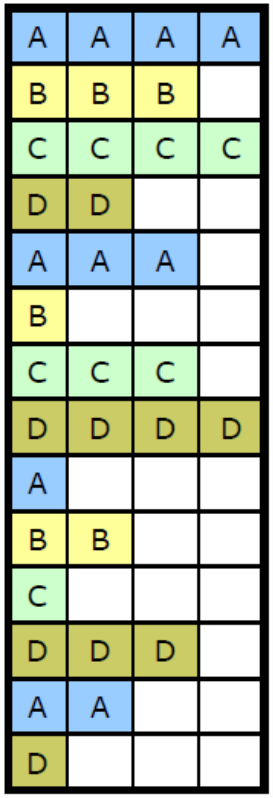
Hablaremos de él más adelante.

Técnicas Multithread

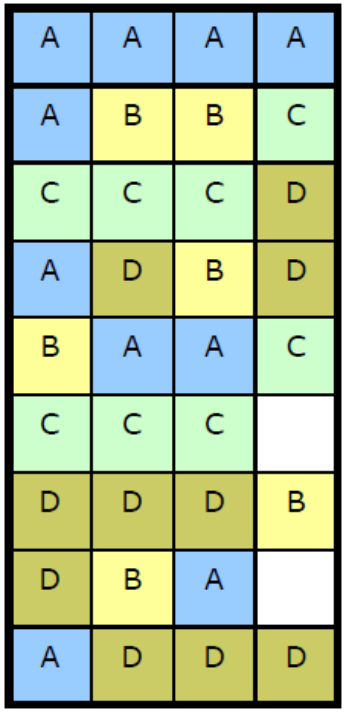
Hay varias aproximaciones al multithreading



Grano Grueso



Grano fino



SMT

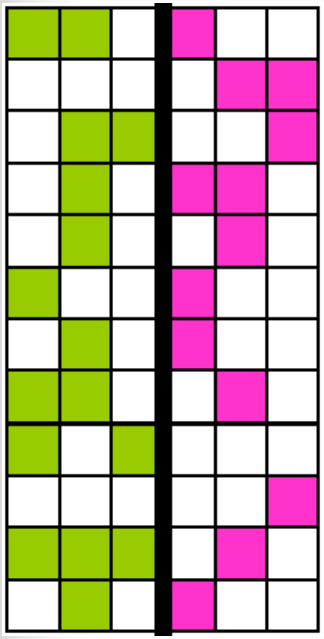
Técnicas Multithread

Clasificando las aproximaciones

Partición Estática

Espacial

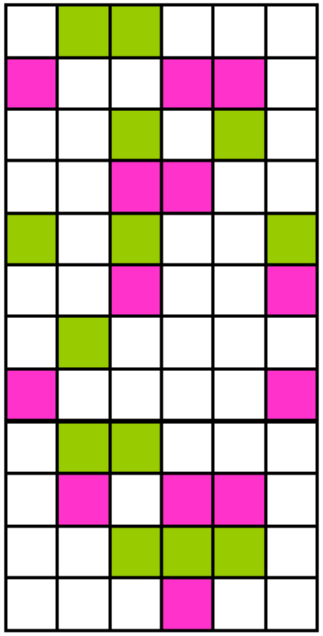
Cores



CMP

Temporal

1 Ciclo

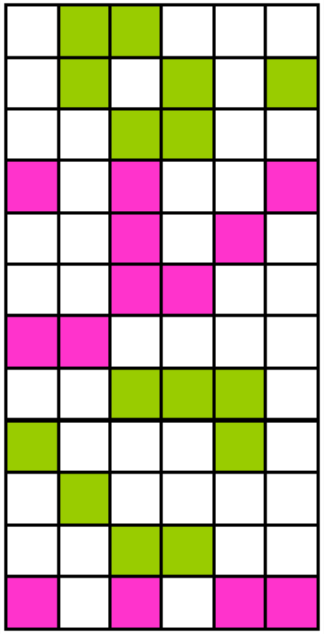


Grano fino

Partición Dinámica

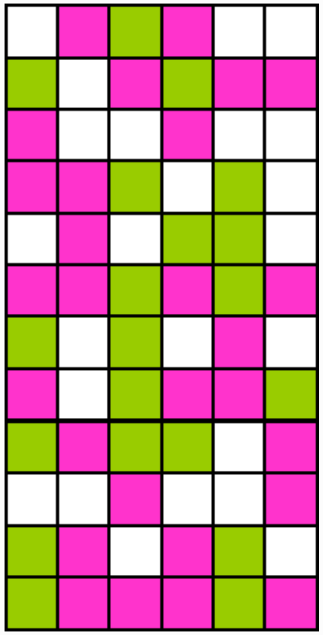
Temporal

Ciclos Vari.



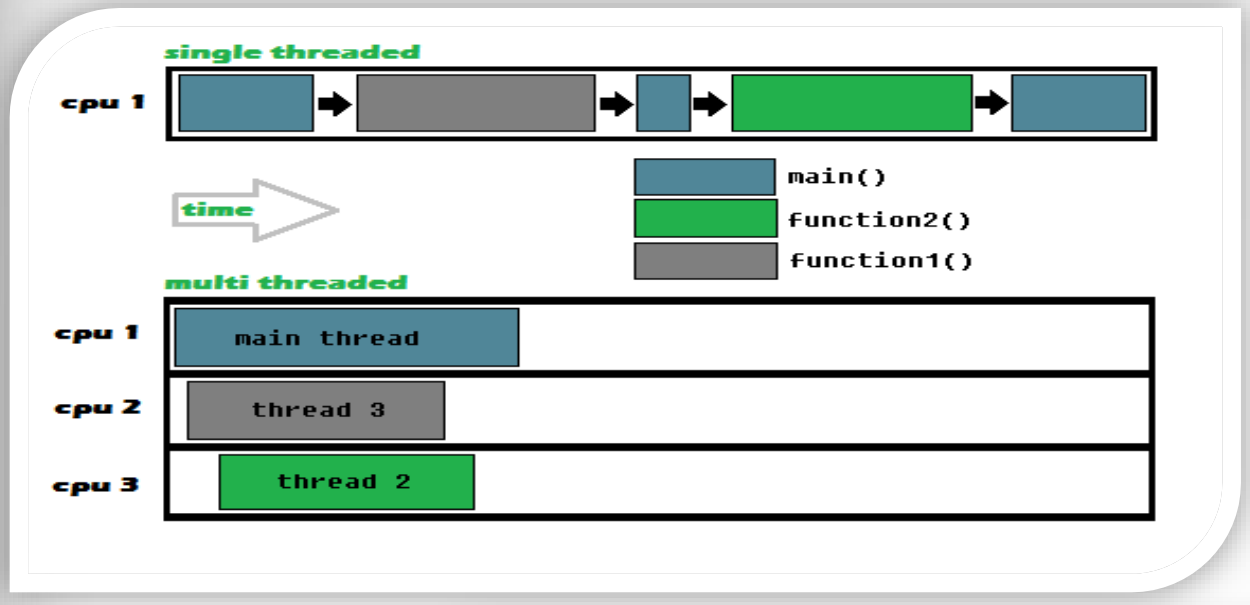
Grano Grueso

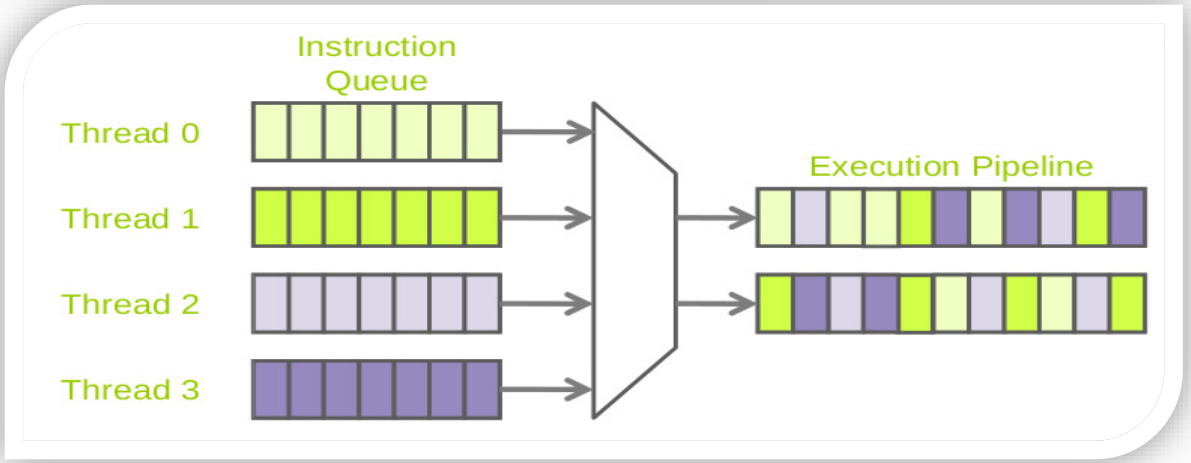
Unidad Funcional



SMT

Técnicas Multithread





Técnicas Multithread

El cambio de ejecución de un thread a otro suele llevar asociado un ***cambio de contexto***

El cambio de contexto ha de guardar:

- ***Contador de programa***
- ***Registros generales***
- ***Registros especiales ….***

  … de estado, de control, punteros, shadows, etc.

- ***Datos de Memoria***

Técnicas Multithread

El cambio de contexto tradicional….

…. Es el SO el encargado de gestionarlo

- Interrupción detiene el *thread.*
- SO guarda contexto del *thread* parado.
- SO restaura contexto de *thread* parado previamente.
- Se reinicia el *thread.*

…. Los *threads* no se ‘enteran’ que su ejecución fue parada, guardada, restaurada y reejecutada

Técnicas Multithread

El cambio de contexto tradicional….

Un fallo de caché suele costar de 16 a 32 ciclos.

Un cambio de contexto puede llevar cientos de ciclos.

En determinadas ocasiones no compensa realizar un cambio de contexto tradicional

Solución: cambio de contexto rápido por hardware

Técnicas Multithread

El cambio de contexto rápido….

- Se basa en replicar registros de contexto (PC, estado, etc.)
- Consume pocos ciclos se selecciona el conjunto de registros a utilizar
- Arquitectura mas compleja
- El rendimiento de cada *thread* es peor por el aumento de la complejidad

VLIW – Very Long Instruction Word

- Idea y primeros procesadores en la década de los 70 y 80
- Se construyeron pocas VLIW CPU's de propósito general:

  (Culler, Multiflow y Cydrome).

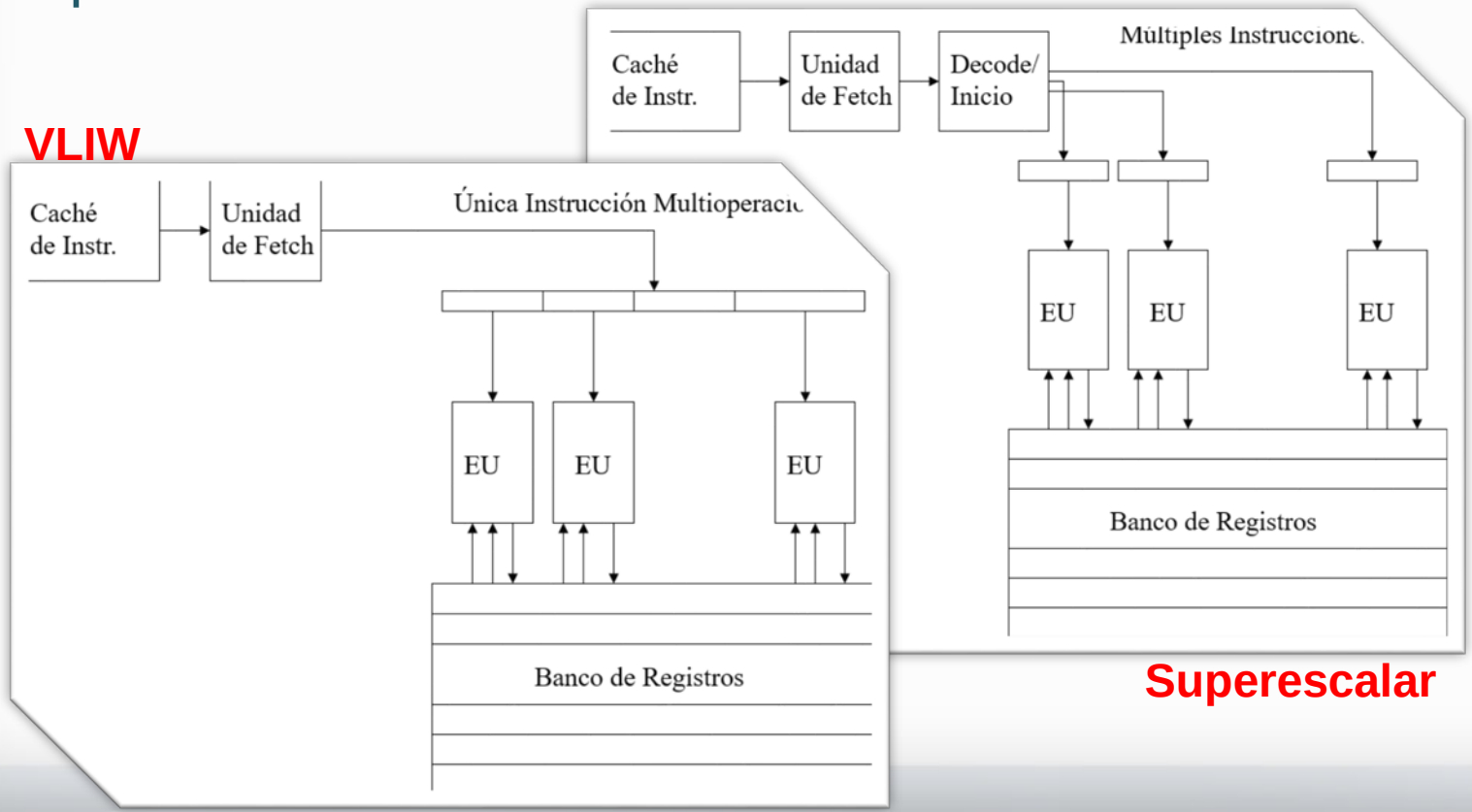
- A pesar de explotar mucho ILP, no tuvieron éxito comercial
- Arquitectura muy difundida en circuitos embebidos

  DSP´s (Digital Signal Processing) Procesadores para procesado digital de señales, telefonos, modems...

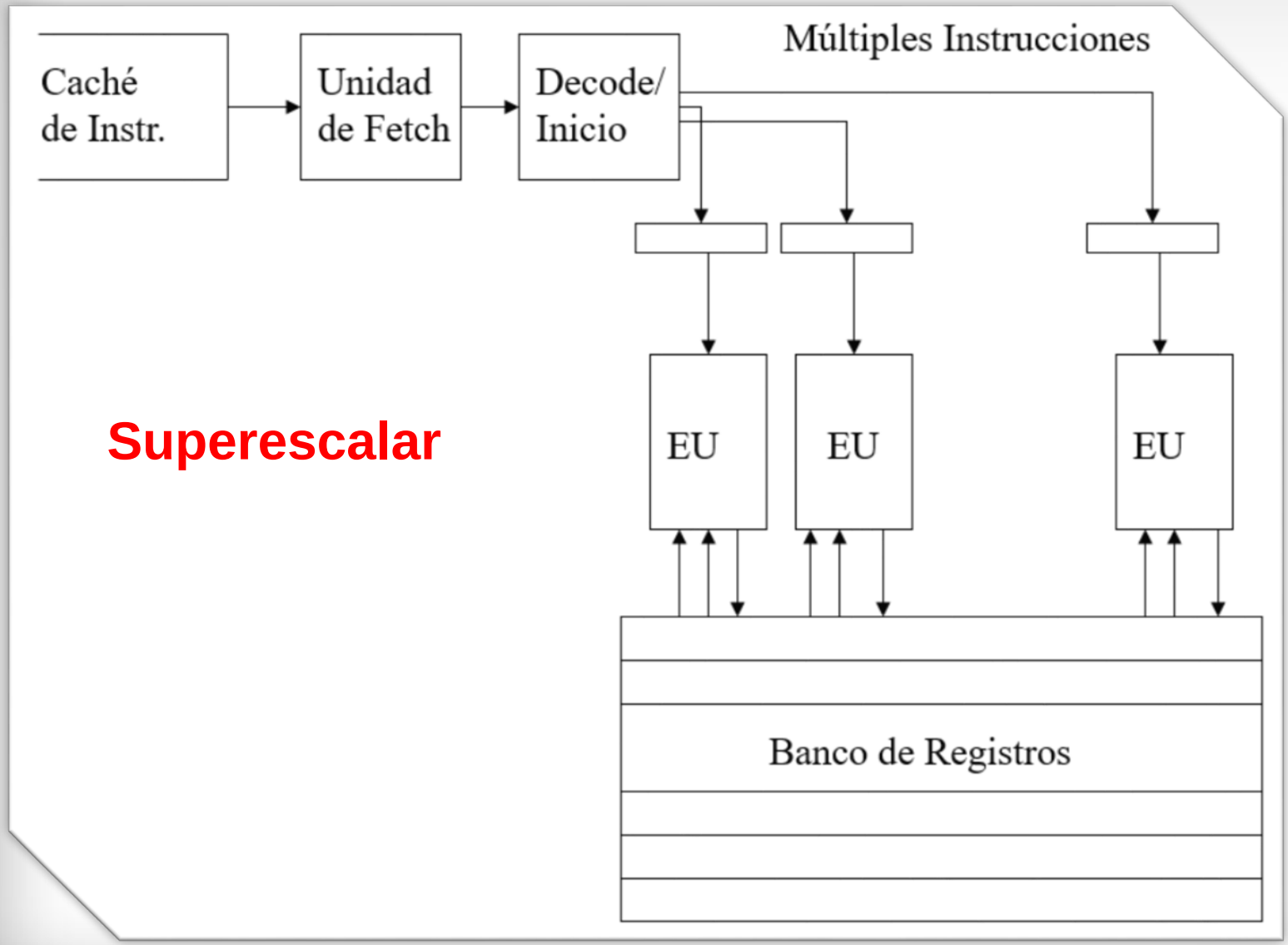
  Multimedia (placas de sonido, video, etc.)

VLIW – Very Long Instruction Word

- Instrucciones muy grandes
  (desde 128 a 1024 bits)
- **Una instrucción especifica varias operaciones**
- Replica de unidades funcionales.

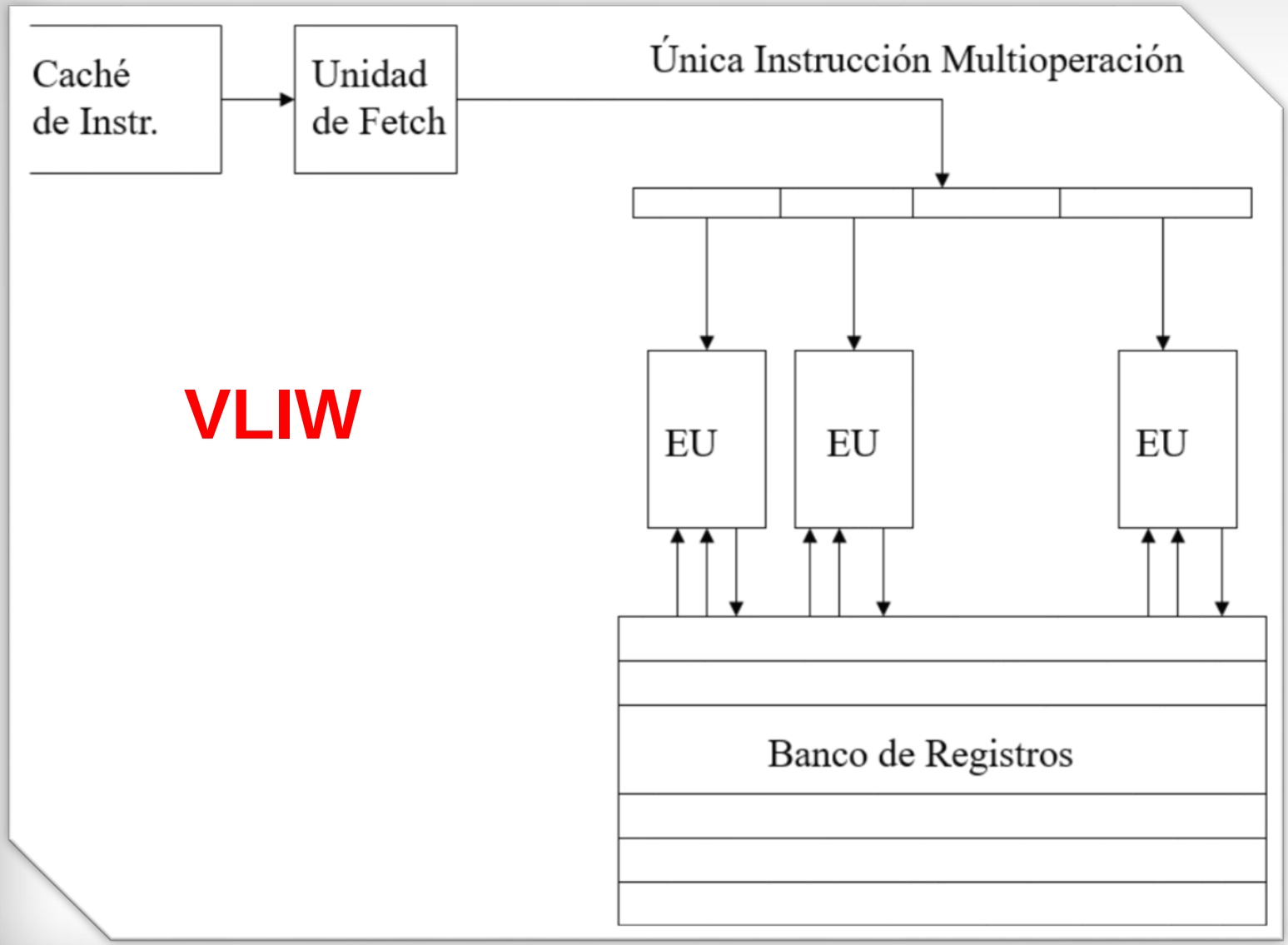


VLIW – Very Long Instruction Word



Superescalar

VLIW – Very Long Instruction Word



VLIW – Very Long Instruction Word

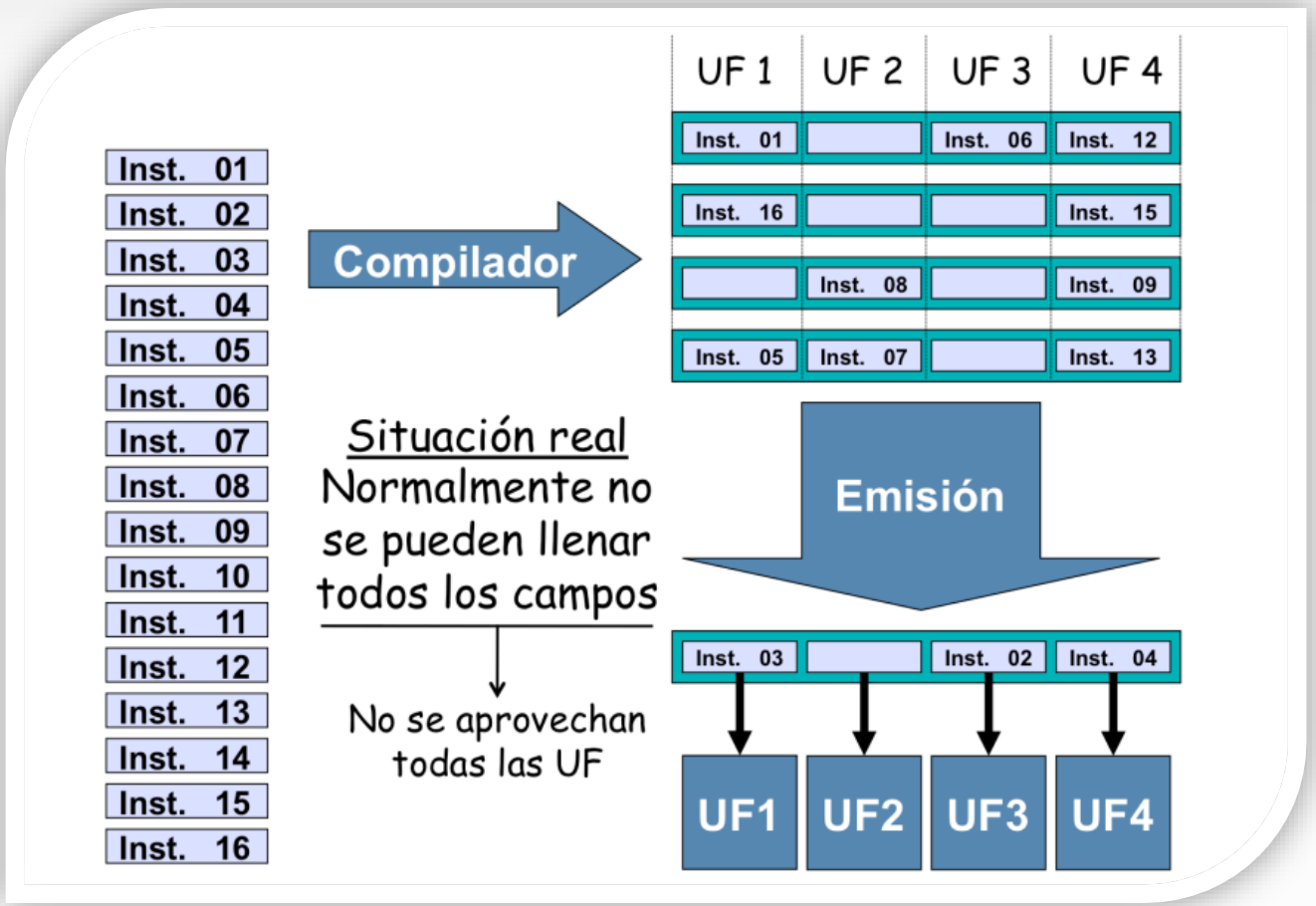
EL COMPILADOR Y VLIW

- Muy apoyada en la tecnología de compiladores que optimicen el código
- La planificación de ejecución de las operaciones es externa al procesador
- El rendimiento del procesador queda determinado por la calidad del compilador
- El compilador debía generar código “horizontal”: es decir debía especificar el paralelismo directamente.

Es el compilador el encargado de gestionar el paralelismo

- Rellenar una plantilla por el compilador ***

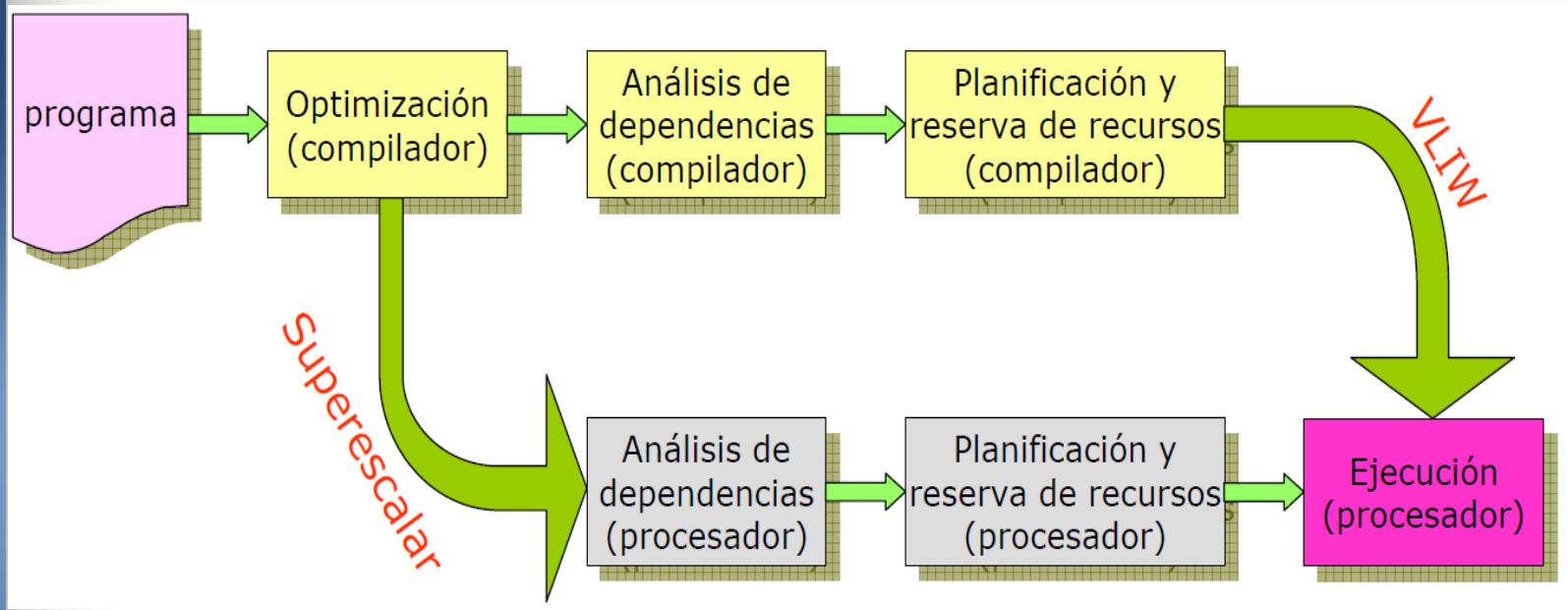
VLIW – Very Long Instruction Word



Procesador tradicional
1 instrucción --- 1 operación

Procesador VLIW
1 instrucción --- varias instrucciones (operaciones)

VLIW – Very Long Instruction Word



¡¡ En VLIW no se puede determinar si hay dependencias en ciertos casos !!

VLIW – Very Long Instruction Word

VENTAJAS

- **Planificación de ejecución realizada por el compilador**
- **Genera arquitecturas mas simples con menos lógica de control**
- **Menor consumo**
- **Mejora la paralización**
- **La simplicidad permite incrementar la velocidad del reloj**

VLIW – Very Long Instruction Word

INCONVENIENTES / LIMITACIONES

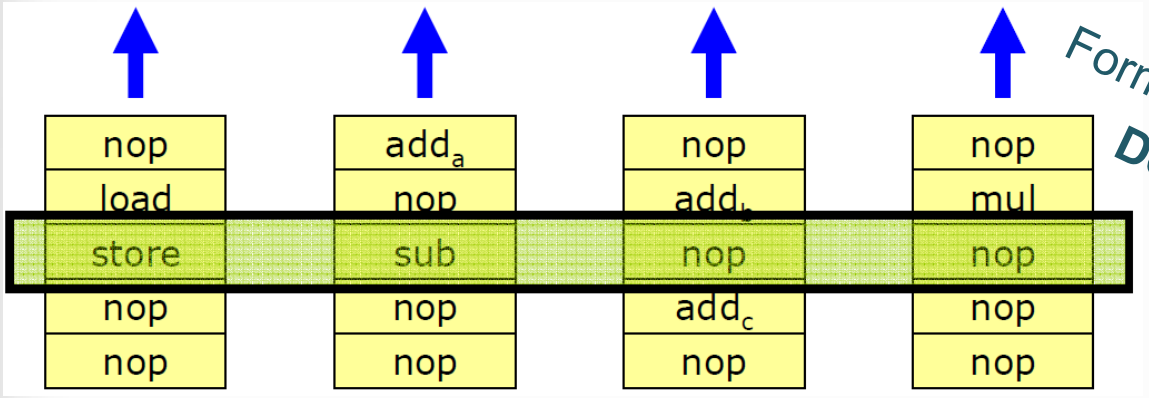
- **Tamaño del código elevado**
- **Difícil aprovechar todas las Unidades Funcionales (UF)**
  - Baja ocupación de las UF
  - Muchas Instrucciones NOP
  - Mucho desperdicio de memoria
- **Difícil mantener la compatibilidad**
  - Incompatibilidad del código en diferentes maquinas
  - Recompilar fuentes para cada nueva versión del procesador

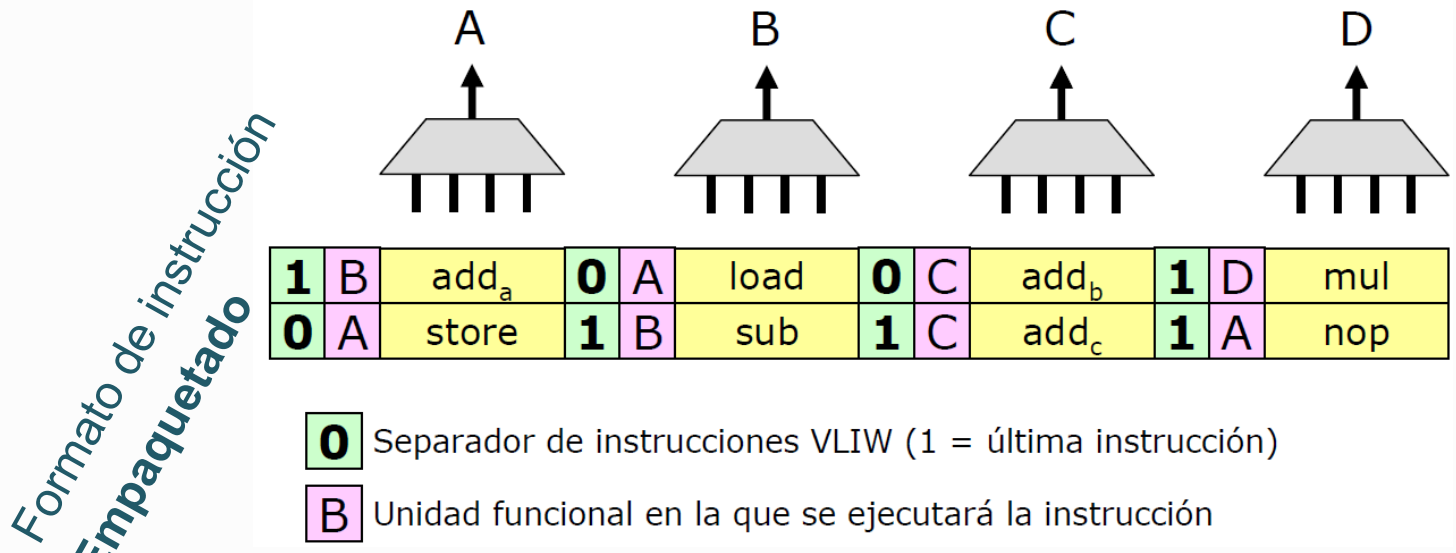
Un mismo compilador NO puede utilizarse para distintos modelos de la misma familia.

- **Conflictos y dificultades**
  - Accediendo a bancos de registros desde diferentes UF
  - Predicción estática de acceso a memoria
  - Predicción estática de saltos

VLIW – Very Long Instruction Word

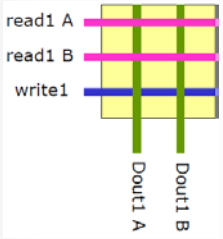
Variantes (Por formato de Instrucción)

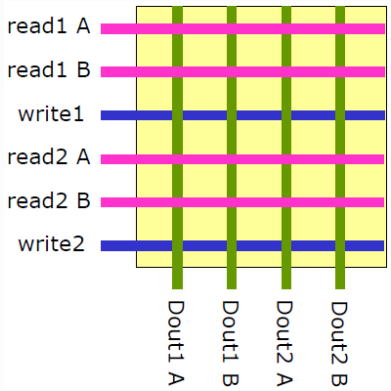




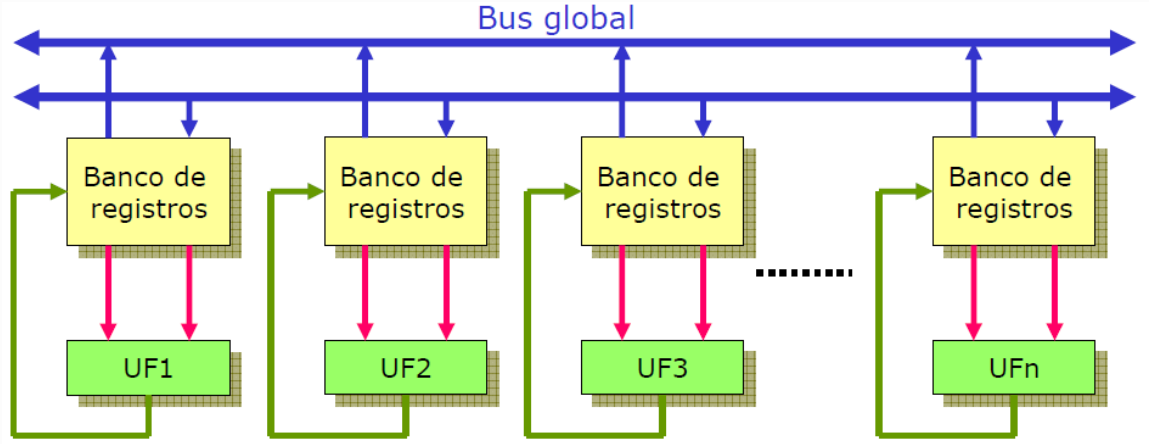
VLIW – Very Long Instruction Word

Variantes (Por arquitectura banco de registros)





Aumentar el numero de puertos al banco de registros.



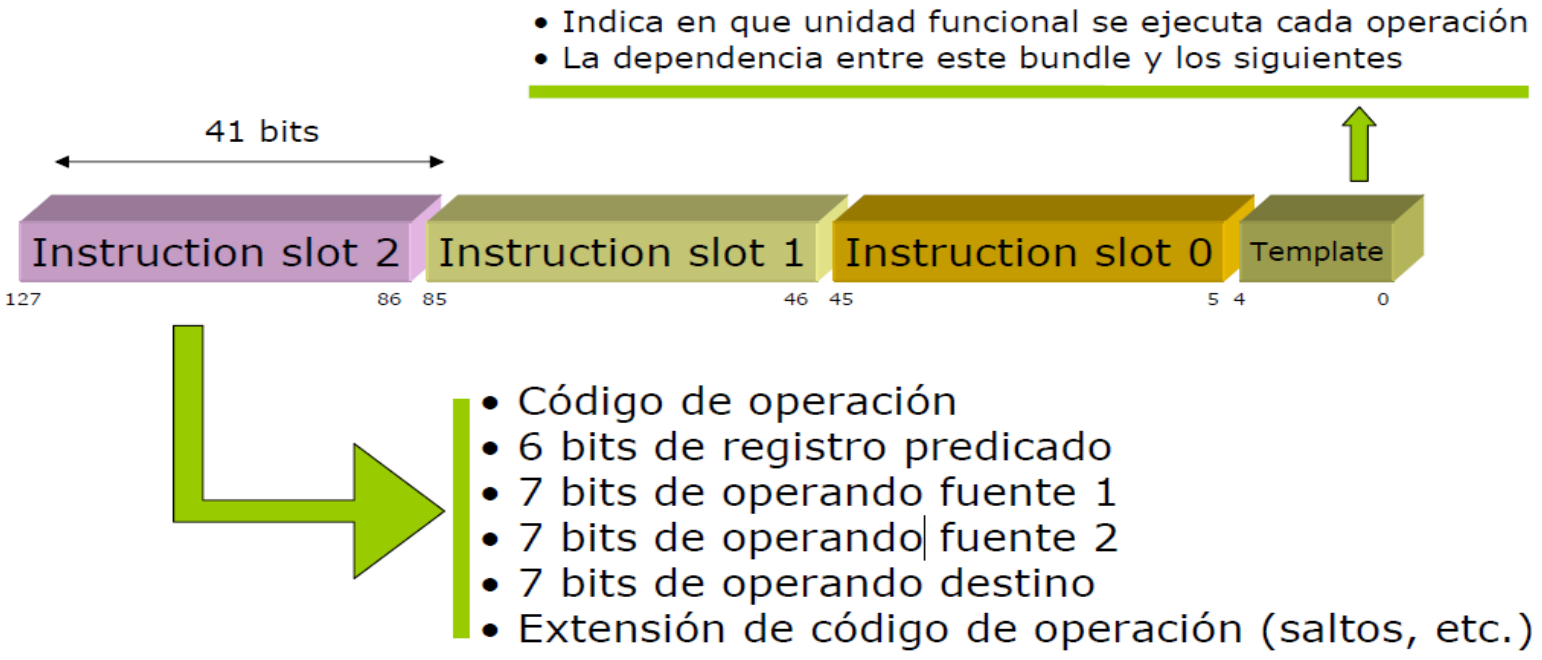
Un banco de reg. Por unidad funcional

VLIW – Very Long Instruction Word

EVOLUCIÓN

- **EPIC (*Explicit Parallel Instruction Computer*)**
  Intel y HP
- **Intel IA-64**
- **Itanium Merced**

Bundle (paquete): Conjunto de **3 instrucciones** y un **template**



Recordatorio

Tanto los Procesadores Vectoriales, los Procesadores Escalares, los Procesadores VLIW y las técnicas Multithread, están sujetos a: