Arquitectura-De-Computadores-Tema-1.pdf

Arrebato

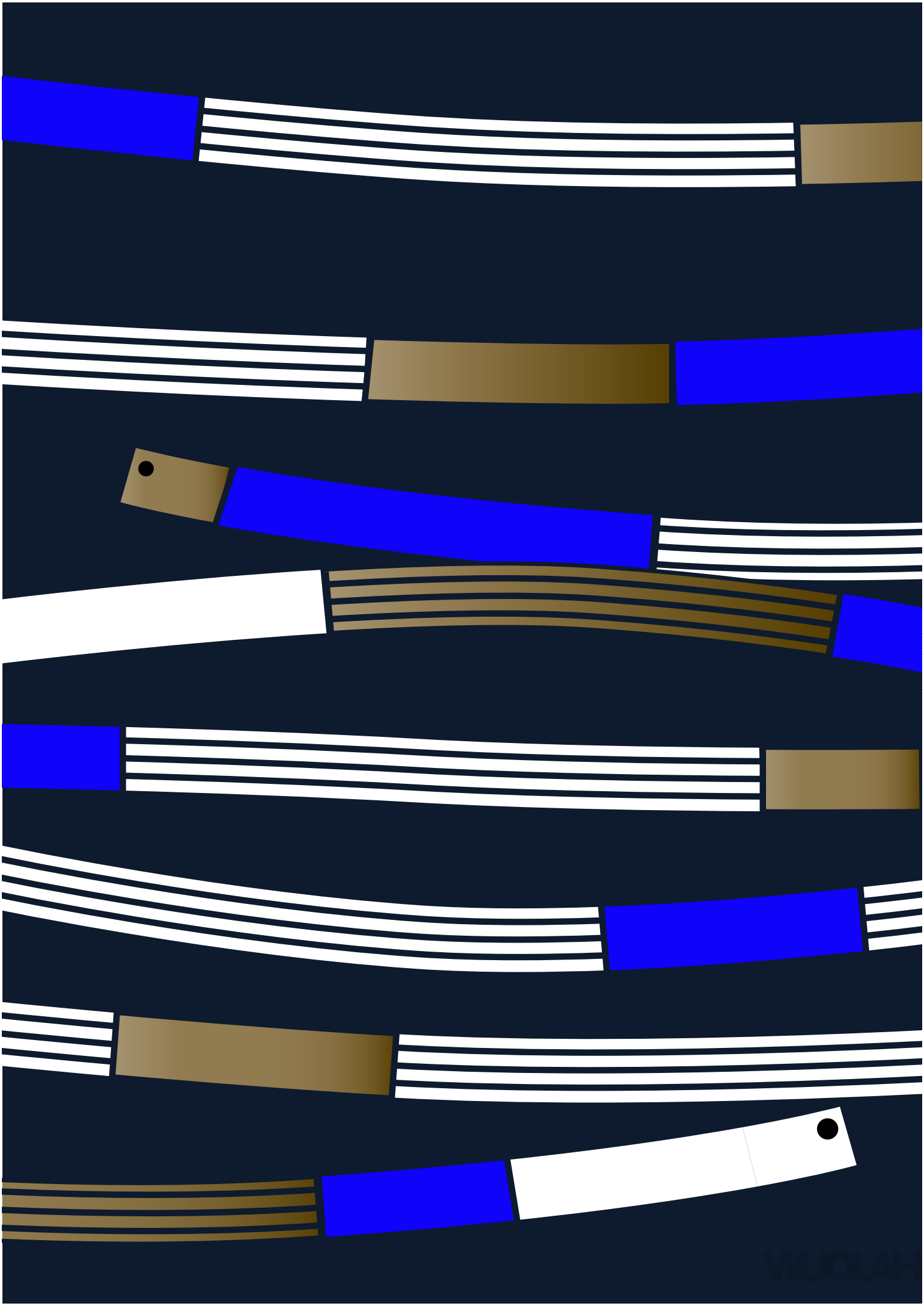
Arquitectura de Computadores

2º Grado en Ingeniería Informática

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
Universidad de Granada

KEEP CALM AND ESTUDIA UN POQUITO

WUOLAH + #QuédateEnCasa



Arquitectura de Computadores

Julio Ortega Lopera Teoría
Mancia Anguita Prácticas

Teoría	Prácticas
60%	40%
min (40%) 2'4	1'6
Actividades 4pts · Examen Ord 2pts	Entregas 2pts · Examen Global 2pts

FUNDAMENTOS Y PROBLEMAS DE ARQUITECTURA DE COMPUTADORES.

AUTORES : Mancia Anguita López / Julio Ortega Lopera

Editorial Técnica Avicam

LIBRERÍA FLEMING AVENIDA DE MADRID, 12

E-Mail: **ciencias@libreriafleming.com**

TLF. 958280183 / 654387692

Tema 1.- Arquitecturas Paralelas: Clasificación y Prestaciones

Lección 1

Dependencias de datos

Dos bloques B_1 y B_2 serán **dependientes** si:

- hacen referencia a una misma pos. de memoria
- B_1 aparece antes que B_2 en el código

Tipos de dependencia:

Read After Write	Write After Write	Write After Read
RAW	**WAW**	**WAR**
$a = b + c$	$a = b + c$	$b = a + 1$
$d = a + c$	...	...
	$a = d + e$	$a = d + e$

Paralelismo implícito en una aplicación

Tareas

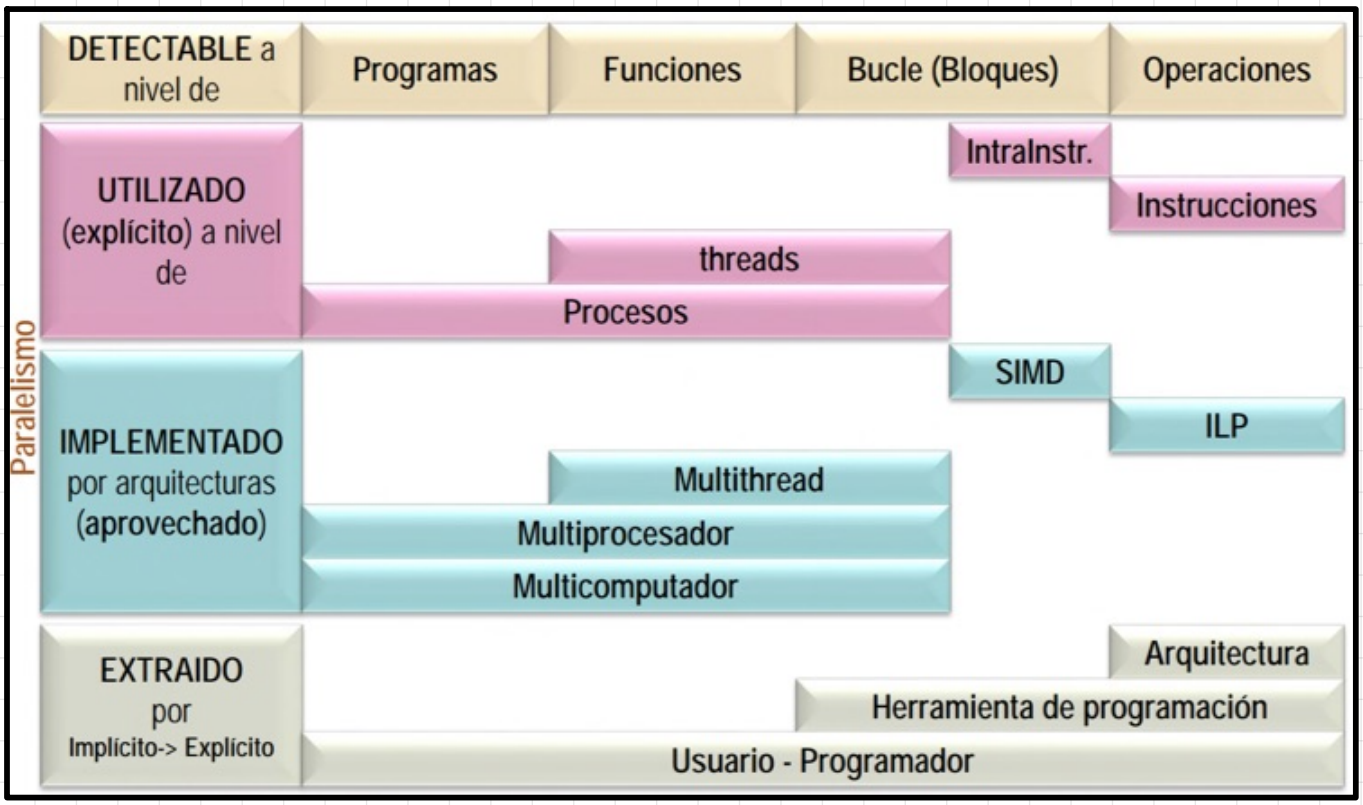
- Extrayendo estructura lógica de funciones
- $\approx$ paralelismo del nivel de función

Datos

- Implícito en operaciones con estructura de datos (vectores, matrices)
- Se puede sacar de la representación matemática
- $\pm$ = paralelismo nivel bucle

Paralelismo Explícito

Instrucciones	Threads o light process	Process o Process
Unidad de Control gestiona Instrucciones por unidad de procesamiento	Menor unidad de ejecución Menor Secuencia que se puede ejecutar en paralelo	Mayor Unidad Un proces consta de uno o varios threads



Lección 2.- Computación paralela y distribuida

Paralela

Múltiples cores/núcleos que actúan como una unidad

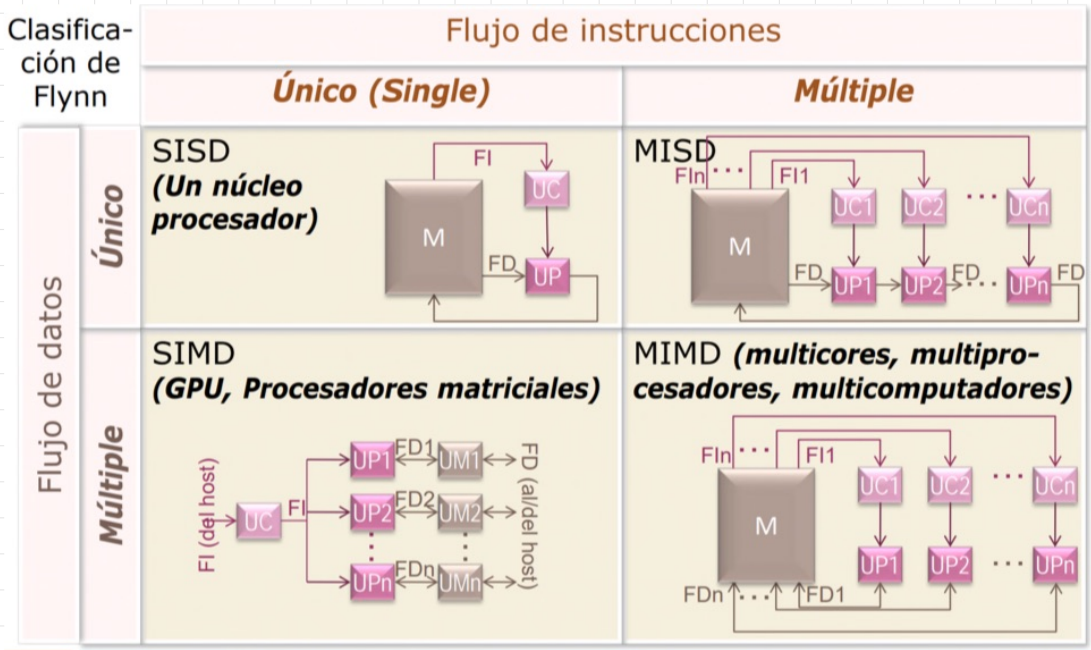
Distribuido

Colección de recursos autónomos en distintas localizaciones físicas

Clasificación de arquitecturas y sistemas paralelos

Los criterios pueden ser: Comerciales, educación, investigación.

Clasificación de Flynn => Flujo de control, Flujo de datos

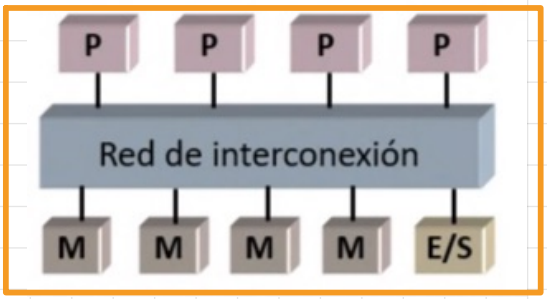


No suelen existir a no ser que sea propósito específico

Clasificación según memoria

Multiprocesadores

- Mismo espacio de direcciones
- Programador NO necesita saber donde están los datos

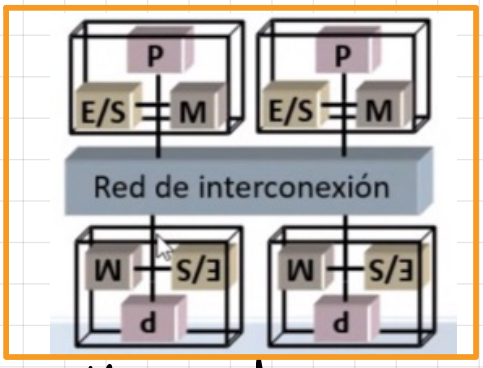


- Mayor Latencia (no hay transferencias paralelas)
- Poco escalable (aumentaría latencia)
- Comunicación implícita
  variables compartidas

- Necesita primitivas sincronización
- Distribución código / datos: no necesaria
- Programación más sencilla

Multicomputadores

- Espacio direcciones propio
- Programador sí necesita saberlo

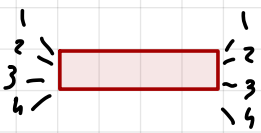


- Menor latencia
- Más escalable
- Comunicación explícita
  paso de mensajes

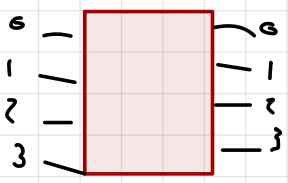
- Sincronización mediante mensajes
- Distribución código/datos compleja
- Programación difícil

Red de interconexión

Bus



Barras cruzadas

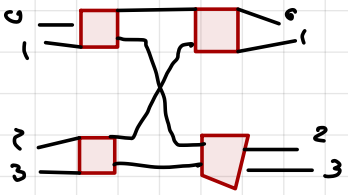


↓

Cualquier combinación si estuviese libre

Mucha complejidad

Multietapa

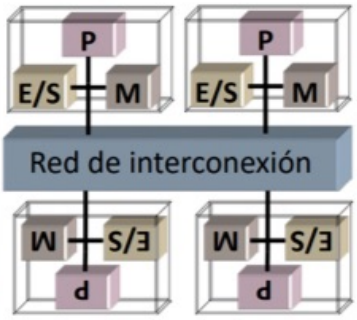
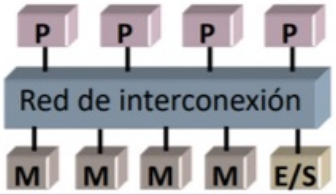


↓

Interconmutadores con branchwidth menor que barras cruzadas

Mayor latencia

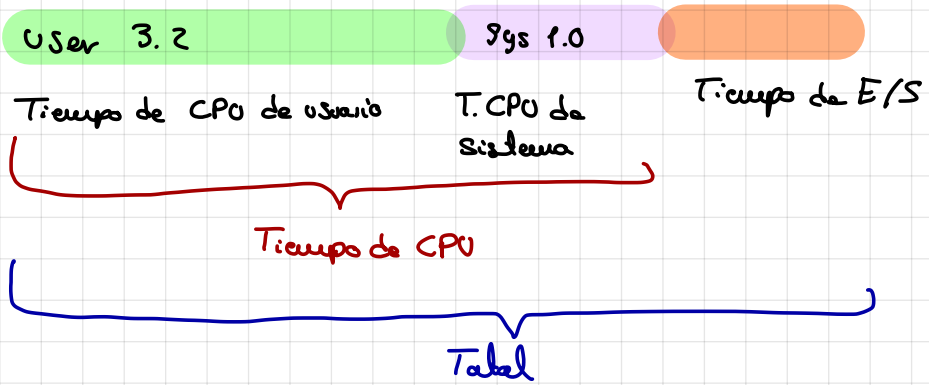
Resumen clasificación según memoria

Multi-computadores Memoria no compartida	**NORMA** *No Remote Memory Access*	*ej. cluster, red de computadores*	Memoria físicamente distribuida 	+ Nivel de empaquetamiento y conexión − + Escalabilidad −
Multi-procesadores Memoria compartida Un único espacio de direcciones	**NUMA** *Non-Uniform Memory Access*	NUMA		
		CC-NUMA		
		COMA		
	UMA *Uniform Memory Access*	**SMP** *Symmetric MultiProcessor*	Memoria físicamente centralizada 	

Lección 3.- Evaluación de prestaciones de una arquitectura

Medidas usuales para evaluar

Tiempo de Respuesta



Flujo de control — Total (elapsed) >= CPU

Varios flujos — Total < CPU ; Total ≥ $\frac{CPU}{n^{o}\,flujos}$

Tiempo de CPU

T_{CPU} = Ciclos De Programa × T_{ciclo} = $\frac{\text{Ciclos De Programa}}{\text{Frecuencia Reloj}}$

Ciclos Instrucción (CPI) = $\frac{\text{Ciclos Programa}}{\text{Nº Instrucciones (NI)}}$

$T_{CPU} = NI \times CPI \times T_{ciclo}$

Cuando hay n tipos de instrucciones distintos

$$\text{Ciclos de Programa} = \sum_{i} CPI_i * I_i$$

$$CPI_{\text{promedio}} = \frac{\sum_{i} CPI_i * I_i}{NI}$$

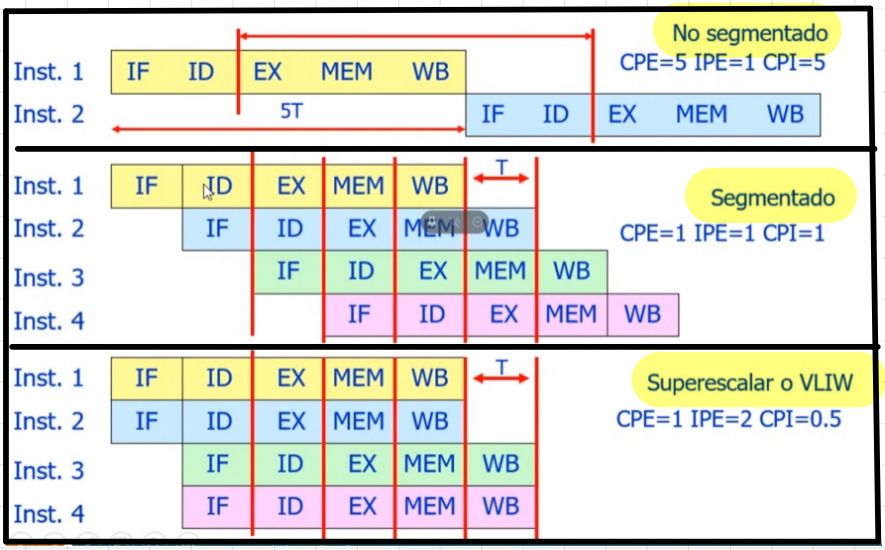
$$CPI = \frac{1}{IPC}$$

Procesadores que lanzan instrucciones paralelas

$$T_{CPU} = NI \times \frac{CPE}{IPE} \times T_{ciclo}$$

CPE = ciclos mínimos entre lanzamiento de instrucciones

IPE = Instrucciones por Emisión



También...

$T_{CPU} = \frac{N^{\circ}\text{oper}}{\text{Op-instruc}} \times CPI \times T_{ciclo}$

Nº operaciones / operaciones por instrucción

Dependencias en T_{CPU} = NI × CPI × T_{ciclo}

Tecnología — Estructura y Organización — Repertorio Instrucciones — Compilador

Productividad: MIPS y MFLOPS

MIPS millones de instrucciones por segundo

$$MIPS = \frac{NI}{T_{CPU} \times 10^6} = \frac{frecuencia}{CPI \times 10^6}$$

Depende del Repertorio de instrucciones (comparación difícil entre máquinas con distintas instrucciones)

Puede variar con el programa (no sirve para caracterizar la máquina)

Puede variar inversamente con las prestaciones (+MIPS −prestaciones)

MIPS Meaningless Indication of Processor Speed Chascarrillo

MFLOPS millones de operaciones en coma flotante

$$MFLOPS = \frac{Operaciones\_coma\,flotante}{T_{CPU} \times 10^6}$$

Mala comparación si no hay ops. en coma flotante

El conjunto de ops no es constante entre máquinas y la potencia varía según qué operación (precisión, suma ≠ multiplicación)

Necesidad de normalización de las instrucciones

Ganancia de Prestaciones

$$S_P = \frac{V_P}{V_1} = \frac{T_1}{T_P}$$

V_P: Velocidad mejorada — T_1: Tiempo ej. base
V_1: Velocidad base — T_P: Tiempo eje. mejorada

Ley de Amdahl

Dado un programa con código secuencial obligatorio, los procesadores no podrán usarse eficazmente.

$$S \leq \frac{P}{1 + f(P-1)}$$

S mejora de velocidad
f fracción de tiempo que no puede aprovechase la mejora
P factor de mejora de un recurso

Ejemplo

Programa 25% uso de coma flotante. Coma flotante se mejora al doble de velocidad

$P = 2$
$f = 0'75$

$$S \leq \frac{2}{1+0'75(2-1)} = 1'14$$

Hay que mejorar el caso más frecuente (lo que más se usa)