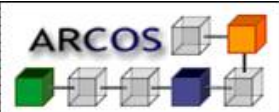
 <p>Universidad Carlos III de Madrid</p>	<p>Departamento de Informática Grado en Ingeniería Informática Arquitectura de computadores</p> <p>Examen Parcial G81-82 6 de marzo de 2012</p>	
---	--	---

Nombre y apellidos.....

Grupo:

Pregunta 1 (1,5 puntos).

Los procesadores Intel Pentium 4 (comercializado en 2003) e Intel Core i7 (comercializado en 2009) tienen los siguientes valores de tres parámetros de la arquitectura.

Tipo de parámetro	Intel Pentium 4	Intel Core i7
Número de transistores	55 millones	774 millones
Tamaño del procesador	101 mm ²	296 mm ²
Frecuencia del procesador	3.4 GHz	3.2 GHz

Razona **justificadamente** para cada caso, las causas del aumento, constancia o disminución de cada parámetro entre estas dos arquitecturas.

SOLUCIÓN:

- **Número de transistores:** el motivo es la ley de Moore que predice que el número de transistores se duplica cada 18 meses aproximadamente.
- **Tamaño del procesador:** el motivo es la mejora en el proceso de fabricación del procesador. Ahora es posible desarrollar obleas de silicio más grandes y con menos fallos por área por lo que resulta rentable producir procesadores con un mayor área.
- **Frecuencia del procesador:** el motivo del estancamiento de la frecuencia se debe a problemas de disipación de energía. Una frecuencia mayor implica un calor mucho mayor producido que puede no ser disipado (originando la destrucción del procesador).

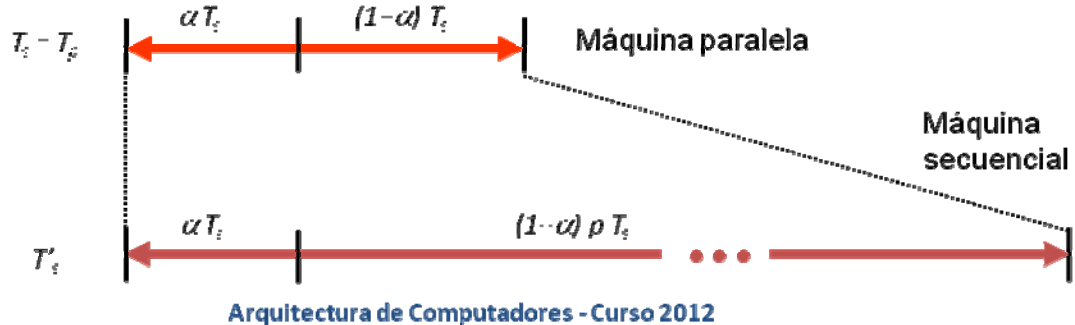
Pregunta 2 (1.5 puntos).

Define la Ley de Gustafson. ¿Para qué tipo de aplicaciones paralelas se aplica?

SOLUCIÓN:

La cantidad de trabajo que se puede hacer en paralelo varía linealmente con el número de procesadores. Con más procesadores se pueden acometer problemas de mayor coste computacional. Es decir, problemas mayores. La ley de Gustafson asume que la parte secuencial (no paralelizable) disminuye con el tamaño del problema dado que es la parte paralela la que aumenta con el número de procesadores.

$$S_p = \frac{T'_s}{T_p} = \frac{\alpha T_s + (1-\alpha)pT_s}{T_s} = p + \alpha(1-p)$$



Pregunta 3 (2.5 puntos).

Dado un computador con las siguientes características:

- CPU con frecuencia de 1GHz (1ns de periodo), CPI de 0.5, 60% de las instrucciones son cargas. El programa no tiene instrucciones de almacenamiento.
- Caché L1 Instrucciones: Tasa de fallos del 20%. Tiempo de acceso de 1 ciclo. El 0% de los bloques está modificado (*dirty bit* a 1).
- Caché L1 Datos: Tasa de fallos del 15%. Tiempo de acceso de 2 ciclos. El 0% de los bloques está modificado (*dirty bit* a 1).
- Caché L2 unificada instrucciones y datos. Tasa de fallos del 10%. Tiempo de acceso de 5 ciclos. El 30% de los bloques (tanto de instrucciones y como de datos) está modificado (*dirty bit* a 1).
- Memoria. Tiempo de acceso de 100 ciclos. **No existe** buffer de escritura (*write buffer*) entre la caché L2 y la memoria.

Se pide:

1. Calcular de forma justificada la expresión que muestra el tiempo medio de acceso a memoria. Asuma que la memoria caché está inicialmente llena.
2. Basándose en los resultados del apartado a anterior, calcular el tiempo de ejecución de un programa con 1000 instrucciones.
3. Razona cómo afectaría al ejercicio la presencia de un buffer de escritura (*write buffer*) entre la caché L2 y la memoria.

SOLUCIÓN

1.-

$$\begin{aligned} T_{Med}^{Instr} &= TL1^{Instr} + TFL1^{Instr} * (TL2 + TFL2 * TM) + TFL1^{Instr} * TFL2 * 0.3 * TM \\ &= 1 + 0.2 * (5 + 0.1 * 100) + 0.2 * 0.1 * 0.3 * 100 \\ T_{Med}^{Dat} &= TL1^{Dat} + TFL1^{Dat} * (TL2 + TFL2 * TM) + TFL1^{Dat} * TFL2 * 0.3 * TM \\ &= 2 + 0.15 * (5 + 0.1 * 100) + 0.15 * 0.1 * 0.3 * 100 \end{aligned}$$

El último término de ambas expresiones tiene en cuenta el tiempo que se invierte en escribir en memoria los bloque modificado que son reemplazados.

2.-

$$T = 1000 * 0.5 * 1ns + 1000 * 0.6 * T_{Med}^{Dat} + 1000 * T_{Med}^{Instr}$$

3.- Un buffer de escritura evitaría añadir un tiempo extra para escribir el bloque a memoria. Es decir, eliminaría el último término de las expresiones del apartado 1.

Pregunta 4 (2 puntos).

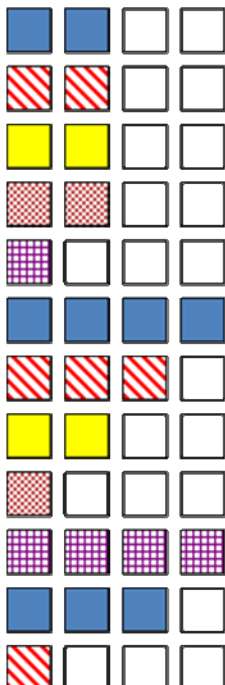
Dos técnicas destinadas a aumentar el número de instrucciones por ciclo ejecutadas en procesadores superescalares son la **multi-hilo de grano fino** y la **multi-hilo simultáneo (SMT)**. Se pide:

- Defina en qué consiste cada una de ellas.
- Indique las ventajas de cada una de ellas.

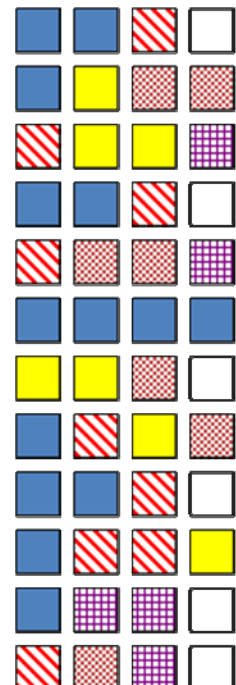
SOLUCIÓN:

- **Multi-hilo de grano fino:** ejecución de varios hilos cambiando cada hilo en cada instrucción/ciclo. Normalmente se hace distribución round-robin.
Ventaja: Puede ocultar detenciones cortas y largas.
Desventaja: Retrasa la ejecución de hilos individuales por reparto.
- **Multi-hilo simultáneo:** se pueden ejecutar varios hilos simultáneamente (mismo ciclo de reloj).
Ventaja: Mayor aprovechamiento de los recursos. Mayor ILP.

Grano fino



SMT



■ Hilo 1

▨ Hilo 2

■ Hilo 3

▩ Hilo 4

▦ Hilo 5

□ Inactivo

Pregunta 3 (2.5 puntos).

Dado el siguiente fragmento de código.

```
Bucle:      LD      F1,0(R2)
            LD      F2,0(R3)
            MULTD   F3,F1,F2
            ADDD    F4,F3,F1
            SD      F4,0(R3)
            ADDI    R2,R2,#8
            ADDI    R3,R3,#8
            SUB     R20,R4,R3
            BNZ     R20,Bucle
```

Cada instrucción tarda 1 ciclo en ejecutarse y algunas tienen una latencia extra (en ciclos) que se indica en la tabla siguiente. La tabla también muestra el intervalo de iniciación asociado al hardware usado por cada instrucción (acceso a memoria para LD y SD), componentes de la ALU para MULTD, ADDD, ADDI, SUB y BNZ.

Instrucción	LD	MULTD	ADDD	SD	ADDI	SUB	BNZ
Latencia	1	4	2	1	0	0	1
Intervalo de iniciación	2	5	3	2	1	1	2

Se pide:

- Indique las dependencias de datos existentes en el código. Indicando los registros que provocan cada dependencia.
- Asumiendo que una arquitectura superescalar de **dos vías** es decir, existen dos *execution pipelines* cada uno capaz de cargar (*fetch*), decodificar, ejecutar y acceder a memoria de forma independiente. **Notar que conforme a los valores de los intervalos de iniciación ni la memoria ni ningún componente de la ALU están segmentados.** Asumir que los resultados se pueden enviar inmediatamente de una unidad de ejecución a la otra (o a ella misma). El hardware **permite la emisión de instrucciones fuera de orden.** Se pide: **desenrollar el bucle dos iteraciones**, indicar la secuencia de ejecución de instrucciones y calcular el número de ciclos que el programa tarda en ejecutarse.

SOLUCIÓN

Dependencias de datos

```
LD      F1,0(R2)
LD      F2,0(R3)
MULTD   F3,F1,F2 -> raw con F1 y F2
ADDD    F4,F3,F1 -> raw con F3 y F1, waw con F3
SD      F4,0(R3) -> raw con F3(ADD)
ADDI    R2,R2,#8 -> war con R2(LD)
ADDI    R3,R3,#8 -> war con R3(LD y SD)
SUB     R20,R4,R3 -> raw con R3 (ADDI)
BNZ     R20,Bucle -> war con R20 (SUB)
```

Desenrollamiento de bucles

Bucle:	LD	F1,0(R2)	LD	F2,0(R3)
	Stall		Stall	
	LD	F1b,0(R2+8)	LD	F2b,8(R3)
	MULTD	F3,F1,F2	Stall	
	Stall		MULTD	F3b,F1b,F2b
	Stall		Stall	
	Stall		Stall	
	ADDI	R2,R2,#16	ADDI	R3,R3,#16
	ADDD	F4,F3,F1	stall	
	Stall		ADDD	F4b,F3b,F1b
	Stall		Stall	
	SD	F4,-16(R3)	SUB	R20,R4,R3
	BNZ	R20,Bucle	SD	F4b,-8(R3)
	Stall		Stall	