



FEBRERO 2007 (Parcial Mañana)

1).- Dada la estructura de unidades funcionales (FD_i : unidades de fetch y decodificación; EJ_1 y EJ_2 : unidades de ejecución de enteros, EJ_3 : unidad de ejecución en coma flotante; ER_i : unidades de escritura de resultado) de la figura que hay a continuación, y la secuencia de instrucciones $I_1 \dots I_{10}$, que tiene las siguientes particularidades:

- Todas las instrucciones tardan un ciclo de reloj en cada fase, excepto I_2 e I_5 cuyas ejecuciones duran 3 ciclos y las instrucciones I_7 e I_7 que duran 2.
- I_2 , I_4 e I_6 operan con números reales.
- Existen las siguientes dependencias reales entre instrucciones:

I_3 depende de I_1

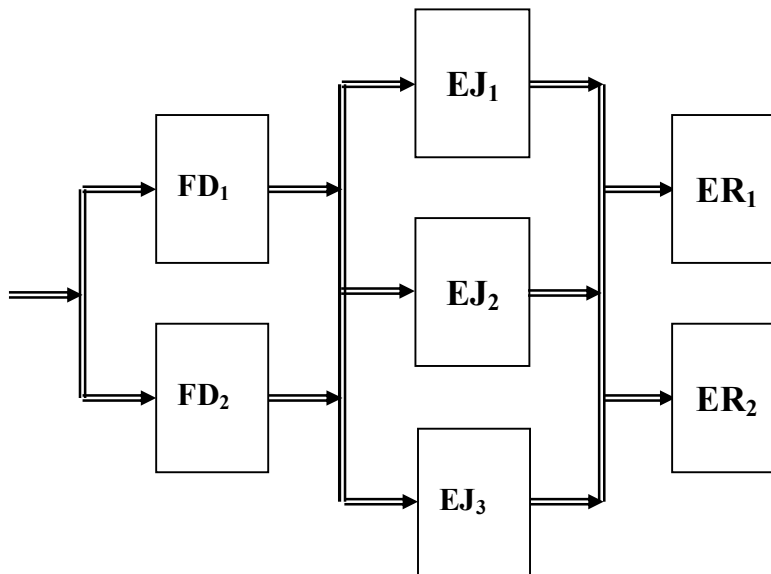
I_6 depende de I_4

I_9 depende de I_8

Se pide:

- Mostrar mediante una tabla la secuencia de ejecución de las instrucciones, en el caso de que no se permita **ningún tipo de desordenamiento** (1,5 pts.)
- Repetir el apartado anterior para el caso de que se permita **desorden, tanto en ejecución, como en escritura de resultados.**(1,5 pts)

NOTA: En caso de coincidencia en una transición entre dos etapas por parte de dos instrucciones, tendrá prioridad la que se encuentre primero en la secuencia del código (subíndice menor).





1 cont.)-

SOLUCIÓN

Ejecución ordenada:

Ciclo	FD ₁	FD ₂	EJ ₁	Ej ₂	EJ ₃	ER ₁	ER ₂
1	I ₁	I ₂					
2	I ₃	I ₄	I ₁		I ₂		
3					I ₂	I ₁	
4	I ₅		I ₃		I ₂		
5	I ₆	I ₇	I ₅		I ₄	I ₂	I ₃
6			I ₅			I ₄	
7	I ₈	I ₉	I ₅	I ₇	I ₆		
8	I ₁₀		I ₈	I ₇		I ₅	I ₆
9						I ₇	I ₈
10			I ₉	I ₁₀			
11			I ₉				
12						I ₉	I ₁₀

Ejecución desordenada:

Ciclo	FD ₁	FD ₂	Ventana	EJ ₁	Ej ₂	EJ ₃	ER ₁	ER ₂
1	I ₁	I ₂						
2	I ₃	I ₄	I ₁ , I ₂	I ₁		I ₂		
3	I ₅	I ₆	I ₃ , I ₄			I ₂	I ₁	
4	I ₇	I ₈	I ₃ , I ₄ , I ₅ , I ₆	I ₃	I ₅	I ₂		
5	I ₉	I ₁₀	I ₄ , I ₆ , I ₇ , I ₈	I ₇	I ₅	I ₄	I ₂	I ₃
6			I ₆ , I ₈ , I ₉ , I ₁₀	I ₇	I ₅		I ₄	
7			I ₆ , I ₈ , I ₉ , I ₁₀	I ₈	I ₁₀	I ₆	I ₅	I ₇
8			I ₉				I ₆	I ₈
9			I ₉	I ₉			I ₁₀	
10				I ₉				
11							I ₉	



2).-

Un microprocesador que posee un fichero de registros “ve” en cada momento los siguientes registros de enteros:

- R₀ a R₇ para los parámetros que le ha pasado la rutina anterior.
- R₈ a R₁₅ para las variables locales.
- R₁₆ a R₂₃ para pasar parámetros a las rutinas que llame.
- R₂₄ a R₃₁ para variables globales.

Se pide:

1. ¿Cuántos registros de enteros hay implementados en el micro, sabiendo que el fichero de registros contiene 8 niveles de anidamiento de rutinas?. Justifique brevemente la respuesta. (0,7 ptos.)
2. ¿De los registros R₄, R₁₁, R₁₈ y R₂₆ seguirían siendo visibles después de ejecutar una llamada a una subrutina?. ¿En qué numero de registro estarían ahora los que siguieran siendo visibles? (0,4 ptos.)
3. Partiendo de las condiciones iniciales, repetir el punto anterior suponiendo ahora que la instrucción que se ejecuta es un retorno de subrutina en vez de una llamada. (0,4 ptos.)

SOLUCIÓN

Apartado 1

Registros para parámetros:	8x8 = 64
Registros para variables locales	8x8 = 64
Registros para variables globales	= 8
TOTAL	136

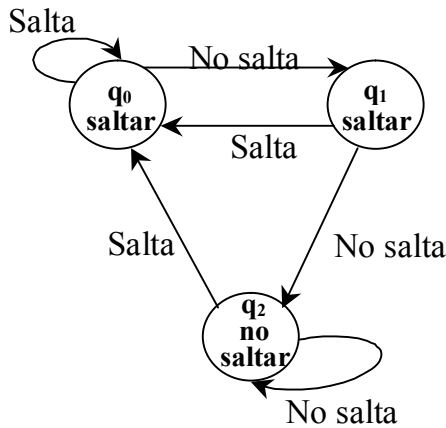
Apartado 2

Apartado 3

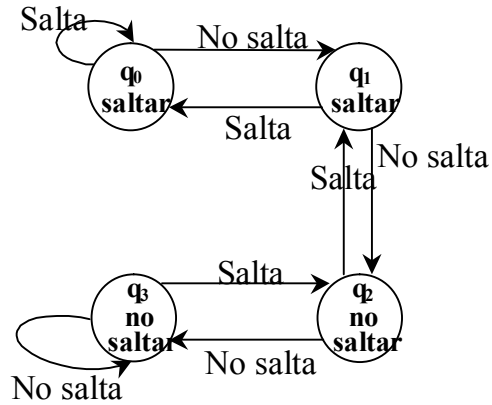
R ₄	No se ve	Se ve como R ₂₀
R ₁₁	No se ve	No se ve
R ₁₈	Se ve como R ₂	No se ve
R ₂₆	Se ve como R ₂₆	Se ve como R ₂₆



3).- Se pretende diseñar un sistema de predicción de saltos y para ello se dispone de los dos autómatas representados a continuación.



Autómata.1



Autómata.2

Suponiendo que el criterio estático que se usará es ambos casos el mismo y que los sitúa inicialmente en el estado q_0 . ¿Cuál de las dos opciones presenta un mejor resultado para una determinada instrucción de salto condicional que, en sus cinco primeros ciclos de ejecución, presenta el comportamiento siguiente?

Se considera imprescindible justificar la respuesta. (1,5 pts)

Ciclo	Resultado Ejecución
1	salta
2	no salta
3	salta
4	no salta
5	no salta

SOLUCIÓN

Ciclo	Resultado Ejecución	Autómata 1			Autómata 2		
		Estado actual	Predicción	Estado siguiente	Estado actual	Predicción	Estado siguiente
1	salta	q_0	salta	q_0	q_0	salta	q_0
2	no salta	q_0	salta (x)	q_1	q_0	salta (x)	q_1
3	salta	q_1	salta	q_0	q_1	salta	q_0
4	no salta	q_0	salta (x)	q_1	q_0	salta (x)	q_1
5	no salta	q_1	salta (x)	q_2	q_1	salta (x)	q_2
		Errores:		3	Errores:		3

Teniendo en cuenta la secuencia propuesta, ambas opciones proporcionan idéntico resultado, luego **no permitiría optar por ninguna de las dos alternativas.**



4).-

A. Si para ejecutar en un procesador que usa la técnica del salto retardado se presenta la siguiente secuencia de instrucciones en el programa fuente:

- [I₁] Instrucción que no es de salto
- [I₂] Salto condicional a eti1
- [I₃] Salto condicional a eti2
- [I₄] Instrucción que no es de salto

¿Cómo quedaría la secuencia de instrucciones en el ejecutable para que el compilador garantice que no se producen errores de ejecución? (1 pto.)

¿Cambiaría en algo la situación si la I₂ fuese de salto incondicional en lugar de condicional? (0,5 ptos)

Se considera imprescindible justificar la respuesta.

Asumiendo que la instrucción I₂ no depende de I₁, el compilador podrá modificar su orden, sin embargo no podrá hacer lo mismo con I₃ puesto que aparecen dos saltos seguidos lo que impide aplicar esta norma al segundo de ellos, en este caso el compilador deberá usar la opción de insertar un NOP. La secuencia quedaría:

I₂, I₁, I₃, NOP, I₄

*La técnica de salto retardado se aplica tanto a saltos condicionales como a incondicionales, por lo tanto **el resultado**, en cuanto al orden de instrucciones que fija el compilador, **sería el mismo**.*

B. Responda justificadamente a las siguientes cuestiones.

- ¿A qué tipo o tipos de conflictos (acceso a recursos, desajustes de tiempo, dependencia de operandos y de saltos) es menos vulnerable una arquitectura RISC en la ejecución de instrucciones? (0,5 ptos)

Por la naturaleza de las instrucciones RISC, a los desajustes de tiempo, puesto que la uniformidad de estas instrucciones tenderá a igualar sus tiempos de proceso en las distintas fases de ejecución. EL resto de conflictos se pueden producir igualmente. Por otro lado, simplifica también el tratamiento de la dependencia de operandos al no admitir, en general, operandos en memoria en instrucciones de proceso.

- ¿Qué características específicas tiene el nivel L1 de caché en los microprocesadores Intel a partir del Pentium? (0,5 ptos)

Se divide en dos bloques de aplicación específica, uno para instrucciones y otro para datos, conectándose respectivamente a las unidades de fetch y ejecución y contribuyendo a reducir el conflicto de acceso a memoria.



5).-

- Dado el siguiente fragmento de programa escrito en lenguaje simbólico, indicar todas las dependencias y pseudodependencias que tiene, especificando para cada una de ellas lo siguiente: entre que instrucciones se genera, con relación a que registro se produce y de qué tipo de dependencia o pseudodependencia se trata. (0,7 ptos)
- Suponiendo que las instrucciones se ejecutan en un procesador superescalar de forma completamente ordenada (tanto en emisión como en actualización de resultado) ¿Cuáles de las situaciones descritas anteriormente supondrían conflicto? Razone la respuesta. (0,4 ptos)
- Suponga ahora que la ejecución se realiza con orden en emisión pero con escritura de resultados desordenada ¿Cambia en algo el resultado del apartado anterior? (0,4 ptos)

I₁: R₁·R₂→R₄
I₂: R₃-R₅→R₆
I₃: R₄→R₅
I₄: R₆→R₂
I₅: R₄+R₆→R₄

SOLUCIÓN

Instrucciones	Registro	Tipo de dependencia
I ₁ e I ₃	R ₄	Dependencia real o de escritura/lectura
I ₂ e I ₄	R ₆	Dependencia real o de escritura/lectura
I ₂ e I ₅	R ₆	Dependencia real o de escritura/lectura
I ₁ e I ₄	R ₂	Antidependencia o de lectura/escritura (pseudodependencia)
I ₂ e I ₃	R ₅	Antidependencia o de lectura/escritura (pseudodependencia)
I ₃ e I ₅	R ₄	Antidependencia o de lectura/escritura (pseudodependencia)
I ₁ e I ₅	R ₄	Pseudodependencia de salida o de escritura/escritura

Si la ejecución es completamente ordenada, los únicos conflictos que se presentarían son los de dependencia real (los tres primeros de la tabla anterior).

Si la ejecución sólo admite desorden en escritura, los conflictos a tener en cuenta serían los de dependencia real y la pseudodependencia de salida (los tres primeros y el último de la tabla).



FEBRERO 2007 (Parcial y Final Tarde)

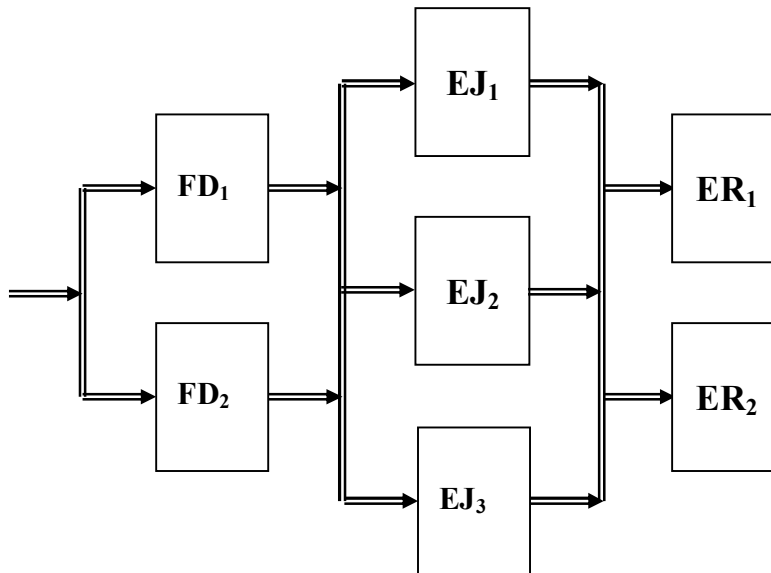
1).- Dada la estructura de unidades funcionales (FD_i : unidades de fetch y decodificación; EJ_1 y EJ_2 : unidades de ejecución de enteros, EJ_3 : unidad de ejecución en coma flotante; ER_i : unidades de escritura de resultado) de la figura que hay a continuación, y la secuencia de instrucciones $I_1 \dots I_{10}$, que tiene las siguientes particularidades:

- Todas las instrucciones tardan un ciclo de reloj en cada fase, excepto I_3 e I_5 cuyas ejecuciones duran 3 ciclos y las instrucciones I_2 , I_7 e I_8 que duran 2.
- I_3 , I_5 e I_7 operan con reales, por lo tanto, deben ejecutarse exclusivamente en la unidad EJ_3 .
- Existen las siguientes dependencias reales entre instrucciones:
 - I_5 depende de I_4
 - I_8 depende de I_7
 - I_9 depende de I_8

Se pide:

- Mostrar mediante una tabla la secuencia de ejecución de las instrucciones, en el caso de que no se permita **ningún tipo de desordenamiento** (0,75/1,5 pts.)
- Repetir el apartado anterior para el caso de que se permita **desorden, tanto en ejecución, como en escritura de resultados.** (0,75/1,5 pts.)

NOTA: En caso de coincidencia en una transición entre dos etapas por parte de dos instrucciones, tendrá prioridad la que se encuentre primero en la secuencia del código (subíndice menor).





2).-

Sabiendo que en un momento determinado 'se ven' desde una subrutina 80 registros (32 de ellos globales) y que el fichero de registros permite anidar hasta 8 niveles de subrutinas, indica la estructura del fichero de registros diferenciando cuántos registros se destinan a variables globales, locales y parámetros (recibidos y enviados al siguiente nivel). Debe indicarse expresamente el número total de registros. (0,75/1,5 ptos.)

NOTA: Suponer que la zona de variables locales y parámetros tienen el mismo tamaño

SOLUCIÓN: Si desde un nivel 'se ven' 80 registros y 32 son globales, hay 48 registros para variables locales y parámetros. Dado que la zona de variables locales y parámetros tienen el mismo tamaño, tendremos 16 bytes para variables locales, 16 para parámetros recibidos y 16 para parámetros enviados. El número total de registros será de $32 + 32 * 8 = 288$ registros.

B) Supóngase que en un microprocesador con salto retardado tenemos el siguiente fragmento de un programa **fuentes** en ensamblador:

[I₁] Instrucción que no es de salto

[I₂] Instrucción que no es de salto

Eti1 [I₃] Salto condicional a eti2

[I₄] Instrucción que no es de salto

¿En qué orden quedarán en el programa **ejecutable**? (0,75/1,5 ptos.)

SOLUCIÓN: Es necesario añadir un NOP porque se puede llegar a I₃ por otro camino distinto al secuencial, con lo cual quedará de la siguiente forma: [I₁], [I₂], [I₃], NOP, [I₄]



3).-

Dado el siguiente fragmento de programa escrito en lenguaje simbólico, indicar todas las dependencias y pseudodependencias que tiene, especificando para cada una de ellas lo siguiente: entre que instrucciones se genera, con relación a que registro se produce, que tipo de dependencia o pseudodependencia es y si se puede solventar, cual es la solución.

(0,8/1,5 Puntos)

- I₁:** R₁→R₅
- I₂:** R₅+R₃→R₃
- I₃:** R₄→R₅
- I₄:** R₄→R₆
- I₅:** R₂→R₄

SOLUCIÓN

Tipo de dependencia	Instrucc.	Reg.	¿Tiene solución?
Dependencia real o de escritura/lectura	I ₁ e I ₂	R ₅	No. Hay que esperar
Pseudodependencia de salida o de escritura/escritura	I ₁ e I ₃	R ₅	Sí. Renombrado de registros
Antidependencia o pseudodependencia de lectura/escritura	I ₂ e I ₃	R ₅	Sí. Renombrado de registros
	I ₃ e I ₅	R ₄	
	I ₄ e I ₅	R ₄	



4).- Indique si es cierta o falsa la siguiente afirmación, **justificando brevemente la respuesta:**

- Si un procesador usa el sistema de *Buffer de Bucles*, para atenuar parones en el pipeline, se pondrá en marcha con cada instrucción de salto condicional. (0,6/1,25 Puntos)

FALSO: No es suficiente que la instrucción sea de salto condicional, pues además debe ser un salto hacia atrás.

- En los microprocesadores Pentium Pro se dispone de ejecución desordenada de instrucciones, pero de escritura ordena. (0,6/1,25 Puntos)

CIERTO: Las instrucciones se dividen en microinstrucciones que se almacenan en la 'Instruction Pool' y se ejecutan de forma desordenada. Posteriormente la 'Retire unit' escribe los resultados de forma ordenada y retira las microinstrucciones de dicha 'Instruction Pool'.



5).-

A) De un procesador vectorial se conocen los siguientes datos:

- Frecuencia=250MHz.
- Número y tipo de unidades vectoriales: una de carga/almacenamiento, una de suma, una de multiplicación y una de división.
- Tiempos de arranque:
 - ❖ Instrucciones de carga y almacenamiento vectorial: 12 ciclos de reloj.
 - ❖ Instrucciones de suma vectorial: 6 ciclos de reloj.
 - ❖ Instrucciones de multiplicación vectorial: 7 ciclos de reloj.
 - ❖ Instrucciones de división vectorial: 20 ciclos de reloj.
- Tiempo de bucle vectorial: 15 ciclos de reloj.
- Longitud máxima de vector = 128.
- No permite encadenamiento de convoyes.

Para la ejecución del siguiente fragmento de programa, calcular:

1. El número de convoyes que se lanzan, indicando además, para cada uno de ellos, las instrucciones que lo componen y los tiempos de arranque a computar. (0,4 ptos.)
2. R_{∞} . (0,4 ptos.)

a→F0	;	[I1]
(Rx)→V1	;	[I2]
F0*V1→V2	;	[I3]
(Ry)→V3	;	[I4]
V2+V3→V4	;	[I5]
V4→(Ry)	;	[I6]

SOLUCIÓN

Apartado 1

Convoy 1: Instrucción I2. Tiempo de arranque=12 ciclos.

Convoy 2: Instrucciones I3 e I4. Tiempo de arranque=Max(7,12)=12 ciclos.

Convoy 3: Instrucción I5. Tiempo de arranque=6 ciclos.

Convoy 4: Instrucción I6. Tiempo de arranque=12 ciclos.

Apartado 2

Por otra parte, como en el fragmento se producen $2*n$ operaciones en coma flotante, tenemos:



5 cont.)-

$$R_n = \frac{2 \cdot n \cdot \text{Frec}}{\left[\frac{n}{MVL} \right] \cdot (\sum T_{arr} + T_{loop}) + n \cdot T_{camp}} = \frac{500 \cdot n}{\left[\frac{n}{128} \right] \cdot (42 + 15) + 4 \cdot n}$$

Y el límite cuando n tiende a infinito será:

$$R_\infty = \frac{500 \cdot n}{\frac{57}{128} \cdot n + 4 \cdot n} = \frac{500 \cdot 128}{57 + 4 \cdot 128} = 112,48 \text{ MFLOPS}$$

- B) Una determinada Universidad está considerando la adquisición de un ordenador vectorial, y le ofrecen las dos máquinas con un coste similar:

	Rendimiento vectorial (R_{vect})	Relación de rendimientos (r)
Máquina A	800 MFLOPS	10
Máquina B	600 MFLOPS	4

¿A partir de qué porcentaje de vectorización del programa, la máquina A tiene mayor rendimiento que la B?. (0,6 pts.)

SOLUCIÓN

$$R = \frac{1}{\frac{f_{esc}}{R_{esc}} + \frac{f_{vect}}{R_{vect}}} = \frac{1}{\frac{1-f}{R_{vect}} + \frac{f}{R_{vect}}} = \frac{R_{vect}}{r \cdot (1-f) + f} = \frac{R_{vect}}{r - f \cdot (r - 1)}$$

Para la máquina A el rendimiento será: $R = \frac{800}{10 - 9 \cdot f}$

Y para la B: $R = \frac{600}{4 - 3 \cdot f}$

Igualándolos, calculamos:

$$\frac{800}{10 - 9 \cdot f} = \frac{600}{4 - 3 \cdot f}; 3200 - 2400 \cdot f = 6000 - 5400 \cdot f$$

$$f = \frac{2800}{3000} = 0,9333 = 93,33\%$$

Por lo tanto, la máquina A tendrá mejor rendimiento para programas cuya relación de vectorización sea mayor que el 93,33%, y la máquina B para los que tengan menor porcentaje de vectorización.



6).- Indique si son ciertas o falsas las siguientes afirmaciones, **justificando brevemente la respuesta**::

- Las MIN's son un ejemplo de redes no bloqueantes. (0,4 ptos.)

FALSO. Las MIN's garantizan que todos los nodos de entrada se conecten con los de salida al menos por un camino, pero no que sea de forma simultánea

- El diámetro de una red depende del número de nodos que tenga.(0,4 ptos.)

FALSO. El diámetro de la red es el máximo de los caminos mínimos y depende esencialmente de la topología de la red.

- El tiempo de transmisión de una red es el que transcurre desde que el primer bit entra en la red hasta que llega al destino. (0,4 ptos.)

FALSO. El tiempo de transmisión es el cociente entre el tamaño del mensaje y el ancho de banda. El que aparece en el enunciado es el tiempo de escape.



7).- Indique si son ciertas o falsas las siguientes afirmaciones, **justificando brevemente la respuesta:**

- En cualquier protocolo de sondeo uno de los posibles estados de una línea es no válido. (0,4 ptos.)

FALSO. Depende del protocolo de sondeo, si es de invalidación (MESI, MSI) sí. Si es de actualización (Dragón) no.

- En un protocolo de coherencia de directorio mediante listas entrelazadas, para que un procesador puede modificar una línea compartida, su caché deberá colocarse como primera de la lista previa solicitud al directorio. (0,4 ptos.)

CIERTO. Sólo la primera de la lista puede realizar esa acción, si no lo es, la caché debe solicitarlo al directorio y, una vez en esa posición, enviar el mensaje de invalidación al resto de cachés.

- La operación de desalojo es común para todos los protocolos de directorio. (0,4 ptos.)

FALSO. Sólo se aplica en aquellos que utilizan mapeado limitado.



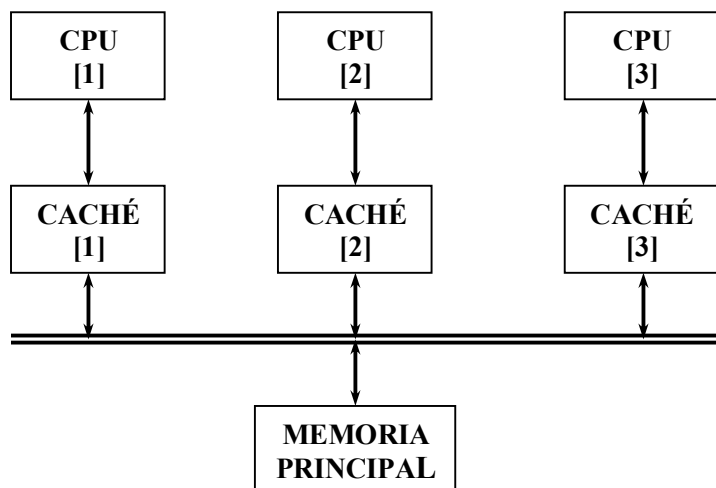
8).-

A).- En el sistema de la figura se sabe que las tres cachés siguen el protocolo de invalidación MESI, y que sobre una determinada línea de información, que inicialmente no está cargada en ninguna caché, las CPUs realizan la secuencia de accesos indicada en la tabla.

Se pide que se rellene el resto de dicha tabla indicando:

- ¿Qué operaciones se desencadenan en el bus?. Especificar también que “snoopy” lanza cada una de ellas.
- ¿En qué estado queda la línea en cada caché al terminar cada acceso de CPU?.

B).- En función de los diferentes estados en los que se puede encontrar una línea en una caché que sigue el protocolo de actualización Dragón, indique las acciones que debería llevar a cabo el controlador de caché si tuviese que descargarla y cómo afectaría este proceso a las otras posibles cachés del sistema.





8 cont).-

SOLUCIÓN

A)

Acceso	Operaciones sobre el bus	Estado final caché		
		[1]	[2]	[3]
Lectura CPU [3] (0.2 ptos.)	La caché[3] realiza una lectura en memoria con sondeo. Al no encontrarse la línea en ninguna caché, la cargará con estado "E".	--	--	E
Lectura CPU [1] (0.2 ptos.)	La caché[1] realiza una lectura en memoria con sondeo, la caché[3] responde activando S y cambia su línea a estado "S". La caché[1] carga la línea como "S".	S	--	S
Escritura CPU [2] (0.2 ptos.)	La caché[2] realiza una lectura en memoria con intención de modificación, sin esperar a la entrega de la línea, modifica la línea y cambia su estado a "M". Las otras dos cachés detectan el acceso e invalidan la línea.	I	M	I

B)

Acceso	Operaciones sobre el bus	Estado final caché		
		[1]	[2]	[3]
Lectura CPU [3] (0.2 ptos.)	La caché[3] realiza una lectura en memoria con sondeo. Al no encontrarse la línea en ninguna caché, la cargará con estado "E".	--	--	E
Lectura CPU [1] (0.2 ptos.)	La caché[1] realiza una lectura en memoria con sondeo, la caché[3] responde activando S y cambia su línea a estado "SC". La caché[1] carga la línea como "SC".	SC	--	SC
Escritura CPU [2] (0.2 ptos.)	La caché[2] realiza una lectura en memoria con sondeo, las otras dos responden activando S. La caché[2] carga la línea como "SM" y la modifica, volcando la actualización al Bus. Las otras dos cachés detectan el acceso y actualizan manteniendo "SC".	SC	SM	SC



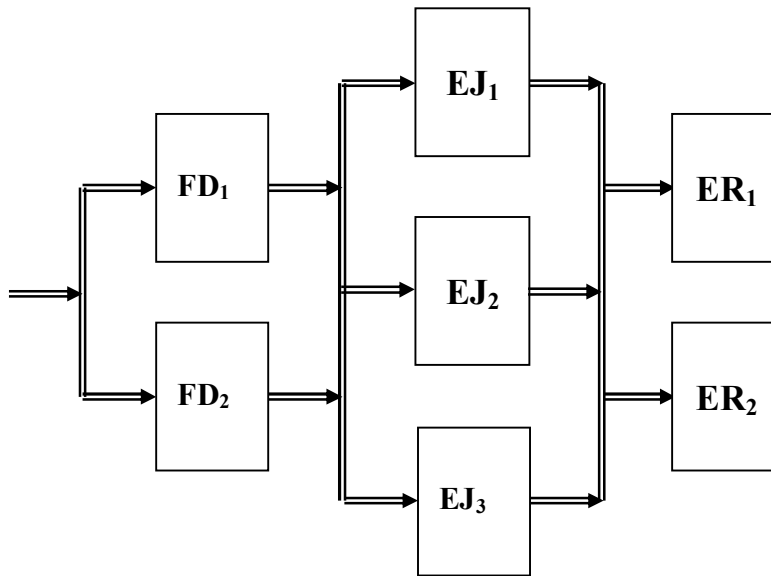
JUNIO 2007 (Parcial y Final Mañana)

1).- Dada la configuración superescalar de la figura que hay a continuación, y la secuencia de instrucciones $I_1, I_2, I_3, I_4, I_5, I_6, I_7, I_8, I_9, I_{10}, I_{11},$ e I_{12} , que tiene las siguientes particularidades:

- Todas las instrucciones tardan un ciclo de reloj en cada fase, excepto I_2, I_5 e I_{10} , cuyas fases de ejecución duran 2 ciclos de reloj, e I_4 e I_9 que dura 3 ciclos.
- Hay una dependencia de operandos entre I_2 e I_1 , y, por lo tanto, la ejecución de I_2 no puede comenzar hasta que termine la fase de escritura de resultados de I_1 . Lo mismo ocurre con respecto a I_5 e I_4 y con I_9 e I_6 .
- I_3, I_6, I_9 e I_{12} son de coma flotante y, por lo tanto, ellas y sólo ellas deben ejecutarse en EJ_3 .

Se pide:

1. Mostrar mediante una tabla la secuencia de ejecución **totalmente ordenada** de las instrucciones. (0,6 pts.)
2. Repetir de nuevo el proceso para el caso de que se permita el **desorden, tanto en la ejecución, como en la escritura de resultados**. (0,6 pts.)





2).-

Explique brevemente en qué consiste y cuándo se aplica la técnica de renombrado de registros. (0,4 ptos.)

Se aplica para resolver las pseudodependencias de operandos. Si no se puede modificar un registro, se almacena un resultado en otro registro que se 'renombrará' al original cuando este pueda ser modificado.

Indique cuáles son las características típicas de las instrucciones de una CPU RISC. (0,4 ptos.)

- *Conjunto de instrucciones limitado, sencillo y homogéneo (Set de instrucciones ortogonal).*
- *Unidad de control cableada vs. UC cableada.*
- *Elevado número de registros en la CPU.*
- *Las instrucciones (salvo lect. y escritura) no tienen operandos en memoria.*

¿Cuáles son las principales ventajas que aporta el nivel L1 de caché de un Pentium? (0,4 ptos.)

La caché trabaja a la misma velocidad de reloj que la CPU. Al disponer de cachés independientes de datos e instrucciones, se puede acceder a ambas de forma simultánea.



3).- Conflicto de dependencia de saltos. ¿Qué alternativas se plantean según el tipo de salto? Explique brevemente las distintas técnicas de mejora de rendimiento respecto a este conflicto. (0,8 ptos.)

Incondicionales: El salto se produce siempre. Detectarlo en fase de fetch o decodificación y modificar el PC con la dirección de salto.

Condicionales: No se puede asegurar si se va a saltar o no. Las técnicas usadas son:

Salto retardado: Se intercambia la instrucción de salto con la anterior. Mientras se salta se ejecuta la instrucción intercambiada.

Flujos múltiples: Etapas del pipeline duplicadas para poder seguir los dos caminos.

Precaptar destino del salto: Leer la siguiente instrucción y la de salto.

Buffer de bucles: Buffer de prefetch amplio. Se activa en los saltos condicionales hacia atrás.

Predicción de salto: Se pueden usar criterios estáticos o dinámicos:

Estáticos: Se aplica un criterio en función de la instrucción.

No cumple condición. Se presupone que no se cumple la condición.

Cumple condición. Se presupone que se cumple la condición.

Decisión según el código de operación. En función del tipo de salto se predice saltar o no saltar.

Evaluación previa de la condición. Se evalúa la condición previamente y se acepta el resultado como la decisión de saltar o no saltar.

Dinámicos: Se aplica un criterio en función de la instrucción y de la historia de las últimas veces que se ejecutó.

Conmutador Saltar/No saltar. Basado en autómatas.

Tabla de historia de saltos (BHT). Además de almacenar un estado se almacena la dirección de salto.



4).- Comente breve y razonadamente la veracidad o falsedad de las siguientes sentencias.

La memoria de un procesador vectorial debe disponer necesariamente de entrelazado de orden alto, y opcionalmente de entrelazado de orden bajo. (0,4 pts.)

FALSO: Para que el rendimiento en el acceso a memoria sea aceptable, se debe disponer de entrelazado a nivel bajo, que permite leer datos consecutivos de forma mucho más eficiente. Opcionalmente se puede disponer de entrelazado a nivel alto que facilita la gestión de tolerancia a fallos.

En un procesador vectorial, la ejecución de distintas instrucciones de acceso a memoria puede presentar diferentes tasas de inicialización. (0,4 pts.)

CIERTO: La tasa de inicialización indica el número de elementos de vector que pueden ser accedidos en un ciclo de reloj. No es lo mismo acceder a un vector con elementos contiguos que utilizar un acceso con separación o con índices.

En un procesador matricial, cada unidad de proceso opera sobre todas las componentes de un vector completo aprovechando su estructura interna segmentada. (0,4 pts.)

FALSO: Eso ocurre en los procesadores vectoriales. En los matriciales se dispone de múltiples unidades de proceso y cada una de ellas opera con un elemento del vector.



5).- De un procesador vectorial se conocen los siguientes datos:

- Frecuencia=1GHz.
- Número y tipo de unidades vectoriales: una de carga, una de almacenamiento, una de suma, una de multiplicación y una de división.
- Tiempos de arranque:
 - Instrucciones de carga y almacenamiento vectorial: 15 ciclos de reloj.
 - Instrucciones de suma vectorial: 5 ciclos de reloj.
 - Instrucciones de multiplicación vectorial: 8 ciclos de reloj.
 - Instrucciones de división vectorial: 20 ciclos de reloj.
- Tiempos de gestión de bucle: 25 ciclos de reloj
- Longitud máxima de vector=64.

Para la ejecución del siguiente fragmento de programa, calcular, **tanto para el supuesto de que no admita encadenamiento de instrucciones en los convoyes como para el de que lo admita:**

- A. El número de convoyes que se lanzan, indicando las instrucciones que los componen y los tiempos de arranque a computar en cada uno de ellos. (0,8 ptos.)
- B. El rendimiento para un vector de 250 elementos. (0,4 ptos.)
- C. $N_{1/2}$. (0,4 ptos.)

a→F0	;	[I1]
(Rx)→V1	;	[I2]
F0*V1→V1	;	[I3]
(Ry)→V2	;	[I4]
V1+V2→V1	;	[I5]
(Rz)→V3	;	[I6]
V1/V3→V4	;	[I7]
V4→(Ry)	;	[I8]



5 cont.)-

SOLUCIÓN:

Sin encadenamiento:

Convoy	Instrucciones	Tiempo de arranque
1	I2	15 ciclos
2	I3, I4	max(8,15) 15 ciclos
3	I5, I6	max(5,15) 15 ciclos
4	I7	20 ciclos
5	I8	15 ciclos
	Total:	80 ciclos

$$R_n = \frac{Op * n * Freq}{\left\lceil \frac{n}{MVL} \right\rceil * (T_{arr} + T_{loop}) + n * T_{campanadas}} \quad N_{1/2} = \frac{\left\lceil \frac{N_{1/2}}{MVL} \right\rceil * (T_{arr} + T_{loop})}{2 * \frac{T_{arr} + T_{loop}}{MVL} + T_{campanadas}}$$

$$R_{250} = \frac{3 * 250 * 1000}{\left\lceil \frac{250}{64} \right\rceil * (80 + 25) + 250 * 5} = \frac{750000}{1670} = 449,10 \text{ MFlops}$$

$$N_{1/2} = \frac{1 * (80 + 25)}{2 * \frac{80 + 25}{64} + 5} = \frac{105}{\frac{530}{64}} = 12,68 \rightarrow 13$$

Con encadenamiento:

Convoy	Instrucciones	Tiempo de arranque
1	I2, I3	15+8 = 23 ciclos
2	I4, I5	15+5 = 20 ciclos
3	I6, I7, I8	15+20+15 = 50 ciclos
	Total:	93 ciclos

$$R_{250} = \frac{3 * 250 * 1000}{\left\lceil \frac{250}{64} \right\rceil * (93 + 25) + 250 * 3} = \frac{750000}{1222} = 613,74 \text{ MFlops}$$

$$N_{1/2} = \frac{1 * (93 + 25)}{2 * \frac{93 + 25}{64} + 3} = \frac{118}{\frac{428}{64}} = 17,65 \rightarrow 18$$



6).- Describa brevemente los tipos de redes de interconexión para sistemas multiprocesador. (0,8 ptos.)

SOLUCIÓN:

REDES DE MEDIO COMPARTIDO: Se caracterizan porque el medio lo comparten todos los usuarios y no lo pueden simultanear. Permite broadcast pero es muy poco escalable.

REDES DIRECTAS: Los diferentes nodos de la red están conectados de forma fija a un subconjunto (pequeño) de otros nodos de red. Cada nodo es un ordenador programable con su procesador, memoria y otros dispositivos. Los nodos disponen de un encaminador que gestiona el envío y recepción de los mensajes. El sistema es altamente escalable y suelen existir múltiples rutas entre dos nodos.

REDES INDIRECTAS: Los diferentes nodos de la red están conectados por medio de conmutadores. Cada nodo dispone de un adaptador de red que se conecta a un conmutador, que se conecta a dispositivos (procesadores, memorias, periféricos) o a otros conmutadores. El sistema es altamente escalable y suelen existir múltiples rutas entre dos nodos.

REDES HIBRIDAS: Pueden construirse redes de interconexión que incluyan dos o más tipos de redes de las anteriormente descritas.



7).- En una red omega 32×32 se establece la conexión entre la entrada 3 y la salida 11H.

Se pide:

1. ¿Qué conmutador se utiliza en cada etapa y cuáles de ellos deberían cruzar sus canales? (0,4 ptos.)
2. ¿Qué conexiones quedarían bloqueadas por la indicada en la etapa 1? (0,4 ptos.)
3. ¿Cuántas conexiones se bloquearían en total por la red realizada? (0,4 ptos.)

NOTA: Suponga que la etapa de conmutación más cercana a la entrada es la C0.

SOLUCIÓN:

1)

Direcciones	Etapas	conmutador	¿cruzar?
<u>000111</u> 0001	0	3	Si
0 <u>001110</u> 001	1	7	No
00 <u>011100</u> 01	2	EH	No
000 <u>111000</u> 1	3	CH	Si
0001 <u>110001</u>	4	8	No

2)

En la etapa 1 se bloquean las conexiones : X101110XXX es decir las que entran por los canales 0BH y 1BH y salen por los canales 10H, 12H, 13H, 14H, 15H, 16H y 17H.

3)

En total se bloquean $(n-2) \cdot 2^{n-1} + 1$ es decir $3 \cdot 2^4 + 1 = 49$ conexiones.



8).- Indique y describa brevemente los distintos tipos de protocolos de directorio. (0,8 ptos.)

SOLUCIÓN:

Protocolos de directorio plano: Se caracterizan porque la información del directorio para un bloque se encuentra en un lugar fijo, normalmente en el nodo origen, y ante un fallo se envía una transacción de red directamente al nodo fuente para buscar en el directorio.

Con mapeado completo: Cada entrada del directorio almacena información de una línea en la totalidad de los nodos. Su gestión es más sencilla que otras alternativas, pero requiere muchos recursos.

Con mapeado incompleto: Cada entrada del directorio almacena el estado de un número limitado de líneas. Si el directorio está lleno y un nuevo nodo requiere cargar la línea será necesario el 'desalojo' de una de las entradas del directorio y por tanto invalidar esa línea en algún nodo. Optimiza el uso de los recursos del sistema.

Basados en caché con lista doblemente enlazada: En este caso en lugar de disponer de una lista que contenga todos los nodos o una parte de ellos, por cada línea de la memoria principal se tiene un puntero al primer nodo que la tenga cargada. A partir de aquí y con una lista doblemente enlazada tendremos acceso a todos los nodos que contienen la línea. El doble enlace permite a cualquier nodo salirse de la lista dejándola en estado coherente. Con este método la información de los directorios es muy pequeña (un puntero) y a cambio se hace crecer a las cachés.

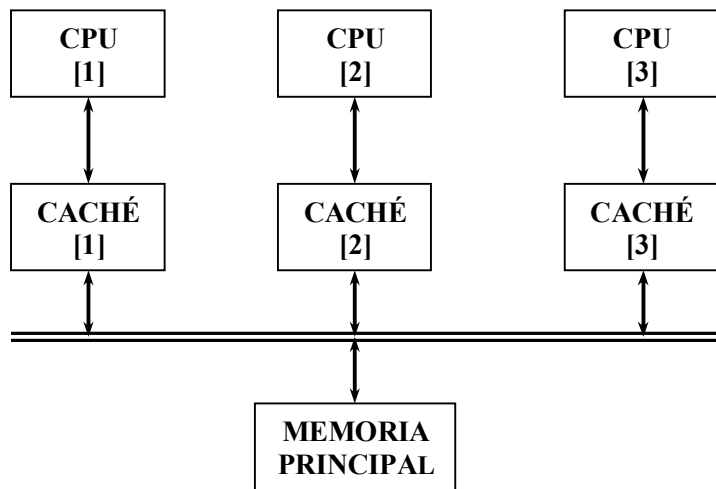
Protocolos de directorio jerárquico: La memoria también está distribuida entre los procesadores, pero la información está organizada en una estructura jerárquica (un árbol)



9).- En el sistema multiprocesador esquematizado en la figura, se realizan una serie de acciones, especificadas en la tabla correspondiente, que afectan a una determinada línea de memoria principal.

Se pide:

1. Suponiendo que inicialmente la línea en cuestión se encuentra en estado E en la caché 1, complete la tabla I considerando que las cachés utilizan el protocolo MESI. (0,6 ptos.)
2. Suponiendo que inicialmente la línea en cuestión se encuentra en estado E en la caché 1, complete la tabla II considerando que las cachés utilizan el protocolo Dragon. (0,6 ptos.)





9 cont.).

SOLUCIÓN

TABLA I: Protocolo MESI

Acciones	Operaciones sobre el bus	Estado final caché	
		[1]	[3]
Escritura CPU [1]	<i>Ninguna operación en los Buses</i>	M	-
Lectura CPU [3]	<i>[3]-BusRd(bloq. por [1]), [1]-BusWB, [1]-S, [3]-BusRd(S) después de desbloquear [1] el acceso.</i>	S	S
Descarga en caché [1]	<i>Ninguna operación en los Buses</i>	-	S
Escritura CPU [3]	<i>[3]-BusRdX</i>	-	M



9 cont.).

TABLA II: Protocolo Dragon

Acceso	Operaciones sobre el bus	Estado final caché	
		[1]	[3]
Lectura CPU [3]	<i>[3]-BusRdM. Acceso bloqueado por [1] [1]-BusWB, [1]-S [1] desbloquea el acceso, [3]-BusRdM(S)</i>	SC	SC
Escritura CPU [3]	<i>[3]-BusUpd(S) [1]-S, [1]-Actualizar</i>	SC	SM
Descarga en caché [3]	<i>[3]-BusWB</i>	SC	-
Lectura CPU [1]	<i>Ningún acceso a los Buses</i>	SC	-



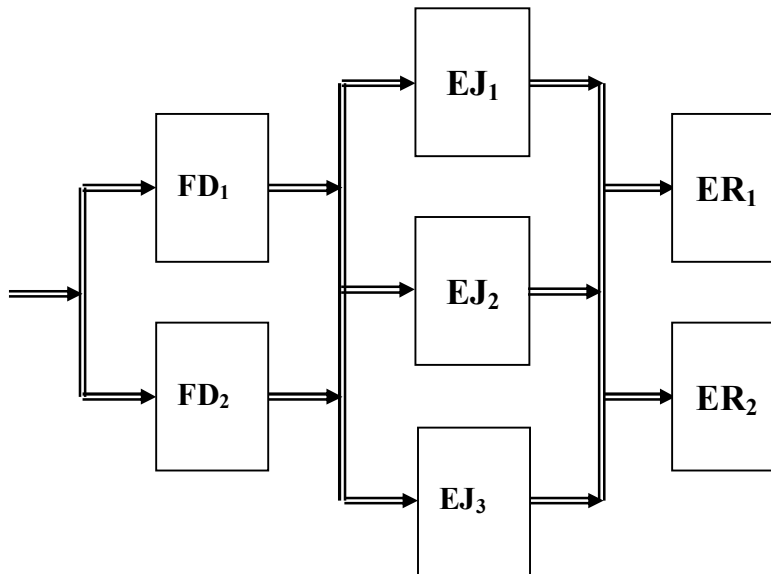
JUNIO 2007 (Parcial y Final Tarde)

1).- Dada la configuración superescalar de la figura que hay a continuación, y la secuencia de instrucciones $I_1, I_2, I_3, I_4, I_5, I_6, I_7, I_8, I_9, I_{10}, I_{11},$ e I_{12} , que tiene las siguientes particularidades:

- Todas las instrucciones tardan un ciclo de reloj en cada fase, excepto I_2, I_5 e I_{10} , cuyas fases de ejecución duran 2 ciclos de reloj, e I_4 e I_9 que dura 3 ciclos.
- Hay una dependencia de operandos entre I_2 e I_1 , y, por lo tanto, la ejecución de I_2 no puede comenzar hasta que termine la fase de escritura de resultados de I_1 . Lo mismo ocurre con respecto a I_5 e I_4 y con I_9 e I_6 .
- I_3, I_6, I_9 e I_{12} son de coma flotante y, por lo tanto, ellas y sólo ellas deben ejecutarse en EJ_3 .

Se pide:

3. Mostrar mediante una tabla la secuencia de ejecución **totalmente ordenada** de las instrucciones. (0,6 pts.)
4. Repetir de nuevo el proceso para el caso de que se permita el **desorden, tanto en la ejecución, como en la escritura de resultados**. (0,6 pts.)





1 cont.)-

SOLUCIÓN

Apartado 1

Ciclo	FD ₁	FD ₂	EJ ₁	Ej ₂	EJ ₃	ER ₁	ER ₂
1	I ₁	I ₂					
2	I ₃		I ₁				
3						I ₁	
4	I ₄	I ₅	I ₂		I ₃		
5	I ₆		I ₂	I ₄			
6				I ₄		I ₂	I ₃
7				I ₄			
8						I ₄	
9	I ₇	I ₈	I ₅		I ₆		
10	I ₉		I ₅	I ₇			
11		I ₁₀	I ₈			I ₅	I ₆
12	I ₁₁	I ₁₂	I ₁₀		I ₉	I ₇	I ₈
13			I ₁₀	I ₁₁	I ₉		
14					I ₉		
15					I ₁₂	I ₉	I ₁₀
16						I ₁₁	I ₁₂

Apartado 2

Ciclo	FD ₁	FD ₂	Ventana	EJ ₁	Ej ₂	EJ ₃	ER ₁	ER ₂
1	I ₁	I ₂						
2	I ₃	I ₄	I₁ , I ₂	I ₁				
3	I ₅	I ₆	I ₂ , I₃ , I₄	I ₄		I ₃	I ₁	
4	I ₇	I ₈	I₃ , I ₅ , I₆	I ₄	I ₂	I ₆	I ₃	
5	I ₉	I ₁₀	I ₅ , I ₇ , I ₈	I ₄	I ₂		I ₆	
6	I ₁₁	I ₁₂	I ₅ , I₇ , I₈ , I₉ , I ₁₀	I ₇	I ₈	I ₉	I ₂	I ₄
7			I₅ , I₁₀ , I ₁₁ , I ₁₂	I ₅	I ₁₀	I ₉	I ₇	I ₈
8			I₅ , I ₁₂	I ₅	I ₁₀	I ₉		
9			I₅ , I₁₂	I ₁₁		I ₁₂	I ₅	I ₉
10							I ₁₀	I ₁₁
11							I ₁₂	



2).-

Explique brevemente en qué consiste y cuándo se aplica la técnica de renombrado de registros. (0,4 ptos.)

Se aplica para resolver las pseudodependencias de operandos. Si no se puede modificar un registro, se almacena un resultado en otro registro que se 'renombrará' al original cuando este pueda ser modificado.

Indique cuáles son las características típicas de las instrucciones de una CPU RISC. (0,4 ptos.)

- *Conjunto de instrucciones limitado, sencillo y homogéneo (Set de instrucciones ortogonal).*
- *Unidad de control cableada vs. UC cableada.*
- *Elevado número de registros en la CPU.*
- *Las instrucciones (salvo lect. y escritura) no tienen operandos en memoria.*

¿Cuáles son las principales ventajas que aporta el nivel L1 de caché de un Pentium? (0,4 ptos.)

La caché trabaja a la misma velocidad de reloj que la CPU. Al disponer de cachés independientes de datos e instrucciones, se puede acceder a ambas de forma simultánea.



3).- Conflicto de dependencia de saltos. ¿Qué alternativas se plantean según el tipo de salto? Explique brevemente las distintas técnicas de mejora de rendimiento respecto a este conflicto. (0,8 ptos.)

Incondicionales: El salto se produce siempre. Detectarlo en fase de fetch o decodificación y modificar el PC con la dirección de salto.

Condicionales: No se puede asegurar si se va a saltar o no. Las técnicas usadas son:

Salto retardado: Se intercambia la instrucción de salto con la anterior. Mientras se salta se ejecuta la instrucción intercambiada.

Flujos múltiples: Etapas del pipeline duplicadas para poder seguir los dos caminos.

Precaptar destino del salto: Leer la siguiente instrucción y la de salto.

Buffer de bucles: Buffer de prefetch amplio. Se activa en los saltos condicionales hacia atrás.

Predicción de salto: Se pueden usar criterios estáticos o dinámicos:

Estáticos: Se aplica un criterio en función de la instrucción.

No cumple condición. Se presupone que no se cumple la condición.

Cumple condición. Se presupone que se cumple la condición.

Decisión según el código de operación. En función del tipo de salto se predice saltar o no saltar.

Evaluación previa de la condición. Se evalúa la condición previamente y se acepta el resultado como la decisión de saltar o no saltar.

Dinámicos: Se aplica un criterio en función de la instrucción y de la historia de las últimas veces que se ejecutó.

Conmutador Saltar/No saltar. Basado en autómatas.

Tabla de historia de saltos (BHT). Además de almacenar un estado se almacena la dirección de salto.



4).- Comente breve y razonadamente la veracidad o falsedad de las siguientes sentencias.

La memoria de un procesador vectorial debe disponer necesariamente de entrelazado de orden alto, y opcionalmente de entrelazado de orden bajo. (0,4 ptos.)

FALSO: Para que el rendimiento en el acceso a memoria sea aceptable, se debe disponer de entrelazado a nivel bajo, que permite leer datos consecutivos de forma mucho más eficiente. Opcionalmente se puede disponer de entrelazado a nivel alto que facilita la gestión de tolerancia a fallos.

En un procesador vectorial, la ejecución de distintas instrucciones de acceso a memoria puede presentar diferentes tasas de inicialización. (0,4 ptos.)

CIERTO: La tasa de inicialización indica el número de elementos de vector que pueden ser accedidos en un ciclo de reloj. No es lo mismo acceder a un vector con elementos contiguos que utilizar un acceso con separación o con índices.

En un procesador matricial, cada unidad de proceso opera sobre todas las componentes de un vector completo aprovechando su estructura interna segmentada. (0,4 ptos.)

FALSO: Eso ocurre en los procesadores vectoriales. En los matriciales se dispone de múltiples unidades de proceso y cada una de ellas opera con un elemento del vector.



5).- De un procesador vectorial se conocen los siguientes datos:

- Frecuencia=1GHz.
- Número y tipo de unidades vectoriales: una de carga, una de almacenamiento, una de suma, una de multiplicación y una de división.
- Tiempos de arranque:
 - Instrucciones de carga y almacenamiento vectorial: 15 ciclos de reloj.
 - Instrucciones de suma vectorial: 5 ciclos de reloj.
 - Instrucciones de multiplicación vectorial: 8 ciclos de reloj.
 - Instrucciones de división vectorial: 20 ciclos de reloj.
- Tiempos de gestión de bucle: 25 ciclos de reloj
- Longitud máxima de vector=64.

Para la ejecución del siguiente fragmento de programa, calcular, **tanto para el supuesto de que no admita encadenamiento de instrucciones en los convoyes como para el de que lo admita:**

- A. El número de convoyes que se lanzan, indicando las instrucciones que los componen y los tiempos de arranque a computar en cada uno de ellos. (0,8 ptos.)
- B. El rendimiento para un vector de 250 elementos. (0,4 ptos.)
- C. $N_{1/2}$. (0,4 ptos.)

a→F0	;	[I1]
(Rx)→V1	;	[I2]
F0*V1→V1	;	[I3]
(Ry)→V2	;	[I4]
V1+V2→V1	;	[I5]
(Rz)→V3	;	[I6]
V1/V3→V4	;	[I7]
V4→(Ry)	;	[I8]



5 cont.)-

SOLUCIÓN:

Sin encadenamiento:

Convoy	Instrucciones	Tiempo de arranque
1	I2	15 ciclos
2	I3, I4	max(8,15) 15 ciclos
3	I5, I6	max(5,15) 15 ciclos
4	I7	20 ciclos
5	I8	15 ciclos
	Total:	80 ciclos

$$R_n = \frac{Op * n * Freq}{\left\lceil \frac{n}{MVL} \right\rceil * (T_{arr} + T_{loop}) + n * T_{campanadas}} \quad N_{1/2} = \frac{\left\lceil \frac{N_{1/2}}{MVL} \right\rceil * (T_{arr} + T_{loop})}{2 * \frac{T_{arr} + T_{loop}}{MVL} + T_{campanadas}}$$

$$R_{250} = \frac{3 * 250 * 1000}{\left\lceil \frac{250}{64} \right\rceil * (80 + 25) + 250 * 5} = \frac{750000}{1670} = 449,10 \text{ MFlops}$$

$$N_{1/2} = \frac{1 * (80 + 25)}{2 * \frac{80 + 25}{64} + 5} = \frac{105}{\frac{530}{64}} = 12,68 \rightarrow 13$$

Con encadenamiento:

Convoy	Instrucciones	Tiempo de arranque
1	I2, I3	15+8 = 23 ciclos
2	I4, I5	15+5 = 20 ciclos
3	I6, I7, I8	15+20+15 = 50 ciclos
	Total:	93 ciclos

$$R_{250} = \frac{3 * 250 * 1000}{\left\lceil \frac{250}{64} \right\rceil * (93 + 25) + 250 * 3} = \frac{750000}{1222} = 613,74 \text{ MFlops}$$

$$N_{1/2} = \frac{1 * (93 + 25)}{2 * \frac{93 + 25}{64} + 3} = \frac{118}{\frac{428}{64}} = 17,65 \rightarrow 18$$



6).- Describa brevemente los tipos de redes de interconexión para sistemas multiprocesador. (0,8 ptos.)

SOLUCIÓN:

REDES DE MEDIO COMPARTIDO: Se caracterizan porque el medio lo comparten todos los usuarios y no lo pueden simultanear. Permite broadcast pero es muy poco escalable.

REDES DIRECTAS: Los diferentes nodos de la red están conectados de forma fija a un subconjunto (pequeño) de otros nodos de red. Cada nodo es un ordenador programable con su procesador, memoria y otros dispositivos. Los nodos disponen de un encaminador que gestiona el envío y recepción de los mensajes. El sistema es altamente escalable y suelen existir múltiples rutas entre dos nodos.

REDES INDIRECTAS: Los diferentes nodos de la red están conectados por medio de conmutadores. Cada nodo dispone de un adaptador de red que se conecta a un conmutador, que se conecta a dispositivos (procesadores, memorias, periféricos) o a otros conmutadores. El sistema es altamente escalable y suelen existir múltiples rutas entre dos nodos.

REDES HIBRIDAS: Pueden construirse redes de interconexión que incluyan dos o más tipos de redes de las anteriormente descritas.



7).- En una red omega 32×32 se establece la conexión entre la entrada 3 y la salida 11H.

Se pide:

- ¿Qué conmutador se utiliza en cada etapa y cuáles de ellos deberían cruzar sus canales? (0,4 pts.)
- ¿Qué conexiones quedarían bloqueadas por la indicada en la etapa 1? (0,4 pts.)
- ¿Cuántas conexiones se bloquearían en total por la red realizada? (0,4 pts.)

NOTA: Suponga que la etapa de conmutación más cercana a la entrada es la C0.

SOLUCIÓN

1)

Direcciones	Etapas	conmutador	¿cruzar?
<u>000111</u> 0001	0	3	Si
0 <u>001110</u> 001	1	7	No
00 <u>011100</u> 01	2	EH	No
000 <u>111000</u> 1	3	CH	Si
0001 <u>110001</u>	4	8	No

2)

En la etapa 1 se bloquean las conexiones : X101110XXX es decir las que entran por los canales 0BH y 1BH y salen por los canales 10H, 12H, 13H, 14H, 15H, 16H y 17H.

3)

En total se bloquean $(n-2) \cdot 2^{n-1} + 1$ es decir $3 \cdot 2^4 + 1 = 49$ conexiones.



8).- Indique y describa brevemente los distintos tipos de protocolos de directorio. (0,8 ptos.)

SOLUCIÓN:

Protocolos de directorio plano: Se caracterizan porque la información del directorio para un bloque se encuentra en un lugar fijo, normalmente en el nodo origen, y ante un fallo se envía una transacción de red directamente al nodo fuente para buscar en el directorio.

Con mapeado completo: Cada entrada del directorio almacena información de una línea en la totalidad de los nodos. Su gestión es más sencilla que otras alternativas, pero requiere muchos recursos.

Con mapeado incompleto: Cada entrada del directorio almacena el estado de un número limitado de líneas. Si el directorio está lleno y un nuevo nodo requiere cargar la línea será necesario el 'desalojo' de una de las entradas del directorio y por tanto invalidar esa línea en algún nodo. Optimiza el uso de los recursos del sistema.

Basados en caché con lista doblemente enlazada: En este caso en lugar de disponer de una lista que contenga todos los nodos o una parte de ellos, por cada línea de la memoria principal se tiene un puntero al primer nodo que la tenga cargada. A partir de aquí y con una lista doblemente enlazada tendremos acceso a todos los nodos que contienen la línea. El doble enlace permite a cualquier nodo salirse de la lista dejándola en estado coherente. Con este método la información de los directorios es muy pequeña (un puntero) y a cambio se hace crecer a las cachés.

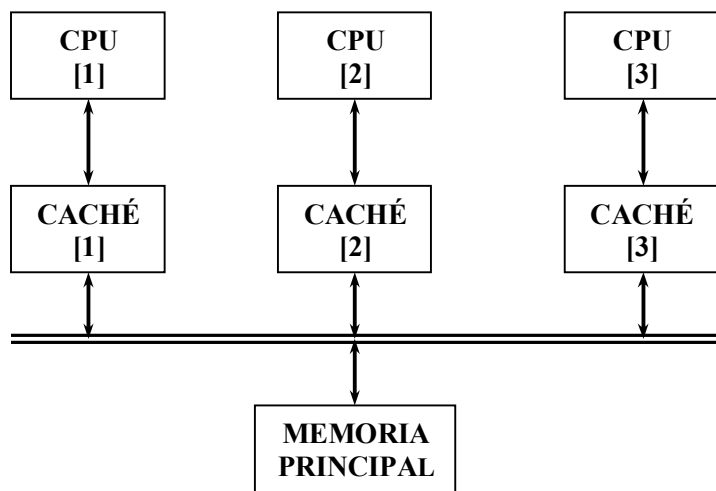
Protocolos de directorio jerárquico: La memoria también está distribuida entre los procesadores, pero la información está organizada en una estructura jerárquica (un árbol)



9).- En el sistema multiprocesador esquematizado en la figura, se realizan una serie de acciones, especificadas en la tabla correspondiente, que afectan a una determinada línea de memoria principal.

Se pide:

3. Suponiendo que inicialmente la línea en cuestión se encuentra en estado E en la caché 1, complete la tabla I considerando que las cachés utilizan el protocolo MESI. (0,6 ptos.)
4. Suponiendo que inicialmente la línea en cuestión se encuentra en estado M en la caché 1, complete la tabla II considerando que las cachés utilizan el protocolo Dragon. (0,6 ptos.)





9 cont.).

SOLUCIÓN

TABLA I: Protocolo MESI

Acciones	Operaciones sobre el bus	Estado final caché	
		[1]	[3]
Escritura CPU [1]	<i>Ninguna operación en los Buses</i>	M	-
Lectura CPU [3]	<i>[3]-BusRd(bloq. por [1]), [1]-BusWB, [1]-S, [3]-BusRd(S) después de desbloquear [1] el acceso.</i>	S	S
Descarga en caché [1]	<i>Ninguna operación en los Buses</i>	-	S
Escritura CPU [3]	<i>[3]-BusRdX</i>	-	M



9 cont.).

TABLA II: Protocolo Dragon

Acceso	Operaciones sobre el bus	Estado final caché	
		[1]	[3]
Lectura CPU [3]	<i>[3]-BusRd. Acceso bloqueado por [1] [1]-BusWB, [1]-S [1] desbloquea el acceso, [3]-BusRd(S)</i>	SC	SC
Escritura CPU [3]	<i>[3]-BusUpd(S) [1]-S, [1]-Actualizar</i>	SC	SM
Descarga en caché [3]	<i>[3]-BusWB</i>	SC	-
Lectura CPU [1]	<i>Ningún acceso a los Buses</i>	SC	-



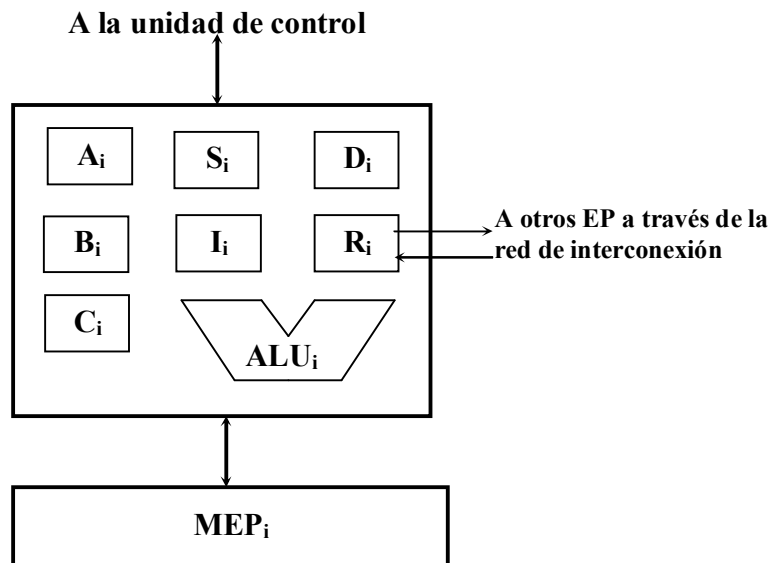
SEPTIEMBRE 2007 (Parcial y Final Tarde)

1).- Describe como es la estructura interna de una unidad de proceso en un procesador matricial indicando el cometido de cada una de sus partes. (1,8 ptos.)

SOLUCIÓN:

Ejemplo de elemento de proceso en un procesador matricial:

- A_i , B_i y C_i son registros de propósito general.
- D_i y R_i sirven para la interconexión con otros EP: en D_i se indica el número o dirección del EP destino y en R_i la información o dato que se.
- S_i es un indicador que señala si el elemento está activo ($S_i=1$) o inactivo ($S_i=0$).
- I_i funciona como índice en los accesos a memoria.
- MEP_i es la memoria local de EPI_i , y ALU_i su ALU.





2).- De un procesador vectorial se conocen los siguientes datos:

- Número y tipo de unidades vectoriales: dos de carga/almacenamiento, una de suma, una de multiplicación y una de división.
- Tiempos de arranque:
 - Instrucciones de carga y almacenamiento vectorial: 12 ciclos de reloj.
 - Instrucciones de suma vectorial: 6 ciclos de reloj.
 - Instrucciones de multiplicación vectorial: 7 ciclos de reloj.
 - Instrucciones de división vectorial: 20 ciclos de reloj.
- Tiempos de bucle:
 - Vectorial: 15 ciclos de reloj.
 - Escalar: 20 ciclos de reloj.
- Longitud máxima de vector=128.
- Se permite encadenamiento de convoyes.

Para la ejecución del siguiente fragmento de programa, calcular:

- El número de convoyes que se lanzan, indicando además las instrucciones que componen cada uno de ellos y los tiempos de arranque a computar en cada uno de ellos. (0,6 ptos.)
- Para un vector de 300 elementos ¿A que frecuencia debe operar la CPU para obtener un rendimiento de 500 MFlops?. (0,6 ptos.)
- ¿A que frecuencia debe operar la CPU para duplicar el rendimiento? (0,5 ptos.)
- $N_{1/2}$. (0,5 ptos.)

(Rx)→V1	:[I1]
(Ry)→V2	:[I2]
0→F0	:[I3]
V1+V2→V3	:[I4]
SNES F0, V3	:[I5]
30→F0	:[I6]
F0+V3→V1	:[I7]
V1→(Rx)	:[I8]

SOLUCIÓN:

A) Esta instrucción realiza una comparación de los elementos de V3 con F0 lo que supone una operación de resta que se realiza en la unidad de suma vectorial.



2 cont.)-

Convoy	Instrucciones	Tiempo de arranque
1	I2, I3	12 + 6 = 18 ciclos
2	I5	6 ciclos
3	I6, I7	6 + 12 = 18 ciclos
	Total:	42 ciclos

B)

$$R_{300} = \frac{3 * 300 * Freq.}{\left[\frac{300}{128} \right] * (35 + 42) + 300 * 3} = 500 MFlops$$

$$Freq = \frac{500 \left(\left[\frac{300}{128} \right] * (35 + 42) + 300 * 3 \right)}{3 * 300} = \frac{500 (3 * 77 + 900)}{900} = 628,33 MHz$$

C)

$$R_{300} = \frac{3 * 300 * Freq.}{\left[\frac{300}{128} \right] * (35 + 42) + 300 * 3} = 1000 MFlops$$

$$Freq = \frac{1000 \left(\left[\frac{300}{128} \right] * (35 + 42) + 300 * 3 \right)}{3 * 300} = \frac{1000 (3 * 77 + 900)}{900} = 1.256,67 MHz$$

D)

$$N_{1/2} = \frac{\left[\frac{N_{1/2}}{128} \right] * (35 + 42)}{2 * \frac{35 + 42}{128} + 3} = \frac{35 + 42}{2 * \frac{35 + 42}{128} + 3} = 18,67 \rightarrow 19$$

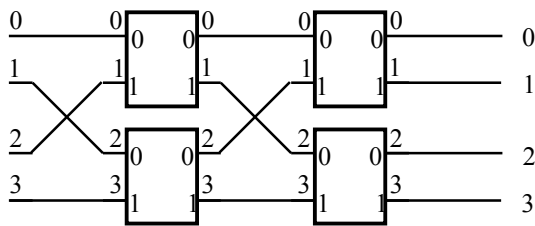


3).- Defina que es una red MIN bloqueante y no bloqueante y dibuja un ejemplo de cada una de ellas. (1,2 ptos.)

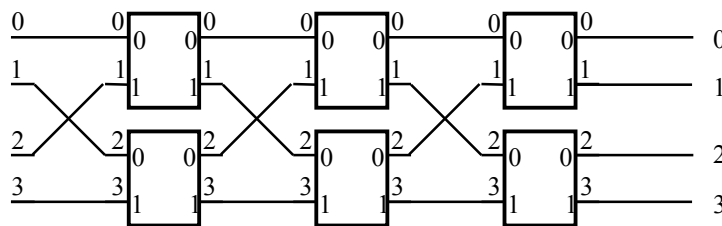
SOLUCIÓN:

Una red MIN no bloqueante es aquella que garantiza que, en un momento dado, siempre habrá al menos un camino posible entre una entrada y una salida de la red. En las redes bloqueantes no se garantiza. Esto se debe a que al conectar una entrada con una salida se usan recursos que pueden ser necesarios para otras conexiones. Las redes no bloqueantes requieren mayor nº de etapas de conexión y de conmutación.

Ejemplo de red bloqueante:



Ejemplo de red no bloqueante:





4).- En una red omega de 16 X 16 se establece la conexión entre la entrada 5 y la salida 5.

Se pide:

1. ¿Por qué conmutador pasa en cada una de las etapas? (0,5 ptos.)
2. ¿A cuáles de los conmutadores anteriores habría que darles la orden de que cruzaran los canales, es decir, de que conectasen la entrada 0 con la salida 1 y la entrada 1 con la salida 0? (0,5 ptos.)
3. ¿Qué conexiones quedarían bloqueadas por ésta en la etapa 1? (0,8 ptos.)

SOLUCIÓN:

1) Etapa 0: $0\boxed{1010}101 \rightarrow$ Conmutador 5

Etapa 1: $0\boxed{10101}01 \rightarrow$ Conmutador 2

Etapa 2: $01\boxed{01010}1 \rightarrow$ Conmutador 5

Etapa 3: $010\boxed{10101} \rightarrow$ Conmutador 2

2) Etapa 0: $0\boxed{1010}101 \rightarrow$ No cruzar (misma línea de entrada y salida)

Etapa 1: $0\boxed{10101}01 \rightarrow$ No cruzar (misma línea de entrada y salida)

Etapa 2: $01\boxed{01010}1 \rightarrow$ No cruzar (misma línea de entrada y salida)

Etapa 3: $010\boxed{10101} \rightarrow$ No cruzar (misma línea de entrada y salida)

3) Etapa 1: $0\boxed{10101}01$

Se bloquean todas las conexiones que usen distinta entrada y la misma salida en ese conmutador: $X\boxed{00101}XX$ excepto el que llega a la misma salida.

0001 0100 Conexión de 1 a 4

0001 0110 Conexión de 1 a 6

0001 0111 Conexión de 1 a 7

1001 0100 Conexión de 9 a 4

1001 0110 Conexión de 9 a 6

1001 0111 Conexión de 9 a 7



5).- En un sistema multiprocesador con memoria distribuida, se mantiene la coherencia de caché con un protocolo de directorio plano y mapeado completo. Indica las acciones realizadas si un procesador quiere realizar una operación de escritura sobre una línea que pertenece a otro nodo y que está alojada en otras dos memorias caché. (1,2 ptos.)

SOLUCIÓN:

Los datos del enunciado indican que la línea no se encuentra en la caché del nodo petionario (el que quiere escribir) ni en la del propietario, sino en otras dos cachés (que llamaremos cache1 y cache2), y que la memoria principal está actualizada. Los pasos que se realizan son los siguientes:

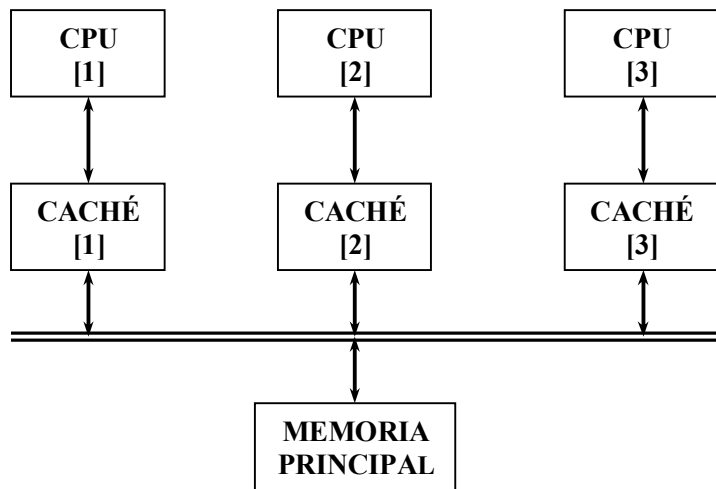
- *El nodo petionario pide la línea para modificarla al nodo propietario.*
- *El nodo propietario ordena invalidar la línea a la cache1.*
- *La caché 1 desactiva el bit V de la línea y envía reconocimiento al nodo propietario.*
- *El nodo propietario desactiva el bit Pr. de Cache1 en su directorio.*
- *Se repiten los tres pasos anteriores con la cache2.*
- *El nodo propietario activa el bit Pr correspondiente al nodo petionario y el bit Iu. Además envía la línea a dicho nodo.*
- *El nodo petionario recibe la línea, la modifica y activa los bits V y D.*



6).- En el sistema de la figura se sabe que las tres cachés siguen un protocolo de coherencia, y que sobre una determinada línea de información, que inicialmente está cargada como exclusiva en la CPU[1].

Se pide rellenar las tablas 1 y 2 que indican una serie de accesos indicando:

- ¿Qué operaciones se desencadenan en el bus? Especificar también qué “snoopy” lanza cada una de ellas.
- ¿En qué estado queda la línea en cada caché al terminar cada acceso de CPU? (0,9 ptos cada tabla.)





6 cont.).

SOLUCIÓN

TABLA 1: Protocolo MESI

Acceso	Operaciones sobre el bus	Estado final caché		
		[1]	[2]	[3]
Escritura CPU [1]	<i>Se realiza la petición de escritura sobre la caché1 que como tiene la línea en exclusiva permite la escritura pasando a estado M y sin hacer ningún uso de los buses externos.</i>	M	-	-
Lectura CPU [3]	<i>La CPU 3 solicita leer. Se produce fallo de caché y se accede a memoria principal para leer la línea. La caché1 detecta el acceso, lo bloquea, actualiza la memoria principal, desbloquea el acceso y activa la línea S para que la caché 3 no cargue la línea como exclusiva.</i>	S		S
Escritura CPU [3]	<i>Como la caché 3 tiene marcada la línea como compartida, lanza un BusRdX. La caché 1 lo detecta e invalida la línea. La caché 3 pasa a estado M.</i>	I		M
Lectura CPU [2]	<i>La CPU 2 solicita leer. Se produce fallo de caché y se accede a memoria principal para leer la línea. La caché 3 detecta el acceso, lo bloquea, actualiza la memoria principal, desbloquea el acceso y activa la línea S para que la caché 2 no cargue la línea como exclusiva.</i>	I	S	S



6 cont.).

TABLA 2: Protocolo Dragon

Acceso	Operaciones sobre el bus	Estado final caché		
		[1]	[2]	[3]
Escritura CPU [1]	<i>Se realiza la petición de escritura sobre la caché1 que como tiene la línea en exclusiva permite la escritura pasando a estado M y sin hacer ningún uso de los buses externos.</i>	M		
Lectura CPU [3]	<i>La CPU 3 solicita leer. Se produce fallo de caché y se accede a memoria principal para leer la línea. La caché1 detecta el acceso y genera un BusUpd que permite que se cargue la línea actualizada y activa la línea S para que la caché 3 no cargue la línea como exclusiva.</i>	SM		SC
Escritura CPU [3]	<i>La CPU 3 solicita escribir. La caché 3 lanza un BusUpd porque está en estado SC al que contesta la caché 1 activando S y actualizando su contenido</i>	SC		SM
Lectura CPU [2]	<i>La CPU 2 solicita leer. Se produce fallo de caché y se accede a memoria principal para leer la línea. La caché3 detecta el acceso y genera un BusUpd que permite que se cargue la línea actualizada y activa la línea S para que la caché 2 no cargue la línea como exclusiva.</i>	SC	SC	SM