Apellidos _____ Nombre _____ Grupo___

Arquitectura e Ingeniería de Computadores. Examen Final (Teoría – parte primer cuatrimestre). 14/09/2009

Instrucciones.- Cada pregunta consta de cinco respuestas, y cada una de las respuestas puede ser cierta o falsa. Marque con un aspa las respuestas que considere ciertas y deje en blanco las que considere falsas. Si considera que alguna respuesta es ambigua y, por tanto, podría considerarse cierta o falsa en función de la interpretación, ponga una llamada y explique sus argumentos al dorso de la hoja.
No se permite la utilización de calculadora.
Puntuación.- Pregunta con todas las respuestas acertadas: 1 punto. Pregunta con un fallo: 0,6 puntos. Pregunta dos fallos: 0,2 puntos. Pregunta con más de dos fallos 0 puntos. La teoría del primer cuatrimestre supone la mitad de la nota del primer cuatrimestre. Tanto la nota de teoría como la de problemas se normalizarán para que el primer cuatrimestre tenga un peso del 65% en la nota final de la asignatura.

1. Supongamos la arquitectura básica del DLX segmentado en 5 etapas sobre la que se implementa una política de saltos retardados. Los saltos se resuelven en la segunda etapa (DE). Marque cuáles de las siguientes afirmaciones son ciertas:

☐ a) Es legal que el compilador realice la siguiente optimización de código:

```
        add r4, r2,r2          add r4,r2,r2
        sub r1,r2,r3           sub r1,r2,r3
        beqz  r1, L1           beqz  r1, L1
        nop            ⟹      sub  r4,r1,r3
        sub  r4,r1,r3          xor  r8,r4,r5
        xor  r8,r4,r5          …
        …                      …
L1:     sub  r5,r4,r7    L1:   sub  r5,r4,r7
```

☒ b) Es legal que el compilador realice la siguiente optimización de código:

```
        sub r1,r2,r1           sub r1,r2,r1
        beqz  r1, L1           beqz  r1, L2
        nop            ⟹      xor  r4,r2,r2
        add  r4,r1,r1          add  r4,r1,r1
        …                      …
L1:     xor  r4,r2,r2    L1:   xor  r4,r2,r2
        ld  r1,0(r4)     L2:   ld  r1,0(r4)
```

☐ c) Si el compilador rellena el hueco de retardo (delay slot) con una instrucción NOP en el 40% de los casos, entonces la penalización media por salto es de 0,4 ciclos de reloj.

☒ d) Es legal que el compilador realice la siguiente optimización de código:

```
        add  r6,r4,r4          …
        sub  r1,r2,r3          sub  r1,r2,r3
        beqz  r1, L1   ⟹       beqz  r1, L1
        nop                    add  r6,r4,r4
        xor  r5,r4,r2          xor  r5,r4,r2
        …                      …
```

☒ e) Si los saltos se resolvieran en la etapa MEM, entonces el delay slot pasaría a ser de tres ciclos de reloj.

2. Para ejecutar el programa P un DLX ejecuta 10^{12} instrucciones, invirtiendo un promedio de cuatro ciclos de reloj por instrucción. El programa contiene 10^{11} operaciones en punto flotante. La frecuencia de reloj es 1 GHz. Marque cuáles de las siguientes afirmaciones son correctas.

☒ a) La penalización media por instrucción es 3 ciclos

☐ b) El rendimiento es 500 MIPS.

☒ c) El rendimiento es 25 MFLOPS

☐ d) La duración media de una instrucción es 3 ns.

☒ e) Supongamos que los cálculos en PF consumen el 60% del tiempo de ejecución. Si introducimos UFs de punto flotante que reducen el tiempo de ejecución de las instrucciones en PF a una sexta parte, entonces el tiempo global de ejecución del programa se reduce la mitad del original.

3. Marque cuáles de las siguientes afirmaciones sobre memoria cache son correctas:

☐ a) La cache no bloqueante es una técnica para reducir la tasa de fallos

☐ b) En un sistema cache de dos niveles (L1,L2) la tasa de fallos local en L2 es igual a la tasa de fallos global en L2

☐ c) El alargamiento de arrays en una técnica para reducir el número de fallos iniciales.

☒ d) Al ejecutar un cierto programa en una cache totalmente asociativa se producen 5000 fallos. Con estos datos podemos afirmar que la suma de fallos iniciales y de capacidad ha sido 5000.

☒ e) La penalización media por instrucción (en ciclos) como consecuencia de no tener una cache perfecta, viene dada por la expresión:
[(Promedio de accesos a memoria por instrucción) x
x (Tasa de fallos) x (Ciclos de penalización por fallo)]

Arquitectura e Ingeniería de Computadores. Examen Parcial (Problemas). 14/09/2009
Problemas correspondientes al 1er cuatrimestre

1) En un DLX con segmentación ejecutamos el siguiente fragmento de código:

```
Loop : DIVD F0,F4,F2
       ADDD F0,F2,F6
       DIVD F8,F8,F2
       ADDI r3,r3,#1
       ADDD F2,F6,F8
       MULD F6,F8,F0
       LD F2, 0(r3)
       SD 0(r5), F6
       MULD F2,F6,F8
       ADDD F6,F8,F0
       ADDD F0, F2, F2
       SUBI r5,r5,#1
       BNEZ r5,Loop
       ADDD F4,F2,F2
       SUBD F6,F0,F0
end :  SUBI r3, r3, #1
```

Se supone que:

- Un dato se puede escribir en un registro y leer su valor en el mismo ciclo.
- Se dispone de lógica de cortocircuito (*forwarding*).
- Los saltos se resuelven en la etapa EX y se espera a que se resuelvan antes de lanzar la siguiente instrucción.
- La detección de todo tipo de riesgos (estructurales y LDE) y generación de paradas se realiza en la etapa de decodificación.
- Los riesgos EDE se resuelven mediante inhibición de escritura.
- Inicialmente r5=1000.
- Dos instrucciones no pueden acceder simultáneamente a la etapa de acceso a memoria ni tampoco a la de escritura en el banco de registros.
- Se dispone de las siguientes unidades funcionales:

UF	Cantidad	Latencia	Segmentación
FP ADDD	1	2	Sí
FP SUBD	1	2	Sí
FP MULD	1	3	Sí
FP DIVD	1	4	No
INT ALU	1	1	No

A) Representar el diagrama instrucción-tiempo para la primera iteración e indicar los cortocircuitos realizados así como las paradas producidas y sus causas. A la vista del diagrama obtenido, indicar el número de ciclos por instrucción (CPI) en régimen estacionario. **(3 ptos)**

B) Considere el mismo código anterior, pero suponiendo ahora que inicialmente r5=3 y que la política de gestión de saltos es tal que los saltos se siguen resolviendo en la etapa EX pero se dispone de un predictor de saltos de 2 bits del tipo BTB, al que se accede en la etapa IF, obteniendo la respuesta al final de dicha fase. Si el estado inicial del predictor es "salto no tomado débil", mostrar el diagrama instrucción-tiempo para el salto y las 3 instrucciones que se ejecutan inmediatamente después del salto, en cada una de las 3 primeras iteraciones del código. Indicar además en cada iteración si el predictor acierta o falla, el nuevo estado del predictor y los ciclos de penalización. **(1.5 ptos)**

C) Si suponemos ahora una máquina como la del apartado A, que resuelve los saltos en la etapa EX, pero que implementa saltos retardados, reescribe el código del apartado A al objeto de minimizar las penalizaciones debidas a los saltos. **(0.5 ptos)**

SOLUCIÓN

A)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
DIVD F0,F4,F2	IF	ID	D1	D2	D3	D4	M	IW																	
ADDD F0,F2,F6		IF	ID	A1	A2	M	WB																		
DIVD F8,F8,F2			IF	ID	ID	ID	D1	D2	D3	D4	M	WB													
ADDI r3,r3,#1				IF	IF	IF	ID	EX	M	WB															
ADDD F2,F6,F8							IF	ID	ID	ID	A1	A2	M	WB											
MULD F6,F8,F0								IF	IF	IF	ID	M1	M2	M3	M	WB									
LD F2,0(r3)											IF	ID	EX	M	WB										
SD 0(r5),F6												IF	ID	ID	EX	M	WB								
MULD F2,F6,F8													IF	IF	ID	M1	M2	M3	M	WB					
ADDD F6,F8,F0															IF	ID	ID	A1	A2	M	WB				
ADDD F0,F2,F2																IF	IF	ID	A1	A2	M	WB			
SUBI r5,r5,#1																		IF	ID	ID	EX	M	WB		
BNEZ r5,Loop																			IF	IF	ID	EX	M	WB	
ADDD F4,F2,F2																									
SUBD F6,F0,F0																									
SUBI r3, r3, #1																									
DIVD F0,F4,F2																							IF	ID	D1…..

XX: Parada por riesgo LDE
XX: Parada por riesgo estructural (siguiente etapa ocupada)
XX: Riesgo estructural: unidad no segmentada

XX: Riesgo EDE (inhibición de escritura)
XX: Riesgo estructural: dos instrucciones intentan acceder simultáneamente a memoria

CPI (régimen estacionario) = 22/13 = 1.69

B)

1ª Iteración (r5=2

Predicción: NTD, el salto realmente se toma → fallo. Nuevo predictor: TD

```
BNEZ r5,Loop        IF  ID  EX  M  WB
ADDD F4,F2,F2           IF  ID  X
SUBD F6,F0,F0               IF  X
DIVD F0,F4,F2                   IF.....
```

2 ciclos de penalización

2ª Iteración (r5=1)

Predicción: TD, el salto realmente se toma → acierto. Nuevo predictor: TF

```
BNEZ r5,Loop        IF  ID  EX  M  WB
DIVD F0,F4,F2           IF  ID  D1.....
ADDD F0,F2,F6               IF  ID.....
DIVD F8,F8,F2                   IF.....
```

0 ciclos de penalización

3ª Iteración (r5=0)

Predicción: TF, el salto realmente no se toma → fallo. Nuevo predictor: TD

```
BNEZ r5,Loop        IF  ID  EX  M  WB
DIVD F0,F4,F2           IF  ID  X
ADDD F0,F2,F6               IF  X
ADDD F4,F2,F2                   IF .....
```

2 ciclos de penalización

Penalización total= 4 ciclos

C) Tomar las dos instrucciones previas al SUBI r5 y ponerlas después del salto. Así se ejecutan siempre, y realmente se deben de ejecutar.

Por tanto, penalizaciones debidas a los saltos: 0