Apellidos _____ Nombre _____ Grupo___

Arquitectura e Ingeniería de Computadores. Examen Final (Teoría – parte primer cuatrimestre). 30/06/2008

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) La siguiente transformación es legal

```
        sub r1,r2,r1              sub r1,r2,r1
        beqz  r1, et              beqz  r1, et1
        nop            ⟹         xor  r4,r2,r2
        add  r4,r1,r1             add  r4,r1,r1
        …                         …
   et:  xor  r4,r2,r2        et:   xor  r4,r2,r2
        ld  r1,0(r4)         et1:  ld  r1,0(r4)
```

☐ b) Rellenar el delay slot con una instrucción del destino del salto es una opción que sólo es legal cuando el salto se toma.

☐ c) La siguiente transformación es legal

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

☒ d) La siguiente transformación es legal

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

☐ e) La arquitectura descrita implementa una forma de predicción dinámica de saltos.

2. Supongamos una arquitectura capaz de lanzar a ejecución una instrucción por ciclo de reloj, con planificación dinámica y ejecución especulativa basada en el uso de un buffer de reordenamiento (ROB). Marque las afirmaciones correctas:

☒ a) Dentro del ROB puede haber dos instrucciones que tengan el mismo registro destino.

☐ b) Cuando una UF termina su operación, envía el resultado al CDB acompañado del número de la estación de reserva de la proviene.

☐ c) Para realizar la fase de lanzamiento (issue) de una instrucción de almacenamiento es preciso que exista un "store buffer" libre.

☐ d) Supongamos dos instrucciones consecutivas de la forma:

```
muld   f2,f4,f6
subd   f8,f2,f8
```

La instrucción subd no podrá completar su fase de ejecución hasta que muld haya completado su fase "commit"

☒ e) El ROB permite implementar un modelo de interrupciones precisas.

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

☒ a) Al ejecutar un programa en un sistema con una memoria cache de un nivel, el número de fallos iniciales no depende del tamaño de la cache.

☒ b) Supongamos un sistema de memoria virtual paginada con páginas de 16 Kbytes. Si implementamos en este sistema una cache directa virtualmente accedida, pero físicamente marcada, con tamaño de bloque de 32 bytes, entonces la cache puede tener un máximo de 512 bloques.

☒ c) Al ejecutar un programa en un sistema con memoria cache totalmente asociativa nunca se producen fallos de conflicto.

☒ d) El alargamiento de arrays en una técnica para reducir el número de fallos de conflicto.

☐ 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 (Tiempo de acceso a la cache)]

Arquitectura e Ingeniería de Computadores. Examen Parcial (Problemas). 30/06/2008

Problemas correspondientes al 1er cuatrimestre

Sea un procesador segmentado con planificación dinámica mediante el algoritmo de Tomasulo

- Los datos que se escriben en la etapa de escritura no se pueden usar hasta el ciclo siguiente.
- Las instrucciones SGTI y BNEZ tienen tratamiento de instrucción entera.
- Existe un único Bus de Datos Común (BDC).

Sea el siguiente fragmento de código:

```
        :
LOOP:   LD F2 0(R1)
        MULD F4,F2,F0
        LD F6 0(R2)
        DIVD F4,F6,F4
        SD 0(R1) F4
        MULD F4,F6,F0
        ADD F4,F8,F2
        ADDI R1,R1,#8
        ADDI R2,R2,#8
        SGTI R3,R2,DONE
        BNEZ R3,LOOP
```

ESTACIONES RESERVA	CANTIDAD
FP ADD	2
FP DIV	1
FP MUL	2
INT ALU	2
LOAD	2
STORE	2

UF	CANTIDAD	LATENCIA	SEGMENTADA
FP ADD	1	2	SI
FP DIV	1	6	SI
FP MUL	1	3	SI
INT ALU	1	1	SI
MEM	1	2	SI

a) Indicar el diagrama instrucción-tiempo para la primera iteración.

b) Suponiendo que se le añade especulación basada en la utilización de un buffer de reordenamiento (ROB), indicar el diagrama instrucción-tiempo para la primera iteración. Se supone que el papel de los "store buffers" es asumido por el ROB.

c) Indicar el diagrama instrucción-tiempo para la primera iteración, suponiendo un superescalar sin especulación con las mismas unidades funcionales ya vistas, que lanza un par de instrucciones de cualquier tipo en un ciclo de reloj. En caso de dependencia de datos entre dos instrucciones de un par se congela el lanzamiento de la segunda. Se supone que existen dos buses comunes de datos, que el banco de registros puede realizar dos escrituras en cada ciclo de reloj, y que existen suficientes estaciones de reserva para que no se produzcan paradas.

Soluciones:

A)

		ISSUE	EJECUCIÓN	WRITE
1	LD F2 0(R1)	1	2-3	4
2	MULD F4,F2,F0	2	5-6-7 (LDE CON 1)	8
3	LD F6 0(R2)	3	4-5	6
4	DIVD F4,F6,F4	4	9-14(LDE CON 2)	15
5	SD 0(R1) F4	5	16-17(LDE CON 4)	
6	MULD F4,F6,F0	6	7-8-9	10
7	ADD F4,F8,F2	7	8-9	11 (BCD CON 6)
8	ADDI R1,R1,#1	8	9	12 (BCD CON 6 Y 7)
9	ADDI R2,R2,,#8	9	10	13 (BCD CON 7 Y 8)
10	SGTI R3,R2,DONE	13 (ESTRUCTURAL HASTA QUE LIBERA 8)	14	16 (BCD CON 4)
11	BNEZ R3,LOOP	14	17 (LDE CON 10)	18

B)

Recordar que con ROB no existen estaciones de reserva de store sino que se incluyen en el ROB. De manera que en el ROB se pueden realizar dos escrituras simultáneas, y en las instrucciones de store con dependencias del tipo LDE el dato se envía directamente al ROB. Esta es la razón por la que las instrucciones de DIVD y store escriben en el ROB en el mismo ciclo de reloj. La información y la etiqueta que se envía es la misma para ambas.

		ISSUE	EJECUCIÓN	WRITE	COMMIT
1	LD F2 0(R1)	1	2-3	4	5
2	MULD F4,F2,F0	2	5-6-7	8	9
3	LD F6 0(R2)	3	4-5	6	10
4	DIVD F4,F6,F4	4	9-14(LDE CON 2)	15	16
5	SD 0(R1) F4	5		15 (LDE CON 4)	17
6	MULD F4,F6,F0	6	7-8-9	10	18
7	ADD F4,F8,F2	7	8-9	11 (BCD CON 6)	19
8	ADDI R1,R1,#1	8	9	12 (BCD CON 6 Y 7)	20
9	ADDI R2,R2,,#8	9	10	13 (BCD CON 7 Y 8)	21
10	SGTI R3,R2,DONE	13 (ESTRUCTURAL HASTA QUE LIBERA 8)	14	16 (BCD CON 4)	22
11	BNEZ R3,LOOP	14	17 (LDE CON 10)	18	23

(*) Supongo que al llegar a la cabeza del ROB envía el store a la unidad de memoria para que realice la escritura, por lo tanto cuando ha acabado su commit todavía está la unidad de memoria realizando la escritura.

	ISSUE	EJECUCIÓN	WRITE
LD F2 0(R1)	1	2-3	4
MULD F4,F2,F0	2 (LDE con 1)	5-6-7 (LDE)	8
LD F6 0(R2)	2	3-4	5
DIVD F4,F6,F4	3	9-14 (LDE)	15
SD 0(R1) F4	4 (LDE con 4)	16-17 (LDE)	
MULD F4,F6,F0	4	6-8	9
ADD F4,F8,F2	5	6,7	8
ADDI R1,R1,#1	5	6	7
ADDI R2,R2,,#8	6	7	9 (bcd)
SGTI R3,R2,DONE	7 (LDE con 9)	10 (LDE)	11
BNEZ R3,LOOP	8 (LDE con 10)	12 (LDE)	13