Arquitectura e Ingeniería de Computadores. Examen Parcial (Problemas). 13/09/2010

Nombre----------Grupo----------

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

```
L0:  MULD F2,F0,F4
     LD F2, 0(r1)
     ADDD F0,F6,F2
     ADDI r5,r0,#1
     DIVD F6,F0,F2
     BNEZ r5,L1
     ADDD F2,F0,F4
     MULD F6,F2,F0
 L1: DIVD F2,F0,F0
     SD 0(r5),F2
     ADDD F6,F0,F2
     SUBI r5,r5,#1
end: LD F6, 0(r5)
```

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 tienen tratamiento de instrucción entera, se resuelven en la etapa de ejecución (EX) y se espera a que se resuelva el salto antes de buscar la siguiente instrucción.
- La detección de los riesgos estructurales, LDE y EDE y generación de paradas se realiza en la etapa de decodificación.
- Dos instrucciones no pueden acceder simultáneamente a la etapa de acceso a memoria ni tampoco a la de escritura en el banco de registros.
- El registro r0 contiene siempre el valor 0.
- Se dispone de las siguientes unidades funcionales:

UF	Cantidad	Latencia	Segmentación
FP ADDD	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 ejecución del código e indicar los cortocircuitos realizados. Indicar claramente las paradas y sus causas. A la vista del diagrama obtenido, determinar el número de ciclos que toma la ejecución completa del código. **(2.5 ptos)**

B) ¿Cómo se vería afectado el valor obtenido en el apartado anterior en el caso de que en lugar de esperar a que los saltos se resuelvan se implementase la técnica de saltos retardados y el compilador tratase de insertar la instrucción de división y el store posteriores al salto en el delay slot? **(1 ptos)**

2) Se dispone de un procesador segmentado con un juego de instrucciones de tipo entero basadas en el procesador DLX. El cálculo de la condición de salto y dirección destino en caso de instrucción de bifurcación se realiza en la etapa de ejecución (EX). Dicho procesador posee un predictor del tipo BTB de 1 bit, que se accede durante la fase IF, obteniendo la respuesta al final de dicha fase. Dada la siguiente secuencia de código, y suponiendo que el buffer del predictor contiene el estado "predicción no salta" para la instrucción de salto, mostrar las fases de las tres instrucciones que se ejecutan inmediatamente después del salto en cada una de las 3 primeras iteraciones. **(1.5 ptos)**

```
        ADDI r1,r0,#3
bucle:  SUB r2,r2,r3
        ADD r5,r1,r5
        SUBI r1,r1,#1
        BNEZ r1,bucle
        ADDI r6,r6,#2
        SUB r2,r5,r2
        OR r3,r3,r2
```

SOLUCIÓN

1)

	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	26	27
MULD F2,F0,F4	IF	ID	M1	M2	M3	M	WB																				
LD F2, 0(r1)		IF	ID	ID	ID	EX	M	WB																			
ADDD F0,F6,F2			IF	IF	IF	ID	ID	A1	A2	M	WB																
ADDI r5,r0,#1						IF	IF	ID	ID	EX	M	WB															
DIVD F6,F0,F2								IF	IF	ID	D1	D2	D3	D4	M	WB											
BNEZ r5,L1										IF	ID	EX	M	WB													
ADDD F2,F0,F4																											
MULD F6,F2,F0																											
DIVD F2,F0,F0											IF	IF	IF	ID	D1	D2	D3	D4	M	WB							
SD 0(r5),F2														IF	ID	ID	ID	ID	EX	M	WB						
ADDD F6,F0,F2															IF	IF	IF	IF	ID	A1	A2	M	WB				
SUBI r5,r5,#1																			IF	ID	ID	EX	M	WB			
LD F6, 0(r5)																				IF	IF	ID	EX	M	WB		

XX: Parada por riesgo LDE
XX: Parada por riesgo estructural (siguiente etapa ocupada)
XX: Riesgo de control (salto espera a resolverse)

XX: Riesgo EDE
XX: Riesgo estructural: dos instrucciones intentan acceder simultáneamente a memoria

A) La ejecución completa del código toma 25 ciclos

B) Si el compilador trata de insertar la instrucción de división en el slot, ésta podría hacer su fetch efectivo en el ciclo en el ciclo 11, sin embargo debería mantenerse en la etapa ID hasta el ciclo 14 por un riesgo estructural (divisor no segmentado) con la instrucción de división anterior, por lo que el diagrama sería el mismo que en el caso a) y no podríamos ahorrarnos ningún ciclo (serían también 25 ciclos en total)

2)

1ª Iteración (r1=2); Predictor = salto no tomado → Fallo → predictor pasa a salto tomado

```
BNEZ r1, bucle     IF  ID  EX  M  WB
ADDI r6,r6,#2          IF  ID   X    X ….
SUB r2,r5,r2               IF   X    X…..
SUB r2,r2,r3                   IF   ID   EX…….
```

2ª Iteración (r1=1); Predictor = salto tomado → Acierto → predictor sigue a salto tomado

```
BNEZ r1, bucle     IF  ID  EX  M  WB
SUB r2,r2,r3           IF  ID   EX   M…….
ADD r5,r1,r5               IF   ID    EX…..
SUBI r1,r1,#                   IF   ID   EX…….
```

3ª Iteración (r1=0); Predictor = salto tomado → Fallo → predictor pasa a salto no tomado

```
BNEZ r1, bucle     IF  ID  EX  M  WB
SUB r2,r2,r3           IF  ID   X   X…….
ADD r5,r1,r5               IF   X    X…..
ADDI r6,r6,#2                  IF   ID   EX……
```