Arquitectura de Computadores
Problemas (hoja 2). Curso 2015-16

1. Sea la siguiente secuencia de código de instrucciones en punto flotante para un computador similar al DLX que aplica gestión dinámica de instrucciones basada en el algoritmo de Tomasulo:

```
LD    F6,x(R1)
LD    F2,y(R1)
MULTD F0,F2,F4
SUBD  F8,F6,F2
DIVD  F6,F0,F6
ADDD  F10,F0,F6
ADDD  F6,F8,F2
SD    z(R1),F6
```

Indica el estado de la unidad de ejecución en punto flotante en el momento en que todas las instrucciones han sido lanzadas a ejecución en la misma, si la latencia de la memoria es tal que sólo la primera se ha ejecutado completamente, habiéndose cargado en F6 el dato x. Suponer que el resto de registros Fi, poseen un valor dado por la expresión [Fi].
Suponer que hay suficientes estaciones de reserva y buffers de load/store, y no se puede escribir un dato desde el CDB e iniciar en el mismo ciclo la operación en espera de ese dato.

2. Aplica el método de Tomasulo a la siguiente versión del bucle AXPY para el procesador DLX funcionando a 800 MHz. Las etapas del pipeline son:

- **IF** (búsqueda),
- **ID** (decodificación),
- **Issue** (lanzamiento a ejecución, aplicando Tomasulo),
- **EX** (ejecución en el operador multiciclo) y
- **WB** (escritura en el bus común de datos).

Todos los operadores multiciclo están segmentados y las latencias son las siguientes: 2 ciclos para load/store, 7 ciclos para suma/resta y 10 ciclos para multiplicación/división.

```
bucle:      LD          F2,x(R1)
            MULTD       F4,F2,F0
            LD          F6,y(R1)
            ADDD        F6,F4,F6
            SD          y(R1),F6
            BNEZ        R1,bucle
            SUBI        R1,R1,#8
```

Suponiendo que en el mismo ciclo se puede escribir un dato en el bus común de datos e iniciar una operación que estuviera en espera de ese dato, dibuja el diagrama instrucción/tiempo correspondiente a las dos primeras iteraciones.
¿Cuál es la velocidad de ejecución sostenida del bucle en MFLOPS?
Suponer suficientes estaciones de reserva y buffer de load/store.

3. Cierto programa consume la mayor parte de su tiempo de ejecución en el siguiente fragmento de código:

$$Y = Y + X$$

donde X e Y son vectores. Dicho programa se pretende ejecutar sobre un computador con gestión dinámica de instrucciones que usa el método de Tomasulo. El computador posee operaciones de punto flotante en su juego de instrucciones, funciona a una frecuencia de 1 Ghz, el CPI de las instrucciones enteras es 1 y el *delay slot* de los saltos es de 1 instrucción.
Las características de las unidades funcionales de la máquina son las siguientes:

	Nº FUs	latencia (ciclos)	Segmentada?	Buffers o ER
Load	1	2	No	3
+	1	3	No	2
*	1	4	No	2
Store	1	2	No	3

El código generado por el compilador es el siguiente:

```
loop: LD    F2,X(R1)
      LD    F4,Y(R1)
      ADDD  F4,F2,F4
      SD    Y(R1),F4
      BNEZ  R1,loop
      SUB   R1,R1,#8
```

Calcula el tiempo necesario, expresado en segundos, para procesar el fragmento de código, así como la velocidad de ejecución del mismo en MFLOPS para vectores de tamaño muy grande.

4. Un computador con gestión dinámica de instrucciones que usa el método de Tomasulo trabaja a una frecuencia de 1,2 GHz. El *delay slot* de los saltos es de 1 instrucción y todas las unidades funcionales multiciclo son segmentadas lineales, con las siguientes etapas:

Load	ADDD	Store
3 etapas	4 etapas	3 etapas

Se pretende ejecutar el siguiente fragmento de código sobre dicha máquina:

```
      ; R1 = 256
loop: SUB   R1, R1, #4
      LD    F0, y(R1)
      LD    F2, z(R1)
      LD    F4, t(R1)
      ADDD  F6, F4, F0
      ADDD  F8, F2, F0
      ADDD  F6, F6, F8
      BNEZ  R1, loop
      SD    x(R1),F6
```

¿Cuántas estaciones de reserva, buffers de lectura y buffers de escritura hacen falta para ejecutarlo lanzando una instrucción por ciclo? Calcula los MFLOPS obtenidos al ejecutar el programa.

5. Un procesador con planificación dinámica de instrucciones mediante el algoritmo de Tomasulo tiene una unidad de punto flotante que emplea 2 ciclos para la suma, 10 para la multiplicación, y 30 para la división. Esta unidad de punto flotante dispone de 1 unidad de multiplicación/división segmentada con 2 estaciones de reserva, y 1 unidad de suma/resta segmentada con 3 estaciones de reserva. Mostrar el diagrama instrucciones-tiempo para el código que se muestra a continuación:

```
DIVD F10, F11, F5
ADDD F6, F10, F1
MULD F7, F6, F4
ADDD F1, F13, F14
ADDD F4, F1, F17
ADDD F15, F16, F17
```

6. Cierto programa consume la mayor parte de su tiempo de ejecución en el siguiente fragmento de código:

T = a +Y +Z, donde T, Y, Z son vectores de n componentes y a un escalar.

Dicho programa se pretende ejecutar sobre un computador DLX cuyas características son:

(a) Frecuencia de reloj: 100 MHz.
(b) Gestión dinámica de instrucciones basada en el algoritmo de Tomasulo (tiempo de transferencia por el bus común de datos: 1 ciclo)
(c) Unidades de coma flotante **no segmentadas**: 1 de carga (4 ciclos, 2 buffer), 1 operador multifunción (3 ciclos, 3 estaciones de reserva) y 1 de almacenamiento (4 ciclos, 2 buffers).
(d) CPI para las instrucciones enteras = 1, y *delay-slot* =1.

El código generado por el compilador disponible es el siguiente:

```
; F0 contiene a
loop: LD    F4,Y(R1)
      LD    F2,Z(R1)
      ADDD  F4,F0,F4
      ADDD  F4,F2,F4
      SD    T(R1),F4
      BNEZ  R1,loop
      SUB   R1,R1,#8
```

Calcula la expresión del tiempo de ejecución del bucle en función del tamaño de los vectores n.

7. Disponemos de un procesador dotado de planificación dinámica de instrucciones según el método de Tomasulo. Se dispone de las unidades funcionales que se muestran en la tabla siguiente:

UF	Cantidad	Latencia	Segmentación
FP ADDD	2	3	Sí
FP MUL/DIV	1	4/10	No

Suponer que en un mismo ciclo no se puede escribir un dato en el bus común e iniciar la ejecución de una operación que estuviera en espera de ese dato, y que existe una estación de reserva por cada unidad funcional.
Para la secuencia de instrucciones de la tabla siguiente, indica en qué ciclo (o ciclos) de reloj se realiza cada una de sus fases. Indica qué tipos de riesgos se producen.

Instrucción	Issue Emisión	Ejecución	Escribe resultado
ADDD F2,F2,F4			
ADDD F0,F4,F0			
MULTD F4,F2,F0			
ADDD F6, F0,F0			
DIVD F4,F4,F6			
ADDD F2,F6,F6			
ADDD F0,F2,F6			
MULTD F2,F2,F0			

8. Disponemos de un procesador dotado de planificación dinámica de instrucciones según el método de Tomasulo. Se dispone de las unidades funcionales que se muestran en la tabla siguiente:

UF	Cantidad	Latencia	Segmentación
FP ADD/SUB	2	3/3	No
FP MUL/DIV	2	6/12	No

Suponer que en un mismo ciclo no se puede escribir un dato en el bus común e iniciar la ejecución de una operación que estuviera en espera de ese dato, y que existe una estación de reserva por cada unidad funcional.

Para la secuencia de instrucciones de la tabla siguiente, indica en qué ciclo (o ciclos) de reloj se realiza cada una de sus fases. Indica qué tipos de riesgos se producen.

Instrucción	Issue Emisión	Ejecución	Escribe resultado
DIVD F2, F2, F6			
ADDD F4, F6, F4			
MULD F8, F2, F4			
DIVD F0, F6, F4			
ADDD F2, F4, F0			
ADDD F8, F8, F10			
MULD F0, F2, F8			
SUBD F12, F2, F4			

9. 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).
- La unidad de ejecución tiene las siguientes características

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

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
```

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.

10. Se dispone de un procesador segmentado con un juego de instrucciones de tipo entero basadas en el procesador DLX. Las fases que atraviesa una instrucción son:

- **IF:** Búsqueda de la instrucción.
- **ID:** Decodificación de la instrucción. Cálculo de la condición de salto y dirección de destino en caso de instrucción de bifurcación.
- **EX:** Fase de ejecución aritmética, y cálculo de la dirección de acceso a memoria, en su caso.
- **ME:** Acceso a memoria de datos, en su caso.
- **WB:** Escritura en registros.

Dicho procesador posee un predictor del tipo **Branch Target Buffer** de **2 bits**, 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-fuerte"** para la instrucción "`BNEZ R1,loop`", mostrar las fases de las dos instrucciones que se ejecutan inmediatamente después de este salto, en cada una de las cuatro primeras iteraciones. En caso de que alguna instrucción se cancele, marcar con una '**X**' las fases que no ejecuta.

```
      ...
      ADD   R1,R0,#4
loop: SUB   R5,R5,R2
      ADD   R2,R2,#1
      SUB   R1,R1,#1
      BNEZ  R1,loop
      AND   R6,R5,#63
      OR    R6,R6,#128
      ...
```

11. Se quiere diseñar un procesador con predicción dinámica de saltos por medio de una tabla de historia de saltos. La mayoría de los programas que el procesador ejecutará están compuestos por bucles anidados a tres niveles:

```
FOR i = 1 ... n
     FOR j = 1 ... n
          FOR k = 1 ... n
               ....
               ....
          END
     END
END
```

¿Cuantos fallos se producirán con las dos alternativas estudiadas en clase, usando 1 bit (2 estados) y 2 bits (4 estados), si en ambos casos se parte de no saltar (en el caso de 2 bits se parte del "no saltar fuerte")?

12. Supongamos un procesador con predicción dinámica de saltos que usa un predictor competitivo (Tournament Predictor) como el del Alpha 21264. Supongamos un programa que posee dos únicas instrucciones de salto con la siguiente estructura:

```
        Inst1
loop:   Inst2
        Inst3
        Inst4
        BEQ R1, loop
        Inst6
        Inst7
        BNE R2, loop
```

donde “Insti“ representa una instrucción que no es de salto. Supongamos que la instrucción BEQ se ha ejecutado 1000 veces con el siguiente comportamiento: T-T-NT-NT-T-T-NT-NT-……El comportamiento de la instrucción BNE es que se toma siempre.

a) Determinar qué entradas de la Tabla de Predicción Local han sido accedidas como consecuencia de las 100 últimas ejecuciones de la instrucción BEQ y determinar cuál es el contenido de dichas entradas al final de las 1000 ejecuciones.

b) Si la instrucción BEQ se ejecuta 1000 veces más, siguiendo el mismo patrón de comportamiento, ¿cuántos fallos de predicción se producen en estas 1000 ejecuciones?