Arquitectura de computadoras
Ejercicios Pipelining

1. Considere un procesador con un pipeline de seis etapas (F, D, CO, FO, E, WO). Para el siguiente segmento de codigo:

   ADD R3, R2, R1
   SUB R4, R3, R5

   (a) ¿En qué ciclo la instrucción ADD actualiza R3?

   (b) ¿En qué ciclo SUB lee R3?

   (c) ¿En qué ciclo SUB ejecuta con el valor de R3?

2. Un procesador cuenta con cuatro etapas que tardan respectivamente 50, 30, 95 y 45 nSeg. Al introducir pipelining, los buffers auxiliares agregan un retardo adicional de 5 nSeg cada uno.

   (a) ¿En cuánto tiempo se ejecutan 100 instrucciones suponiendo que no hay ningún tipo de bloqueo entre ellas?

   (b) ¿Qué podría hacerse para mejorar el desempeño del procesador?

3. Implemente un programa en ensamblador para una arquitecura de 2 direcciones que permita ejecutar el código $X = (A + B)/(C + D)$. El procesador tiene un pipeline de cuatro etapas (Fetch, Busca operando, Ejecuta, Guarda) y memorias cache separadas para instrucciones y datos. El código debe evitar bloqueos de pipeline.

4. Las etapas de un procesador con pipeline con 5 etapas tienen las siguientes latencias:

   F: 1 nSeg; D: 1.5 nSeg; E: 1 nSeg; M: 2 nSeg; W: 1.5 nSeg.

   Los registros entre etapas introducen una latencia de 0.1 nSeg.

   (a) ¿Cuál es el periodo de reloj que controla este pipeline?

   (b) Si hay un bloqueo de un ciclo cuada cuatro instrucciones, ¿Cuál es el CPI efectivo?

   (c) Cuál es la aceleración (*speedup*) obtenida al introducir el pipeline en la arquitectura

5. Considere el siguiente código:

```
Loop:  LD      R1,0(R2)    ;R1← M[R2+0]
       DADDI   R1,R1,#1    ;R1←R1+1
       SD      R1,0,(R2)   ;M[R2+0]←R1
       DADDI   R2,R2,#4    ;R2←R2+4
       DSUB    R4,R3,R2    ;R4←R3-R2
       BNEZ    R4,Loop     ;Si R4 != 0, va a Loop
```

   El valor inicial de R3 es R2+396.

   (a) ¿Qué hace este código?

(b) Identifique los posibles conflictos de datos. Muestre el registro, las instrucciones fuente y destino que ocasionan el conflicto, y el tipo de dependencia.

(c) Muestre el diagrama de tiempo para el pipeline de 5 etapas que se ha usado en clase. Suponga que no hay atajos, pero que R y W al banco de registros sí pueden ejecutarse en el mismo ciclo. La instrucción de slto vacía el pipeline.

(d) Si todas las referencias a memoria toman un ciclo, ¿Cuántos ciclos toma ejecutar este código?

(e) Ahora muestre el diagrama de tiempo para el pipeline suponiendo atajos. Suponga que los saltos usan una técnica de predicción "no tomado". En este caso, ¿Cuánto tiempo toma ejecutar el código?

(f) Repita la pregunta anterior pero ahora suponga que la técnica de predicción de salto es "tomado" siempre.

(g) Suponga ahora que el procesador es superpipeline con 2 subetapas por etapa (pipeline de 10 etapas). Los atajos sólo pueden realizarse del final de la etapa donde se genera el resultado (2a subetapa), al principio de aquélla donde se necesita (1a subetapa). Repita la pregunta anterior suponiendo que la predicción es "tomado" siempre.

(h) Para el pipeline de 5 etapas, si la etapa más lenta es de 0.8 nSeg y los registros entre etapas introducen un retraso de 100 ps, ¿Cuál es el ciclo de reloj? ¿Cuál sería el ciclo de reloj para el procesador superpipeline de 10 etapas?

(i) Ignorando el tiempo tomado para llenar el pipeline inicial, calcule el CPI para los resultados obtenidos en los incisos (f) y (g).

6. El procesador A tiene un pipeline de 5 etapas con un ciclo de reloj de 1 nSeg. Introduce un bloqueo por conflictos de datos cada 5 instrucciones. El procesador B tiene 12 etapas y un ciclo de reloj de 0.6 nSeg, pero los conflictos de datos introducen un retraso de 3 ciclos cada 8 instrucciones. En ambos casos, la frecuencia de saltos es de 20% de las instrucciones y la tasa de error en la predicción del destino es de 5%.

   (a) ¿Cuál es el speedup del procesador B sobre el A considerando sólo conflictos de datos?

   (b) Si la penalización por fallos en la predicción de saltos es de 2 ciclos para el procesador A y 5 para el B, ¿cuál es el CPI de cada procesador tomando en cuenta los bloqueos de pipeline debidos a saltos?

7. Considere un procesador con el siguiente predector de saltos de dos niveles. El primer nivel es un predictor de un bit que siempre predice igual que la última ejecución de salto: si el salto se tomó, predice que se tomará y viceversa. Este predictor se inicializa con el valor "salto tomado". Las predicciones correctas en este nivel no incurren en ningún costo.

   El segundo nivel es un predictor de dos bits. Sus estados son:

   00 No salto con alta probabilidad
   01 No salto con baja probabilidad
   10 Salto con baja probabilidad
   11 Salto con alta probabilidad

Este predictor se inicializa a 10; sus predicciones incurren en un costo de 1 ciclo de reloj. Las correcciones hechas por la unidad de evaluación tienen un costo de 2 ciclos.

Los costos anteriores no son acumulativos. si el predictor de primer nivel toma la decisión correcta y el segundo nivel lo "corrige" incorrectamente pero la unidad de saltos valida el primer predictor, no hay costo.

Ahora considere el siguiente código:

```
i = 0;
   do {
      j = 0;
      do {
         j = j + 1;
       } while (j < 10);
       i = i + 1;
   } while (i < 10);
```

(a) ¿Cuál es el costo promedio de salto para la ramificación al final del loop externo?

(b) ¿Cuál es el costo promedio de salto para la ramificación al final del loop interno?

(c) ¿Cuál es el costo promedio de salto para todos los saltos en el programa?

8. (a) Identifique todas las dependencias de datos en el código siguiente, que se ejecuta en el pipeline MIPS de 5 etapas. ¿Qué dependencias pueden resolverse con la técnica de *forwarding*?

```
ADD   R2,R5,R4
ADD   R4,R2,R5
SW    R5,100(R2)
ADD   R3,R2,R4
```

(b) Para la ejecución del siguiente código, ¿Qué registros se leen durante el quinto ciclo y cuáles se escriben al final del quinto ciclo de reloj?

```
ADD   R1,R2,R3
ADD   R4,R5,R6
ADD   R7,R8,R9
ADD   R10,R11,R12
ADD   R13,R14,R15
```

9. Un procesador con despacho dinámico tiene tres unidades funcionales (FU): una LD/STORE que toma dos ciclos, una ADD/SUB, de un ciclo de ejecución, y una MUL/DIV con 2 y 4 ciclos de ejecución, respectivamente. Tiene un solo arreglo de registros y una estación de reservación con un espacio por FU. El procesador tiene una política de emisión y completado de instrucciones fuera de orden (OOI, OOC).

   Comenzando con la secuencia de instrucciones siguiente en el buffer de instrucciones, y con los espacios libres en las estaciones de reservación, identifique el ciclo en el que la instrucción será emitida y en el que escribirá su resultado.

```
LD    R6, 34(R12)
LD    R2, 45(R13)
MUL   R0, R2, R4
SUB   R8, R2, R6
DIV   R10, R0, R6
ADD   R6, R8, R2
```

10. Considere un procesador VLIW con tres unidades enteras (X, Y, Z), una de multiplicación (M), y dos LD/ST (LS0, LS1). Las instrucciones ALU tienen una latencia de 1; M de 5 y LD/ST de 3. Se puede ejecutar un salto por ciclo y utiliza la unidad Z. Para el siguiente código, desenrolle el loop e implemente un pipeline por software para maximizar el desepeño. Muestre el código en el prólogo y el epílogo. Siéntase en libertad de utilizar más registros y renombrarlos, así como de reorganizar el código para aumentar el desempeño.

    ¿Cuál es la tasa media de multiplicaciones por ciclo en el medio del loop?

```
//R0 is hardwired to zero
//R31 is the link register
//R1 contains pointer to input array (elements are word sized)
//R2 contains pointer to output array (elements are word sized)
// (Output array is larger than input array)
//R3 contains the size of the input array in words

function:
ADDI R16, R0, 0x456
ADDI R17, R0, 0x789
ADDI R18, R0, 0x901
ADD R4, R0, R0
ADD R5, R0, R0

loop:
LW R6, 0(R1)
MUL R8, R4, R16
MUL R9, R5, R17
MUL R10, R6, R18
ADDI R1, R1, 4
SUBI R3, R3, 1
ADD R8, R8, R9
ADD R8, R8, R10
ADD R4, R5, 0
ADD R5, R6, 0
SW R8, 0(R2)
ADDI R2, R2, 4
BNEZ R3, loop
MUL R8, R4, R16
```

```
MUL R9, R5, R17
ADD R8, R8, R9
SW R8, 0(R2)
ADDI R2, R2, 4
ADD R4, R5, 0
MUL R8, R4, R16
SW R8, 0(R2)
JR R31
```