

UNIVERSIDAD CARLOS III DE MADRID

Tema 6. Segmentación II

Departamento de Ingeniería de Sistemas y
Automática

RAÚL PÉRULA MARTÍNEZ
LUIS ENRIQUE MORENO LORENTE
ALBERTO BRUNETE GONZALEZ
CESAR AUGUSTO ARISMENDI GUTIERREZ
DOMINGO MIGUEL GUINEA GARCIA ALEGRE
JOSÉ CARLOS CASTILLO MONTOYA



Universidad
Carlos III de Madrid



Esta obra se publica bajo una licencia Creative Commons Reconocimiento-NoComercial-CompartidIgual 3.0 España.



Ejercicio 1

Suponiendo que se tiene el siguiente código en un cierto lenguaje ensamblador.

```
loop: ld    f2, 0(r1)
      multd f4, f2, f0
      multd f8, f2, f2
      ld    f6, 0(r2)
      addd  f6, f4, f6
      subd  f8, f8, f6
      sd    0(r2), f8
      addi  r1, r1, 8
      addi  r2, r2, 8
      subi  r3, r3, 1
      beqz  r3, loop
```

Donde se lista en primer lugar el operando de destino y a continuación los operandos fuente, y en el caso de las instrucciones de salto se listan primero los operandos a ser comparados y después la dirección de salto.

Se pide:

1. Dado el esquema de la arquitectura del procesador DLX mostrar el cronograma de ejecución suponiendo que el control de riesgos se realiza por interbloqueo.
2. Muestre el cronograma suponiendo que además de hardware para interbloqueo dispone de hardware para realizar anticipación de datos (forwarding).



Ejercicio 2

Sea el siguiente fragmento de código de un cierto lenguaje vectorial que procesa un vector X.

```
. . .  
%  
X = X*a  
%  
. . .
```

Donde X es una matriz de dimensión $n \times 1$ (vector columna) y a es una matriz de dimensión 1×1 (escalar). Se compila para un procesador segmentado con arquitectura DLX de cinco etapas que calcula la dirección efectiva de salto en la etapa ID. Un extracto del código es el siguiente (se han numerado las instrucciones como i_0 a i_6):

```
i_0      ld    f0, dir_a  
i_1  loop: ld    f6, 0(r2)  
i_2      multd f6, f6, f0  
i_3      sd    0(r2), f6  
i_4      addi  r2, r2, 8  
i_5      sgt   r3, r2, r1  
i_6      beqz  r3, loop
```

Se pide:

1. Muestre el cronograma de ejecución de las instrucciones correspondientes a dos iteraciones del bucle, suponiendo que el control de riesgos se realiza por interbloqueo.
2. Muestre el cronograma de ejecución de las instrucciones correspondientes a dos iteraciones del bucle, suponiendo que además de hardware para interbloqueo, dispone de hardware para realizar anticipación de datos.



Ejercicio 3

Suponiendo que se tiene el siguiente código en un cierto lenguaje ensamblador.

```
L0:  ld    r3, #10
      ld    r1, DIR_X
      ld    r2, DIR_Y
1  L1:  ld    f2, 0(r1)
2      ld    f4, 4(r1)
3      add  f6, f2, f4
4      add  f8, f2, f6
5      store 0(r2), f8
6      add  r1, r1, #4
7      add  r2, r2, #4
8      sub  r3, r3, #-4
9      blt  r3, 0, L1
```

Donde se lista en primer lugar el operando de destino y a continuación los operandos fuente, y en el caso de las instrucciones de salto se listan primero los operandos a ser comparados y después la dirección de salto.

Se pide:

1. Obtener el grafo de dependencias entre las instrucciones 1 a 9, en una iteración del bucle.
2. Obtener el cronograma de ejecución suponiendo que la arquitectura del procesador es la DLX, que se utiliza planificación estática y que los riesgos de datos en la CPU se resuelven mediante:
 - a. Software (compilador), insertando instrucciones **no-op**.
 - b. Hardware, mediante interbloqueo.
 - c. Hardware, mediante interbloqueo y anticipación.
3. Indique si es posible alguna reordenación del código que mejore la velocidad de ejecución y muestre en dicho caso el cronograma de ejecución del código reordenado.