UNIVERSIDAD CARLOS III DE MADRID

Trabajo 3

Departamento de Ingeniería de Sistemas y Automática

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

Universidad Carlos III de Madrid

Enunciado

Dado el siguiente programa:

```
for (int i = 0; i < 10; ++i) {
     if (i < 2)
          X[i] = (double) i;
     else{
          X[i] = X[i-1] + X[i-2];
          gold = X[i] / X[i-1];
     }
}
```

Que se traduce en el siguiente código ensamblador WinMIPS64:

```
.data
     Xtab: .space 80
     Zero: .double 0.0
     One:  .double 1.0
     Gold: .double 0.0
.text
     main:
     daddi r1,r0,Xtab  ; X address
     daddi r2,r0,Gold  ; Gold number address
     daddi r3,r0,0     ; loop index r3 = 0
Loop:
     1     slti r4,r3,2      ; Test r3 < 2
     2     beqz r4,L2        ; Jump if condition is false
L1:
     3     mtc1 r3,f2
     4     cvt.d.l f4,f2     ; f4 = (double) r3
     5     s.d f4,0(r1)
     6     j L3
L2:
     7     l.d f4,-8(r1)
     8     l.d f6,-16(r1)
     9     add.d f8, f4, f6  ; f8 = X[i-1] + X[i-2]
     10    s.d f8,0(r1)
     11    div.d f10,f8,f4   ; f10 = X[i] / X[i-1]
     12    s.d f10,0(r2)
L3:
     13    daddi r3,r3,1
     14    daddi r1,r1,8
     15    slti r4,r3,10
     16    bnez r4,Loop
Finish:
     halt
```

Considerar la siguiente arquitectura:

- 2 ciclos para la suma en coma flotante.
- 5 ciclos para la multiplicación.
- 10 ciclos para la división.

[A mano]

1. Obtener el grafo de dependencias de las instrucciones del bucle.
2. Considerando que `i < 2` (se ejecuta L1), determinar el cronograma y calcular el CPI de una iteración del bucle:
   a) Con interbloqueo hardware y *forwarding* (sin predicción de salto).
   b) Con interbloqueo hardware, *forwarding* y salto retardado.
3. Considerando que i > 2 (se ejecuta L2), determinar el cronograma y calcular el CPI de una iteración del bucle con el algoritmo de *Tomasulo* y predicción de salto dinámica.

[Con WinMIPS64]

4. Determinar el CPI completo del programa con **WinMIPS64** (con *forwarding*). Comprobar que el programa funciona analizando el contenido de la memoria. Indicar los valores del vector X y el valor del "Golden number" al final de la ejecución del programa. Si es necesario, dé valores iniciales al vector X.
   Tenga en cuenta que el valor en la memoria de datos está en formato hexadecimal doble. Al hacer doble clic con el botón derecho del ratón, se muestra el mismo número en formato decimal.
5. Desenrollar el bucle para que cada iteración corresponda con 2 del código original. Comprobar que el programa sigue funcionando de la misma forma y dando el mismo resultado en el WinMIPS64.
6. ¿Cuál es el nuevo CPI? Calcular el *speedup*.
7. Reordenar las instrucciones del bucle, eliminar las instrucciones inútiles y renombrar los registros para optimizar aún mas el código. Comprobar que el programa sigue funcionando. ¿Cual es el CPI final?

Nota: Entregar los ficheros *.s* realizados en los distintos apartados.