UNIVERSIDAD CARLOS III DE MADRID

Trabajo 2

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

Suponiendo que el siguiente programa:

```
for (int i = 0; i < 10; ++i) {
     if (X[i] > 0)
          Y[i] = 1 / X[i] + 1 / (X[i] * X[i]);
     else
          Y[i] = 0;
}
```

tiene la siguiente traducción al ensamblador de WinDLX:

```
.data
Xtab: .double 0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9
Ytab: .space      10*8
One: .double 1.0
counter: .word 12
.text

main:
      l.d f30,One(r0)
      daddi r3,r0,10    ; Indice del loop
      daddi r1,r0,Xtab  ; Dirección de X
      daddi r2,r0,Ytab  ; Dirección de Y
      mtc1 r0,f0        ; Inicializar f0 y f1 a cero (r0 siempre es = 0)
      mtc1 r0,f1
Loop:
      1     l.d f2,0(r1)            ; load X[i]
      2     c.le.d f2,f0            ; ¿Es X[i] menor o igual que 0?
      3     bc1t L2                 ; Saltar a L2 si la bandera FP es TRUE
L1:
      4     div.d f4,f30,f2         ; 1/X[i]
      5     add.d f12,f0,f4         ; f12 = 1/X[i]
      6     mul.d f4,f2,f2          ; X[i]*X[i]
      7     div.d f6,f30,f4         ; 1/(X[i]*X[i])
      8     add.d f12,f12,f6        ; f12 = 1/X[i] + 1/(X[i]*X[i])
      9     s.d f12,0(r2)           ; Almacenar f12 en Y[i]
      10    j L3
L2:
      11    s.d f0,0(r2)            ; Almacenar f0 en Y[i]
L3:
      12    daddi r1,r1,8
      13    daddi r2,r2,8
      14    daddi r3,r3,-1
      15    bnez r3,Loop
            halt
```

Se consideran los siguientes parámetros para las unidades de coma flotante:

- 2 UF para las divisiones que tardan 5 ciclos.
- 1 UF para las multiplicaciones que tarda 3 ciclos.
- 1 UF para las sumas que tarda 2 ciclos.

[A mano]

1. Obtener el grafo de dependencias de las instrucciones del bucle. (0,5 Puntos)
2. Considerando que `X[i] > 0` (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). (1 Punto)
   b) Con interbloqueo hardware, *forwarding* y predicción de salto dinámica. (1 Punto)
   c) Con planificación dinámica usando el algoritmo de *Tomasulo* y predicción de salto dinámica. (1,5 Puntos.)

[Con WinMIPS64]

3. Determinar el CPI completo del programa con Winmips64. Comprobar que el programa funciona analizando el contenido de la memoria.Indicar los valores del vector Y al final de la ejecución del programa. (0,5 Puntos)
   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.
4. Desenrollar el bucle para que cada iteración corresponda con 2 del código actual. Comprobar que el programa sigue funcionando y dando el mismo resultado. (2 Puntos)
5. ¿Cuál es el nuevo CPI? Calcular el *speedup*. (0,5 Puntos)
6. Reordenar las instrucciones del bucle, eliminar las instrucciones inútiles y renombrar los registros para optimizar aun más el código. Comprobar que el programa sigue funcionando y dando el mismo resultado. ¿Cuál es el CPI final? (3 Puntos)

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