Universidad de Alcalá
DEPARTAMENTO DE AUTOMÁTICA
ARQUITECTURA Y TECNOLOGÍA DE COMPUTADORES

Laboratorio de Arquitectura e Ingeniería de Computadores
PRÁCTICA II

PLANIFICACIÓN ESTÁTICA DE INSTRUCCIONES

OBJETIVO

El objetivo de la práctica es profundizar en el estudio de las técnicas de segmentación de procesadores. Para ello, se pondrán en práctica las técnicas de aprovechamiento de la segmentación. En concreto, se van a utilizar la reordenación de código y el desenrollado de bucles.

INTRODUCCIÓN TEÓRICA.

En la práctica anterior se estudió el cauce segmentado del procesador DLX, y se pusieron en práctica algunas técnicas para aprovechar este cauce segmentado. En esta práctica se amplía sobre lo visto en la anterior (reordenación) y se estudia una nueva técnica llamada "desenrollado de bucles".

Reordenación de código.

La reordenación de código es una técnica que busca aprovechar las características de un cauce con adelantamiento de resultados, eliminando problemas causados por la estructura del cauce, como el caso de los parones tras la carga escalar del DLX. Como ejemplo se utilizará el siguiente código:

```
Bucle:    ld   f0,0(r1)
          add  f4,f0,f2
          sd   0(r1),f4
          subi r1,r1,8
          bnez r1,bucle
```

Este código suma a las componentes de un vector de números en coma flotante una constante que se encuentra en el registro f2. Si las suma en coma flotante necesita 3 ciclos para su ejecución

En el código visto se dará un parón de un ciclo entre la instrucción 1 y la 2, y otro de 2 ciclos entre la 2 y la 3 y otro parón más antes del salto. Además se utiliza un ciclo buscando la siguiente instrucción después del salto que será inútil si el salto se produce.

Se puede aprovechar más el tiempo reordenando el código (y modificando constantes para tener más flexibilidad), como se muestra a continuación:

```
Bucle:  ld    f0,0(r1)
                                    ;parón
        addd  f4,f0,f2
        subi  r1,r1,8
                                    ;parón
        sd    8(r1),f4
        bnez  r1,Bucle
                                    ;parón
```

con lo que se eliminan 2 de los 5 ciclos de parón en cada iteración.

Desenrollado de bucles.

El desenrollado de bucles consiste en solapar distintas iteraciones del mismo bucle, es decir, hacer una nueva versión del bucle e incluir las operaciones correspondientes a varias iteraciones de la versión original en una sola de la versión modificada. En el desenrollado hay que tener cuidado con las dependencias de datos entre iteraciones. El bucle anterior desenrollado en factor de 4 quedaría:

```
Bucle:  ld   f0,0(r1)       ;componente 1
        add  f4,f0,f2
        sd   0(r1),f4
        ld   f6,-8(r1)      ;componente 2
        add  f8,f6,f2
        sd   -8(r1),f8
        ld   f10,-16(r1)    ;componente 3
        add  f12,f10,f2
        sd   -16(r1),f12
        ld   f14,-24(r1)    ;componente 4
        add  f16,f14,f2
        sd   -24(r1),f16
        subi r1,r1,32       ;control del bucle
        bnez r1,Bucle
```

El desenrollado permite eliminar el recargo que supone el control del bucle en 3 de cada 4 iteraciones. Obsérvese cómo afecta el desenrollado a los desplazamientos en los accesos a memoria.

Por último, es posible aplicar ambas técnicas conjuntamente, y reordenar el código desenrollado, consiguiendo una nueva mejora en la ejecución.

Una de las dificultades del desenrollado de bucles es conseguir que se ejecuten todas las iteraciones del bucle original. La solución más fácil es utilizar un factor de desenrollado divisor del número total de iteraciones, pero esto no siempre es posible. Por ejemplo, si el bucle original tiene 11 iteraciones, el único factor de desenrollado posible es 11, ya que es primo. En estos casos se puede utilizar otro factor de desenrollado, teniendo en cuenta que las iteraciones que faltan se deben ejecutar. Por ejemplo, se puede descomponer un bucle de 11 iteraciones en 3 "iteraciones sueltas" y un bucle con 2 iteraciones que engloban a 4 de las originales. A ambas partes ("iteraciones sueltas" y el bucle desenrollado) se les puede aplicar reordenamiento por separado.

Esta última opción se puede aplicar también a bucles con cualquier número de iteraciones. En general, si se tiene un bucle de *n* iteraciones se le puede aplicar un factor de desenrollado *m*, descomponiéndolo en dos bucles de la siguiente manera:

- Un bucle, con cuerpo igual al original, que se ejecuta (`n mod m`) veces.

- Otro bucle, con cuerpo desenrollado `m` veces, que se ejecuta `(n / m)` veces.

El desenrollado de bucles es una técnica muy eficaz, y por ello la utilizan muchos compiladores optimizadores. Sin embargo puede ser difícil de aplicar en el caso de bucles con dependencias complejas entre iteraciones.

DESARROLLO DE LA PRÁCTICA

1.- Aplicar desenrollado y reordenación a un bucle que calcula el producto escalar de dos vectores de 64 componentes en doble precisión. Estudiar qué factor de desenrollado es el máximo posible, teniendo en cuenta los recursos de la arquitectura DLX (concretamente, el número de registros disponibles).

2.- Aplicar desenrollado y reordenación a un bucle que analiza un vector A de 77 componentes y pone un 1 en cada componente de otro vector B si la componente correspondiente de A tiene valor 0 y un 0 en cualquier otro caso. Analizar el desenrollado óptimo del bucle.

3.- Aplicar desenrollado y reordenación a un bucle que calcula la suma de las componentes de un vector A de 100 componentes.