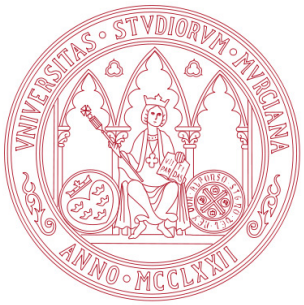

Universidad de Murcia

Facultad de Informática

TÍTULO DE GRADO EN
INGENIERÍA INFORMÁTICA

Arquitectura y Organización de Computadores

Práctica 2: Parámetros de diseño de un procesador superescalar

(todos los grupos)

CURSO 2018 / 2019

Departamento de Ingeniería y Tecnología de Computadores

Área de Arquitectura y Tecnología de Computadores

Boletines de prácticas

B2.1. Objetivos

El objetivo general de esta práctica es el de mostrar al alumno la influencia que diversos aspectos de diseño de un procesador superescalar tienen sobre el rendimiento del mismo. Para ello, se va a utilizar como programa de prueba el algoritmo de resolución de una ecuación en derivadas parciales por el método de las diferencias finitas resultante de la práctica anterior. Dicho programa será ejecutado sobre un simulador (RSIM) que modelará la arquitectura de un procesador superescalar y la jerarquía de memoria. A través de los parámetros de configuración del simulador se analizará el impacto que sobre el rendimiento tienen varios aspectos de diseño del procesador superescalar. Más concretamente, los objetivos de esta práctica son:

- Mostrar las Instrucciones Procesadas por Ciclo (IPC) como una métrica de rendimiento útil a la hora de evaluar la eficiencia de un núcleo superescalar.
- Que el alumno entienda la influencia que sobre el rendimiento tienen los parámetros de diseño típicos de un procesador superescalar, como tamaño del buffer de reordenación, grado de emisión, esquema de predicción de saltos, ejecución especulativa, etc.
- Comprender la influencia de la jerarquía de memoria en el rendimiento de estos procesadores y cuáles son sus parámetros característicos más influyentes.
- Estudiar las causas que limitan el rendimiento de una arquitectura superescalar, identificando los posibles cuellos de botella.
- Comprender qué optimizaciones hardware son las que tienen una mayor incidencia en la ejecución del algoritmo propuesto.

B2.2. Descripción

Para esta práctica se partirá del código del algoritmo de resolución de una ecuación en derivadas parciales por el método de las diferencias finitas resultante de la práctica anterior. Tan sólo cambiaremos el tamaño del problema, de cara a tener unos tiempos de simulación que no sean excesivamente elevados. Para ello, tomaremos una matriz de 60×60 elementos, y cambiaremos el valor de *TOL* a 0.1.

Para la simulación de este algoritmo utilizaremos el simulador RSIM de la Universidad de Rice, que es de libre distribución. Dicho simulador permite tanto la simulación de arquitecturas monoprocesador como arquitecturas multiprocesador de memoria compartida (aunque físicamente distribuida). En esta práctica, vamos a utilizar el simulador para evaluar el rendimiento de un núcleo de ejecución (procesador) superescalar. La configuración del procesador es la siguiente:

- Procesador superescalar de 2 vías homogéneo (2 instrucciones por ciclo para todas las etapas).
- Tamaño del buffer de reordenación (*Active list*): 16
- Esquema de predicción de saltos: static
- Unidades funcionales:
  - Enteras (*ALU*): Una unidad y las latencias que trae el simulador por defecto.
  - Punto flotante (*PFU*): Una unidad y las latencias que trae el simulador por defecto.

  - Generación de direcciones (*Addr. gen.*): Una unidad y las latencias que trae el simulador por defecto.

- Jerarquía de memoria:
  - L1 datos: 4 KiB, asociativa por conjuntos de 2 vías, con postescritura, y con un tiempo de acierto de 2 ciclos. No se modela la caché de instrucciones.
  - L2 unificada (datos e instrucciones): 16 KiB, asociativa por conjuntos de 2 vías, con postescritura, y con un tiempo de acierto de 11 ciclos (9 para comprobar las etiquetas + 2 para el acceso a los datos).
  - En todos los casos, el tamaño de bloque es de 32 bytes.
  - Latencia de memoria principal: 300 ciclos.
  - El procesador implementa las operaciones avanzadas de memoria de prebúsqueda controlada por hardware (opción -p) y ejecución de cargas especulativas (opción -K).

Nota: Para cualquier parámetro que no haya sido especificado se toma el valor que da el simulador por defecto.

Para facilitar la realización de la práctica, se incluye un script «`ejecuta`» que muestra cómo lanzar una simulación y guarda los resultados en un nuevo subdirectorio del directorio `outputs`. Este script se encarga de generar un fichero «`rsim.config`» y pasar el resto de opciones de configuración a través de opciones. Durante la realización de la práctica, se deberá modificar este script para fijar los parámetros necesarios para el experimento que se desee realizar (se recomienda copiar estos ficheros y conservar las versiones de los ficheros utilizados en cada ejercicio). Por ejemplo, el siguiente comando en el directorio `execs`:

```
./ejecuta
```

Realiza una simulación de jacobi para una matriz de 60×60 y guarda el resultado en un subdirectorio de `outputs` usando la hora actual como nombre. El nombre del directorio de salida se puede modificar usando la variable de entorno «`TEST_ID`». Por ejemplo:

```
TEST_ID=test1 ./ejecuta
```

B2.3. Bloque I. Identificación de los cuellos de botella de la configuración actual

En este primer bloque vamos a analizar las estadísticas detalladas que el simulador ofrece sobre la ejecución de la aplicación, tratando de encontrar los cuellos de botella que la configuración actual del núcleo de ejecución presenta. Para ello, nos vamos a concentrar en las estadísticas que el simulador ofrece de la fase 1 del programa (que es la que incluye todo el cálculo). Concretamente, se pide:

1. Calcular el IPC obtenido de la ejecución de la aplicación. ¿Cuál sería el IPC sin detenciones (ideal) para la configuración actual del procesador?

   Nota: A la hora de calcular el IPC, recuerda que debes tener en cuenta sólo las instrucciones correctamente especuladas del programa (es decir, sólo las que se llegan a graduar).

2. Definimos la *Tasa de Fallos de Especulación* (TFE) del procesador como el resultado de dividir el número de instrucciones emitidas (*Issued*) menos el número de instrucciones confirmadas (*Graduated*), entre el número de instrucciones emitidas (*Issued*). Es decir $TFE = \frac{Issued - Graduated}{Issued}$. Calcular el TFE del procesador. ¿Por qué el número de instrucciones emitidas y confirmadas no coincide?

3. Para cada uno de los siguientes elementos del sistema indica a la vista de las estadísticas cómo se está utilizando (grado de utilización) y si piensas que podría constituir un cuello de botella:

   - Buffer de reordenación.
   - Cola de memoria.
   - Esquema de predicción de saltos.
   - Unidades funcionales.
   - Cache L1.
   - Cache L2.

B2.4. Bloque II. Influencia de la jerarquía de memoria del procesador

En este apartado vamos a estudiar el impacto que la organización de la jerarquía de memoria tiene en el rendimiento. A la hora de realizar cada ejercicio, se debe partir de la mejor configuración obtenida en el ejercicio anterior. Concretamente se pide:

4. La cache L1 se podría hacer de escritura directa en lugar de postescritura. Explica la diferencia y comprueba qué efecto tiene esta decisión en el rendimiento.

5. Estudia la influencia de los parámetros de diseño de la cache L2. Supongamos que podemos aumentar el tamaño de la L2 sin afectar al tiempo de acierto. En concreto, probaremos con cachés de 16 KiB, 32 KiB y 64 KiB. Explica los resultados obtenidos teniendo en cuenta los tamaños de las estructuras de datos de la aplicación.

6. Continuando con la caché L1, probaremos con tamaños de 4 KiB, 8 KiB y 16 KiB, pero teniendo en cuenta que el tiempo de acierto aumenta en 1 ciclo en el caso de 16 KiB. Hay que tener en cuenta que el tamaño de la L1 debe ser siempre menor que el de la L2. Explica los resultados obtenidos considerando los tamaños de las estructuras de datos que la aplicación va a manejar.

B2.5. Bloque III. Influencia de los parámetros de diseño del procesador

En este segundo bloque, vamos a estudiar la influencia que tienen los principales parámetros de diseño de la microarquitectura en el rendimiento del procesador. A la hora de realizar cada ejercicio, se debe partir de la mejor configuración obtenida en el ejercicio anterior. Concretamente, se pide la evaluación por separado de los siguiente parámetros, calculando el IPC y el TFE obtenidos en cada caso:

7. Analiza la influencia del tamaño del buffer de reordenación (*Active List*). Los tamaños que vamos a probar para este recurso son: 16, 32 y 64 entradas. ¿Qué conclusiones obtienes? ¿Crees que podría mejorarse algo más si utilizáramos un tamaño aún mayor? Explica la respuesta.

8. Estudia la influencia del número de instrucciones emitidas por ciclo. Puede tomar los valores 2, 4 y 8. Explica los resultados obtenidos.

9. Estudia la influencia del número de unidades funcionales. Concretamente, y por separado, analiza cómo varía el comportamiento del programa al doblar el número de ALUs, el número de FPUs y el número de unidades de generación de direcciones. Explica los resultados obtenidos.

10. Evalúa el predictor de saltos. Tomando la mejor configuración del apartado anterior, analiza cómo influye en el rendimiento el cambiar el esquema de predicción a *2bitagree* con 64 contadores (parámetro de configuración *bpbsize*). Explica los resultados obtenidos.

B2.6. Bloque IV. Buscar posibles optimizaciones hardware para mejorar el rendimiento del programa

En este último apartado, vamos a estudiar qué optimizaciones hardware podríamos aplicar al procesador para que nos diera el máximo rendimiento para la aplicación que estamos utilizando como ejemplo.

11. Dada la siguiente relación de costes por cambio respecto a la configuración inicial, calcula el coste que tendría cada configuración propuesta en los bloques II y III:

    - Aumentar el tamaño de la L1: 3000 € al pasar de 4 KiB a 8 KiB y 6000 € al pasar de 8 KiB a 16 KiB.
    - Aumentar el tamaño de la L2: 3000 € al pasar de 16 KiB a 32 KiB y 3000 € al pasar de 32 KiB a 64KiB.
    - Cambiar la política de escritura directa por la de postescritura en la L1 tiene un coste de 1000 €.
    - Aumentar el grado de emisión del procesador: 1000 € al pasar de 2 a 4 y 3000 € al pasar de 4 a 8.
    - Mejorar la predicción de saltos: usar el *2bitagree* con 64 entradas cuesta 1000 €.
    - Aumentar el tamaño del buffer de reordenación: 2000 € al pasar de 16 a 32, 2000 € al pasar de 32 a 64.
    - Añadir una unidad funcional: 1000 € (independientemente del tipo de la unidad funcional).

12. Supón que disponemos de un presupuesto limitado consistente en 12 000 euros. Encuentra la configuración que, respetando dicho presupuesto, consigue una mayor eficiencia, definida como el resultado de dividir el IPC de la configuración propuesta entre el obtenido por la mejor configuración de los ejercicios anteriores. Por último, ¿en qué factor la configuración elegida acelera la ejecución del programa con respecto a la configuración inicial? A la vista de los resultados, ¿qué conclusiones obtienes?