Jiuxe
www.wuolah.com/student/Jiuxe

3428

Ejercicio_10_tema1_resuelto_alternativa.pdf

Ejercicios Resueltos

2° Arquitectura de Computadores

Grado en Ingeniería Informática

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
UGR - Universidad de Granada

Ejercicio 10. Un compilador ha generado un código máquina optimizado para el siguiente programa

```
for (i=0;i<N;i++)
  if ((i%2) = = 0)
      par=par+c*x[i];
  else
      impar=impar-c*x[i];
```

sin utilizar instrucciones de salto dentro de las iteraciones del bucle(es decir, ha **desenrollado** el bucle) . El código tiene **7 instrucciones fuera del bucle y 9 instrucciones dentro**

El computador donde se ejecuta dispone de un procesador superescalar de 32 bits a 2 GHz capaz de terminar dos instrucciones de coma flotante por ciclo, con dos caches internas (una para datos y otra para instrucciones) de 512 KBytes cada una, mapeo directo, política de actualización de postescritura, líneas de 32 bytes, y latencia de un ciclo de reloj. La memoria principal del computador tiene una latencia de 30 ns. y ciclos burst 6-1-1-1 a través de un bus de memoria de 200 Mhz. (a) ¿Cuál es la velocidad pico del procesador?. (b) ¿Cuál es el tiempo mínimo que tarda en ejecutarse el programa para $N=2^{11}$. (c) ¿Cuántos MFLOPS alacanza en el programa?.

(NOTA: Considere la situación más favorable para los datos en memoria y cache (compatible con las características de la máquina y el programa; par, impar, c, y x[] son números de 32 bits en coma flotante).

Solución: Puesto que el programa en ensamblador tiene 7 instrucciones fuera del bucle y 9 dentro (el resultado queda en un registro) y se ha desenrollado el bucle, el número de iteraciones es

Iteraciones = $N / 2 = 2^{11}/2=2^{10}$

y el número de instrucciones ejecutadas:

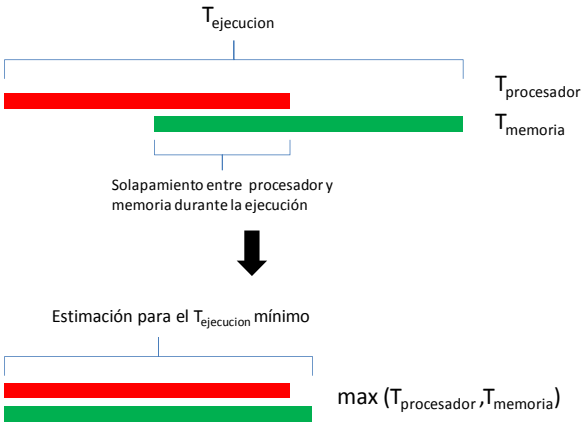
NInstrucciones $= 9*2^{10}+7=9223$

Así:

(a) Velocidad pico del Procesador (nos dicen en el enunciado que puede terminar **2 instrucciones de coma flotante por ciclo**):

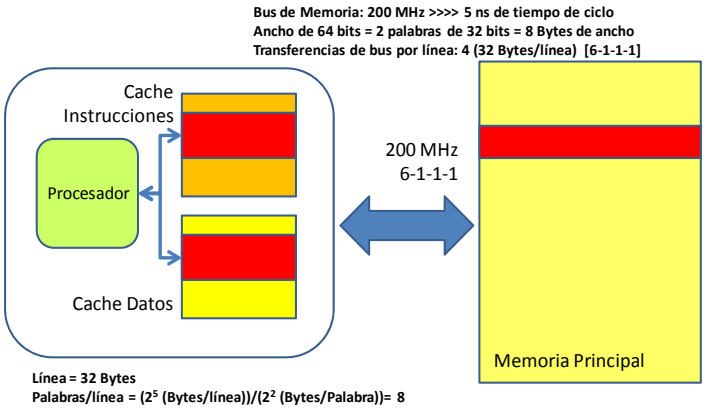
$$\mathbf{V_{pico}FP=2\ instFP/ciclo * 2.10^9\ ciclos/sg = 4.10^9\ instFP/sg = 4\ GFLOPS}$$

(b) Tiempo mínimo de procesamiento: $\mathbf{Tmin = max\ (T_{procesador}, T_{memoria})}$



$T_{procesador}$=NInstrucciones * CPI * Tciclo = 9223*(1/2)*0.5 (ns) = 9223*0.25=2305.75 ns

Tiempo de acceso a memoria:



Los accesos a memoria para cargar el código se desprecian porque el código ocupa sólo dos líneas y se generarían, por tanto sólo dos faltas de cache (7*4 bytes/inst+9*4 bytes/inst.= 28 + 36 = 64 bytes si se divide por 32 bytes/línea nos da 64 Bytes/32(Bytes/lines) = 2).

En el caso de acceso a los datos, se producirán más fallos. De hecho, se producirán tantos fallos como líneas ocupe el array x[] almacenado en memoria, ya que una vez que se cargue una línea, se accede a todos los datos de esa línea de forma consecutiva (suponemos que los accesos al dato c, que estaría en una línea aparte, darían lugar a un fallo de acceso, y sería despreciable frente al número de fallos para acceder a x[], lo mismo que hemos hecho con los fallos para el acceso a la cache de instrucciones)

.

Accesos a memoria (32 bits = 4 Bytes/dato): **$2 * 2^{10} = 2^{11}$** (coincide con el número de datos del array)

Fallos de cache = Número de líneas del array

Fallos= (4 bytes/dato * 2^{11} datos del array) / (2^5 bytes/línea) = 2^8 líneas

Tasa de fallos = (1 – a) = 2^8 / 2^{11} = 1/8=0.125 a= 7/8=0.875

Algunos datos: **Tciclo de Bus de memoria 200 MHz >>>> 1/(200*10^6) = 1000/200 * 10^{-9} = 5 ns.**

6-1-1-1 >>>>> 9 * 5 ns =45 ns

Tiempo de ciclo del procesador = 0.5 ns

Tiempo medio de acceso a un dato

$t_{memoria}$ = 0.875*0.5 + 0.125*(45+0.5)=6.125 ns

$T_{memoria}$ = 2^{11} * 6.125 = 12544 ns

Tiempo mínimo de procesamiento = max ($T_{procesador}$,$T_{memoria}$)=(2305.5, 12544) = 12544 ns

(c) MFLOPS = (2 (operaciones fp por elemento) * 2^{11} elementos) / ((12544.10^{-9})*10^6) = 326.54 (un 8% de diferencia con respecto al cálculo exhaustivo que se hace en el documento de problemas resueltos del Tema 2)

Se puede afinar algo más la estimación si se tiene en cuenta que, en el caso de que no haya faltas de cache, el tiempo de acceso a memoria está solapado con el de procesamiento. Según esto, sólo tendríamos que tener en cuenta en el cálculo del tiempo de acceso a memoria los accesos cuando hay faltas de cache:

$T_{memoria}$ = 2^8 * 45 ns = 11520 ns.

En este caso

MFLOPS = (2 (operaciones fp por elemento) * 2^{11} elementos) / ((11520.10^{-9})*10^6) = 355.56 (un 0.4% de diferencia con respecto al cálculo exhaustivo que se hace en el documento de problemas resueltos del Tema 2)

La diferencia entre el cálculo exhaustivo y la estimación que hemos realizado es de 4.5 ns, es decir, 9 ciclos del procesador en los que no se produciría el solapamiento entre el procesador y el acceso a memoria. No hay que olvidarse que lo que estamos haciendo aquí es una estimación aproximada que también podríamos hacer en casos más complicados (donde tener una estimación exhaustiva solo sería posible si se utiliza algún tipo de simulador suficientemente preciso del sistema).

Si no se despreciasen los fallos de cache correspondientes al acceso a las instrucciones y al dato c tendríamos que

$T_{memoria}$ = (2^8 + 3)* 45 ns = 11655 ns.

◄