Arquitectura e Ingeniería de Computadores - 3º Grado Ing. Informática - ETSINF - UPV

Ejercicios y soluciones de la Unidad Temática 1. "Introducción a la Arquitectura de Computadores"

Ejercicio 1.1.

En la fase de diseño de un computador, hay que optar entre dos opciones A y B, que afectan a la frecuencia de reloj y al número de instrucciones de tipo *l*oad/store necesarias. Sabiendo que:

- El reloj de la opción A es 5 % más rápido que el de la B.
- Un programa compilado para A tiene un 30 % de instrucciones del tipo *load/store*, mientras que el mismo programa compilado para B permite reducir en 1/3 el número de instrucciones de este tipo.
- CPI es 1 para todas las instrucciones

decídase qué diseño es mejor.

Solución:

El objeto bajo estudio es un computador que ejecuta un programa, y los datos del enunciado nos remiten a la expresión analítica del tiempo de ejecución de un programa.

La relación entre los tiempos de ejecución de las dos alternativas es:

$$\frac{t_A}{t_B} = \frac{I_A \cdot CPI_A \cdot T_A}{I_B \cdot CPI_B \cdot T_B}$$

Solamente quedan por relacionar los parámetros de la ecuación:

$$I_B = I_A - \frac{1}{3} \cdot \frac{30}{100} \cdot I_A = \frac{90}{100} I_A$$

$$CPI_B = CPI_A = 1$$

$$\frac{T_B}{T_A} = \frac{f_A}{f_B} = \frac{105}{100}$$

y queda:

$$\frac{t_A}{t_B} = \frac{100 \cdot 100}{90 \cdot 105} = 1{,}058$$

Conclusión: La máquina B es 5.8 % más rápida que la A.

□

Ejercicio 1.2.

En un computador *Load/Store*, se ha medido la frecuencia de aparición y los CPI de las siguientes categorias de instrucciones:

Operación	%	CPI
ALU	43	1
LOAD	21	2
STORE	12	2
Saltos	24	2

También se ha observado que el 25 % de las operaciones aritméticas tienen un operando que no se reutiliza, por lo que se plantea la conveniencia de añadir instrucciones aritméticas que tengan uno de sus operandos fuente en la memoria, de manera que secuencias de instrucciones como

```
ld r1,o(r10)
dadd r3,r1,r2
```

pueden ser sustituidas por:

```
daddm r3,o(r10),r2
```

Las nuevas instrucciones tienen un CPI = 2 y al emplearlas también se aumenta en uno el CPI de las instrucciones de salto. Considera que no hay ninguna variación en el coste del computador y justifica cuantitativamente la respuesta.

Solución:

Se trata de evaluar el impacto de una modificación del juego de instrucciones sobre el tiempo de ejecución de un programa tipo. La modificación afecta al número de instrucciones del programa y al CPI. Si denominamos a al procesador original y b al modificado, podremos escribir los tiempos de ejecución de uno y otro en función del número de instrucciones I_a de la versión original del programa y el tiempo de ciclo del reloj t_c.

Para el procesador original a basta con calcular el valor medio de CPI_a a partir de los datos tabulados en el enunciado:

$$\begin{aligned} T_a &= I_a \cdot CPI_a \cdot t_c \\ &= I_a \cdot (0{,}43 \cdot 1 + 0{,}21 \cdot 2 + 0{,}12 \cdot 2 + 0{,}24 \cdot 2) \cdot t_c \\ &= 1{,}57 \cdot I_a \cdot t_c \end{aligned}$$

Para el procesador b hay que calcular el número de instrucciones I_b y el CPI_b. En el número de instrucciones hay que descontar el 25 % de las instrucciones de cálculo:

$$I_b = (1 - (0{,}25 \cdot 0{,}43))I_a = (1 - 0{,}1075) \cdot I_a = 0{,}8925 I_a$$

La tabla siguiente muestra cómo queda el número de instrucciones de cada tipo y sus CPI:

Operación		CPI
ALUnueva	$0{,}25 \cdot 43 = 10{,}75$	2
ALU	$43 - 0{,}25 \cdot 43 = 32{,}25$	1
LOAD	$21 - 0{,}25 \cdot 43 = 10{,}25$	2
STORE	12	2
Saltos	24	3
Total	89.25	

Como la suma de instrucciones no es 100, en el cálculo del índice CPI hemos de normalizar las frecuencias de cada tipo de instrucción:

$$CPI_b = \frac{0{,}1075 \cdot 2 + 0{,}3225 \cdot 1 + 0{,}1025 \cdot 2 + 0{,}12 \cdot 2 + 0{,}24 \cdot 3}{0{,}8925} = \frac{1{,}7025}{0{,}8925} = 1{,}91$$

El tiempo de ejecución resultante es:

$$T_b = I_b \cdot CPI_b \cdot t_c = 0{,}8925 I_a \cdot 1{,}91 \cdot t_c = 1{,}703 \cdot I_a \cdot t_c$$

La modificación no es interesante, porque la aceleración obtenida es menor que la unidad:

$$S = \frac{1{,}57 \cdot I_a \cdot T_a}{1{,}703 \cdot I_a \cdot T_a} < 1$$

□

Ejercicio 1.3.

En un computador con el procesador *MIPS R2000* a 100 MHz y coprocesador se compila y ejecuta el programa *P* que realiza dos millones de operaciones de coma flotante. El compilador traduce cada operación de coma flotante en una instrucción del coprocesador o en una rutina con instrucciones de enteros, según las opciones de compilación indicadas. Variando estas opciones, se ha compilado *P* dos veces, generando dos códigos ejecutables: P_h (código para *MIPS R2000* con coprocesador) y P_s (código para *MIPS R2000* sin coprocesador). Se han ejecutado los dos programas y se han obtenido las siguientes medidas:

Programa	Tiempo de ejecución	CPI medio medido
P_h	92 milisegundos	3.1 ciclos
P_s	1.2 segundos	1.2 ciclos

Se pide:

1. El número de instrucciones que se ejecutan por unidad de tiempo en ambas versiones del programa *P*, expresadas en *MIPS* y el número total de instrucciones ejecutadas en cada caso.
2. El número medio de instrucciones de enteros que sustituye cada operación en coma flotante en P_s.
3. La productividad de ambos programas expresadas en *MFLOPS*.

Solución:

1. Para el cálculo de la productividad en MIPS, podemos insertar la ecuación del tiempo de ejecución del programa en la definición:

$$\text{Prod(MIPS)} = \frac{I}{T_{ej} \cdot 10^6} = \frac{I}{I \cdot CPI \cdot T \cdot 10^6} = \frac{1}{CPI \cdot T \cdot 10^6}$$

A partir de esta expresión, como conocemos los CPIs de ambos programas y el periodo de reloj ($\frac{1}{f} = \frac{1}{100 \cdot 10^6} = 10$ns), podemos calcular los MIPS. Por otra parte, como también conocemos el tiempo T_{ej} de ejecución global en ambos casos, se puede obtener el número total de instrucciones como:

$$I = \text{Prod(MIPS)} \cdot T_{ej} \cdot 10^6$$

Sustituyendo se obtiene:

Programa	Productividad (MIPS)	número de instrucciones
P_h	32.26	2.97 millones
P_s	83.33	100 millones

2. En P_s todas las instrucciones son de enteros, pero unas son para realizar las operaciones de coma flotante y el resto para otros tipos de datos y otras necesidades del programa. Estas últimas también se encuentran entre el total de instrucciones de P_h, y podemos obtener su número: 2.97 millones de instrucciones en total $-$ 2 millones de instrucciones de coma flotante = 0.97 millones de instrucciones enteras comunes a ambos programas. Con este razonamiento, obtenemos el número de instrucciones de P_s que realizan los cálculos de coma flotante: 100 millones $-$ 0.97 millones = 99.03 millones.

   Resumiendo, los 2 millones de operaciones de coma flotante se traducen en 99.03 millones de instrucciones enteras. La relación pedida es:

$$\frac{99{,}03 \text{ millones}}{2 \text{ millones}} = 49{,}65 \text{ instrucciones}$$

3. Para este cálculo basta con aplicar la definición de MFLOPS. En ambos casos se realiza el mismo número de operaciones, pero en distintos tiempos de ejecución. El resultado es:

Programa	Productividad (MFLOPS)
P_h	21.74
P_s	1.67

   Estas medidas de productividad en *MFLOPS* son coherentes con las prestaciones reales de estos programas. El programa P_h es 13 veces más rápido que P_s tanto si atendemos al tiempo de ejecución total como a la relación entre las productividades en *MFLOPS*. Sin embargo, la relación entre las productividades medidas en *MIPS* sugieren, incorrectamente, que el programa más rápido es P_s.

□

Ejercicio 1.4.

Se ha escrito en lenguaje C un programa de compresión MP3 para comparar dos procesadores, uno antiguo A y otro más moderno B. El procesador A funciona a 200 MHz, sólo tiene instrucciones de enteros y su CPI es 1. El procesador B funciona a 900 MHz, incorpora las mismas instrucciones enteras y además instrucciones multimedia *IM* y su CPI medio depende de la aplicación, porque las instrucciones de enteros duran 1 ciclo y las instrucciones *IM* duran 3 ciclos. El código ejecutable generado por el compilador puede contener o no instrucciones *IM* según las opciones que se utilicen. Cada instrucción *IM* es equivalente a una serie de instrucciones de enteros que realizan la misma operación. Para las pruebas se preparan dos códigos ejecutables: *H* (con instrucciones *IM*) y *S* (sin ellas). Se anotan los siguientes tiempos de ejecución: el procesador A con el código *S* necesita 50 segundos para comprimir cierta canción mientras que al procesador B con el código *H* le basta con 8 segundos. Además, se ha visto que con el código *H* se ejecutan un 36 % menos de instrucciones que con el código *S*.

1. ¿Cuál es el valor medio del CPI del procesador B ejecutando el código *H*?
2. ¿Cuál es la proporción de instrucciones *M* en el programa *H*?
3. ¿Cuántas instrucciones enteras son equivalentes a una instrucción *IM* en promedio?
4. ¿Cuánto tiempo necesitará el procesador B para comprimir la misma canción con el programa *S*?

Solución:

1. El tiempo de ejecución del código H en el procesador B es

$$T_{H,B} = I_H \cdot CPI_{H,B} \cdot t_B,$$

donde

- El número de instrucciones ejecutadas I_H depende del código H.
- El CPI medio $CPI_{H,B}$ depende de la distribución de instrucciones en H y el CPI de cada instrucción en B.
- El periodo de reloj t_B depende del tipo de procesador (B).

A partir de esta ecuación se puede despejar $CPI_{H,B}$. Así,

$$T_{H,B} = I_H \cdot CPI_{H,B} \cdot t_B \rightarrow 8s = I_H \cdot CPI_{H,B} \cdot 1,11ns \rightarrow CPI_{H,B} = \frac{8s}{I_H \cdot 1,11ns}.$$

Por otro lado, sabemos que con el código H se ejecutan un 36 % menos instrucciones que con el código S. Por lo tanto,

$$I_H = 0{,}64 \cdot I_S \rightarrow CPI_{H,B} = \frac{8s}{0{,}64 \cdot I_S \cdot 1,11ns}.$$

.

I_S se puede obtener a partir de la ecuación del tiempo de ejecución del código S en el procesador A. Así,

$$T_{S,A} = I_S \cdot CPI_{S,A} \cdot t_A \rightarrow 50s = I_S \cdot 1 \cdot 5ns \rightarrow I_S = \frac{50s}{5ns} = 10 \cdot 10^9.$$

Por lo tanto,

$$CPI_{H,B} = \frac{8s}{0{,}64 \cdot 10 \cdot 10^9 \cdot 1,11ns} = 1{,}125.$$

Alternativamente, podemos expresar la aceleración en términos de la ecuación del tiempo de ejecución de los programas. Así,

$$S = \frac{50}{8} = 6{,}25 = \frac{I_S \cdot CPI_{S,A} \cdot t_A}{I_H \cdot CPI_{H,B} \cdot t_B} = \frac{I_S \cdot 1 \cdot 5ns}{0{,}64 \cdot I_S \cdot CPI_{H,B} \cdot 1{,}11ns}.$$

Despejando, obtenemos que $CPI_{H,B} = 1{,}125$.

2. El valor medio del CPI recién calculado corresponde a una mezcla de instrucciones. Existe una proporción x de instrucciones *IM* que consumen tres ciclos y el resto, $(1 - x)$, son las instrucciones enteras que consumen un ciclo. Si despejamos x en la fórmula

$$CPI_{H,B} = 1{,}125 = (1 - x) \cdot 1 + x \cdot 3$$

tendremos que la proporción de instrucciones *IM* es de 6.25 %

3. Por cada 100 instrucciones en S hay 64 instrucciones en H, de las cuales el 6.25 % son *IM* (es decir, $0{,}625 \cdot 64 = 4$ instrucciones). En consecuencia, hay 60 instrucciones comunes a ambos códigos. Por tanto, en *S* hay 40 (100 en total - 60 comunes) instrucciones enteras que serán sustituidas por 4 instrucciones *IM* en H. Luego una instrucción *IM* sustituye 10 instrucciones de enteros.

4. El tiempo de ejecución del código S en el procesador B es

$$T_{S,B} = I_S \cdot CPI_{S,B} \cdot t_B = 10 \cdot 10^9 \cdot 1 \cdot 1{,}11ns = 11{,}1s.$$

Alternativamente, se puede razonar que, independientemente del procesador, el número de instrucciones ejecutadas con el código S es I_S; y que $CPI_{S,A} = CPI_{S,B}$ ya que S está compuesto exclusivamente por instrucciones de enteros, las cuales tienen un CPI igual a 1 en ambos procesadores. Por lo tanto, el cambio sólo afecta al reloj, así que, de la relación entre las frecuencias, resulta que $S = 900/200 = 4{,}5$ y el tiempo de ejecución queda en $t = 50/4{,}5 = 11{,}1$ segundos.

□

Ejercicio 1.5.

En cierta CPU, todas las instrucciones enteras se ejecutan en un ciclo de reloj, mientras que las de coma flotante necesitan 5 ciclos de reloj para completarse. La mayoría de los programas a ejecutar incluyen un 20 % de operaciones en coma flotante. Desde el punto de vista del análisis de costes y prestaciones, ¿es interesante rediseñar la parte de coma flotante del procesador para que sea 5 veces más rápida a costa de duplicar el coste total de la CPU? Justifica la respuesta.

Solución:

Se trata de comparar un diseño inicial, que llamaremos A, con un diseño alternativo, que llamaremos B. El proceso bajo medida es el computador ejecutando un programa tipo, y la unidad de tiempo que se puede utilizar es el ciclo de reloj, que coincide en ambos diseños.

Buscaremos la solución por dos caminos: mediante la ecuación del tiempo de ejecución de los programas y utilizando la ley de Amdahl.

- Mediante la ecuación del tiempo de ejecución

  Como los factores I y T no cambian entre las dos alternativas, sólo falta calcular los CPI de cada una de ellas:

$$CPI_A = 0{,}2 \cdot 5 + 0{,}8 \cdot 1 = 1{,}8$$

$$CPI_B = 0{,}2 \cdot 1 + 0{,}8 \cdot 1 = 1$$

  Relacionando los tiempos de ejecución,

$$S = \frac{T_A}{T_B} = \frac{I \cdot 1{,}8 \cdot T}{I \cdot 1 \cdot T} = 1{,}8$$

  El nuevo diseño de CPU es 1.8 veces más rápido, pero su coste se duplica. Por ello, desde un punto de vista de prestaciones y costes estricto, no es interesante aplicar la mejora propuesta.

- Aplicando la ley de Amdahl

  Hay que partir de la expresión general

$$S' = \frac{1}{(1-F) + \frac{F}{S}}$$

  donde S es la mejora de velocidad propuesta, y F es el porcentaje del tiempo en que se puede emplear la optimización propuesta. El enunciado suministra la mejora de velocidad, $S = 5$,

pero queda un factor por determinar. La fracción de tiempo no es la proporción de instrucciones (20 %) afectadas por el cambio, puesto que cada tipo de instrucción tiene una duración. La fracción de tiempo significativa es la que dedica la CPU A a operaciones en coma flotante respector del total. Si un programa consta de n instrucciones, el 20 % de coma flotante consumen $5 \cdot (0{,}2 \cdot n)$ ciclos, que hay que comparar con el tiempo total consumido de $1{,}8 \cdot n$ ciclos:

$$F = \frac{1}{1{,}8} = 0{,}55$$

Sustituyendo en la ley de Amdahl:

$$S' = \frac{T_A}{T_B} = \frac{1}{(1 - 0{,}55) + \frac{0{,}55}{5}} = 1{,}8$$

De nuevo, la opción *b* es 1.8 veces más rápida que la *a*.

□

Ejercicio 1.6.

El coprocesador de un computador mejora en un factor de 5 el procesamiento de números en coma flotante. El tiempo de ejecución de cierto programa es de 1 minuto con el coprocesador instalado, y de 2.5 minutos sin éste. Calcular el porcentaje del tiempo de ejecución que el programa realiza operaciones en coma flotante sin el coprocesador instalado.

Solución:

Aquí se aplica la Ley de Amdahl de forma inversa. Disponemos de dos aceleraciones: la referida a la parte modificada S y aceleración resultante S'. Despejando F en la ley de Amdahl, obtenemos esta expresión:

$$F = \frac{S' \cdot S - S}{S' \cdot S - S'}$$

Sustituyendo los datos:

$$F = \frac{2{,}5 \cdot 5 - 5}{2{,}5 \cdot 5 - 2{,}5} = 0{,}75 = 75\,\%$$

que es el porcentaje de tiempo buscado.

□

Ejercicio 1.7.

Ejercicio 1.8.

Un computador dispone de un procesador load/store con un único nivel L1 de memoria cache (interna). Las instrucciones de acceso a memoria son el 30 % del total de ejecutadas, y el 5 % de las mismas generan un fallo que requiere un acceso a la memoria principal durante 10 ciclos de reloj y que se añade al tiempo de ejecución de la instrucción en ausencia de fallo. En ausencia de fallos de cache, el CPI del procesador es 1.

Se estudia la conveniencia de añadir un segundo nivel L2 de cache (externa). Las pruebas realizadas han demostrado que el segundo nivel de cache resuelve el 90 % de fallos de primer nivel, reduciendo la penalización a sólo 2 ciclos de reloj; el 10 % restante sigue penalizando 10 ciclos de reloj.

1. ¿Cuál es la fracción de tiempo que consume el procesador en acceder a la memoria principal con un solo nivel de cache?

2. En promedio, ¿qué penalización sufrirá el procesador a cada fallo de caché L1 si existe la caché L2?

3. ¿Cuál es la aceleración total del computador al añadir L2?

4. ¿Cuál es la fracción de tiempo que consume el procesador en acceder a la memoria principal cuando hay dos niveles de cache?

5. Si el computador está valorado en 1000 € y se hace uso del análisis de prestaciones-coste, ¿cuál sería la máxima inversión que debería hacerse en añadir la cache L2?

Solución:

1. Si tomamos 100 instrucciones, 30 serán load/store y 1,5 producirán un fallo que suponen una penalización de 15 ciclos que se añaden a los 100 ciclos base. O sea, que de cada 115 ciclos 15 corresponderán a la penalización:

$$F = \frac{15}{115} = 13\,\%$$

2. De 100 fallos en L1, 90 se resolverán en 2 ciclos, y los 10 restantes en 10 ciclos. Luego la penalización promedio será:

$$P = \frac{90 \cdot 2 + 10 \cdot 10}{100} = 2{,}80\ ciclos$$

3. Aquí resulta aplicable la ley de Amdahl, tomando $F = 13\,\%$ y $S = \frac{10}{2{,}8} = 3{,}57$

$$S' = \frac{1}{1 - 0{,}13 + \frac{0{,}13}{3{,}57}} = 1{,}10$$

4. Cada 100 instrucciones requieren ahora $1{,}5 \cdot 2{,}8$ ciclos de penalización, o sea que requerirán 104.2 ciclos. De éstos, $1{,}5 \cdot 10\,\% \cdot 10$ ciclos corresponderán al acceso a la memoria. O sea:

$$F = \frac{1{,}5 \cdot 0{,}1 \cdot 10}{104{,}2} = 1,4\,\%$$

5. Si la aceleración es del 10 %, podremos invertir hasta 100 €

□

Ejercicio 1.9.

Se dispone de un programa P cuyo tiempo de ejecución en un computador C es de $120\ s$. Se pretende acelerar la ejecución de dicho programa realizando varias mejoras en la plataforma de ejecución:

1) Comprar una tarjeta controladora de discos RAID, que permitiría reducir el tiempo de acceso a disco a un 40 % del original, reduciendo el tiempo de ejecución total del programa P a $84\ s$. El coste de dicha tarjeta controladora es de 90 €.

2) Adquirir una tarjeta aceleradora de video, puesto que se sabe que el programa P dedica el 30 % del tiempo a procesos de representación gráfica, valorada en 125 €. La incorporación de dicha tarjeta aceleradora reduciría el tiempo dedicado a representación gráfica a $1/3$ del original.

Dadas estas características:

1. Si el computador original hubiera costado 300 €, ¿qué mejoras serían rentables por separado basándonos en el análisis de coste/prestaciones? Justifica la respuesta.

2. ¿Cuánto debería habernos costado el computador original C para que fuera interesante la incorporación simultánea de ambas mejoras desde el punto de vista de coste/prestaciones?. Justifica la respuesta.

Solución:

1. Para aplicar el análisis de coste y prestaciones necesitamos cuantificar el efecto global de cada mejora (S'_d para el disco y S'_v para el sistema de vídeo) y compararlo con el incremento de precio correspondiente.

   En el caso del disco, tenemos que

   $$S'_d = \frac{T_{\text{original}}}{T_{\text{nuevo}}} = \frac{120}{84} = 1{,}43$$

   Si el coste original fue $C_o = 300$ € y el coste con la mejora es $C_d = 390$ €, la relación es:

   $$\frac{C_d}{C_o} = \frac{390}{300} = 1{,}3$$

   Luego la mejora de la controladora de disco es rentable en este caso.

   Para calcular S'_v podemos aprovechar la ley de Amdahl, haciendo $F_v = 30\,\%$ y la aceleración $S_v = 3$.

   $$S'_v = \frac{1}{(1 - F_v + \frac{F_v}{S_v}} = \frac{1}{(1 - 0{,}3) + \frac{0{,}3}{3}} = 1{,}25$$

   Y la relación entre los costes:

   $$\frac{C_v}{C_o} = \frac{425}{300} = 1{,}42$$

   Luego, en este caso, la nueva adquisición es claramente desaconsejable desde el punto de vista del análisis de coste/prestaciones.

2. La mejora global, S'_{dv}, la podemos obtener a partir de la ley de Amdahl:

   $$S'_{dv} = \frac{1}{1 - F_d - F_v + \frac{F_d}{S_d} + \frac{F_v}{S_v}}$$

   donde falta por determinar S_d y F_d. Para obtener S_d

   $$S_d = \frac{1}{0{,}4} = 2{,}5$$

   y F_d se obtiene a partir de:

   $$S'_d = \frac{1}{(1 - F_d) + \frac{F_d}{S_d}} \;\Rightarrow\; 1{,}43 = \frac{1}{(1 - F_d) + \frac{F_d}{2{,}5}}$$

$$F_{\text{disco}} = \frac{0{,}75}{1{,}5} = 0{,}5$$

Asi pues:

$$S'_{dv} = \frac{1}{1 - F_d - F_v + \frac{F_d}{S_d} + \frac{F_v}{S_v}}$$

$$S'_{dv} = \frac{1}{1 - 0{,}5 - 0{,}3 + \frac{0{,}5}{2{,}5} + \frac{0{,}3}{3}} = \frac{1}{0{,}2 + 0{,}2 + 0{,}1} = \frac{1}{0{,}5} = 2$$

La mejora global cuesta $90 + 125 = 215$ €. Así pues, si el incremento de coste debe ser igual o inferior a la aceleración global para que sea rentable la incorporación de ambas mejoras, el coste límite del computador original debería ser:

$$\begin{aligned} C_{\text{mejorado}} &\leq S'_{\text{disco+video}} \cdot C_{\text{original}} \\ C_{\text{original}} + 215 &\leq 2 \cdot C_{\text{original}} \\ 1 \cdot C_{\text{original}} &\geq 215 \\ C_{\text{original}} &\geq 215\,€ \end{aligned}$$

□

Ejercicio 1.10.

Se ha monitorizado la ejecución de cierta aplicación en un computador, habiéndose obtenido un tiempo de ejecución de 10 minutos, así como que la mayor parte de su tiempo se consume en los procedimientos *P1* y *P2*. Con el objeto de reducir el tiempo global de ejecución se modifica el código de ambos procedimientos. Si la mejora obtenida en cada uno de los dos procedimientos *P1* y *P2* fuera infinita, el tiempo de ejecución de la aplicación descendería hasta 2 minutos. Finalmente, se consigue hacer un procedimiento *P1* que hace su función 5 veces más rápido y mejorar el *P2* en un 100 %, haciendo que el tiempo de ejecución de la aplicación sea de 4.5 minutos.

Calcula cuánto tiempo se consumía en los procedimientos *P1* y *P2* en la aplicación original.

Solución:

Sea $F1$ y $F2$ el tiempo consumido en cada uno de los procedimientos, y R el tiempo consumido en el resto del código. Sabemos que:

$F1 + F2 + R = 10$

Al mejorar los procedimientos $P1$ y $P2$, el programa consumirá un tiempo $F1'$ y $F2'$ en cada uno de ellos. Cuando la mejora en estos procedimientos fuera infinita ($F1' = 0$ y $F2' = 0$), el tiempo de ejecución es de 2 minutos:

$R = 2$

Por lo tanto:

$F1 + F2 + 2 = 10 \rightarrow F1 + F2 = 8$

Por otra parte, sabemos que al mejorar $P1$ y $P2$ en 5 y 2 veces, respectivamente, el tiempo de ejecución pasa a ser de 4.5 minutos:

$\frac{F1}{5} + \frac{F2}{2} + 2 = 4{,}5$

Tenemos un sistema de dos ecuaciones con dos incógnitas:

$$\left.\begin{array}{rcl} F1 + F2 & = & 8 \\ \frac{F1}{5} + \frac{F2}{2} & = & 2{,}5 \end{array}\right\}$$

Operando y despejando, obtenemos:

$F1 = 5$ min y $F2 = 3$ min

□

Ejercicio 1.11.

La empresa Farmax dispone de un un supercomputador HAL para trabajos de análisis químico. El ordenador está dotado de un procesador RIX/300 con un reloj a 300 MHz. El sistema se utiliza exclusivamente para ejecutar el programa de análisis *Espectroquimix* que hace uso intensivo de instrucciones de coma flotante. Después de monitorizar el sistema, se han obtenido los datos siguientes:

- La fracción de tiempo que el procesador emplea en ejecutar instrucciones de coma flotante es 75 %.
- La frecuencia de las instrucciones de coma flotante en *Espectroquimix* es: (ver tabla).
- Los CPI de cada tipo de instrucción de CF (ver tabla).

frecuencia (%)	operación	CPI
30	suma	5
10	resta	5
40	multiplicación	10
20	división	40

La dirección de Farmax pide al ingeniero en informática al mando que aumente la velocidad de cálculo. Después de consultar con los proveedores hay que valorar la siguiente mejora: Sustituir el procesador del computador por el procesador RIX/400E, compatible binario con el anterior, que funciona a 400 MHz y tiene la parte de coma flotante muy mejorada. Los CPI de las instrucciones enteras no cambian, pero los de coma flotante quedan así:

operación	CPI
suma	3
resta	3
multiplicación	7
división	25

Calcula:

1. La mejora obtenida ejecutando instrucciones enteras.
2. El CPI medio de las instrucciones de coma flotante de los procesadores RIX/300 y RIX/400E cuando ejecutan *Espectroquimix*.
3. La productividad en MFLOPS obtenida con el procesador RIX/300 cuando ejecuta *Espectroquimix*.

4. La aceleración obtenida en la ejecución de código en coma flotante por el cambio de procesador.
5. La aceleración global obtenida en la ejecución de *Espectroquimix*.
6. La fracción del tiempo que el procesador RIX/400E emplea en ejecutar instrucciones de coma flotante.
7. La productividad en MFLOPS obtenida con el procesador RIX/400E cuando ejecuta *Espectroquimix*.

Solución:

En este problema hay que realizar un análisis separado del rendimiento de dos procesadores ejecutando instrucciones enteras e instrucciones de coma flotante.

Denominaremos A al computador actual basado en el procesador RIX/300 y B al alternativo, basado en el RIX/400E.

1. Restringiendo el análisis a las instrucciones enteras, tenemos que el tiempo de ejecución de las instrucciones enteras de *Espectroquimix* será

$$T_{ent} = I_{ent} \cdot CPI_{ent} \cdot t_C$$

Como ambos procesadores son compatibles binarios, el número de instrucciones enteras no cambia al pasar de un procesador a otro, puesto que ejecutan el mismo programa. Como el CPI medio (desconocido) de las instrucciones enteras tampoco cambia, la aceleración se deberá exclusivamente al cambio de frecuencia de reloj. Por tanto,

$$S_{ent} = \frac{t_{C(A)}}{t_{C(B)}} = \frac{f_B}{f_A} = \frac{400}{300} = 1{,}333$$

El procesador RIX/400E es 1.33 veces más rápido que el RIX/300

2. Las frecuencias de las instrucciones no cambian, pero sí los CPI de cada una.

$$CPI_{cfA} = 0{,}30 \cdot 5 + 0{,}10 \cdot 5 + 0{,}40 \cdot 10 + 0{,}20 \cdot 40 = 14 \text{ ciclos}$$

$$CPI_{cfB} = 0{,}30 \cdot 3 + 0{,}10 \cdot 3 + 0{,}40 \cdot 7 + 0{,}20 \cdot 25 = 9 \text{ ciclos}$$

3. Productividad en MFLOPS.

$$Prod_{CF_A} = \frac{x(\text{op c.f.})}{T_{ej} \cdot 10^6}$$

Teniendo en cuenta que el número x de operaciones en coma flotante coincide con el número de instrucciones en coma flotante ejecutadas:

$$Prod_{CF_A} = \frac{x(\text{op c.f.})}{T_{ej} \cdot 10^6} = \frac{I_{cf}}{T_{ej} \cdot 10^6}$$

Por otra parte, sabemos que el cálculo en coma flotante ocupa una fracción $F_{cf} = 75\,\%$ de tiempo:

$$F_{cf} = 0{,}75 = \frac{T_{ejcf}}{T_{ej}} \rightarrow T_{ej} = \frac{T_{ejcf}}{F_{cf}} = \frac{T_{ejcf}}{0{,}75}$$

Y por otra parte, el tiempo que se dedica a la coma flotante es:

$$T_{ejcf} = I_{cf} \cdot CPI_{cf}(ciclos) = I_{cf} \cdot 14(ciclos) = I_{cf} \cdot 14 \cdot t_C$$

donde $t_C = \frac{1}{300 \cdot 10^6}$ ns
Sustituyendo:

$$Prod_{CF_A} = \frac{I_{cf}}{T_{ej} \cdot 10^6} = \frac{I_{cf}}{\frac{T_{ejcf}}{0{,}75} \cdot 10^6} = \frac{I_{cf}}{\frac{I_{cf} \cdot 14 \cdot t_C}{0{,}75} \cdot 10^6} = \frac{0{,}75}{14 \cdot \frac{1}{300 \cdot 10^6} \cdot 10^6} = 16{,}1 \text{ MFLOPS}$$

4. En el caso de la coma flotante, hay que considerar las dos modificaciones: la reducción del CPI_{cf} y el incremento de la frecuencia de reloj.

$$S_{cf} = \frac{CPI_{cfA}}{CPI_{cfB}} \cdot \frac{f_B}{f_A} = \frac{14}{9} \cdot \frac{400}{300} = 2{,}07$$

5. Aquí hay que considerar por separado los enteros y la coma flotante; para cada concepto hay una fracción de tiempo medida y una aceleración que hemos calculado. Aplicando la ley de Amdahl, obtenemos la aceleración global S_g:

$$S_g = \frac{1}{1 - F_{ent} - F_{cf} + \frac{F_{ent}}{S_{ent}} + \frac{F_{cf}}{S_{cf}}} = \frac{1}{\frac{0{,}25}{1{,}33} + \frac{0{,}75}{2{,}07}} = 1{,}82$$

6. La nueva fracción será

$$F' = F_{cf} \cdot \frac{S_g}{S_{cf}} = 0{,}75 \cdot \frac{1{,}82}{2{,}074} = 0{,}66$$

7. Procediendo de forma similar al apartado 3:

$$Prod_{CF_B} = \frac{0{,}66}{9 \cdot \frac{1}{400 \cdot 10^6} \cdot 10^6} = 29{,}3 \text{ MFLOPS}$$

□

Ejercicio 1.12.

Un servidor de Internet, dedicado a servir peticiones de red, incluye una placa base con dos procesadores con cuatro núcleos cada uno basados en la microarquitectura AMD Barcelona. Cada procesador dispone de dos controladores de memoria que distribuyen equitativamente el ancho de banda entre todos los núcleos. Las aplicaciones que se ejecutan en el servidor están compuestas de muchas tareas independientes que se ejecutan concurrentemente. Cada tarea pasa el 50 % de su tiempo de ejecución procesando instrucciones, y el 30 % realizando accesos a memoria que no pueden solaparse con la ejecución de instrucciones.

Despues de años de funcionamiento la compañia necesita actualizar el servidor para proveer de servicio a más clientes. Entre las opciones disponibles se consideran dos configuraciones: a) Una placa con dos procesadores Intel Nehalem EX o b) Una placa con dos procesadores AMD Magny-Cours. Cada procesador de Intel incluye 8 nucleos y 8 controladores de memoria, mientras que cada uno de los AMD dispone de 12 nucleos y 4 controladores. Asumiendo que cada uno de los nucleos Intel y AMD son igual de rápidos, que ambos tipos de núcleos son un 20 % más rápidos que un núcleo AMD Barcelona, y que cada controlador de los nuevos procesadores es exactamente igual de rápido que los del AMD Barcelona ¿Qué configuración obtendrá la máxima aceleración?

Solución:

La fracción del tiempo de ejecución global consumida por el procesador y la memoria es 50 % y 30 %, respectivamente.

Respecto al procesador, hay dos factores de aceleración. Por un lado, puesto que las aplicaciones son independientes, añadir más núcleos proporciona una aceleración lineal. Por otro lado, los nuevos núcleos aceleran la ejecución en un factor de 1.2 (son un 20 % más rápidos). Por lo tanto, la aceleración obtenida por los procesadores Intel y AMD es $S_{IntelCPU} = (1{,}2) * (8/4) = 2{,}4$ y $S_{AMDCPU} = (1{,}2) * (12/4) = 3{,}6$, respectivamente.

Respecto a la memoria, puesto que los controladores de memoria son idénticos, la aceleración se debe exclusivamente al mayor número de controladores de memoria. Por tanto, la aceleración obtenida por los controladores Intel y AMD es $S_{IntelRAM} = 8/2 = 4$ y $S_{AMDRAM} = 4/2 = 2$, respectivamente.

Aplicando la Ley de Amdahl al tiempo de ejecución global (que incluye ambas fracciones: procesador y memoria), la aceleración global obtenida por los procesadores Intel es:

$$S_{Intel} = \frac{1}{(1 - 0{,}5 - 0{,}3) + \frac{0{,}5}{2{,}4} + \frac{0{,}3}{4}} = 2{,}069 \tag{1}$$

La aceleración global obtenida por los procesadores AMD es:

$$S_{AMD} = \frac{1}{(1 - 0{,}5 - 0{,}3) + \frac{0{,}5}{3{,}6} + \frac{0{,}3}{2}} = 2{,}045 \tag{2}$$

Por lo tanto, las placas Intel son marginalmente mejores para esta aplicación.

□

Ejercicio 1.13.

Se plantea modificar el diseño de la arquitectura de un procesador *l*oad/store para añadir un indicador de acarreo. La modificación afecta al diseño de la UAL y al juego de instrucciones. El análisis del uso del juego de instrucciones y del *CPI* medio del procesador original, con reloj a 500 MHz, es el siguiente:

tipo	frecuencia	*CPI*
aritmética	50 %	1
carga	20 %	2
almacenamiento	10 %	1.2
bifurcación	20 %	1.2

Se ha valorado que la modificación permitirá ahorrar una de cada diez instrucciones aritméticas, pero el *CPI* de éstas subirá a 1.2 y el de las bifurcaciones a 1.5. Por otra parte, al rediseñar el decodificador, la UAL y la detección de riesgos, se ha visto que la máxima frecuencia de reloj aplicable es de 400 MHz.

Estudia la mejora siguiendo estos pasos:

1. Calcula el *CPI* del procesador original
2. Calcula el *CPI* del procesador modificado
3. Determina qué diseño es más rápido y cuantifica la respuesta

Solución:

1. De la tabla de frecuencias se deduce que *CPI*(original) = 1.26 ciclos.
2. Nótese que la modificación reduce en $10\,\% \cdot 50\,\% = 5\,\%$ el número de instrucciones necesarias.

$$\mathit{CPI}(modificado) = \frac{0{,}45 \cdot 1{,}2 + 0{,}2 \cdot 2 + 0{,}1 \cdot 1{,}2 + 0{,}2 \cdot 1{,}5}{0{,}95} = 1{,}43$$

3. Hay que comparar los tiempos de ejecución de los programas en ambos procesadores. Si I es el número de instrucciones de un programa en el procesador original, la comparación será:

$$\frac{T(\text{modificado})}{T(\text{original})} = \frac{(I \cdot 0{,}95) \cdot 1{,}43 \cdot 2{,}5}{I \cdot 1{,}26 \cdot 2} = 1{,}35$$

   donde los tiempos de ciclo de reloj se han expresado en ns. El diseño original es un 35 % más rápido que el modificado.

□

Ejercicio 1.14.

Se tiene un procesador load/store de 32 bits dotado de instrucciones que manipulan *bytes*, *halfwords* y *words*. Sobre este procesador se ha monitorizado la aplicación del programa TEX, y se ha obtenido que el 11 % de las referencias a datos se hacen a *bytes* y *halfwords* y el resto a *words*. También se sabe que el 36 % de las instrucciones del programa son de acceso a la memoria, siendo las instrucciones de *load* el doble de frecuentes que las de *store*. Finalmente, se ha medido que CPI=1.

Las instrucciones de lectura de memoria son las mismas del juego del *MIPS*: (*lb*, *lbu*, *lh*, *lhu* y *lw*). Los mismo se puede decir de las escrituras (*sb*, *sh* y *sw*)

Con el objeto de mejorar las prestaciones, se plantea realizar las siguientes modificaciones:

- Eliminar las instrucciones de acceso a *bytes* y *halfwords* del juego de instrucciones. Como consecuencia, los programas que necesiten esta funcionalidad deberán utilizar otras instrucciones del procesador:

Con acceso a *bytes*	Sin acceso a *bytes*
`lb/lbu/lh/lhu r,A`	`lw r,A'` `extract n,r`

`sb/sh r,A`	`lw r',a'`
	`insert n,r,r'`
	`sw r',a'`

  Las nuevas instrucciones de inserción (*insert*) y extracción (*extract*) disponen de las versiones adecuadas para procesar todos los tipos de datos originales *bytes* y *halfwords* con y sin signo.

- Aumentar la frecuencia de reloj, al simplificar el diseño de la interfaz del núcleo del procesador con la memoria cache.

¿En cuánto habrá que incrementar la frecuencia de reloj para que sea interesante incorporar las modificaciones propuestas?

Solución:

La resolución exige la evaluación del incremento de instrucciones por los cambios en el juego. Una vez obtenido de este dato, la ecuación del tiempo de ejecución de los programas permitira obtener el incremento de la frecuencia del reloj que compense el cambio.

El número de instrucciones eliminadas depende de la frecuencia de instrucciones de acceso de memoria y del tipo de dato accedido. Por ejemplo, la frecuencia f_{hb} de instrucciones de lectura que hacen referencia a *bytes* y *halfwords* (*lb*, *lbu*, *lh* y *lhu*) es de $11\,\%$ respecto del total de lecturas, que a su vez son los $2/3$ de los accesos a memoria. Como los accesos a memoria suman el $36\,\%$ de las instrucciones, podemos expresar esta frecuencia como $f_{lhb} = 2/3 \cdot 0{,}11 \cdot 0{,}36$ respecto del total de instrucciones. Igualmente, se puede expresar la frecuencia de instrucciones de escritura en memoria afectadas por el cambio como $f_{shb} = 1/3 \cdot 0{,}11 \cdot 0{,}36$

Cada instrucción de lectura eliminada se tendrá que sustituir con dos instrucciones: una de lectura de memoria *lw* y luego la instrucción de extracción correspondiente *extract*. Por otra parte, cada instrucción de escritura eliminada se ha de sustituir por tres instrucciones: lectura *lw*, inserción *insert* y escritura *sw*.

Balance final: el incremento bruto por la eliminación de las instrucciones de lectura es de $0,11 \cdot \frac{2}{3} \cdot 0,36 \cdot 1 = 2{,}6\,\%$, y por la eliminación de instrucciones de escritura es $0,11 \cdot \frac{1}{3} \cdot 0,36 \cdot 2 = 2{,}6\,\%$

Incremento en número total de instrucciones: 5.2 %

De la ecuación del tiempo de ejecución de los programas se deduce que habrá que compensar con un incremento mínimo de la frecuencia de reloj en 5.2 %

□

Ejercicio 1.15.

Un equipo de arquitectos de computadores está estudiando mejorar el diseño de un computador que dispone de un juego de instrucciones semejante al *MIPS*. Se plantea la conveniencia de incluir en el juego las instrucciones de manejo de la pila, `push` y `pop` que permitirían sustituir secuencias de instrucciones como:

```
...
sw r1,0[sp]   ; apila r1 y r2
sw r2,-4[sp]
subi sp,sp,8
...
lw r4,0[sp]   ; desapila r4
addi sp,sp,4
...
```

por:

```
...
push r1  ; apila r1 y r2
push r2
...
pop r4   ; desapila r4
...
```

Se sabe que el compilador para este computador sólo utilizará la pila como soporte para las llamadas a subprograma, tanto para el paso de parámetros como para salvar o restaurar el valor de los registros destinados a variables locales, como muestran los siguientes fragmentos de código:

Programa principal:

```
...
sw r1,0[sp]     ; llamada a subprograma
sw r2,-4[sp]    ; (dos parámetros)
subi sp,sp,8
jal subprograma
addi sp,sp,8
...
```

Subprograma:

```
; Punto de entrada al subprograma
sw r1,0[sp],    ; crea espacio para tres
sw r2,-4[sp]    ;      variables locales
sw r3,-8[sp]
sub sp,sp,#12
...             ; comienza el código ...
...
lw r3,0[sp]     ; vuelta al prog. principal
lw r2,4[sp]
lw r1,8[sp]
add sp,sp,#12
jr r31
```

De las medidas de uso del computador sin estas instrucciones se tiene que el 1 % de las instrucciones son llamadas a subprograma. También se dispone de la siguiente estadística del número de variables locales y parámetros de los subprogramas:

Var. loc.	%	Parámetros	%
0	30	0	45
1	25	1	20
2	20	2	15
3	15	3	10
4	10	4	10

Para poder implementar las dos instrucciones, es necesario reducir la frecuencia del reloj, pero se sabe que no cambiará el CPI medio. Si en el diseño original la frecuencia de reloj era de 100 MHz, ¿cuál será la frecuencia mínima del diseño modificado para que interese la modificación propuesta?

Solución:

El cambio en el juego de instrucciones afecta a dos factores de la ecuación del tiempo de ejecución: el número de instrucciones I y el tiempo de ciclo de reloj t_C. Llamemos T_{ej} al tiempo de ejecución en el diseño original sin instrucciones de pila, y T'_{ej} al tiempo de ejecución después de la modificación propuesta.

Veamos el impacto en el número de instrucciones con el juego modificado I' en referencia al original I:

- En el paso de parámetros: se ahorra una instrucción si hay algún parámetro. Esto ocurre en el 55 % de las llamadas a procedimiento.
- En las variables locales: Se ahorran dos instrucciones si hay alguna variable local. Esto ocurre en el 70 % de los casos

El número de instrucciones es $I' = I \cdot (1 - (0{,}55 + 2 \cdot 0{,}7) \cdot 0{,}01) = 0{,}98 \cdot I$

Así que la condición que debe satisfacer el reloj del procesador modificado es

$$\frac{T_{ej}}{T'_{ej}} = \frac{I \cdot 1 \cdot 10}{0{,}98 \cdot \ t'_C} > 1$$

Para mantener el tiempo de ejecución, el reloj del diseño modificado habrá de oscilar a 98 MHz

□

Ejercicio 1.16.

Estudia la posibilidad de añadir un nuevo modo de direccionamiento indexado al *MIPS* para instrucciones de *load* y *store*. La dirección del operando en memoria se calcula mediante la suma de dos registros y un desplazamiento de 11 bits, de manera que secuencias de instrucciones como

```
dadd r1,r1,r2
ld rd,o(r1)
```

pueden ser sustituidas por:

```
ldi rd,o(r1,r2)
```

El CPI no queda afectado, pero el periodo de reloj del MIPS mejorado es 5 % más largo que el original. Las medidas tomadas sobre el MIPS original indican que el 24 % de las instrucciones ejecutadas son del tipo *load/store*, y en el 10 % de los casos pueden ser sustituidas por las nuevas instrucciones.

Deduce cuál de las dos máquinas es más rápida, cuantificando su velocidad respecto de la más lenta.

Solución:

En función de los componentes del tiempo de ejecución de un programa tipo, se tiene que en el procesador original, el tiempo de ejecución de un programa tipo es:

$$T_{ej} = I \cdot CPI \cdot t_c$$

Después de la modificación, el tiempo de ejecución sería

$$T'_{ej} = I' \cdot CPI' \cdot t'_c = (1 - 0{,}24 \cdot 0{,}1) \cdot I \cdot CPI \cdot (1{,}05 \cdot T_{clk})$$

La aceleración debida a la mejora sería

$$S = \frac{T'_{ej}}{T_{ej}} = 1'024$$

Conclusión: el procesador original es un 2.4 % más rápido que el mejorado.

□

Ejercicio 1.17.

Las extensiones SIMD para MIPS64, conocidas como *MIPS64 SIMD Architecture* (MSA) añaden un nuevo banco de registros con 32 registros de 128 bits (registros `w0`, `w1`, etc.). Dependiendo de la instrucción, cada registro puede tratarse como un vector de 16 bytes (8 bits por componente del vector), un vector de 8 *halfwords*

(16 bits por componente), un vector de 4 *words* (32 bits por componente), o un vector de 2 *doublewords* (64 bits por componente).

Para especificar el tipo de componente, la síntaxis de una instrucción SIMD puede incluir uno de los posibles sufijos: `.b`, `.h`, `.w`, y `.d`; que corresponden a bytes, halfwords, words, y doublewords, respectivamente. Así, la instrucción `addv.w w5,w1,w2` suma uno a uno los 4 componentes **word** de los registros vectoriales `w1` y `w2` (es decir, suma el primer componente de `w1` con el primer componente de `w2`, el segundo con el segundo, y así sucesivamente) y almacena los componentes del vector resultante en el registro `w5`.

Se pretende usar las extensiones MSA para acelerar el siguiente algoritmo, que realiza la operación vectorial $\vec{z} = A \cdot \vec{x} + \vec{y}$:

```
    addi r10,r0,N    ; N es el tamaño de los vectores
    addi r11,r0,0
    addi r12,r0,A
loop:
    lw r20,X(r11)
    lw r21,Y(r11)
    mul r20,r20,r12
    add r21,r21,r20
    sw r21,Z(r11)
    addi r11,r11,+4
    addi r10,r10,-1  ; 1 iteración procesa 1 componente de los vectores x, y, z
    bne r10,r0,loop
```

Para ello se implementan dos versiones que hacen uso de las extensiones MSA. La primera versión (MSA1) ejecuta el siguiente código (las instrucciones MSA están marcadas en negrita):

```
    addi r10,r0,N    ; N es el tamaño de los vectores
    addi r11,r0,0
    ldi.w w12,A
loop:
    ld.w w20,X(r11)
    ld.w w21,Y(r11)
    mulv.w w20,w20,w12
    addv.w w21,w21,w20
    st.w w21,Z(r11)
    addi r11,r11,+16
    addi r10,r10,-4  ; 1 iteración procesa 4 componentes de los vectores x, y, z
    bne r10,r0,loop
```

Teniendo en cuenta que cada iteración del bucle en MSA1 procesa 4 componentes (en vez de 1 componente, como el algoritmo original). Responde a las siguientes preguntas:

1. Asumiendo que N es un múltiplo de 4, ¿cuantas instrucciones ejecuta MSA1 en total?

2. Teniendo en cuenta que el CPI de todas las instrucciones es 1 y que las extensiones MSA no afectan a la frecuencia del procesador ¿cuál es la aceleración obtenida por MSA1 respecto al algoritmo original?

3. La segunda versión (MSA2) mejora MSA1 mediante el uso de la instrucción maddv, que combina las instrucciones `mulv.w` y `addv.w`. De manera que el siguiente fragmento de código:

```
mulv.w w20,w20,w12
addv.w w21,w21,w20
```

se sustituye por:

```
maddv.w w21,w20,w12
```

Sin embargo, debido a las dependencias de datos con las instrucciones `ld.w` y al hecho de que realiza dos operaciones, el CPI de la instrucción `maddv.w` es 3. Teniendo esto en cuenta, ¿cuanto debería acelerarse la frecuencia del procesador para que la ejecución de MSA2 ofrezca mejores prestaciones que la de MSA1?

Solución:

1. ¿Cuantas instrucciones ejecuta MSA1 en total?

$$I_{MSA1} = 3 + \frac{8 \cdot N}{4} \approx 2 \cdot N$$

2. ¿Cuál es la aceleración obtenida por MSA1 respecto al algoritmo original?

$$T_{MSA1} = I_{MSA1} \cdot CPI \cdot t = 2 \cdot N \cdot t$$

   Por otro lado:

$$I_{orig} = 3 + 8 \cdot N \approx 8 \cdot N \rightarrow T_{orig} = 8 \cdot N \cdot t$$

   Por lo tanto:

$$S = \frac{T_{orig}}{T_{MSA1}} = \frac{8 \cdot N \cdot t}{2 \cdot N \cdot t} = 4$$

3. ¿Cuanto debería acelerarse la frecuencia del procesador para que la ejecución de MSA2 ofrezca mejores prestaciones que la de MSA1?

$$I_{MSA2} = 3 + \frac{7 \cdot N}{4} \approx 1{,}75 \cdot N$$

$$CPI_{MSA2} \approx \frac{6}{7} \cdot 1 + \frac{1}{7} \cdot 3 = \frac{9}{7} \approx 1{,}28$$

$$T_{MSA2} = I_{MSA2} \cdot CPI_{MSA2} \cdot t_{MSA2} = 1{,}75 \cdot N \cdot 1{,}28 \cdot t_{MSA2} = 2{,}24 \cdot N \cdot t_{MSA2}$$

   Para que MSA2 ofrezca mejores prestaciones que MSA1 debe cumplirse que:

$$T_{MSA2} < T_{MSA1} \rightarrow 2{,}24 \cdot N \cdot t_{MSA2} < 2 \cdot N \cdot t \rightarrow \frac{2{,}24}{2} < \frac{t}{t_{MSA2}} \rightarrow 1{,}12 < \frac{t}{t_{MSA2}}$$

   Es decir, el reloj tiene que ser al menos aproximadamente un 12 % más rápido.

□