Problema

Dispone del siguiente fragmento de código intermedio:

```
Loop: LD   F0,0(R1)
      ADDD F4,F0,F2
      SD   0(R1),F4
      SUBI R1,R1,#8
      BNEZ R1,Loop
```

y de un procesador VLIW con un formato de instrucción de 5 slots (4 bytes por slot) que admite dos operaciones de carga/almacenamiento (2 ciclos de latencia), dos operaciones en coma flotante (3 ciclos de latencia) y una operación entera/salto (1 ciclo de latencia). Sin considerar la existencia del hueco de retardo de salto en la planificación, se pide que:

a) Transforme el código intermedio en código VLIW para el procesador indicado.
b) A partir del código anterior y mediante el desenrollamiento del bucle original, complete los slots libres del código VLIW del apartado anterior.
c) Realice el desenrollamiento software del bucle original. Considere que un slot de operación en coma flotante puede ejecutar restas enteras.
d) Calcule para los dos apartados anteriores el número de operaciones por ciclo reloj, el número de ciclos consumidos para un vector de 800 elementos, el tamaño del código en memoria y el porcentaje de espacio desaprovechado.

Solución

a) Una solución valida aunque no óptima es la siguiente.

	Carga/almacenamiento	**Carga/almacenamiento**	**Operaciones FP**	**Operaciones FP**	**Enteras/saltos**
1	`LD F0,0(R1)`				
2					
3			`ADDD F4,F0,F2`		
4					
5					
6	`SD 0(R1),F4`				
7					
8					`SUBI R1,R1,#8`
9					`BNEZ R1,Loop`

Número de ciclos consumidos: 9
Número de operaciones realizadas: 5
Operaciones por ciclo: 0,555
Tamaño en memoria: 9 instrucciones * 20 bytes = 180 bytes
Espacio utilizado: 5 operaciones * 4 bytes = 20 bytes
% espacio desaprovechado: 88,89
Ciclos ejecutados para 800 elementos: 9 ciclos * 800 iteraciones : 7200 ciclos
Instrucciones procesadas: 9 * 800 iteraciones: 7200 instrucciones

Otra solución válida que mejora a la anterior es la que se muestra a continuación.

	Carga/almacenamiento	Carga/almacenamiento	Operaciones FP	Operaciones FP	Enteras/saltos
1	LD F0,0(R1)				
2					
3			ADDD F4,F0,F2		
4					
5					
6	SD 0(R1),F4				SUBI R1,R1,#8
7					BNEZ R1,Loop

Número de ciclos consumidos: 7
Número de operaciones realizadas: 5
Operaciones por ciclo: 0,71
Tamaño en memoria: 7 instrucciones * 20 bytes = 140 bytes
Espacio utilizado: 5 operaciones * 4 bytes = 20 bytes
% espacio desaprovechado: 85%
Ciclos ejecutados para 800 elementos: 7 ciclos * 800 iteraciones : 6300 ciclos
Instrucciones procesadas: 7 * 800 iteraciones: 6300 instrucciones

b) En base a la solución no óptima del apartado (a) se tendría:

	Carga/almacenamiento	Carga/almacenamiento	Operaciones FP	Operaciones FP	Enteras/saltos
1	LD F0,0(R1)	LD F6,-8(R1)			
2	LD F10,-16(R1)	LD F14,-24(R1)			
3	LD F18,-32(R1)	LD F22,-40(R1)	ADDD F4,F0,F2	ADDD F8,F6,F2	
4			ADDD F12,F10,F2	ADDD F16,F14,F2	
5			ADDD F20,F18,F2	ADDD F24,F22,F2	
6	SD 0(R1),F4	SD -8(R1),F8			
7	SD -16(R1),F12	SD -24(R1),F16			
8	SD -32(R1),F20	SD -40(R1),F24			SUBI R1,R1,#48
9					BNEZ R1,Loop

Número de ciclos consumidos: 9
Número de operaciones realizadas: 20
Operaciones por ciclo: 20/9=2,222
Tamaño en memoria: 9 instrucciones * 20 bytes = 180 bytes
Espacio utilizado: 20 operaciones * 4 bytes = 80 bytes
% espacio desaprovechado: 55 %
Ciclos ejecutados para 800 elementos: 9 ciclos * 134 iteraciones : 1206 ciclos
Instrucciones procesadas: 9 * 134 iteraciones: 1206 instrucciones

En base a la solución óptima del apartado (a) se tendría:

	Carga/almacenamiento	**Carga/almacenamiento**	**Operaciones FP**	**Operaciones FP**	**Enteras/saltos**
1	`LD F0,0(R1)`	`LD F6,-8(R1)`			
2					
3			`ADDD F4,F0,F2`	`ADDD F8,F6,F2`	
4					
5					
6	`SD 0(R1),F4`	`SD -8(R1),F8`			`SUBI R1,R1,#16`
7					`BNEZ R1,Loop`

Número de ciclos consumidos: 7
Número de operaciones realizadas: 8
Operaciones por ciclo: 8/7=1,14
Tamaño en memoria: 7 instrucciones * 20 bytes = 140 bytes
Espacio utilizado: 8 operaciones * 4 bytes = 32 bytes
% espacio desaprovechado: 77%
Ciclos ejecutados para 800 elementos: 7 ciclos * 400 iteraciones : 2800 ciclos
Instrucciones procesadas: 7 * 400 iteraciones: 2800 instrucciones

c) Lo primero que hay que realizar es obtener el patrón de ejecución con el objeto de visualizar el prólogo, el patrón que se repite y el epílogo.

Iteracción 1	Iteracción 2	Iteracción 3	Iteracción 4	Iteracción 5	Iteracción 6
LD F0,0(R1)					
	LD F0,-8(R1)				
ADDD F4,F0,F2		LD F0,-16(R1)			
	ADDD F4,F0,F2		LD F0,-24(R1)		
		ADDD F4,F0,F2		LD F0,-32(R1)	
SD 0(R1),F4			ADDD F4,F0,F2		LD F0,-40(R1)
	SD -8(R1),F4			ADDD F4,F0,F2	
		SD -16(R1),F4			ADDD F4,F0,F2
			SD -24(R1),F4		
				SD -32(R1),F4	
					SD -40(R1),F4

Tras visualizar el esquema, hay que trasladarlo a las instrucciones VLIW del procesador disponible.

Carga/almacenamiento	Carga/almacenamiento	Operaciones FP	Operaciones FP	Enteras/saltos
LD F0,0(R1)				
LD F0,-8(R1)				
LD F0,-16(R1)		ADDD F4,F0,F2		
LD F0,-24(R1)		ADDD F4,F0,F2		
LD F0,-32(R1)		ADDD F4,F0,F2		
LD F0,-40(R1)	SD 0(R1),F4	ADDD F4,F0,F2	SUBI R1,R1,#8	BNEZ R1,Loop
	SD -8(R1),F4	ADDD F4,F0,F2		
	SD -16(R1),F4	ADDD F4,F0,F2		
	SD -24(R1),F4			
	SD -32(R1),F4			
	SD -40(R1),F4			

Dado que la instrucción de comparación realiza la comparación con 0, es necesario reajustar los desplazamientos de las instrucciones de carga/almacenamiento y el contenido del registro R1 con el objeto de que el último elemento almacenado lo sea en la posición de memoria M[8] tal y como sucede en el bucle original (observe en el bucle escalar original que se almacena en M[0+R1] y tras decrementar se comprueba que R1 sea cero, en caso afirmativo el bucle concluye).

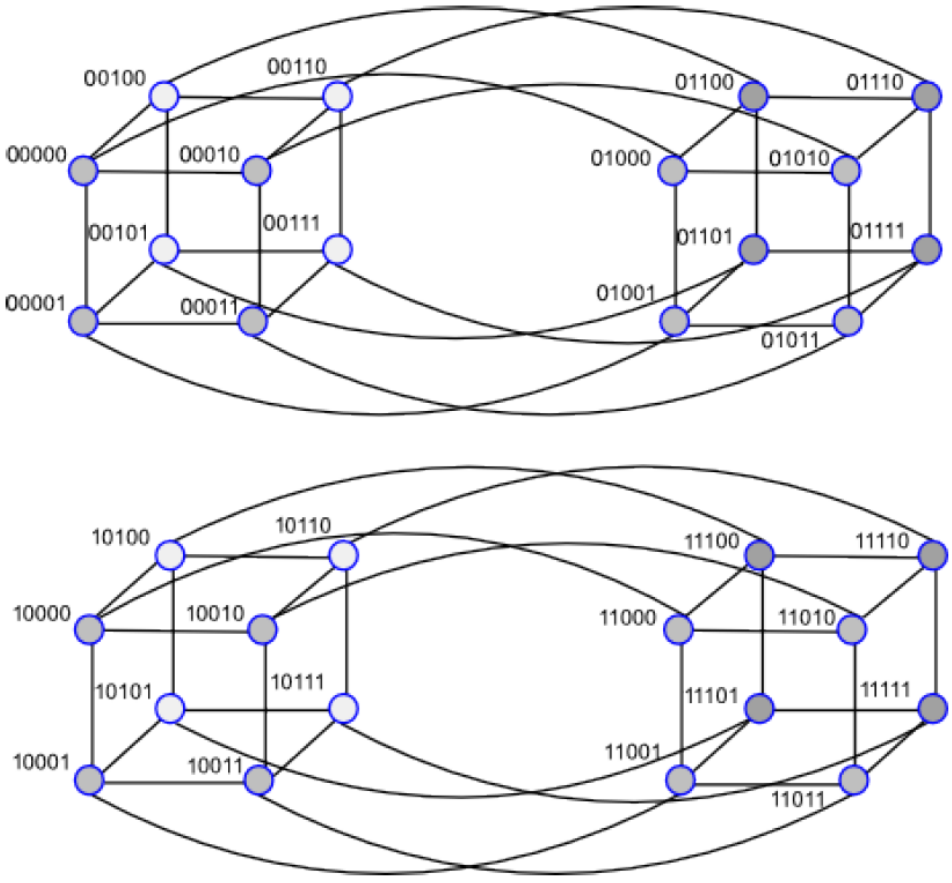
En este caso, el valor inicial de R1 debe ser R1 = R1-48 se procede al proceder al ajuste de los desplazamientos de las instrucciones de carga/almacenamiento. Se tiene así:

Carga/almacenamiento	Carga/almacenamiento	Operaciones FP	Operaciones FP	Enteras/saltos
LD F0,48(R1)				
LD F0,40(R1)				
LD F0,32(R1)		ADDD F4,F0,F2		
LD F0,24(R1)		ADDD F4,F0,F2		
LD F0,16(R1)		ADDD F4,F0,F2		
LD F0,8(R1)	SD 48(R1),F4	ADDD F4,F0,F2	SUBI R1,R1,#8	BNEZ R1,Loop
	SD 48(R1),F4	ADDD F4,F0,F2		
	SD 40(R1),F4	ADDD F4,F0,F2		
	SD 32(R1),F4			
	SD 24(R1),F4			
	SD 16(R1),F4			

Número de ciclos consumidos: 1 ciclo
Número de operaciones realizadas: 5 operaciones
Operaciones por ciclo: 5 operaciones/ciclo
Tamaño en memoria: 11 instrucciones * 20 bytes = 220 bytes
Espacio utilizado: 20 operaciones * 4 bytes = 80 bytes
Espacio desaprovechado: 63 %
Ciclos ejecutados para 800 elementos: 5 del prólogo + 5 del epílogo + 95 iteraciones de 1 ciclo : 105 ciclos
Instrucciones procesadas: 105 instrucciones

Solución al problema 3

a)



b) La distancia de Hamming para los procesadores 00000 y 11111 se calcula utilizando la operación XOR. Así 00000 ⊕ 11111 = 11111, siendo la distancia el número de bits a 1 en el resultado de dicha operación, es decir, 5. El esquema del camino más corto será

$$00000 \rightarrow 00001 \rightarrow 00011 \rightarrow 00111 \rightarrow 01111 \rightarrow 11111$$

Dado que todos los bits de la distancia de Hamming entre ambos procesadores tiene valor 1, el camino más corto se realizará cambiando todos los bits del procesador inicial de 0 a 1, desde el menos significativo al más significativo.

c) La conectividad de arco de una red hipercubo se describe como “el menor número de arcos que deben eliminarse para obtener dos redes disjuntas”. Esta conectividad se puede calcular como el log2(p). Dado que un hipercubo de dimensión 5 tiene 32 procesadores, log2(32) = 5.