

Correction – exercice 2 TD2 IF4-ARH

1. Le corps de chaque itération est indépendant donc c'est une boucle parallèle.

2. Code de la boucle et ordonnancement

2.1 Choix :

R1 = adresse du dernier élément du vecteur, c'est à dire l'élément avec la plus grande adresse
F2 = valeur du scalaire

Pour simplifier on suppose que l'adresse du premier élément est zéro.

2.2 Code DLX de la Boucle :

Boucle :	LD	F0,0(R1)	; F0 = élément du vecteur
	ADDD	F4,F0,F2	; ajouter le scalaire à F2
	SD	0(R1),F4	; ranger le résultat
	SUBI	R1,R1,#8	; décrémenter le pointeur
			; de 8 octets (par double mot
	BNEZ	R1,Boucle	; branchement si R1 !=0

2.3 Exécution de la boucle sans ordonnancement

Boucle :		Démarre au cycle
LD	F0,0(R1)	1
nop		2 suspension
ADDD	F4,F0,F2	3
nop		4 suspension
nop		5 suspension
SD	0(R1),F4	6
SUBI	R1,R1,#8	7
BNEZ	R1,Boucle	8
nop		9

on a donc 9 cycles par itération.

2.4 Exécution de la boucle avec ordonnancement – branchement retardé

Boucle :	LD	F0,0(R1)	1
	nop		2
	ADDD	F4,F0,F2	3
	SUBI	R1,R1,#8	4
	BNEZ	R1,Boucle	5
	SD	8(R1),F4	6 ; branchement retardé modifié car échanger avec SUBI

résultat :

Le temps d'exécution a été réduit de 9 à 6 cycles. On termine une itération et on range un élément du vecteur tous les 6 cycles, le traitement à proprement dit (chargement, addition et rangement) ne prend que 3/6 cycles.

Les 3 cycles restants sont dus à la gestion de la boucle (SUBI, BNEZ) et une suspension. Pour éliminer ces 3 cycles, on doit faire plus d'opérations dans la boucle par rapport aux instructions de gestion de boucle.

Le compilateur détermine : échange entre SUBI et SD, modifie l'adresse où SD range F4 : 8(R1) au lieu de 0(R1).

2.5 Exécution de la boucle avec déroulage de boucle

La solution : pour augmenter le nombre d'instructions par rapport au branchement et aux instructions de gestion de boucle s'appelle le déroulage de boucle.

Le déroulage de boucle permet d'augmenter le nombre d'instructions par rapport au branchement et aux instructions de gestion de boucle.

Donc le déroulage de boucle consiste à dupliquer le corps de la boucle puis à adapter le code de terminaison de boucle.

Boucle déroulée 4 fois (4 copies du corps de boucle)

Boucle :	LDF	0,0(R1)	2 cycles	
	ADDD	F4,F0,F2	3 cycles	<u>6 cycles</u>
	SD	0(R1),F4	1 cycle	
1 -----				
	LD	F6,-8(R1)		
	ADDD	F8,F6,F2		<u>6 cycles</u>
	SD	-8(R1),F8		
2 -----				
	LD	F10,-16(R1)		
	ADDD	F12,F10,F2		<u>6 cycles</u>
	SD	-16(R1),F12		
3 -----				
	LD	F14,-24(R1)	2 cycles	<u>6 cycles</u>
	ADDD	F16,F14,F2	3 cycles	
	SD	-24(R1),F16	1 cycle	
4 -----				
	SUBI	R1,R1,#32	1 cycle	<u>3 cycles</u>
	BNEZ	R1, BOU	2 cycles	

Résultat :

On a éliminé 3 branchements et 3 décréments de R1. Les adresses des chargements et des rangements ont été rectifiées pour permettre la fusion des instructions SUBI sur R1

Sans ordonnancement cette boucle s'exécute en 27 cycles (4*6 +3).

Soit 6.8 cycles pour chacun des quatre éléments.

Remarque : le déroulage de boucle est fait normalement tôt dans le processus de compilation, de telle façon que les calculs redondants puissent être visibles et éliminés par l'optimiseur.

Boucle déroulée et ordonnancée

Boucle :			Démarre au cycle
	LD	F0,0(R1)	1
	LD	F6,-8(R1)	2
	LD	F10,-16(R1)	3
	LD	F14,-24(R1)	4
	ADDD	F4,F0,F2	5
	ADDD	F8,F6,F2	6
	ADDD	F12,F10,F2	7
	ADDD	F16,F14,F2	8
	SD	0(R1),F4	9
	SD	-8(R1),F8	10
	SD	-16(R1),F12	11
	SUBI	R1,R1,#32	12
	BNEZ	R1,Boucle	13
	SD	8(R1),F16	14

Résultat :

Le temps d'exécution de la boucle déroulée est de 14 cycles, soit 3.5 cycles par élément par rapport à 6.8 cycles avant ordonnancement et à 6 cycles avec ordonnancement pour la boucle non déroulée.

SD 8(R1),F16 $8 - 32 = -24$