

## TD-3 : Pipelines scalaires et prédiction de branchement

### Exercice 1

On considère une architecture RISC possédant 4 types d'instructions :

- A = arithmétiques et logiques : registre- registre
- C = chargement : lecture mémoire
- R = rangement : écriture mémoire
- B = branchement. Les branchements conditionnels se font sur code condition

Cette architecture a deux implantations, l'une avec le pipeline P1 dont les phases sont décrites table I et l'autre avec le pipeline P2 qui est décrit table II. Dans les deux cas, les caches instructions et données sont séparés.

LI	Lecture de l'instruction	LI1	Début lecture de l'instruction
LR	Pour toutes les instructions, décodage de l'instruction et lecture des opérandes registre. Pour les types C et R, calcul de l'adresse mémoire Pour le type B, calcul de l'adresse de branchement	LI2	Fin lecture de l'instruction
EX	Pour le type A, opération UAL. Pour le type R, lecture du 3 <sup>ème</sup> opérande. Pour les types C et R, accès mémoire	LR	Décodage de l'instruction et lecture des opérandes registre
RR	Rangement du résultat arithmétique (A) ou de la lecture (C) dans le banc de registres.	EX	Pour le type A, opération UAL. Pour le type R, lecture du 3 <sup>ème</sup> opérande. Pour les types C et R, calcul de l'adresse mémoire. Pour le type B, calcul de l'adresse de branchement
		M1	Pour les types C et R, début accès mémoire
		M2	Pour les types C et R, fin accès mémoire
		RR	Rangement du résultat arithmétique (A) ou de la lecture (C) dans le banc de registres.
<i>Table 1 : PIPELINE 1</i>		<i>Table 2 : PIPELINE 2</i>	

- 1) Définir l'ensemble des dépendances qui peuvent exister entre une instruction qui génère une donnée ou une adresse et une instruction qui utilise cette donnée ou cette adresse.
- 2) Pour toutes les dépendances définies dans la question précédente, on introduit tous les mécanismes d'anticipation nécessaires (bypass) pour supprimer ou réduire les suspensions. Définir dans chaque cas le « bypass » nécessaire et en déduire le nombre de cycles de suspension (avec 0 quand il n'y a pas de suspension)..
- 3) Quelle est la taille du délai de branchement pour P1 et pour P2 ?
- 4) Quelles sont les ressources matérielles nécessaires aux deux pipelines ?

### Exercice 2

On considère une architecture à cache commun instructions et données

- 1) Un pipeline à cinq étages (LI, LR, EX, MEM, RR) est-il utile ?
- 2) Proposer un pipeline d'exécution, qui permette la lecture du registre Rd pendant la phase EX des instructions de rangement, en supposant que le banc de registre n'admet que 2 accès simultanés en lecture.
- 3) Donner la liste des aléas de données de ce pipeline avec anticipation.

### Exercice 3

Soit le code suivant, utilisant le jeu d'instructions MIPS et en supposant les branchements non retardés.

1	ld r5,(r1)
2	add r5, r5, 1
3	ld r6, 4(r1)
4	stlt r2,r5,r6
5	bne r2,r0, Suite
6	addi r5,r5,-1
7	ld r6, 8(r1)
8 Suite	st r6,(r1)

- 1) Soit un processeur avec le pipeline simple à 5 étages LI, LR, EX, AM, ER. Le processeur est complètement pipeliné et a tous les mécanismes d'anticipation nécessaires. Toutes les instructions prennent un cycle et les instructions de chargement prennent deux cycles lorsqu'il y a utilisation du résultat par l'instruction suivante. On suppose que le branchement n'est pas pris et qu'il est correctement prédit. Donner le schéma du pipeline lors de l'exécution du code sur le processeur.
- 2) Soit un processeur avec un pipeline à six cycles (le cache données prend deux cycles AM et AM2). Dans un tel pipeline, les instructions de chargement prennent trois cycles lorsqu'il y a utilisation du résultat par l'instruction suivante. On suppose que le branchement n'est pas pris et qu'il est correctement prédit. Donner le schéma du pipeline lors de l'exécution du code sur le processeur.
- 3) Réordonner les instructions pour avoir l'exécution la plus rapide possible avec le pipeline de la question 1. On suppose que le branchement n'est pas pris et qu'il est correctement prédit. Donner le schéma du pipeline lors de l'exécution du code sur le processeur.
- 4) Soit le code original et le pipeline de la question 1. On suppose que le branchement est prédit non pris. En faite, le branchement est pris (ce qui provoque une mauvaise prédiction). Le processeur détecte la mauvaise prédiction dans l'étage EX du pipeline. Donner le schéma du pipeline lors de l'exécution du code sur le processeur.

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	ld r5,(r1)																		
2	add r5, r5, 1																		
3	ld r6, 4(r1)																		
4	cmplti r2,r5,r6																		
5	bne r2,r0, Suite																		
6	addi r5,r5,-1																		
7	ld r6, 8(r1)																		
8	st r6,(r1)																		

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	ld r5,(r1)																		
2	add r5, r5, 1																		
3	ld r6, 4(r1)																		
4	cmplti r2,r5,r6																		
5	bne r2,r0, Suite																		
6	addi r5,r5,-1																		
7	ld r6, 8(r1)																		

8	st r6,(r1)																	
---	------------	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	ld r5,(r1)																		
2	add r5, r5, 1																		
3	ld r6, 4(r1)																		
4	cmplti r2,r5,r6																		
5	bne r2,r0, Suite																		
6	addi r5,r5,-1																		
7	ld r6, 8(r1)																		
8	st r6,(r1)																		

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	ld r5,(r1)																		
2	add r5, r5, 1																		
3	ld r6, 4(r1)																		
4	cmplti r2,r5,r6																		
5	bne r2,r0, Suite																		
6	addi r5,r5,-1																		
7	ld r6, 8(r1)																		
8	st r6,(r1)																		

### Exercice 4 : prédicteurs de branchement

Soit le programme C suivant :

```

long a=0, b=0, n=0, p=0;
main()
{
int i, n, p;
for (i=0; i<24; i++)
    {
    a = (a+1)%2
    b= (b+1)%3
    if (a>=b)
        n++;
    else p++;
    }
}
    
```

On considère le branchement conditionnel correspondant au if(a>=b). Le branchement est pris (P) si (a<b) et non pris (NP) autrement.

On associe un prédicteur à ce branchement. On utilise soit un prédicteur 1 bit, soit un compteur 2 bits. Le compteur 2 bits a 4 états : fortement non pris (FNP), faiblement non pris (fNP), faiblement pris (fP) et fortement pris (FP) auxquels on peut associer les valeurs 0, 1, 2 et 3.

Dans tous les cas, les prédicteurs sont initialisés à NP (ou FNP pour le compteur 2 bits)

Pour les 24 itérations de la boucle, quel est le nombre de prédictions correctes dans les cas suivants :

- a) on utilise un prédicteur 1 bit par branchement
- b) on utilise un prédicteur 2 bits par branchement

