Département de génie électrique et de génie informatique
Université Laval
Prof. : André Zaccarin

GIF-17455 – Architecture des microprocesseurs

Examen Partiel

Lundi le 20 novembre 2006 Durée: 9h30-11h20

Question 1 (35 points)

Répondez à sept des huit questions suivantes (chaque sous-question vaut 5 points)

a) Pour un processeur avec ordonnancement dynamique qui supporte la spéculation, donnez deux obstacles structurels qui peuvent empêcher le lancement d'une instruction.

b) Pour un processeur avec ordonnancement dynamique qui ne supporte pas la spéculation, dans quelle situation ne peut-on pas effectuer l'écriture en mémoire même si l'adresse d'écriture et la donnée à écrire sont disponibles ?

c) Pour un processeur avec ordonnancement dynamique qui supporte la spéculation, les valeurs mises sur le bus commun des données par les unités fonctionnelles sont-elles lues par le fichier des registres ? Pourquoi ?

d) La table de prédiction des branchements contient 2048 bits. Des prédicteurs (2,2) sont utilisés. Comment cette table est-elle indexée lorsque le processeur exécute une instruction de branchement ?

e) En plus des prédicteurs de branchement, quelle autre structure matérielle est utilisée pour supporter le processeur lors de l'exécution des branchements ? Pourquoi est-elle utile ?

f) Donnez un exemple où la prédication est préférable à l'utilisation de branchements.

g) Est-ce que le code compilé pour un processeur VLIW peut-être exécuté sur un autre processeur VLIW qui a le même jeu d'instructions mais une architecture matérielle différente ? Pourquoi ?

h) Décrivez brièvement le support matériel qui permet à un compilateur de déplacer une instruction de lecture avant une instruction de branchement, et de ne pas immédiatement traiter une exception qui serait générée par cette instruction de lecture.

Question 2 (20 pts)

Deux branchements consécutifs dans un programme ont le comportement suivant :

	1	2	3	4	5	6	7	8	9
BR1	NP	NP	P	NP	P	NP	P	NP	P
BR2	NP	NP	P	NP	P	NP	P	P	NP

où « P » désigne un branchement pris, et « NP », un branchement non pris.

a) Pour le branchement BR2, donnez la performance d'un prédicteur à deux bits, initialisé à l'état 00 (non pris).

b) Pour le même branchement BR2, donnez un prédicteur qui aurait une meilleure performance. Les prédicteurs doivent être initialisés à l'état « non pris »

Question 3 (20 pts)

Le code suivant

```
A :   L.D      F2,  0 (R1)      %  MEM(R1+0)  →  F2
      MUL.D    F2,  F2,  F0     %  F2*F0  →  F2
      S.D      F2,  0 (R1)      %  F2 → MEM(R1+0)
      DADDIU   R1,  R1,  #8     %  R1 + 8 → R1
      BNE      R1,  R3,  A      % Si R1 ≠ R3, retourne à « A »
```

est exécuté sur un processeur à ordonnancement statique qui lance une instruction à chaque cycle. Le processeur a suffisamment de ressources pour qu'il n'y ait pas d'aléas structurels.

Le processeur a deux unités fonctionnelles

- une pour les opérations entières et logique, qui requière un cycle pour l'exécution (pipeline IF-ID-EX-MEM-WB)
- une pour les multiplications en point flottant, réalisée en pipeline, qui requière 4 cycles lors de l'exécution (donc pipeline IF-ID-EX1-EX2-EX3-EX4-MEM-WB).

Les accès mémoires, tant pour les lectures que les écritures, prennent un cycle à l'étage MEM. Les adresses sont calculées au cycle EX et ce calcul prend un cycle.

La destination de branchement et la condition de branchement sont déterminées au cycle ID. Le processeur est réalisé avec une instruction de branchement retardé.

a) Donnez les cycles de suspension causés par les dépendances de données et indiquez où ils se produisent.

b) Déroulez et ordonnancez le code pour qu'il s'exécute sans suspension de pipeline causée par les dépendances sur les données. Vous devez minimiser le nombre de déroulements.

Question 4 (25 pts)

Le code suivant est exécuté sur deux processeurs à ordonnancement dynamique, un supportant la spéculation et l'autre non.

```
A :   LD      R2,  0 (R1)     %  MEM(R1+0)  →  R2
      DDIV    R2,  R2,  R4    %  R2/R4  →  R2
      SD      R2,  0 (R1)     %  R2 → MEM(R1+0)
      DADDIU  R1,  R1,  #4    %  R1 + 4  →  R1
      BNE     R2,  R6,  A     % Si R2 ≠R6, retourne à « A »
```

Les informations suivantes décrivent les processeurs:

Ressources : les seules ressources limitées sont les unités fonctionnelles telles que décrites ci-dessous, et le bus commun des données (CDB) – une seule instruction peut écrire sur le bus pendant un cycle.

Accès mémoires : il y a une (1) unité fonctionnelle uniquement pour le calcul des adresses. Le calcul d'une adresse prend 1 cycle, et l'accès mémoire prend aussi 1 cycle.

Branchement : l'évaluation de la condition de branchement prend 1 cycle. Cette opération est effectuée dans une unité fonctionnelle dédiée aux branchements. Les branchements sont prédits comme étant pris, et l'instruction qui suit le branchement est lancée le cycle suivant le lancement du branchement. Ne tenez pas compte du calcul de l'adresse de la destination du branchement.

Unités fonctionnelles :

- il y a une (1) unité fonctionnelle pour les opérations (addition, soustraction, multiplication) sur les entiers et les opérations logiques. Il faut 1 cycle pour exécuter ces opérations.
- Il y a une (1) unité fonctionnelle pour les divisions sur les entiers. Il faut 4 cycles pour exécuter une division. Cette unité fonctionnelle n'est pas réalisée en pipeline.

Timing :

- Une valeur mise sur le bus au cycle #i peut être utilisée dans une unité fonctionnelle au cycle suivant, soit au cycle #i+1
- Une unité fonctionnelle est libre lorsqu'elle diffuse le résultat de l'opération sur le bus.

Complétez les deux tableaux de la page suivante en donnant, pour chaque instruction et lorsqu'il y a lieu, les cycles de lancement, d'exécution, d'accès mémoire, de diffusion sur le bus et de garantie.

Pour le processeur non spéculatif, vous devez justifier les délais dans la progression d'une instruction.

NOM :_____ # Matricule _____

	Lance-ment (IS)	Exécution (EX)	Accès mémoire (MEM)	Écriture sur le bus (WB)	Justification de délai
LD R2, 0 (R1)					
DDIV R2, R2, R4					
SD R2, 0 (R1)					
DADDIU R1, R1, #4					
BNE R2, R6, A					
LD R2, 0 (R1)					
DDIV R2, R2, R4					

	Lance-ment (IS)	Exécution (EX)	Accès mémoire (MEM)	Écriture sur le bus (WB)	Garantie (commit)
LD R2, 0 (R1)					
DDIV R2, R2, R4					
SD R2, 0 (R1)					
DADDIU R1, R1, #4					
BNE R2, R6, A					
LD R2, 0 (R1)					
DDIV R2, R2, R4					