MR1 AEV Mars 2012

# **AUCUN DOCUMENT**



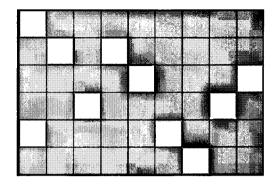
N° de Place : ......

Exercice 1 Cours 6pts
Commentez les trois transparents du cours suivants (rendre cette feuille dans la copie svp)

Decode	Execute	WriteBack
11 12 13 14 14 13 14 14 15 16 16	11	
Decode	Execute	WriteBack
11 12 13 14 14 15 16 16	11   12   13   14   15   15   16   17   17   17   17   17   17   17	
Decode Window	Execute	WriteBack
13 14 11 12 13 14 14 15 16 15 15 15 15 15 15 15 15 15 15 15 15 15	11 12 13 13 16 12 15 15 15 15 15 15 15 15 15 15 15 15 15	11 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

### Exercice 2 Table de réservation (6pts)

Voici une table de réservation d'un pipe-line (les blancs sont actifs, noirs inactifs)



- 1) Quel est le vecteur de collision associé à cette table ?
- 2) En appliquant l'algorithme glouton, donnez la trace des temps d'attente en nombre de cycle (exemple de notation **2 4 (5 2)\*** représente : attendre 2 cycles, puis attendre 4 cycles puis attendre toujours 5 puis 2 cycles pour la suite...). Donnez le rendement de cet algorithme.
- 3) Donnez l'automate qui représente tous les enchainements possibles sans collision y compris celui de l'algorithme glouton. Donnez la trace et le rendement pour chaque cycle présent dans l'automate.

### **Exercice 3 Pipeline 10pts**

Un processeur pipeline à mémoire unique (pas de cache) supporte les instructions avec la syntaxe suivante:

ALUop Rdest, Rsrc1, Rsrc2
ALUop Rdest, Rsrc1, Immediate
MEMop Rdata, Immediate(Raddr) /\* Ld et St pour Load /Store\*/
Control Rsrc1, Target /\* control beq, jsr, etc. \*/

Considérez que les instructions prennent le nombre de cycle suivant dans l'étage EX:

integer ALU 1
data access LD/ST 1
fp add 3 /\* Floating Point \*/
fp/integer multiply 6

Ce processeur est pipeline sur 5 étages (chargement de l'instruction, décodage, exécution ou calcul de l'adresse, accès à la mémoire, mise à jour du banc de registres) ; il dispose alors d'une exécution différente pour chaque type d'instruction:

Integer ALU et data access IF->ID->EX->MEM->WB

FP Add IF->ID->A1->A2->A3->MEM->WB

FP/Integer Multiply IF->ID->M1->M2->M3->M4->M5->M6->MEM->WB

On garantit que la lecture des registres s'effectue dans la première moitié du cycle d'exécution (EX, A1, M1) et l'écriture dans la deuxième moitié du cycle WB du processeur. Les registres entiers et flottants sont différents et les instructions ne peuvent quitter l'étage ID que si tous leurs opérandes sources sont disponibles.

On définit trois types de rupture dans le pipeline.

 Instructions de branchement : lorsque des instructions de branchement changent la valeur du compteur ordinal

#### MR1 AEV Mars 2012

## **AUCUN DOCUMENT**

N° de Place:.....

- Data dépendance : lorsque des instructions dépendent des résultas produits par des instructions précédentes
- Conflits hardware : le matériel ne peut satisfaire en même temps l'ensemble des actions à effectuer
- a) Calculez Pour le code suivant le nombre de cycles nécessaires pour exécuter ce code en considérant qu'il n'y a pas de solution matérielle pour éliminer les conflits du pipeline. Représentez le chronogramme du programme.

```
ld r1, 0(r4)
ld r2, 4(r4)
mul r3, r1, r2
st r3, 0(r5)
ld f1, 8(r4)
ld f2, 12(r4)
fadd f3, f1, f2
st f3, 8(r5)
add r5, r5, 8
sub r4, r4, 16
```

- b) Proposez trois solutions matérielles qui permettent de diminuer les trois ruptures de pipeline présentées.
- c) Calculez le nombre de cycles nécessaires pour exécuter ce code en considérant que le hardware supporte vos propositions. Représentez le chronogramme du programme.
- d) En autorisant le réordonnancement des instructions sur le processeur amélioré en question (b), peut-on gagner des cycles ? Combien de cycle gagne-t-on ?
- e) On utilise un branchement retardé de DEUX cycles. Réordonnez le code suivant pour tirer partie de cette solution. On considère que les registres ne seront pas utilisés en dehors de la boucle.

```
loop:
lw r1, 0(r4)
lw r2, 8(r4)
add r3, r1, r2
st r3, 0(r5)
add r5, r5, 8
sub r4, r4, 16
beq r4, loop
nop
```