Compilation Avancée
Master Informatique
Université de Pierre et Marie Curie
2005-2006

Examen Compilation Avancée – 2nde partie (10 points)

Septembre 2006

A traiter sur une copie séparée

Tous documents autorisés – 7 questions – 2 pages

Réordonnancement de code

Considérer le code assembleur de référence suivant (les commentaires sont donnés en *italique*) :

LD R2, R3 ; *chargement dans le registre R2 du contenu mémoire situé à l'adresse R3*
LD R4, R5 ; *chargement dans le registre R4 du contenu mémoire situé à l'adresse R5*
ADD R5, R2, R4 ; *R5 = R2 + R4*
MUL R6, R7, R8 ; *R6 = R7 * R8*
LD R8, R10 ;chargement *dans le registre R8 du contenu mémoire situé à l'adresse R10*
SUB R2, R6, R11 ; *R2 = R6 – R11*
LD R11, R13 ; *chargement dans le registre R11 du contenu mémoire situé à l'adresse R13*
OR R4, R4, R5 ; *R4 = R4 OU R5*
AND R15, R8, R9 ; *R15 = R8 ET R9*
SUB R13, R11, R1 ; *R13 = R11 – R1*

1. Donner un exemple de dépendances de données de chaque type (RAW : *Read After Write*, WAW : *Write After Write*, WAR *: Write After Read*).

2. Les opérations mémoire (LD, ST) prennent 4 cycles d'exécution, les instructions de multiplication 3 cycles et toutes les autres opérations 1 cycle. Tous les accès à la mémoire sont des succès. Le processeur est pipeliné (toutes les opérations sont pipelinées) : une instruction est exécutée par cycle excepté en cas de dépendances (exemple : pour le code MUL R1, R2, R3 – ADD R4, R1, R5 – MUL est exécutée à t et ADD à t + 3). Donner le temps d'exécution du code de référence (on comptabilisera les temps d'exécution des instructions et les délais générés par les dépendances de données).

3. Réordonner ce code sans renommage de registres afin d'obtenir une meilleure performance. Donner cette performance.

4. Ecrire le code de référence en utilisant du renommage de registres nécessaire afin d'éliminer les fausses dépendances (WAW et WAR). Puis réordonner le code afin d'obtenir la meilleure performance possible. Donner cette performance.

Fonctionnement des caches

On considère un cache de 16 Koctets avec une taille de bloc (ou ligne) de 32 octets. La mémoire est adressée par octets et les adresses sont sur 32 bits.

Soient A et B, deux tableaux de réels double-précision (8 octets) de taille n = 1000. A est implanté à partir de l'adresse 0x00030000 et B à partir de l'adresse 0x00030FA0.

On étudie la boucle :

```
double A[n], B[n] ;
for (k = 0 ; k < n ; k++)
   for (i = 0 ; i < n  ; i++)
        S = S + A[i] + B[i];
```

5. Donner le nombre d'échecs en lecture sur le cache à correspondance directe (on considère uniquement les échecs liés aux accès à A et B ; on suppose que i, k, n et S sont stockés dans des registres i.e. pas dans le cache).

6. Donner le nombre d'échecs en lecture sur le cache associatif par ensemble de 2 blocs avec une politique de remplacement LRU (*Least Recently Used*).

7. Proposer une optimisation du code qui permet d'avoir le même nombre d'échecs dans les questions 5) et 6). Donner ce code.