ARCHITECTURE DES ORDINATEURS
Corrigé Examen Décembre 2010
3H – Tous documents autorisés
Les questions sont indépendantes

On utilise le jeu d'instructions NIOS II et le jeu d'instructions ARM.

PROGRAMMATION ASSEMBLEUR

Soit le code C

```
int a, b, c, d;  // a, b, c et d sont initialement dans R1, R2, R3 et
                    R4.
while (a != b) {
if (a < b) c++;
      else d++;
a += ((b – a) / 2);}
```

Question 1) Ecrire le code assembleur NIOS correspondant au code C, en essayant de minimiser le nombre de branchements.

```
Boucle :
        BEQ R1,R2, Sortie
        BGE R1,R2, L1
        ADDI R3,R3,1
        BEQ R0,R0, L2
L1:     ADDI R4,R4, 1
L2 :    SUB R5,R2,R1
        SRAI R5, R5,1
        ADD R1,R1,R5
        BEQ R0,R0, Boucle
Sortie :
```

Version minimisant le nombre de branchements (2 branchements)

```
Boucle :
        BEQ R1,R2, Sortie        // aller à sortie si R1=R2
        CMPLT R5, R1,R2 // si R1<R2 alors R5=1 sinon 0
        CMPGE R6,R1,R2 // s R1>=R2 alors R6 = 1 sinon 0
        ADD R3,R3,R5  // c++
        ADD R4,R4,R6  //d++
        SUB R5,R2,R1 // b-a
        SRAI R5, R5,1   //(b-a)/2
        ADD R1,R1,R5   // a+=(b-a)/2
        BEQ R0,R0, Boucle :: aller à boucle
Sortie :
```

Question 2) Ecrire le code assembleur ARM correspondant au code C

```
Boucle :
        CMP R1,R2
        BEQ Sortie
        ADDLT R3,R3,#1
        ADDGE R4,R4, #1
        SUB R5,R2,R1
        ADD R1,R1, R5 ASR #1
        B Boucle
Sortie.
```

CACHES

Soit un cache de 4 Ko à correspondance directe, avec des lignes (blocs) de 32 octets. Le processeur a des registres de 32 bits et des adresses de 32 bits.

Question 3) Quels sont les nombres de bits nécessaire pour l'index, l'étiquette et le déplacement (adresse dans la ligne) ?

4 Ko = 2^{12}.
Lignes de 32 octets : 2^5.
Nombre de lignes : $2^7 = 128$.
Déplacement :5 bits
Index : 7 bits
Etiquette : 20 bits

Soit le code C travaillant sur des flottants en double précision (64 bits).

```
double A[1024], i ;
for (i = 0; i < 512; i++) {
      A[i] = A[i] + A[i+512];}
```

Figure 1

Question 4) Pour un cache à écriture simultanée et écriture non allouée, quel est le nombre de défauts de cache lors de l'exécution du code C (figure 1). On rappelle qu'avec l'écriture non allouée, il n'y a pas de défauts de cache en écriture.

Entre l'adresse de A[i] et l'adresse de A[i+512], il y a 512 * 8 = 4096 octets, et les 12 bits de poids faible de l'adresse seront identiques. Les deux adresses iront dans la même ligne et il y aura donc 1 défaut de cache à chaque accès, soit deux défauts lecture par itération.
Nombre total de défauts : 1024.

Question 5)

a) Donner une technique logicielle (modification du code) et une technique matérielle (modification du cache, sans changer la taille de ligne ou taille du cache) qui permettraient de diminuer le nombre de défauts de cache.

Les lignes sont de 32 octets, soit 4 doubles. Il faut donc accéder à tous les éléments d'une ligne avant qu'elle soit éjectée ;

Technique logicielle.
Déroulage de boucle d'ordre 4, soit le programme

```
int A[1024], i ;
for (i = 0; i < 512; i += 4) {
      A[i] = A[i] + A[i+512];
      A[i+1] = A[i+1] + A[i+513];
      A[i+2] = A[i+2] + A[i+514];
      A[i+3] = A[i+3] + A[i+515];}
```

Technique matérielle
Utiliser un cache associatif par ensemble (deux voies).

b) Après modification, quel serait le nombre de défauts de cache pour exécuter le programme de la figure 1 ?
Deux défauts de cache toutes les 4 itérations, soit 256 défauts

Question 6) Avec le cache initial (sans modification logicielle ou matérielle), mais à réécriture et allocation d'écriture, quel est le nombre de défauts de cache pour le code de la figure 1 ? On rappelle qu'avec l'écriture allouée, il y a des défauts de cache en écriture.

Comportement pour 4 itérations
Lecture A(0) charge 0 1 2 3
Lecture A(256) charge 256 257 258 259
Ecriture A(0) charge 0 1 2 3
Lecture A(1) OK
Lecture A(257) charge 256 257 258 259
Ecriture A(1) charge 0 1 2 3
Lecture A(2) OK
Lecture A(258) charge 256 257 258 259
Ecriture A(2) charge 0 1 2 3
Lecture A(3) OK
Lecture A(258) charge 256 257 258 259
Ecriture A(3) charge 0 1 2 3
On a donc 9 défauts de cache pour 4 itérations, soit 2,25 défauts par itération
Total : 2,25 * 512 = 1280

EXECUTION DE BOUCLES

On ajoute au jeu d'instructions NIOS II des instructions flottantes simple précision (32 bits) (Figure 1) et 32 registres flottants F0 à F31 (F0 est un registre normal).
Les additions, soustractions et multiplications flottantes sont pipelinées. Une nouvelle instruction peut démarrer à chaque cycle. Les latences sont de 2 cycles pour LF et de 3 cycles pour les instructions flottantes.
Les branchements ne sont pas retardés.

LF	2	LF ft, déplac(rs)	ft ← MEM [rs +SIMM]
SF	1	SF ft, déplac(rs)	ft → MEM [rs + SIMM]
FADD	3	FADD fd, fs,ft	fd ← fs + ft (addition flottante simple précision)
FMUL	3	FMUL fd, fs,ft	fd ← fs * ft (multiplication flottante simple précision)
FSUB	3	FSUB fd, fs,ft	fd ← fs - ft (soustraction flottante simple précision)

Figure 2 : Instructions flottantes ajoutées (Ce ne sont pas les instructions NIOS)

Soit le programme assembleur P1, qui travaille sur un tableau de flottants simple précision (float) X[N] rangé en mémoire, avec N = 512. L'adresse de A[0] est initialement contenue dans le registre R1. F0 contient une constante flottante 1.0.

```
        ADDI R2 ;R1, 1024
Boucle :LF F1, 0(R1)
        LF F2, 1024(R1)
        FADD F1,F1,F2
        SF F1, 0 (R1)
        FSUB F2,F2,F1
        SF F2, 1024(R1)
        ADDI R1,R1,4
        BLT R1,R2, Boucle
```

Question 7) Donner le code C correspondant au programme P1

```
float X[512]; i ;
for (i = 0; i < 256; i++) {
      X[i] = X[i] + X[i+256];
      X[i+256] = X[i+256] – X[i];
}
```

Question 8) En optimisant, montrer l'exécution cycle par cycle du programme assembleur P1 et donner le nombre de cycles par itération de la boucle du programme assembleur en supposant qu'il n'y a ni pénalité de branchement, ni défauts de cache. Quel est le temps d'exécution total en cycles d'horloge ?

```
        ADDI R2, R1,1024₁₀
1Boucle :LF F1, 0(R1)
2       LF F2, 1024(R1)
3       ADDI R1,R1,4
4       FADD F3,F1,F2
5       FSUB F4,F2,F1
6
7       SF F1, -4(R1)
8       SF F2, 1020 (R1)
9       BLT R1,R2, Boucle
```

9 cycles

Temps total : 1 + 256 *9 = 2305 cycles

Question 9) Quel est le nombre de cycles par itération de la boucle avec déroulage d'ordre 4, en supposant qu'il n'y ni pénalité de branchement, ni défauts de cache ?

```
        ADDI R2, R1,512_10
1Boucle :LF F1, 0(R1)
2       LF F3, 4(R1)
3       LF F5, 8(R1)
4       LF F7, 12(R1)
5       LF F2, 1024(R1)
6       LF F4, 1028 (R1)
7       LF F6, 1032 (R1)
8       LF F8, 1036 (R1)
9       FADD F9,F1,F2
10      FADD F10,F3,F4
11      FADD F11,F5,F6
12      FADD F12,F7,F8
13      FSUB F2,F2,F1
14      FSUB F4,F4,F3
15      FSUB F6,F6,F5
16      FSUB F8,F8,F7
17      SF F9, 0(R1)
18      SF F10, 4(R1)
19      SF F11, 8(R1)
20      SF F12, 12(R1)
21      SF F2, 1024(R1)
22      SF F4, 1028 (R1)
23      SF F6, 1032 (R1)
24      SF F8, 1036 (R1)
25      ADDI R1,R1,16
26      BLT R1,R2, Boucle
```

Soit 26/4 =6,5 cycles /itération
Temps d'exécution total : 1 +6.5*256 = 1665 cycles

PREDICTION DE BRANCHEMENT

Soient trois branchements b1, b2 et b3 appartenant à une boucle interne, qui est exécutée de nombreuses fois à l'intérieur d'une boucle externe. La boucle interne est exécutée 10 fois, et les branchements ont le comportement ci-dessous

	1	2	3	4	5	6	7	8	9	10
B1	N	N	N	N	P	N	P	P	P	P
B2	N	P	P	P	N	P	N	N	N	N
B3	P	P	P	P	P	P	P	P	P	N

Question 10) En supposant que la boucle interne ait déjà été exécutée un certain nombre de fois, quelle est le nombre de bonnes prédictions pour les trois branchements avec

a) un prédicteur statique « toujours pris »
b) un prédicteur dynamique « 1 bit »
c) un prédicteur dynamique « 2 bits ».

Prédicteur statique

	1	2	3	4	5	6	7	8	9	10
B1	N	N	N	N	**P**	N	**P**	**P**	**P**	**P**
B2	N	**P**	**P**	**P**	N	**P**	N	N	N	N
B3	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	N

B1 : 5/10
B2 : 4/10
B3 : 9/10

Prédicteur 1 bit

	1	2	3	4	5	6	7	8	9	10
B1	N	N	N	N	P	N	P	P	P	P
P1	P	**N**	**N**	**N**	N	P	N	**P**	**P**	**P**
B2	N	P	P	P	N	P	N	N	N	N
P2	**N**	N	**P**	**P**	P	N	P	**N**	**N**	**N**
B3	P	P	P	P	P	P	P	P	P	N
P3	N	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	P

B1 : 6/10
B2 : 6/10
B3 : 8/10

Prédicteur 2 bits

	1	2	3	4	5	6	7	8	9	10
B1	N	N	N	N	P	N	P	P	P	P
P1	fp	**fn**	**FN**	**FN**	FN	**fn**	FN	fn	**fp**	**FP**
B2	N	P	P	P	N	P	N	N	N	N
P2	**FN**	fn	**FP**	**FP**	FP	**fp**	FP	fp	**fn**	**FN**
B3	P	P	P	P	P	P	P	P	P	N
P3	**fp**	**FP**	**FP**	**FP**	**FP**	**FP**	**FP**	**FP**	**FP**	FP

B1 : 6/10
B2 : 6/10
B3 : 9/10