TD 11 - Révisions

PROGRAMMATION ASSEMBLEUR ET EXECUTION PIPELINE

Soit les programmes assembleur MIPS32 de la Figure 1.
Les hypothèses sont les suivantes :
R1 contient initialement l'adresse de l'élément A[0] d'un tableau d'entiers 32 bits A [N].
R2 contient initialement l'adresse de l'élément B[0] d'un tableau d'entiers 32 bits B[N].

Programme P1	Programme P2
ADDI R5, R1, 512	ADDI R5, R1, 512
Loop : LW R3, (R1)	Loop : LW R3, (R1)
BLEZ R3, Suite	LW R4, (R2)
LW R4, 0(R2)	BLEZ R3,R0, Suite
ADD R4, R4, R3	ADD R4, R4, R3
SW R4,0 (R2)	SW R4, (R2)
Suite : ADDI R1, R1, 4	Suite : ADDI R1, R1, 4
ADDI R2, R2, 4	ADDI R2, R2, 4
BNE R1,R5, Loop	BNE R1,R5, Loop

Figure 1 : Programmes assembleur

Ecrire le programme C correspondant au programme assembleur P1 de la Figure 1.

On considère maintenant que les programmes assembleurs s'exécutent sur un processeur pipeliné.
Toutes les instructions entières ont une latence de 1, sauf l'instruction LW qui est pipelinable et a une latence de 2. Les branchements conditionnels ont une latence de 2 lorsque le branchement est pris (l'instruction qui suit le branchement est annulée) et de 1 lorsque le branchement est non pris.

Donner le nombre de cycles par itération de boucle pour le programme P1 lorsque le branchement BLEZ est pris et lorsqu'il n'est pas pris. Même question pour le programme P2. Dans les deux cas, on considérera les branchements BNE comme pris.

EXECUTION DE BOUCLES

Le processeur MIPS a 32 registres flottants F0 à F31 (F0 est un registre normal) et les instructions flottantes indiquées en figure 2.
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 4 cycles pour les instructions flottantes.
Les branchements ne sont pas retardés.

LWC1	2	LXC1 ft, déplac(rs)	ft ← MEM [rs +SIMM(déplac)]
SWC1	1	SWC1 ft, déplac(rs)	ft → MEM [rs + SIMM(déplac)]]
ADD.S	4	ADD.S fd, fs,ft	fd ← fs + ft (addition flottante simple précision)
MUL.S	4	MUL.S fd, fs,ft	fd ← fs * ft (multiplication flottante simple précision)
SUB.S	4	SUB.S fd, fs,ft	fd ← fs - ft (soustraction flottante simple précision)

Figure 2: Instructions flottantes MIPS

Q 1) On considère le programme de la Figure 3. Les tableaux X, Y et Z sont rangés successivement en mémoire. L'adresse de X[0] est initialement dans R3. Optimiser ce programme sans déroulage de boucle. Quel est son temps d'exécution (en nombre de cycles par itération) du programme optimisé ? Quel est son temps d'exécution total ?

```
float X[800], Y[800], Z[800], A;
int i ;
for (i=0; i<800; i++)
Z[i] = A* X[i] + Y[i];
```

```
        LWC1 F0, A              //F0 ← A
        ADDI   R5, R3, 3200
Boucle :LWC1    F1,0(R3)
        LWC1   F2,3200 (R3)
        MUL.S  F1,F1,F0
        ADD.S  F2,F2,F1
        SWC1    F2,6400(R3)
        ADDI    R3,R3,4
        BNE    R3,R5, Boucle
```

Figure 3 : Programme C P3 et programme assembleur

CACHES.

On suppose que le processeur utilisé a un cache données de 16 Ko, avec des blocs de 64 octets. Il utilise l'écriture simultanée (write through) non allouée. On rappelle qu'avec l'écriture simultanée, il n'y a pas de défauts de cache en écriture

Soit le programme suivant P4

```
#define N 128
double X[N], Y[N], Z[N] ;
for (i=0 ; i<N ; i++){
if (X[i] > Y[i]) Z[i] = X[i]+Y[i] ;
        else Z[i] = 0.0 ; }
```

On suppose que les tableaux X[N], Y[N], Z[N] sont rangés à partir de l'adresse hexadécimale 0x1000 0000.

Quelles sont les adresses de Y[0] et Z[0] ? , Y[][]

Quel est le nombre de bits pour l'adresse dans la ligne, l'index et l'étiquette

- En correspondance directe
- Avec associativité deux voies (deux lignes par ensemble)

Quel est le nombre total de défaut de caches lors de l'exécution du programme P3 pour les deux cas suivants : a) correspondance directe, b) associativité deux voies?

Pour quelle valeur minimale de N étant une puissance de 2 aura-t-on deux défauts de cache par itération avec la correspondance directe ?

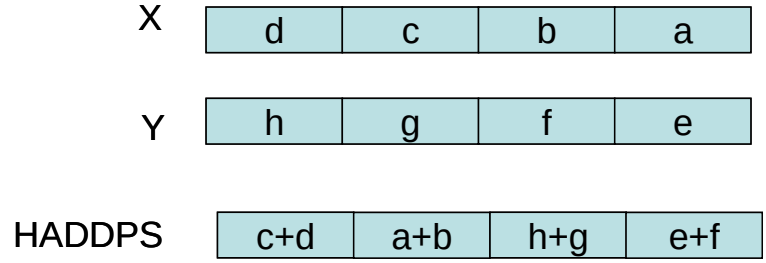
INSTRUCTIONS SIMD IA32

Soit le programme C suivant correspondant au produit matrice vecteur

```
Float X[N], Z[N], Y[N][N],S ;
for (i=0;i<N;i++) {
      S=0.0;
      for (j=0;j<N;j++)
            S+=X[j]*Y[i][j];
      Z[i]=S;
      }
```

Réécrire le programme avec un déroulage d'ordre 4 de la boucle j.

L'effet de l'instruction SIMD HADDPS sur des mots de 4 floats est défini par la figure suivante :



Comment peut-on effectuer l'addition horizontale (d+c+b+a) à l'aide de l'instruction HADDPS ?

Soit le programme SIMD ci-dessous

```
#define PADD(a,b) _mm_add_ps(a,b) // addition SIMD de 4 floats
#define PMUL(a,b) _mm_mul_ps(a,b) // multiplication SIMD de 4 floats
#define MOVSS(a,b)  _mm_store_ss(&a,b) // rangement float poids faible
                                 de b dans la variable a de type float
#define HADDPS(a,b) _mm_hadd_ps(a,b) // addition horizontale

typedef union VEC{
            float float_word[N];
      __m128 quad_word[N/4];

} VEC ;

VEC X,Z,Z2;
MAT Y;
float S;
__m128 SS;

for (i=0;i<N;i++) {
      SS=DUP4(0.0f);
```

```
for (j=0;j<N/4;j++)
      SS=PADD (SS, PMUL (X.quad_word[j],Y.quad_word[i][j]));
      SS=HADDPS(SS,SS); SS=HADDPS(SS,SS);
      MOVSS(Z2.float_word[i], SS);
}
```

Expliquer comment on passe du programme scalaire avec déroulage de 4 au programme SIMD.

EXECUTION D'INSTRUCTIONS

On utilise le jeu d'instructions MIPS

On considère que les registres du processeur contiennent les huit chiffres hexadécimaux suivants :

R0	0000 0000
R1	1234 5678
R2	7FFF FFFF
R3	4433 2211
R4	FFFF FFFF
R5	8123 0000

Q 2) Donner le contenu des registres R6 à R11 (sous forme de huit chiffres hexadécimaux) après exécution des instructions suivantes. Indiquer les cas de débordement

a) ADD R6, R1, R5
b) ADD R8, R2, R4
c) ADD R9, R1,R3
d) SUB R10, R3, R4

Q 3) Donner l'instruction ou la suite des instructions MIPS32 pour effectuer les actions suivantes :

a) Lorsque CP= 8000 0000H, sauter à l'adresse 8000 1004H
b) Lorsque CP=8000 0000H, sauter à l'adresse 9000 8000H
c) Diviser par 16 le contenu du registre R4, interprété en signé
d) Multiplier par 31 le contenu du registre R5