Partiel Architecture des Ordinateurs – 22 Octobre2018

Tous documents autorisés. Calculatrices interdites. Durée 2h

Une justification concise sera donnée lorsque c'est nécessaire. Lorsqu'un résultat est demandé en notation héxadécimale, la notation binaire ne sera pas acceptée.

1. Représentation des entiers sur 8 bits[4 pts]

Q1. On note # l'opération effectuée par un additionneur sur 8 bits. Pour les opérations suivantes, donner le résultat en notation **hexadécimale**, donner la retenue et indiquer si le résultat est égal à celui de l'opération arithmétique d'addition lorsque les opérandes sont interprétées en naturels ou en relatifs (compléments à deux), suivant le format de la table.

Opération	Résultat	Retenue (0/1)	Correct en naturel(Oui/Non)	Correct en relatifs (Oui/Non)
0x24 # 0x94	0xB8	0	Oui	Oui
0x80 # 0xFF	0x7F	1	Non	Non
0x7F # 0x01	0x80	0	Oui	Non
0x80 # 0x80	0x00	1	Non	Non

Q2. Donner la représentation en complément à deux sur 8 bits des nombres suivants : -10, 200, -1, -128.

-10 = 0xF6

200 : impossible

-1 = 0xFFFFFFFF

-128 = 0x80

2. Représentation des réels, norme IEEE754 [3 pts]

On considère le format IEEE754 simple précision.

Q3. Coder -11,5 (représentation **hexadécimale**). :0xC1380000

Q4. Donner la valeur (représentation décimale) des nombres réels suivants : 0x41200000, 0xBE000000.

0x41200000 = 1

0xBE000000 = -0.125

Q5. Effectuer les additions suivantes : 1.25 + 65, $0.125+2^{30}$

1.25 + 65 = 66.25

$0.125 +2^{30} = 2^{30}$

3. Jeu d'instructions MIPS [6 pts]

Q6. Donner le codage des instructions :

a) ADDU r1, r10, r2 → 0x01420821

b) LW r1, 12(r5) → 0x8CA1000C

c) SLL r3, r2, 10 → 0x00021A80

d) LUI r2, 0xABCD → 0x3C02ABCD

Q7. L'état initial des registres est : R1 = 0x85463215 et R2=0x0F0FF0F0.

Donner le contenu du registre R3 après l'exécution des instructions

a) ADDI R3,R2,0xF0FF

R3 = 0x0F0FE1EF

b) ANDI R3, R2, 0x0F0F

R3 = 0x00000000

c) SLT R3, R1, R2

R3 = 1

d) SRA R3, R1, 8

R3 = 0xFF854632

Q8. La mémoire est organisée en Big Endian. L'état de la mémoire est donnée par la Table 1. Le registre R2 contient 0x10000000. Donner l'**état final** de la mémoire après exécution de la séquence Seq 1.

Seq 1
LW R1, 0(R2) SLL R1, R1, 4 SB R1, 0(R2) SLL R1, R1, 2 SB R1, 1(R2) SLL R1, R1, 2 SB R1, 2(R2) SLL R1, R1, 2 SB R1, 3(R2)

Table 1		
Adresse	Valeur	Valeur
0x1000 0000	0x65	0x50
0x1000 0001	0xAB	0x40
0x1000 0002	0x12	0x00
0x1000 0003	0x55	0x00
0x1000 0004	0xFE	0xFE
0x1000 0005	0x65	0x65
0x1000 0006	0x15	0x15
0x1000 0007	0x4C	0x4C

4. Conditionnels et boucles [4 pts]

Q7. On supposera que les tableaux x et y sont implémentés en mémoire aux adresses 0x1000 0000 et 0x2000 0000. La variable s à l'adresse 0x1001 0000. Ecrire en assembleur MIPS32 le code de la boucle suivante :

```
int x[100];
int y[100];
```

```
int s=0 ;
for(int i=0 ; i<100 ; i++) {
 s += x[i] + y[99-i]
}
```

Solution simple :

```
LI R1, 0x10000000
LI R2, 0x20000000
ADDI R2,R2,396
ADDI R3, R3, 400
XOR R6,R6,R6
bcl :
    LW R4,0(R1)
    LW R5,0(R2)
    ADDU R4,R4,R5
    ADDU R6,R6,R4
    ADDI R1, R1, 4
    ADDI R2, R2, -4
    ADDI R3,R3,-4
    BEQ R3,R0,bcl
li R1,0x10010000
SW R6,0(R1)
```

5. Procédures [3 pts]

On considère le fragment de code suivant :

```
Lb1 :
        ADDU R3,R0,R1
        SLT R4,R2,R1
        BEQ R4,R0,suite
        ADDU R3,R0,R2
suite :
        JR R31
main :
        LI R1, 0x80001000
        LI R2, 0x50101000
        JAL Lb1
        SUBU R3,R3,R1
```

Q8. Donner la valeur du registre R3 après l'exécution de ce programme

R3=0

Q9. Que fait la fonction « Lb1 » ?

C'est la fonction « min »

Q10. Que faudrait-il rajouter dans le « main » pour pouvoir utiliser ce programme comme une procédure à part entière ?

Sauver le registre R31 ! Et jr R31