Département d'informatique **Module : ADO** **LicAca : S3**
UFA Sétif **Corrigé du Devoir** **Année 2012/2013**

Exercice 1

a-Le plus petit nombre positif correspond à la plus petite mantisse positive et au plus petit exposant :

- la plus petite mantisse :**1000000** soit **0.1 (0.5 <=m<1) ;**
- le plus petit exposant biaisé : **0000 =0 ;**
- le plus petit exposant réel = **0 - 2^3 =-8_{10}** (le biais=$2^{4-1} = 2^3$).

Le plus petit nombre réel positif est **$0.1 * 2^{-8} = 2^{-1} * 2^{-8} = 2^{-9}$.**

Le plus grand nombre positif correspond à la plus grande mantisse positive et au plus grand exposant :

- la plus grande mantisse = **1111111** soit **0.1111111_2 = 1.0000000-0.0000001=1-2^{-7}** ;
- le plus grand exposant biaisé: **$1111=15_{10}$** ;
- le plus grand exposant réel : **15_{10} -8_{10} = 7_{10}.**
- Le plus grand nombre réel positif est donc **:+(1-2^{-7}) *2^7 = 2^7-2^0= 128-1= 127_{10} .**

L'intervalle fermé sous la forme demandée :**[2^{-9} , 127] = [1* 2^{-9} , 127 * 2^0]**

b- X =AE8 et Y=9D0 .

En passant de l'hexadécimal au binaire et en prenant le biais =2^3 = 8 sans utiliser le bit caché ; on trouve**: X = -13 * 2^{-7} ; Y = -5_{10} * 2^{-8}**

Z = X – Y = (-5 * 2^{-8}) - (- 13 *2^{-7}) on normalise les exposants

= **(-5 * 2^{-8}) - (-26 * 2^{-8}) = 21 * 2^{-8}**

Exercice 2

Il ne peut jamais y avoir de débordement avec des opérandes de signe différents.

– Carry (C) : retenue sur le dernier bit.

– Débordement (V) : quand les opérandes sont de même signe et que le résultat est de signe différent.

la table de vérité

x	y	c	s	V	C
0	0	0	0	0	0
0	0	1	1	1	0
0	1	0	1	0	0
0	1	1	0	0	1
1	0	0	1	0	0
1	0	1	0	0	1
1	1	0	0	1	1
1	1	1	1	0	1

Les expression logique de C et V en fonction des bits de poids fort, des opérandes et du résultat de l'addition (les bits de poids forts nous renseignent sur le signe).

$$C = \bar{x}y\bar{s} + x\bar{y}\bar{s} + xy$$
$$V = \bar{x}\bar{y}s + xy\bar{s}$$

Exercice 3

Des exercices similaires sont faits en série 3.

Exercice 4

en écrivant -63 en C2 et en faisant l'addition on trouve : FFFFFFCA

alors le contenu de la mémoire est le suivant :

M[0x1FE] =0xFF , M[0x1FF] = 0xFF , M[0x200] = 0xFF , M[0x201] = 0xCA

Remarque :ce qui est entre crochets ce sont des adresses et sont obtenues en ajoutant 1 en Hexa : 1FF = 1FE + 1

200 = 1FF + 1

Exercice 5

(a) 10 bits d'adresse, 1024 mots et 8 bits par mot

Possible, raisonnable. Le nombre de mot mémoire correspondant exactement au nombre de possibilités d'adressage (2^{10} = 1024).

Taille = 1024 octets, soit 1Ko.

(b) 10 bits d'adresse, 1024 mots et 10 bits par mot

Possible, raisonnable. Mais on adresse que les mots (ce qui est suffisant à priori), pas les octets.

Taille = 1024 * 10 = 10240 bits, soit 1280 octets, soit 1,25Ko.

(c) 9 bits d'adresse, 1024 mots et 10 bits par mot

Impossible. 9 bits d'adresse ne fournissent que 2^9 = 511 possibilités, ce qui est insuffisant pour adresser tous les mots mémoires.

(d) 11 bits d'adresse, 1024 mots et 10 bits par mot

Possible, déraisonnable. La capacité d'adressage 2^{11} = 2048 est trop grande pour le nombre de mots. 1 bit d'adresse est inutilisé.

Taille = 1024 * 10 = 10240 bits, soit 1280 octets, soit 1,25Ko.

(e) 10 bits d'adresse, 10 mots et 1024 bits par mot

Possible, déraisonnable Le nombre de mot mémoire est vraiment petit et les mots sont très long. En outre, 4 bits seulement sont suffisants pour adresser 10 mots.

Taille = 10 * 1024 = 10240 bits, soit 1280 octets, soit 1,25Ko.

Exercice 6

calculer la somme des nombres naturels de p à q inclus.

```
          #$5 =q , $4 = p , $2 = som
            Add   $2 ,$0 , $0
boucle: bgt      $4, $5, fin    # si  p>q   aller à fin
           add      $2, $2, $4     # som = som+p
           addi     $4, $4, 1       #  p= p+1
             j         boucle
fin :
```

Exercice 7

écrire une suite d'instructions pour tester un nombre dans le registre $4 s'il est pair on met 1 dans le registre $5 sinon 0 (utiliser la division)

```
             addi   $3, $0 ,2
             div     $4, $3
             mfhi    $t0          # le reste de la division dans hi on le transfère dans $t0
             beq    $t0, $zéro, pair
             addi     $5, $0,0
              j  suite
pair         addi   $5,$0,1
suite :
```

Exercice 8 EMD1 2011 2012

a-Soit la séquence d'instructions MIPS suivante :

```
            add   $s4, $0 , $0
            add   $s3, $s1,$0
   etiq :    blt   $s3 ,$s2, fin
            sub   $s3, $s3,$s2
            addi  $s4,$s4, 1
             j      etiq
    fin :
```

Supposons que initialement $s1=11 $s2=3 donnez les valeurs que prennent $s3 et $s4 Au cours de l'exécution.

Réponse : 0.5 pts

$s3	**11 8 5 2**
$s4	**0 1 2 3**

Quelle est la fonction réalisée : **1pt**

Réponse :cette séquence d'instructions effectue la division entière du contenu du registre $S1 par le contenu du registre $S2 par des soustractions successives ; $S4 représente le quotient et $S3 représente le reste

b-Traduire le code machine de l'instruction suivante en code assembleur MIPS : 3908003F. **0.5 pts**

Réponse 3908003F= 00111001010001010001000000000111111

En examinant les 6 bits de poids fort c'est le codop de XORI ;c'est le type I , les 5bits qui suivent représentent le registre $t0 et et les 5 suivants aussi ; l'instruction sera alors :

XORI $t0 , $t0, 0*003F

Quelle est la fonction réalisée par cette instruction. **1 pt**

Réponse : cette instruction complémente les 6 bits de poids faible du registre $t0 ;
les autres bits restent inchangés le résultat sera mis dans $t0

c-Ecrire une suite d'instructions MIPS pour compter le nombre de 1 dans un mot de 32 bits ; supposer que le mot se trouve dans le registre $s1 et le compteur dans le registre $s2.

Réponse : 3 pts

Une solution possible serait la suivante :

```
            add   $S2 ,$zéro, $zéro      #mise à zéro du compteur de bits à1
            addi  $t4, $zéro, 32         #initialiser le compteur des bits du mot
 suivant :  beq   $t4 ,$zéro ,fin        #si tous les bits sont testés on termine
            andi  $t1 , $S1 , 1          #masquage( tous les bits à 0 sauf le bit 0)
            add   $S2,  $S2 , $t1        #le résultat= 1 ou 0 sera ajouté au compteur
            srl   $S1,  $S1 , 1          # décalage à droite pour tester le bit suivant
            subi  $t4,  $t4 , 1          #décrémenter le compteur de bits du mot
             j    suivant
  fin :
```