Contrôle I.S.N : Vendredi 15/02/13

Jean-Pierre Vallon

Représentation numérique de l'information :

1. Donner la représentation en machine de l'entiers relatif -7 sur 8 bits
2. Trouver la représentation décimale de l' entier relatif dont la représentation binaire sur 8 bits est 1111 1100
3. Comment est représenté le nombre entier 11 sur 64 bits ? et le nombre à virgule 11,0 sur 64 bits ?
4. (a) Décomposer la fonction booléenne f suivante définie par le tableau , avec les fonctions **non**, **et**, **ou**.

x	y	f(x,y)
0	0	1
0	1	1
1	0	0
1	1	1

   (b) Vérifier que l'expression obtenue peut se simplifier en : non(x) ou y

Programmation Java :

1. Tout nombre entier N peut s'écrire sous la forme $N = m \times 2^n$
   où $1 \leq m < 2$. On dit que n est le logarithme binaire de N
   Ecrire une fonction **récursive** qui retourne le logarithme binaire d'un entier N entré en paramètre.
2. (a) Ecrire une fonction Java inverseBit qui étant donné un mot de 4 bits entré en paramètre retourne un autre mot de 4 bits dont les bits sont contraires à ceux du mot entré en paramètre.
   (b) Compléter la classe suivante sur votre feuille pour tester la fonction inverseBit

```
class ControleIsn{
        static ..... inverseBit (.....){

                }
        public static void main(String[] args){
        //initialiser un mot mot1 de 4 bits
        //pour tester inverseBit
        //initialiser un mot mot2 de 4 bits
        //qui recevra le retour de inverseBit
        //appeler la fonction inverseBit
        //afficher mot2
        }
}
```

Représentation numérique de l'information :

1. On part de la représentation de 7 sur 8 bits c'est à dire 0000 0111 , on change chaque bit en son contraire on obtient 1111 1000 et on ajoute 1 on obtient 1111 1001.
2. 1111 1111 sur 8 bits représente -1 donc -2 est représenté par 1111 1110, -3 par 1111 1101 et -4 par 1111 1100.
3. l'entier 11 est représenté par

   0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 1011 sur 64 bits.

   0x000000000000000B en hexadécimal

   le nombre à virgule 11.0 est positif donc son bit de signe est 0

   Combien de 2 y-a-t-il dans 11 ?

   $11 = 1,375 \times 2^3$. Ce qui est l'équivalent de l'écriture scientifique en base 2.

   3 est l'exposant et la mantisse en machine n'est pas 1,375 mais 0,375 .

   Sur 64 bits l'exposant 0 a le représentant binaire associé à 1023 c'est à dire (sur 11 bits) 011 1111 1111, donc 3 sera représenté par 100 0000 0010.

   Il nous reste à coder 0,375 en binaire sur 52 bits.

   $0,375 \times 2 = 0,75$ donc le premier des 52 bits est 0

   $0,75 \times 2 = 1,5$ donc le second est 1 et on recommence avec 0,5.

   $0,5 \times 2 = 1$ donc le troisième est 1 et tous les autres valent 0.

   Donc en tout : 0100 0000 0010 0110 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 représente 11,0

   Donc en hexadécimal 0x4026000000000000
4. On utilise la méthode vue en classe :

   On introduit deux fonctions $g_1(x) = f(x,0)$ et $g_2(x,1) = f(x,1)$ donc $f(x,y) = mux(y, g_1(x), g_2(x))$.

   On constate dans le tableau que $g_1(x) = non(x)$ et $g_2(x) = 1$ donc f(x,y)=mux(y,non(x),1)

   = (non(y) et non(x)) ou (y et 1)

   Donc f(x,y) = (non(y) et non(x)) ou (y)

   Montrons que f(x,y) = non(x) ou y en faisant le tableau de non(x) ou y

x	y	non(x)	non(x) ou y
0	0	1	1
0	1	1	1
1	0	0	0
1	1	0	1

   On voit que les fonctions f et non(x) ou y ont les mêmes images pour les mêmes valeurs de x et y donc

   f (x,y) = non(x) ou y

Programmation Java

```
static  int logBin(int n){
//retourne le logarithme binaire d'un entier naturel n
                if(n < 2) return 0;
                else return 1+ logBin(n/2);
}



static  int[] inverseBit(int[] mot1){

                int[] mot2 = new int[4];
                for(int i = 0;i <= 3;i = i+1;){
                        if(mot1[i] == 0){
                                mot2[i] = 1;
                        }
                }
                return mot2;
}

class ControleIsn{

        static  int[] inverseBit(int[] mot1){

                int[] mot2 = new int[4];
                for(int i = 0;i <= 3;i = i+1;){
                        if(mot1[i] == 0){
                                mot2[i] = 1;
                        }
                }
                return mot2;
}
//*************** Main**********************

        public static void main(String[] args){

        //initialiser un mot mot1 de 4 bits
        //ce n'est qu'un exemple pour tester

        int[] mot1 = new int[4];

        mot1[0] = 0;
        mot1[1] = 1;
        mot1[2] = 0;
        mot1[3] = 1;
```

```
        //pour tester inverseBit
        //initialiser un mot mot2 de 4 bits
        //qui recevra le retour de inverseBit

        int[] mot2 = new int[4];

        //appeler la fonction inverseBit

        mot2 = inverseBit(mot1);

        //afficher mot1

        System.out.println("Voici le mot 1");
        for(int i = 3;i >= 0;i = i -1;){
                System.out.print(mot1[i]);
        }

        //afficher mot2

        System.out.println("Voici le mot 2");
        for(int i = 3;i >= 0;i = i -1;){
                System.out.print(mot2[i]);
        }
        }
}
```