NSI – 1ere	COURS Séquence 2-AB : Entiers relatifs en binaire	LFV

Dans le cours précédent nous avons vu comment représenter des nombres entiers *positifs* dans différentes bases. Nuos allons ici nous intéresser au cas des entiers *relatifs* mais uniquement en base 2 (binaire).

I. Représentation des entiers relatifs en complément à 2 (puissance n)

Dans cette première partie nous ne travaillerons qu'avec des octets.

1. Un peu de vocabulaire

Sur un octet (c'est-à-dire un ensemble de 8 bits), on a vu que les bits à droite correspondent aux petites puissances de 2 alors que les bits à gauche correspondent aux grandes puissances de 2. On dit donc que :

- le bit le plus à droite est le bit de poids faible,

- le bit le plus à gauche est le bit de poids fort.

2. Méthode naïve pour représenter les entiers relatifs

La méthode naïve est de rtéserver le bit de poids fort pour indiquer le signe. Bit de poids fort à zéro : nombre positif. Bit de poids fort à un : nombre négatif. Ainsi :

0`000 1001` correspond à `+9`

1`001 1001` correspond à `-9`

Cette méthode a un inconvénient majeur : si on utilisait cette représentation des entiers négatifs, l'addition de deux entiers poitif et négatif ne pourrait pas être effectuée comme celle de deux entiers positif et positif :

` 0000 1000`
`+0000 0110`
`-----------`
` 0000 1110`
	` 0000 1000`
	`+1000 0110`
	`-----------`
	` 1000 1110`

On a : `8 + 6 = 14`	On a malheureusement : `8 + (-6) = -14` !!!

Cette méthode naïve n'est donc pas satisfaisante car l'addition binaire est une opération effectuée très très très très souvent au cœur des processeurs. Il est donc primordial – pour des raisons de performances – que l'addition binaire soit effectuée le plus efficacement possible.

3. Représentation en complément à 2 (puissance n) pour les entiers relatifs

a) Interprétation 1 de cette représentation

La méthode du complément à 2 (qui signifie en réalité complément à 2 puissance n) sur n bits consiste à représenter un entier x négatif par la représentation de l'entier positif $2^n + x$ sur n bits.

Exemple : soit à représenter $x = -114$ sur $n = 8$ bits

On a : $2^n + x = 256 - 114 = 142 = 128 + 8 + 4 + 2 = 1000\ 1110_2$

Donc -114 se représente sous la forme `1000 1110` en complément à deux.

b) Interprétation 2 de cette représentation

La méthode du complément à 2 pour représenter un entier x négatif sur n bits peut aussi se voir comme la succession suivante d'opérations :

- Représenter le nombre positif associé en binaire (c'est-à-dire $-x$).
- Inverser tous les bits de cette représentation.
- Ajouter 1 à la nouvelle représentation obtenue.

Exemple : soit à représenter $x = -114$ sur $n = 8$ bits

$-x = 114 = 64 + 32 + 16 + 2 = 0111\ 0010_2$

Inversion : `1000 1101`

Ajout de 1 : `1000 1110`

c) Interprétation 3 de cette représentation

La méthode du complément à 2 pour représenter un entier x négatif sur n bits peut aussi se voir comme la succession suivante d'opérations :

- Représenter le nombre positif associé en binaire (c'est-à-dire $-x$).
- En partant du bit de poids faible (de la droite), inverser tous les bits situés *strictement après* le premier 1

Exemple : soit à représenter $x = -114$ sur $n = 8$ bits

$-x = 114 = 64 + 32 + 16 + 2 = 0111\ 0010_2$

On identifie ce qui est situé *strictement après* le premier 1 en partant de la droite : $\underline{0111\ 00}\ 10_2$

On inverse cela : `1000 1110`

4. Propriétés du complément à 2 (puissance n)

La première propriété de la représentation des entiers en complément à 2 est que *le bit de poids fort* (le plus à gauche) indique le signe de l'entier (on parle *de bit de signe*). Lorsque ce bit de signe est égal à 1 l'entier représenté est négatif et, a contrario, lorsque ce bit de signe est égal à 0 l'entier représenté est positif.

La seconde propriété est que la représentation en complément à 2 permet à l'addition binaire de fonctionner de façon similaire pour deux entiers positifs et pour un entier positif et un entier négatif :

`0000 1000`
`+0000 0110`
`-----------`
`0000 1110`
	`1111`
	`0000 1000`
	`+1111 1010`
	`-----------`
	`0000 0010`

On a : `8 + 6 = 14`	On a bien : `8 + (-6) = -2`

5. Passer de la représentation à l'entier en base 10

- Si le bit de signe est zéro, il suffit de le considérer comme un entier positif :

  `0101 0011` représente l'entier positif $64 + 16 + 2 + 1 = 83$

- Si le bit de signe est un**,** on a deux alternatives :

  Alternative 1 : on le considère comme un entier positif PUIS on soustrait 2^n :

  `1101 0101` représenterait l'entier positif $128 + 64 + 16 + 4 + 1 = 213$

  ce qui correspond à l'entier relatif $213 - 256 = -43$

  Alternative 2 : on inverse les bits situés strictement à gauche du premier 1 en partant de la droite :

  `1101 0101` conduit à `0010 1011`

  ce qui correspond à $32 + 8 + 2 + 1 = 43$ soit -43.

II. Nombre de bits utilisés

1. Le cas des octets

Votre connaissance parfaite des cours précédents vous permet de savoir immédiatement que sur un octet (8 bits) on peut coder $2^8 = 256$ valeurs (si on code les entiers positifs on va de 0 à 255).

Si on représente les entiers relatifs en complément à deux , on peut représenter :

- les entiers positifs de 0 [`0000 0000`] à 127 [`0111 1111`]
- les entiers négatifs de -128 [`1000 0000`] à -1 [`1111 1111`]

Ainsi sur 8 bits, en complément à 2 on peut représenter les entiers de -2^7 à $2^7 - 1$. Plus généralement :

Nombre de bits	Plage des entiers positifs	Plage des entiers relatifs en complément à deux
8	0 à $255 = 2^8 - 1$	$-128 = -2^7$ à $127 = 2^7 - 1$
16	0 à $65\,535 = 2^{16} - 1$	$-32768 = -2^{15}$ à $32767 = 2^{15} - 1$
32	0 à $4\,294\,967\,295 = 2^{32} - 1$	$-2\,147\,483\,648 = -2^{31}$ à $2\,147\,483\,647 = 2^{31} - 1$
64	0 à $18\,446\,744\,073\,709\,551\,616 = 2^{64} - 1$	$-9\,223\,372\,036\,854\,775\,808 = -2^{63}$ à $9\,223\,372\,036\,854\,775\,807 = 2^{63} - 1$
k	0 à $2^k - 1$	-2^{k-1} à $2^{k-1} - 1$

2. Impact de l'addition et de la multiplication sur le nombre de bits utilisés

La règle à retenir est la même que pour le nombre de chiffres utilisés en base 10.

a) Pour l'addition

Soit K le nombre de chiffres à utiliser pour écrire le résultat d'une somme S de deux nombres écrits avec k chiffres.
En se rappelant des additions vues à l'école primaire (voir ci-contre), il est clair que K est égal à au plus $k + 1$.

```
 111     11
  12 345 789
+ 99 854 123
 -----------
 112 199 912
```

C'est la même chose en binaire. Soit deux nombres entiers relatifs x_1 et x_2 représentés sur k bits.

> Soit S la somme de deux entiers représentés sur k bits.
> Alors S peut se représenter sur au plus $(k + 1)$ bits

$-2^{k-1} \leq x_1 \leq 2^{k-1} - 1$
$-2^{k-1} \leq x_2 \leq 2^{k-1} - 1$
En faisant la somme des termes de gauche, du milieu et de droite :
$-2^{k-1} - 2^{k-1} \leq x_1 + x_2 \leq 2^{k-1} - 1 + 2^{k-1} - 1$
$-2^k \leq x_1 + x_2 \leq 2^k - 2$ (en utilisant le fait que $2 \times 2^{k-1} = 2^k$)
Ainsi $x_1 + x_2$ peut se représenter avec au plus $k + 1$ bits.
En corollaire de cette propriété on pourrait parler avec enthousiasme des dépassement de capacité (que se passe-t-il sur une machine en 32 bits si on obtient une somme sur 33 bits ?) …

b) Pour la multiplication

Soit K le nombre de chiffres à utiliser pour écrire le résultat d'un produit de deux nombres écrits avec p et q chiffres.
En se rappelant que $10^p \times 10^q = 10^{p+q}$ on a aisément l'intuition que $K \approx p + q$.

C'est la même chose en binaire. Soit deux entiers relatifs x_1 et x_2 représentés sur p et q bits.
$-2^{p-1} \leq x_1 \leq 2^{p-1} - 1$
$-2^{q-1} \leq x_2 \leq 2^{q-1} - 1$
En faisant le produit on obtient (cela demande un peu de réflexion) :

$$-2^{p-1}.2^{q-1} + 2^{\max(p,q)-1} \leq x_1.x_2 \leq 2^{p-1}.2^{q-1}$$
$$-2^{p+q-2} \leq x_1.x_2 \leq 2^{p+q-2} \leq 2^{p+q-1} - 1$$

Bref au vu de ce calcul on en déduit (à cause du terme de droite) qu'il faut au plus $p + q$ bits pour représenter le produit.

> Soit P le produit de deux entiers représentés sur p et q bits.
> Alors P peut se représenter sur au plus $(p + q)$ bits.

III. Exercices

Exercice 1 :

Coder les entiers suivants en complément à deux sur un octet :

- 117
- -87
- -55
- -12
- 0
- 84
- -128
- 127
- -118
- 49

Exercice 2 :

Donner la valeur des entiers représentés en complément à 2 par les octets ci-dessous :

- `1001 0011`
- `1010 1010`
- `0110 0110`
- `0000 0010`
- `1111 1111`
- `1000 0000`
- `0111 1111`
- `1010 0101`

Exercice 2 :

Quelle est la plage d'entiers relatifs que l'on peut coder en complément à 2 sur 12 bits ?

Exercice 3 :

Essayez d'effectuer les additions binaires suivantes.
(Les additions en binaire seront vues en cours ultérieurement.)

` 0010 1000`
`+0011 0110`
`-----------`
	` 0000 1010`
	`+1111 1110`
	`-----------`
		` 1010 1000`
		`+0001 1011`
		`-----------`
			` 0110 1001`
			`+1000 1010`
			`-----------`

Exercice 4 :

Dans chacun des cas, évaluez le nombre de bits nécessaires pour coder en complément à 2 les entiers x_1 et x_2 indiqués puis le nombre de bits nécessaires pour coder les entiers $x_1 + x_2$ et $x_1 \times x_2$.

- $x_1 = 456$ et $x_2 = 34$
- $x_1 = 567\,456$ et $x_2 = -765$
- $x_1 = -234\,567\,786\,763\,456$ et $x_2 = 123\,456\,789\,987\,654\,321$