Chapitre 3

Arithmétique entière

1 Représentation des entiers relatifs sur 8 bits :

Pour les entiers positifs compris entre 0 et 127 (2^7-1), la représentation est le codage simple en binaire (restes pris dans l'ordre inverse des divisions euclidiennes successives de l'entier positif par 2 jusqu'au quotient nul, on complémente par des zéros à gauche si nécessaire). Pour les entiers négatifs compris entre -128 et -1, on utilise les 7 bits restants pour coder les entiers de 128 à 255, il suffit alors de prendre la convention d'ajouter 256 ($= 2^8$) à l'entier négatif :

- 19 s'écrit $16 + 2 + 1 = 2^4 + 2^1 + 2^0$ soit 00010011
- -19 s'obtient en ajoutant 256 au nombre : 256 - 19 = 237 s'écrit $2^7 + 2^6 + 2^5 + 2^3 + 2^2 + 2^0$ soit en binaire sur 8 bits 11101101

Le bit de gauche (poids fort) sert à déterminer le signe de l'entier.
Le bit de droite est appelé le bit de poids faible.

2 Limitation en taille :

Sur 16 bits (respectivement 32) le plus grand entier représentable en machine est $2^{15} - 1 = 32767$ (respectivement 2^{31}-1 = 2147483637).

3 Dépassement de capacité et validité du résultat :

A cause de la notation binaire en complément à 2, les calculs sont effectués modulo la taille de l'intervalle défini par le plus petit entier 10000000 et le plus grand 01111111, soit 65536 en amplitude sur 16 bits.
On peut disposer d'entiers long, mais le problème demeure pour des valeurs élevées. En C, les "short" occupent 2 octets, les "long" 4 octets.

Exemple : sur 4 bits

0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	-8
1001	-7
1010	-6
1011	-5
1100	-4
1101	-3
1110	-2
1111	-1

Le complément à 2 représente le plus de nombres (0 à sa propre représentation). Il conduit à l'algorithme d'addition le plus simple.

En base 2, l'addition repose sur les égalités :
$0 + 0 = 0$
$0 + 1 = 1 + 0 = 1$
$1 + 1 = 10$ (le 1 à gauche du résultat est dû à la retenue).

Exemple 1011 + 1101 :

$$\begin{array}{rr} & {\scriptstyle 1111} \\ & 1011 \\ + & 1101 \\ \hline & 11000 \end{array}$$

Test de la validité du résultat (sur 4 bits) :
Il faut que le bit de poids faible de la somme du bit de poids fort de chaque terme de la somme à laquelle on rajoute la retenue (si elle existe) soit égal au bit de poids fort du résultat, sinon, l'addition n'est pas valable et il y a un dépassement de capacité.

$$\begin{array}{rrr} & 0111 & (7) \\ + & 1100 & (-4) \\ \hline & 0011 & (3) \end{array}$$

Ici, on a $0 + 1 + 1 = 10$, donc le bit de poids faible de cette somme (0) est égal au bit de poids fort du résultat (0) et l'addition est correcte.

$$\begin{array}{rrr} & 1011 & (-5) \\ + & 1101 & (-3) \\ \hline & 1000 & (-8) \end{array}$$

Ici, on a $1 + 1 + 1 = 11$, donc le bit de poids faible de cette somme (1) est égal au bit de poids fort du résultat (1) et l'addition est valable.

$$\begin{array}{rrr} & 1101 & (-3) \\ + & 1010 & (-6) \\ \hline & 0111 & (7) \end{array}$$

Dans ce cas, nous avons $1 + 1 + 1 = 11$, le bit de poids faible de cette somme (1) est différent du bit de poids fort du résultat (0) donc l'addition est fausse.

3.1 Notation

$B = b_{n-1} \cdots b_0$ un nombre binaire sur n bits. La valeur correspondant à B dépend de la représentation choisie :

V : $\{0,1\}^n \to Z \subset \mathbb{Z}$

R : $Z \to \{0,1\}^n$ son inverse

Notons

— $V_b(B)$, $R_b(N)$ les applications associées aux entiers non signés (écriture dans le système de numérotation binaire).

— $V_2(B)$, $R_2(N)$ les applications associées aux entiers signés en complément à 2.

Si $N \geq 0$ on note $R_1(N)$ le complément logique de $R_b(N)$ c'est-à-dire la différence binaire, **1** - $R_b(N)$ où **1** est le nombre binaire de n bits tous égaux à 1.

Les entiers en complément à 2 sont obtenus de la façon suivante :

— Un entier $N \in \left[0, 2^{n-1} - 1\right]$ a la représentation non signée : $R_2(N) = R_b(N)$

— Un entier négatif -$N \in \left[-2^{n-1}, -1\right]$ a la représentation non signée de $2^n - N$: $R_2(-N) = R_b(2^n - N)$

<u>Remarque</u> :

Le bit de poids fort (rang $n-1$) est à 1 si et seulement si l'entier est négatif. On a :

$$\begin{aligned} (1)\ R_2(N) &= R_b(N)\ ssi\ N \in \left[0, 2^{n-1} - 1\right] \\ &= R_b(2^n + N) = R_1(N+1)\ ssi\ N \in \left[0, 2^{n-1} - 1\right] \end{aligned}$$

avec :

$R_1(N+1)$ complément logique de $R_b(N+1)$

De plus,

$$(2)\ V_2(B) = V_b(B) - b_{n-1}2^n\ \ avec\, B\ =\ b_{n-1} \cdots b_0$$

avec :

$V_2(B)$ valeur décimale du nombre binaire codé en représentation en complément à 2.

$V_b(B) = \displaystyle\sum_{0 \leq i \leq n-1} b_i 2^i$

<u>Définition</u> :

Soient B et B' deux nombres binaires. On appelle <u>somme binaire</u> de B et B', notée $B + B'$, le nombre binaire obtenu en ne conservant que les n premiers bits de la <u>somme en base 2</u> de B et B'. On note $z(B + B')$ la retenue de la somme $B + B'$, le $(n+1)^{eme}$ bit de la somme en base 2 de B et B'.

Remarque :
On a la relation suivante :

$$(3)\ V_b(B+B') = V_b(B) + V_b(B') - 2^n z(B+B')$$

3.2 Proposition

Soient $B = R_2(N)$ et $B' = R_2(N')$ 2 entiers en complément à 2. Le complément à 2 de la somme, $B'' = R_2(N+N')$ est donné par :
$B'' = B + B'$ avec validité du résultat ssi :

$$b''_{n-1} = (z(B+B') + b_{n-1} + b'_{n-1})_0$$

où $(X)_0$ désigne le bit de poids faible de X.

démonstration :
Posons $x_0 = (z(B+B') + b_{n-1} + b'_{n-1})_0$.
Si $N = 0$ ou $N' = 0$, c'est trivial.

Supposons donc les deux non nuls, il y a 3 cas selon le signe.

1^{er} cas : N, $N' > 0$
$b_{n-1} = b'_{n-1} = 0$ d'où $z(B+B') = 0$ et $x_0 = 0$
On a $N + N' = V_b(B) + V_b(B') \overset{(3)}{=} V_b(B+B')$
On en déduit d'après (2) :

$$N + N' = V_2(B+B') + b''_{n-1}2^n$$

la somme est correcte ssi $b''_{n-1} = 0$ d'où $b''_{n-1} = x_0$

2^e cas : N, $N' < 0$
$b_{n-1} = b'_{n-1} = z(B+B') = x_0 = 1$
On a :

$$\begin{aligned} N + N' &= V_b(B) - 2^n + V_b(B') - 2^n \\ &= V_b(B) + V_b(B') - 2^{n+1} \\ &\overset{(3)}{=} V_b(B+B') + 2^n \underbrace{z(B+B')}_{=1} - 2^{n+1} \\ &= V_b(B+B') - 2^n \end{aligned}$$

d'après (2)

$$N + N' = V_2(B+B') + (b''_{n-1} - 1)2^n$$

la somme est correcte ssi $b''_{n-1} = 1$ d'où $b''_{n-1} = x_0$

3^e cas : $N > 0$, $N' < 0$

On a $b_{n-1} = 0$ et $b'_{n-1} = 1$. Calculons $N + N'$.

$$N + N' = V_b(B) + V_b(B') - 2^n \overset{(3)}{=} V_b(B + B') + (z(B + B') - 1)2^n$$

d'après (2) :

$$N + N' = V_2(B + B') + (b''_{n-1} + z(B + B') - 1)2^n$$

la somme est correcte ssi $b''_{n-1}+z(B+B')-1 = 0$ c'est-à-dire $b''_{n-1}+z(B+B') = 1$

Si $z(B + B') = 0$, on a $x_0 = 1$ et $b''_{n-1} = 1$, d'où $b''_{n-1} = x_0$
Si $z(B + B') = 1$, on a $x_0 = 0$ et $b''_{n-1} = 0$, d'où $b''_{n-1} = x_0$

4 Autres opérations binaires

En base 2, la soustraction repose sur les égalités :
0 - 0 = 0
1 - 0 = 1 et 1 - 1 = 0
0 - 1 = 1 (emprunt 1)
Au cours de l'opération on ajoute au premier terme (0) de la somme 10 (en base 2) et un emprunt de 1 est reporté sur la position suivante (à gauche) du deuxième terme (1) de la somme.
Exemple 101111 - 11011 :

$$\begin{array}{rr} & 1_1 01111 \\ - & {}_1\,11011 \\ \hline & 0\ 10100 \end{array}$$

En base 2, la multiplication repose sur les égalités :
$0 \times 0 = 0$
$0 \times 1 = 1 \times 0 = 0$
$1 \times 1 = 1$
Exemple 1011×11 :

$$\begin{array}{rr} & 1011 \\ \times & 11 \\ \hline & 1011 \\ & 1011\ \, \\ \hline & 100001 \end{array}$$

Remarque : il s'agit d'une succession d'addition et de décalage.