Codage des nombres

Introduction : Le chapitre codage-système de numérotation a traité le codage des nombres entiers naturels. Dans cette partie, nous allons voir comment coder les nombres réels et les nombres entiers négatifs.

Codage des nombres réels

En base 10, le nombre 145,32, à titre d'exemple, est égal à :

$$145{,}32 = 1 \times 10^2 + 4 \times 10^1 + 5 \times 10^0 + 3 \times 10^{-1} + 2 \times 10^{-2}$$

De la même façon, le nombre 111,011 en base 2, est égal à :

$$(111{,}011)2 = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}$$

$$= 1 \times 4 + 1 \times 2 + 1 \times 1 + 0 \times \frac{1}{2} + 1 \times \frac{1}{4} + 1 \times \frac{1}{8}$$

$$= 4 + 2 + 1 + 0 + 0{,}25 + 0{,}125$$

$$= 7{,}375$$

De la même façon, le nombre 4C,4E en base 16, est égal à :

$$(AC,4E)_{16} = 10 \times 16^1 + 12 \times 16^0 + 4 \times 16^{-1} + 14 \times 16^{-2}$$

$$= 160 + 12 + 4 \times 0{,}0625 + 14 \times 0{,}00390625$$

$$= 160 + 12 + 0{,}25 + 0{,}0546875$$

$$= 172{,}3046875$$

Conversion réel décimal en base 2

Pour la partie entière :

- On utilise la méthode de la division entière comme pour les entiers

Pour la partie fractionnaire :

- On multiplie la partie fractionnaire par 2
- On note la partie entière obtenue
- On recommence cette opération avec la partie fractionnaire du résultat et on arrête quand la partie fractionnaire devient nulle ou quand la précision souhaitée est atteinte.
- La suite des parties entières obtenues dans l'ordre de leur calcul est la partie fractionnaire dans la base 2

Exemple 1 : Convertir le nombre décimal 18,6875 en binaire

- 18 en base 10 s'écrit 10010 en base 2
- 0,6875 en base 10 s'écrit en base 2:

$$0{,}6875 \times 2 = 1{,}375 = 1 + 0{,}375$$

$$0{,}375 \times 2 = 0{,}75 = 0 + 0{,}75$$

$$0{,}75 \times 2 = 1{,}5 = 1 + 0{,}5$$

$$0{,}5 \times 2 = 1 = 1 + 0$$

$$(8{,}6875)_{10} = (10010{,}1011)_2$$

Exemple 2 : Convertir le nombre décimal 34,3 en binaire

- 34 en base 10 s'écrit 100010 en base 2
- 0,3en base 10 s'écrit en base 2:

$$0{,}3 \times 2 = 0{,}6 = 0 + 0{,}6$$

$$0{,}6 \times 2 = 1{,}2 = 1 + 0{,}2$$

$$0{,}2 \times 2 = 0{,}4 = 0 + 0{,}4$$

$$0{,}4 \times 2 = 0{,}8 = 0 + 0{,}8$$

$$0{,}8 \times 2 = 1{,}6 = 1 + 0{,}6$$

$$0{,}6 \times 2 = 1{,}2 = 1 + 0{,}2$$

....

$(34{,}3)_{10} = (100010{,}010011)$ si on se contente de cette précision.

Exemple 3 : Convertir le nombre décimal 172,3046875 en hexadécimal

- 172 en base 10 s'écrit AC en base 16
- 0,3046875 en base 10 s'écrit en base 16 :

$$0{,}3046875 \times 16 = 4{,}875 = 4 + 0{,}875$$

$$0{,}875 \times 16 = 14 = 14 + 0$$

Comme le chiffre 14 est en hexadécimal E, le nombre décimal 172,3046875 s'écrit en hexadécimal :

$$(172{,}3046875)_{10} = (AC{,}4E)_{16}$$

Codage binaire de réels en virgule flottante: Le standard IEEE 754

Introduction

Le standard IEEE 754 est la norme la plus employée particulièrement par l'unité de calcul en virgule flottante des processeurs (FPU : floating-point unit) pour effectuer des opérations sur des nombres à virgule flottante. Il a été mis au point en 1985 par l'IEEE (Institute of Electrical and Electronics Engineers). La norme définit quatre formats pour représenter des nombres à virgule flottante en base 2 :

- Le format simple précision
- Le format simple précision (float)

- Le format double précision (double)
- Le format double précision étendue

Le format double précision est pratiquement le seul qui soit utilisé actuellement.

Stockage d'un nombre flottant

D'une façon générale, un nombre flottant est formé de trois éléments : la mantisse, l'exposant et le signe et s'écrit :

$$n = s \times 2e \times m$$

- s désigne le bit de signe
- e désigne l'exposant
- m désigne la mantisse

Pour déterminer s, e et m, il faut :

- Ecrire le nombre en binaire abstraction faite de son signe
- Décaler la virgule vers la gauche jusqu'à ce qu'il ne reste qu'un 1 sur sa gauche en multipliant par pour chaque décalage
- Déterminer e
- S = 0 si le nombre est positif et à 1 si le nombre est négatif.

L'exposant peut être positif ou négatif. Cependant, la comparaison entre les nombres flottants dans la représentation complément à 2 est une opération relativement difficile. Pour contourner cette difficulté, l'exposant est décalé de telle sorte que le stockage du nombre soit sous forme d'un nombre non signé.

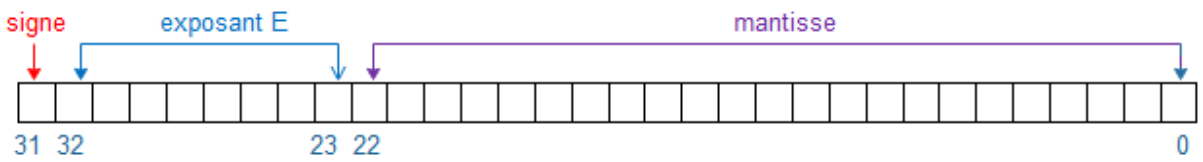
Si l'on appelle e le nombre de bits de l'exposant, le décalage est donné par :

$$D = 2^{e-1} - 1$$

- Le bit de poids fort est le bit de signe. Il est mis à 0 si le nombre est positif et mis à 1 si le nombre est négatif.
- Les e bits suivants représentent l'exposant décalé
- Les m bits de poids faible suivants représentent la mantisse

Format simple précision (32 bits)

En simple précision, le nombre flottant est mémorisé dans un mot de 32 bits (1 bit de signe, 8 bits exposant, 23 bits mantisse. Le format simple précision est indiqué ci-dessous.



Le décalage D est donc donné par :

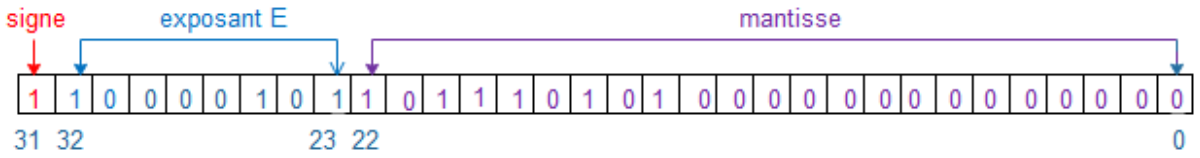
$$D = 2^{8-1} - 1 = 2^7 - 1 = 128 - 1 = 127$$

$$E = e + 127$$

Exemple : Nous nous proposons de coder le nombre décimal – 110,625 en format simple précision

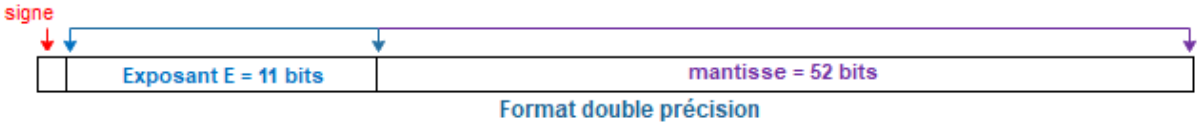
- Le signe s = 1 (Puisque le nombre est négatif)
- Le nombre sans le signe s'écrit en binaire 1101110,101
- Après décalage de la virgule vers la gauche, on obtient $1{,}101110101 \times 2^6$
- La mantisse est la partie à droite de la virgule et complété avec des 0 pour obtenir 23 bits. Donc m = 10111101000000000000000
- L'exposant e = 6 et E = 6 + 127 = 133 en décimal et 10000101 en binaire

On a donc -110,625 qui s'écrit sous le format suivant :



Format double précision (64 bits)

En double précision, le nombre flottant est mémorisé dans un mot de 64 bits (1 bit de signe, 11 bits d'exposant, 52 bits de mantisse). Le format simple précision est indiqué ci-dessous.



Format double précision

Pour le format double précision, le décalage D est donné par :

$$D = 2^{11-1} - 1 = 2^{10} - 1 = 1024 - 1 = 1023$$

$$E = e + 1024$$

Exemple : Nous nous proposons de coder le nombre décimal 64 en format double précision

- Le bit signe s = 0 puisque le nombre à coder est positif
- Le nombre décimal 64 s'écrit en binaire 1000000 = $1{,}0 \times 2^6$
- L'exposant e = 6 et E = e + D = 6 + 1023 = 5 + 1024 soit E en binaire E = 10000000101

On obtient donc comme codage du nombre 64 0 10000000101
00……0

Soit le format suivant :

Représentation du nombre 64 en format double de précision