III Représentation approximative des nombres réels

Nous avons vu comment représenter les entiers, naturels ou relatifs. Intéressons nous maintenant aux nombres réels, que nous appelons ***flottants*** en informatique.

1) Ecriture d'un nombre flottant en base deux

En base 10 , le nombre 61,154 est sous forme décomposée $6*10^1+1*10^0+1*10^{-1}+5*10^{-2}+4*10^{-3}$

De même, en base deux 1101,101 signifie $1*2^3+1*2^2+0*2^1+1*2^0+1*2^{-1}+0*2^{-2}+1*2^{-3}=13,625$

Il est plus difficile de passer de la base 10 à la base 2.

Exemple : Ecrire 61,154 en base 2.

Nous n'avons aucun problème à écrire 61 en base deux : 111101

Comment exprimer 0,154 en base 2 ?

0,154 *2 =0.308	0
0.308*2=0.616	0
0.616*2=1.232	1
0.232*2=0.464	0
0.464*2=0.928	0
0.928*2=1.856	1
0.856*2=1.712	1
0.712*2=1.424	1
0.424*2=0.848	0 etc….

On obtient donc 61,154 =111101,00100111…. en base 2 .

Exercices

1) Trouver la représentation décimale de 1101101,011

2) Trouver la représentation binaire de 24,625

Remarques importantes : En base 10, 61,154 = $6,154*10^1$ et 0.0061154= $6.154*10^{-3}$

Il en va de même en base 2 : 1101,1101= $1,1011101*2^{11}$, l'exposant 11 correspondant à un décalage de 3 vers la droite de la virgule.

De même $0.0011=1,1*2^{-11}$, l'exposant -11 correspondant à un décalage de 3 vers la gauche.

Un nombre à développement décimal fini ne l'est pas forcément en base 2.

Exercices

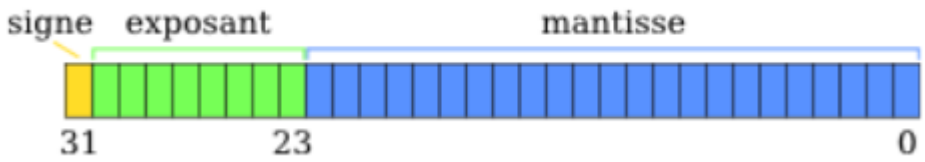
1. Montrer que 0.1 n'admet pas de développement fini en base 2.
2. Déterminer un décimal avec trois chiffres après la virgule qui admet un développement fini en base 2. Donner ce développement

2) Représentation en machine d'un nombre flottant

Les machines utilisent une norme pour coder les nombres réels : La Norme IEEE 754. Pour cela , le nombre devra préalablement être écrit sous la forme $1,XXXXX.2^e$(avec e l'exposant)

- Le nombre est codé sur 32 bits
- Le bit de poids le plus fort sert à coder le signe (0 pour positif, 1 pour négatif)
- L'exposant est codé sur les 8 bits consécutifs au signe.
- La mantisse est codée sur les 23 bits restants et correspondant aux bits situés après la virgule.

La principale difficulté vient du codage de l'exposant : En effet, il nous faut pouvoir coder les exposants négatifs. Nous avons 8 bits et pouvons alors coder théoriquement 256 valeurs. Deux valeurs étant réservées, il nous reste 254 valeurs, que l'on fait varier de -126 à 127. Pour que l'exposant soit toujours positif, on ajoute 127 à la valeur de l'exposant.



Exemple : Codons 1101,10110110.

Etape 1 : On transforme l'écriture pour se conformer à la norme IEEE 754 : $1,101101110110*2^{11}$

Etape 2 : L'exposant à coder est 127+3 = 130 =10000010. Le signe est positif dans le bit le plus à gauche sera 0.

On obtient 01000001010110111011000000000000

Exercices

1) Donner la représentation en machine sur 32 bits de 214,875. De 1/32.

2) Quelle est l'écriture décimale de 11011010101101111000000000000000 ? Quelle est l'écriture hexadécimale de ce nombre ?

Remarque : Le codage sur 32 bits est appelé « simple précision ». Il existe aussi le codage sur 64 bits appelé « double précision ». L'exposant est alors codé sur 11 bits et le décalage est de 1023 et non plus 127.