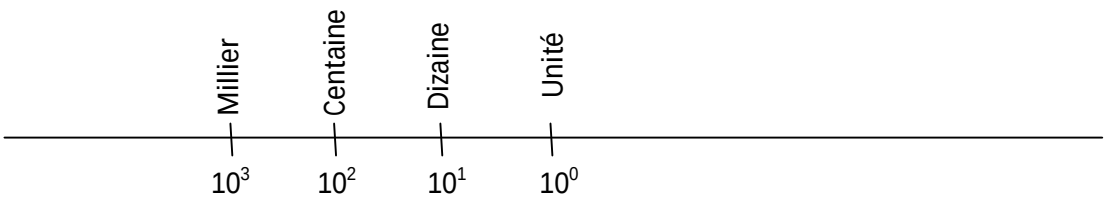
Système de numération

Table des matières

1. Les différentes bases et types de nombres

1.1. La base 10 – décimale

Il est plus facile pour comprendre le mécanisme des bases de bien comprendre la base 10.
Décimal signifie en base 10 (DEC) donc 10 chiffres de 0 à 9.
Après on utilise des multiples de la base à la puissance de leurs poids (rang).



Exemple de décomposition

$$421 = 400 = 4 \times 10^{2}$$
$$\quad\;\; 20 = 2 \times 10^{1}$$
$$\quad\;\; 1 = 1 \times 10^{0}$$

Pour les nombres à virgule (les réels) on procède de la même manière.

$$12.5 = 10 = 1 \times 10^{1}$$
$$\quad\;\; 2 = 2 \times 10^{0}$$
$$\quad\;\; 0.5 = 5 \times 10^{-1}$$

1.2.La base de 2 – binaire (BIN, 0xb, %)

Ici le bit (binary digit) ne peut prendre que 2 éléments 0 ou 1, au-delà il faut utiliser des multiples de la base (2) élevés à la puissance de leur poids.

On appelle :
MSB le bit de poids le plus fort (Most Significant Bit)
LSB le bit de poids le plus faible (Less Significant Bit)

Exemple de décomposition

$$110 = 1 \times 2^{2}$$
$$\quad\;\; 1 \times 2^{1}$$
$$\quad\;\; 0 \times 2^{0}$$

A lui seul, le MSB représente 4/7 du nombre maxi

Le bit étant très petit, on l'associe en différents formats :

Le quartet :	4 bits	(nibble en anglais).
L'octet :	8 bits	(byte en anglais).
Le doublet :	16 bits.	
Le quadlet :	32 bits.	
L'octlet :	64 bits.	

D'après la IEC 60027-2, le kilooctet vaut 10^{3} et non plus 1024.

1.3.La base de 16 – Hexa décimal (HEX. 0x..,$)

Dans cette base, nous avons besoin de 16 éléments soit de 0 à 9 et 6 éléments supplémentaires A, B, C,D , E et F ayant le poids respectifs de 10,11,12,13,14 et 15.

Exemple.

$$2A = 2 \times 16^{1}$$
$$\quad\;\; A \times 16^{0}$$

On remarque l'aspect compact de l'écriture hexadécimal

$$13_{(10)} = 0000\ 1101_{(2)} = D_{(16)}$$

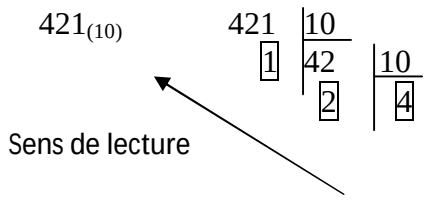
2. Conversion décimal vers bases

Afin de réaliser la conversion d'un nombre décimal vers une base quelconque il faut :

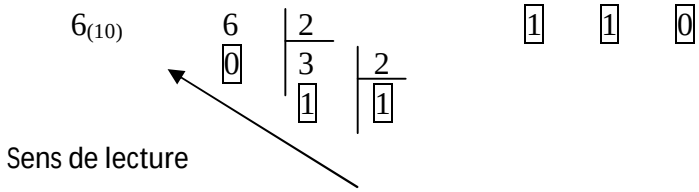
- Diviser par la base la partie entière
- Multiplier par la base la partie décimale

Puis on garde le reste et on effectue de nouveau l'opération sur le quotient ou sur la partie décimale si nécessaire.

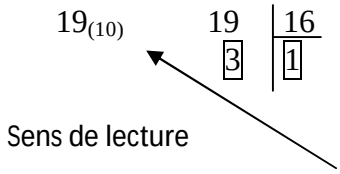
Exemple base 10 → 10



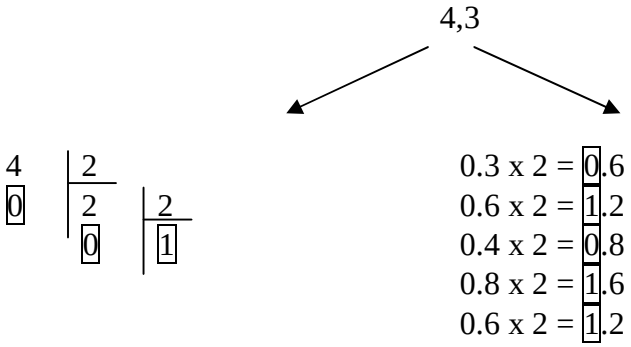
Exemple base 10 → 2



Exemple base 10 → 16



Exemple base 10 → 2



On s'arrête car cela recommence.

$4{,}3_{(10)} = 1\ 0\ 0\ ,0\ 1\ 0\ 1\ 1$

Le problème est comment faire la virgule (voir § 6 - les réels).

3. Conversion base vers décimal

On exprime le nombre en fonction des multiples de la base à la puissance du rang.

$$X = a_n . b^n + a_{n-1} . b^{n-1} + \ ... + a_0 . b^0$$

Exemple 2 → 10

$$1010 = 1.2^3 + 0.2^2 + 1.2^1 + 0.2^0$$

Ecrire les $0.b^n$ ne sert pas à grand-chose si ce n'est de ne pas se tromper dans le rang du poids.

Exemple 2 → 16

$$30A = 3.16^2 + 0.16^1 + A.16^0$$

4. Conversion entre bases

4.1.Binaire vers hexa

On convertit le mot binaire par quartet (paquet de 4). On obtient ainsi directement la valeur en hexa (2^4=16).

00101010 → 0010 1010 → 0x2A
2 A

11010110 →1101 0110 → 0xD6
D 6

8	4	2	1		
0	0	1	0	→	2 soit 2
1	0	1	0	→	10 soit A
8	4	2	1		
1	1	0	1	→	13 soit D
0	1	1	0	→	6 soit 6

4.2.Décimal vers binaire

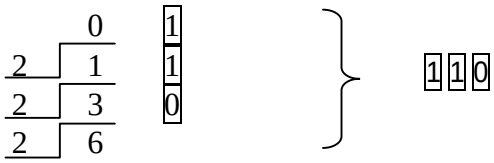
On remplit un tableau avec les poids des différents bits d'un octet par exemple.

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

On teste du MSB jusqu'au LSB afin de composer le nombre

Exemple sur un quartet 8 4 2 1
On cherche 5 0 1 0 1
8>5 on ne prend pas
4≤5 on garde reste 1
2>1 on ne prend pas
1≤1 on garde

On peut également modifier l'écriture de la division



5. Les nombres négatifs – Le compléments à 2

On va se servir du bit le plus fort (MSB) afin d'indiquer le signe du nombre binaire (signe).

Si le MSB = 1 alors le signe est négatif « - »
Si le MSB = 0 alors le signe est positif « + »

Exemple

5 → 0101 ; -5 → 1101

Dans cette représentation, il existe 2 zéros donc on oublie mais on garde le principe du bit MSB pour le signe.

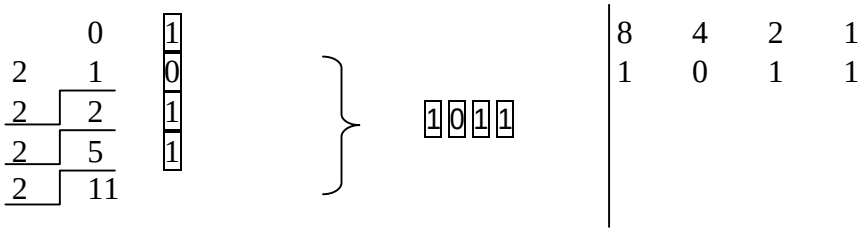
Le complément à 2 permet de remédier à ce problème.

5.1.Conversion d'un nombre décimal négatif en binaire A2

Si on désire un nombre sur n bits, il faut traduire en binaire le résultat de l'équation suivante.

$B = 2^n - |X|$, ici B est le nombre décimal à traduire, |X| la valeur absolue du nombre négatif.
Exemple sur un quartet X = - 5

$B = 2^n - |X| = 2^4 - |-5| = 11$



2 méthodes mais – 5 →1011 (A2)

5.2.Conversion d'un nombre décimal positif en binaire A2

Rien de particulier à faire si ce n'est traduire en binaire. 5 → 0101 $_{(A2)}$

5.3.Conversion d'un nombre binaire A2 en décimal

A - Extraire le bit de signe.
B - Complémenter les bits restants.
C – Ajouter 1 (effectuer l'addition rappel 1+1 = 0 retenue 1)
D – Traduire le résultat de l'addition en décimal
E – Ajouter le signe

Exemple

1	0	1	1 $_{(A2)}$		
	0	1	1	1	signe « – » car MSB = 1
	1	0	0		Complémentation
+			1		
	1	0	1		Additionner +1
	1	0	1	→ 5	Traduire le résultat de l'addition

On rajoute le signe donc - 5.

6. Les réels – Floating point conversion

Afin de représenter les nombres réels, on utilise le bit de signe, la conversion divisionnaire et donc l'exposant.

6.1.Exposant d'un nombre.

Il existe au moins deux types de notations d'exposant :

- Exposant scientifique → multiple de 3 (0, 3, -3, …)
- Exposant normalisé

On ramène le nombre réel à un chiffre réel et le reste en décimal.

Pour se faire, on écrit le nombre associé à sa base puissance 0 et on décale la virgule en incrémentant ou décrémentant la puissance.

0.000145	$0.000145x10^{0}$	$0.145\ x10^{-3}$	$1.45\ x10^{-4}$
3162.75	$3162.75x10^{0}$	$3.16275x10^{3}$	$3.16275x10^{3}$
0.17	$0.17x10^{-0}$	$0.17x10^{-0}$	$1.7x10^{-1}$
101.010	$101.010x2^{0}$	$101010x2^{-3}$	$1.01010x2^{2}$

6.2.Format du flottant (FLOAT)

Les flottants peuvent être utile sur les nombres trop grand ou s'il y a une partie décimale.

D'après la IEEE 754 le format est le suivant

Signe	exposant biaisé	Mantisse ou significante
1 bit	8 bits	23 bits

6.3.Définitions

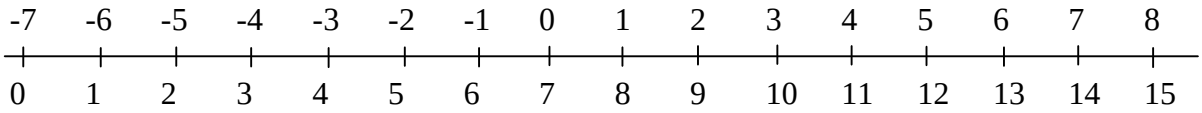
- Mantisse : valeur approchée de la partie décimale d'un nombre réel. 1.125 → 0.125
- Exposant biaisé : L'exposant normalisé sera positif ou négatif. Afin de ne coder que des exposants positifs, on va décaler (biaisé) la valeur de l'exposant à convertir.

$$e_b = e_n + (2^{k-1} - 1)$$

Exposant à coder en binaire

Exposant normalisé

k : le nombre de bit sur lequel on code l'exposant biaisé



Exemple sur 3 bits
$1.0101 \times 2^{-3}_{(2)}$

donc k = 3
exposant normalisé = -3
$e_b = -3 + [2^{(3-1)}-1]$
$e_b = -3 + 2^2-1$
$e_b = 0_{(10)}$
$e_b = 000_{(2)}$

Exemple sur 8 bits
$1.10111 \times 2^{4}_{(2)}$

donc k = 8
exposant normalisé = 4
$e_b = 4 + [2^{(8-1)}-1]$
$e_b = 131_{(10)}$
$e_b = 1000\ 0011_{(2)}$

6.4.Conversion décimal → flottant

A – Convertir la valeur absolue du nombre en binaire, valeur décimale si besoin

B – Ajouter x b^0 à la fin pour indiquer le type et le rang

C – Normalisé en décalant la virgule ← + → -

D – Placer la matisse d'après le format retenu

E – Convertir l'exposant normalisé en exposant biaisé $e_b = e_n + (2^{k-1} - 1)$

F – Mettre le bit de signe

Exemple avec le format sur 8 bits du nombre 4,75

1	3	4
Signe	exposant	mantisse

A) 4 , 75

4	2		0.75 x 2 = 1.5
0	2	2	0.5 x 2 = 1
	0	1	

B) 100.11×2^0

C) 1.0011×2^2

D) [.] [.][.][.] [0][0][1][1]

E) $e_b = 2 + [2^{(3-1)}-1] = 2 + 3 = 5 \rightarrow 101_{(2)}$

F) Le signe est positif donc le bit de signe est à 0

En résumé

[0] [1][0][1] [0][0][1][1]

6.5.Conversion flottant → décimal

Dans l'autre sens, on procède de même.
11010011, on identifie le format

[1]	[1][0][1]	[0][0][1][1]
-	exposant	mantisse

On calcule l'exposant normalisé $e_n = e_b - (2^{k-1} - 1) = 5 - 3 = 2$
On écrit la mantisse avec un 1 devant 1.0011
On écrit la forme développée (avec l'exposant) → 1.0011×2^2
On écrit le nombre binaire avec l'exposant 2^0 → 100.11×2^2
On convertit la partie réelle et décimale sans oublier pas le signe.

$100.11 \rightarrow 1x2^2+0x2^1+0x2^0. 1x2^{-1}+1x2^{-1} \rightarrow -4.75$