

Numération et conversion



Comment représenter un nombre décimal dans d'autres systèmes de numération ?

Système de numération

Pour compter des objets et les représenter par des nombres, on utilise des « systèmes de numération ».

Un système de numération est composé de trois éléments :

- ⇨ Base du système
- ⇨ Symbole
- ⇨ Poids du symbole

En informatique et électronique, les systèmes de numération indispensables sont :

- ⇨ Décimal
- ⇨ Binaire
- ⇨ Hexadécimal

Système décimal

Le système décimal ou base 10 est le système que nous utilisons tous les jours. La création de ce système découle d'un choix naturel : Dictée par le nombre des doigts des deux mains.

Les caractéristiques du système décimal sont :

- ⇨ Base : 10
- ⇨ Symboles : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9
- ⇨ Poids : 10

Exemple avec le nombre 5 816 :

	Poids Fort			Poids Faible
Symbole	5	8	1	6
Poids	10^3	10^2	10^1	10^0
Valeur du poids	1 000	100	10	1
Valeur de chaque symbole	5 000	800	10	6

On peut écrire : $5 \times 10^3 + 8 \times 10^2 + 1 \times 10^1 + 6 \times 10^0$

Ainsi on en déduit : $5 \times 1\,000 + 8 \times 100 + 1 \times 10 + 6 \times 1$

On a donc : $5\,000 + 800 + 10 + 6 = 5\,816$

En mathématique : $N^0 = 1$

Système binaire

Le système binaire ou base 2 est le système utilisé par les ordinateurs pour faire des calculs et communiquer.

Ce système utilise deux symboles appelés BIT (Binary digIT)

Les caractéristiques du système binaire sont :

- ⇨ Base : 2
- ⇨ Symboles : 0 et 1
- ⇨ Poids : 2

Exemple avec le nombre binaire : %1101

	Poids Fort			Poids Faible
Symbole	1	1	0	1
Poids	2^3	2^2	2^1	2^0
Valeur du poids	8	4	2	1
Valeur de chaque symbole	8	4	0	1

On peut écrire : $1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

Aussi, on en déduit : $1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1$

On a donc en base 10 : $(1101)_2 = (8 + 4 + 0 + 1)_{10} = (13)_{10}$

Le bit de poids le plus faible dans un nombre binaire se nomme le LSB (Least Significant Bit).

Le bit de poids le plus fort dans un nombre binaire se nomme le MSB (Most Significant Bit).

Un nombre binaire de 8 bits se nomme un octet.

Pour faciliter la lecture, on écrit un nombre binaire par bloc de 4 séparé par un espace.

Exemple, le nombre binaire 1100101011101101010 s'écrira : 0110 0101 0111 0110 1010

Système hexadécimal

Le système hexadécimal ou base 16 est un système inventé pour l'électronique numérique et l'informatique. Il offre un compromis entre la base 2 (binaire) et la base 10 (décimal).

Les caractéristiques du système hexadécimal sont :

- ⇨ Base : 16
- ⇨ Symboles : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E et F
- ⇨ Poids : 16

Exemple pour le nombre hexadécimal $B18D.

	Poids Fort			Poids Faible
Symbole	B	1	8	D
Poids	16^3	16^2	16^1	16^0
Valeur du poids	4 096	256	16	1
Valeur de chaque symbole	45 056	256	128	13

Valeurs des symboles alphabétiques :

A	10
B	11
C	12
D	13
E	14
F	15

On peut écrire : $11 \times 16^3 + 1 \times 16^2 + 8 \times 16^1 + 13 \times 16^0$

Aussi, on en déduit : $11 \times 4\,096 + 1 \times 256 + 8 \times 16 + 13 \times 1$

On a donc en base 10 : $(B18D)_{16} = (45\,056 + 256 + 128 + 13)_{10} = (45\,453)_{10}$

Table Binaire/Hexadécimal/Décimal

Décimal		Hexadécimal	Binaire				
0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	1
0	2	2	0	0	0	1	0
0	3	3	0	0	0	1	1
0	4	4	0	0	1	0	0
0	5	5	0	0	1	0	1
0	6	6	0	0	1	1	0
0	7	7	0	0	1	1	1
0	8	8	0	1	0	0	0
0	9	9	0	1	0	0	1
0	0	A	0	1	0	1	0
1	1	B	0	1	0	1	1
1	2	C	0	1	1	0	0
1	3	D	0	1	1	0	1
1	4	E	0	1	1	1	0
1	5	F	0	1	1	1	1
1	6	10	1	0	0	0	0
1	7	11	1	0	0	0	1
1	8	12	1	0	0	1	0
1	9	13	1	0	0	1	1
2	0	14	1	0	1	0	0
2	1	15	1	0	1	0	1
2	2	16	1	0	1	1	0
2	3	17	1	0	1	1	1
2	4	18	1	1	0	0	0
2	5	19	1	1	0	0	1
2	6	1A	1	1	0	1	0
2	7	1B	1	1	0	1	1
2	8	1C	1	1	1	0	0

Binaire vers décimal

Pour convertir un nombre binaire en décimal, on écrit le poids de chacun de ses chiffres, on multiplie chaque chiffre par son poids et enfin on additionne le tout.

Si l'on souhaite convertir %1 1001 0110 en décimal :

⇨ Ecrire le poids de chacun de ses chiffres

1	1	0	0	1	0	1	1	0
2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
256	128	64	32	16	8	4	2	1

⇨ On multiplie chaque chiffre par son poids

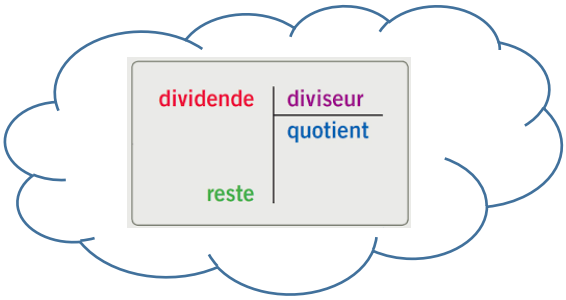
1	1	0	0	1	0	1	1	0
256	128	64	32	16	8	4	2	1
256	128	0	0	16	0	4	2	0

⇨ On additionne le tout

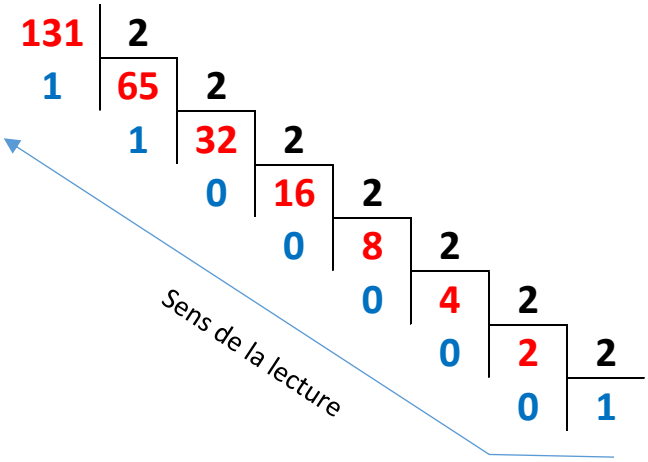
$$\%1\ 1001\ 0110 = (256 + 128 + 16 + 4 + 2)_{10} = (406)_{10}$$

Décimal vers binaire

Pour convertir un nombre décimal en binaire, on divise le nombre à convertir par 2 jusqu'à que le quotient est strictement inférieur à 2. Le résultat est donné en lisant le dernier quotient et les restes de la dernière vers la première division.



Si l'on souhaite convertir 131 en binaire :



On obtient donc le résultat suivant :

$$(131)_{10} = \%10000011$$

Autre technique :

On souhaite convertir 185 en binaire.

J'écris le tableau suivant :

128	64	32	16	8	4	2	1

Pour chaque valeur je test si le nombre rentre.

Est-ce que 128 rentre dans 185 ? Oui, alors le premier bit est 1. Je fais 185 - 128 = 57.

Est-ce que 64 rentre dans 57 ? Non, alors le second bit est 0.

Est-ce que 32 rentre dans 57 ? Oui, alors le troisième bit est 1. Je fais 57 – 32 = 25.

Est-ce que 16 rentre dans 25 ? Oui, alors le quatrième bit est 1. Je fais 25 – 16 = 9.

Est-ce que 8 rentre dans 9 ? Oui, alors le cinquième bit est 1. Je fais 9 – 8 = 1.

Est-ce que 4 rentre dans 1 ? Non, alors le sixième bit est 0.

Est-ce que 2 rentre dans 1 ? Non, alors le septième bit est 0.

Est-ce que 1 rentre dans 1 ? Oui, alors le huitième bit 1. 1 – 1 = 0.

185 = 1011 1001

Binaire vers hexadécimal

La conversion du binaire en hexadécimal est la raison même de l'utilisation du système hexadécimal. Il suffit de grouper le nombre binaire par groupe de 4 en partant du **LSB**.

Si l'on souhaite convertir %10110100001001111010111

⇨ On groupe le nombre binaire par groupe de 4

					1	0	1	1	0	1	0	0	0	1	0	0	1	1	1	1	0	1	0	1	1	1

⇨ On met la valeur du poids en dessous

					1	0	1	1	0	1	0	0	0	1	0	0	1	1	1	1	0	1	0	1	1	1
4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1

⇨ On calcule le résultat de chaque bit en multipliant la ligne 1 par la ligne 2

					1	0	1	1	0	1	0	0	0	1	0	0	1	1	1	1	0	1	0	1	1	1
4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1
0	0	0	0	0	2	0	8	4	0	1	0	0	0	1	0	0	2	1	8	4	0	1	0	4	2	1

⇨ On additionne chaque quartet (groupe de 4 bits)

					1	0	1	1	0	1	0	0	0	1	0	0	1	1	1	1	0	1	0	1	1	1
4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1
0	0	0	0	0	2	0	8	4	0	1	0	0	0	1	0	0	2	1	8	4	0	1	0	4	2	1
0			2				13				1				3				13				7			

⇨ On remplace les nombres supérieurs à 9 par leur équivalent en hexadécimal

					1	0	1	1	0	1	0	0	0	1	0	0	1	1	1	1	0	1	0	1	1	1
4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1	8	4	2	1
0	0	0	0	0	2	0	8	4	0	1	0	0	0	1	0	0	2	1	8	4	0	1	0	4	2	1
0			2				13				1				3				13				7			
0			2				D				1				3				D				7			

On obtient donc le résultat suivant :

$$\%10110100001001111010111 = \$2D13D7$$