Représentation des nombres

Un système d'écriture des nombres se décrit toujours en précisant les symboles utilisés (l'alphabet) et les règles d'association de ces symboles.
Ainsi dans notre système de numération décimale, l'alphabet est constitué de dix symboles : les dix chiffres 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9. À l'aide de ces dix symboles, on peut écrire n'importe quel nombre entier.

Le nombre quarante-sept s'écrit 47 en base 10 ce qui signifie que ce nombre est égal à $47 = 4 \times 10^1 + 7 \times 10^0$.

De même le nombre 3010 s'écrit ainsi :

$3010 = 3\times10^3 + 0\times10^2 + 1\times10^1 + 0\times10^0$.

Représentation des nombres entiers en base 2

Dans un système numérique (informatique), tout nombre est représenté par un ensemble d'élément binaires appelés «bits » (**B**inary dig**IT**)
La valeur ou la fonction prise par chacun de ces bits ne peut être que **0** ou **1**. Ainsi un bit peut représenter une valeur mais aussi un signe, une erreur, un changement de valeur, une parité, …
Chaque bit est affecté d'un poids binaire

MSB : Most Significant Bit : Bit de poids fort
LSB : Least Significant Bit : Bit de poids faible

Les puissances de 2

MSB **LSB**

Bn		**B10**	**B9**	**B8**	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
2^n	…	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	…	**1024**	**512**	**256**	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**

Ex : $28_{dec} = 1.2^4 + 1.2^3 + 1.2^2 + 0.2^1 + 0.2^0 = 11100_{bin}$

$10111_{bin} = 23_{dec}$

n variables d'entrée : 2^n combinaisons de sortie

Écriture binaire

L'écriture binaire d'un nombre entier s'appuie sur la décomposition de cet entier en somme de puissances de 2 distinctes.

Voici ce qu'il en est pour 47 et 3010.

$47 = 32 + 8 + 4 + 2 + 1$

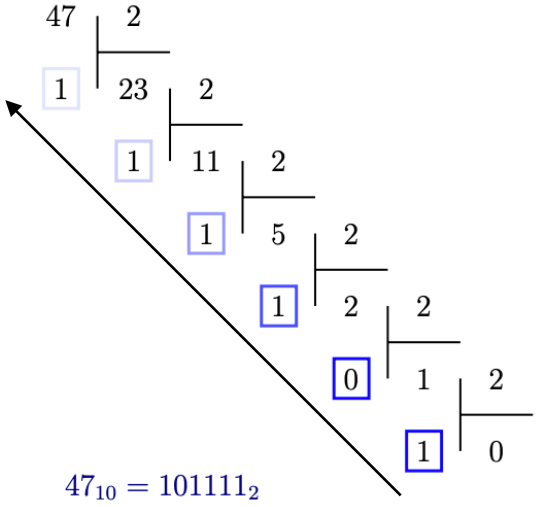
$= 2^5 + 2^3 + 2^2 + 2^1 + 2^0$

$= 1\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 1\times2^1 + 1\times2^0$

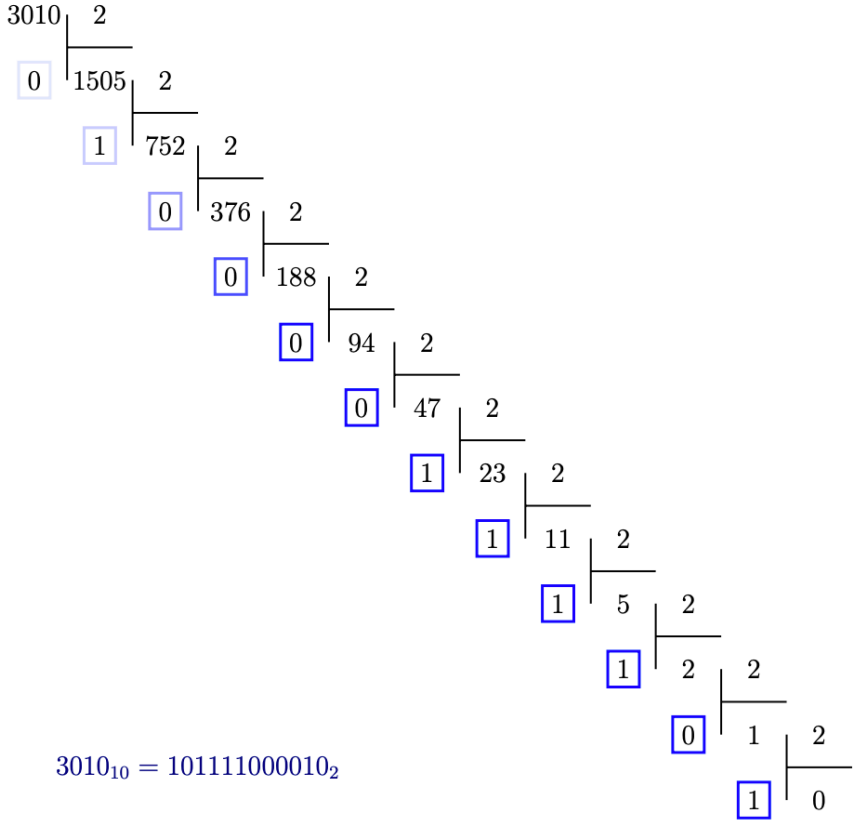
En utilisant les deux chiffres 0 et 1,

ce nombre s'écrit donc en binaire :

$47 = 101111_2$.



Pour 3010 : 3010 = 2048 + 512 + 256 + 128 + 64 + 2 = 1011110000010_2



$3010_{10} = 101111000010_2$

Voici les écritures binaires des entiers de 0 à 7.

$$0_{10} = 0_2$$

$$1_{10} = 1_2$$

$$2_{10} = 10_2$$

$$3_{10} = 11_2$$

$$4_{10} = 100_2$$

$$5_{10} = 101_2$$

$$6_{10} = 110_2$$

$$7_{10} = 111_2$$

Bits et octets

Les chiffres binaires 0 et 1 sont appelés ***bits***. Ce mot est la contraction de l'expression anglaise *binary digit*. Les entiers de 0 à 7 sont les huit seuls entiers qui peuvent s'écrire en binaire avec trois bits seulement (en complétant par des bits nuls à gauche si nécessaire).
Un ***octet*** est un nombre qu'on peut écrire en binaire sur huit bits. Ce sont les entiers compris entre 0 et 255. Ils sont au nombre de 256.

Remarque : Le mot anglais *byte* désigne ce qu'en français on nomme octet. Donc byte $\neq$ bit.

Mesures de quantité de mémoire

Longtemps les préfixes *kilo*, *méga*, . . . , appliqués aux octets pour mesurer des tailles de mémoire désignaient des puissances de 2. Or pour toute autre mesure dans le système international, ces mêmes préfixes désignent des puissances de 10. Une partie de la confusion vient que $2^{10} = 1024$ est proche de 1000. Pour remédier à cela, la commission électronique internationale (IEC) a spécifié de nouveaux préfixes pour les puissances de 2.

Nom	Puissance	Abbréviation	Nom	Puissance	Abbréviation
kilooctet	10^3	ko	kibioctet	2^{10}	Kio
megaoctet	10^6	Mo	mébioctet	2^{20}	Mio
gigaoctet	10^9	Go	gibioctet	2^{30}	Gio
téraoctet	10^{12}	To	tébioctet	2^{40}	Tio
pétaoctet	10^{15}	Po	pébioctet	2^{50}	Pio

Exemple :

Soit une clé USB d'une capacité de stockage de 32 Go (réellement Gio). Le préfixe « giga » est un multiplicateur décimal qui vaut 10^9 mais s'il est bien adapté au calcul décimal, il l'est moins au calcul binaire car il ne correspond pas à une puissance entière de 2.

1 Gio = …… octets

32 Gio = …… octets

Les bases 8 et 16

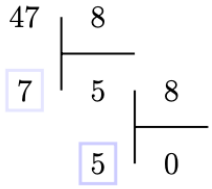
Les écritures décimales et binaires des nombres entiers ne sont pas les seules utilisées. En informatique deux autres bases sont plus ou moins fréquemment employées :

- la base 8 utilisant les huit chiffres de 0 à 7 ; on parle alors d'écriture ***octale*** des nombres ;
- la base 16 utilisant les dix chiffres usuels de 0 à 9 et les six premières lettres de l'alphabet de A à F ; on parle alors d'écriture hexadécimale des nombres.

L'octal

Déterminons les écritures octales des nombres 47 et 3010. Pour cela il suffit d'écrire chacun de ces deux nombres comme somme de puissances de 8.

Une méthode pour le faire est de diviser le nombre à écrire par la base autant de fois que nécessaire.



$47_{10} = 57_8$

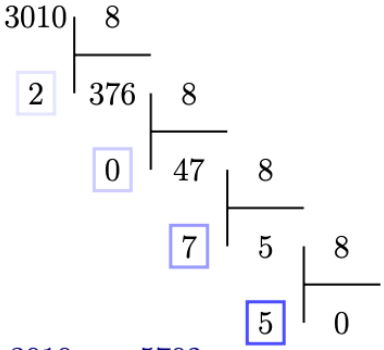
Pour 47 :

$47 = 5\times8^1 + 7\times8^0$ —> $47 = 57_8$

Pour 3010 : En prenant les restes successifs des divisions qui précèdent, on trouve

$3010 = 5 \times 8^3 + 7 \times 8^2 + 0 \times 8^1 + 2 \times 8^0$

$= 5702_8$.



$3010_{10} = 5702_8$

L’hexadécimal

$0_{10} = 0_{16}$

$1_{10} = 1_{16}$

$2_{10} = 2_{16}$

$3_{10} = 3_{16}$

$4_{10} = 4_{16}$

$5_{10} = 5_{16}$

$6_{10} = 6_{16}$

$7_{10} = 7_{16}$

$8_{10} = 8_{16}$

$9_{10} = 9_{16}$

$10_{10} = A_{16}$

$11_{10} = B_{16}$

$12_{10} = C_{16}$

$13_{10} = D_{16}$

$14_{10} = E_{16}$

$15_{10} = F_{16}$

$16_{10} = 10_{16}$

$17_{10} = 11_{16}$

$18_{10} = 12_{16}$

$19_{10} = 13_{16}$

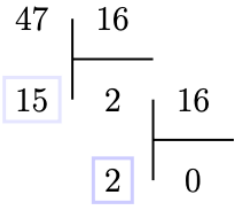
$20_{10} = 14_{16}$

Déterminons les écritures hexadécimales des nombres 47 et 3010. Pour cela il suffit d'écrire chacun de ces deux nombres comme somme de puissances de 16 avec la technique des divisions successives :

• Pour 47

$47 = 16\times2 + 15 = 2 \times 16^1 + 15 \times 16^0$.

$47 = 2F_{16}$.

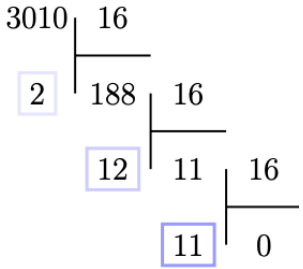


$47_{10} = 2F_{16}$

• Pour 3010

$3010 = 11 \times 16^2 + 12 \times 16^1 + 2 \times 16^0$.

$3010 = BC2_{16}$.



$3010_{10} = BC2_{16}$

Intérêt de ces bases

Pour quelle raison les informaticiens accordent-ils une attention particulière à ces deux bases ?

Tout simplement parce que $8 = 2^3$ et $16 = 2^4$, et comme on l'a vu, tout nombre compris entre 0 et 7 peut-être écrit en binaire à l'aide de trois bits, et tout entier n'excédant pas 15 a une écriture binaire sur quatre bits.

Cette remarque a pour conséquence que le passage du binaire à l'octal ou le contraire peut-être obtenu sans calcul arithmétique, et de même pour les conversions binaire/ hexadécimal.

Pour passer du binaire à l'octal, il suffit de regrouper les bits par paquets de 3 en commençant par la droite, puis de convertir chacun de ces paquets en un chiffre octal. Voici un exemple avec $n = 101111_2$

$$n = \overbrace{101}^{5}\overbrace{111}^{7} = \overline{57}_8.$$

On est passé de l'écriture binaire de n à son écriture octale sans aucun calcul.

De même pour passer du binaire à l'hexadécimal, on regroupe les bits par paquets de quatre.

$$n = \overbrace{0010}^{2}\overbrace{1111}^{F} = \overline{2F}_{16}.$$

Notez dans ce dernier exemple l'ajout de deux bits nuls en tête pour avoir un paquet de quatre bits complet.

Pour le passage de l'octal au binaire, ou de l'hexadécimal au binaire, c'est encore plus simple puisqu'il suffit de traduire chacun des chiffres de la représentation par un paquet de trois ou quatre bits selon la base de départ.
En revanche, comme 16 n'est pas une puissance de 8, il n'y a pas de moyen direct de passer de l'octal à l'hexadécimal ou le contraire. Il faut se servir de l'écriture binaire comme passage intermédiaire.

L'octal et l'hexadécimal sont donc utilisés comme moyen commode d'écriture du binaire. L'écriture octale est trois fois plus courte que l'écriture binaire, et l'écriture hexadécimale quatre fois plus courte.

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

caractère	code ASCII	code Hexa
Espace	32	20
!	33	21
"	34	22
#	35	23
$	36	24
%	37	25
&	38	26
'	39	27
(	40	28
)	41	29
*	42	2A
+	43	2B
,	44	2C
-	45	2D
.	46	2E
/	47	2F

caractère	code ASCII	code Hexa
0	48	30
1	49	31
⋮	⋮	⋮
9	57	39
:	58	3A
;	59	3B
<	60	3C
=	61	3D
>	62	3E
?	63	3F
@	64	40
A	65	41
B	66	42
⋮	⋮	⋮
Z	90	5A

caractère	code ASCII	code Hexa
[	91	5B
\	92	5C
]	93	5D
^	94	5E
_	95	5F
`	96	60
a	97	61
b	98	62
⋮	⋮	⋮
z	122	7A
{	123	7B
|	124	7C
}	125	7D
~	126	7E
T. supp.	127	7F

Exercices

Exercice 1 (Écritures)

1. Écrire en binaire le nombre $n = 2024$.

2. Déterminer les écritures octale et hexadécimale de ce nombre de deux façons différentes.

Exercice 2 (Pair ou impair ?)
Comment reconnaître qu'un nombre entier est pair ou impair lorsqu'on dispose de son écriture binaire ?

Exercice 3

1. Quel est le plus grand nombre entier qu'on peut écrire en binaire avec 8 bits ? 32 bits ? 64 bits ?

2. Comparez ces nombres avec 2^t pour $t = 8, 32, 64$.

Exercice 4

Binaire (0b)	Hexadécimal (0x)	Décimal	Octal
10010010101			
	ED3		
		150	
			2024

Exercice 5

La trame de codes ASCII suivante est reçue sur une ligne RS232. Il s'agit d'une suite de caractères. Elle est codée sans bit de parité. Décoder le message.

$"30 39 2F 32 30 32 34 20 3A 20 42 6F 6E 6A 6F 75 72 20 21"

Exercice 6

On veut convertir le message ci-dessous en trame de codes ASCII.

BTS CIEL - Option : IR

Exercice 7

En utilisant ce tableau, déduire directement l'écriture binaire des nombres suivants :

31, 65, 192, 255 et 512

Bn		**B10**	**B9**	**B8**	**B7**	**B6**	**B5**	**B4**	**B3**	**B2**	**B1**	**B0**
2^n	…	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	…	**1024**	**512**	**256**	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**