# Numération binaire & hexadécimale

## 1 Numération décimale

L'écriture décimale, **en base** 10, d'un nombre entier est de la forme:

$$a_n \times 10^n + \ldots + a_2 \times 10^2 + a_1 \times 10^1 + a_0 \times 10^0$$

On rappelle que : $10^1 = 10$ et $10^0 = 1$.

La numération en base 10, utilise 10 chiffres : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9.

### Exemple

Comment se décompose le nombre entier 206 ?

$$\begin{aligned} 206 &= 2 \times 100 + 0 \times 10 + 6 \times 1 \\ &= 2 \times 10^2 + 0 \times 10^1 + 6 \times 10^0 \end{aligned}$$

On peut représenter cette écriture dans un tableau :

| $10^3$ | $10^2$ | $10^1$ | $10^0$ |
|---|---|---|---|
| 0 | 2 | 0 | 6 |

## 2 Numération binaire

La numération qui utilise seulement les chiffres 0 et 1 est une numération en base 2 dite **binaire**.

Cela signifie que l'on décompose les nombres entiers en une somme de produits avec des puissances de 2.

### Exemple

Le nombre entier 206 se décompose ainsi:

$$\begin{aligned} 206 &= 1 \times 128 + 1 \times 64 + 1 \times 8 + 1 \times 4 + 1 \times 2 \\ &= 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 \end{aligned}$$

On peut représenter cette écriture dans le tableau :

| $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |

### Définition

En informatique, chaque chiffre 0 et 1 est appelé un **bit** de l'anglais **Bi**nary digi**t**.

On appelle **mot binaire** une valeur composée de plusieurs bits.

Le bit situé le plus à **gauche** correspond à la plus grande puissance de 2 est appelé **bit de poids fort**.

Le bit situé le plus à **droite** correspond à la plus petite puissance de 2 est appelé **bit de poids faible**.

On rencontre souvent des mots de 8 bits : ils forment un **octet**.

Dans notre exemple, le nombre 206 en base 10 se note sur un octet en base 2 par le mot binaire 11001110.

Donc :

$$206_{10} = 11001110_2$$

**Remarque**

1. Il existe des mots binaires de 16, 32 ou 64 bits selon les besoins et les valeurs utilisées.
2. La mémoire et les unités de stockage sont exprimées en octets. Aujourd'hui les volumes sont de l'ordre du tera-octet : $10^{12}$ octets !

## 2.1 Conversion de binaire à décimal

**Méthode**

Pour convertir un nombre entier de binaire à décimal, il suffit de calculer la somme de chaque puissance de 2 qui a pour coefficient 1.

Il faut connaitre la position du coefficient pour associer la puissance de 2.

Le tableau ci-dessous redonne les 8 premières puissances de 2, soit pour 1 octet :

| $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|---|---|---|
| 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |

**Exemple**

Soit le nombre entier en base 2 : $001011_2$

En décimal, avec le tableau, on peut écrire :

$$\begin{aligned} 001011_2 &= 1 \times 2^3 + 1 \times 2^1 + 1 \times 2^0 \\ &= 8 + 2 + 1 \\ &= 11 \end{aligned}$$

Donc $11_{10} = 1011_2$

## 2.2 Conversion de décimal à binaire

**Méthode**

Pour convertir un nombre entier de décimal à binaire, il suffit de décomposer le nombre en une somme de puissance de 2 en commençant par la plus grande puissance possible.

On soustrait cette puissance au nombre décimal et on recommence jusqu'à obtenir 0.

Il faut connaitre les puissances de 2 rappelées dans le tableau ci-dessus:

**Exemple**

Soit le nombre entier 41 en notation décimale, en base 10.

- La plus grande puissance de 2 que l'on peut enlever à 41 est : $32 = 2^5$; il reste alors $41 - 32 = 9$.
- La plus grande puissance de 2 que l'on peut enlever à 9 est : $8 = 2^3$; il reste alors $9 - 8 = 1$.
- La plus grande puissance de 2 que l'on peut enlever à 1 est : $1 = 2^0$; il reste alors $1 - 1 = 0$.

$$\begin{aligned} 41_{10} &= 1 \times 2^5 + 1 \times 2^3 + 1 \times 2^0 \\ &= 101001_2 \end{aligned}$$

Le nombre obtenu est sur 6 bits mais en l'écrivant sur 1 octet : 00101001

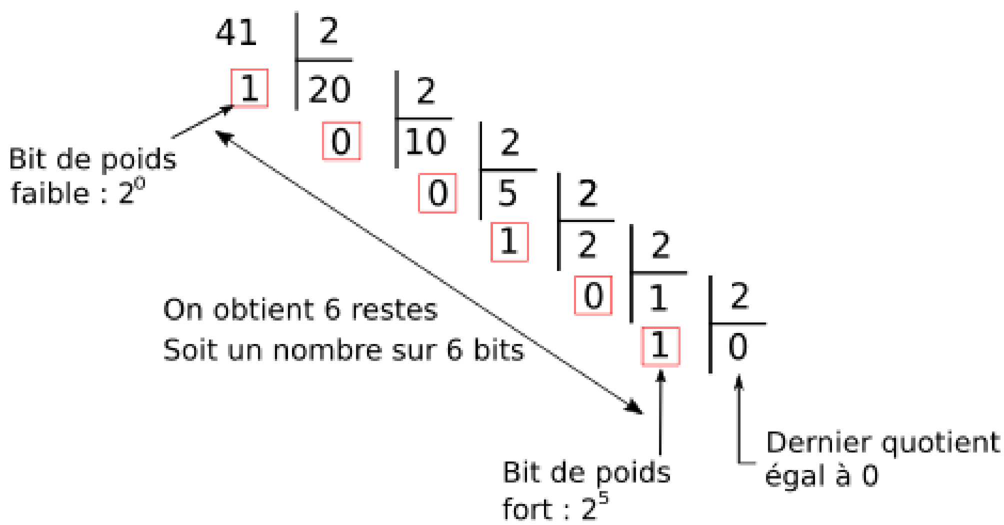
**Méthode**

Pour convertir un nombre entier de décimal en binaire, on calcule la division euclidienne par 2 et on garde le reste.

- Si le quotient est différent de 0, on recommence la division euclidienne du quotient et 2.
- Si le quotient est 0, on obtient la valeur binaire en relevant les restes obtenus en **partant du dernier calculé**.

**Exemple**

Soit le nombre entier 41 en notation décimale. Ci-dessous les différentes divisions euclidiennes à calculer.




Comme pour la méthode précédente, on obtient $41_{10} = 101001_2$

Le nombre obtenu est sur 6 bits mais sur 1 octet : 00101001

## 2.3 Exercice

Donner l'écriture décimale des nombres entiers positifs codés en binaire sur 1 octet.

1. 01010101
2. 10101010
3. 10111010
4. 11001101

## 2.4 Exercice

Donner l'écriture binaire des nombres entiers positfs exprimés en décimal. Faire apparaître le calcul.

1. 13
2. 54
3. 132
4. 245

# 3 Numération hexadécimale

**Définition**

La numération **hexadécimale** est une numération en **base** 16.

Cette numération utilise donc 16 chiffres : - Les 10 chiffres de la numération décimale : de 0 à 9; - et on ajoute les 6 premières lettres de l'alphabet : A, B, C, D, E et F.

En notation décimale : $A = 10$, $B = 11$, $C = 12$, $D = 13$, $E = 14$ et $F = 15$.

En notation binaire sur 4 bits : $A = 1010$, $B = 1011$, $C = 1100$, $D = 1101$, $E = 1110$ et $F = 1111$.

**Exemple**

Soit les nombres entiers 41 et 206 en notation décimale.

En notation hexadécimale :

$$41_{10} = 2 \times 16^1 + 9 \times 16^0 \text{ donc } 41_{10} = 29_{16}$$
$$206_{10} = 12 \times 16^1 + 14 \times 16^0 \text{ donc } 206_{10} = CE_{16}$$

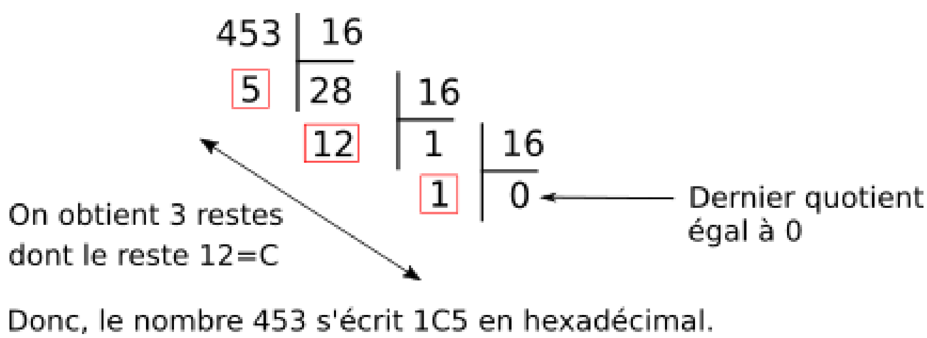
## 3.1 Conversion décimale en hexadécimale

**Méthode**

La conversion d'un nombre décimal en hexadécimal est délicate. On peut utiliser les mêmes méthodes que pour la conversion en binaire, à savoir par soustraction de puissances de 16 ou les divisions euclidiennes par 16.

**Exemple**

Soit le nombre entier positif 453 à convertir en hexadécimal: $453_{10} = 1C5_{16}$




## 3.2 Conversion binaire en hexadécimale

**Remarque**

La conversion de binaire à hexadécimal et inversement est beaucoup plus simple.

Chaque chiffre hexadecimal correspond à un groupe de 4 bits appelé **quartet**.

**Méthode**

Lorsqu'une valeur binaire est codée sur un octet, on casse en 2 fois 4 bits le nombre pour obtenir 2 quartets.

Chaque quartet correspond à un chifre hexadécimal.

**Exemple**

Le nombre 41 s'écrit 00101001 en binaire soit $\underbrace{0010}\underbrace{1001}$.

On obtient les quartets 0010 et 1001.

Le premier quartet vaut 2 et le second quartet vaut 9 en hexadécimal.

Donc:

$$41_{10} = 00101001_2 = 29_{16}$$

### 3.3 Exercice

Convertir en notation binaire et décimale les nombres suivants écrits en notation hexadécimale:

1. 3E
2. 4F8B
3. AA553C

### 3.4 Exercice

Convertir en écriture hexadécimale les nombres suivants (attention à la base):

1. $168_{10}$
2. $1001_{10}$
3. $10100101_2$

## 4 Pour info

Il existe d'autres bases de numération comme la base octale. Sur les systèmes d'exploitation Linux, les droits des utilisateurs sont exprimés en base octale. Un utilisateur qui a les droits 777 sur un dossier ou un fichier a tous les droits.

De façon générale, une base de valeur $b$ permet d'écrire des nombres dans cette base s'exprimant par :

$$a_n \times b^n + a_{n-1} \times b^{n-1} + \ldots + a_1 \times b^1 + a_0$$

Les coefficients $a_i$ sont strictement inférieurs à $b$.

**Exemple**

En base octale $b = 8$, les coefficients sont les nombres entiers compris entre 0 et 7.

Le nombre 755 a pour valeur : $7 \times 8^2 + 5 \times 8^1 + 5 = 493$ en base 10.