# Chapitre 02 : Représentation de l'information

## 1. Introduction

### 1.1. Représentation de l'information traitée par ordinateur

Les informations traitées par un ordinateur peuvent être de différents types (texte, nombres, images, son, vidéos, etc.) mais elles sont toujours représentées et manipulées par l'ordinateur sous forme numérique (digitale). En fait, toute information sera traitée comme une suite de 0 et de 1. L'unité d'information est donc les chiffres binaires (0 et 1) que l'on appelle bit (pour binary digit : chiffre binaire).

On utilise la représentation binaire car elle est simple, facile à réaliser techniquement à l'aide de bistables (système à deux états réalisés à l'aide de transistors).

Le codage d'une information consiste à établir une correspondance entre la représentation externe (habituelle) de l'information (texte, image, …etc), et sa représentation interne dans la machine, qui est toujours une suite de bits.

### 1.2. Quantité de l'information traitée

L'unité de base de mesure de la quantité d'information en informatique est donc le **bit** tel qu'un bit peut prendre la valeur 0 ou 1.

**Question :** Combien d'états peut-on représenter avec 3 bits ? avec 4 bits ? et avec n bits en général ?

Chaque 8 bits constituent 1 **Octet** (Byte en anglais) symbolisé par Ø (et symbolisé par B en anglais).

Aussi :

- $2^{10}$ bits = 1024 bits = 1 **Kb** (1 Kilo bits) / $2^{10}$ Ø = 1024 Ø = 1 **KØ** (1 Kilo Ø)
- $2^{10}$ Kb = 1024 Kb = 1 **Mb** (1 Méga bits) / $2^{10}$ KØ = 1024 KØ = 1 **MØ** (1 Méga Ø)
- $2^{10}$ Mb = 1024 Mb = 1 **Gb** (1 Giga bits) / $2^{10}$ MØ = 1024 MØ = 1 **GØ** (1 Giga Ø)
- $2^{10}$ Gb = 1024 Gb = 1 **Tb** (1 Téra bits) / $2^{10}$ GØ = 1024 GØ = 1 **TØ** (1 Téra Ø)

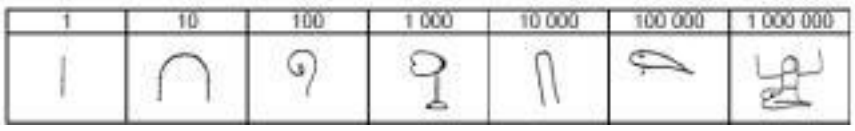
**Question :** Convertir 2 GØ en bits puis en Kb?

Il est donc clair que l'information est traitée sous forme binaire dans un ordinateur, que ça soit du texte (association de codes conventionnés à chaque caractère), des images (association de codes à chaque couleur de pixel de l'image), du son (association de codes à chaque fréquence de son), …etc. Il est donc indispensable de voir de plus près la manipulation des données binaires ainsi que leur relation avec d'autres systèmes de numération.

## 2. Systèmes de numération

Au fil du temps plusieurs systèmes de numération sont apparus. De la mésopotamienne qui était positionnelle (la position du chiffre indique son rang, comme dans la numération arabe qu'on utilise aujourd'hui) à l'égyptienne et la romaine qui était additionnelle (le nombre représenté est égal à la somme des symboles représentés), à celle des chinois qui excellaient dans les calculs (création du boulier) et qui est également positionnelle, …etc.

**Exemple :** numération égyptienne




 **= 345**

### 2.1. Représentation

Un nombre : $(XXX)_b$ indique la représentation d'un nombre XXX dans la base **b**.

Les bases usuelles qu'on connait et utilise tous les jours sont la base 10 (système décimale) pour représenter les différentes grandeurs et les différents chiffres et nombre (monnaies, n° de tel, tailles, date, …) et la base 60 (système sexagésimal) pour représenter le temps.

Comment un nombre est représenté dans une base b ?

1. Si $b \leq 10$, on utilise simplement les chiffres de 0 à b-1

**Exemple :** base 8 (système octal) : n'importe quel nombre sera la combinaison de chiffres appartenant à l'ensemble {0,…, 7}

2. Si b > 10, on utilise simplement les chiffres de 0 à 9 ensuite des lettres dans l'ordre alphabétique.

**Exemple :** base 16 (système hexadécimal) : n'importe l'ensemble quel nombre sera la combinaison de symboles appartenant à {0,…, 9,A, B, C, D, E, F} tel que : (A=10, ….., F=15).

Donc chaque système de numération utilise un ensemble de symboles (chiffres) pour représenter les différents nombres. Le nombre de ces chiffres est toujours égal à l'ordre de la base elle-même. Autrement dit : la base du système de numération est égale au cardinal de l'ensemble des symboles utilisés dans cette base.

**Exemple :**

- En binaire, base du système binaire = 2 ; ensemble des symboles utilisés : A = {0,1}, Card (A) = 2 = base du système binaire.

- En octal, base du système octal = 8 ; ensemble des symboles utilisés : A ={0,1,2,3,4,5,6,7}, Card (A) =8 =base du système octal.

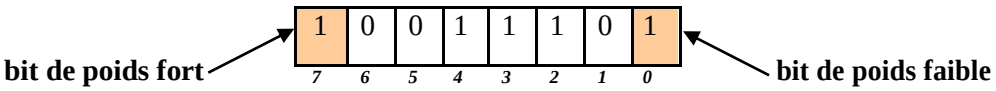
**Alors :**

- Un nombre de **n** chiffres (symboles) est une suite ($a_i$), $\mathbf{0 \leq i \leq n-1}$ :

$$a_{n-1} \ldots\ldots a_1 a_0$$

**tel que :** $a_0$ est le terme de poids faible et $a_{n-1}$ est le terme de poids fort.

**Exemple :** Soit le mot binaire de 8 bits : 10011101




Les systèmes de numération qui nous intéresse dans le domaine informatique sont : le décimal, le binaire, l'octal et l'hexadécimal.

## 2.2. Le système décimal

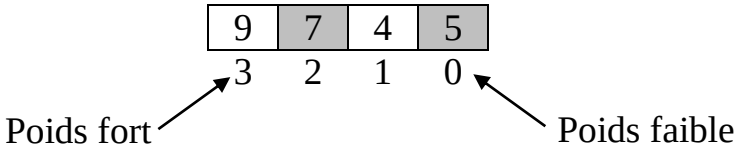
Le système décimal est celui dans lequel nous avons le plus l'habitude d'écrire. Chaque chiffre peut avoir 10 valeurs différentes : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, de ce fait, le système décimal a pour base 10. La valeur de chaque chiffre dépend de sa position, c'est-à-dire que c'est une numération positionnelle : la position la plus à droite exprime les unités, la position suivante : les dizaines, ensuite les centaines,…etc.

Par exemple, si on décompose le nombre 9745, nous aurons :

- $9745 = 9\times1000 + 7\times100 + 4\times10 + 5\times1$
- $9745 = 9\times10^3 \times 7\times10^2 + 4\times10^1 + 5\times10^0$

Nous remarquons que chaque chiffre du nombre est à multiplier par une puissance de 10.

Cette puissance représente le poids du chiffre.




L'exposant de cette puissance est nul pour le chiffre situé le plus à droite et s'accroît d'une unité pour chaque passage à un chiffre vers la gauche.

**Remarque :** Cette façon d'écrire les nombres est appelée système de numération de position. Elle est valable pour tous les systèmes de numération que nous verrons dans ce cours (décimal, binaire, octal et hexadécimal).

## 2.3. Le système octal

Suivant ce que nous avons cité dans la ***section 2.1***, le système octal utilise un système de numération ayant comme base 8 (octal : latin octo=huit) et utilise donc 8 symboles :

de 0..jusqu'à..7. Ainsi, un nombre exprimé en base 8 pourra se présenter de la manière suivante par exemple : $(745)_8$

**Rappel :** Lorsque l'on écrit un nombre, il faudra bien préciser la base dans laquelle on l'exprime pour lever toutes ambiguïtés (745 existe aussi en base 10 par exemple). Ainsi le nombre sera mis entre parenthèses (745 dans notre exemple) et indicé d'un nombre représentant sa base (8 est mis en indice). Par convention, quand on ne précise pas la base, elle est par défaut égale à 10.

## 2.4. Le système binaire

Comme nous l'avons vu plus haut, dans le système binaire, chaque chiffre ne peut avoir que l'une des 2 valeurs : 0 ou 1. De ce fait, le système a pour base 2.

**Exemple :** Représentions des nombres de 0 à 16 en décimal et leurs équivalents en binaire et octal

| $(\text{Système décimal})_{10}$ | $(\text{Système octal})_{8}$ | $(\text{Système binaire})_{2}$ |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 2 | 2 | 10 |
| 3 | 3 | 11 |
| 4 | 4 | 100 |
| 5 | 5 | 101 |
| 6 | 6 | 110 |
| 7 | 7 | 111 |
| 8 | 10 | 1000 |
| 9 | 11 | 1001 |
| 10 | 12 | 1010 |
| 11 | 13 | 1011 |
| 12 | 14 | 1100 |
| 13 | 15 | 1101 |
| 14 | 16 | 1110 |
| 15 | 17 | 1111 |
| 16 | 20 | 10000 |

## 2.5. Le système hexadécimal

Le système hexadécimal utilise les 16 symboles suivant : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E et F. De ce fait, le système a pour base 16.

**Exemple :** si nous reprenons le tableau précédent mais en valeurs décimales et leurs équivalents binaires et hexadécimales, nous aurons :

| $(\text{Système décimal})_{10}$ | $(\text{Système Hexadécimal})_{16}$ | $(\text{Système binaire})_{2}$ |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 2 | 2 | 10 |
| 3 | 3 | 11 |
| 4 | 4 | 100 |
| 5 | 5 | 101 |
| 6 | 6 | 110 |
| 7 | 7 | 111 |
| 8 | 8 | 1000 |
| 9 | 9 | 1001 |
| 10 | A | 1010 |
| 11 | B | 1011 |
| 12 | C | 1100 |
| 13 | D | 1101 |
| 14 | E | 1110 |
| 15 | F | 1111 |
| 16 | 10 | 10000 |

# 3. Conversion et changement de base

## 3.1. Conversion d'un nombre de base b quelconque en nombre décimal

Tout nombre entier naturel peut se coder comme la somme pondérée des puissances de sa base b, quel que soit cette base :

Soit $a_n\ a_{n-1}\ \ldots\ a_2\ a_1\ a_0$ exprimé en base b noté $(a_n\ a_{n-1}\ \ldots\ a_2\ a_1\ a_0)_b$. La valeur de ce nombre en décimal est égale à :

$$(a_n \times b^n) + (a_{n-1} \times b^{n-1}) + \ldots + (a_2 \times b^2) + (a_1 \times b^1) + (a_0 \times b^0)$$

**Exemple :** Convertissons les nombres suivants en décimal : $(1011)_2$, $(16257)_8$ et $(F53)_{16}$

- $\mathbf{(1011)_2 = (1 \times 2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0)_{10} = (1\times8 + 0\times4 + 1\times2 + 1\times1)_{10} = (11)_{10}}$
- $\mathbf{(16257)_8 = 1 \times 8^4 + 6\times8^3 + 2\times8^2 + 5\times8^1 + 7\times8^0 = 1\times4096 + 6\times512 + 2\times64 + 5\times8 + 7}$

  $\mathbf{= 4096 + 3072 + 128 + 40 + 7 \quad = \quad 7343}$

- $\mathbf{(F53)_{16} = 15 \times 16^2 + 5 \times 16^1 + 3 \times 16^0 = 15 \times 256 + 5 \times 16 + 3 = 3840 + 80 + 3 = 3923}$

**Remarque :** Dans le cas où il y a une partie fractionnaire $a_1\ a_2\ \ldots\ a_n$ (les nombres fractionnaires sont ceux qui comportent des chiffres après la virgule), sa valeur sera égale en décimal à la somme suivante :

$$\mathbf{a_1 \times b^{-1} + a \times b^{-2} + \ldots\ldots + a_n \times b^{-n}}$$

**Exemple :** Convertissons les nombres fractionnaires suivants en décimal : $(1110,101)_2$ , $(642,21)_8$ et $(A3F,C)_{16}$

- $(1110,101)_2 = (1\times2^3 + 1\times2^2 + 1\times2^1 + 0\times2^0 + 1\times2^{-1} + 0\times2^{-2} + 1\times2^{-3})_{10}$

  $= (8 + 4 + 2 + 0 + 1/2 + 1/4 + 1/8)10 = (14,625)_{10}$
- $(642,21)_8 = (6\times8^2 + 4\times8^1 + 2\times8^0 + 2\times8^{-1} + 1\times8^{-2})_{10}$

  $= (6\times64 + 4\times8 + 2\times1 + 2\times 0,125 + 1\times 0,015625)_{10}$

  $= (384 + 32 + 2 + 0.25 + 0.015625)10 = (418.265\ 625)10$

- $(A3F,C)_{16} = (10\times16^2 + 3\times16^1 + 15\times16^0 + 12\times16^{-1})10$

  $= (10\times256 + 3\times16 + 15\times1 + 12\times 0{,}0625)10$

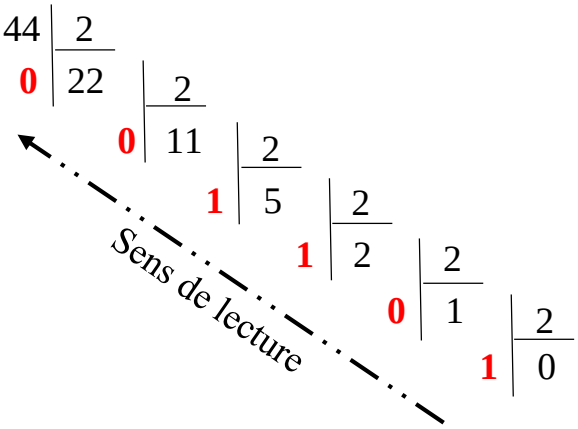
  $= (2623.75)10$

## 3.2. Conversion d'un nombre décimal en nombre binaire

Pour obtenir l'expression binaire d'un nombre exprimé en décimal, il suffit de diviser successivement ce nombre par 2 jusqu'à ce que le quotient obtenu soit égal à 0. Les restes de ces divisions lus de bas en haut représentent le nombre binaire.

**Exemple :** Convertissons le nombre décimal 44 en binaire :

$(44)_{10} = (101100)_2$

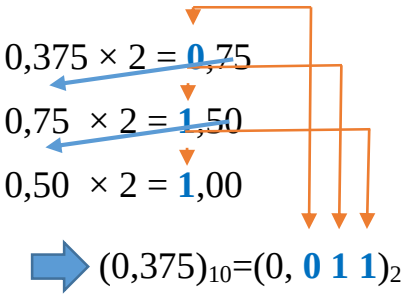



### 3.2.1. Conversion partie décimal « fractionnaire → binaire »

La partie entière est converti du décimal en binaire en utilisant la méthode de la division successive comme on vient de le voir, quand à la partie fractionnaire du nombre décimal, elle sera convertie en partie fractionnaire binaire en utilisant une méthode de multiplication successives.

La partie entière du résultat de chaque multiplication est considérée comme constituant de la partie fractionnaire binaire et la partie fractionnaire du résultat de la multiplication est repris pour être lui-même multiplié par 2, et ainsi de suite jusqu'à ce que le produit obtenu est 1.00, le processus de conversion est alors achevé.

**Exemple :** Convertissons les nombres fractionnaires décimaux 0.375 et 0.84375 en binaire :




$(0,375)_{10}=(0, \mathbf{0\ 1\ 1})_2$

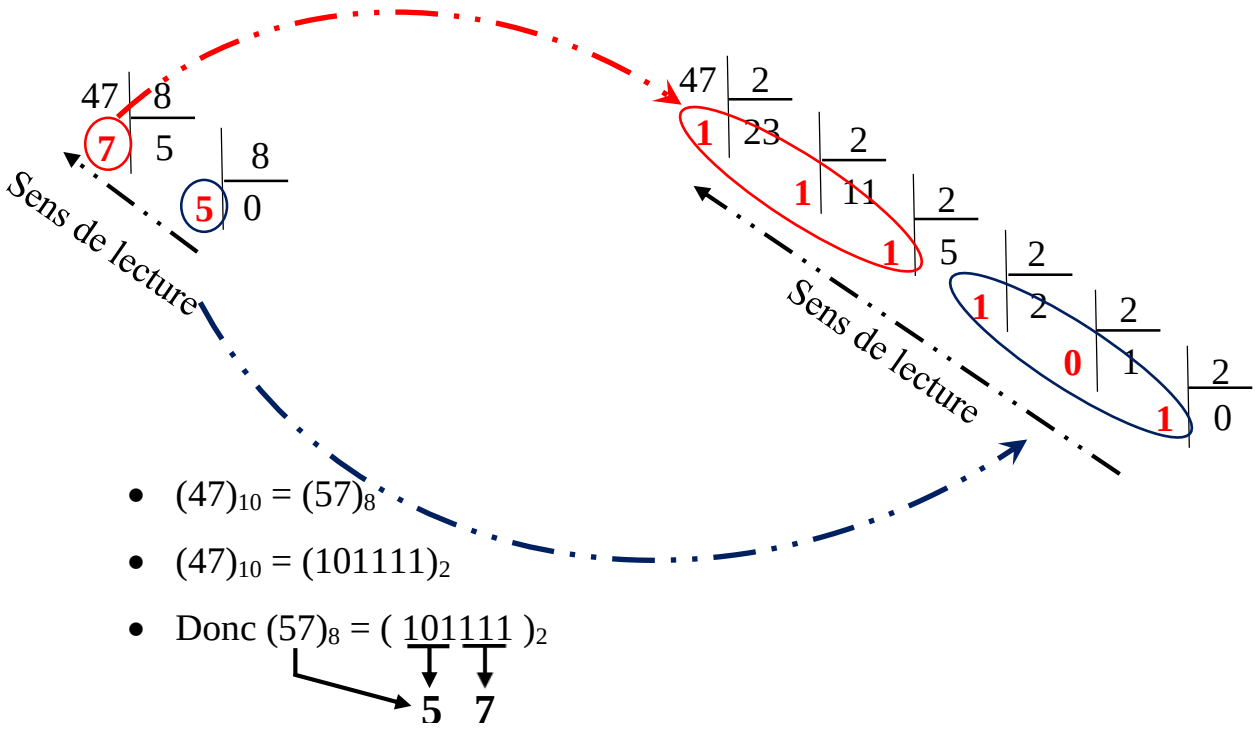
0,84375 × 2 = 1,6875
0,6875 × 2 = 1,375
0,375 × 2 = 0,75
0,75 × 2 = 1,50
0,50 × 2 = 1,00

$(0,84375)_{10} = (0,\mathbf{11011})_2$

## 3.3. Relation entre les nombres binaires et les nombres octaux

D'abord, si nous souhaitons obtenir l'expression octale d'un nombre exprimé en décimal, il suffit de suivre la méthode de la division successive par 8 (comme nous l'avons fait pour convertir vers la base 2) jusqu'à ce que le quotient obtenu soit égal à 0. Les restes de ces divisions lus de bas en haut représentent le nombre octal.

**Exemple :** Convertissons $(47)_{10}$ dans le système octal et le système binaire. Nous obtenons :




- $(47)_{10} = (57)_8$
- $(47)_{10} = (101111)_2$
- Donc $(57)_8 = (\ 101111\ )_2$

5 7

Nous pouvons remarquer qu'après 3 divisions en binaire nous avons le même quotient qu'après une seule en octal. De plus le premier reste en octal obtenu peut être mis en relation directe avec les trois premiers restes en binaire

- $(111)_2 = 1\times2^2 + 1\times2^1 + 1\times2^0 = 1\times4 + 1\times2 + 1\times1 = (7)_8$

Et il en est de même pour le caractère octal suivant :

- $(101)_2 = 1\times2^2 + 0\times2^1 + 1\times2^0 = 1\times4 + 0\times2 + 1\times1 = (5)_8$

Cette propriété d'équivalence entre chaque chiffre octal et chaque groupe de 3 chiffres binaires vient du fait que 8 est une puissance de 2 : $8=2^3$. Elle nous permet de passer facilement d'un système à base 8 à un système à base 2 et vice versa.

Exemple de conversion binaire octal et octal binaire :

Binaire ( 101 111 100 001)$_2$

Octal ( 5 7 4 1 )$_8$

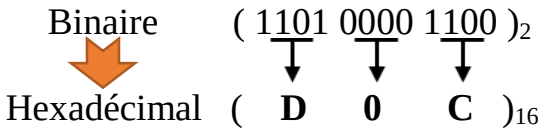
Octal ( 3 6 2 )$_8$

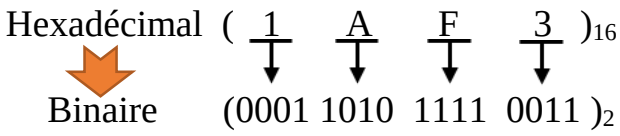
Binaire ( 011 101 111 )$_2$

## 3.4. Relation entre les nombres binaires et les nombres hexadécimaux

Si nous avons besoin d'obtenir l'expression hexadécimal d'un nombre exprimé en décimal, il faut toujours suivre la méthode de la division successive, cette fois-ci par 16 (comme nous l'avons fait pour convertir vers les bases 2 et 8) jusqu'à ce que le quotient obtenu soit égal à 0. Les restes de ces divisions lus de bas en haut représentent le nombre hexadécimal (tout en prenant compte que les restes de 10 à 15 sont codés : de A à F).

La propriété d'équivalence que nous venons de voir, dans ***3.3***, entre le binaire et l'octal existe entre l'hexadécimal et le binaire du moment que 16 est aussi une puissance de 2 : $16=2^4$. Donc la règle est la même mais nous travaillerons par groupe de 4 chiffres binaires maintenant au lieu de 3.




**Remarques :**

*1 : Pour la conversion de tout nombre entier de la base 10 vers une base quelconque, on procède toujours par divisions successives. On divise le nombre à convertir par la base dans laquelle nous voulons le convertir, puis le quotient obtenu par la base, et ainsi de suite jusqu'à l'obtention d'un quotient nul. La suite des restes obtenus correspond aux chiffres dans la base visée.*

*2 : Pour la conversion de la partie fractionnaire décimal vers son équivalent octal (ou hexadécimal), on procède toujours par multiplications successives comme on a fait pour la conversion décimal-fractionnaire vers le binaire dans la section **3.2.1**. On multiplie la partie fractionnaire par 8 (ou 16), la partie entière du résultat entre dans la constitution de la partie fractionnaire octale (hexadécimal) ensuite, la partie fractionnaire du résultat de la multiplication est lui-même multiplié à nouveau par 8 (par 16) et ainsi de suite jusqu'à obtention d'un résultat égal à 0,00.*

**Exemple :** Convertissons le nombre décimal 418,265625 en octal :

- 418 ÷ 8 = 52 reste 2 0,265625 × 8 = **2**.125
- 52 ÷ 8 = 6 reste 4 0,125 × 8 = **1**.00
- 6 ÷ 8 = 0 reste 6 0,00 × 8 = **0**.00

Donc :

$(418{,}265625)_{10} = (642{,}21)_8$

*(De la même façon, on procède pour la conversion fractionnaire décimale vers son équivalent hexadécimal).*

# 4. Les opérations de base en binaire

## 4.1. Addition et multiplication binaire

Le principe du calcul numérique est le même dans les systèmes de numération de position. Donc nous raisonnerons de la même façon que pour réaliser des opérations dans le système décimal où on a l'habitude d'effectuer nos opérations arithmétiques quotidiennement.

### a. Addition binaire

Révisons d'abord la familière addition décimale. L'addition de 2 nombres décimaux s'effectue selon un algorithme à 3 étapes :

***Etape 1 :*** *ajouter les chiffres les plus à droite (première colonne)*

***Etape 2 :*** *noter le chiffre d'unité de cette somme à la même colonne toujours et si cette somme dépasse* ***9****, on reporte à la colonne suivante la retenue (chiffre de deuxième position de la somme obtenue)*

***Etape 3*** *: s'il y a d'autres colonnes, répéter les 2 étapes précédentes sans oublier de rajouter la retenue jusqu'à ce qu'il n'y ait plus de colonnes.*

En binaire, l'algorithme est le mê me sauf que la retenue sera en cas où la somme dépasse **2** (et non pas **9**), tel que :

*0 + 0 = 0  0 + 1 = 1  1 + 0 = 1  1 + 1 = 0 avec retenue de 1*

*1 + 1+ 1 = 1 avec retenue de 1*

***Exemple :*** *évaluons la somme binaire : 111+101*

```
  1 1
  1 1 1
+ 1 0 1
-------
1 1 0 0
```

## b. Multiplication binaire

Se résume comme en décimal en multiplication de nombres par des chiffres suivi d'additions décalée. En fait, en binaire c'est encore plus simple du moment que la multiplication par 0 ou 1 donne 0 ou le nombre lui-même (pas de tableaux de multiplication à apprendre comme en décimal !).

**Exemple :** évaluons le produit binaire : 1101011 × 10110

Ça revient à faire ceci si on procède à l'addition des termes un à un sans oublier de prendre le décalage en considération :

**Donc :** 1101011 × 10110 = 100100110010

```
      1101011
    X   10110
-------------
      0000000
+     1101011.
+    1101011..
+   0000000...
+  1101011....
-------------
 100100110010
```

***Remarque :*** *pour les multiplications des nombres fractionnaires, la règle est la même qu'en décimal.*

**Exemple :** évaluons le produit binaire : 11,01 × 101,1

```
      11,01
   X  101,1
-----------
      1101
+     1101.
+    0000..
+   1101...
-----------
 100011,111
```

## c. Soustraction binaire

La soustraction s'effectue suivant le principe de retenue comme en décimal :

0 - 0 = 0  1 - 1 = 0  1 - 0 = 1  0 - 1 = 1 avec une retenue de 1 sur la colonne suivante.

**Exemple :** évaluons les soustractions suivantes :

```
   0
  11101
- 1011
-------
  10010
```

```
    011
  11000
- 10011
-------
    101
```

```
   00  0  1  00
  1101,00110
-  110,11011
-----------
  110,01011
```

**Remarque :**

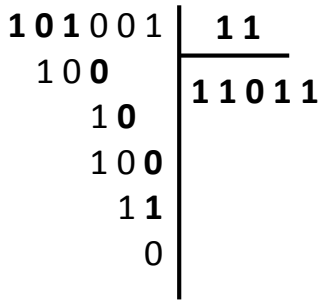
***1 :*** *Dans le cas de fractions, il faut d'abord aligner verticalement les virgules avant de commencer l'opération de soustraction.*

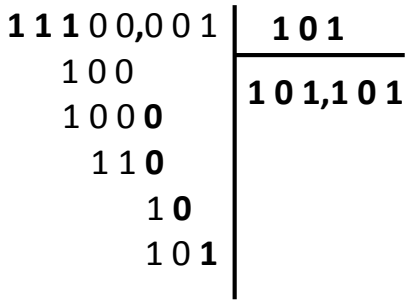
***2 :*** *Quand dans une colonne, apparait la différence 0 - 1, nous opéron une retenue sur la 1ère colonne non nulle et tous les 0 juste avant deviennent des 1.*

## d. Division binaire

Il s'agit de multiplications et de soustractions successives comme en décimal. En cas de nombres fractionnaires, on déplace d'abord la virgule, ensuite on effectue l'opération de division

**Exemple :** effectuons les divisions : 1010001 ÷ 11 et 111,00001 ÷ 1,01 , nous avons :




***Remarque :*** *Nous avons étudié ces opérations du côté purement arithmétiques, mais du point de vue 'structure machine' il peut y avoir quelques problèmes qui peuvent être détectés pour ne pas fournir un résultat faux. Par exemple, si on travaille sur 6 bits, l'addition suivante :* ***111001 + 010010*** *fournit un résultat sur 7 bit :1001011, donc le 1 le plus à gauche sera perdu ! on parle alors de dépassement de capacité (over flow). Il faut donc qu'il y ait un indicateur de dépassement et l'erreur doit être signalée.*