# Chapitre 1

## I. SYSTEMES DE NUMERATION

Il existe de nombreux systèmes de numérisation, les plus utilisés sont: Décimal (base 10), Binaire (base 2), Octal (base 8) et Hexadécimal (base 16).

### 1. 1.Système décimal (base 10) :

Le système décimal comprend 10 chiffres qui sont {0, 1, 2, 3, 4, 5, 6, 7, 8, 9} c'est un système qui s'est imposé tout naturellement à l'homme qui possède 10 doigts. Ecrivons quelques nombres décimaux sous la forme polynomiale :

*Exemples :*

$(5462)_{10}= 5*10^3 + 4*10^2 + 6*10^1 + 2*10^0$

$(239.537)_{10}= 2*10^2 + 3*10^1 + 9*10^0 + 5*10^{-1} + 3*10^{-2} + 7*10^{-3}$

### 1. 2.Système binaire (base 2) :

Dans ce système de numération il n'y a que deux chiffres possibles {0, 1} qui sont souvent appelés bits « binary digit ». Comme le montre les exemples suivants, un nombre binaire peut s'écrire sous la forme polynomiale.

*Exemples :*

$(111011)_2= 1*2^5 + 1*2^4 + 1*2^3 +0*2^2 + 1*2^1 + 1*2^0$

$(10011.1101)_2= 1*2^4 + 0*2^3 + 0*2^2 + 1*2^1 + 1*2^0 + 1*2^{-1} + 1*2^{-2} + 0*2^{-3} + 1*2^{-4}$

### 1. 3.Système Octal (base 8) :

Le système octal ou base 8 comprend huit chiffres qui sont {0, 1, 2, 3, 4, 5, 6, 7}. Les chiffres 8 et 9 n'existent pas dans cette base.

*Exemples :*

$(4527)_8= 4*8^3 + 5*8^2 + 2*8^1 + 7*8^0$

$(1274.632)_8= 1*8^3 + 2*8^2 + 7*8^1 +4*8^0+ 6*8^{-1} + 3*8^{-2} + 2*8^{-3}$

### 1. 4.Système Hexadécimal (base 16) :

Le système Hexadécimal ou base 16 contient seize éléments qui sont {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}. Les chiffres A, B, C, D, E, et représentent respectivement 10, 11, 12, 13, 14 et 15.

*Exemples :*

$(3256)_{16}= 3*16^3 + 2*16^2 + 5*16^1 + 6*16^0$

$(9C4F)_{16}= 9*16^3 + 12*16^2 + 4*16^1 + 15*16^0$

$(A2B.E1)_{16}= 10*16^2 + 2*16^1 + 11*16^0 +14*16^{-1}+ 1*16^{-2}$

# I. CHANGEMENT DE BASE

Il s'agit de la conversion d'un nombre écrit dans une base $\mathbf{B_1}$ à son équivalent dans une autre base $\mathbf{B_2}$

## ***2.1. Conversion d'un nombre de base quelconque en un nombre décimal :***

La valeur décimale d'un nombre N, écrit dans une base B, s'obtient par sa formepolynomiale décrite précédemment.

*Exemples :*

$(1011101)_2 = 1*2^6 + 0*2^5 + 1*2^4 + 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0 = (93)_{10}$

$(7452)_8 = 7*8^3 + 4*8^2 + 5*8^1 + 2*8^0 = (3882)_{10}$

$(D7A)_{16} = 13*16^2 + 7*16^1 + 10*16^0 = (3450)_{10}$

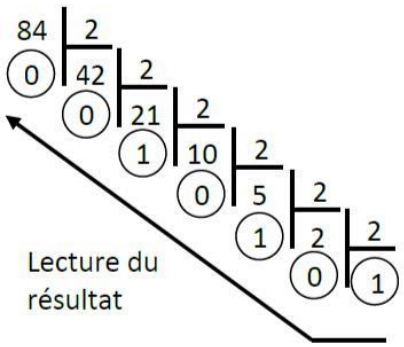
## ***2.2. Conversion d'un nombre décimal en un nombre de base quelconque:***

### *2.2.1. Nombre décimal entier*

Pour convertir un nombre décimal entier en un nombre de base **B** quelconque, il faut faire des divisions entières successives par la base **B** et conserver à chaque fois le reste de la division. On s'arrête lorsqu'on obtient un résultat inferieur à la base B. Le nombre recherche N dans la base **B** s'écrit de la gauche vers la droite en commençant par le dernier résultat allant jusqu'au premier reste.

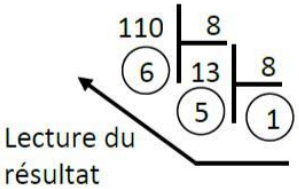
*Exemples :*

➲ $(84)_{10}=(\ ?\ )_2$



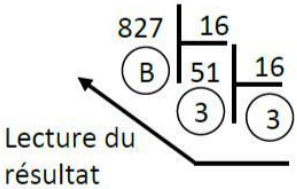

$(84)_{10}=(1010100)_2$

➲ $(110)_{10}=(\ ?\ )_8$




$(110)_{10}=(156)_8$

➲ $(827)_{10}=(\ ?\ )_{16}$



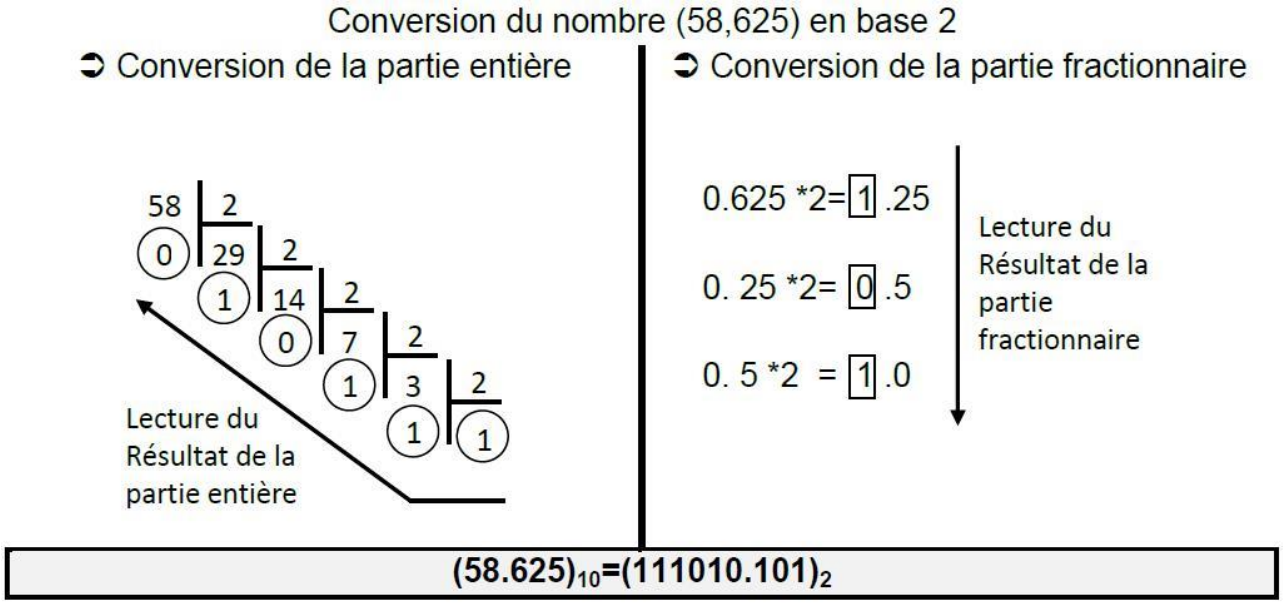

$(827)_{10}=(33B)_{16}$

*2.2.2. <u>Nombre décimal à virgule :</u>*

Pour convertir un nombre décimal à virgule dans une base **B** quelconque, il faut :

- Convertir la partie entière en effectuant des divisions successives par **B** (comme nous l'avons vu précédemment).
- Convertir la partie fractionnaire en effectuent des multiplications successives par **B** et en conservant à chaque fois le chiffre devenant entier.

*Exemples :*




*Remarque:*

Parfois en multipliant la partie fractionnaire par la base **B** on n'arrive pas à convertir toute la partie fractionnaire. Ceci est dû essentiellement au fait que le nombre à convertir n'a pas un équivalent exacte dans la base **B** et sa partie fractionnaire est cyclique.

*Exemple* :

$(0.15)_{10}=(\ ?)_2$

0.15 *2 = 0 .3

0.3 *2 = 0 .6

0.6 *2 = 1 .2

0.2 *2 = 0 .4

0.4 *2 = 0 .8

0.8 *2 = 1 .6

0.6 *2 = 1 .2

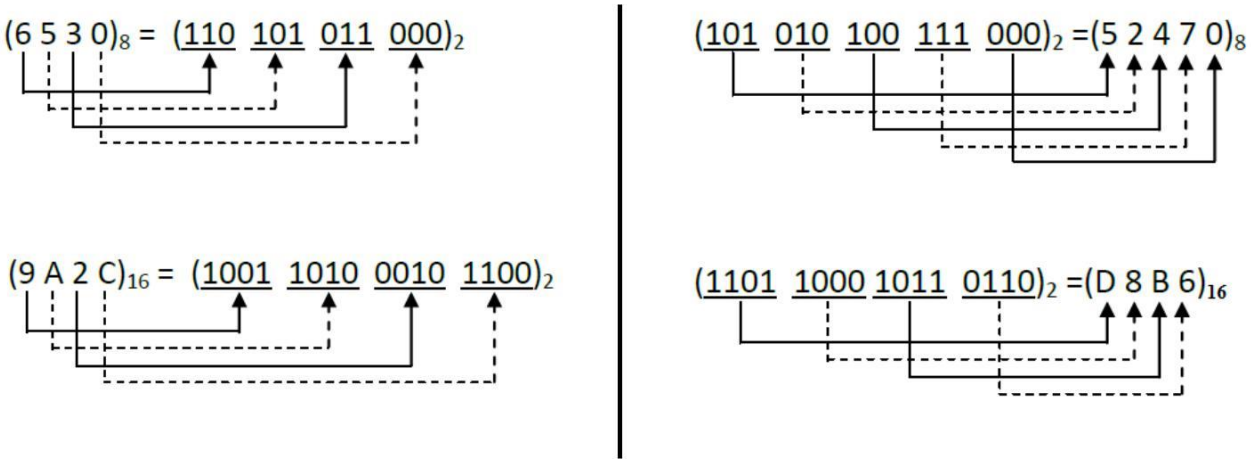
$(0.15)_{10}= (0.001001\mathbf{1001})_2$

## *2.3. Autres conversions:*

Pour faire La conversion d'un nombre d'une base quelconque $\mathbf{B_1}$ vers une autre base $\mathbf{B_2}$ il faut passer par la base **10**. Mais si la base $\mathbf{B_1}$ et $\mathbf{B_2}$ s'écrivent respectivement sous la forme d'une puissance de 2 on peut passer par la base 2 (binaire) :

- Base octale (base 8) : $8=2^3$ chaque chiffre octal se convertit tout seul sur 3 bits.
- Base hexadécimale (base 16) : $16=2^4$ chaque chiffre hexadécimal se convertit tout seul sur 4 bits.

*Exemple* :




Le tableau ci-dessous représente un récapitulatif sur ces systèmes :

| Décimal | Binaire | Octal | Hexadécimal |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| 2 | 10 | 2 | 2 |
| 3 | 11 | 3 | 3 |
| 4 | 100 | 4 | 4 |
| 5 | 101 | 5 | 5 |
| 6 | 110 | 6 | 6 |
| 7 | 111 | 7 | 7 |
| 8 | 1000 | 10 | 8 |
| 9 | 1001 | 11 | 9 |
| 10 | 1010 | 12 | A |
| 11 | 1011 | 13 | B |
| 12 | 1100 | 14 | C |
| 13 | 1101 | 15 | D |
| 14 | 1110 | 16 | E |
| 15 | 1111 | 17 | F |

## II. Représentation des nombres signes en binaire :

Ici, nous allons examiner comment représenter les nombres négatifs dans la base 2. Nous allons devoir spécifier quel est le nombre de bits utilisé pour représenter un nombre.

Nous savons que toute information est codée dans un ordinateur avec un 1 ou un 0. Aucun autre symbole ne peut être utilisé. Jusqu'alors nous avons représenté les nombres en supposant qu'ils étaient positifs. Pour les nombres avec un signe, deux systèmes sont possibles :

- la représentation signe-grandeur
- la représentation en complément a deux

### *3.1. La représentation signe-grandeur (SG):*

Le principe est d'utiliser un bit pour représenter le signe du nombre. Par exemple, dans un nombre sur 8 bits « b7…b0 », on a la valeur absolue du nombre sur les bits « b6…b0 » et le bit b7 vaut 0 pour un nombre positif et 1 pour un bit négatif.

*Exemple* :

Sur 8 bits, on représentera

+53 = 0011 0101

- 53 = 1011 0101

L'utilisation du bit de signe présente des inconvénients :

- d'une part, il y a deux représentations distinctes pour une même valeur ; sur huit bits, 0000 0000 et 1000 0000 valent tous les deux 0.

- D'autre part, l'addition et la soustraction sont un peu compliqué : pour additionner deux nombres, il faut additionner les deux valeurs absolues si les bits de signe sont identiques. S'ils sont différents il faut soustraire la plus petite valeur absolue de la plus grande. La comparaison de deux nombres est aussi un peu compliquée, puisqu'il faut là aussi traiter le bit de signe comme un cas particulier.

### *3.2. La représentation en complément a deux :*

La représentation la plus courante pour les entiers dans l'ordinateur est la représentation dite en complément à 2, qui présente deux avantages :

D’une part on peut additionner des nombres en complément à deux avec les règles usuelles de l'addition, sans se soucier de leur signe : le résultat de l'addition aura le bon signe ; D’autre part, chaque nombre qu'on peut représenter ne possède qu'une représentation unique.

- Il est facile de calculer l'opposé d'un nombre en complément à 2 en effectuant deux opérations successives : on bascule tous les bits et on ajoute 1.

Le basculement de tous les bits consiste à remplacer chaque bit à 0 de la chaîne de bits d'origine par un bit à 1 dans le résultat et chaque bit à 1 de la chaîne de bits d'origine par

un bit à 0. Cela s'appelle le complément à 1 d'un nombre. Ajouter 1 au complément à 1 de la représentation d'un nombre permet d'obtenir son complément à 2.

- Pour convertir un nombre négatif du décimal vers le complément à 2 on calcule la représentation binaire de sa valeur absolue (le nombre sans le signe), puis on calcule le complément à 2 de ce nombre.

*Exemple*:

Comment représenter $(-84)_{10}$ en complément à 2 sur 8 bits? On calcule sa représentation en binaire $(84)_{10}$ = $(1\ 010\ 100)_2$

Le nombre se représente donc sur 8 bits avec 01 010 100. (Attention à ne pas oublier d'ajouter autant de bits à 0 à gauche que nécessaire pour avoir une représentation sur 8 bits.) On calcule son complément à 1 :

$C_1$ (01 010 100) = 10 101 011

Finalement on ajoute 1 pour obtenir son complément à 2

$C_2$ (01 010 100) = $C_1$ (01 010 100) + 1 = 10 101 011 + 1 = 10 101 100

$(-84)_{10}$ = 10 101 100 en Complément à 2

- Pour convertir un nombre négatif du complément à 2 vers le décimal on calcule son complément à 2 (on obtient sa valeur absolue), on la convertit en décimal et on rajoute un signe devant le nombre obtenu.

*Exemple*:

Quelle est la représentation en décimal d'un nombre représenté en complément à 2 par :

11 000 011 sur 8 bits?

On calcule le complément à 2. $C_2$ (11000011)=$C_1$ (1100 0011)+1=00111100+1=0011 1101 On convertit le résultat : 111 101 = $(61)_{10}$

On ajoute le signe et on obtient le résultat $(-61)_{10}$.

- En complément à 2 est que les additions peuvent se faire sans s'inquiéter du signe des opérandes.

*Exemple :*

On a vu plus haut que sur 8 bits $(-84)_{10}$ = 10101100. Essayons donc d'ajouter $(23)_{10}$ = $(10111)_2$ à -84

| | | |
|---|---|---|
| | 10 101 100 | $-84$ |
| + | 10 111 | $+23$ |
| = | 11 000 011 | $= -61$ |

La valeur obtenue correspond bien à ce qu'on attendait.

Il faut prendre garde à ne conserver du résultat que le nombre de bits utilisés pour la représentation. L'addition peut produire un bit à 1 supplémentaire qu'il est important d'ignorer.

*Exemple :*

Additionnons les représentations en complément à 2 de -61 et de -23.
Pour cela, il faut commencer par calculer la représentation en complément à 2 de -23 :

$C_2$ (00 010 111) = $C_1$ (00 010 111) + 1 = 11101000 + 1 = 11101001

On peut maintenant poser l'addition :

```
          11 000 011     -61
+         11 101 001     -23
------------------------------
=   1     10 101 100   = -84
```

On ignore le neuvième bit du résultat de l'addition et le résultat obtenu est bien égal à -84.

## IV. Opérations Arithmétiques en Binaire

Chaque chiffre d'un nombre écrit en binaire est appelé bit (*binary digit*). Le bit de droite d'un nombre est appelé : *bit de poids faible* et celui de gauche *bit de poids fort.*

### *4.1 Addition et soustraction*

#### *4.1.1. Binaire :*

L'addition de deux nombres binaires se résume par la table suivante :

| Signe de ± l'opération | | 2ème terme **0** | 2ème terme **1** |
|---|---|---|---|
| 1er terme | **0** | 0 | 1 |
| 1er terme | **1** | 1 | 10 |

La procédure pour effectuer l'addition de deux nombres en binaire est la même que dans le système décimal.

*Exemple* :

```
    1101110
+    100010
-----------
=  (10010000)2
```

```
   1011,011
+   110,01
----------
  10001,101
```

```
  1 0 1 1 1 0 1 1 0 0 0   Premier terme
- 0 0 0 0 1 1 0 0 1 1 1   Second terme
-----------------------
  1 0 1 0 1 1 1 0 0 0 1   Différence
```

### *4.1.2. Octale :*

Comme pour le système binaire, on applique les mêmes règles pour les nombres octaux.

Toutefois, dans ce cas, on aura la retenue **"1"** à gauche à chaque fois que la somme dépasse **7** et il serait bon de se servir du tableau de la figure suivante :

| ± | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| 2 | 2 | 3 | 4 | 5 | 6 | 7 | 10 | 11 |
| 3 | 3 | 4 | 5 | 6 | 7 | 10 | 11 | 12 |
| 4 | 4 | 5 | 6 | 7 | 10 | 11 | 12 | 13 |
| 5 | 5 | 6 | 7 | 10 | 11 | 12 | 13 | 14 |
| 6 | 6 | 7 | 10 | 11 | 12 | 13 | 14 | 15 |
| 7 | 7 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |

*Exemple* :

```
1   1      Retenues
  3 2 6    Premier terme
+ 7 3 5    Second terme
-------
1 2 6 3    Somme
```

```
  5 2 4    Premier terme
- 2 6 3    Second terme
-------
  2 4 1    Différence
```

### *4.1.3. Hexadécimal :*

Si l'on doit additionner deux chiffres hexadécimaux, on utilisera le tableau suivant avec la même méthode que pour les calculs octaux.

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

*Exemple* :

```
   1 1      Retenues
 3 A D 2    Premier terme
+B O F F    Second terme
 E B D 1    Somme
```

```
  5 E 2     Premier terme
- 3 D A     Second terme
  2 0 8     Différence
```

## 4.2. Multiplication

### 4.2.1. Binaire :

La multiplication de deux nombres binaires se résume par la table suivante :

| Signe de X l'opération | | Multiplicateur | |
|---|---|---|---|
| | | **0** | **1** |
| Multiplicande | **0** | 0 | 0 |
| | **1** | 0 | 1 |

La procédure pour effectuer la multiplication de deux nombres en binaire est la même que dans le système décimal.

*Exemple :*

```
      101011
    x 11011
----------
      101011
     101011.
    000000..
   101011...
  101011....
----------
10010001001
```

### 4.2.2. Octal :

La multiplication de deux nombres octaux se résume par la table suivante :

| X | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| **1** | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| **2** | 2 | 4 | 6 | 10 | 12 | 14 | 16 |
| **3** | 3 | 6 | 11 | 14 | 17 | 22 | 25 |
| **4** | 4 | 10 | 14 | 20 | 24 | 30 | 34 |
| **5** | 5 | 12 | 17 | 24 | 31 | 36 | 43 |
| **6** | 6 | 14 | 22 | 30 | 36 | 44 | 52 |
| **7** | 7 | 16 | 25 | 34 | 43 | 52 | 61 |

*Exemple :*

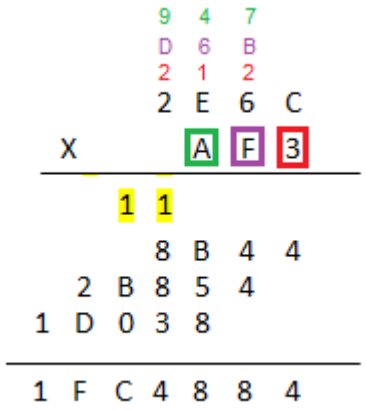
```
    3 1
    3 1
    7 6 2
X     4 5
-----------
  4 6 7 2
3 7 1 0
-----------
4 3 7 7 2
```

### *4.2.3. Octal :*

La multiplication de deux nombres hexadécimaux se résume par la table suivante :

| X | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **1** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| **2** | 2 | 4 | 6 | 8 | A | C | E | 10 | 12 | 14 | 16 | 18 | 1A | 1C | 1E |
| **3** | 3 | 6 | 9 | C | F | 12 | 15 | 18 | 1B | 1E | 21 | 24 | 27 | 2A | 2D |
| **4** | 4 | 8 | C | 10 | 14 | 18 | 1C | 20 | 24 | 28 | 2C | 30 | 34 | 38 | 3C |
| **5** | 5 | A | F | 14 | 19 | 1E | 23 | 28 | 2D | 32 | 37 | 3C | 41 | 46 | 4B |
| **6** | 6 | C | 12 | 18 | 1E | 24 | 2A | 30 | 36 | 3C | 42 | 48 | 4E | 54 | 5A |
| **7** | 7 | E | 15 | 1C | 23 | 2A | 31 | 38 | 3F | 46 | 4D | 54 | 5B | 62 | 69 |
| **8** | 8 | 10 | 18 | 20 | 28 | 30 | 38 | 40 | 48 | 50 | 58 | 60 | 68 | 70 | 78 |
| **9** | 9 | 12 | 1B | 28 | 2D | 36 | 3F | 48 | 51 | 5A | 63 | 6C | 75 | 7E | 87 |
| **A** | A | 14 | 1E | 2D | 32 | 3C | 46 | 50 | 5A | 64 | 6E | 78 | 82 | 8C | 96 |
| **B** | B | 16 | 21 | 2C | 37 | 42 | 4D | 58 | 63 | 6E | 79 | 84 | 8F | 9A | A5 |
| **C** | C | 18 | 24 | 37 | 3C | 48 | 54 | 60 | 6C | 78 | 84 | 90 | 9C | A8 | B4 |
| **D** | D | 1A | 27 | 3C | 30 | 4E | 5B | 68 | 75 | 82 | 8F | 9C | A7 | B6 | C3 |
| **E** | E | 1C | 2A | 42 | 34 | 54 | 62 | 70 | 7E | 8C | 9A | A8 | B6 | C4 | D2 |
| **F** | F | 1E | 2D | 3C | 4B | 5A | 69 | 78 | 87 | 96 | A5 | B4 | C3 | D2 | E1 |

*Exemple :*




## IV. CODAGE DE L'INFORMATION

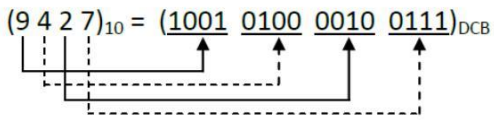
Le codage de l'information est nécessaire pour le traitement automatique de celui-ci. Parmi les codes les plus rencontrés, autre que le code binaire naturel on cite le code **DCB**, le code **excédent trois** …

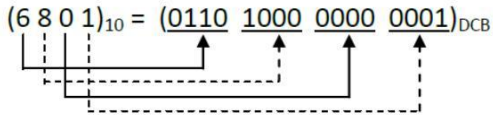
### *1. Le code décimal codé binaire (code DCB)*

Ce code est utilisé dans de nombreux systèmes d'affichage, de comptage ou même les calculatrices de poche. Dans le code BCD chaque chiffre d'un nombre décimal (de $(0)_{10}$ à

$(9)_{10}$ est codé à l'aide de 4 bits (de $(0000)_2$ à $(1001)_2$). Ainsi le code BCD n'utilise que 10 mots de codes de 4 bits.

*Exemple :*



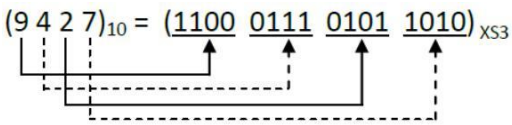

## 2. ***Le code Excédent trois (XS 3)***

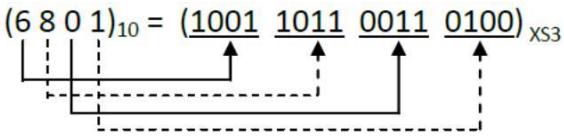
Le code excédent 3 utilise, tout comme le code BCD, 10 mots de codes, auxquels on fait correspondre les 10 chiffres décimaux.

| N10 | NXS3 | N10 | NXS3 |
|---|---|---|---|
| 0 | 0 0 1 1 | 5 | 1 0 0 0 |
| 1 | 0 1 0 0 | 6 | 1 0 0 1 |
| 2 | 0 1 0 1 | 7 | 1 0 1 0 |
| 3 | 0 1 1 0 | 8 | 1 0 1 1 |
| 4 | 0 1 1 1 | 9 | 1 1 0 0 |

Tableau de conversion Décimal  Code à excédent 3

*Exemple :*




Nombres de codes possibles avec N chiffres en base quelconque B

*Partons de l'exemple des nombres décimaux* 1 chiffre ⇒ 10 codes différents

2 chiffres ⇒ 100 codes car pour chaque dizaine on peut associer 10 codes pour les unités

3 chiffres ⇒ 1000 codes ( de 000 à 999)

…

n chiffres ⇒ $10^n$ codes

*En hexadécimal*
1 chiffre ⇒ 16 codes ( de 0 à F)
2 chiffres ⇒ 16 x 16 codes = 256 (00 à FF)

…
n chiffres ⇒ $16^n$ codes possibles

*En binaire*
1 chiffre ⇒ 2 valeurs possibles 0 et1
2 chiffres ⇒ 4 combinaisons possibles

…
n bits ⇒ $2^n$ codes possibles

Nombre de codes possibles avec N chiffres en base B = $B^N$