CHAPITRE I : Les systèmes de numération et de codage

I. Introduction

L'ensemble des outils informatiques sont basés sur les mêmes principes de calcul (loi de tout ou rien). Les calculs habituels sont effectués dans le système de numération décimal, par contre le calculateur électronique ne peut pas utiliser ce système car le circuit électronique ne permet pas de distinguer 10 états. Le système de numération binaire ne comportera que 2 états 0 et 1.

II. Numération

La numération permet de représenter un mot(ou nombre) par la juxtaposition ordonnée de variable (ou symboles) pris parmi un ensemble. Connaitre la numération revient à connaitre le mécanisme qui permet de passer d'un mot à un autre (comptage, opération).

1. Les systèmes de numération

a. Représentation d'un nombre

Dans un système de numérotation en base B, un nombre noté $N_{(B)}$ égal à :

$$N_{(B)} = \sum_{k=0}^{n-1} a_k . B^k = a_{n-1}a_{n-2} \dots a_2 a_1 a_{0\,(B)}$$

Avec :

B : base ou nombre de chiffres différents qu'utilise le système de numérotation.

a_k : Chiffre de rang k

B^k : Pondération associée à a_k

b. Système décimale : (Base10)

Ce système de numération, usuel dans la vie quotidienne, dispose de dix symboles (en l'occurrence des chiffres) qui sont:{0, 1, 2, 3, 4, 5, 6, 7, 8,9}

On parle que l'on travaille en base 10.

Exemple :

$7239 = (7.10^3 + 2.10^2 + 3.10^1 + 9.10^0)_{10}$

c. Système binaire : (Base2)

La numération binaire (ou base 2) utilise deux symboles appelés ***BIT*** (***B***inary dig***IT***) : 0 et 1

Cette base est très commode pour distinguer les 2 états logiques fondamentaux.

On écrit :

$$(a_{n-1}, a_{n-2}, ..., a_1, a_0)_2 = a_{n-1}2^{n-1} + a_{n-2}2^{n-2} + ... + a_1 2^1 + a_0 2^0$$

Exemple :

$(4)_{10} = 1.2^2 + 0.2^1 + 0.2^0 = (100)_2$

$11110010_{(2)} = 1.2^7 + 1.2^6 + 1.2^5 + 1.2^4 + 0.2^3 + 0.2^2 + 1.2^1 + 0.2^0 = 242_{(10)}$

Un code à *n* chiffres en base 2 distingue 2^n états ou combinaisons.

Les puissances successives de 2 (1, 2, 4, 8, 16, 32, …) sont appelées poids binaires.

d. Système octal : Base(8)

Ce système de numération est très peu utilisé de nos jours. Anciennement, il servait au codage des nombres dans les ordinateurs de première génération. Il utilise 8 symboles : 0, 1, 2, 3, 4, 5, 6, 7.

$$(N)_8 = a_{n-1}8^{n-1} + a_{n-2}8^{n-2} + ... + a_1 8^1 + a_0 8^0$$

Exemple:

$(572)_8 = (5.8^2 + 7.8^1 + 2.8^0)_{10} = (378)_{10}$

e. Système hexadécimal : Base(16)

Ce système de numération est très utilisé dans les systèmes ordinateurs et micro ordinateurs ainsi que dans le domaine des transmissions de données. Il comporte 16 symboles les chiffres de 0 à 9 et les lettres $\{A, B, C, D, E, F\}$

$$(N)_{16} = a_{n-1}16^{n-1} + a_{n-2}16^{n-2} + ... + a_1 16^1 + a_0 16^0$$

Exemple:

$(D62C)_{16} = (13.16^3 + 6.16^2 + 2.16^1 + 12.16^0)_{10} = (54828)_{10}$

$F7_{(16)} = F.16^1 + 7.16^0 = 247_{(10)}$

Note bien:

Table de correspondance entre nombre décimaux, binaires, octaux et hexadécimaux :

$N_{(10)}$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$N_{(16)}$	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$N_{(2)}$	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

$N_{(8)}$	0	1	2	3	4	5	6	7	-	-	-	-	-	-	-	-

2. Conversions

a. Conversion du système Décimal vers une base quelconque

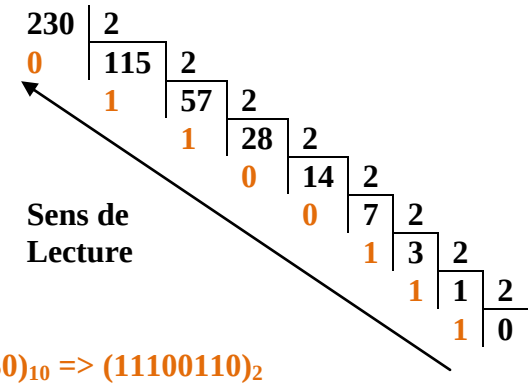
Pour convertir un nombre de la base **10** vers une base **B** quelconques, il faut faire des divisions successives par B et retenir à chaque fois le reste jusqu'à l'obtention à un quotient inférieur à la base B, dans ce cas le nombre s'écrit de la gauche vers la droite en commençant par le dernier quotient allant jusqu'au premier reste.

❖ **Conversion du système Décimal vers le Binaire par division successive**

Pour transférer de la base décimale vers une base$B$, on applique la méthode de division successive. On divise le nombre $N_{(B)}$ que l'on désire convertir par 2, puis on réitère l'opération avec le dividende obtenu jusqu'à son annulation. Le nombre cherché s'écrit en plaçant les restes des divisions successives dans l'ordre inverse de leur obtention (sens de lecture de bas vers le haut).

Exemple :

$(230)_{10}$ à convertir en base 2



Le résultat est donc : $(230)_{10}$ => $(11100110)_2$

❖ **Conversion du système Décimal vers le Binaire par soustraction**

Cette méthode consiste à retrancher du nombre la plus grande puissance de 2 possibles, et ainsi de suite dans l'ordre décroissant des puissances. Si on peut retirer la puissance de 2 concernée, on note (1) sinon on note (0) et on contenue de la même manière jusqu'à la plus petite puissance de 2 possible (20 pour les entiers).

Exemple :

$(230)_{10}$ à convertir en base 2

De	230	On peut retirer	128	reste	102	**1**	Sens
De	102	On peut retirer	64	reste	38	**1**	
De	38	On peut retirer	32	reste	6	**1**	

De	6	On ne peut pas retirer	16	reste	6	**0**
De	6	On ne peut pas retirer	8	reste	6	**0**
De	6	On peut retirer	4	reste	2	**1**
De	2	On peut retirer	2	reste	0	**1**
De	0	On ne peut pas retirer	1	reste	0	**0**

Le résultat est donc : $(230)_{10}$ => $(11100110)_2$

b. Conversion du système Binaire vers l'hexadécimal

Pour convertir du binaire vers l'hexadécimal, on divise le nombre binaire en tranches de 4, en partant de la droite pour la partie entière et en partant de la gauche pour la partie fractionnaire. Chacun des paquets est ensuite converti en hexadécimal.

Exemple :

$(110101110001)_2 = (1101\ \ 0111\ \ 0001)_2 = (D71)_{16}$

c. Conversion du système Hexadécimal vers le Binaire

C'est le processus directement inverse, on écrit chaque quartet sur 4 bits en complétant éventuellement avec des zéros.

Exemple :

$(FA3)_{16} = (1111\ 1010\ 0111)_2$

d. Conversion du système Binaire vers l'Octal et inversement

On reprend les mêmes principes de la conversion Binaire-Hexadécimal et Hexadécimal-Binaire mais cette fois ci en groupant les données en tranches de 3.

Exemple:

$(101010)_2 = [101]_2\ \ [010]_2 = (52)_8$

NB: pour la conversion Octal-Hexadécimal et Hexadécimal-Octal, la plus simple méthode et de passer par le système Binaire,

Exemple:

$(34.61)8 = (011100,110001)_2 = (1C.C4)_{16}$.

3. Représentation des nombres comportant une partie fractionnaire

a. Conversion de la base 10 vers une Base quelconque

Principe de conversion

- **Partie entière :**

  Divisions entières successives par la base (condition d'arrêt : quotient nul).

  Lecture du reste

- **Partie fractionnaire :**

  Multiplications successives par la base (condition d'arrêt : partie fractionnaire nulle).

  Lecture de la partie entière

Exemple:

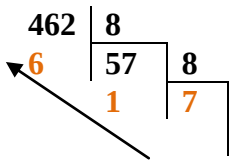
Soit à convertir le nombre $(462,625)_{10}$ vers une base quelconque .Pour résoudre ce problème il faut procéder comme suit :

Convertir la partie entière (462)

Convertir la partie fractionnaire en faisant des multiplications successives par la base et en conservant à chaque fois le chiffre devenant entier.

$(462,625)_{10=}$ $(?)_8$, $(462,625)_{10=}$ $(?)_2$, $(462,625)_{10=}$ $(?)_{16}$

Partie entière



$(462)_{10}$ => $(716)_8$

Partie fractionnaire

0,625*8=5 ,00

Le résultat est donc : $(462, 625)_{10}$ =$(716,5)_8$

$(462)_{10}$ =$(716)_8$=$(111001110)_2$

0,625*2=1,25

0, 25*2= 0,5

0, 5*2= 1,0

Le résultat est donc : $(462, 625)_{10}$ =$(111001110,101)_2$

$(462)_{10}$ =$(111001110)_2$= $(1CE)_{16}$

0,625*16=10 ,00

Le résultat est donc : $(462, 625)_{10}$ = $(1CE, A)_{16}$

Remarque :

Parfois en multipliant la partie décimal par la Base B, on n'arrive pas à convertir toute la partie entière .ceci est du essentiellement au fait que le nombre à convertir n'a pas un équivalent exact dans la Base B et sa partie décimale est cyclique.

Exemple

0,1 5*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

Le résultat est donc : $(0, 15)_{10} = (0, 0010011001\ldots)_2$
On dit $(0,15)_{10}$est cyclique dans la Base 2 de période 1001

b. Conversion d'une Base quelconque vers la Base 10

Pour ce type de conversion, il suffit de représenter le nombre par une combinaison linéaire des puissances successives de la Base et faire la somme, le résultat ainsi trouvé s'écrit directement dans la BASE 10.

Exemple

$(0,001011)_2 = 0*2^{-1}+0*2^{-2}+1*2^{-3}+0*2^{-4}+1*2^{-5}+1*2^{-6} = (0,171875)_{10}$

$(0,32)_8 = 3*8^{-1}+2*8^{-2} = (0,40625)_{10}$

$$(AF,19)_{16} = 10 * 16^1 + 15 * 16^0 + 1 * 16^{-1} + 9 * 16^{-2} = (175,09765625$$

4. Représentation des nombres signés

La plupart des dispositifs numériques traitent également les nombres négatifs. Le signe (+) ou (-) est identifié par un bit, dit le bit de signe et le nombre ainsi formé est dit signé . On peut identifier trois principales façons de représenter les nombres négatifs:

1. Représentation en valeur absolue et signe (VAS).
2. Représentation par le complément restreint appelé complément à 1.
3. Représentation par le complément vrai appelé complément à 2.

❖ **La représentation en valeur absolue et signe**

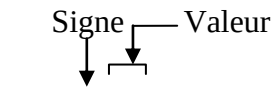
Il s'agit ici d'utiliser un bit pour représenter le signe de la valeur à représenter. Selon que le nombre est positif ou négatif, le bit d'extrême gauche prendra par convention la valeur 0 ou la valeur 1 (0 : positif, 1 : négatif). Par exemple, sur 4 bits, 1 bit sera réservé au signe et trois bits seront utilisés pour représenter les nombres en valeur absolue:

Sur n bits:

- Signe : bit de poids fort (0 : positif, 1 : négatif)
- Valeur absolue : n − 1 bits
- Intervalle de valeurs représentées : $[-2^{n-1} + 1, 2^{n-1} - 1]$

Exemple :

Sur 3 bits, l'intervalle de valeurs représentées : [−3, + 3]



0 0 0	0
0 0 1	1
0 1 0	2
0 1 1	3
1 0 0	-0
1 0 1	-1
1 1 0	-2
1 1 1	-3

Inconvénients: Cette méthode impose que le signe soit traité indépendamment de la valeur. Il faut donc des circuits différents pour l'addition et la soustraction. De plus, on obtient deux représentations différentes pour 0, soit +0 et -0.

❖ **La notation en complément à 1**

On pourrait définir le complément à 1 comme ce qu'il faut ajouter à une valeur pour obtenir la valeur maximale représentable sur le nombre de bits disponibles. On appel complément à un d'un nombre N un autre nombre N' tel que :

$$N+N'=2^n-1$$

n : est le nombre de bits de la représentation du nombre N.

Exemple :

Soit N=1010 sur 4 bits donc son complément à un de N :

$N'= (2^4 - 1)-N$

$N'= (16-1)-(1010)_2= (15) - (1010)_2 = (1111)_2 – (1010)_2 = 0101$

On constate que le complément à 1 d'un nombre binaire se trouve simplement en remplaçant les 0 par des 1 et les 1 par des 0.

Notons que l'utilisation du complément à 1 pour représenter les nombres négatifs nous donne encore une double représentation pour le 0.

Exemple :

Valeur	Complément à 1
000	111
001	110

010	101
011	100

Exemple :

On va déterminer la valeur décimale représentée par la valeur 101011 en complément à 1 sur 6 bits :

Le bit poids fort indique qu'il s'agit d'un nombre négatif. Le complément à 1 de la valeur (101011)

-CA1 (101011) = - $(010100)_2$= - $(24)_{10}$

❖ **La notation en complément à 2**

La représentation en complément à deux (complément à vrai) est la représentation la plus utilisée pour la représentation des nombres négatifs dans la machine.

Le complément à 2 d'une valeur binaire est ce qu'il faut ajouter à cette valeur pour qu'elle atteigne une unité de plus que la valeur maximale qu'on peut représenter sur n bits. C'est donc le (complément à 1) + 1.

Cette technique élimine le problème de la double représentation du 0 (+0 et -0) comme c'est le cas dans la représentation "signe et valeur absolue" ou celle du complément à 1.Cela s'explique parce que

le complément à 2 permet d'éliminer la double représentation de 0 tout en gardant la facilité de reconnaître le signe par le bit d'extrême gauche. Notons que le complément à 2 du complément à 2 d'un nombre redonne le nombre.

Ainsi, sur 4 bits, avec le signe représenté sur le bit le plus significatif et 3 bits qui permettent de représenter les valeurs, on peut représenter les entiers de -8 à 7, soit un entier négatif de plus qu'un complément à 1.

Sur n bits:

- Complément à 1 : + 1. $|x| + (-|x|) = 2^n$
- Intervalle de valeurs représentées : $[-2^{n-1}, 2^{n-1} - 1]$

Exemple :

Valeur	Complément à 2
001	111
010	110
011	101

5. Opérations arithmétiques

a. L'addition

Il suffit de savoir que :

0+0=0

0+1=1

1+0=1

1+1=10

Et d'effectuer éventuellement une retenue comme dans le cas d'une addition décimal

Exemples:

	1	**0**	**1**	**1**	**0**	**1**
+		**1**	**0**	**0**	**1**	**0**
	1	**1**	**1**	**1**	**1**	**1**

	1	**1**	**1**	**1**	**1**
		1	**1**	**0**	**1**	**1**
+			**1**	**1**	**0**	**1**
	1	**0**	**1**	**0**	**0**	**0**

Remarque: L'addition s'effectue de la même manière dans les autres bases.

Exercice:

Effectuer les opérations suivantes:

$(37)_8 + (65)_8 + (116)_8 = (242)_8$

$(D5E)_{16} + (2F36)_{16} = (3C94)_{16}$

b. La soustraction

On peut opérer comme dans la soustraction décimale. Voilà ci dessous la table de soustraction binaire:

0-0=0

0-1=1 avec un retenue de 1

1-0=1

1-1=0

Exemple:

	1		**1**	**1**			
	1	**0**	**1**	**0**	**0**	**1**	**1**
-		**1**	**0**	**1**	**1**	**0**	**1**
	0	**1**	**0**	**0**	**1**	**1**	**0**

Remarque: la soustraction s'effectue de la même manière dans les autres bases.

Exercice:

Effectuer les opérations suivantes:

$(137)_8 - (63)_8 = (54)_8$

$(F23)_{16} - (2A6)_{16} = (C7D)_{16}$

$(FD28)_{16} - (E5E)_{16} - (2F36)_{16} = (FD28)_{16} - [\ (E5E)_{16} + (2F36)_{16}\ ] = (FD28)_{16} - (3D94)_{16} = (BF94)_{16}$

c. La multiplication

La multiplication en binaire est très simple, voilà la table de multiplication:

$$0 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 0 = 0$$
$$1 \times 1 = 1$$

Remarque: On doit bien tenir compte des décalages.

Exemple:

					1	**0**	**1**	**1**
					1	**1**	**0**	**1**
×					**1**	**0**	**1**	**1**
			1	**0**	**1**	**1**		
+		**1**	**0**	**1**	**1**			
	1	**0**	**0**	**0**	**1**	**1**	**1**	**1**

Exercice:

Effectuer les opérations suivantes:

$(237)_8 * (63)_8 = (17655)_8$

		2	**3**	**7**
×			**6**	**3**
+		**7**	**3**	**5**
	1	**6**	**7**	**2**	
=	**1**	**7**	**6**	**5**	**5**

$(F3)_{16} * (206)_{16} = (1EBB2)_{16}$

d. La division

La division entre deux nombres binaires est identique à la division euclidienne.

Exemple:

1	**1**	**1**	**0**	**1**	**1**	**0**	**1**
1	**0**	**1**			**1**	**0**	**1**
	1	**0**	**0**	**1**			
		1	**0**	**1**			
		1	**0**	**0**			

Il suffit en fait de soustraire 101 lorsqu'on le peut, et d'abaisser le chiffre suivant :

$$11101 = 101 \times 101 + 100$$

6. Les codes

a. Notion de codage

On appelle codage l'opération qui consiste à faire correspondre à tout caractère (lettre, chiffre, signe,...) un symbole où un ensemble de symboles particuliers appelés mot de code.

b. Codes binaires

❖ **Les codes numériques pondérés:**

✓ **les codes binaires purs:**

Ceux sont des codes qui donnent à chaque combinaison une équivalence décimale et dans laquelle chaque rang d'élément binaire à un poids précis. Le code binaire naturel et ses dérivés (octal et hexadécimal) répondent aux règles classiques de l'arithmétique des nombres positifs.

Exemple:

$(7)_{10}$ sur 8 bits = $(0000\ 0111)_2$

✓ **les codes DCB (Décimal Codé Binaire):**

Ce code DCB, en Anglais BCD (Binary Coded Decimal), consiste à représenter chaque chiffre d'un nombre décimal par son équivalent binaire sur 4 bits.

Exemple

7	2	3	9	**code décimale**
↓	↓	↓	↓	
0111	0010	0011	1001	**code B.C.D**

❖ **Les codes numériques non pondérés:**

✓ **Le code Gray:**

Appelé aussi code binaire réfléchi, il appartient à la famille dite « codes à distance » de faite qu'une représentation codé ne diffère de celle qui la précède que par un bit comme le montre le tableau ci-dessous.

Code décimale	Code binaire	Code Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

- ✓ **Codes alphanumériques (codage des caractères)**

Les caractères également doivent être représentés en binaire de manière unique. La convention adoptée est d'associer à chaque caractère un nombre décimal et de convertir ce nombre en binaire. Il existe plusieurs normes:

Code ASCII:

Le code ASCII (ASCII = American Standard Code for Information Interchange) est initialement un code à 7 bits, qui permet le codage de 128 caractères. Il englobe des lettres, des chiffes, des signes de ponctuations et un certain nombre de signaux de commande. Toutes ces correspondances sont fixées par l'American National Standards Institutes

Code ASCII (8 bits):

Ce code est normalisé ISO (International Standard Organisation). C'est une extension du code ASCII étendu. Il permet le codage de caractères sur 8 bits, soit 256 caractères possibles.

UNICODE (16 bits):

Ce code permet de représenter des caractères appartenant à plusieurs langues (arabes, hébreu, japonais, coréen,...) : 65536 caractères.

Code EBCDIC (IBM).

Le code EBCDIC (Extended Binary Decimal Interchange Code) est un code à 8 éléments binaires utiles, soit 256 combinaisons possibles.

- Table du code ASCII standard (7 bits):

	000	***001***	***010***	***011***	***100***	***101***	***110***	***111***
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

Exemple

$A => (65)_{ASCII} \Rightarrow (01000001)_2 \Rightarrow (41)_H$

$[=> (91)_{ASCII} \Rightarrow (01011011)_2 \Rightarrow (5B)_H$

Le code à barres

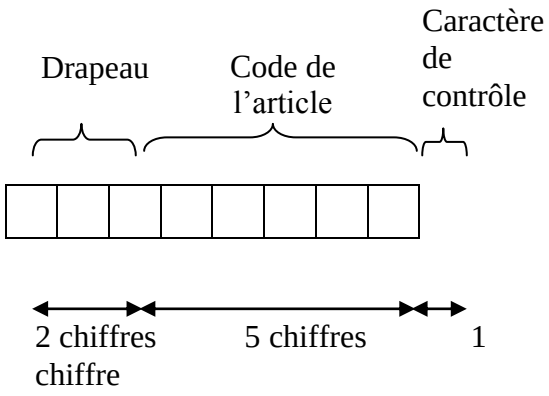
Le code à barres qui figure sur la plupart des emballages des produits de consommation courante est la fiche d'identité, traduite en code, du produit sur lequel il est apposé. Il peut indiquer le pays d'origine, le nom du fabricant, celui du produit, sa référence.

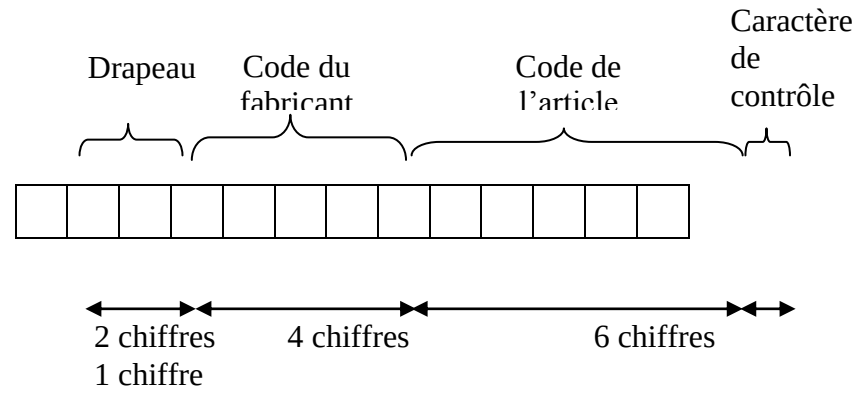
Il permet de suivre la traçabilité du produit. Le code imprimé parfois directement sur l'emballage, se présente également sous la forme d'une étiquette rectangulaire collée. Il est

composé de barres et d'espaces larges ou étroits dont le nombre correspond à un ensemble de données numériques ou alphanumériques.

Ce marquage comporte un certain nombre de barres verticales, ainsi que des chiffres au nombre de 13. Le premier chiffre indique le pays d'origine, les 5 suivants sont ceux du code du fabricant, les 6 autres

ceux du code de l'article, le 13^{e} est une clé de contrôle. Les barres sont le codage de ces chiffres sur 7 bits. A chaque chiffre est attribué un ensemble de 7 espaces, blanc ou noirs.

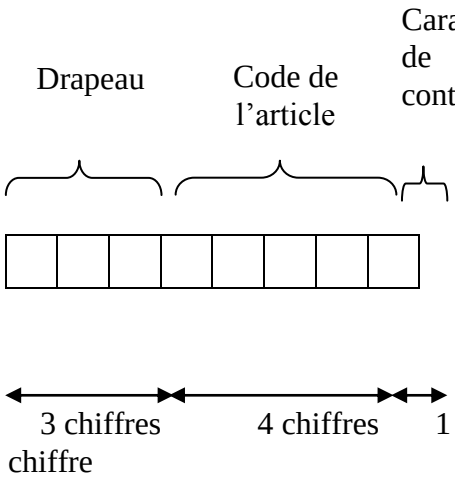




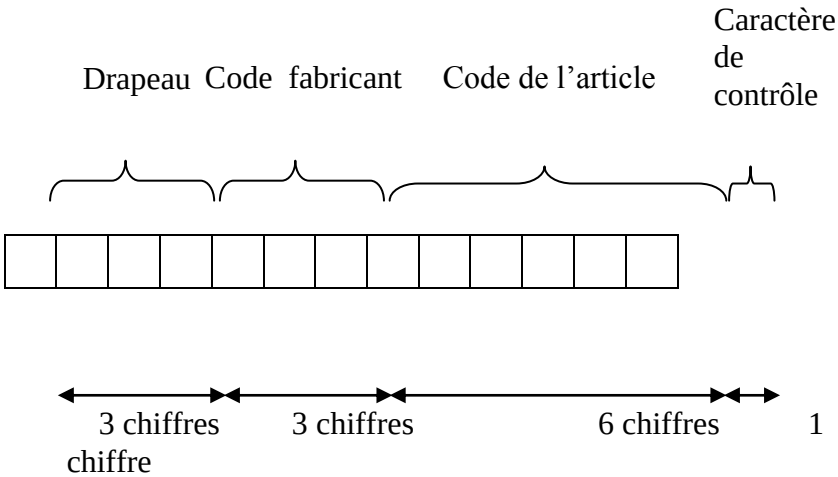
Remarque :

Dans le cas où le code pays comporte 3 caractères, le code fabricant ne comporte que 3 caractères.

Exemples :



La norme EAN 8 chiffres



La norme EAN 13 chiffres