1. Numération & Codage

1. Variables et mots

- **Variable discrète :**

  Une variable discrète est une variable qui prend ses valeurs dans un ensemble S de cardinalité C finie, non vide.

- **Variable binaire (ou variable commutation) :**

  C'est une variable discrète dont la cardinalité de l'ensemble S est C=2. Les éléments de l'ensemble S sont les valeurs que peut prendre la variable discrète. Ces éléments sont notés de façon complémentaire : « Vrai / Faux », « Oui / Non », « Ouvert / Fermé », « 0 / 1 ».

  Une variable « a » est binaire : $S = \{0,1\}$ $a \in S$ $a = 0$ OU $a = 1$

- **Combinaison de « n » variables discrètes :**

  Une combinaison de « n » variables discrètes est une <u>juxtaposition ordonnée</u> de ces variables prises dans l'ensemble des ses valeurs. Cette combinaison s'appelle « MOT »

  Exemple : $S = \{0,1,2,3\}$

  22131 est une des combinaisons de 5 variables de S

- **Mot binaire :**

  Un mot binaire de n **digits** est un ensemble ordonné de n variables binaires.

  Digit ou Binary digit (BIT) = nombre valant 0 ou 1

  Si on prend n variables binaires, on aura 2^n combinaisons possibles, c'est à dire 2^n écritures différentes de mots.

  Exemple :

  Avec n = 3, on a $2^3 = 8$ combinaisons : 000, 001, 010, 011, 100, 101, 110, 111

2. Numération

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

- Les systèmes de numération les plus courants sont :

  - Système <u>décimal</u> : il comprend 10 symboles appelés chiffres : {0,1,2,3,4,5,6,7,8,9}
  - Système <u>binaire</u> : il comprend 2 symboles appelés BIT (Binary digIT) : 0 et 1
  - Système <u>octal</u> : il utilise 8 symboles qui sont les chiffres de 0 à 7,
  - Système <u>hexadécimal</u> : 16 symboles : Les chiffres de 0 à 9 et les lettres A, B, C, D, E, F

- Le nombre de symboles que possède le système de numération est appelé **Base**

- Lorsqu'un mot (ou nombre) est écrit, la **position respective** des symboles détermine leurs **poids**.

- Le système décimal, appelé aussi système à base 10, est dit à poids positionnels : c'est à dire que la valeur d'un chiffre dépend de sa position (appelée rang) dans le nombre :

  …, centaines, dizaines, unités, dixièmes, centièmes, …

  Exemple :

  $742,59 = 7\times100 + 4\times10 + 2\times1 + 5\times0,1 + 9\times0,01$

  Ou encore : $742,59 = 7\times10^2 + 4\times10^1 + 2\times10^0 + 5\times10^{-1} + 9\times10^{-2}$

  - 7 représente le chiffre de poids fort,
  - 9 est le chiffre de poids faible
  - Les poids des rangs sont des puissances de 10 (système décimal)

3. Codage

- **Définition :**

  Un nombre décimal est un nombre exprimé dans le système décimal ; c'est à dire à base 10. Le codage est l'opération de transformation de l'écriture d'un nombre décimal dans une base B quelconque.

- **Ecriture dans une base B :**

Tout entier décimal N peut s'écrire dans une base B quelconque. B est la cardinalité de S.

Un nombre N s'écrit en juxtaposant n symboles :

$$N = (a_{n-1}\ a_{n-2} \dots a_0)_B \quad \text{où les } a_i \in S$$

$$0 \leq a_i \leq B-1$$

Ce nombre N a pour valeur décimale :

$$N = a_{n-1}.B^{n-1} + a_{n-2}.B^{n-2} + \dots + a_1.B^1 + a_0.\ B^0 = \sum_{i=0}^{n-1} a_i.B^i$$

$$0 \leq N \leq B^n-1$$

- Cette forme est appelée forme polynomiale.
- L'élément a_i est le symbole de rang i et son poids est B^i.
- a_{n-1} est le symbole le plus significatif (de poids le plus fort)
- a_0 est le symbole le moins significatif (de poids le plus faible)
- Les termes $\mathbf{B^i}$ sont appelés **coefficients de pondération** ou Poids
- N est codé sur n bits

- **Codages courants :**

Codage	B	a_i
Binaire	2	0, 1
Ternaire	3	0, 1, 2
Octal	8	0, 1, 2, 3, 4, 5, 6, 7
Hexadécimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

4. Codage binaire (B = 2)

- Le système binaire (ou la numération à base 2) est le système de codage utilisé en électronique numérique et son application aux systèmes informatiques. On dispose de 2 symboles {0,1} appelés bits.

- Lorsqu'on code en binaire, on cherche généralement à utiliser un nombre fixe de bits: On parle alors de format de 4 bits, de 8 bits (octet) , de 16 bits, 32, 64 bits...

- Format de n bits : il permet de représenter tous les nombres entiers N compris entre 0 et 2^n-1. Ce nombre au format de n bits s'appelle communément « mot de n bits »

- Exemple : mot de 4 bits :

$$(1101)_2 = 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0$$
$$(1101)_2 = 1\times\mathbf{8} + 1\times\mathbf{4} + 0\times\mathbf{2} + 1\times\mathbf{1} = 13$$

- Le nombre binaire 1101 représente le nombre décimal 13
- le chiffre 1 à gauche représente le bit de poids fort
- le chiffre 1 à droite représente le bit de poids faible
- Les poids des rangs sont des puissances de 2
- On dit que le mot de 4 bits est pondéré 8-4-2-1

- **Comptage binaire :**

Si on utilise 4 bits pour coder les nombre décimaux :
- Il existe 16 combinaisons (2^4=16)
- On peut alors compter de 0 à 2^4 - 1 = 15
- Le bit de poids faible (celui de droite) sera pondéré 2^0 = 1
- Le bit de poids fort (celui de gauche) sera pondéré 2^3 = 8

Tableau de la suite des nombres binaires :

N	$\mathbf{a_3}$	$\mathbf{a_2}$	$\mathbf{a_1}$	$\mathbf{a_0}$
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

- **Méthodes de codage d'un nombre N en binaire :**

Méthode 1 :
- On sait que $0 \leq N \leq B^n-1$ et $B = 2$
- On détermine le nombre n de bits minimum pour coder N.
- On positionne les bits a_i à 0 ou 1 de telle façon que

$$a_{n-1}.2^{n-1} + a_{n-2}.2^{n-2} + \ldots + a_1.2^1 + a_0.\ 2^0 = N$$

Exemple : On veut coder 18

- Codage sur 1 bit ($n = 1$) : on compte de 0 à $2^1 - 1 = 1$,
- Codage sur 2 bits ($n = 2$) : on compte de 0 à $2^2 - 1 = 3$,
- Codage sur 3 bits ($n = 3$) : on compte de 0 à $2^3 - 1 = 7$,
- Codage sur 4 bits ($n = 4$) : on compte de 0 à $2^4 - 1 = 15$,
- Codage sur 5 bits ($n = 5$) : on compte de 0 à $2^5 - 1 = 31$,

→ Il faut 5 bits pour coder le nombre 18 :

16	8	4	2	1
1	0	0	1	0

$18 = (10010)_2$

Méthode 2 : Méthode des divisions euclidiennes successives

Soit $N = a_{n-1}.B^{n-1} + a_{n-2}.B^{n-2} + \ldots + a_1.B^1 + a_0.\,B^0$ avec $B = 2$

Effectuons les divisions euclidiennes successives de N par 2 jusqu'à ce que le quotient devienne nul :

$$(a_{n-1}.2^{n-1} + a_{n-2}.2^{n-2} + \ldots + a_1.2 + a_0) = \underbrace{(a_{n-1}.2^{n-2} + a_{n-2}.2^{n-3} + \ldots + a_1)} \times 2 + \boxed{a_0}$$

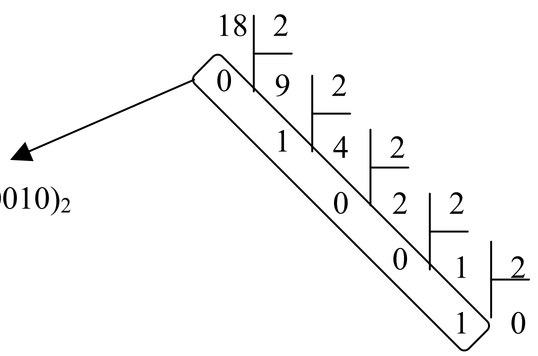
$$\overbrace{(a_{n-1}.2^{n-2} + a_{n-2}.2^{n-3} + \ldots + a_1)} = \underbrace{(a_{n-1}.2^{n-3} + a_{n-2}.2^{n-4} + \ldots + a_2)} \times 2 + \boxed{a_1}$$

$$\vdots$$

$$\overbrace{(a_{n-1})} = (0) \times 2 + \boxed{a_{n-1}}$$

⇒ Les restes des divisions successives forment le mot binaire du poids le plus faible au poids le plus fort.

Exemple : On veut coder 18



$18 = (10010)_2$

5. Codage octal (B = 8)

- La numération octale (à base 8) est utilisée par les informaticiens. On dispose de 8 symboles qui ne sont autres que les chiffres de 0 à 7. Si on code sur n chiffres, on peut représenter tous les nombres entiers décimaux compris entre 0 et 8^n-1.

- Exemple :

Mot octal de 3 chiffres. Il sera pondéré 8^2-8^1-8^0 soit : 64-8-1

$$(721)_8 = 7\times 8^2 + 2\times 8^1 + 1\times 8^0$$

$$(721)_8 = 7\times \mathbf{64} + 2\times \mathbf{8} + 1\times \mathbf{1} = 465$$

Le nombre $(721)_8$ représente le nombre décimal 465

- **Codage d'un nombre N en octal :**

Comme pour la conversion en binaire, on effectue ici la méthode des divisions euclidiennes successives par 8

Exemple : convertir le nombre décimal 99 en octal

99 = 8 × 12 + 3 (chiffre de poids faible)
12 = 8 × 1 + 4
1 = 8 × 0 + 1 (chiffre de poids fort)

→ 99 = $(143)_8$

- **Passage du Binaire en Octal (Encodage) :**

Il suffit de faire des regroupements de 3 bits sur le mot binaire. En effet, un mot binaire de 3 bits permet de coder les nombres entiers décimaux compris entre 0 et 7:

$$N = \underbrace{a_{n-1}.2^{n-1} + a_{n-2}.2^{n-2} + a_{n-3}.2^{n-3}} + \ldots + \underbrace{a_3.2^3} + \underbrace{a_2.2^2 + a_1.2^1 + a_0.\,2^0}$$

$$N = a'_{n-1}.8^{n-1} + \ldots \quad a'_1.8^1 \quad + \quad a'_0.8^0$$

Exemple 1 : $39 = (\underbrace{100}_{4}\ \underbrace{111}_{7})_2$

Exemple 2 : soit le nombre binaire 1110111

Il faut regrouper 3 bits sur ce mot à partir de la droite: 1 110 111
On complète alors par des zéros à gauche pour avoir 3 bits : 001 110 111
Ce qui donne en tenant compte de la pondération 4-2-1 des bits respectifs:
D'où le résultat : $(1110111)_2 = (167)_8$

6. Codage Hexadécimal (B = 16)

- La numération hexadécimal (à base 16) est apparue avec la logique programmée. Elle est largement utilisée en programmation.

- On dispose de 16 symboles: les chiffres de 0 à 9 et les lettres de A à F qui correspondent aux valeurs décimales 10 à 15

- Si on code sur n chiffres, on peut représenter tous les nombres entiers décimaux compris entre 0 et 16^n-1.

- Voici l'équivalence des codages dans les 3 numérations (décimal, hexadécimale) :

N	Héxa	N	Héxa	N	Héxa	N	Héxa	N	Héxa
0	0	**5**	5	**10**	A	**15**	F	**20**	14
1	1	**6**	6	**11**	B	**16**	10	**21**	15
2	2	**7**	7	**12**	C	**17**	11	**26**	1A
3	3	**8**	8	**13**	D	**18**	12	**32**	20
4	4	**9**	9	**14**	E	**19**	13	**100**	64

- **Exemple :**

Mot hexadécimal de 4 symboles. Il sera pondéré 16^3-16^2-16^1-16^0 soit : 4096-256-16-1

$(20AC)_{16} = 2\times16^3 + 0\times16^2 + A\times16^1 + C\times16^0$

$(20AC)_{16} = 2\times\mathbf{4096} + 0\times\mathbf{256} + 10\times\mathbf{16} + 12\times\mathbf{1} = 8364$

Le nombre $(20AC)_{16}$ représente le nombre décimal 8364

- **Codage d'un nombre N en octal :**

Comme pour la conversion en binaire, on effectue ici la méthode des divisions euclidiennes successives par 16

Exemple : convertir le nombre décimal 92 en hexadécimal

92 = 16 × 5 + 12 (chiffre de poids faible = C)
5 = 16 × 0 + 5 (chiffre de poids fort)

→ $92 = (5C)_{16}$

- **Passage du Binaire en Hexadécimal (Encodage) :**

  ⇒ Il suffit de faire des regroupement de 4 bits sur le mot binaire car avec 4 bits il est possible de coder les nombres de 0 à 15:

$$N = \underbrace{a_{n-1}.2^{n-1} + a_{n-2}.2^{n-2} + a_{n-3}.2^{n-3}} + \ldots + \underbrace{a_3.2^3 + a_2.2^2 + a_1.2^1 + a_0.\ 2^0}$$
$$N = a''_{n-1}.16^{n-1} + \ldots + a''_0.16^0$$

Exemple :

Soit le nombre binaire 100111.
Il faut regrouper 4 bits sur ce mot à partir de la droite: 10 0111
On complète alors par des zéros à gauche pour avoir 4 bits : 0010 0111
Ce qui donne en tenant compte de la pondération de chaque bit 8-4-2-1:
0010 = 2
0111 = 7

D'où le résultat : $(100111)_2 = (27)_{16}$

7. Autres codages binaires

7.1 Codage quelconque : exemple de codage binaire auto-complémenté

- Code pondéré avec des poids p_i entiers (positif ou négatif) au lieu de B^i (avec i =0..n-1)

$$N = a_{n-1}.\ p_{n-1} + \ldots + a_2.\ p_2 + a_1.p_1 + a_0.\ p_0 \qquad a_i \in \{0,1\},\ p_i \in \mathbb{Z}$$

- exemple :

	Code binaire naturel				**Code binaire auto-complémenté**			
N	**8**	**4**	**2**	**1**	**8**	**4**	**-2**	**-1**
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	1	1	1
2	0	0	1	0	0	1	1	0
3	0	0	1	1	0	1	0	1
4	0	1	0	0	0	1	0	0
5	0	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	0
7	0	1	1	1	1	0	0	1
8	1	0	0	0	1	0	0	0
9	1	0	0	1	1	1	1	1

Complément à 1 :
0 → 1
1 → 0

Symétrie

7.2 Codage BCD (Binary Code Decimal)

- Il faut prendre le codage binaire naturel des 10 chiffres décimaux.

- Chaque chiffre d'un nombre décimal est donc codé sur un mot binaire de 4 bits, car il faut 4 bits pour coder jusqu'à 9 . Or 4 bits permettent de coder les nombres de 0 à 15 : Il ne faut donc pas tenir compte du codage dépassant 9.

0 : 0000	1 : 0001	2 : 0010	3 : 0011	4 : 0100	
5 : 0101	6 : 0110	7 : 0111	8 : 1000	9 : 1001	10 à 15 : interdit

- **Exemple** : Codage d'un entier 234 en BCD : $(0010\ 0011\ 0100)_{BCD}$

- **Opération d'addition** :

Binaire naturel

9	=	1	0	0	1
+ 4	=	0	1	0	0
13	=	1	1	0	1

Binaire naturel → BCD

13	=	(hors norme)	1	1	0	1
(+6)	=	(décalage)	0	1	1	0

13 = $(0\ 0\ 0\ 1\ 0\ 0\ 1\ 1)_{BCD}$

7.3 Codage Complément à 2

- Définitions :

  - le complément à 2 = codage des nombres signés (+ et -),

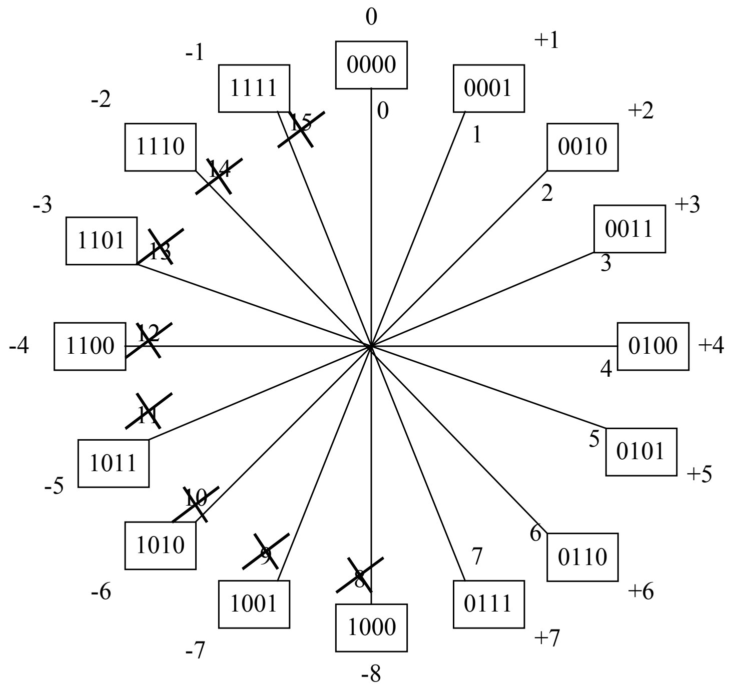
  - Nombre binaire de n bits → Codage des nombres entiers décimaux de 0 à 2^n-1

  - Codage des nombres signés N à partir de n bits : $-2^{n-1} \leq N \leq 2^{n-1}-1$:
    - codage du nombre sur (n-1) bits
    - 1 bit de signe (celui de gauche : 0 : positif , 1 : négatif)

- Exemple : n = 4 :

  Il est possible de coder :

  - Nombres non signés sur 4 bits : 2^4=16 : 0 à 16
  - Nombres signés sur 4 bits : on « coupe en 2 » 16 / 2 = 8 : -8 à +7



- Comment trouver le complément à 2 d'un nombre binaire (l'opposé du nombre décimal):

  - **Méthode 1 :**

    on affecte de poids négatif le bit de poids fort:

    $(1101)c_2 = 1\times(-2^3) + 1\times2^2 + 0\times2^1 + 1\times2^0$

    $(1101)c_2 = 1\times\mathbf{(-8)} + 1\times\mathbf{4} + 0\times\mathbf{2} + 1\times\mathbf{1} = -3$

  - **Méthode 2 :**
    - on prend le complément à 1 du nombre (les 0 deviennent des 1 et les 1 des 0)
    - on ajoute 1

Exemple :

	$(0110)_2 = 6$	:	Nombre 6
	$(1001)_2 = \overline{6}$	:	complément à 1 de 6
+1 :	$(0001)_2$	:	On ajoute 1
	$(1010)_2$	:	On obtient (-6)

  - **Méthode 3 :**
    - 1. On recopie le nombre binaire de la droite vers la gauche jusqu'au premier 1
    - 2. On complémente les autres (les 0 deviennent des 1 et les 1 des 0) :

$(0110)_2 = 6$

$(1010)_2 = -6$