Chapitre 2 :
Systèmes de numération et codage numérique

Justine Philippe

ISEN | école d'ingénieurs

Sommaire

- Système de numération
- Système binaire
- Détection d'erreur
- Codage numérique

Sommaire

- Système de numération
- Système binaire
- Détection d'erreur
- Codage numérique

Définitions

- Un système de numération décrit la façon avec laquelle les nombres sont représentés

- Un système de numération est défini par :
  - Un alphabet A : ensemble de symboles ou chiffres
  - Des règles d'écriture des nombres : juxtaposition de symboles

- **Exemple : Numération décimale**
  - C'est le système de numération le plus pratiqué actuellement
  - L'alphabet est composé de dix chiffres : A = {0,1,2,3,4,5,6,7,8,9}
  - Le nombre 10 est la base de cette numération
  - C'est un système positionnel. Chaque position possède un poids.
    ex : $4134 = 4x10^3 + 1x10^2 + 3x10^1 + 4x10^0$

Définitions

- Un système de numérotation positionnel pondéré à base b est défini sur un alphabet de b chiffres (ou digit) :

$$A = \{c_0, c_1, \dots, c_{b-1}\} \; avec \; 0 \leq c_i < b$$

- Soit $N = (a_{n-1}a_{n-2} \dots a_1 a_0)_b$: représentation en base b sur n chiffres
  - a_i : chiffre de l'alphabet de poids i (position i)
  - a_0 : chiffre de poids 0 appelé chiffre de poids faible
  - a_{n-1} : chiffre de poids n-1 appelé chiffre de poids fort

- La valeur de N en base 10 est donnée par :

$$(N)_{10} = a_{n-1}b^{n-1} + a_{n-2}b^{n-2} + \cdots + a_0 b^0 = \sum_{i=0}^{n-1} a_i b^i$$

$(101)_2 \rightarrow (\;?\;)_{10} \Rightarrow 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 4 + 0 + 1 = (5)_{10}$

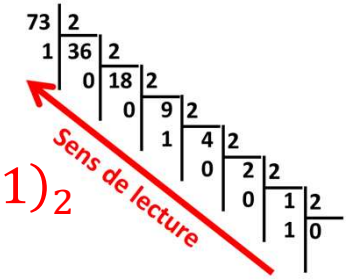
Bases de numération

Nom	Dimension	Symboles	Exemple $(496)_{10}$	Remarques
Décimale	10	{0,1,2,3,4,5,6,7,8,9}	496	-
Binaire	2	{0,1}	111110000	C'est avec ce système que fonctionnent les ordinateurs
Octale	8	{0,1,2,3,4,5,6,7}	760	Utilisée il y a un certain temps en Informatique, elle permet de coder 3 bits en un seul symbole
Hexadécimale	16	{0,1,2,3,4,5,6,7,8,9, A,B,C,D,E,F}	1F0	Cette base est très utilisée dans le monde de la micro-informatique, elle permet de coder 4 bits par un seul symbole

Transcodage (ou conversion de base)

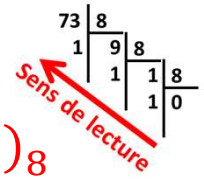
- Opération qui permet de passer de la représentation d'un nombre exprimé dans une base à la représentation du même nombre mais exprimé dans une autre base
- Changement de base de la base 10 vers une base b : Méthode des divisions successives
  - On divise le nombre par la base b
  - Puis le quotient par la base b
  - Et ainsi de suite jusqu'à l'obtention d'un quotient nul
  - La suite des restes correspond aux symboles de la base visée
  - On obtient en premier le chiffre de poids faible et en dernier le chiffre de poids fort

Ex : $(73)_{10} \longrightarrow (?)_2$



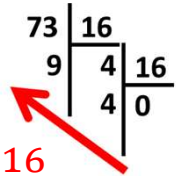
$(1001001)_2$

$(73)_{10} \longrightarrow (?)_8$



$(111)_8$

$(73)_{10} \longrightarrow (?)_{16}$



$(49)_{16}$

Transcodage (ou conversion de base)

- **Changement de base de la base 2 vers la base 10 :** $\sum_{i=0}^{n-1} a_i b^i$
- **Changement de la base 2 vers la base 8 :** On regroupe les bits en sous-ensembles de 3 bits puis on remplace chaque groupe par le symbole correspondant dans la base 8

Symbole octal	0	1	2	3	4	5	6	7
Symbole binaire	000	001	010	011	100	101	110	111

- **Changement de la base 2 vers la base 16 :** On regroupe les bits en sous-ensembles de 4 bits puis on remplace chaque groupe par le symbole correspondant dans la base 16

Symbole hexadécimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Symbole binaire	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

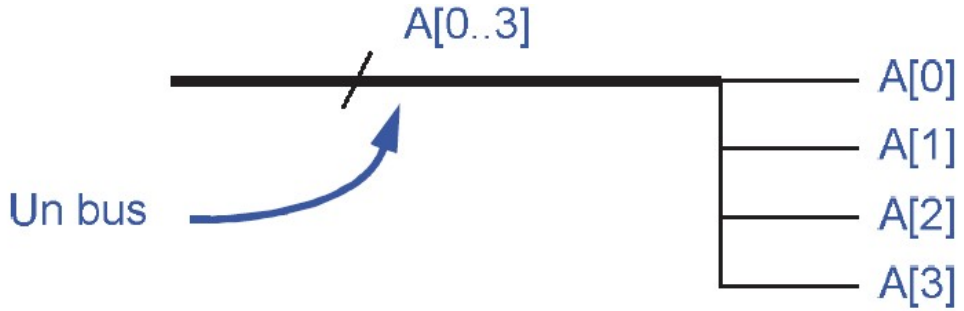
Sommaire

- Système de numération
- Système binaire
- Détection d'erreur
- Codage numérique

Encodage des nombres

- 1 bit = 2 valeurs, il faut donc en associer plusieurs pour représenter des grands nombres

- Symbolisation sous forme de bus :



4 bits : 2^4 valeurs

- Plusieurs formalisations pour un même nombre binaire :
  - Ex : ‘11100011’ en binaire naturel => 227
    ‘11100011’ en complément à 2 => -29
    ‘11100011’ en code ASCII => π

Encodage en binaire naturel

- Chaque bit, situé en position *n*, a un poids dans la représentation de 2^n. (Attention, *n* commence à 0 !)

Most Significant Bit

MSB	→	→	→	LSB	Least Significant Bit
2^4	2^3	2^2	2^1	2^0	
A[4]	A[3]	A[2]	A[1]	A[0]	
0	0	0	0	0	→ **0**
0	0	1	0	0	→ **$2^2 = 4$**
0	0	1	1	1	→ **$2^2 + 2^1 + 2^0 = 7$**
1	1	0	1	0	→ **$2^4 + 2^3 + 2^1 = 26$**

- Pour un mot de *k* bits, il y a donc 2^k valeurs, de 0 à 2^k-1

  → **Dynamique de codage**

Encodage en binaire naturel

- Représentation standard : sous forme d'octet (8 bits)
- Nombre représentés entre 0 et 255 (2^8-1) :
  - En décimal :
    par exemple '165'
  - En binaire :
    par exemple '1010 0101'
  - En hexadécimal : écriture par groupes de 4 bits (entre 0 et F)
    par exemple 'A5' → '1010 0101'
    A 5

Arithmétique en base 2

- Addition :

A	B	A + B	Retenue (Carry)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$\begin{array}{r} {}^{1}{}^{1}101100 \\ +\ 011010 \\ \hline = 1000110 \end{array}$$

- Soustraction :

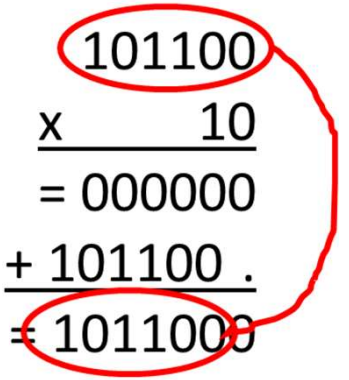
A	B	A - B	Retenue (Borrow)
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$$\begin{array}{r} {}^{1}101{}^{1}00 \\ -\ 011010 \\ \hline = 010010 \end{array}$$

Arithmétique en base 2

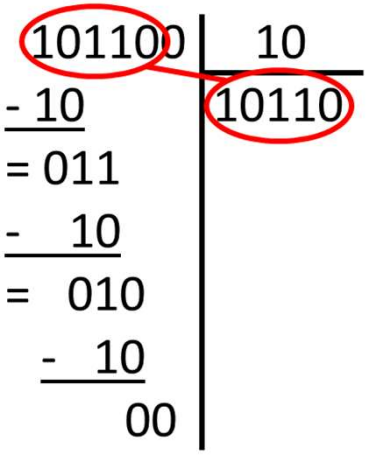
- Multiplication :

A	B	A x B
0	0	0
0	1	0
1	0	0
1	1	1



- Division :

A	B	A ÷ B
0	0	X
0	1	0
1	0	X
1	1	1



Code binaire signé

- Le bit le plus significatif est utilisé pour représenter le signe du nombre :
  - Si le bit le plus fort vaut 1, alors il s'agit d'un **nombre négatif**
  - Si le bit le plus fort vaut 0, alors il s'agit d'un **nombre positif**

- Les autres bits codent **la valeur absolue du nombre**

- Exemples : $(+2)_{10} = (010)_2$
  $(-2)_{10} = (110)_2$

Problème : Ce codage nécessite un traitement particulier pour le signe et introduit deux codes différents pour le chiffre 0

Encodage en complément à 1

- Ce code est également appelé complément logique ou restreint :
  - Les nombres positifs sont codés de la même façon qu'en binaire pur
  - Un nombre négatif est codé en inversant chaque bit de la représentation de sa valeur absolue

- Le bit le plus significatif est utilisé pour représenter le signe du nombre :
  - Si le bit le plus fort vaut 1, alors il s'agit d'un **nombre négatif**
  - Si le bit le plus fort vaut 0, alors il s'agit d'un **nombre positif**

Exemple : -24 en complément à 1 sur 8 bits

|-24| en binaire pur $\rightarrow (00011000)_2$

puis on inverse les bits $\rightarrow (11100111)_{Cà1}$

Problème : Ce codage introduit deux codes différents pour le chiffre 0

Encodage en complément à 2

- Permet de représenter les nombres négatifs

- 2 conditions à respecter pour préserver les règles de calcul dans un anneau commutatif :
  - Représentation du 0 : ‘0000 0000’
  - Respect de l’addition : **la représentation d’un nombre ‘x+1’ doit être égale à la représentation du nombre ‘x’ +1**

ex: ‘-1’ + 1 = ‘0’

‘1000 0001’ + 1 = ‘0000 0000’

‘1111 1111’ + 1 = ‘0000 0000’ 

Encodage en complément à 2

- Dynamique de codage pour *k* bits : $-2^{k-1} \rightarrow 2^{k-1}-1$

  ex : 8 bits = 256 valeurs = -128 → 127

MSB							LSB		
0	1	1	1	1	1	1	1	→	**127**
				…					
0	0	0	0	0	0	0	1	→	**1**
0	0	0	0	0	0	0	0	→	**0**
1	1	1	1	1	1	1	1	→	**-1**
1	1	1	1	1	1	1	0	→	**-2**
				…					
1	0	0	0	0	0	0	0	→	**-128**

Le 0 compte comme un nombre positif

- Propriété corollaire : $-A = \overline{A} + 1$

Codage des nombres réels

Codage en virgule fixe :

- Etant donné une base b, un nombre x est représenté, en format virgule fixe, par :
  - $x = \left(a_{n-1}a_{n-2} \dots a_1 a_0, a_{-1}a_{-2} \dots a_{-p}\right)_b$
  - a_{n-1} est le chiffre de poids fort (MSB)
  - a_{-p} est le chiffre de poids faible (LSB)
  - n est le nombre de chiffres avant la virgule
  - p est le nombre de chiffre après la virgule
- La valeur de x en base 10 est : $x = \sum_{i=-p}^{n-1} a_i b^i$

Exemple :

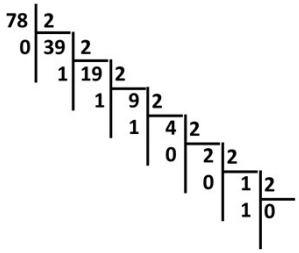
$$(101{,}01)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} = (5{,}25)_{10}$$

Codage des nombres réels

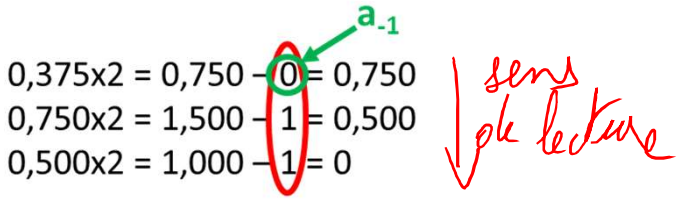
Codage en virgule fixe :

- Le passage de la base 10 à la base 2 est défini par :
  - Partie entière est codée sur p bits (divisions successives par 2)
  - Partie décimale est codée sur q bits en multipliant par 2 successivement jusqu'à ce que la partie décimale soit nulle ou le nombre de bits q est atteint
- Ce codage était utilisé par les premières machines

Exemple : $(78,375)_{10} \rightarrow (?)_2$



$(78)_{10} \rightarrow (1001110)_2$



$(78,375)_{10} \rightarrow (1001110,0110)_2$

Codage des nombres réels

Codage en virgule flottante :

- Soit un nombre N dans une base B. Sa notation décimale peut être écrite en utilisant la mantisse M et l'exposant E : $N = \pm M \times B^E$

- La mantisse est à virgule fixe et l'exposant E un nombre signé (complément à 2)

- D'une manière générale, le nombre peut être écrit :

Signe	Exposant	Mantisse

- Le signe est codé sur un bit ayant le poids le plus fort :
  - Le signe - : bit = 1
  - Le signe + : bit = 0

Sommaire

- Système de numération
- Système binaire
- Détection d'erreur
- Codage numérique

Opérations sur les nombres relatifs

- Un résultats faux est renvoyé s'il y a dépassement de mémoire
- Exemple : $A = (a_3a_2a_1a_0)_b \; et \; B = (b_3b_2b_1b_0)_b$

$$\begin{array}{r} a_3\,a_2\,a_1\,a_0 \\ +\; b_3\,b_2\,b_1\,b_0 \\ \hline = s_4\,s_3\,s_2\,s_1\,s_0 \end{array}$$

Il y a erreur si le nombre de digits est supérieur au nombre de digits des termes de l'opération

1010
+0101
1111
⇒ résultat 4 bits pas d'erreur

+1 1100
+0111
10011
⇒ ~~résultat~~ sur 5 bits
⇒ erreur.

Problèmes de conversions erronées

Exemple : $(0,1)_{10} \rightarrow (?)_2$

0,1x2 = 0,2 – 0 = 0,2
0,2x2 = 0,4 – 0 = 0,4
0,4x2 = 0,8 – 0 = 0,8
0,8x2 = 1,6 – 1 = 0,6
0,6x2 = 1,2 – 1 = 0,2
0,2x2 = 0,4 – 0 = 0,4
0,4x2 = 0,8 – 0 = 0,8

$$0 \times 2^{-1} + 0 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4} + 1 \times 2^{-5} \simeq 0{,}05$$

→ Introduction d'une erreur

Sommaire

- Système de numération
- Système binaire
- Détection d'erreur
- Codage numérique

Généralités

- Un codage est une opération qui établit une correspondance entre les éléments de deux ensembles
- Il permet d'établir une correspondance entre le langage humain et le langage machine
- Un codage est dit **pondéré** s'il associe à chaque variable un poids qui est une valeur numérique (ex : code binaire naturel)
- Un codage est dit **non pondéré** lorsque seule la combinaison des variables est significative (ex : code de Gray)

Code BCD (Binary Coded Decimal)

- Code décimal le plus utilisé en électronique
- Contient des mots-code qui sont la traduction en binaire naturel (sur 4 bits) de chacun des dix chiffres du système décimal

Exemple :

1	9	8	2	en décimal
↓	↓	↓	↓	
0001	1001	1000	0010	en BCD

0101	0000	0111	0011	en BCD
↓	↓	↓	↓	
5	0	7	3	en décimal

Décimal	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Code de Gray (ou binaire réfléchi)

- Un seul bit change de valeur entre deux codes successifs. Ce code est obtenu par réflexions successives.
- Il est utilisé dans les tableaux de Karnaugh, afin de coder les valeurs d'entrées de telle sorte qu'une seule d'entre elles ne change de valeur d'une ligne à l'autre, ou d'une colonne à l'autre

a	b	c	Décimal
0	0	0	0
0	0	1	1
0	1	1	2
0	1	0	3
1	1	0	4
1	1	1	5
1	0	1	6
1	0	0	7

$$(b_2 b_1 b_0)_2 \rightarrow (abc)_{Gray}$$

$$a = b_2$$
$$b = b_2 \oplus b_1$$
$$c = b_1 \oplus b_0$$

Codage avec bit de parité

- Ce codage permet de vérifier la parité des nombres après transmission
- Le bit de parité P vaut 0 si le nombre de bits à 1 est pair

a	b	c	P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Codage P parmi N

Le code P parmi N est un code à N bits dont P bits sont à 1 et (N-P) bits sont à 0. La lecture de ce code peut être associée à la vérification du nombre des 1 et des 0 dans l'information, ce qui permet de contrôler l'information lue par la détection du code erroné.

Exemple : Code 2 parmi 5

Nombre de combinaisons possibles :

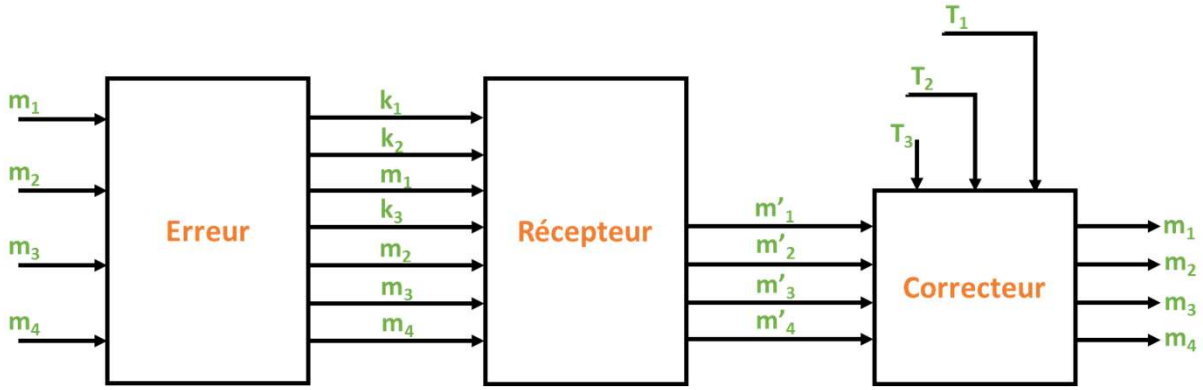
$$C = \frac{N!}{P!\,(N-P)!}$$

Décimal	7	4	2	1	0
0	1	1	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	0
7	1	0	0	0	1
8	1	0	0	1	0
9	1	0	1	0	0

ISEN école d'ingénieurs

Codage de Hamming

- Ce codage a été mis au point par R. Hamming à la fin des années 1940 afin de corriger les erreurs de transmission
- Il consiste à introduire des bits de contrôle en plus des bits transportant l'information
- Les bits de contrôle servent à effectuer des tests de parité avec les bits transportant l'information



Tests de parité :

$$\begin{cases} T_1 = k_1 m_1 m_2 m_4 \\ T_2 = k_2 m_1 m_3 m_4 \\ T_3 = k_3 m_2 m_3 m_4 \end{cases}$$

Récapitulatif (A savoir)

- Systèmes binaire, octale et hexadécimal
- Encodage binaire des entiers naturels et relatifs
- Arithmétique en base 2
- Détection d'erreur
- Codes BCD et binaire réfléchi
- Notion de bit de parité

Fin du chapitre 2