Codage de l'information

Cours

1. Les systèmes de numération

Depuis l'antiquité, les hommes ont essayé de représenter les nombres entiers par des symboles. Il existe essentiellement deux grands principes :

1. chaque symbole possède une valeur, et on ajoute ces valeurs pour obtenir le nombre. Le système de numération des romains repose essentiellement sur ce principe (XXXII=32), avec en plus un aspect soustractif (IX=9).
2. on utilise un nombre petit de symboles (les chiffres) dont la valeur dépend de la position. Chaque décalage vers la gauche du symbole le multiplie par une certaine quantité appelée la base. Par exemple, en écriture décimale 2345 signifie 5+4×10+3×100+2× 1000.

Le second système possédant de nombreux avantages par rapport au premier, notamment pour effectuer les opérations, c'est lui qui s'est peu à peu imposé. Nous utilisons actuellement le système décimal (la base est 10) mais ce ne fut pas toujours le cas. Les babyloniens utilisaient un système sexagésimal, ce qui signifie que la base était 60. C'est pourquoi une minute fait 60 secondes!
En informatique, les base 2 (écriture binaire) et base 16 (écriture hexadécimale, où les chiffres représentant 11,12,...15, sont notés A,B,...,F) sont très utilisées.

2. Les bases fréquemment utilisées en sciences de l'ingénieur

Les bases de numération utilisées sont :

- **Le décimal (Base 10) : {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}**

Le système décimal est le système universellement utilisé. C'est la base de référence, ce qui signifie qu'un nombre est de manière implicite décimal dés lors qu'il est écrit sans précision de sa base.

- **Le binaire (Base 2) : {0, 1}**

C'est la base de numération couramment utilisé en électronique. C'est un système à base **2** qui est donc composé des caractères **0 et 1**. Chacun de ces chiffres est appelé 'Bit', contraction des mots Binary Unit ou Binary Digit.

- **L'hexadécimal (Base 16). Cette base utilise des nombres et des lettres :{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F}**

Ce système à base **16** est le plus utilisé en électronique numérique car il permet une manipulation de quartets en représentation compacte. Ce qui, dans les systèmes actuels à grande capacité mémoire par exemple, est un avantage non négligeable. La base 16 est une forme contractée de la base 2.

2.1 Notation

Lorsqu'on écrit un nombre, il faut spécifier dans quelle base il est écrit, sinon il y a des risques de confusion. Par convention, on indique la base en indice :

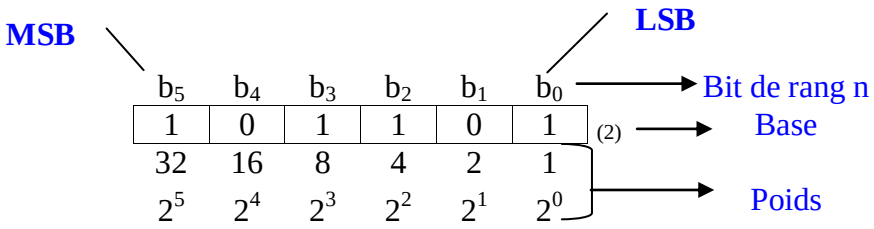
$3A9_{(16)} = (3A9)_{16}$ →base 16 : hexadécimal
$238_{(10)} = (238)_{10}$ →base 10 : décimal
$0100\ 1101_{(2)} = (0100\ 1101)_2$ →base 2 : binaire

En programmation, la notation permettant de voir que l'on est en hexadécimal dépend du langage :
Exemple avec $(AE4F)_{16}$:

Langage	Préfix	Exemple
C, C++, java	0x	0xAE4F
Pascal	$	$AE4F
Basic	&h	&hAE4F
HTML	#	#AE4F

2.2 Vocabulaire

On définit alors par « bit », la plus petite unité d'information manipulable par un circuit logique qui est 0 ou 1. Il est possible de traiter des mots de plusieurs bits. Un mot de 4 bits est appelé un quartet, de 8 bits un octet (Byte) et de 16 bits un Word.



On appelle le bit de rang 0 (b_0) **le bit de poids faible (LSB)** et le bit de rang le plus élevé (ici b_5) **le bit de poids fort (MSB)**.

La formule du poids P est : **$P = Base^{Rang}$**
La formule du nombre de combinaisons possibles est : **$N = Base^{Nbre\ de\ Bit}$**

3. Conversion d'un nombre exprimé dans une base B en base 10

$$N_{(10)} = \sum_{i=0}^{i=n-1} a_i.B^i$$

B : La base du nombre à convertir
n : Le nombre de chiffres ou de bits du nombre exprimé en base B
a_i : Le chiffre ou bit de rang i du nombre exprimé en base B
N : Le nombre exprimé en base B

$$N_{(10)} = a_0.B^0 + a_1.B^1 + a_2.B^2 + \ldots + a_{(n-2)}.B^{(n-2)} + a_{(n-1)}.B^{(n-1)}$$

Exemple
Soit $N = 483_{(16)}$ à convertir en base 10.
D'après les notations précédentes :

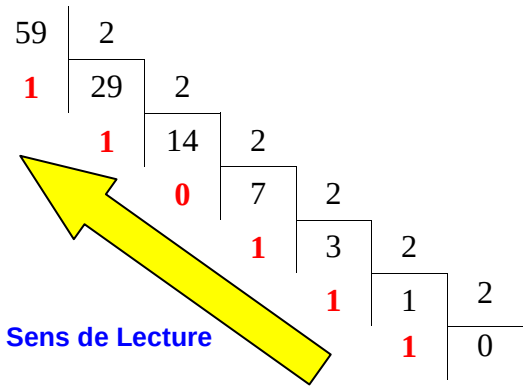
B	n	a_0	a_1	a_2
16	3	3	8	4

$(N)_{10} = 3 \times 16^0 + 8 \times 16^1 + 4 \times 16^2 = 3 + 128 + 1024 = 1155$
$(483)_{16} = (1155)_{10}$

4. Passage du décimal vers une base n

Pour convertir un nombre décimal en une base n, on divise ce nombre par n, puis tant que le quotient obtenu est supérieur ou égal à n, on divise à nouveau le résultat par n. Le nombre cherché est donné par les restes des divisions successives et le résultat de la dernière division. Le poids fort du nombre cherché est le résultat de la dernière division et le poids faible le reste de la première division.

Exemple : Soit N = $59_{(10)}$ à coder en binaire (base 2)



N = $(59)_{10}$ = $(11\ 1011)_2$
On peut écrire le résultat sur un octet : N = $(59)_{10}$ = $(0011\ 1011)_2$

5. Passage du binaire à l'hexadécimal

Il faut séparer le nombre exprimé en binaire en paquets de 4 bits **en commençant par les bits de poids faibles**. Ensuite, on calcul le poids total de chaque paquet de 4 bits pour obtenir le nombre en hexadécimal.

Exemple : soit N = $(101\ 0101\ 0110\ 1010)_2$ à exprimer en hexadécimal

$(0101)_2 = (4 + 1)_{10} = (5)_{10} = (5)_{16}$ .....

$(0101)_2 = (4 + 1)_{10} = (5)_{10} = (5)_{16}$ .....

$(0110)_2 = (4 + 2)_{10} = (6)_{10} = (6)_{16}$ .....

$(1010)_2 = (8 + 2)_{10} = (10)_{10} = (A)_{16}$ .....

$(0101\ 0101\ 0110\ 1010)_2 = (556A)_{16}$ .....

6. Passage de l'hexadécimal au binaire

Il faut convertir chaque "chiffre" du nombre hexadécimal en un paquet de 4 bits.

Exemple : Soit N = $1F4A_{(16)}$ à exprimer en binaire

$(1)_{16} = (1)_{10} = (0001)_2$ .....

$(F)_{16} = (15)_{10} = (1111)_2$ .....

$(4)_{16} = (4)_{10} = (0100)_2$ .....

$(A)_{16} = (10)_{10} = (1010)_2$ .....

$(1F4A)_{16} = (0001\ 1111\ 0100\ 1010)_2 = (1\ 1111\ 0100\ 1010)_2$ .....

7. Les Codes

7.1 Binaire pur

Toute donnée entrant, par exemple, dans un ordinateur ou un automate est transmis en code binaire pur.
Le code binaire pur utilise la numération binaire. La construction se fait par copie du code binaire pur en partant du poids faible (LSB) en rajoutant les bits de poids supérieur pour atteindre le bit de poids fort.

7.2 Binaire réfléchi ou code gray

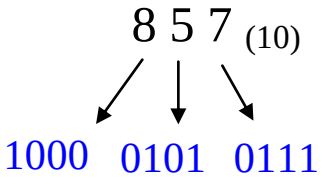
Dans ce codage, un seul bit change d'état entre deux valeurs voisines. Il ne peut pas exister de résultat temporaire aberrant entre deux valeurs voisines. Du fait de cette propriété, le code Gray (ou code réfléchi) est très utilisé, notamment dans les codeurs absolus de position.

	Binaire pur	Binaire réfléchi
0	0000	**0000**
1	0001	**0001**
2	0010	**0011**
3	0011	**0010**
4	0100	**0110**
5	0101	**0111**
6	0110	**0101**
7	0111	**0100**

7.3 Code BCD (Binary Code Décimal) – Décimal codé en binaire

Ce code conserve les avantages du système binaire naturel et du système décimal. Chaque chiffre du code décimal est représenté par un quartet binaire, mais on compte en base 10, ce qui veut dire que la valeur la plus élevée dans un quartet est $9_{(10)} = 1001_{(2)}$.

Le chiffre 857 sera donc représenté par :



Donc $857_{(10)} = 1000\ 0101\ 0111_{(BCD)}$

Attention : $10_{(10)} = 1010_{(2)} = 0001\ 0000_{(BCD)}$

7.3 Code ASCII

Le code ASCII (American Standard Code for Information Interchange) est principalement utilisé pour la transmission en série des caractères alphanumériques (Chiffres, lettres, caractères spéciaux) dans les systèmes informatiques.