Les bases de numération

Cours systèmes de numérations :

Vous remarquerez qu'en informatique tout fonctionne avec des puissances de 2.

Par exemple, on peut mettre quatre (2^2) unités de disque sur un bus IDE, on peut brancher jusqu'à 128 (2^7) périphériques sur un bus USB, on peut utiliser des barrettes de 256 Mo (2^8 Mo) ou de 512 Mo (2^9 Mo), etc. Les exemples ne manquent pas. Et c'est normal, puisque l'ordinateur travaille en binaire (base 2). Mais comme nous, les êtres humains, nous travaillons en décimal (base 10), on est souvent amené à jongler entre les différentes bases de numération.

Objectif

À la fin de cette séquence, vous maîtriserez les bases de numération utiles en informatique (binaire octal et hexadécimal). Vous saurez faire des calculs élémentaires dans chaque base et vous saurez faire des conversions entre le binaire, l'octal, l'hexadécimal et le décimal.

Contenu

1. Principes de fonctionnement d'une base de numération

Dans cette partie, nous allons traiter des bases de numération utilisées en informatique (binaire et hexadécimal). Elles sont peu utilisées par l'être humain, qui préfère la base décimale (question de morphologie !). Toutefois, il est impératif de les connaître (il faut savoir parler le même langage que l'ordinateur).

Une base de numération est une sorte de langage mathématique. Elle définit un alphabet (chiffres) et une syntaxe de constitution des mots (nombres). Nous allons partir de la base décimale pour en déduire les principes d'une base de numération quelconque.

La **numération de position** (Mathématiques) est une sorte de procédé d'écriture des nombres, dans lequel chaque position d'un chiffre est reliée à la position voisine par un multiplicateur, qui est nommé la base du système de numération.

On parle de numération décimale de position car elle est en base 10.

Elle diffère, par exemple, de la numération romaine qui n'est pas un système de numération positionnel, mais un système additif dans lequel chaque symbole a une valeur fixe.

1A. La base décimale (base 10)

En base 10, l'alphabet est composé de 10 symboles {0, 1, …, 9}.

Dans un nombre, un **poids** est associé à chaque chiffre. Ce poids est le coefficient par lequel il faudra multiplier le chiffre pour obtenir sa valeur réelle. Ce coefficient est constitué de la base à la puissance du rang du chiffre dans le nombre.

Exemple : on peut décomposer le nombre 425 exprimé en base décimale de la façon suivante :

Rang	2	1	0	Nombre
Poids =	10^2 100	10^1 10	10^0 1	
Chiffres	4	2	5	
Valeur	4 x 100 +	2 x 10 +	5 x 1	= 425

Pour compter, on utilise le même principe :

Rang	2	1	0	Nombre
Poids =	10^2 100	10^1 10	10^0 1	
Chiffres	0	0	0	0
	0	0	1	1
	0	0	2	2
				...
	0	0	9	9
	0	1	0	10
	0	1	1	11
				...
	0	1	9	19
	0	2	0	20
				...
	0	9	9	99
	1	0	0	100

Tableau 2

Jusqu'ici, c'est très simple. Mais, en général, dès que l'on passe à la base 2, ça se complique un petit peu, alors que **toutes les bases de numération fonctionnent sur un principe identique.**

La seule différence réside dans l'alphabet qui est plus ou moins développé suivant la base. On peut donc généraliser ce que nous venons de dire à une base b quelconque.

1B. Une base b quelconque

Par la suite, nous utiliserons la notation (N)b pour signifier un nombre N exprimé dans la base b.

Dans une base b, l'alphabet est composé de b **chiffres** : le plus petit chiffre de la base est égal à 0, le plus grand chiffre de la base est égal à b-1. De façon plus formelle : pour un symbole a quelconque de l'alphabet, a $\in$ {0, 1, …, b-1}.

Les mots sont des nombres. On peut les décomposer sous la forme suivante :

(N)b = aibi + ai-1bi-1 + … + a2b2 + a1b1 + a0b0
où a est un symbole de l'alphabet, b la base et i le rang.

2. Le système binaire (base 2)

En binaire, l'alphabet est le plus simple qui puisse s'imaginer : {0,1}

2A. Compter

Reprenons le tableau vu avec la base 10 et adaptons-le à la base 2 :

Rang	3	2	1	0	Nombre
Poids =	2^3 $(8)_{10}$	2^2 $(4)_{10}$	2^1 $(2)_{10}$	2^0 $(1)_{10}$	
Chiffres	0	0	0	0	0
	0	0	0	1	1
	0	0	1	0	10
	0	0	1	1	11
	0	1	0	0	100

2B. Vocabulaire

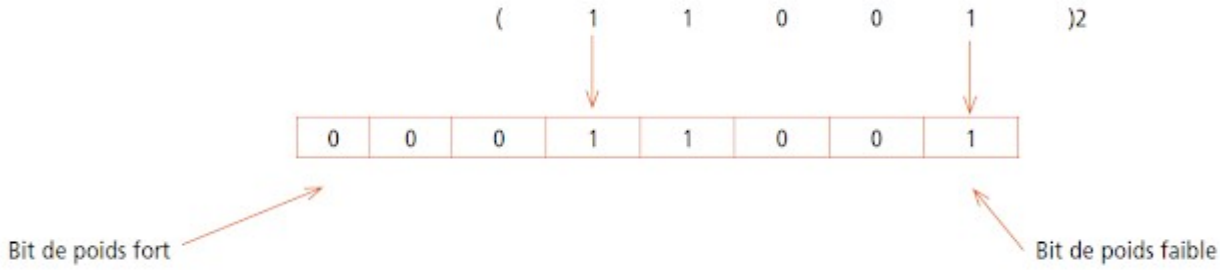
Chaque 0 ou 1 d'un nombre binaire s'appelle un **bit** (acronyme de *Binary digIT* en anglais).

Pour faciliter le traitement, l'ordinateur travaille souvent sur des groupes de bits dont la taille est :

8 bits = 1 octet
16 bits = 2 octets = 1 mot
32 bits = 4 octets = 1 double mot

Veuillez bien noter ceci, car c'est une source d'erreurs :
Ne pas confondre byte en anglais qui signifie **octet** en français (et non bit !)

Représentons sur un octet le nombre :



Observation

On a comblé à gauche par des 0. Ceux-ci sont non significatifs, mais l'ordinateur comme l'être humain, a horreur du vide. Ainsi, un bit qui n'est pas explicitement fixé à 1 est fixé à 0.

Il est indispensable que vous connaissiez de mémoire :

• les premières puissances de 2 (de 2^0 à 2^7) soit 1, 2, 4, 8, 16, 32, 64, 128 ;
• 2^8 et 2^{16} soit 256 et 65 536 ;
• 2^{24} et 2^{32} (approximativement) soit 16,7 millions et 4 milliards.

2C. Arithmétique élémentaire

Toute base de numération permet de faire des calculs. Nous ne voyons ici que l'addition et la soustraction mais ce n'est pas limitatif, tout calcul est possible.

2C1. L'addition

L'addition de deux bits se déroule de la façon suivante :

0 + 0 = 0
0 + 1 = 1
1 + 1 = 0 et une retenue de 1.

Dans certains cas, lorsque la retenue se propage, on peut être amené à calculer :

1 + 1 + 1 = 1 **et une retenue de 1**.

On souhaite effectuer l'opération suivante : (1101)2 + (110)2

Retenues		1			
		1	1	0	1
	+		1	1	0
	1	0	0	1	1

2C2. La soustraction

La soustraction de deux bits se déroule de la façon suivante :

0 – 0 = 0
0 – 1 = 1 et une retenue de -1.
1 – 0 = 1
1 – 1 = 0

On souhaite effectuer l'opération suivante : $(1101)^2$ - $(110)^2$

Retenues		-1	-1		
		1	1	0	1
	-		1	1	0
		0	1	1	1

Nous en avons fini avec la base 2. Passons maintenant à la base 16.

3. L'hexadécimal (base 16)

La particularité de la base 16, comparé à la base 10 et à la base 2, est d'avoir un alphabet plus étendu (16 symboles) :

Base 16	Base 10	Base 2
0	0	0
1	1	01
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Tableau 4 : correspondances base 16, base 10, base 2

Oui, la curiosité de cette base est d'introduire des lettres pour les chiffres dépassant le 9.

C'est assez troublant, surtout lorsque l'on fait des calculs. Mais, je vous assure que c'est une pure histoire de convention.

L'hexadécimal est très souvent utilisé en informatique pour visualiser une série d'informations numériques (vidages mémoires). En effet, elle est plus condensée que le binaire.

Pour visualiser un octet, il suffit toujours de deux symboles hexadécimaux (contre huit symboles en binaire).

Par exemple :

$$(0101\ \ 1010)_2 = (5\ \ A)_{16}$$

Valeur remarquable à connaître : $(FF)_{16} = (1111\ 1111)_2 = (255)_{10}$

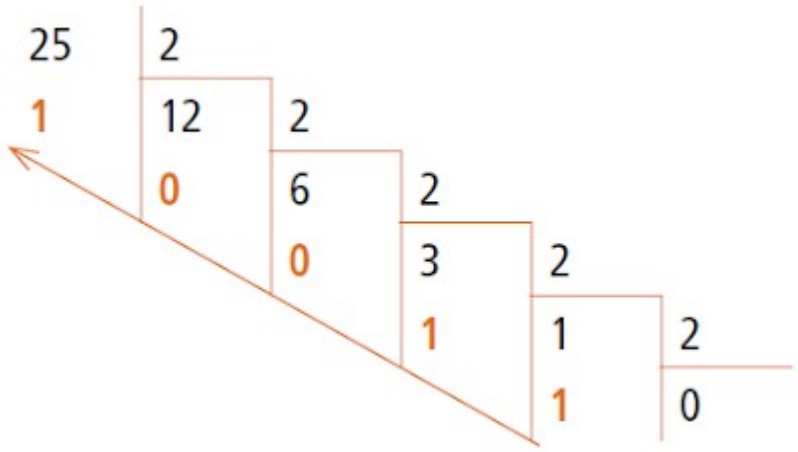
4. Les conversions de base

L'être humain et la machine ne raisonnent pas dans la même base de numération, nous serons donc souvent amenés à faire des conversions.

4A. Conversion en binaire

4A1. Du décimal au binaire

Prenons un exemple. Pour convertir $(25)_{10}$ en binaire, on fait des divisions entières successives par la base :



Lorsque l'on ne peut plus diviser, on s'arrête. Le résultat est constitué des restes des divisions lus de droite à gauche. Ici, on obtient : $(25)_{10} = (1\ 1001)_2$

4A2. Du binaire au décimal

Pour convertir $(10\ 1001)_2$ en décimal, on décompose le nombre en puissances de deux :

Rang	5	4	3	2	1	0	Nombre à convertir
Nombre	1	0	1	0	0	1	
Poids	2^5 $(32)_{10}$	2^4 $(16)_{10}$	2^3 $(8)_{10}$	2^2 $(4)_{10}$	2^1 $(2)_{10}$	2^0 $(1)_{10}$	Puissances de 2
Valeur	1 x 32 +	0 x 16 +	1 x 8 +	0 x 4 +	0 x 2 +	1 x 1	$= (41)_{10}$

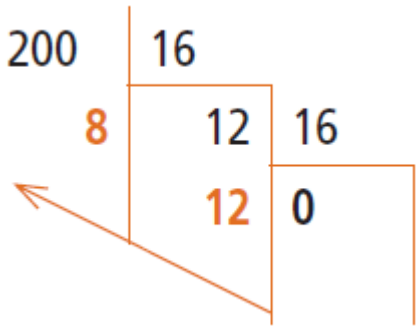
Tableau 5

Donc, $(10\ 1001)_2 = (41)_{10}$

4B. Conversion en hexadécimal

4B1. Du décimal à l'hexadécimal

Pour convertir $(200)_{10}$ en hexadécimal, on fait des divisions entières successives par la base :



Comme pour le binaire, on lit le résultat de droite à gauche, soit $(12\ 8)_{16}$. Mais **attention**, le « chiffre » 12 n'existe pas en hexadécimal, on doit indiquer le chiffre « C ». Le résultat est donc $(C8)_{16}$.

Donc, $(200)_{10} = (C8)_{16}$.

4B2. De l'hexadécimal au décimal

Pour convertir $(1B\ 2C)_{16}$ en décimal, on décompose le nombre en puissances de seize :

Rang	4	3	2	1	0	Nombre à convertir
Nombre	0	1	B	2	C	
Poids	16^4 $(65\ 536)_{10}$	16^3 $(4\ 096)_{10}$	16^2 $(256)_{10}$	16^1 $(16)_{10}$	16^0 $(1)_{10}$	Puissances de 16
Valeur	0 x 65 536 +	1 x 4 096 +	11 x 256 +	2 x 16 +	12 x 1	$= (6\ 956)_{10}$

Donc, $(1B\ 2C)_{16} = (6\ 956)_{10}$

4B3. De l'hexadécimal au binaire et réciproquement

Chaque symbole hexadécimal correspond à des paquets de 4 bits. Par exemple, pour le nombre $(ABCD)_{16}$:

(A	B	C	$D)_{16}$
(10	11	12	$13)_{10}$
(1010	1011	1100	$1101)_{2}$

Donc, $(ABCD)_{16} = (1010\ 1011\ 1100\ 1101)_2$

À retenir

Vous devez être capable de convertir des nombres exprimés dans l'une des trois bases de numération que nous venons d'étudier vers n'importe laquelle de ces bases.