Le système binaire

Comment comptons nous en décimal ?

Depuis la fin du moyen-age, nous comptons en **base 10**. Certains prétendent que cette pratique est venue du fait que nous a 10 doigts. Il en découle principalement deux choses :

- Il existe **10 chiffres** : 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9.
- Avec ces chiffres ont peut compter de 0 à 9.

Le nombre 563 peut s'écrire : $563 = 500 + 60 + 3$

que l'on peut aussi écrire : $563 = 5 \times 100 + 6 \times 10 + 3 \times 1$

ou encore : $563 = 5 \times 10^2 + 6 \times 10^1 + 3 \times 10^0$

Un peu d'histoire

Jusqu'à la fin du moyen-age, on utilisait en Europe un système visécimal (ou vigésimal), utilisant la **base 20**. A cette époque, les chiffres arabes n'étaient pas encore utilisés.

On trouve des traces de ce système dans la langue française (quatre-vingt). L'hôpital des quinze vingts, fondé en 1260 à Paris pouvait accueillir 300 patients.

Le binaire

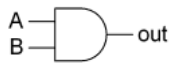
Présentation

Le binaire est le mode de comptage non plus en base 10 mais **en base 2**. Le binaire est un concept essentiel de l'informatique. En effet, les **processeurs** des systèmes numériques sont constitués de composants électroniques soumis ou non à une tension. Le binaire permet de modéliser le fonctionnement de ces composants. Le chiffre 0 sera utilisé pour signifier une absence de tension, et le chiffre 1 pour signifier la présence d'une tension.

L'assemblage de ces composants permet de réaliser les **opérations logiques de base** qui permettront à leur tour de concevoir des systèmes complexes.

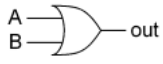
Opérations logiques de base

Porte ET (AND)



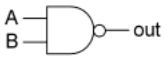
Entrées		Sortie
A	B	A ET B
0	0	0
0	1	0
1	0	0
1	1	1

Porte OU (OR)



Entrée		Sorties
A	B	A OU B
0	0	0
0	1	1
1	0	1
1	1	1

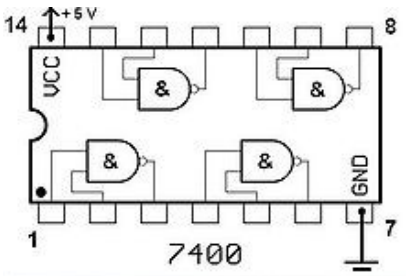
porte NON ET (NAND)

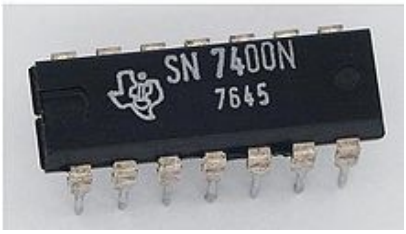


Entrée		Sorties
A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0

A NAND B = NON (A ET B)

Exemple de circuit logique





Circuit intégré de la série 7400 à quatre portes logiques NAND.

L'indication « 7645 » indique que le circuit a été fabriqué la 45^{e} semaine de 1976.

Source : wikipedia - Liste des circuits intégrés de la série 7400

Les nombres en binaire jusqu'à dix-neuf

valeur en décimal	équivalent en binaire
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001

valeur en décimal	équivalent en binaire
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
16	10000
17	10001
18	10010
19	10011

On peut écrire : $(19)_{10} = (10011)_2$

On nomme **bit** (de l'anglais binary digit, soit « chiffre binaire ») les chiffres de la numération binaire.

Un calcul informatique n'est donc qu'une suite d'opérations sur des paquets de 0 et de 1, appelés **octets** lorsqu'ils sont regroupés par huit.

Additionner et soustraire en binaire

Ça se fait comme en décimal. (attention aux retenues).

1101 + 1001 = 10110
10110 – 1001 = 1101

Multiplier deux nombres en binaire

Là encore, il faut poser la multiplication comme en décimal.

Exemple : calculer $(1010)_2 \times (10)_2$:

$$\begin{array}{r} 1010 \\ \times \quad 10 \\ \hline 0000 \\ +\ 10100 \\ \hline 10100 \end{array}$$

Diviser deux nombres en binaire

Il faut poser la division comme en décimal : 10010000111 : 1011 = 1101001 (reste 100)

Sachant que 1 : 10 = 0,1 on en déduit que $(0,5)_{10} = (0,1)_2$

Algorithmes de conversion décimal ↔ binaire

Première méthode : les puissances de deux

Pour convertir un nombre décimal en binaire, on doit décomposer notre nombre décimal en une somme de puissances de 2.

...	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
...	4096	2048	1024	512	256	128	64	32	16	8	4	2	1

Ainsi, si l'on prend l'exemple du nombre 26, on obtient la décomposition suivante :

on a $26 = 16 + 10$ avec $10 = 8 + 2$
d'où $26 = 16 + 8 + 2$ qui peut s'écrire $26 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$
on en déduit $(26)_{10} = (11010)_2$

Deuxième méthode : les divisions euclidiennes par deux

- On divise le nombre décimal par 2 et on note le reste de la division (soit un 1 soit 0).
- On refait la même chose avec le quotient précédent, et on note de nouveau le reste.
- On recommence la division, et ce jusqu'à ce que le quotient soit 0.

Exemple : écrire $(26)_{10}$ en binaire

$\frac{26}{2} = 13$ (reste = **0**) | $\frac{13}{2} = 6$ (reste = **1**) | $\frac{6}{2} = 3$ (reste = **0**) | $\frac{3}{2} = 1$ (reste = **1**)

Il faut lire le résultat de droite à gauche : 1010, précédé par 1 : $(26)_{10} = (11010)_2$

Représentation des nombres entiers négatifs

Dans un système numérique le chiffre « 0 » est utilisé pour signifier une absence de tension, et le chiffre « 1 » pour signifier la présence d'une tension. Dans la modélisation binaire d'un système numérique il n'y a pas de place pour le signe « − » utilisé dans le système décimal.

La façon informatique de représenter un nombre négatif est de prévoir un bit de signe, placé au début du nombre :

0	0	0	1	0	0	1	1
bit de signe	**7 bits de représentation d'une valeur comprise entre 2^0 et $2^7 - 1$**						

Un bit de signe nul indique une valeur positive
Un bit de signe positionné à 1 indique une valeur négative.

Première méthode : le complément à un

On se contente de changer la valeur de chaque bit de valeur

On a : 19 = 00010011 et -19 = 11101100

Dans cette méthode la valeur zéro a deux représentations : 00000000 et 11111111 ce qui complique le test de valeur nulle d'un résultat

Deuxième méthode : le complément à deux

On se contente d'additionner « 1 » au résultat du complément à un.

On a : 19 = 00010011 et -19 = 11101100 + 1 = 11101101

```
   00010011
+  11101101
------------
   00000000
```

la valeur zéro a une seule représentation :

Codage Hexadécimal

Le système hexadécimal est un système de numération positionnel en base 16. Les 16 symboles sont les chiffres arabes pour les dix premiers chiffres et les lettres A à F pour les six suivants. Pour les nombres compris entre 0 et 255, le codage hexadécimal permet une représentation aisée des octets (un octet est codé très simplement sur deux chiffres en hexadécimal).

Les éditeurs hexadécimaux donnent une représentation hexadécimale des données binaires des fichiers.

Binaire	Hexadécimal	Décimal
0000 0000	00	000
0000 0001	01	001
0000 0010	02	002
0000 1010	0A	010
0000 1011	0B	11

Binaire	Hexadécimal	Décimal
0000 1111	0F	015
0001 0000	10	016
0001 0001	11	017
0001 0010	12	018
1111 1111	FF	255