

La numération

Le système décimal

C'est le système de numération que nous utilisons couramment. Il utilise les neuf chiffres 0,1,2,3,4,5,6,7,8,9.

Un nombre quelconque peut ainsi être décomposé en puissances de 10.

Par exemple :

$$525 = 5 \times 10^2 + 2 \times 10^1 + 5 \times 10^0$$

Ce principe de décomposition se retrouvera pour les différents types de systèmes de numération (binaire, hexadécimal pour nous)

Le système binaire

C'est le système de numération à base 2. On peut ainsi représenter les nombres avec les 2 symboles 0 et 1

Le bit de poids fort est celui qui est le plus à gauche (en anglais, c'est le **LSB** : « **Least Significant Bit** »)

Le bit de poids faible est celui qui est le plus à droite (en anglais, c'est le **MSB** : « **Most Significant Bit** »)

Exemples:

❑ $(00010011)_2$, le **bit de poids faible** est le 1 écrit le plus à droite

❑ $(1001)_2$, le **bit de poids fort** est le 1 écrit le plus à gauche

Le système hexadécimal

Il s'établit, comme le dit son nom, en base 16. Comme dans le système décimal nous ne possédons que 10 chiffres, nous complétons jusqu'à 16 avec les premières lettres de l'alphabet.

En résumé nous aurons:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F


Nous pourrions utiliser ce tableau de correspondances pour les futures conversions à effectuer entre les différents systèmes de numération.

C'est par l'utilisation de ce système que nous désignons un emplacement dans la mémoire d'un ordinateur.

Conversion du système Décimal vers le système Binaire

On réalise des divisions successives du nombre en base 10 par 2. Les restes constituent les caractères binaires de la conversion

Exemple: conversion de 14 en base 2

$$\begin{array}{r|l} 14 & 2 \\ \hline 07 & 2 \\ \hline 1 & 3 \\ \hline & 1 \\ \hline & 1 \end{array}$$


D'où: 14 en base 2: 1110

Nous pouvons aussi utiliser une méthode comme ci-dessous:

$$26 = 16 + 8 + 2 \quad 26 = 1 \times 16 + 1 \times 8 + 1 \times 2 \quad 26 = 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^1$$

$$26 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + \dots$$

$$26 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 \text{ (rajouter les puissances de 2 si nécessaire)}$$

$$26 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

Conversion Binaire vers Décimal

Pour le convertir en décimal, on procède de la manière suivante : on multiplie par 2^0 la valeur du rang 0, par 2^1 la valeur du rang 1, par 2^2 la valeur du rang 2, [...], par 2^{10} la valeur du rang 10, etc.

Exemple:

Pour convertir 101 0110 en décimal, on a donc $0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 0 \times 2^3 + 1 \times 2^4 + 0 \times 2^5 + 1 \times 2^6$.

Ensuite, il suffit simplement de remplacer les puissances de 2 par leurs valeurs et de faire la somme : $0 \times 1 + 1 \times 2 + 1 \times 4 + 0 \times 8 + 1 \times 16 + 0 \times 32 + 1 \times 64 = 86$.

Conversion binaire vers l'hexadécimal

Il suffit de regrouper les bits par quatre (en commençant depuis la droite):

Exemple:

Convertissons 01011111 en binaire.

0101	1111
5	15
5	F

Donc, 01011111 binaire = 5F hexadécimal

Conversion de l'hexadécimal vers le binaire

C'est le processus inverse du précédent:

Exemple:

Convertissons 5F en binaire.

5	F
5	15
0101	1111

Donc, 5F hexadécimal = 01011111 binaire

Conversion du décimal en hexadécimal

Nous aurons besoin des puissances de 16:

$$16^0 = 1$$

$$16^1 = 16$$

$$16^2 = 256$$

$$16^3 = 4096$$

$$16^4 = 65536$$

...

Exemples:

$$1680 = 6 \times 256 + 9 \times 16 + 0 \times 1 = 6 \times 16^2 + 9 \times 16^1 + 0 \times 16^0 \text{ soit } 690 \text{ (lire six-neuf-zéro)}$$

$$2009 = 7 \times 16^2 + 13 \times 16^1 + 9 \times 16^0 \text{ soit } 7D9$$

On peut aussi utiliser une division euclidienne par 16 en retenant les restes:

$$5283 / 16 = 330 \text{ il reste } \mathbf{3}$$

$$330 / 16 = 20 \text{ il reste } \mathbf{10}$$

$$20 / 16 = \mathbf{1} \text{ il reste } \mathbf{4}$$

D'où: 3A4

Conversion de l'hexadécimal en décimal

Similairement aux exemples précédents, les conversions dans ce sens sont très simples. Ici, il suffit de voir que chaque rang dans le nombre hexadécimal est une puissance de 16. Les puissance de 16 s'incrémentent de la gauche vers le droite:

Exemples:

$$7D9 = 7 \times 16^2 + 13 \times 16^1 + 9 \times 16^0 = 2009 \text{ en décimal}$$

$$4F2C = 4 \times 16^3 + F \times 16^2 + 2 \times 16^1 + C \times 16^0$$

$$4F2C = 4 \times 16^3 + 15 \times 16^2 + 2 \times 16^1 + 12 \times 16^0$$

$$4F2C = 4 \times 4096 + 15 \times 256 + 2 \times 16 + 12 \times 1$$

4F2C en hexadécimal est alors égal à 20 268 dans le système décimal

Complément à 2 et codage des nombres

Nous nous limitons au codage des nombres positifs et des nombres négatifs pour le principe

Sur n bits, nous pouvons coder les nombre de 0 à $2^n - 1$
par exemple sur 3 bits, nous aurons:

0 : 000

1 : 001

2: 010

3: 011

4 : 00

5: 101

6: 110

7: 111

Obtention d'un entier négatif

Soit un entier positif, x . On obtient le nombre opposé de la façon suivante:

- on remplace les 0 par les 1 et les 1 par les 0
- on ajoute 1 au résultat

Exemple:

Soit le nombre 4 codé sur 8 bits: 00000100

On inverse les 0 et les 1: 11111011

On ajoute 1 en binaire:

$$\begin{array}{r} 11111011 \\ + \\ 00000100 \\ \hline 11111111 \end{array}$$

$(11111111)_2$ en binaire correspond à -1 en décimal.

En utilisant cette technique, on peut coder sur un intervalle de : $[-2^{n-1} - 1, 2^{n-1}-1]$