Mémento C6 www.infodauphine.com : Le binaire (base 2)

Base 10 et Base 2

Base 10: le système décimal (puissance de 10). On compte en faisant des groupes de 10.
Base 2: le système binaire (puissance de 2). On compte en faisant des groupes de 2.

Binaire non signé

1	0	0	1	1
2^4	2^3	2^2	2^1	2^0
16	0	0	2	1

10011 en base 2 (binaire) ➔ 19 en base 10 (décimal)

Conversion en base 2 d'un nombre décimal

On calcul le **quotient entier par 2** et le **reste** de cette division. On recommence jusqu'à ce que le **quotient soit nul**. Le dernier reste obtenu est alors le 1er bit et le premier reste est le dernier bit (*rappel*: le dernier bit indique si le nombre est pair ⇔ le reste d'un nombre par 2 indique si le nombre est pair).

Nombre de bits

Un **octet** est formé de **8 bits** (c'est l'unité conventionnelle en binaire, Kilo-octet = 2^{10} (1024) octets (Kilo➔ Méga ➔ Giga)
19 sur un octet ➔ 00010011 . 19 sur deux octects (16 bits): 00000000 00010011

Opérations

Addition 1 +1 = 0 et je retiens 1 1 +1+1 = 1 et je retiens 1	**En VBA:** une variable *r* pour la retenue, une variable *s* pour la somme entre les 2 bits et la retenue, en parcourant les bits depuis la droite. Pour calculer la retenue et le bit d'addition, on fait soit des branchements, soit on utilise le reste par 2 (pour le bit d'addition) et le quotient entier par 2 (pour la retenue) de la somme obtenue.

Décalage des bits vers la **gauche**: **multiplication** par 2. Vers la **droite**: **division** entière par 2.
Les **derniers bits** indiquent lorsqu'ils sont **nuls** par quel puissance de 2 le nombre est **divisible** (par 2 avec 1 bit nul à la fin, par 4 avec les 2 derniers bits nuls, etc. Logique: finit par 00 ⇔ c'est 2 divisions par 2 sans reste

Bit de signe et représentation valeur absolue

Le premier bit (le plus à gauche) n'est plus une puissance de 2, mais **le signe du nombre** (1 pour négatif)
On perd donc une puissance de 2. sur 8 bits: -127 = 1111 1111 , 127 = 0111 1111

Complémentaire (complément à 1)

0➔1 1➔0 exemple : 10101➔01010 (ils sont complémentaires l'un de l'autre)

Complément à 2

Un **nombre négatif** en **complément à 2** est le **complémentaire** de son équivalent positif auquel on **ajoute 1**.
En complément à 2, le **premier bit** représente une **puissance de 2 négative**.
Un nombre positif sera donc là aussi un nombre dont le 1er bit est nul

1	0	0	1	1
-2^4	2^3	2^2	2^1	2^0
-16	0	0	2	1

10011 en complément à 2 ➔ -13 , mais 00011 ➔ 3

En complément à 2 sur 8 bits: - 127 = *1000 0001* (car -128+1), -128 = *1000 0000* (car -128+0),
-1 = *1111 1111* (-128+127), mais 127 est toujours *0111 1111* (car 0*-128 + 127)

Valeurs maximales et nombres de valeurs

	Non signé	**En valeur absolue**	**En complément à 2**
valeur **maximale** sur **8** bits	2^8 -1 (255)	2^7 - 1 (127)	2^7 -1 (127)
valeur **maximale** sur **n** bits	2^n - 1	2^{n-1} -1	2^{n-1} -1
valeur **minimale** sur **n** bits	0	$-(2^{n-1}-1)$	$-(2^{n-1})$
nombre de **valeurs** sur **8** bits	2^8 (256)	2^8 - 1 (255)	2^8 (256)
nombre de **valeurs** sur **n** bits	2^n	2^n - 1	2^n

Rappels

Les nombres **positifs** sont exprimés **de la même façon** dans les 2 variantes.
Les nombres binaires dont le premier bit est nul sont forcément positifs, quel que soit la variante.
En effet, en **binaire non signé**, le 1er bit représente $0 * 2^7$, alors qu'en **complément à 2**, il représente $0 * -2^7$
Les nombres **négatifs**:

- ne peuvent pas être exprimées en Binaire non signé
- sont exprimés avec premier bit non nul en représentation valeur absolue
- sont exprimés avec un premier bit non nul représentant $(1* -2^{n-1})$ en complément à 2