

Système de numération et Codage :

- Un système de numération est défini par :
 - la base du système : b
 - les symboles / digits

Exp : système binaire, décimal, octal, hexadécimal
 (2) (10) (8) (16)

Représentation des nombres positifs :

$$N_{(b)} = \sum_{i=0}^{n-1} a_i b^i = (a_n \dots a_1 a_0)_{(b)}$$

b : base du système

a_i : chiffre de rang i ($a_i < b$) tire

* Poids fort : Most significant bit (MSB)

* Poids faible : Least significant bit (LSB)

Exp : 1992 $\rightarrow$ LSB

LSB MSB

$$1992,65 = 1 \cdot 10^3 + 9 \cdot 10^2 + 9 \cdot 10^1 + 2 \cdot 10^0 + 6 \cdot 10^{-1} + 5 \cdot 10^{-2}$$

Partie entière Partie décimale

• Le système décimal :

Comptage en décimal :

Sur 1 position : $0, 1, \dots, 9 = 10^1 - 1$

Sur 2 positions : $0, 1, \dots, 99 = 10^2 - 1$

...

Sur n positions : n combinaisons

$$\text{avec } \begin{cases} \min = 0 \\ \max = 10^n - 1 \end{cases}$$

• Le système binaire :

Utilise 2 bits : "0" et "1"

$\rightarrow$ Un nombre à n bits distingue 2^n états

$\rightarrow$ des puissances successives de 2 (4, 8, 16, 32, ...)

sont appelés poids binaires.

$$\text{Exp : } (1011)_2 = 1 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 1 \cdot 2^3 \\ = 1 + 2 + 8 = (11)_{10}$$

* Conversion d'un système de numération Vers un autre :

• Conversion d'une base quelq $\rightarrow$ base 10 :

$\rightarrow$ Faire la somme du développement polynômial.

$$\text{Exp : } (BAC)_{16} = B \cdot 16^2 + A \cdot 16^1 + C \cdot 16^0 \\ = 11 \cdot 16^2 + 10 \cdot 16 + 12 \cdot 1 \\ = (2988)_{10}$$

• base 10 $\rightarrow$ base 2 :

Méthode 1 : division successives :

Exp :

$$\begin{array}{r|l} 68 & 2 \\ \hline 0 & 34 \\ 0 & 17 \\ 0 & 8 \\ 1 & 4 \\ 0 & 2 \\ 0 & 1 \\ 1 & 0 \end{array} \quad \begin{array}{l} \text{2} \\ \text{2} \\ \text{2} \\ \text{2} \\ \text{2} \\ \text{2} \\ \text{2} \end{array}$$

Sens de Lecture

$$68_{(10)} = 1000100_{(2)}$$

Méthode 2 : soustractions successives

• On retire du nombre, la plus grande puissance de 2 possible. Ensuite, on retire la plus grande puissance de 2 possible à ce reste etc.

Si on peut retirer le poids binaire, on note 1, sinon on note 0.

Exp : $n = 68$

Poids	64	32	16	8	4	2	1	
68 - ...	4	4	4	4	0	0	0	
$(68)_{10} =$	(1	0	0	0	1	0	0)	2

$$m = 325$$

Poids	256	128	64	32	16	8	4	2	1
325 - ...	69	69	5	5	5	5	1	1	0

$$(325)_{10} = (1\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 1)_2$$

+ Exo d'un nombre réel: $P. \frac{45}{625}$ P. fractionnelle

- P. entière : Méthode des divisions successives
- P. fractionnelle : Méthode des multiplications successives par 2.

$$\text{Exemple: } 45,625 \rightarrow 45 \mid 2$$

$$\Rightarrow (45)_{10} = (101101)_2$$

$$\begin{array}{l} 0,625 \times 2 = 1,25 \\ 0,25 \times 2 = 0,5 \\ 0,5 \times 2 = 1 \end{array} \quad \begin{array}{l} \text{Sens de lecture} \\ \downarrow \end{array}$$

$$\Rightarrow (0,625)_{10} = (0,101)_2$$

$$\Rightarrow (45,625)_{10} = (101101,101)_2$$

Rq: Le nombre de bits après la virgule détermine la précision.

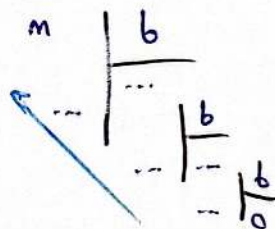
Une longueur fixe en base 10, peut être infinie en base b.

$$\begin{array}{l} (0,6)_{10} \rightarrow (0,1001)_2 \\ 0,6 \times 2 = 1,2 \\ 0,2 \times 2 = 0,4 \\ 0,4 \times 2 = 0,8 \\ 0,8 \times 2 = 1,6 \end{array}$$

- base 10 vers une base X:

La conversion se fait en prenant les restes des div successives sur la base X

en sens inverse



• Conversion d'une base X à une base Y

→ Il n'y a pas de méthode directe

On doit passer par la base 10 ou base 2.

• binaire → Octal / Hexa:

→ Découper les nombres binaires en groupes de 3 bits / 4 bits (en commençant par la droite) puis traduire chaque groupe par son équivalent en octal / hexa.

Exp:

$$(001\ 011\ 100, 010)_2 = (1\ 3\ 4, 2)_8$$

$$(01011\ 100, 0100)_2 = (5\ C, 4)_{16}$$

Rq: Le regroupement se fait de droite à gauche pour la partie entière

• De gauche à droite pour la partie fractionnelle

• Octal / Hexa → binaire

→ L'inverse du procédé précédent.

Remplacer chaque symbole par sa valeur en 3 bits / 4 bits.

$$\text{Exp: } (B43)_{16} = (1011\ 0100\ 1001)_2$$

Hexa	Octal	Binaire				Décimal
0	0	0	0	0	0	0
1	1	0	0	1	1	1
2	2	0	1	0	2	2
3	3	0	1	1	3	3
4	4	1	0	0	4	4
5	5	1	0	1	5	5
6	6	1	1	0	6	6
7	7	1	1	1	7	7
8	10	1	0	0	0	8
9	11	1	0	0	1	9
A	12	1	0	1	0	10
B	13	1	0	1	1	11
C	14	1	1	0	0	12
D	15	1	1	0	1	13
E	16	1	1	1	0	14
F	17	1	1	1	1	15
1n	8n	1	0	0	0	16

Représentation des nombres signés :

- Le signe (+) et (-) sont identifiés par un bit, dit le bit signe $\rightarrow$ nbr signé
- Le format de représentation doit être fixe pour identifier le bit de signe.

Dans une base b au format : MSB = bit de signe

si $N =$

b-1							
-----	--	--	--	--	--	--	--

 N est négatif
 $N =$

0							
---	--	--	--	--	--	--	--

 N est positif

Exp : $(721,6)_8$: Négatif
 $(0100)_2$: Positif

Les codes :

$\rightarrow$ Faire correspondre à un caractère un symbole.

2 types : $\left\{ \begin{array}{l} \text{code numérique : codage des nombres sautés} \\ \text{" alpha-numérique : codage d'une information globale} \end{array} \right.$

Les codes binaires :

Des combinaisons de bits qui expriment une information en binaire.

Les codes numériques pondérés :

Des codes qui donnent pour chaque combinaison, un équivalent décimal dans lequel chaque rang d'éléments binaires a un poids précis.

Le code binaire pur / naturel :

Exp : Sur 8 bits : $(7)_{10} = (0000\ 0111)_2$

Le code DCB / BCD : (décimal codé binaire)

Il consiste à représenter chaque chiffre d'un nombre décimal par son équivalent sur 4 bits.

$(236)_{10} \rightarrow (0010\ 0011\ 0110)_{DCB} \rightarrow 12\text{ bits}$

$(236)_{10} \rightarrow (1110\ 1100)_2$ (binaire pur) $\rightarrow 8\text{ bits}$

$\rightarrow$ Ce n'est pas un système de numération et ses nombres en DCB ne sont pas des nombres binaires purs.

$\rightarrow$ Il faut plus de bits en DCB pour représenter des nombres décimaux de plus d'un chiffre, qu'en binaire.

Opérations en DCB : Addition :

si le résultat $> 9 \Rightarrow$ Ajouter en additionnant +6
 $(=1001)$ $(=0110)$

(Dû au fait qu'on représente un nombre modulo 10 avec un code modulo 16 : $16 = 10 + 6$)

Exp : $08 + 07$

0000	1000
+	0000 0111
0000	1111 > 1001
	+0110 $\leftarrow$
0001	0101 $\checkmark$
	<u>1</u> <u>5</u>

$08 + 07$

0000	1001
+	0000 1000
0001	0001 < 1001
	$\rightarrow +0110$
0001	0111 $\checkmark$
	<u>1</u> <u>1</u>

$\hookrightarrow$ Rq : Lorsqu'il y a une retenue auxiliaire, on ajoute 6 au résultat même si la valeur est < 9 .

Soustraction :

• Lorsque le résultat $< 9 \Rightarrow$ on fait rien

• Résultat $> 9 \Rightarrow$ on soustrait 6 au résultat

• Lorsqu'il y a une retenue soustractive, on soustrait 6 même si la valeur < 9 .

Exp : $83 - 69,2 = 13,8$

1000	0011	0000
-	0110 1001	0010
0001	1001	1110
	-0010	-0110
0001	0011	1000
	<u>1</u> <u>3</u> <u>8</u>	

Les codes numériques non pondérés :

Les bits n'ont pas de poids prédéfini. La valeur du nombre ne dépend pas de la position des bits, mais d'une autre logique.

Code Gray / Reflex

n° valeurs successives ne diffèrent que d'un seul bit.

- Commence avec 0000...
- Pour passer à la valeur suivante, on inverse le bit le plus à droite possible sans répéter une valeur déjà utilisée. Ex: 00 → 01 → 11 → 10 ...

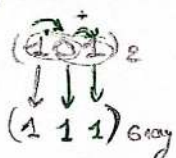
Décimal	Gray
0	0 0 0 0
1	0 0 0 1
2	0 0 1 1
3	0 0 1 0
4	0 1 1 0
5	0 1 1 1
6	0 1 0 1
7	0 1 0 0
8	1 1 0 0
9	1 1 0 1
10	1 1 1 1
11	1 1 1 0
12	1 0 1 0
13	1 0 1 1
14	1 0 0 1
15	1 0 0 0

symétrie 1
symétrie 2
symétrie 3
(... symétrie 4)

Rq: Le code Gray a la même densité (m même nombre de bits) que le code binaire pur.
→ Code cyclique qui se referme sur lui-même.

• Passage binaire → Gray

- Garder le MSB
- Faire la somme des bits adjacents 2 à 2 en rejetant la retenue.



Exp:

$$(N)_2 = 11100110$$

$$(N)_{\text{Gray}} = 10100101$$

He: On additionne le nombre à lui-même, on le décale vers la gauche d'un rang, on abandonne le chiffre de plus petit poids (LSB) (on n'est pas capte des retenues).

Exp:

$$\begin{array}{r} 11000110 \\ 11000110 \\ \hline (10100101)_{\text{Gray}} \end{array}$$

• Passage Gray → binaire:

- Garder le même MSB.
- Faire la somme de chaque nouveau bit créé avec le bit adjacent du code Gray en rejetant la retenue.

$$(N)_{\text{Gray}} = 10100101$$

$$(N)_2 = 11000110$$

• Complément à b-1 / Complément à b:

(Sert à représenter les nombres négatifs en binaire)

- On appelle complément à b-1 d'un nombre N, un autre nombre noté $N_{(b-1)}$ tq:

$$N + N_{(b-1)} = b^m - 1$$

avec m: nombre de bits de N

- Le complémentaire à b de N, noté N_{cb} est donné par: $N_{cb} = N_{(b-1)} + 1$

Exemple: $N = (48)_{10} = (00001100)_2 \Rightarrow 8 \text{ bits (m)}$

Le complément à 1: $(11110011)_{C1}$

(on inverse tous les bits)

Le complément à 2: $(11110100)_{C2}$

($\Rightarrow 11110011 + 1$)

• Représentation des valeurs négatives:

3 façons ① Signe

Sauf le bit du signe (MSB) change.

$(5)_{10} = (00000101)_2 \Rightarrow 8 \text{ bits}$

↑ bit de signe

$(-5)_{10} = (10000101)_2$

⇒ Valeur absolue sur (m-1) bits

⇒ Intervalle de valeurs

représentées: $[-2^{m-1} + 1, 2^{m-1} - 1]$

② Complément à b-2:

MSB: bit de signe

$(5)_{10} = (00000101)_2$

$(-5)_{10} = (11111010)_{C2}$

③ Complément à b:

$(-5)_{10} = (11111011)_{C2}$

→ C'est la façon utilisée en informatique.
→ $[-2^{m-1}, 2^{m-1} - 1]$

Opérations arithmétiques :

Addition en binaire :

$0+1=1$
 $1+1=10$

$$\begin{array}{r}
 1011 \rightarrow 11 \\
 + 0110 \rightarrow 6 \\
 \hline
 10001 \rightarrow 17
 \end{array}$$

Soustraction en binaire :

Méthode directe :
 $1-0=1$
 $0-0=0$
 $1-1=0$

$$\begin{array}{r}
 1011 \rightarrow 11 \\
 - 0110 \rightarrow 6 \\
 \hline
 0101 \rightarrow 5
 \end{array}$$

$0-1 \Rightarrow$ on doit emprunter à gauche
 • Si à gauche, on a un 0, on continue à emprunter jusqu'à trouver 1
 • Chaque emprunt ajoute 2 au bit actuel.

Méthode indirecte :

On fait une addition en utilisant un complément à 2.
 $11-6=5$

$$\begin{array}{r}
 1011 \\
 + 1010 \\
 \hline
 10101
 \end{array}$$

$$\begin{array}{r}
 (11)_{10} = (1011)_2 \\
 (6)_{10} = (0110)_2 \\
 = (1001)_{c2} \\
 = (1010)_{c2}
 \end{array}$$

Bit de débordement $\Rightarrow$ A rejeter

Rq : Lorsqu'on fait la somme de 2 nombres de même signe, on doit prévoir un bit de débordement, à part les bits de signe.

Multiplication en binaire :

On décale à gauche à chaque fois et on fait la sommation totale. $11 \times 6 = 66$

$$\begin{array}{r}
 1011 \\
 \times 110 \\
 \hline
 + 0000 \\
 + 10110 \\
 + 101100 \\
 \hline
 1000110 \Rightarrow (66)_{10}
 \end{array}$$

Division en binaire :

$\Rightarrow$ Suite de soustractions
~~RRRRR~~, on fait descendre un bit.

Exp : $44/11 = 4$

$$\begin{array}{r}
 101100 \mid 100 \\
 - 100 \\
 \hline
 0011 \\
 - 110 \\
 \hline
 0100 \\
 - 100 \\
 \hline
 000
 \end{array}$$

L'addition en octal / Hexa :

Comme en décimal, mais avec des restes qui se dépassent à 8 ou 16.

Octal :

$(4355)_8 + (451)_8$

$$\begin{array}{r}
 4355 \\
 + 451 \\
 \hline
 (5056)_8
 \end{array}$$

$6+5=11=1 \cdot 8^1 + 3 \cdot 8^0 \Rightarrow$ on pose 3
 $4+3+1=8 \Rightarrow$ on pose 0 et on retient 1

Hexa :

$(4865)_{16} + (7A51)_{16} = ?$

$$\begin{array}{r}
 4865 \\
 + 7A51 \\
 \hline
 (C2B6)_{16}
 \end{array}$$

$A+8=18=1 \cdot 16 + 2 \cdot 16^0 \Rightarrow$ on pose 2

La soustraction en octal / Hexa :

Méthode directe :

On peut emprunter un bit à gauche (qui devient 8 ou 16)

$(130)_8 - (66)_8 = ?$

$$\begin{array}{r}
 130 \\
 - 66 \\
 \hline
 (42)_8
 \end{array}$$

Méthode indirecte :

$(035E)_{16} - (0F2C)_{16}$

$(0F2C)_{16} = (F0D3)_{c16} = (F0D4)_{c16}$

$$\begin{array}{r}
 035E \\
 + F0D4 \\
 \hline
 (F432)_{16}
 \end{array}$$

$E+4=12=1 \cdot 16 + 2 \cdot 16^0$
 $D+5+1=19=1 \cdot 16 + 3 \cdot 16^0$

$(0130)_8 - (0066)_8 = (0130)_8 + (-0066)_8$
 $= (0130)_8 + (7711)_{c7}$
 $= (0130)_8 + (7712)_{c8}$

$$\begin{array}{r}
 0130 \\
 + 7712 \\
 \hline
 00042
 \end{array}$$

$1+7=8=1 \cdot 8^1 + 0 \cdot 8^0$