Université Ibn Zohr
École Nationales des Sciences Appliquées-Agadir

Systèmes de numération
Codage de l'information

Professeur A. ABENAOU
Département de Génie Informatique

a.abenaou@uiz.ac.ma

Année universitaire 2020/2021

1

Introduction

- Un des principes fondamentaux de l'informatique est que : Toute information (instructions ou donnée) est représentée dans un ordinateur par des nombres.
- Ces nombres sont des nombres binaires.
- Le codage de l'information permet d'établir une correspondance entre la représentation externe de l'information et sa représentation binaire.

Types d'Informations

- Les nombres : 1, 2, 3675, 123456789,...
- Les lettres: a, b, c, é, ê, è, à, ...
- Les images
- Les sons
- Etc...

Les Systèmes de Numération

- La base d'un système de numération pondéré est le nombre de chiffres différents qu'utilise ce système de numération.
- Système pondéré: numération de position dont la place du chiffre dans le nombre est importante

- En *électronique numérique* , les systèmes les plus utilisés sont :
  - le système binaire
  - le système octal
  - le système hexadécimal

Les Systèmes de Numération

- De manière générale, un nombre s'écrivant

$N = a_{n-1}\, a_{n-2} \ldots a_1\, a_0 . a_{-1} \ldots a_{-m}$ dans une base B s'écrit :

$$N = a_{n-1} B^{n-1} + a_{n-2} B^{n-2} + \ldots + a_1 B^1 + a_0 B^0 + a_{-1} B^{-1} \ldots + a_{-m} B^{-m}$$

dans la base décimale.

Où les $a_i \in \{0,1,\ldots, B-1\}$ *représentent les chiffres du rang i du nombre N et* $B^0 = 1$.

On note : $N = (a_{n-1}\, a_{n-2} \ldots a_2\, a_1\, a_0 . a_{-1} \ldots a_{-m})_B$

Les Systèmes de Numération

- **Système de *Numération décimale:***

C'est le système que nous utilisons usuellement = système de **base** $B = 10$ qui utilise donc 10 symboles (ou chiffres) différents: 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9

$a_i \in \{ 0,1,2,3,4,5,6,7,8,9 \}$

$$N = a_{n-1}\,10^{n-1} + a_{n-2}\,10^{n-2} + \ldots + a_1\,10^{1} + a_0\,10^{0} + a_1\,10^{-1} + \ldots + a_m\,10^{-m}$$

Exemple:

$$(5392{,}75)_{10} = 5x\,10^{3} + 3\,x10^{2} + 9\,x10^{1} + 2\,x10^{0} + 7x10^{-1} + 5x10^{-2}$$

Les Systèmes de Numération

- ***Numération décimale :***

- **Les puissances de 10 sont appelés les poids ou les valeurs de position**

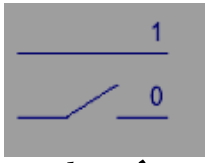
- **Le poids est égal à la base élevée à la puissance de son rang.**

	Unité	Dizaine	Centaine	Milliers	10*Milliers	100*Milliers
Chiffre	a_0	a_1	a_2	a_3	a_4	a_5
Rang	0	1	2	3	4	5
Poids	10^0	10^1	10^2	10^3	10^4	10^5

Les Systèmes de Numération

• Numération binaire :

- *Toute communication à l'intérieur de l'ordinateur est faite avec des signaux électriques*
- *Pour la simplicité et fiabilité, ces signaux ont deux états seulement :*
  1 → allumé (présence de signal électrique)
  0 → éteint (absence de signal électrique)



- *Une unité d'information (0 ou 1) est appelée bit (de l'anglais binary digit)*

Les Systèmes de Numération

• Numération binaire :

- La numération en base $B = 2$ utilise deux symboles 0 et 1.
- Cette base est très commode pour distinguer deux états logiques fondamentaux (Vrai et faux).

$$N_{10} = a_{n-1}\,2^{n-1} + a_{n-2}\,2^{n-2} + \dots + a_1\,2^1 + a_0\,2^0 + a_1\,2^{-1} + \dots + a_m\,2^{-m}$$

a_{n-1} = bit de poids fort ; a_{-m} = bit de poids faible avec $a_i \in \{0,1\}$

$$\rightarrow \quad N_2 = a_{n-1}\,a_{n-2}\,\dots\,a_1\,a_0\,,\,a_1\,\dots\,a_m$$

Exemple:

$$(22)_{10} = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

$$(22)_{10} = (10110)_2$$

Les Systèmes de Numération

- ***Numération binaire :***

- Les puissances successives de 2 (1, 2, 4, 8, 16, 32, 64, 128, 256,...) sont appelées *poids binaires*

$$2^n = (10000....00000)_2$$

(n zéro)

Rang n	0	1	2	3	4	5	6	7	8	9	10	11
Poids 2^n	1	2	4	8	16	32	64	128	256	512	1024	2048

Les Systèmes de Numération

• Numération binaire :

- 1 bit : 
  0 ou 1 = 2 (2^1) valeurs possibles
- 2 bits : 
  00,01,10 ou 11 = 4 (2^2) valeurs possibles
- 3 bits : 
  000,001,010,011,100,101,110 ou 111 = 8 (2^3) valeurs possibles
- 4 bits : 
  0000,0001,0010,…,1111 = 16 (2^4) valeurs possibles
- 8 bits : 
  00000000, 00000001,…,11111111 = 256 (2^8) valeurs possibles
- 16 bits : 
  0000000000000000, 1111111111111111 = 65 536 (2^{16}) valeurs possibles
- 32 bits : 
  00000000000000000000000000000000,… = 4 294 967 296 valeurs possibles

Les Systèmes de Numération

• *Numération binaire :*

- **Capacité d'une base = plus grand nombre, exprimé en base 10, que l'on peut représenter avec n chiffres dont on dispose dans ladite base**

  En utilisant n bits, on peut former 2^n nombres différents et le plus grand d'entre eux est égal à (2^n-1).

- ***Exemple :***

  Capacité du nombre binaire de 5 bits ?

Les Systèmes de Numération

- **Numération binaire :** *Donner les 16 premières combinaisons*

Combinaison	a_3	a_2	a_1	a_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Les Systèmes de Numération

- *Numération octale :*

C'est le système de **base** $B = 8$ qui utilise 8 symboles différents :

0, 1, 2, 3, 4, 5, 6, 7

$$N_{10} = a_{n-1}\,8^{n-1} + a_{n-2}\,8^{n-2} + \ldots + a_1\,8^1 + a_0\,8^0 + a_1\,8^{-1} + \ldots + a_m\,8^{-m}$$

$$N_8 = a_{n-1}\,a_{n-2}\,\ldots a_1\,a_0 . a_1\,\ldots a_m$$

$\rightarrow$ $a_i \in \{0,1,2,3,4,5,6,7\}$

Exemple: $(350)_8 = 3 \times 8^2 + 5 \times 8^1 + 0 \times 8^0$

$= 192 + 40 = (232)_{10}$

Les Systèmes de Numération

- *Numération octale :*

- La succession des nombres par ordre croissant est le suivant :

0,1,2,3,4,5,6,7,10,11,12,13,14,15, 16, 17, 20,21,...

Rang n	**0**	**1**	**2**	**3**	**4**	**5**
Poids 8^n	**1**	**8**	**64**	**512**	**4096**	**32768**

Les Systèmes de Numération

- *Numération hexadécimale :*

C'est le système de **base** ***B*** **= 16** qui utilise 16 symboles différents : **0, 1, 2, 3, 4, 5, 6, 7,8,9,A,B,C,D,E,F**

$$N_{10} = a_{n-1}.16^{n-1} + a_{n-2}.16^{n-2} + ... + a_0.16^0 + a_{-1}.16^{-1} + ...+ a_{-m}.16^{-m}$$

$$N_{16} = (a_{n-1}\ a_{n-2}\ ....\ a_2\ a_1\ a_0\ ,\ a_{-1}....\ a_{-m})_{16}$$

→ $a_i \in \{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F\}$

Exemple: $(AF)_{16} = 10 \times 16^1 + 15 \times 16^0$

$= 160 + 15 = (175)_{10}$

Les Systèmes de Numération

- *Numération hexadécimale :*

Base 10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Base 16	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

La succession des nombres par ordre croissant est le suivant:

0, 1, 2, 3, 4, 5,6,7,8,9,A,B,C,D,E,F,10,11,12,13,.....,1E,1F,20,21,....

Rang n	**0**	**1**	**2**	**3**	**4**
Poids 16^n	**1**	**16**	**256**	**4096**	**65536**

Les Systèmes de Numération

•Numération hexadécimale :

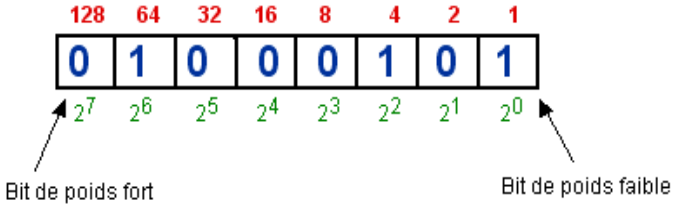
- Exemples d'utilisation
  - adresses de mémoire
  - codes d'erreur
  - codes des couleurs

- Facilite la représentation d'une séquence trop longue
  - 10110110001000000110001101001 1 (binaire)
  - 2D8818D3 (hexadécimale)
  - Codes des couleurs: FFFFFF

Les Systèmes de Numération

- **Unités de quantité d'Information**
  - *bit – une unité binaire (0 ou 1)*
  - *octet (ou Byte) – groupe de 8 bits*
  - *Kilo-octets (Ko) = 1024 octets*
  - *Méga-octets (Mo) = 1024Ko = 1048576 octets*
  - *Giga-octets (Go) = 1024 Mo = 1073741824 octets*

- **Description d'un octet**



Conversion d'une base B vers la base 10

- ***Il suffit de calculer en base 10 la somme totale des puissances pondérées dans la base B***

- *Base 2:* $(1001011)_2$?

$(1001011)_2 = 1x2^6 + 0x2^5 + 0x2^4 + 1x2^3 + 0x2^2 + 1x2^1 + 1x2^0$

$= (75)_{10}$

- *Base 8:* $(723)_8$?

$\mathbf{(723)_8 = 7x\ 8^2 + 2x8^1 + 3x\ 8^0 = (467)_{10}}$

- *Base 16:* $(A3D)_{16}$?

$\mathbf{(A3D)_{16} = 10x\ 16^2 + 3x\ 16^1 + 13x\ 16^0 = (2621)_{10}}$

Conversion de la base 10 à une base B

- ***Méthode par soustractions successives***

Elle utilise le développement polynomial

$$N_{10} = a_{n-1} B^{n-1} + a_{n-2} B^{n-2} + ... + a_1 B^1 + a_0 B^0$$

$$\Rightarrow \quad N_B = (a_{n-1}\, a_{n-2} \ldots a_2\, a_1\, a_0)_B$$

Procédé:

- *Dresser une table donnant les valeurs des différentes puissances de la base B*
- *Au nombre décimal donné, retrancher la plus grande puissance de B possible*
- *Répéter le processus à partir des restes obtenus*

- ***Remarque:*** *cette méthode ne s'applique qu'aux nombres entiers*

Conversion de la base 10 à une base B

- ***Méthode par soustractions successives : Exemples***

$(1251)_{10} = (\ ?\ )_2$

n	0	1	2	3	4	5	6	7	8	9	10	11
2^n	1	2	4	8	16	32	64	128	256	512	1024	2048

1251
-1024 — $1x2^{10}$
227
- 128 — $1x2^7$
99
- 64 — $1x2^6$
35
- 32 — $1x2^5$
3
- 2 — $1x2^1$
1 — $1x2^0$

$$(1251)_{10} = 1x2^{10} + 0x2^9 + 0x2^8 + 1x2^7 + 1x2^6 + 1x2^5 + 0x2^4 + 0x2^3 + 0x2^2 + 1x2^1 + 1x2^0$$

$$= (10011100011)_2$$

Conversion de la base 10 à une base B

- ***Méthode par soustractions successives : Exemples***

$(6718)_{10} = (\ ?\ )_8$

n	0	1	2	3	4	5
8^n	1	8	64	512	4096	32768

$(6718)_{10} = 1 \times 8^4 + 5 \times 8^3 + 0 \times 8^2 + 7 \times 8^1 + 6 \times 8^0$

$= (1\ 5\ 0\ 7\ 6)_8$

$6718 - 4096 \times 1 = 1 \times 8^4$

$2622 - 512 \times 1 = 1 \times 8^3$

$2110 - 512 \times 1 = 1 \times 8^3$

$1598 - 512 \times 1 = 1 \times 8^3$

$1086 - 512 \times 1 = 1 \times 8^3$

$574 - 512 \times 1 = 1 \times 8^3$

$62 - 8 \times 7 = 7 \times 8^1$

$6 - 1 \times 6 = 6 \times 8^0$

0

Conversion de la base 10 à une base B

- ***Méthode par soustractions successives : Exemples***

$N = (3786)_{10} = (\ ?\ )_{16}$

- Cherchons la plus grande puissance de 16 contenue dans 3786.

n	0	1	2	3	4
16^n	1	16	256	4096	65536

On a 3768> 256 (16^2) et 3786 < 4096 (16^3)

- Nous retenons donc : 16^2
- Cherchons le plus grand multiple de 16 contenu dans N :

$$N = 14 \times 16^2 + 202$$

- Recommençons avec le reste et ainsi de suite jusqu'à l'obtention d'un reste inférieur à 16 :

$$202 = 12 \times 16^1 + 10$$

- Ce qui donne : $N = 14 \times 16^2 + 12 \times 16^1 + 10 \times 16^0$

Ou encore : $N = E \times 16^2 + C \times 16^1 + A \times 16^0$

Donc : $(3786)_{10} = (ECA)_{16}$

Conversion de la base 10 à une base B

- ***Méthode par divisions ou multiplications***

Procédé:

- *Tout nombre, non entier, sera converti en considérant:*
  - *Sa partie entière à laquelle on appliquera des divisions successives*
  - *Sa partie fractionnaire à laquelle on appliquera des multiplications successives*

- ***Remarque:***
- *Cette méthode est relativement plus simple et s'applique aussi bien aux nombres entiers qu'aux nombres non entiers.*

Conversion de la base 10 à une base B

- **Conversion de la partie entière**

✔ Diviser le nombre à convertir par la base B

✔ Conserver le reste

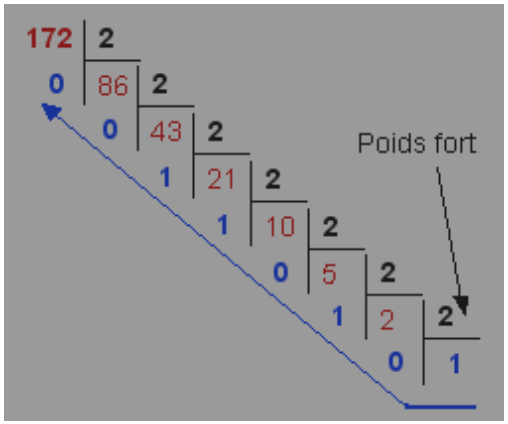
✔ Répéter le processus à partir du nouveau quotient obtenu

✔ Arrêter lorsque le quotient est nul

✔ Écrire les restes à partir du dernier et de gauche à droite pour obtenir le nombre en base B

Méthode par divisions

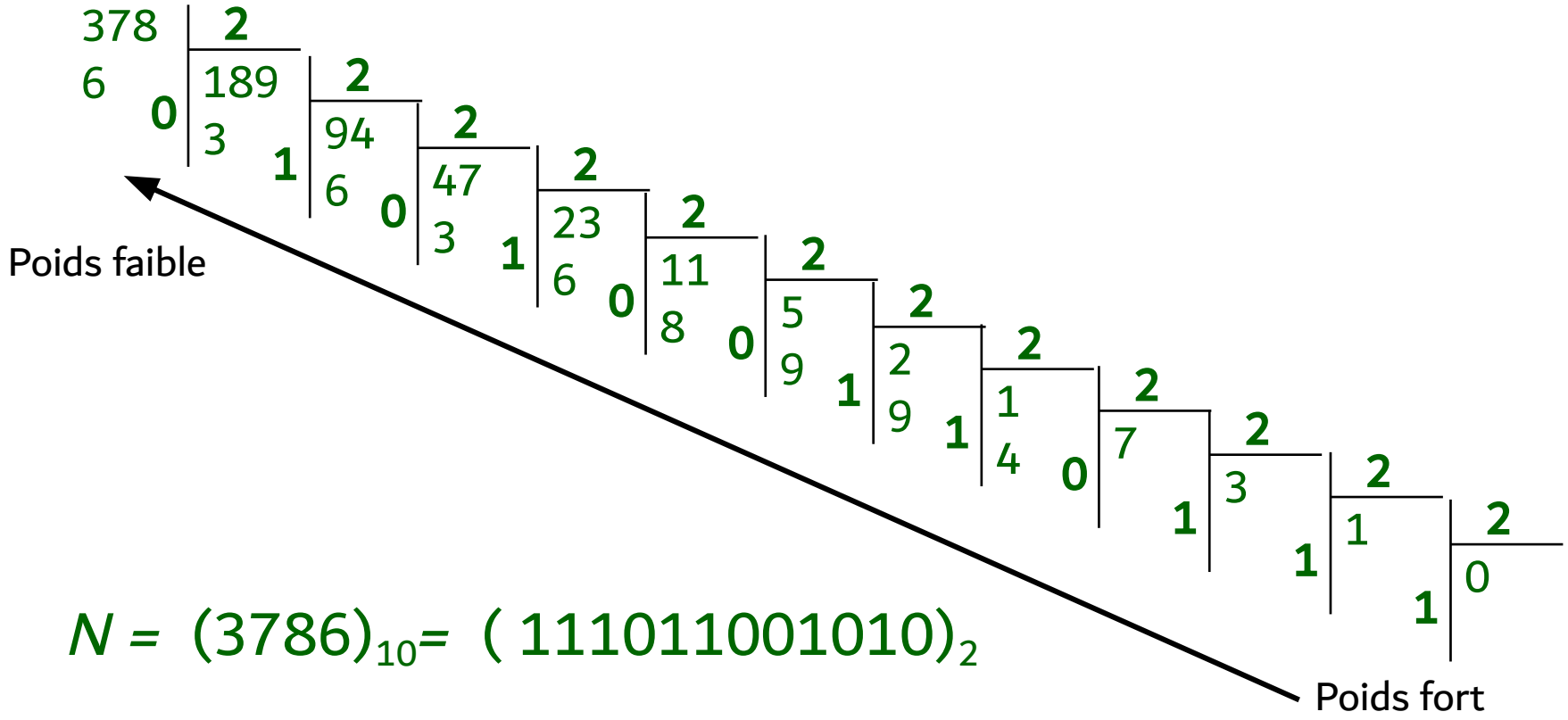
$(172)_{10} = (\ ?\ )_2$



Conversion de la base 10 à une base B

- ***Conversion de la partie entière: Exemples***

$(3786)_{10} = (\ ?\ )_2$



$N = (3786)_{10} = (111011001010)_2$

Conversion de la base 10 à une base B

- ***Conversion de la partie entière: Exemples***

$(358)_{10} = (\ ?\ )_{8}$

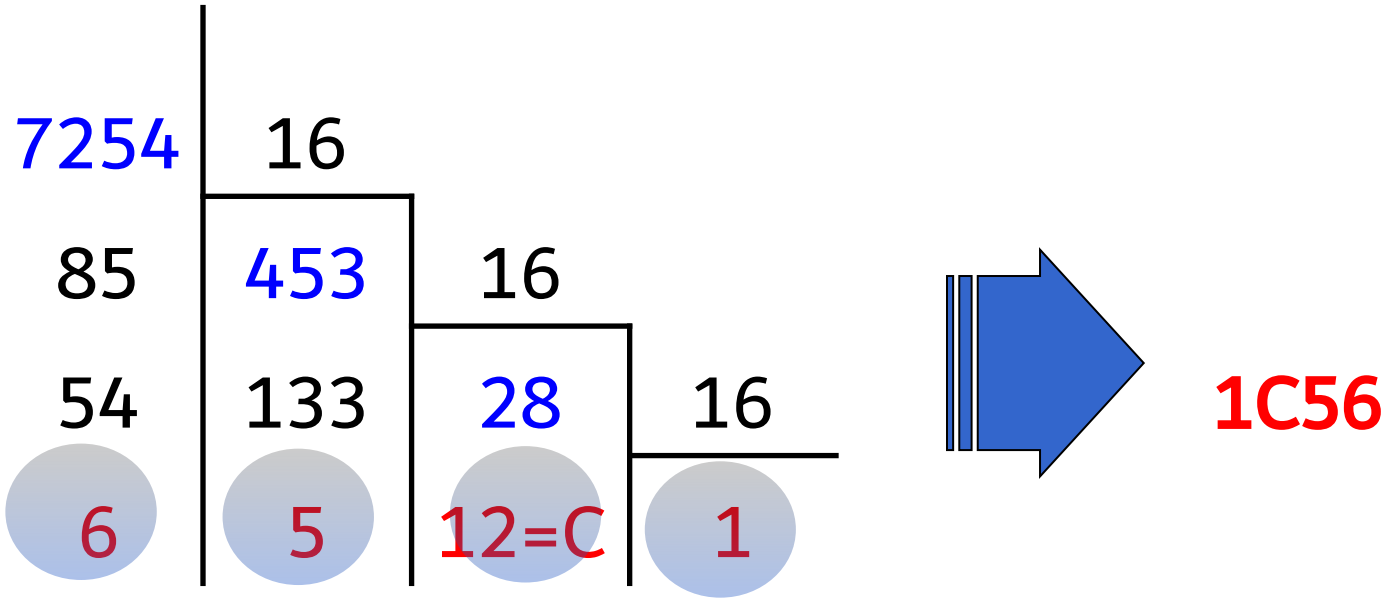
$(254)_{10} = (\ ?\ )_{16}$

$358_{10} = (546)_{8}$

$(254)_{10} = (\ FE\ )_{16}$

Conversion de la base 10 à une base B

Exemple 2



$(7254)_{10}=(1C56)_{16}$

Conversion de la base 10 à une base B

- ***Conversion de la partie fractionnaire***

✔ ***Multiplier la partie fractionnaire du nombre à convertir par la base B***

✔ ***Soustraire et Conserver sa partie entière***

✔ ***Répéter le processus à partir de la nouvelle partie fractionnaire obtenu***

✔ ***Arrêter lorsque la précision désirée est atteinte***

Exemple 1 : $(0,75)_{10} = (\ ?\ )_2$

$0,75 \times 2 = 1,5$ (on garde 1 et reste 0,5)

$0,5 \times 2 = 1,0$ (on garde 1 et reste 0 : terminé)

$(0,75)_{10} = 1 \times 2^{-1} + 1 \times 2^{-2} = (0,11)_2$

Conversion de la base 10 à une base B

- ***Conversion de la partie fractionnaire: Exemple 2***

$(0,65)_{10} = (\ ?\ )_2$

0,65 × 2 = 1,3 on garde 1, reste 0,3

0,3 × 2 = 0,6 on garde 0, reste 0,6

0,6 × 2 = 1,2 on garde 1, reste 0,2

0,2 × 2 = 0,4 on garde 0, reste 0,4

0,4 × 2 = 0,8 on garde 0, reste 0,8

0,8 × 2 = 1,6 on garde 1, reste 0,6

0,6 × 2 = 1,2 on garde 1, reste 0,2

….

$(0,65)_{10} = (0,10\underline{1001})_2$

Conversion de la base 10 à une base B

- ***Conversion de la partie fractionnaire: Exemple 3***

$(0,732)_{10} = (\ ?\ )_8$

$0,732 \times 8 = 5,856$ on garde 5, reste 0,856

$0,856 \times 8 = 6,848$ on garde 6, reste 0,848

$0,848 \times 8 = 6,784$ on garde 6, reste 0,784

$0,784 \times 8 = 6,272$ on garde 6, reste 0,272

$0,272 \times 8 = 2,176$ on garde 2, reste 0,176

....

D'où $\mathbf{(0,732)_{10} = (0,56662...)_8}$

Conversion du binaire vers Base 2^n et vice-versa

- **Binaire vers Octal**

✔ Grouper les bits par blocs de 3 à partir du bit de poids faible

✔ Convertir ensuite directement ces blocs en octal

Exemple : $(\underbrace{110}\ \underbrace{101}\ \underbrace{110}\ \underbrace{001}{,}\underbrace{001}\ \underbrace{111})_2 = (6561{,}17)_8$

Triplet

- **Octal vers Binaire**

✔ Traduire chaque chiffre du nombre en base 8 en nombre de 3 bits en base 2

Exemple : 3 1 5 7

$(3157)_8 = (\overbrace{011}\ \overbrace{001}\ \overbrace{101}\ \overbrace{111})_2$

Conversion du binaire vers Base 2^n et vice-versa

- **Binaire vers hexadécimal**

✔ Grouper les bits par blocs de 4 à partir du bit de poids faible

✔ Convertir ensuite directement ces blocs en hexadécimal

Exemple : $(1101\ 0111\ 0001)_2 = (D71)_{16}$

D 7 1 Quartet

- **Hexadécimal vers Binaire**

✔ Traduire chaque chiffre du nombre en base 16 en nombre de 4 bits en base 2

Exemple :

B C 3 4

$(BC34)_{16} = (1011\ 1100\ 0011\ 0100)_2$

Conversion d'une Base i vers une base j

- ***Les deux bases sont des puissances de 2***

✔ ***On utilise la base 2 comme base relais***

Base i ***Base 2*** ***Base j***

Exemple : Convertir en octal le nombre $(D71)_{16}$

$$(D71)_{16} = (\underbrace{1}_{6}\underbrace{101}_{5}\underbrace{110}_{6}\underbrace{001}_{1})_2 = (6561)_8$$

- ***Les deux bases ne sont pas des puissances de 2***

✔ ***On utilise la base 10 comme base relais***

Base i ***Base 10*** ***Base j***

Tableau de correspondance entre nombre de différentes bases

décimal (10)	Binaire (2)	Octal (8)	Hexadécimal (16)
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11

Opérations Mathématiques: Addition

• L'addition binaire classique est analogue à l'addition décimale. Il faut commencer par le bit de poids faible en utilisant les relations suivantes :

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 0 avec un report de 1

Exemple **:**

10 en binaire correspond à 2 en décimal

```
          1 1 1 1 1     :  report
  1 1 1 1 1 1
+ 1 0 1 1 0 1
-------------
1 1 0 1 1 0 0
```

Opérations Mathématiques: Addition

- • Les circuits numériques ne permettent pas d'additionner plus de deux nombres binaires à la fois. Les additions sont, en fait, une successions d'additions de deux nombres : le résultat de l'addition du premier et du deuxième est ajouté au troisième et ainsi de suite.

- • L'addition binaire est l'opération la plus importante des circuits numériques car les autres opérations comme la soustraction, la division et la multiplication en découlent.

- • Additionner deux nombre égaux revient à multiplier par $(10)_2$ en base 2, donc à lui ajouter un 0.

Exemple : 1 + 1 = 10 ; 1001 + 1001 = 10010 .

Opérations Mathématiques: Addition

- Calculer : $(BE)_{16} + (8D)_{16}$

1	1 1
190	B E
+141	+ 8 D
331	1 4 B

- Calculer

$(457)_8 + (673)_8$ $(5AF)_{16} + (9B6)_{16}$

Remarque:

L'addition peut faciliter la conversion d'une base vers la base décimale en décomposant le nombre à convertir

Exemple:$(14A6)_{16}$ est la somme de

$(1000)_{16} + (400)_{16} + (A0)_{16} + (6)_{16}$.

$\Rightarrow (14A6)_{16} = 1x16^3 + 4x\ 16^2 + 10\ x\ 16^1 + 6$

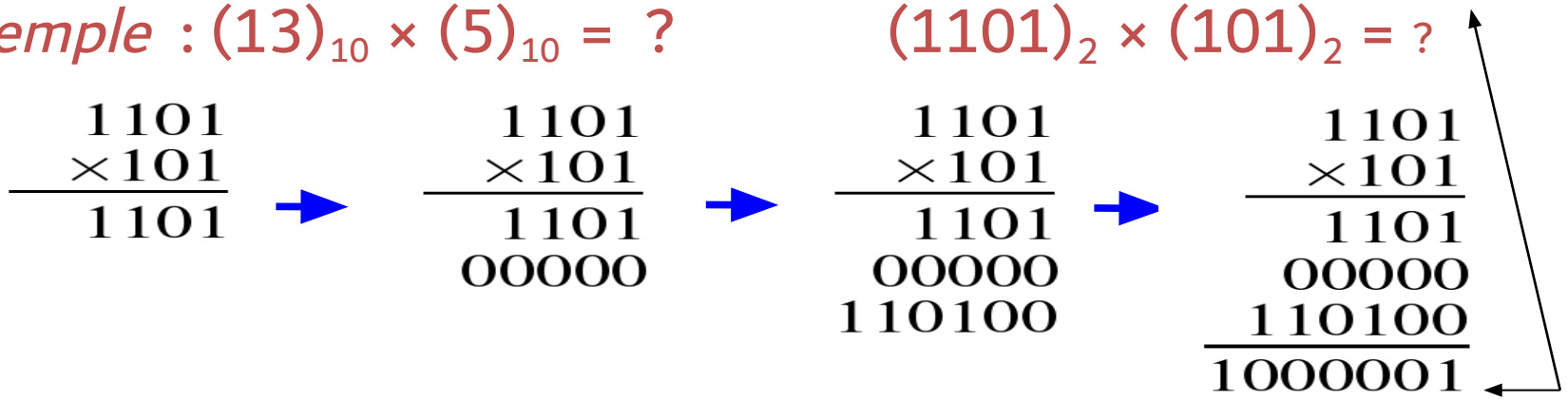
$\Rightarrow (14A6)_{16} = 4096 + 1024 + 160 + 6 = (5286)_{10}$

Opérations Mathématiques: Multiplication

- La multiplication binaire est analogue à la multiplication décimale en utilisant les relations suivantes :

$$0 \times 0 = 0$$
$$0 \times 1 = 0$$
$$1 \times 0 = 0$$
$$1 \times 1 = 1$$

Exemple : $(13)_{10} \times (5)_{10} = ?$ $(1101)_2 \times (101)_2 = ?$



Opérations Mathématiques: Soustraction

- Table:

0 - 0 = 0

0 - 1 = 1 et j'emprunte 1

1 - 0 = 1

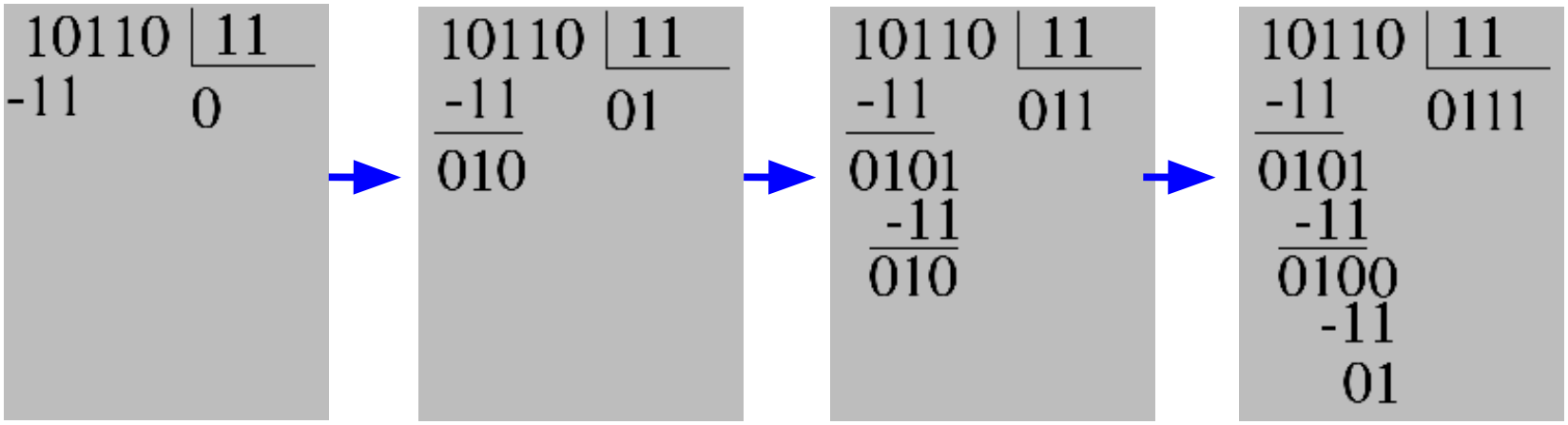
1 - 1 = 0

Exemple **:**

```
  1 0 1 1 0 1 1
-   1 0 1 1 1
        1   1 1        :  Emprunt
  0 1 0 1 1 0 0
```

Opérations Mathématiques : Division

- La division binaire s'effectue à l'aide de soustractions et de décalages, comme la division décimale, sauf que les digits du quotient ne peuvent être que 1 ou 0
- Le bit du quotient est 1 si on peut soustraire le diviseur, sinon il est 0



Représentation binaire des nombres relatifs

Représentation valeur absolue et Signée

- On ajoute un autre bit (à gauche) pour symboliser le signe :
  - Si ce bit = 0 ⇒ Le nombre est Positif
  - Si ce bit = 1 ⇒ Le nombre est négatif

- Exemple : $(29)_{10} = (11101)_2$ (représenté sur 5 bits)

$(+29)_{10} = (\ 0\ 11101\ )_2$ $\qquad$ $(-29)_{10} = (\ 1\ 11101)_2$

Le bit signe

Les bits grandeur

Nombre Signé

Représentation binaire des nombres relatifs

Valeur absolue et signée – Exemple sur 8 bits

- Limites sur 8 bits:

+ 127 $(0\ 1111111)_2$

+ 0 $(0\ 0000000)_2$

- 0 $(1\ 0000000)_2$

- 127 $(1\ 1111111)_2$

Inconvénient:

Cette représentation ne convient pas pour représenter les nombres négatifs : 0 a deux représentations, la somme de deux nombres opposés n'est pas nulle

Représentation binaire des nombres relatifs

Complément à 1 d'un nombre binaire

• Le complément à 1 « ou Complément restreint (CR) » d'un nombre est obtenu en inversant chaque bit (0 → 1 et 1 → 0).

⇒ Complémenter chaque bit de grandeur

- Exemple : $(+29)_{10} = (011101)_2$ (représenté sur 6 bits)

$CR(29)_{10} = (100010)_2$

- Limites sur 8 bits:

+ 127 $(0\ 1111111)_2$

+ 0 $(0\ 0000000)_2$

- 0 $(1\ 1111111)_2$

- 127 $(1\ 0000000)_2$

- **Sur n bits:**

  $x + CR(x) = 2^n - 1$

Représentation binaire des nombres relatifs

Complément à 2 d'un nombre binaire

Le complément à 2 « ou Complément Vrai (CV) » d'un nombre est le complément à 1 auquel on ajoute 1

$\Rightarrow$ CV = CR + 1

- Exemple : $(+29)_{10} = (011101)_2$ (représenté sur 6 bits)

$CR(29) = (100010)_2$

$CV(29) = CR(29) + 1 = (100011)_2$

$(-29)_{10} = CV(29)$

✔ Ce type de codage permet la représentation des nombres négatifs dans les ordinateurs et ce sur 8, 16 ou 32 bits .

Représentation binaire des nombres relatifs

Complément à 2 d'un nombre binaire

soit x un nombre binaire qui s'écrit sur n bits alors
CV(x) = - x

- **En effet :**

on a:

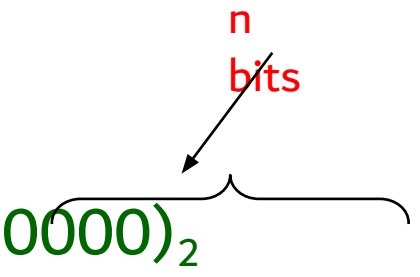
$$x + CR(x) = 2^n - 1$$

$$\Rightarrow \quad x + CR(x) + 1 = 2^n$$

$$\Rightarrow \quad x + CV(x) = 2^n = (1000....00000)_2$$

$$\Rightarrow \quad x + CV(x) = 0 \text{ sur n bits}$$

$$\Rightarrow \quad CV(x) = - x$$



Représentation binaire des nombres relatifs

• Représentation sur 4 bits des nombres positifs et négatifs dans les ordinateurs

	a_3	a_2	a_1	a_0
+7	0	1	1	1
+6	0	1	1	0
+5	0	1	0	1
+4	0	1	0	0
+3	0	0	1	1
+2	0	0	1	0
+1	0	0	0	1
0	0	0	0	0
-1	1	1	1	1
-2	1	1	1	0
-3	1	1	0	1

Entiers signés en binaire : résumé

- Exemple
- Codage de -57 pour les 3 méthodes, Sur 8 bits

- 57 = 111001
- Valeur absolue +57 = $(00111001)_2$
- Nombre signé -57 : 10111001
- Complément à 1 : CR(57) = 11000110
- Complément à 2 : -57 = CV(57) = 11000111

Dans tous les cas:

- Si bit de poids fort = 0 : entier positif
- Si bit de poids fort = 1 : entier négatif

Représentation octale des nombres relatifs

- Complément vrai ou complément à 8

  De la même façon, en notation octale, on peut représenter les nombres négatifs

  Pour A>0 : - A = CV(A) = CR(A)+1

  Où CR(A) le complément restreint de A, est obtenu en complémentant à 7 le nombre A exprimé en octale

  Exemple:

  Sur 4 digits, $A = (+5)_{10} = (0005)_8$

  $(-5)_{10} = (?)_8$

  $CR(A) = (7772)_8$

  $(-5)_{10} = CV(A) = CR(A)+(0001)_8 = (7773)_8$

Représentation hexadécimale des nombres relatifs

- Complément à 16

  De la même façon, en notation hexadécimale, on peut utiliser le complément vrai ou le complément à 16 pour représenter les nombres négatifs.

  Pour A>0, -A = CV(A) = CR(A)+1

  Où CR(A) est obtenu en complémentant à F le nombre A exprimé en hexadécimale

  Exemple:

  Sur 4 digits : +22 = $(0016)_{16}$

  CR(22) = $(FFE9)_{16}$

  -22 = CV(22) = CR(22)+$(0001)_{16}$ = $(FFFA)_{16}$

Problèmes liés à la longueur des nombres

Les circuits traitant des nombres de n bits, y compris le bit de signe, peuvent manipuler tous les nombres compris entre -2^{n-1} et $2^{n-1}-1$

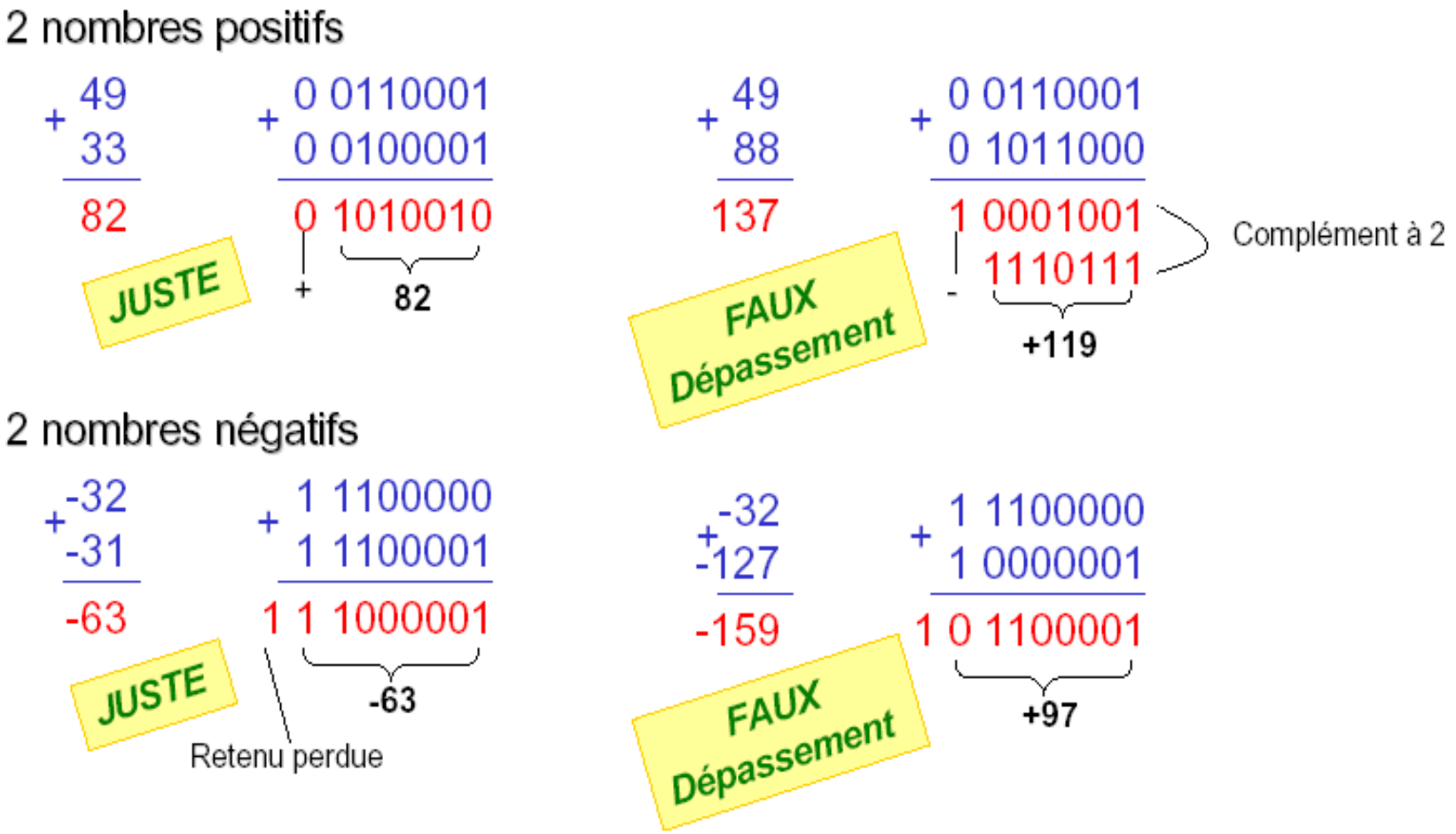
- Exemple :n = 8 ⇒ intervalle allant de – 128 à + 127
- Les résultats partiels ou définitifs d'opérations mathématiques ne doivent pas sortir de cet intervalle

- **Cas de l'addition:**

- Une retenue peut apparaître (CARRY)
- Un dépassement de capacité est aussi possible (OVERFLOW)
- L'Overflow est un dépassement de mémoire, car le nombre de bits en notre disposition est insuffisant pour coder l'information.

Problèmes liés à la longueur des nombres

- Addition sur 8 bits ⇒ intervalle allant de – 128 à + 127



Représentation des réels

- Nombres à virgule fixe

Soit N un nombre réel à Virgule fixe, alors

Inconvénients :

- étendue de représentation limitée
- 32 bits seulement pour la partie entière
- 32 bits seulement pour la partie fractionnaire
- Perte de précision pour les petits nombres

Représentation des réels

- Nombres à virgule flottante

$$N = (-1)^s \times 1M \times B^e$$

où : M = mantisse

B = base

e = exposant

s = signe de la mantisse

Exemples:

$(101)_{10} = 1.01 \times 10^2 = 0.101 \times 10^3 = \ldots$

$(-5)_{10} = (-101)_2 = -1.01 \times 2^2$

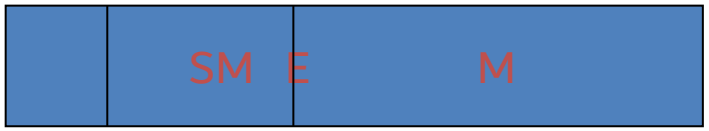
Représentation des réels

- Nombres à virgule flottante

Standard IEEE 754 (1985)

simple précision sur 32 bits :

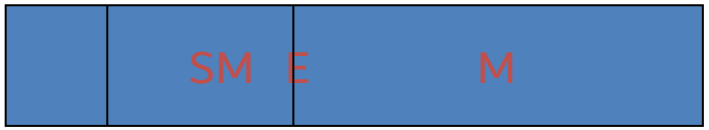
- **1 bit de signe de la mantisse**
- **8 bits pour l'exposant**
- **23 bits pour la mantisse**



1bit 8 bits 23 bits

double précision sur 64 bits :

- **1 bit de signe de la mantisse**
- **11 bits pour l'exposant**
- **52 bits pour la mantisse**



1bit 11 bits 52 bits

SM = 0 pour + , 1 pour -
E = codage de l'exposant par excédant càd e+127 ou e+1023
M = mantisse normalisée

En simple précision, $x = (-1)^{SM} * 2^{(E-127)} * 1.M$
En double précision, $x = (-1)^{SM} * 2^{(E-1023)} * 1.M$

Représentation des réels

- Nombres à virgule flottante

Norme IEEE 754 avec simple précision

Exemple :

$(1000)_{10} = (3E8)_{16} = (1111101000)_2 = 1.111101000 \times 2^9$

SM = 0 car nombre positif

e = 9 donc*E*= *e* + 127 = 136 = $(10001000)_2$

M = 111101000

0 10001000 11110100000000000000000

Exemple: Quel est le nombre A codé avec simple précision par:

A= 1 10000011 01000000000000000000000

A= $(-1)^1$ x $2^{131-127}$ x 1.01 = -1 x 2^4 x 1.25 = -20

Autre Représentation interne de l'information

- Données numériques
  - Le BCD
  - Le BCD est un code dans lequel chaque chiffre d'un nombre décimal est codé en binaire sur 4 bits.
  - Ces chiffres peuvent être représentés sur un octet individuel, c'est le BCD non compacté.
  - Exemple :

$(327)_{10} \rightarrow (00000011\ 00000010\ 00000111)_{BCD}$

Autre Représentation interne de l'information

- Codage des caractères

Le code ASCII (American Standard Code for Information Interchange) Code standard Américain pour l'échange d'information

C'est le système de codage universel. C'est un code à 7 positions, le huitième bit étant réservé à la parité, ce qui fait 2^7=128 caractères représentables. Ce code comprend :

- des fonctions de commandes (transmissions de données, tabulations, Retour chariot ...)
- des symboles de ponctuations
- quelques symboles usuels en informatique (@...)
- les chiffres
- les majuscules , les minuscules

Autre Représentation interne de l'information

- Codage des caractères

Table des caractères de contrôle (00 à 31)

ASCII	Caract.	Signification	ASCII	Caract.	Signification
00	NUL	*null*, nul	16	DLE	*data link escape*, échap. liaison données
01	SOH	*start of heading*, début d'en-tête	17	DC1	*device control 1*, commande unité 1
02	STX	*start of text*, début de texte	18	DC2	*device control 2*, commande unité 2
03	ETX	*end of text*, fin de texte	19	DC3	*device control 3*, commande unité 3
04	EOT	*end of transmission*, fin de transmission	20	DC4	*device control 4*, commande unité 4
05	ENQ	*enquiry*, interrogation	21	NAK	*negative acknowledge*, acc. récep. nég.
06	ACK	*acknowledge*, accusé de réception	22	SYN	*synchronous idle*, inactif synchronisé
07	BEL	*bell*, sonnerie	23	ETB	*end of transmission block*, fin tran. bloc
08	BS	*backspace*, espacement arrière	24	CAN	*cancel*, annuler
09	HT	*horizontal tabulation*, tabulation horiz.	25	EM	*end of medium*, fin du support
10	LF	*line feed*, saut de ligne	26	SUB	*substitute*, substitut
11	VT	*vertical tabulation*, tabulation verticale	27	ESC	*escape*, échappement
12	FF	*form feed*, saut de page	28	FS	*file separator*, séparateur de fichiers
13	CR	*carriage return*, retour chariot	29	GS	*group separator*, séparateur de groupes
14	SO	*shift out*, hors code	30	RS	*record separator*, sép. d'enregistr.
15	SI	*shift in*, en code	31	US	*unit separator*, séparateur d'unités

Autre Représentation interne de l'information

- Codage des caractères
  - Table des caractères imprimables

32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49
	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/	0	1

50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67
2	3	4	5	6	7	8	9	:	;	<	=	>	?	@	A	B	C

68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U

86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103
V	W	X	Y	Z	[	\	]	¨	_	`	a	b	c	d	e	f	g

104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121
h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y

122	123	124	125	126
z	{	|	}	~

Autre Représentation interne de l'information

- Codage des caractères

Exemple 1: Coder en ASCII le mot **INFORMATIQUE**

Caractère	Code ASCII	Code ASCII en binaire
I	73	01001001
N	78	01001110
F	70	01000110
O	79	01001111
R	82	01010010
M	77	01001101
A	65	01000001
T	84	01010100
I	73	01001001
Q	81	01010001
U	85	01010101
E	69	01000101

Le codage sur un octet du mot INFORMATIQUE est:

01001001 01001110 01000110 01001111 01010010 01001101 01000001 01010100 01010100 01001001 01010001 01010101 01000101

Autre Représentation interne de l'information

- Codage des caractères

Exemple 2

Donner le codage ASCII de la chaîne de caractères ``Bonjour Monsieur !'',

En décimal :

- 66 111 110 106 111 117 114 32 77 111 110 115 105 101 117 114 32 33

En hexadécimal

- 42 6F 6E 6A 6F 75 72 20 4D 6F 6E 73 69 65 75 72 20 21

En binaire (sur 4 Bits)

- 0100 0010 0110 1111 0110 1110 0110 1010 0110 1111 0111 0101 0111 0010 0010 0000 0100 1101 0110 1111 0110 1110 0111 0011 0110 1001 0110 0101 0111 0101 0111 0010 0010 0000 0010 0001.