



Numération - Codage

Chapitre II

Des « 1 » et des « 0 »

- 1949, « **BI**nary digiT »
- Bit est une **unité de mesure en informatique** désignant la quantité élémentaire d'information représentée par un chiffre binaire.
- Un bit ne peut prendre que deux valeurs : 0 ou 1. Selon le contexte, numérique, logique, ou électronique numérique, on les appelle « faux » et « vrai » ou « ouvert » et « fermé » ou « **niveau HAUT** » et « **niveau BAS** ».
- Le plus **petit paquet traitable** (ou adressable) est appelé **byte (octet)**.
- Lorsqu'un microprocesseur est conçu pour traiter simultanément plusieurs bytes, on appelle « **mot** » le **paquet de bytes**.
- Les tailles de mot les plus courantes sont de 8, 16, 32 et 64 bits. On parlera alors par exemple de « microprocesseur 64 bits ».

Des « 1 » et des « 0 »

- **Un octet peut prendre $2^8=256$ valeurs différentes, entre 00000000 et 11111111.**
 - ♦ 1 kilo-octet (ko ou Ko) = 2^{10} octets = 1 024 octets
→ 1 **kibioctet** (Kio)
 - ♦ 1 méga-octet (Mo) = 2^{20} octets = 1 024 ko
= 1 048 576 octets
→ 1 **mébioctet** (Mio)
 - ♦ 1 giga-octet (Go) = 2^{30} octets = 1 024 Mo
= 1 073 741 824 octets
→ 1 **gibioctet** (Gio)
 - ♦ 1 téra-octet (To) = 2^{40} octets = 1 024 Go
= 1 099 511 627 776 octets
→ 1 **tébioctet** (Tio)

Plan

- **Système de numération et changements de base**
 - ♦ Système décimal
 - ♦ Système binaire
 - ♦ Systèmes octal et hexadécimal
 - ♦ Changements de base
- **Les signes plus (+) et moins (–)**
 - ♦ Module et signe
 - ♦ Complément à 2
- **La virgule**
 - ♦ Virgule fixe
 - ♦ Virgule flottante
 - ♦ La norme IEEE 754
- **Codage**
 - ♦ Code DCB
 - ♦ Code 2 parmi 5
 - ♦ Code Gray
 - ♦ Code ascii

Systemes de numération

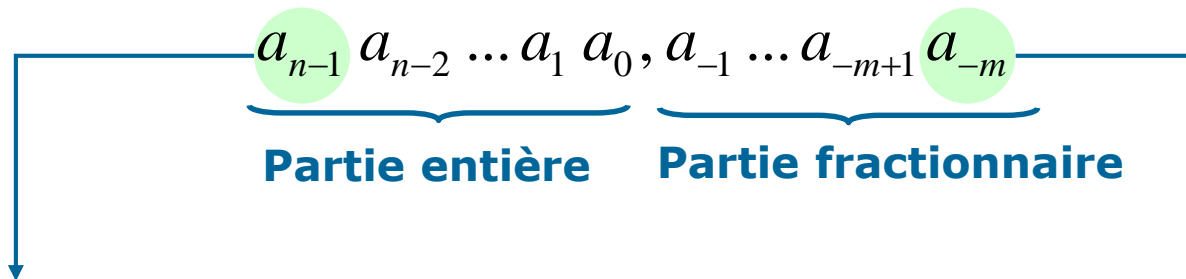
- **Utilisation de codes pondérés**
 - ♦ position (rang) du symbole (chiffre) dans le nombre détermine son poids
- **Principe de numération : Juxtaposition de symboles appelés chiffres**
- **Nombre de symboles = Base de numération**
 - ♦ Ex : Système décimal : $\{0, 1, 2, \dots, 9\} \rightarrow 10$ symboles

Base

- Soit une base b associée à b symboles $\{S_0, S_1, S_2, \dots, S_{b-1}\}$
- Un nombre positif N dans un système de base b s'écrit sous forme polynomiale :

$$N = \sum_{i=-m}^{n-1} a_i b^i \quad a_i \in \{S_0, S_1, S_2, \dots, S_{b-1}\}$$
$$= a_{n-1} \cdot b^{n-1} + a_{n-2} \cdot b^{n-2} + \dots + a_1 \cdot b^1 + a_0 \cdot b^0 + a_{-1} \cdot b^{-1} + \dots + a_{-m+1} \cdot b^{-m+1} + a_{-m} \cdot b^{-m}$$

- Représentation de position :



Chiffre le plus significatif
En binaire :
Most Significant Bit (MSB)

Chiffre le moins significatif
En binaire :
Least Significant Bit (LSB)

Décimal (Base 10)

- **Dix chiffres : 0, 1, 2, ..., 9**

$$N = \sum_{i=-m}^{n-1} a_i b^i = \sum_{i=-m}^{n-1} a_i \times 10^i$$

- **Exemple :**

$$\begin{aligned} 1978,265 &= \sum_{i=-3}^3 a_i \times 10^i \\ &= 1 \times 10^3 + 9 \times 10^2 + 7 \times 10^1 + 8 \times 10^0 + 2 \times 10^{-1} + 6 \times 10^{-2} + 5 \times 10^{-3} \end{aligned}$$

Binaire (Base 2)

- Deux chiffres : 0 et 1

$$N = \sum_{i=-m}^{n-1} a_i b^i = \sum_{i=-m}^{n-1} a_i \times 2^i$$

- Exemple :

$$(1110)_2 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = (14)_{10}$$

$$(1110,101)_2 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} = (14,625)_{10}$$

**Avec n bits on peut former 2^n nombres différents.
Le plus petit est 0. Le plus grand est $(2^n - 1)$.**

Exemple : Avec 8 bits $\rightarrow 2^n = 2^8 = 256$ nombres différents.

Le plus petit $\rightarrow (00000000)_2 = (0)_{10}$

Le plus grand $\rightarrow (11111111)_2 = (255)_{10}$

Octal (Base 8)

- **8 chiffres : 0, 1, 2, 3, 4, 5, 6, 7**
- **Exemple :**

$$(370)_8 = 3 \times 8^2 + 7 \times 8^1 + 0 \times 8^0 = 248$$

Hexadécimal (Base 16)

- **16 chiffres : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F**
- **Tableau de correspondance :**

Hex.	Décimal
A	10
B	11
C	12
D	13
E	14
F	15

- **Exemple :**

$$\begin{aligned}(AC53)_{16} &= A \times 16^3 + C \times 16^2 + 5 \times 16^1 + 3 \times 16^0 \\ &= 10 \times 16^3 + 12 \times 16^2 + 5 \times 16^1 + 3 \times 16^0 = (44115)_{10}\end{aligned}$$

Tableau

	Décimal	Binaire	Octal	Hexadécimal
Base <i>b</i>	10	2	8	16
	0	0000	00	0
	1	0001	01	1
	2	0010	02	2
	3	0011	03	3
	4	0100	04	4
	5	0101	05	5
	6	0110	06	6
	7	0111	07	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

Changement de base

- **De base b à base 10 : La somme du développement en polynôme du nombre dans la base b .**

$$N = \sum_{i=-m}^{n-1} a_i b^i$$

$$(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = (13)_{10}$$

$$(1A7)_{16} = 1 \times 16^2 + A \times 16^1 + 7 \times 16^0 = 16^2 + 10 \times 16 + 7 = (423)_{10}$$

$$(1101,10)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} = (13,625)_{10}$$

$$(43,2)_5 = 4 \times 5^1 + 3 \times 5^0 + 2 \times 5^{-1} = 20 + 3 + 0,4 = (23,4)_{10}$$

Changement de base

- **De base 10 à base b**

- ♦ Méthode 1 : Soustraction

- ♦ Exemple : $(363)_{10} = (?)_2$

Recherche de la puissance 2 juste supérieure : $2^9 = 512$

$$363 = 1 \times 2^8 + 107 \quad \longrightarrow \text{MSB}$$

$$107 = 0 \times 2^7 + 107$$

$$107 = 1 \times 2^6 + 43$$

$$43 = 1 \times 2^5 + 11$$

$$11 = 0 \times 2^4 + 11$$

$$11 = 1 \times 2^3 + 3$$

$$3 = 0 \times 2^2 + 3$$

$$3 = 1 \times 2^1 + 1$$

$$1 = 1 \times 2^0 + 0 \quad \longrightarrow \text{LSB}$$

$$(363)_{10} = (101101011)_2$$

Changement de base

- **De base 10 à base b**

- ♦ Méthode 2 : divisions successives par b

- ♦ Division par b :

$$N = \overbrace{q \cdot b}^{\text{quotient}} + r \longrightarrow \text{reste}$$

$$N = a_{n-1} \cdot b^{n-1} + a_{n-2} \cdot b^{n-2} + \dots + a_1 \cdot b^1 + a_0$$

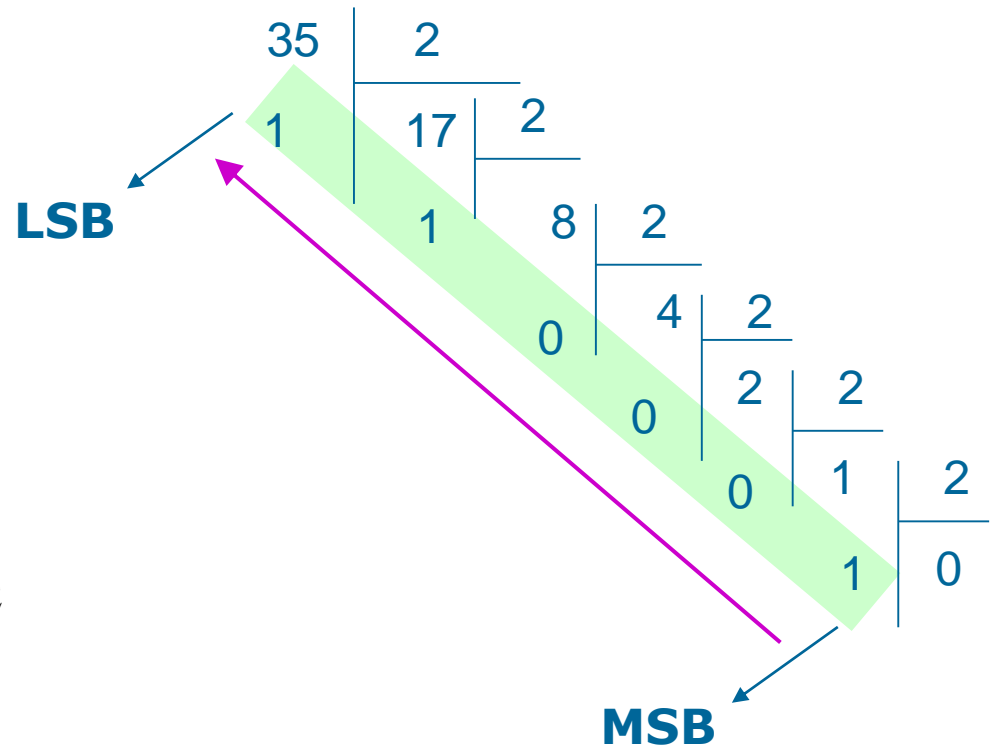
- | | | |
|---|---|---------------|
| ♦ 1 ^{ère} division par b : | $q = a_{n-1} \cdot b^{n-2} + a_{n-2} \cdot b^{n-3} + \dots + a_1$ | $r = a_0$ |
| ♦ 2 ^{ème} division par b : | $q = a_{n-1} \cdot b^{n-3} + a_{n-2} \cdot b^{n-4} + \dots + a_2$ | $r = a_1$ |
| ⋮ | ⋮ | ⋮ |
| ♦ (N-1) ^{ème} division par b : | $q = a_{n-1}$ | $r = a_{n-2}$ |
| ♦ N ^{ème} division par b : | $q = 0$ | $r = a_{n-1}$ |

Changement de base

- **De base 10 à base b**

- ♦ Méthode 2 : divisions successives par b

- ♦ Exemple 1 : $(35)_{10} = (?)_2$



$$(35)_{10} = (100011)_2$$

Changement de base

- ◆ Exemple 2 : $(363)_{10} = (?)_{16}$

363		16		
11		22		16
(B)		6		1
				16
				0


$(363)_{10} = (16B)_{16}$

Changement de base

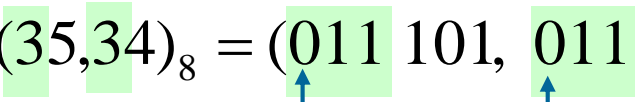
- **Conversions $b^p \leftrightarrow b^k$**
 - ♦ Conversion Binaire \leftrightarrow Octal : $2 \leftrightarrow 2^3$
 - ♦ Conversion Binaire \leftrightarrow Hexadécimal : $2 \leftrightarrow 2^4$
- **Dans le cas général, on passe d'abord par la base b .**

Conversion : octal \rightarrow binaire

- Octal \rightarrow Binaire ($2^3 \rightarrow 2$)
- 8 symboles en octal : 0, 1, 2, 3, ..., 7
- Chaque symbole en octal s'écrit sur 3 bits en binaire.
- Éclatement en 3 bits
- Exemples :


$$(345)_8 = (011\ 100\ 101)_2$$

$$(65,76)_8 = (110\ 101,\ 111\ 110)_2$$


$$(35,34)_8 = (011\ 101,\ 011\ 100)_2$$

Ce 0 peut être supprimé.

Ce 0 ne peut pas être supprimé.

Octal	Binaire
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Conversion : binaire \rightarrow octal

- **Binaire \rightarrow Octal ($2 \rightarrow 2^3$)**
- **Chaque symbole en octal s'écrit sur 3 bits en binaire.**
- **Regroupement de 3 bits, à partir du poids faible**
- **Exemple :**

$$(11001010010110)_2 = (\overset{\overleftarrow{3}}{011} \overset{\overleftarrow{3}}{001} \overset{\overleftarrow{3}}{010} \overset{\overleftarrow{3}}{010} \overset{\overleftarrow{3}}{110})_2 = (31226)_8$$

$$(110010100,10101)_2 = (\overset{\overleftarrow{3}}{110} \overset{\overleftarrow{3}}{010} \overset{\overleftarrow{3}}{100}, \overset{\overrightarrow{3}}{101} \overset{\overrightarrow{3}}{010})_2 = (624,52)_8$$

Remarque : Le regroupement se fait de droite à gauche pour la partie entière et de gauche à droite pour la partie fractionnaire .

Conversion : hexadécimal \rightarrow binaire

- **Binaire \rightarrow Hexadécimal ($2 \rightarrow 2^4$)**
- **16 symboles en hexadécimal : 0, 1, ..., 9 ,A, B, C, D, E, F**
- **Chaque symbole en hexadécimal s'écrit sur 4 bits en binaire.**
- **Éclatement en 4 bits**
- **Exemples :**

$(AB3,4F6)_{16}$

$= (1010\ 1011\ 0011, 0100\ 1111\ 0110)_2$

Hex.	Binaire
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Conversion : binaire \rightarrow hexadécimal

- Hexadécimal \rightarrow Binaire ($2^4 \rightarrow 2$)
- Chaque symbole en octal s'écrit sur **4 bits en binaire**.
- Regroupement de 4 bits, à partir du poids faible
- Exemple :

$$(11001010010101)_2 = (\overset{4}{\leftarrow} 0001 \overset{4}{\leftarrow} 1001 \overset{4}{\leftarrow} 0100, \overset{4}{\rightarrow} 1010 \overset{4}{\rightarrow} 1000)_2 = (194, A8)_{16}$$

Changement de base – Partie fractionnaire

- **Décomposition en puissances négatives de b**

$$N = a_1 \cdot b^{-1} + a_2 \cdot b^{-2} + a_3 \cdot b^{-3} + \dots + a_{n-1} \cdot b^{-(n-1)} + a_n \cdot b^{-n}$$

- ♦ 1^{ère} division par b^{-1} (multiplication par b) :

$$\frac{a_1 \cdot b^{-1} + a_2 \cdot b^{-2} + a_3 \cdot b^{-3} \dots + a_n \cdot b^{-n}}{b^{-1}} = a_1 + a_2 \cdot b^{-1} + a_3 \cdot b^{-2} \dots + a_n \cdot b^{-n+1}$$

- ♦ 2^e division par b^{-1} (multiplication par b) :

$$\frac{a_2 \cdot b^{-1} + a_3 \cdot b^{-2} \dots + a_n \cdot b^{-n+1}}{b^{-1}} = a_2 + a_3 \cdot b^{-1} + \dots + a_n \cdot b^{-n+2}$$

- ♦ 3^e division par b^{-1} (multiplication par b) :

$$\frac{a_3 \cdot b^{-1} + \dots + a_n \cdot b^{-n+2}}{b^{-1}} = a_3 + \dots + a_n \cdot b^{-n+3}$$

⋮

- ♦ n^e division par b^{-1} (multiplication par b) :

$$\frac{a_n \cdot b^{-1}}{b^{-1}} = a_n$$

Changement de base – Partie fractionnaire

- **Exemple : $(0,25)_{10} = (?)_2$**

- ♦ Partie entière : $(0)_{10} = (0)_2$
- ♦ Partie fractionnaire : $(0,25)_{10}$

$$(0,25)_{10} = (0,01)_2$$

$$0,25 \times 2 = 0,5$$

$$0,5 \times 2 = 1,0$$

- **Exemple : $(27,3)_{10} = (?)_2$**

- ♦ Partie entière : $(27)_{10} = (11011)_2$
- ♦ Partie fractionnaire : $(0,3)_{10}$

$$0,3 \times 2 = 0,6$$

$$0,6 \times 2 = 1,2$$

$$0,2 \times 2 = 0,4$$

$$0,4 \times 2 = 0,8$$

$$0,8 \times 2 = 1,6$$

$$(27,3)_{10} = (11011,0 \text{ } 1001 \text{ } 1001 \text{ } 1001\dots)_2$$

$$0,6 \times 2 = 1,2 \quad \downarrow \text{Cyclique}$$

On s'arrête à $2^{-2} \rightarrow 11011,01 = 27,25$

On s'arrête à $2^{-5} \rightarrow 11011,01001 = 27,28125$

On s'arrête à $2^{-6} \rightarrow 11011,010011 = 27,296875$

On s'arrête à $2^{-9} \rightarrow 11011,010011001 = 27,298828125$

erreur= 0,05

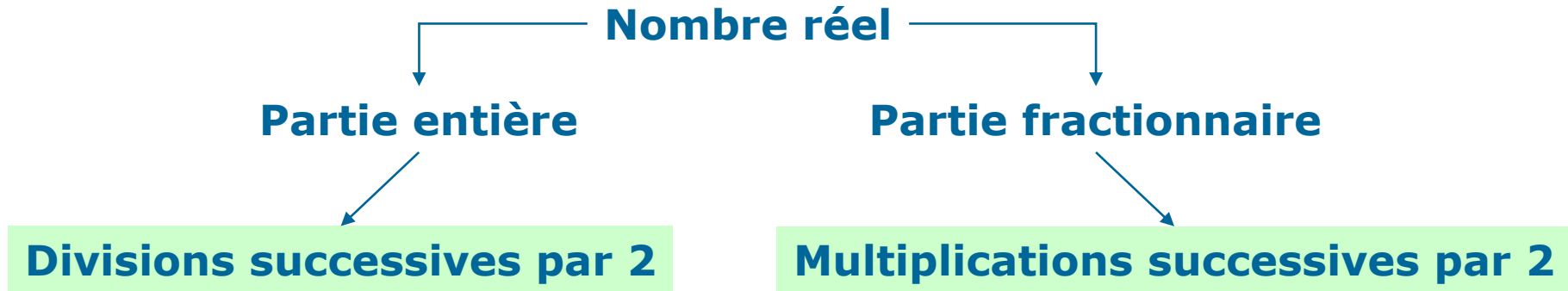
erreur= 0,01875

erreur= 0,003125

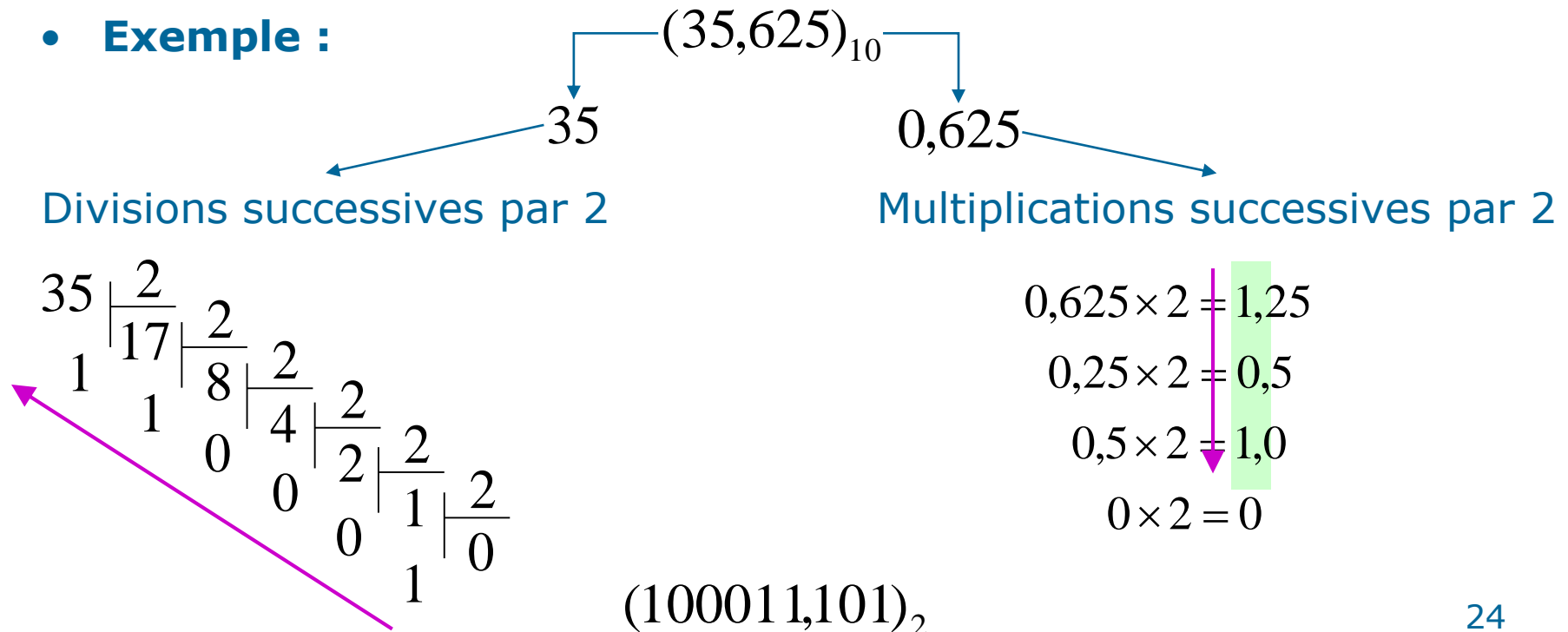
erreur= 0,001171875

...

Conversion d'un nombre réel : décimal → binaire



- Exemple :



Les signes plus (+) et moins (–)

- **Format** **nombre de bits utilisés**
- **Convention** **protocole de codage**
- **Dynamique** **différence entre le max et le min**
- **Résolution** **différence entre deux niveaux consécutifs**

- **Exemple :**
 - ♦ format 8 bits
 - ♦ convention entiers positifs
 - ♦ dynamique $2^8 - 1$
 - ♦ résolution 1 (constante sur la dynamique)
 - ♦ $(255)_{10} = (1111\ 1111)_2$ $(7)_{10} = (0000\ 0111)_2$

Les signes plus (+) et moins (–)

- Il existe 3 méthodes pour représenter les nombres négatifs :

- ♦ Signe et valeur absolue

- ▶ Un bit de signe (0 pour + et 1 pour –)
- ▶ Bits de norme
- ▶ Exemple sur 8 bits: $(47)_{10} = (00101111)_2$
 $(-47)_{10} = (10101111)_2$

- ♦ Complément à 1 (complément restreint)

- ▶ Remplacer les 0 par les 1 et les 1 par les 0
- ▶ Exemple : $(-47)_{10} = (11010000)_2$

- ♦ Complément à 2 (complément à vrai)

- ▶ Ajouter 1 au complément à 1
- ▶ Exemple : $(-47)_{10}$
 - Complément à 1 de $(00101111)_2 = (11010000)_2$
 - Ajouter 1 au $(11010000)_2 = (11010001)_2$

Les signes plus (+) et moins (−)

- **Complément vrai (complément à b)**

$$\text{complément} \longleftarrow \tilde{N} = \overset{\text{base}}{b^n} - \overset{\text{nombre}}{N} \quad \text{bits}$$

- ♦ Avec n bits, le « 1 » déborde.
Le nombre b^n est interprété comme 0.

- ♦ Nous avons donc : $\tilde{N} = 0 - N = -N$

$$b^n = 1 \overbrace{00\dots0}^{n \text{ bits}} \begin{matrix} \swarrow & \searrow & \searrow \\ b^n & b^{n-1} & b^0 \end{matrix}$$

- **Complément vrai en base 2 (complément à 2):**

$$\begin{cases} \tilde{N} = 2^n - N = (2^n - 1) - N + 1 \\ \tilde{N} = -N \end{cases}$$

Les « 1 » sont remplacés par les « 0 » et réciproquement = Complément à 1

Le nombre le plus grand avec n bits
= Tous les bits sont à « 1 »

Les signes plus (+) et moins (−)

- **Complément à 2 = Complément à 1 + 1 (LSB)**

- **Rappel : addition de 2 bits**

Addition binaire	Retenue	Somme
0 + 0 = 0	0	0
0 + 1 = 1	0	1
1 + 0 = 1	0	1
1 + 1 = 10	1	0

- **Exemple : $(-12)_{10}$ sur 8 bits**

$$(12)_{10} = (00001100)_2$$

$$11110011 \leftarrow \text{Complément à 1 de 12}$$

$$+ \quad 1$$

$$\hline 11110100 \leftarrow \text{Complément à 2 de 12}$$

$$(-12)_{10} = (11110100)_2$$

♦ Contrôle

$$11110100 \xrightarrow{\text{Complément à 1}} 00001011 \xrightarrow{\text{Complément à 2}} 00001100$$

$$(-(-12))_{10} = (12)_{10} = (00001100)_2$$

Les signes plus (+) et moins (–)

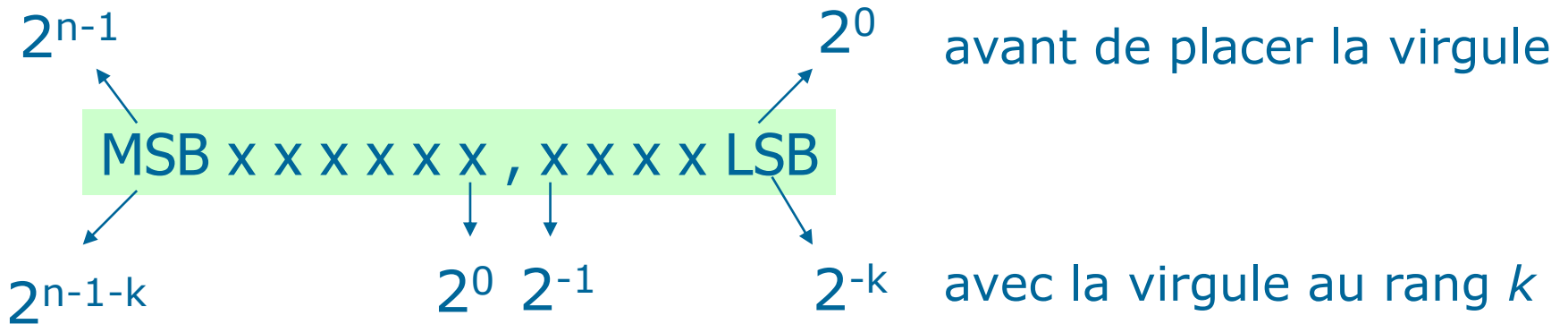
- Sur 4 bits

Décimal signé	Binaire Module + signe	Binaire Complément à 2
8	—	—
7	0111	0111
6	0110	0110
5	0101	0101
4	0100	0100
3	0011	0011
2	0010	0010
1	0001	0001
0	0000 ou 1000	0000
-1	1001	1111
-2	1010	1110
-3	1011	1101
-4	1100	1100
-5	1101	1011
-6	1110	1010
-7	1111	1001
-8	—	1000
	De -7 à +7	De -8 à +7

- Dynamique sur n bits ?

Virgule fixe

- Par convention, on place la virgule quelque part et on interprète.
- Binaire : n bits



- ♦ Résolution :
différence entre deux valeurs consécutives = 2^{-k}
- ♦ Dynamique :
différence entre les valeurs minimale et maximale = 2^{n-1-k}

Virgule fixe

- **Bon format pour l'addition :**

$$\begin{array}{r} 12,3 \\ + 23,4 \\ \hline 35,7 \end{array}$$

→ La virgule reste fixe.

- **Mauvais format pour la multiplication :**

$$\begin{array}{r} 13,4 \\ \times 10,1 \\ \hline 135,34 \end{array}$$

→ La virgule change de place.

Dépassement à gauche (débordement)
Dépassement à droite (arrondi)

Virgule flottante

$$N = M \times b^E$$

Diagram illustrating the components of a floating-point number representation:

- M is labeled **mantisse** (mantissa).
- b is labeled **base**.
- E is labeled **exposant** (exponent).

- On stocke la chaine de bits **ME** dans le calculateur.
- Exemple : Codage de Pi sur 5 chiffres de mantisse et 2 chiffres d'exposant en base 10

$0,3141 \times 10^1$

$0,0003 \times 10^4$

L'exposant augmente → La précision diminue

Précision maximale → Virgule à l'extrême gauche de la mantisse

- La représentation de la virgule flottante est **normalisée**, quand la virgule est à l'extrême gauche de la mantisse.

Virgule flottante

- **En binaire :**

$$N = M \times b^E$$

Diagram illustrating the components of the floating-point representation formula $N = M \times b^E$:

- M (Mantissa) is labeled "n bits" with an arrow pointing to it.
- b (Base) is labeled "2" with an arrow pointing to it.
- E (Exponent) is labeled "p bits" with an arrow pointing to it.

- **Dynamique : $N_{\max} - N_{\min}$**

La représentation en virgule flottante utilise la plus grande dynamique.

$$N_{\max} = 0,\underbrace{11\dots1}_{n \text{ fois}} \times \overbrace{2^{11\dots1}}^{p \text{ fois}} = 2^{2^p-1}$$
$$N_{\min} = 0,100\dots0 \times 2^{-11\dots1} = 2^{-2^p}$$

- **Résolution :** $0,00\dots1 \times 2^E = 2^{-n+E}$

La norme IEEE 754

- **La représentation en virgule flottante a été normalisée par IEEE 754.**
- **Taille de mots :**
 - ♦ 32 bits (simple précision)
 - ♦ 64 bits (double précision)
- **Le but de la norme IEEE 754 :**
 - ♦ Garantir un comportement identique d'une machine à l'autre
 - ♦ Définir le comportement en cas de dépassement de capacité (exceptions)
 - ♦ Garantir un arrondi exact pour les opérations élémentaires ($+$, $-$, \times , \div , $\sqrt{}$)

La norme IEEE 754

- Simple précision (sur 32 bits)

Signe (s)	Exposant (e)	Mantisse (m)
-----------	--------------	--------------

Nombre de bits → 1 8 23

$$N = (-1)^s \times 2^{e-127} \times \left(1 + \sum_{i=1}^{23} m_i 2^{-i} \right)$$

$$E = e - 127$$

$$E_{\min} < E < E_{\max} \quad E_{\min} = -126 \quad E_{\max} = 127$$

- Exemple :

1 10000010 001100000000000000000000

- ♦ $s=1 \rightarrow$ nombre négatif
- ♦ $e=1 \times 2^1 + 1 \times 2^7 = 130 \rightarrow e-127 = 130 - 127 = 3$
- ♦ $m=1 + (1 \times 2^{-3} + 1 \times 2^{-4}) = 1,1875$
- ♦ **$N = -2^3 \times 1,1875 = -9,5$**

La norme IEEE 754

- Le bit de signe est « 1 » pour négatif et « 0 » pour positif.
- La mantisse vaut toujours 1,xxxx et on ne stocke que xxxx
- L'exposant est en excédent 127
- La valeur 0 correspond à des 0 partout (en fait 2^{-127})
- Exemple :

0 01111111 00000000000000000000000000000000

- ♦ $s=0 \rightarrow$ nombre positif
- ♦ $e=1 \times 2^0 + 1 \times 2^1 + \dots + 1 \times 2^6 = 127 \rightarrow e-127 = 127 - 127 = 0$
- ♦ $m=1 + 0 = 1$
- ♦ $N=2^0 \times 1 = 1$

La norme IEEE 754

- **Convertisseur :**

- ♦ http://ajdesigner.com/fl_ieee_754_word/ieee_32_bit_word.php

- **Exemple : N=-5**

- ♦ Méthode 1

$$\begin{array}{lcl} s = 1 & & \\ 5 = 2^2(1 + M) & \begin{array}{l} \nearrow \\ \searrow \end{array} & \begin{array}{l} M = \frac{1}{4} = 2^{-2} \longrightarrow 010000000000000000000000 \\ e - 127 = 2 \longrightarrow e = 129 \longrightarrow 10000001 \end{array} \end{array}$$

- ♦ Méthode 2

- ▶ $s=1$ (car $N < 0$)
 - ▶ Conversion binaire de $|N| = (101)_2$
 - ▶ Représentation au format : $1,01 \times 2^{+2}$
 - ▶ $M = 010\ 0000\ 0000\ 0000\ 0000$
 - ▶ $e - 127 = 2 \longrightarrow e = 129 \longrightarrow 10000001$

$$N = S \mid E \mid M$$

$$N = 1 \mid 10000001 \mid 010\ 0000\ 0000\ 0000\ 0000$$

Codage

- **Le codage est une opération qui établit une correspondance entre les éléments de deux ensembles**
- **Choix de code → Application**
(Codes détecteurs, codes correcteurs, ...)
- **Codes pondérés : La position de chaque symbole dans chaque mot est affectée d'un poids fixe.**
 - ♦ Binaire et dérivés
 - ▶ Les poids fixes de droite à gauche : 1, 2, 4, 8, ...
 - ♦ DCB (Décimal Codé Binaire) ou BCD (Binary Coded Decimal) :
Chaque chiffre d'un nombre décimal (de 0 à 9) est codé à l'aide de 4 bits (de 0000 à 1001)
 - ▶ Les poids fixes de gauche à droite : 8, 4, 2, 1
 - ▶ Le code DCB n'utilise que 10 mots de codes de 4 bits

Code pondéré – DCB

Décimal	DCB
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	0001 0000
11	0001 0001
12	0001 0010
13	0001 0011
14	0001 0100
15	0001 0101

- L'apparition des combinaisons suivantes signifie qu'une erreur s'est produite :

Décimal	Binaire
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

- Exemple : $(1995)_{10} = (?)_{DCB}$
 $(0001\ 1001\ 1001\ 0101)_{DCB}$

Codes non pondérés – Code de Gray

- Binaire réfléchi (Gray) : Seul un bit change entre deux nombres consécutifs.**

0
1

Sur 1 bit

0	0	0
1	0	1
2	1	1
3	1	0

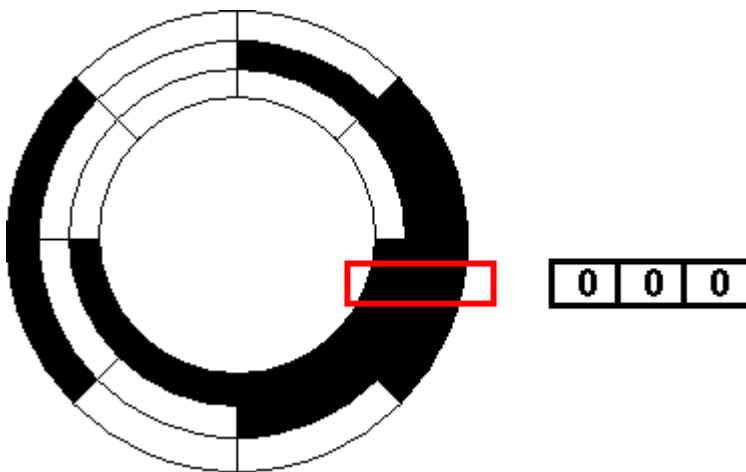
Sur 2 bits

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

Sur 3 bits

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

Sur 4 bits



- Application : Télécom, codeur rotatif, tableau de Karnaugh, etc.**

Codes non pondérés – p parmi n

- **Chaque chiffre est codé sur n bits dont uniquement p sont « 1 ».**
 - ♦ Le nombre de combinaisons de p éléments sélectionnés dans un ensemble de n éléments est donné par :

$$C_n^p = \frac{n!}{p!(n-p)!}$$

- **Exemple : Code 2 parmi 5**

- ♦ 5 bits dont 2 sont « 1 »
- ♦ Positions des «1 » :
 - ▶ 01236
 - ▶ 74210

Dans 01236 : 5=2+3

On prend les positions 2 et 3 pour les «1 »

Dans 74210 : 5 = 4+1

On prend les positions 4 et 1 pour les «1 »

Chiffre	01236	74210
0	01100	11000
1	11000	00011
2	10100	00101
3	10010	00110
4	01010	01001
5	00110	01010
6	10001	01100
7	01001	10001
8	00101	10010
9	00011	10100

- **Application : Codes-barres**

Code non pondéré – Code ASCII

- Le code ASCII restreint représente 128 caractères sur 7 bits, le 8^e bit étant un bit de parité.

$2^3=8 \rightarrow 3 \text{ bits}$

$2^4=16$
↓
4 bits

MSB \ LSB	0	1	2	...	7
0					
1					
2					r
⋮					
E					
F			/		

Le code ASCII de « / » est 2F \rightarrow 010 1111 \rightarrow parité paire 1010 1111
Le code ASCII de « r » est 72 \rightarrow 111 0010 \rightarrow parité paire 0111 0010