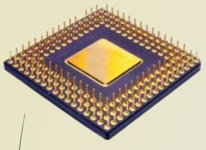


2023 – 2024

JFA - 1

Architecture de l'Ordinateur

DUT Informatique – Semestre 1

Ressource R 1.03

Responsable : Jean-François ANNE

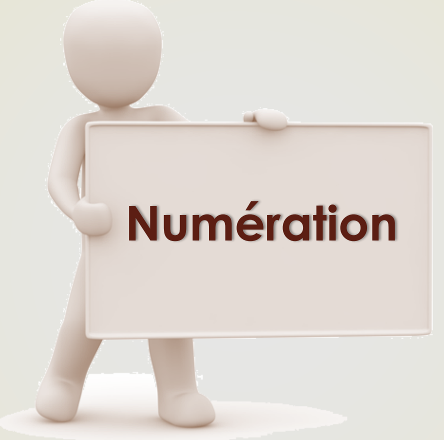
jean-francois.anne@unicaen.fr

http://www.jfanne.fr

31/08/2023

JFA - 2

Sommaire



JFA - 3

DUT Informatique – Semestre 1

Ressource R 1.03

Responsable : Jean-François ANNE

31/08/2023

JFA - 4

Numération

La numération permet de représenter un nombre (mot), par la juxtaposition ordonnée de symboles pris parmi un ensemble. Connaître la numération revient à utiliser le mécanisme qui permet de passer d'un nombre à une autre.

- Les systèmes de numération les plus courants en informatique sont :
  - Le système ***Décimal*** comprend 10 symboles appelés chiffres : {0,1,2,3,4,5,6,7,8,9};
  - Le système ***Binaire*** comprend 2 symboles appelés bit (Binary Digit) : {0,1}
  - Le système ***Octal*** comprend 8 symboles appelés chiffres : {0,1,2,3,4,5,6,7};
  - Le système ***Hexadécimal*** comprend 16 symboles : {0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F};
- Le ***nombre de symboles*** que possède le système de numération est appelé ***Base***.
- Lorsqu'un nombre est écrit, la position respective des symboles détermine leur poids.
- Ces systèmes sont dits à poids positionnels : c'est-à-dire que la valeur d'un chiffre dépend de sa position (appelé ***rang***) dans le nombre.

JFA - 5

Vocabulaire

- ***Mots Binaires (groupes de bits) :***
  - Un bit est la plus petite unité d'information, et est noté **b**
  - Un mot de **4 bits** s'appelle **un Quartet ou semi octet (**Nibble ou Nybble**) ou hexit (**hex digit**) ou tetrade**.
  - Un mot de **8 bits** s'appelle **un Octet** (Byte), noté **o** ou **B**.

    1 B = 1 o = 8 b

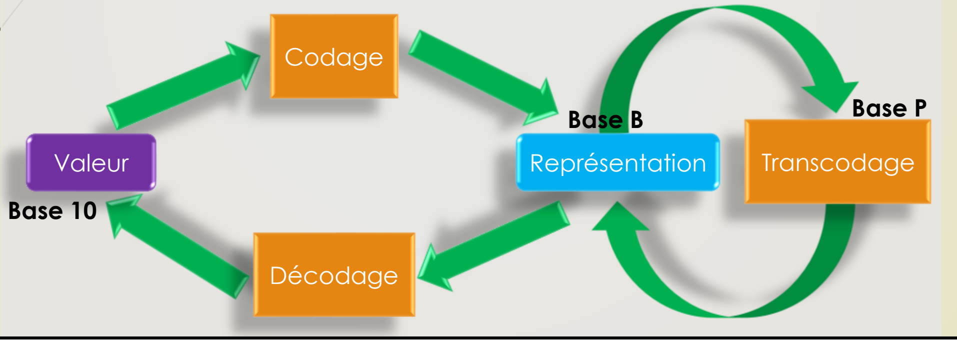
    1 ko = 8000 b = 1kB
  - Un mot de **16 bits** s'appelle un **Mot** (Word).
  - Un mot de **32 bits** s'appelle un **Double Mot** (Dword).
  - Un mot de **64 bits** s'appelle un **Quadruple Mot** (Qword).

- ***Poids Fort, Poids Faible :***
  - Le chiffre (Bit) le plus à gauche est dit : chiffre de poids fort. En Binaire, il est appelé M.S.B. (**Most Significant Bit**).
  - Le chiffre (Bit) le plus à droite est dit : chiffre de poids faible. En Binaire, il est appelé L.S.B. (**Least Significant Bit**).

JFA - 6

Vocabulaire 2

- ***Codage, Décodage, Transcodage :***
  - Le **codage** consiste à convertir un nombre écrit en base 10 (décimal) dans une base B quelconque.
  - Le **décodage** consiste à convertir un nombre écrit dans une base B vers la base 10.
  - Le **transcodage** consiste à convertir un nombre écrit dans une base B dans une autre base P (différente de 10).



JFA - 7

Multiples IEC : (Depuis 1998 !)

- ***Depuis Décembre 1998, la commission Internationale Electrotechnique (IEC) a modifié les préfixes et symboles des multiples binaires des données :***
  - Un mot de 1024 bits (2^{10} bits) s'appelle **un kibi bit (kilobinary) (Kibit)**.
    1 kibibit = 1 Kibit = 2^{10} bits = **1024 bits**
  - Un mot de 1024 Kibits (2^{20} bits) s'appelle **un mebi bit (megabinary) (Mibit)**.
    1 mebibit = 1 Mibit = 2^{20} bits = **1024 Kibits**
  - Un mot de 1024 Mibits (2^{30} bits) s'appelle **un gibi bit (gigabinary) (Gibit)**.
    1 gibibit = 1 Gibit = 2^{30} bits = **1024 Mibits**
  - Un mot de 1024 Gibits (2^{40} bits) s'appelle **un tebi bit (terabinary) (Tibit)**.
    1 tebibit = 1 Tibit = 2^{40} bits = **1024 Gibits**

  - On les utilise pour les bits, octets, bytes, words, Dwords, … :
    1 kibibit = 1 024 bits = 128 octets
    1 mébibit = 1 048 576 bits = 131 072 octets = 128 kibioctets
    100 mébibits = 104 857 600 bits = 12,5 mébioctets

JFA - 8

Multiples SI (Système International) :

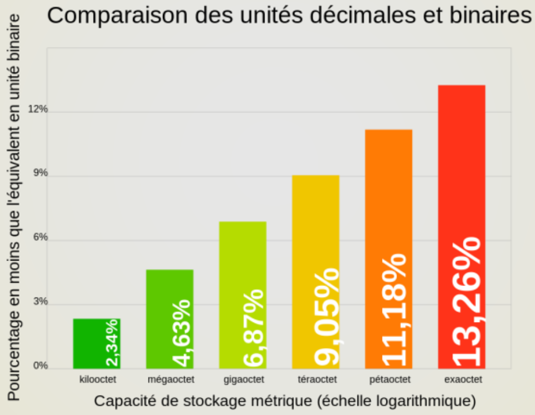
- ***Depuis Décembre 1998, la commission Internationale Electrotechnique (IEC) a modifié les préfixes et symboles des multiples binaires dans le traitement et la transmission des données :***
  - Un mot de 1000 bits (10^3 bits) s'appelle **un kilobit (kbit)**.
    1 kilobit = 1 kbit = 10^3 bits = **1000 bits**
  - Un mot de 1000 kbits (10^6 bits) s'appelle **un megabit (Mbit)**.
    1 megabit = 1 Mbit = 10^6 bits = **1000 kilobits**
  - Un mot de 1000 Mbits (10^9 bits) s'appelle **un gigabit (Gbit)**.
    1 gigabit = 1 Gbit = 10^9 bits = **1000 Megabits**
  - Un mot de 1000 Gbits (10^{12} bits) s'appelle **un terabit (Tbit)**.
    1 terabit = 1 Tbit = 10^{12} bits = **1000 Gigabits**

  - On les utilise pour les bits, octets, bytes, words, Dwords, … :
    1 kilobit = 1 000 bits = 125 octets

JFA - 9

Comparaisons des unités IEC et SI :

- ***Le graphique ci-dessous montre la différence entre les différentes unités en octets. Il montre le rapport entre l'interprétation des dimensions en représentation binaire par rapport à la représentation métrique.***



http://data.abuledu.org/wp/?LOM=6089

- ***Voir article de Wikipédia :***
  - https://fr.wikipedia.org/wiki/Pr%C3%A9fixe_binaire

JFA - 10

Généralisation

- On peut généraliser cette représentation en modifiant le nombre de symboles disponibles, c'est-à-dire en modifiant la base. Soit une base b, associée à b symboles : {S0, S1, ... S_{b-1}}. Un nombre N s'écrit alors avec la règle suivante :

$$N = (a_n a_{n-1} \ldots a_i \ldots a_0 , a_{-1} \ldots a_{-m})_b \text{ avec } a_i \in \{S_0, S_1, \ldots S_{b-1}\}$$

- Ce nombre s'écrit sous forme polynomiale :

$$N = a_n . b^n + a_{n-1} . b^{n-1} + \ldots + a_i . b^i + \ldots + a_0 . b^0 + a_{-1} . b^{-1} + \ldots + a_{-m} . b^{-m}$$

ou

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

avec :
a_i est le chiffre (digit) de rang i, son poids est b^i.
a_n est le chiffre le plus significatif (**M.S.B.**).
a_{-m} est le chiffre le moins significatif (**L.S.B.**).
a_n ... a_0 représente la partie entière.
a_{-1} ... a_{-m} représente la partie fractionnaire.

JFA - 11

La base 2 ou binaire

- C'est la base la plus utilisée en informatique et en électronique. On dispose de 2 symboles {0, 1} appelés **bits**.
  Le nombre N s'écrit alors :

$$N = (a_n a_{n-1} \ldots a_i \ldots a_0 , a_{-1} \ldots a_{-m})_2$$

- Ou sous sa forme polynomiale :

$$N = a_n.2^n + a_{n-1}.2^{n-1} + \ldots + a_i.2^i + \ldots + a_0.2^0 + a_{-1}.2^{-1} + \ldots + a_{-m}.2^{-m}$$

- ***Comptage :***

Décimal	0	1	2	3	4	5	6	7	8	9	10
Binaire	**0**	**1**	**10**	**11**	**100**	**101**	**110**	**111**	**1000**	**1001**	**1010**

- ***Exemple :***

$$(101)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = (5)_{10}$$

$$(1010{,}1)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} = (10{,}5)_{10}$$

JFA - 12

La base 16 ou Hexadécimal

- C'est la base la plus utilisée en programmation et en réseau. On dispose de 16 symboles {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}.
  Le nombre N s'écrit alors :

$$N = (a_n a_{n-1} \ldots a_i \ldots a_0 , a_{-1} \ldots a_{-m})_{16}$$

- Ou sous sa forme polynomiale :

$$N = a_n.16^n + a_{n-1}.16^{n-1} + \ldots + a_i.16^i + \ldots + a_0.16^0 + a_{-1}.16^{-1} + \ldots + a_{-m}.16^{-m}$$

- ***Comptage :***

Décimal	0	1	2	…	9	10	11	12	13	14	15	16
Hexadécimal	**0**	**1**	**2**	**…**	**9**	**A**	**B**	**C**	**D**	**E**	**F**	**10**

- ***Exemple :***

$$(AB8)_{16} = A \times 16^2 + B \times 16^1 + 8 \times 16^0$$
$$(AB8)_{16} = 10 \times 256 + 11 \times 16 + 8 \times 1$$
$$(AB8)_{16} = (2744)_{10}$$

$$(B48E, C)_{16} = B \times 16^3 + 4 \times 16^2 + 8 \times 16^1 + E \times 16^0 + C \times 16^{-1}$$
$$(B48E, C)_{16} = (46222{,}75)_{10}$$

JFA - 13

Conversion de bases :

- Pour manipuler les données, on va être obligé de faire des conversions de représentation de la valeur d'un nombre.

  On devra donc faire des conversions entre les différentes bases :
  - base 10,
  - base 2,
  - base 16,

JFA - 14

Conversion de la base B vers la base 10

- Pour convertir un nombre écrit dans une base B vers la base 10, on exploite directement la forme polynomiale, en effectuant les différentes opérations pour obtenir une représentation en base 10 :

$$N = a_n.b^n + a_{n-1}.b^{n-1} + \ldots + a_i.b^i + \ldots + a_0.b^0 + a_{-1}.b^{-1} + \ldots + a_{-m}.b^{-m}$$

- ***Méthode :***
  - Soit $N_2 = 1110$

    $N = 1.2^3 + 1.2^2 + 1.2^1 + 0.2^0$

    $N = 8 + 4 + 2 + 0$

    $N_{10} = 14$
  - Soit, $10110,11_2$ à convertir en base 10 :

    $(10110{,}11)_2 = 1\times 2^4 + 0\times 2^3 + 1\times 2^2 + 1\times 2^1 + 0\times 2^0 + 1\times 2^{-1} + 1\times 2^{-2}$

    $(10110{,}11)_2 = 22{,}75_{10}$

- ***Exercices :***

  $(722FA)_{16} = 467706_{10}$

  $(5845)_9 = 4334_{10}$

JFA - 15

Conversion de la base 10 vers la base B

- Pour convertir un nombre écrit dans une base 10 vers la base B, on va séparer le nombre en deux parties et appliquer un calcul différent pour chacune des parties :
  - La partie entière
  - La partie fractionnaire

- ***Pour la partie entière :***
  On effectue une division successive de la partie entière du nombre en base 10 par la base B jusqu'à 0. On ne garde que les **restes** de la division **en partant de la fin** vers le début.

- ***Exemple :*** $25{,}375_{10}$ en base 2 et $250{,}25390625_{10}$ en base 16

25 | 2
1 | 12 | 2
0 | 6 | 2
0 | 3 | 2
1 | 1 | 2
1 | 0

$25_{10} = 11001_2$

250 | 16
A 10 | 15 | 16
F 15 | 0

$250_{10} = FA_{16}$

JFA - 16

Conversion de la base 10 vers la base B

- ***Pour la partie fractionnaire :***
  On effectue une multiplication successive de la partie fractionnaire du nombre en base 10 par la base B jusqu'à 0. On ne garde que les **parties entières** dans l'ordre, puis on recommence avec la partie fractionnaire :

0,375 x 2 = 0,75 –> 0
0,75 x 2 = 1,5 –> 1
0,5 x 2 = 1 –> 1

$0{,}375_{10} = 0{,}011_2$

Donc :
$25{,}375_{10} = 11001{,}011_2$

0,25390625 x 16 = 4,0625 –> 4
0,0625 x 16 = 1 –> 1

$0{,}25390625_{10} = 0{,}41_{16}$

Donc :
$250{,}25390625_{10} = FA{,}41_{16}$

- ***Exercices :***

$23{,}9375_{10} = 10111{,}1111_2$

$31{,}75_{10} = 11111{,}11_2$

$23{,}9375_{10} = 17{,}F_{16}$

$31{,}75_{10} = 1F{,}C_{16}$

JFA - 17

Conversion de la base B vers la base B^k

- Pour passer d'un base B vers une puissance k de la base B soit B^k. On regroupe les chiffres de la base B à partie de la virgule par bloc de k chiffres, et on fait la conversion de chaque Bloc en base B^k.

- ***Passage de la base 2 à la base 8 = 2^3 :***

$$N = (010111110101{,}011010)_2$$
N = (010 111 110 101 , 011 010)2
N = (2 7 6 5 , 3 2)8
$$N = (2765{,}32)_8$$

- ***Passage de la base 2 à la base 16 = 2^4 :***

$$N = (010111110101{,}011010)_2$$
$N = (0101\ 1111\ 0101\ ,\ 0110\ 1000)_2$
$N = (5\ F\ 5\ ,\ 6\ 8)_{16}$
$$N = (5F5{,}68)_{16}$$

Exemples :

$$N = (111010110011011)_2 = (TCR)_{32}$$

JFA - 18

Conversion de la base B^k vers la base B

- Pour passer d'un base B^k vers la base B. On convertit chaque ***chiffre*** de la base B^k à partir de la virgule par bloc de k chiffres en base B.

- ***Passage de la base 8 (2^3) à la base 2 :***

$$N = (2765{,}32)_8$$
N = (2 7 6 5 , 3 2)8
N = (010 111 110 101 , 011 010)2
$$N = (010111110101{,}011010)_2$$

- ***Passage de la base 16 (2^4) à la base 2 :***

$$N = (5F5{,}68)_{16}$$
$N = (5\ F\ 5\ ,\ 6\ 8)_{16}$
$N = (0101\ 1111\ 0101\ ,\ 0110\ 1000)_2$
$$N = (010111110101{,}011010)_2$$

JFA - 19

Conversions diverses - 1

- Pour passer d'une base B^k vers une base B^p. On convertit le nombre d'abord en base B, puis en base B^p.
  - ***Passage de la base 8 (2^3) à la base 16 (2^4) :***

$(731,54)_8=(111011001,101100)_2$

$(111011001,101100)_2=(1D9,B0)_{16}$

- ***Exercices :***

$(7860)_9 = (7Q0)_{27}$

$(87543)_9 = (2PGC)_{27}$

$(ABEF)_{16} = (1AVF)_{32}$

JFA - 20

Conversions diverses - 2

- Pour passer d'un base B vers une base P. S'il n'y a aucune relation entre les deux bases, on convertit le nombre d'abord en base 10, puis en base P.
  - ***Passage de la base 5 à la base 17 :***

$2340_5 = ?_{17}$

$2340_5 = 345_{10} = 135_{17}$

- ***Exercices :***

$6B2G_{17} = 32707_{10} = 2021312_5$