I. PRINCIPLE OF A BASE

The base is the number used to define a number system.

- The base of the decimal system is ten “10”.
- The Binary Base is two "2".
- That of the Octal system is eight ‘’8’’.
- The base of Hexadecimmal is sixteen "16"

Whatever the numerical base used, it follows the following relationship:

$$\sum_{i=0}^{i=n}(b_i a^i) = b_i a^n + ... + b_5 a^5 + b_4 a^4 + b_3 a^3 + b_2 a^2 + b_1 a^1 + b_0 a^0$$

where : **b_i** : digit of the base of rank i
and : **ai** : power of base a with exponent of rank i
b_0 is called the least significant digit.
And b_i the most significant digit.

The notation ***( )$_b$*** indicates that the number is written in base b.

Exemple : base 10

1986 = (1 x 10^3) + (9 x 10^2) + (8 x 10^1) + (6 x 10^0)

I.1 THE DECIMAL SYSTEM

The decimal system is the one in which we are most accustomed to writing and using.

Each number can have ten ‘’10’’ different values :

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, therfore, the decimal system has the base 10.

Any number written in the decimal system satisfies the following rule

745 = 7 x 100 + 4 x 10 + 5 x 1

745 = 7 x 10 x 10 + 4 x 10 + 5 x 1

745 = 7 x 10^2 + 4 x 10^1 + 5 x 10^0

Each digit of the number, must be multiplied by a power of 10, this is what we call the weight of the digit.

The exponent of this power is zero for the digit located furthest to the right , and increases by one for each one-digit move to the left.

$12435 = 1 \times 10^4 + 2 \times 10^3 + 4 \times 10^2 + 3 \times 10^1 + 5 \times 10^0$.

This way of writing numbers is called the positional number system.

In our conventional system, we use powers of 10 to weight the value of numbers according to their position, However, it is possible to imagine other number systems based on a different integer.

I.2 THE BINARY SYSTEM

It is a numbering system using base two ‘’2’’. We commonly call bit in French(from the English binary digit, or “binary digit”, the digits of positional binary numbering. These can only take two values (0 and 1).

I.3 THE OCTAL SYSTEM

The octal number system is the base 8 number system, and uses the numbers 0 to 7.
Thus, a number expressed in base 8 could be presented as follows : $(745)_8$
When we write a number, we must clearly specify the base in which we express it to remove any possible indeterminations. (745 also exists in base 10).

$(745)_8$: Thus the number will be put in parentheses (745 in our example) and subscripted with a number representing its base (8 is subscripted)

I.4 THE HEXADECIMAL SYSTEM

It is a base 16 positional number system. It thus uses 16 symbols, he first ten digits (0,1,2,3,4,5,6,7,8,9) and the letters A to F for the following six:

10	**11**	**12**	**13**	**14**	**15**
A	**B**	**C**	**D**	**E**	**F**

The correspondence between base 2, base 10 and base 16 is indicated in the table below:

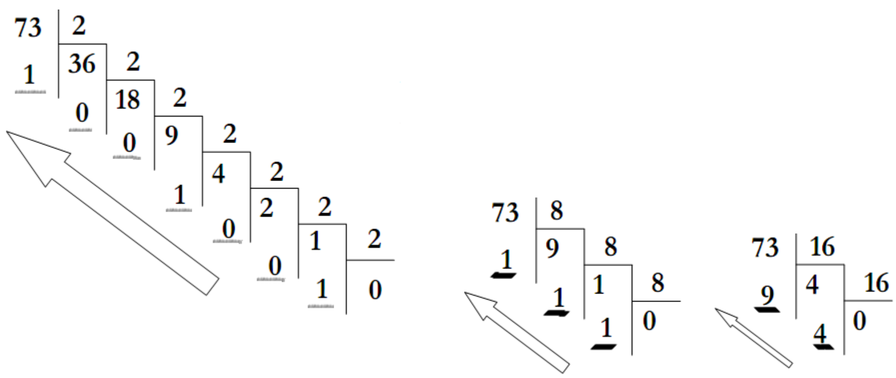
Base 10	Base 16	Base 2
0	0	0
1	1	1
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

II. CONVERSION FROM DECIMAL TO BASE B

The rule to follow is successive divisions :

- We divide the number by the base b
- Then the quotient by the base b
- So on until obtaining a quotient of zero or less than the base
- The sequence of remains corresponds to the symbols of the target base.
- We obtain the least significant figure first and the most significant figure last.

- *Exemple*



$(73)_{10}= (1001001)_2= (111)_8=(49)_{16}$

➢ Conversion from base to the decimal base

Let N be a number represented in binary base by : N = (1010011101)2

$N=(1*2^9) + (0*2^8) + (1*2^7) + (0*2^6) + (0*2^5) + (1*2^4) + (1*2^3) + (1*2^2) + (0*2^1) + (1*2^0)$

$= 512 + 0 + 128 + 0 + 0 + 16 + 8 + 4 + 0 + 1= (669)_{10}$

Let N be a number represented in Octal base by : $N = (251)_8$

$N = (2*8^2) + (5*8^1) + (1*8^0) = 128 + 40 + 1 = (169)_8$

Let N be a number represented in hexadecimal by : $N = (1C9)_{16}$

$N= (1*16^2) + (C*16^1) + (9*16^0) = (1*16^2) + (12*16^1) + (9*16^0) = 256 + 192 + 9 = (457)_{10}$

III. CODING FRACTIONAL NUMBERS

Coding a positif fractional number in base B:

$(N)_B = a_n a_{n-1} a_{n-2} \ldots a_1 a_0$

To code a positive fractional number, add fractional part after a comma :

$(N)_B = a_n a_{n-1} a_{n-2} \ldots a_1 a_0 , b_1 b_2 \ldots b_{m-1} b_m$

The decimal value of such a number is given by calculating

$a_n B^n + a_{n-1} B^{n-1} + \ldots a_1 B^n + a_0 + b_1 B^{-1} + b_2 B^{-2} + \ldots + b_{m-1} B^{m+1} + b_m B^{-m}$

Exemple

- $123,45 = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 + 4 \times 10^{-1} + 5 \times 10^{-2}$
- $(101,101)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1}$
  $+ 0 \times 2^{-2} + 1 \times 2^{-3}$
  $= 4 + 1 + 0,5 + 0,125 = 5,625$

III.1 Conversion of a real decimal number to base B

For the real part

- Use the integer division method as for integers.

For the fractional part

- Multiply the fractional part by B .
- Note the integer part obtained.
- Repeat this operation with the fractional part of the result and so on.
- Stop when the fractional part is zero.

- Or when the desired precision is achieved.
- Because we cannot always obtain a conversion to a finite number of digits for the fractional part.

➢ The fractional part in the base B is est la concatenation of the integer parts obtained in the order to their calculations.

Exemple: Conversion of $(12.6875)_{10}$ in binary base.

- Conversion de 12 : donne $(1100)_2$
- Conversion de 0,6875
  - ```
    0,6875 x 2 = 1,375 = 1 + 0,375
    0,375  x 2 = 0,75  = 0 + 0,75
    0,75   x 2 = 1,5   = 1 + 0,5
    0,5    x 2 = 1     = 1 + 0
    ```
  - $(12,6875)_{10} = (1100,1011)_2$

IV CONVERSION BETWEEN BASES.

In computer science, the binary, octal and hexadécimal bases are frequently used, all these bases being powers of 2, 2^1, 2^3, 2^4

There are some particulary simple conversions.

➢ ***Conversion Octal – Binary***

To convert a number from octal to binary, simply replace each digit constituting the octal number with its 3 digits binary equivalent.

Exemple

$(742)_8 = (111\ 100\ 010)_2$

$(1\ 6\ 7\ 3)_8 = (001\ 110\ 111\ 011)_2 = (1\ 110\ 111\ 011)_2$
    ```

➢ *Conversion Hexadecimal – Binary*

To convert a number from hexadecimal to binary, simply replace each digit constituting the hexadecimal number with its 4 digits binary equivalent.

Exemple

(A 9 2)$_{16}$ = (1010 1001 010)$_{2}$

(8 B 7 F)$_{16}$ = (1000 1011 0111 1111)$_{2}$

➢ *Conversion binary - Octal*

To convert a binary number to Octal, simply make groups of three digits (starting from the right).

Exemple

Binary	1110010	11001100100	111100110010101011100101
Grouping	**00**1/110/010	**0**11/001/100/100	111/100/110/010/101/011/100/101
Pseudo Octal	1 / 6 / 2	3 / 1 / 4 /4	6 / 4 / 6 /2/ 3 / 4 /5
Octal	162	3144	7462345

➢ *Conversion hexadecimal - binary*

To convert a binary number to hexadecimal, simply make groups of four digitss (starting from the right).

Exemple

Binary	110010	101001100100	110110110110101011100101
Grouping	**00**11 / 0010	1010/0110/0100	/1101/1011/0110/1010/1110/0101
Pseudo hexadecimal	3 / 2	10 /6 /4	13/11/6/10/14/5
Hexadecimal	32	A64	DB6AE5

IV. ARITHMETICS OPERATIONS IN BASE B

- Arithmetics operations (addition, subtraction, multiplication, division) are carried out in any base B with the same rules in Base 10.
- A deduction or carryover appears when we reach or exceedthe valu B of the Base.

Arithmetic operations (addition, subtraction, multiplication, division) can be performed in any base B.

- With the same rules as for the base 10.
- Also detention (hold) but depending on the base.
- When we add two digits a and b in base B, if the sum of the decimal values a and b exceed or equal to B, then there is a detention.

1 The Addition

1.1 Binary Addition :

- Adding two binary numbers is perfectly analogous to adding two decimal numbers.
- In fact, Binary addition is simpler since there are fewer cases to learn.
- We start by adding the digits of of the lowest order, the digits of the second row are then added, and so on. The same rules apply to binary addition.

- However, we will see four cases, which can arise when we add two binary digits, regadless of the rank. These four cases are :

  0 + 0 = 0

  1 + 0 = 1

  1 + 1 = 10 = 0 + carry over 1 to the left row.

  1 + 1 + 1 = 11 = 1 + carry over 1 to the left row.

Exemple :

	1	**1**			
		0	**1**	**1**	**0**
+		**1**	**1**	**0**	**1**
=	**1**	**0**	**0**	**1**	**1**

0110+1101 =10011

1.2 Addition in Any Base X (X=4, 8, 16, 3, …)

- Addition in any base X, is analogous to decimal addition.
- If the sum of two digits located in a column in the base, is equal to or exceeds the base, we remove the value of the base (We subtract X) from this sum.
- We put the rest of the result , and we retain 1 in the next column.

Exemple : $(2572)_8 + (310)_8 = (?)_8$

	1	**1**		
	2	**5**	**7**	**2**
+		**4**	**1**	**0**
=	**3**	**2**	**0**	**2**

7+1=8 ; 8-8=0 , we put **0 ,** and retain 1
5+4=9+1=10 ; 10-8=2, we put **2** and retain 1

- We apply the same principle whatever the base.

2- The Subtraction

2.1 Binary Subtraction :

- Subtration of binary number (the diminisher) of another number (the diminuand) is similar to the decimal subtraction, and involved a loan of 1 (really, we imprint 2, i.e, the base), In the case, where a digit of the diminisher is greater than that of the same rank of the diminuande.
- This borrow will be added to the next rank bit of the diminisher as 1.

Exemple

		1	**1**	**1**
	1	**1**	**0**	**0**
-	$_1$	$_1$**1**	$_1$**1**	**1**
=	**0**	**1**	**0**	**1**

1100 - 111 = 101

2.2 Subtraction in any Base X (X=3, 4, 8, 16, …)

- Subtration of a number (the diminisher) of another number (the diminuand) is similar to the decimal subtraction, and involved a loan of 1 (really, we imprint the X value, i.e, the base itself), In the case, where a digit of the diminisher is greater than that of the same rank of the diminuande.
- This borrow will be added to the next rank bit of the diminisher as 1.

Exemple : $(2572)_8$ - $(610)_8$ = $(?)_8$

		1		
	2	5	7	2
-	$_1$	6	1	0
=	1	7	6	2

- **5-6, we can't do it ;**
- **We make a print of 1,we have printes the base, 8**
- **5+8= 13 ; 13-6= 7 ;**
- **We return the 1 due to the next column.**

3. The Multiplication

3-1 Binary Multiplication.

- We multiply Binary numbers in the same way as we multiply decimal numbers.
- The process is simpler because the digits of the multiplier are always 0 or 1, so we always multiply by 0 or 1.
- Binar multiplication is based on the following four rules :
  - 1 x 1 = 1
  - 1 x 0 = 0
  - 0 x 1 = 0
  - 0 x 0 = 0

Exemple

				1	**1**	**0**	**1**
			*****	**1**	**0**	**1**	**1**
				1	**1**	**0**	**1**
			^{1}**1**	**1**	**0**	**1**	**.**
		^{1}**0**	**0**	**0**	**0**	**.**	**.**
^{1}	^{1}**1**	**1**	**0**	**1**	**.**	**.**	**.**
1	**0**	**0**	**0**	**1**	**1**	**1**	**1**

3.2 Multiplication in any base X (X=3,4,8,16,…)

- We multiply numbers in any base X in the same way as we multiply decimal numbers..
- When the multiplication of two digits aquals or exceeds the base X, we convert the result to the base in which we work.

Exemple : $(2103)_8$ x $(307)_8$ = $(?)_8$

					1			
		1			2			
			2	1	0	3		
*				3	0	7		
		$^{1}1$	$^{1}6$	7	2	5	3*7=21 =(2*8)+ 5	2*7=14 = (1*8) + 6
+		0	0	0	0	.		
+	6	3	1	1	.	.	3*3=9 =(1*8)+ 1	
=	6	5	0	0	2	5	7+1=8 -8= 0, retained 1	6+1+1 =8-8= 0, retained 1

4 The DIVISION

- The principale of binary division is similar to that of decimal division, but simpler in that each partial quotient is either equal to (possible division) or to 0 (impossible division).
- The division is the oppposite operation of multiplication, in the sense that one number is repeatedly subtracted from another until it is no longer possible, each time with a shift to the right.
- The first step is to subtract, starting from the left, the divisor from the dividend. If subtraction is not possible, the divisor is shifted one position to the right, then subtraction is performed.
- The next subtraction takes place between the result of the previous subtraction, increased to the right next bit of the dividend according to the rule stated previously
- This step is repeated until the dividend bits are exhausted. For each subtraction, we put 1 in the result, otherwise, we put 0.

Exemple:

```
  1 1 1 0 0 1 1 1      |     1 0 1
- 1 0 1                |-------------
  -------              |
  0 1 0 0 0            |   1 0 1 1 1 0
-     1 0 1            |
    ---------          |
    0 0 1 1 1          |
-       1 0 1          |
      ---------        |
        0 1 0 1        |
-         1 0 1        |
       ----------      |
          0 0 0 1      |
-           0 0 0      |
         ----------    |
             0 0 1     |
```

111100111 : 101 = 101110 reste 1.