Exemple1 :
Convertir le nombre binaire N=11010101101,001(2) en Octal.

Solution :

N= 0110 1010 1101 , 0010 $_{(2)}$
6 A D , 6 $_{(16)}$

Exemple 2 :
Convertir le nombre Hexadécimal N=AB0,C1$_{(16)}$ en binaire.

Solution
Ecrire par blocs de quatre bits, la valeur binaire des chiffres du nombre hexadécimal. On obtient :

A B 0 , C 1 $_{(16)}$
1010 1011 0000 , 1100 0001 $_{(2)}$

7 Les opérations arithmétiques en binaire :

7.1 L'addition

L'algorithme de l'addition des nombres binaires est le même que celui de l'addition des nombres décimaux [3].

Dressons la table d'addition
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0 et retenue de 1

Exemple :

```
    1  1
 10001101
+    1001
---------
=10010110
```

Remarque : 1 + 1 + 1 = 1 et retenue de 1.

7.2 La soustraction

Dressons la table de soustraction

0 - 0 = 0
1 - 0 = 1
1 - 1 = 0
0 - 1 = 1 avec retenue

Comme dans le cas du système décimal, la soustraction en binaire est également basée sur la notion de retenue

Exemple

```
      1 1 0 1 1 0
-       1 1 1 0 0
  -------------
=       0 1 0 1 0
```

7.3 La multiplication

Toujours comme dans le cas du système décimal, chaque nombre de la ligne du bas sera distribué à la ligne du haut. Ensuite, chaque ligne sera sommée (ne pas oublier les retenues) et le résultat sera alors obtenu.

Exemple

```
                    1 0 1 0
x                 1 0 1 0 1 0
-------------------------------
                  0 0 0 0 0 0
+               1 0 1 0 1 0 0
+             0 0 0 0 0 0 0 0
+           1 0 1 0 1 0 0 0 0
+         0 0 0 0 0 0 0 0 0 0
+     1 0 1 0 0 0 0 0 0 0 0 0
-------------------------------
=     1 1 0 1 0 0 1 0 0 0 0 0
```

7.4 La division

Pour la division en binaire, c'est le même algorithme classique de la division décimale enseigné dans les classes élémentaires. Elle s'effectue à l'aide de soustractions et de décalages, Les chiffres du quotient, plus simples, sont 0 ou 1 ; 1 lorsque le diviseur est plus petit que le dividende et 0 sinon.

Exemple

```
1011   | 100
-100   | 10,11
 0011
-000
  110
 -100
   100
  -100
     0
```

8 Les opérations arithmétiques en Octal :

8.1 L'addition

Comme pour le système binaire, on applique les mêmes règles pour les nombres octaux. Toutefois, dans ce cas, on aura la retenue **«1»** à gauche à chaque fois que la somme dépasse la valeur **7** car $\mathbf{7_{(8)}+1_{(8)}=10_{(8)}}$. (Voir le tableau 3)

Pour additionner deux chiffres octaux, on cherche ces chiffres sur le tableau 3**:** un en début de colonne, l'autre en début de ligne ; la somme se trouve dans la case correspondant au croisement entre colonne et ligne [3]. Par exemple, si l'on doit additionner les chiffres octaux **7** et **3,** on cherche la somme dans la case située au croisement de la colonne **7** et de la ligne **3.** Dans cette case, on trouve **12.** Ainsi, dans le système octal, on aura**: 7 + 5 = 14.**

Exemple d'addition en base octale:

$$\begin{array}{lccc} & {}^{1}1 & {}^{1}2 & 3_{(8)} \\ + & 4 & 5 & 7_{(8)} \\ \hline & 6 & 0 & 2_{(8)} \end{array}$$

±	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

Tableau 3 : Tableau d'addition et de soustraction dans le système octal

8.2 La soustraction

Comme en décimal mais en se limitant à 7 (octal). Il faut juste apprendre de nouvelles tables. Pour faire la soustraction de deux chiffres octaux, on cherche ces chiffres sur le tableau 3**:** le diminuende dans la case correspondant au croisement entre colonne et ligne, le diminuteur en début de colonne (ou en début de ligne), le résultat de soustraction se trouve en début de ligne (ou en début de colonne).

Exemple de soustraction octale:

	$^{1}1$	$^{1}2$	$_{1}3_{(8)}$
-	$_{1}4$	$_{1}5$	$7_{(8)}$
	4	4	$4_{(8)}$

9 Les opérations arithmétiques en Hexadécimal :

9.1 L'addition

Comme en décimal, l'addition s'effectue chiffre par chiffre. Toutefois, dans ce cas, on aura la retenue **«1»** à gauche à chaque fois que la somme dépasse la valeur **F** car $F_{16)}+1_{16)}=10_{(16)}$. (Voir le tableau 4)

Exemple d'addition Hexadécimale:

	1	$^{1}2$	$A_{(16)}$
+	E	5	$7_{(16)}$
	F	8	$1_{(16)}$

9.2 La soustraction

En se basant le tableau 4, la soustraction de deux chiffres hexadécimaux, se fait en cherchant ces chiffres sur le tableau**:** le diminuende dans la case correspondant au croisement entre colonne et ligne, le diminuteur en début de colonne (ou en début de ligne), le résultat de soustraction se trouve en début de ligne (ou en début de colonne).

Exemple de soustraction Hexadécimale**:**

	F	$^{1}2$	$A_{(16)}$
-	$_{1}E$	5	$7_{(16)}$
	0	D	$3_{(16)}$

±	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
1	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10
2	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11
3	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
4	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13
5	5	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14
6	6	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15
7	7	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16
8	8	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17
9	9	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18
A	A	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19
B	B	C	D	E	F	10	11	12	13	14	15	16	17	18	19	20
C	C	D	E	F	10	11	12	13	14	15	16	17	18	19	20	21
D	D	E	F	10	11	12	13	14	15	16	17	18	19	20	21	22
E	E	F	10	11	12	13	14	15	16	17	18	19	20	21	22	23
F	F	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24

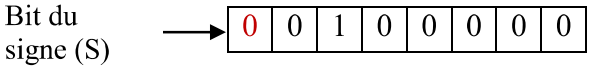
Tableau 4 : Tableau d'addition et de soustraction dans le système Hexadécimal

10 Les Entiers Négatifs

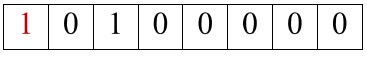
Les signe + et – ne sont pas reconnus par un ordinateur lequel ne connait que deux états : 0 et 1. On les représente donc par un bit qui occupera la case de gauche du nombre considéré. Ce bit s'appelle le bit de signe. Donc, par convention, on représente le signe + par 0 et signe – par 1. Les nombre négatifs sont représentés en machine par une des trois méthodes :**Signe et Valeur absolue**, **en complément à 1** ou **en complément à 2.**

10.1 Représentation des nombres négatifs en SVA (signe et valeur absolue)

C'est la représentation la plus simple d'un nombre négatif, il suffit de coder sa valeur absolue en binaire puis rajouter le bit du signe. Ainsi le nombre +32 est représenté sur 8 bits par :



Le nombre -32 s'écrira en SVA par :



Question :

Peut-on représenter le nombre -8 sur 04 bits.

Réponse :

Il est impossible de représenter le chiffre -8 sur 4 bits car sa valeur absolue $|-8_{(10)}|$ qui est égale à $1000_{(2)}$ prends déjà 04 bits et donc on aura besoin au minimum de 5 bits pour pouvoir représenter son bit de signe.

S	VA			Chiffre En décimal
0	0	0	0	+0
0	0	0	1	+1
0	0	1	0	+2
0	0	1	1	+3
0	1	0	0	+4
0	1	0	1	+5
0	1	1	0	+6
0	1	1	1	+7

S	VA			chiffre En décimal
1	0	0	0	-0
1	0	0	1	-1
1	0	1	0	-2
1	0	1	1	-3
1	1	0	0	-4
1	1	0	1	-5
1	1	1	0	-6
1	1	1	1	-7

Tableau 5 : Représentation des nombres par la méthode Signe et Valeur Absolue

Question :
Quels sont les nombres qu'on peut représenter sur 04 bits ?

Réponse : d'après le tableau 5, on peut représenter sur 4 bits, l'intervalle de nombres entiers : De $[-(2^3 - 1) , (2^3 - 1)]$ soit de [-7 , +7].

Plus généralement, si on travaille sur n bits, l'intervalle des valeurs qu'on peut représenter en SVA est : $[-(2^{n-1} - 1), +(2^{n-1} - 1)]$.

Cette méthode présente deux inconvénients :

- Le zéro possède deux (2) représentations distinctes 0000 et 1000 soit +0 et -0;
- Les tables d'additions et de multiplication sont compliquées, à cause du bit de signe qui doit être traité à part.

10.2 Représentation des nombres négatifs en CP1 (Complément à 1)

En binaire, on forme le complément à 1 (CP1) d'un nombre en soustrayant de 1chaque bit de ce nombre. Donc pour obtenir le complément a 1 d'un nombre binaire, il suffit de complémenter (ou d'inverser) chaque bit. Le 1 devient 0 et le 0 devient 1.

Exemple :
$N= 1\,0\,0\,0\,1\,1\,1\,0_{(2)} \rightarrow 0\,1\,1\,1\,0\,0\,0\,1_{(2)}$
$N= 0\,0\,1\,1\,0_{(2)} \rightarrow 1\,1\,0\,0\,1_{(2)}$

Remarque :

- la somme d'un nombre binaire et de son complément à 1 est un nombre binaire composé uniquement de 1.

- Le bit de poids fort est utilisé pour représenter le signe du nombre :

- Si ce bit = 1 alors il s'agit d'un nombre négatif
- Si ce bit = 0 alors le nombre est positif.

Question :
Quelle est la valeur décimale du nombre binaire suivant : 10110110 ?

Réponse :
Le bit de poids fort indique qu'il s'agit d'un nombre négatif. Donc la Valeur décimale = - CP1(10110110) = - $(01001001)_2$= - $(73)_{10}$

Question :
Quels sont les nombres qu'on peut représenter sur 04 bits ?

Valeurs En CP1				Valeurs En binaire	chiffre En décimal
0	0	0	0	0000	+0
0	0	0	1	0001	+1
0	0	1	0	0010	+2
0	0	1	1	0011	+3
0	1	0	0	0100	+4
0	1	0	1	0101	+5
0	1	1	0	0110	+6
0	1	1	1	0111	+7

Valeurs En CP1				Valeurs En binaire	Chiffre En décimal
1	0	0	0	-0111	-7
1	0	0	1	-0110	-6
1	0	1	0	-0101	-5
1	0	1	1	-0100	-4
1	1	0	0	-0011	-3
1	1	0	1	-0010	-2
1	1	1	0	-0001	-1
1	1	1	1	-0000	-0

Tableau 6 : Représentation des nombres par la méthode CP1

D'après le tableau 6, on peut déduire que sur 4 bits :

- Le plus grand nombre positif représentable est donc 0111 ce qui représente $2^3 - 1$ soit +7
- Le plus petit négatif est -0111. Ce qui donne $-(2^3 - 1)$ soit -7

Réponse :
Donc, on constate que sur 04 bits, on peut représenter les nombres qui sont dans l'intervalle [-$7_{(10)}$,+$7_{(10)}$], soit [-$(2^3 - 1)$, +$(2^3 - 1)$]

Plus généralement, si on travaille sur n bits, l'intervalle des valeurs qu'on peut représenter en CP1 est : [-$(2^{n-1} - 1)$, +$(2^{n-1} - 1)$].

Limite de la méthode :
Même cas pour la méthode SVA, le zéro possède deux (2) représentations distinctes. Par exemple sur 8 bits +$0_{(10)}$=**$00000000_{(2)}$**, -$0_{(10)}$= cp1(00000000)= **$11111111_{(2)}$**