SYSTELME DE NUMERATION-ARITHMETIQUE BINAIRE

GottfriedWilhelm von Leibnitz juin 1646, Allemagne

Ce philosophe d'origine Allemande est l'inventeur d'une machine permettant de calculer directement les 4 opérations de base. Il est aussi celui qui a introduit la notion de binaire en Occident.

I- Systèmes de numération

Le système de numération binaire est le plus important de ceux utilisés dans les circuits numériques, sans négliger l'importance des autres systèmes. Le système décimal revêt de l'importance en raison de son acceptation universelle pour représenter les grandeurs du monde courant.

De ce fait, il faudra parfois que des valeurs décimales soient converties en valeur binaire avant d'être introduites dans le circuit numérique. Par exemple, lorsque vous composer un nombre décimal sur votre ordinateur (ou sur votre calculatrice), les circuits interne convertissent ce nombre décimal en valeur binaire.

De même, il y aura des situations où des valeurs binaires données par un circuit numérique devront être converties en valeur décimale pour qu'on puisse les lire. Par exemple, votre ordinateur calcule la réponse à un problème au moyen du système binaire puis converti les réponses en des valeurs décimales avant de les afficher.

I-1 Système décimal

Le nombre 2857, par exemple, est le résultat de la somme suivante :

$$2857 = 2000 + 800 + 50 + 7$$

Soit encore

$$2857 = 2.10^3 + 8.10^2 + 5.10^1 + 7.10^0$$

Le nombre 2857 est écrit dans le système de numération décimal ou encore système à base 10.

Le système décimal correspond 10 nombres ou symboles qui sont {0,1…….,9} ; en utilisant ces symboles comme chiffres dans un nombre, on parvient à exprimer n'importe quelle grandeur. Le système décimal s'est imposé à l'homme puisque ce dernier possède 10 doigts.

D'une façon générale, dans un système de numération à base b, un nombre N est représenté par la suite :

$$N = a_n b^n + a_{n-1} b^{n-1} + ... + a_0 b^0 + a_{-1} b^{-1} + a_{-2} b^{-2} .....$$

Où a_i est un chiffre tel que $0 \le a_i \le (b-1)$. La partie fractionnaire, appelé aussi radical, est séparée de la partie entière par une virgule.

Les b^i sont appelés ''poids '', b^i est le poids de a_i.

Exemple :

$$53986{,}329 = 5.10^4 + 3.10^3 + 9.10^2 + 8.10^1 + 6.10^0 + 3.10^{-1} + 2.10^{-2} + 9.10^{-3}$$

Avec :

10^4	10^3	10^2	10^1	10^0	10^{-1}	10^{-2}	10^{-3}	**Poids**
4	**3**	**2**	**1**	**0**	**-1**	**-2**	**-3**	**Rang**

10^4 : Poids le plus fort

10^{-3} : Poids le plus faible

On note que le chiffre 4 est celui qui a le poids le plus élevé (MSB) et le chiffre -3 a le poids le plus faible (LSB).

I-2 Système binaire

Le système de numération binaire n'est qu'une autre façon de représenter les quantités. A première vue, ce système peut vous sembler plus complexe que le décimal ; il est pourtant plus simple puisqu'il ne possède que deux chiffres binaires, ou bits, sont le 1 et le 0. Le binaire est donc un système à base 2 car il comprend deux chiffres.

La proposition du 1 ou du 0 dans un nombre binaire indique son poids positionnels et détermine sa valeur dans le nombre, tout comme nous l'avons vu pour le système décimal. Dans un nombre binaire, les poids positionnels correspondent à des puissances de 2.

Exemple : $1011{,}11)_2 = 1.2^3 + 0.2^2 + 1.2^1 + 1.2^0 + 1.2^{-1} + 1.2^{-2}$

Avec : 2^3 est le bit de poids le plus fort et 2^{-2} est le bit de poids le plus faible.

$$N = \sum_{i=0}^{i=n} a_i 2^{i} \qquad \text{avec } 0 \leq a_i \leq 1 \quad i \geq 0$$

Il existe deux autres systèmes de numération très utilisés dans les circuits numériques : les systèmes à base 8 et à base 16 respectivement appelés système octal et système hexadécimal.

Pour chacun des systèmes binaire, octal et hexadécimal, l'ensemble des symboles possibles est :

- Pour le système binaire : {0,1}
- Pour le système octal : {0, 1, 2, 3, 4, 5, 6, 7}
- Pour le système hexadécimal :{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, F}

Exemples :

$$N = 754{,}15)_8 = 7.8^2 + +5.8^1 + 4.8^0 + 1.8^{-1} + 5.8^{-2}$$
$$N = A6F)_{16} = A.16^2 + 6.16^1 + F.16^0$$
$$N = B6D{,}3A)_{16} = B.16^2 + 6.16^1 + D.16^0 + 3.16^{-1} + A.16^{-2}$$

Le tableau ci-dessous montre les équivalences entre les systèmes binaire, octal, hexadécimal et le décimal.

Nombres décimaux	Binaire	Octal	Hexadécimal
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

Dans un système numérique, il peut arriver que trois ou quatre de ces systèmes de numération cohabitent, d'où l'importance de pouvoir convertir un système dans un autre.

Le passage d'un système décimal à un autre système de numération s'appelle le codage, l'opération inverse est le décodage. Le passage d'un système non décimal à un autre système non décimal est le transcodage.

Quelques définitions :

- Quartet : mot binaire de 4 bits
- Octet : mot binaire de 8 bits
- Mot : mot binaire de 16 bits
- Long mot : mot binaire de 32 bits

II- Changement de base : Codage - décodage

II-1 Conversion d'un nombre décimal en un nombre de base b quelconque.

Le passage du système décimal aux trois systèmes binaire, octal et hexadécimal s'effectue à l'aide de la division successive par b. Cette technique consiste à diviser par b autant de fois que cela est nécessaire pour obtenir un quotient nul, on écrit les restes dans l'ordre inverse où ils ont été obtenus ou bien on écrit le premier reste à la position de bit de poids le plus faible (LSB) et le dernier reste à la position de bit de poids le plus fort (MSB).

Considérons par exemple : $N = a_5b^5 + a_4b^4 + a_3b^3 + a_2b^2 + a_1b^1 + a_0b^0$

N peut être écrit successivement :

$$
\begin{aligned}
N &= b\left[a_5b^4 + a_4b^3 + a_3b^2 + a_2b^1 + a_1\right] + a_0 = bQ_1 + a_0 \\
Q_1 &= b\left[a_5b^3 + a_4b^2 + a_3b^1 + a_2\right] + a_1 = bQ_2 + a_1 \\
Q_2 &= b\left[a_5b^2 + a_4b^1 + a_3\right] + a_2 = bQ_3 + a_2 \\
Q_3 &= b\left[a_5b1 + a_4\right] + a_3 = bQ_4 + a_3 \\
Q_4 &= b\left[a_5\right] + a_4 = bQ_5 + a_4 \\
Q_5 &= b.0 + a_5 = b.0 + a_5
\end{aligned}
$$

Les restes lus de bas en haut donnent le nombre N dans la base b :

$$N = (a_5a_4a_3a_2a_1a_0)_b$$

Exemples d'application :

- Convertissez le nombre $19)_{10}$ en son équivalent binaire
- Convertissez le nombre $266)_{10}$ en son équivalent octal
- Convertissez le nombre $423)_{10}$ en son équivalent hexadécimal

II-2 De la base b à la base décimale

Tout nombre écrit dans une base b quelconque peut être transformée en son équivalent décimal en additionnant les termes obtenus du produit de chaque chiffre par son poids.

Exemples d'application :

- Convertissez le nombre $(11011,101)_2$ en son équivalent décimal
- Convertissez le nombre $(372)_8$ en son équivalent décimal
- Convertissez le nombre $(24,6)_8$ en son équivalent décimal
- Convertissez le nombre $(356)_{16}$ en son équivalent décimal
- Convertissez le nombre $(2AFD)_8$ en son équivalent décimal

II-3 Conversion octal-binaire et binaire-octal

La conversion octal-binaire s'effectue en transformant chaque chiffre du nombre octal en son équivalent binaire de trois chiffres. Pour le processus inverse, il suffit de faire avec le nombre binaire des groupes de trois bits en partant du chiffre de poids le plus faible, puis de convertir ces triplets en leur équivalent octal ; au besoin on ajoute des zéros à gauche du bits de poids le plus fort pour obtenir un nombre juste de triplets.

Exemples d'application :

- Convertissez le nombre $(472)_8$ en binaire

4	7	2
100	111	010

$(472)_8 = (100111010)_2$

- Convertissez le nombre $(5431)_8$ en binaire
- Convertissez le nombre $(11010110)_2$ en octal

II-4 Conversion hexadécimal-binaire et binaire-hexadécimal

Pour la conversion hexadécimal-binaire, on remplace chaque chiffre du nombre hexadécimal par son équivalent de 4 bits. L'opération inverse, consiste à faire diviser le nombre binaire en groupe de 4 bits, puis on substitue à chaque groupe son chiffre hexadécimal équivalent. Au besoin, on ajoute des zéros à gauche pour obtenir un dernier groupe de 4 bits.

Exemples d'application :

- Convertissez le nombre $(9F2)_{16}$ en binaire

9	F	2
1001	1111	0010

$(9F2)_8 = (100111110010)_{16}$

- Convertissez le nombre $(1110100110)_2$ en hexadécimal

II-5 Cas des nombres fractionnaires

Considérons un nombre N ayant une partie fractionnaire et une partie entière. La division successive par la base b s'applique à la partie entière alors que pour la partie fractionnaire, on procède de la manière suivante :

- On multiplie la partie fractionnaire exprimé dans la base 10 par b, on obtient un résultat $N1)_{10}$.
- La partie entière de $N1)_{10}$ exprime le premier chiffre de la partie fractionnaire en base b.
- On calcule $N2)_{10}=N1)_{10}$ - (partie entière de $N1)_{10}$) la valeur du second chiffre est obtenu en appliquant à $N2)_{10}$ la même procédure que $N1)_{10}$.

Exemples d'application :

Convertir un nombre fractionnaire de base b en décimal

$N=0,01010)_2$ en décimal

=0+0.25+0+0.0625+0=0.3125

Convertir un nombre fractionnaire de base décimal en base b

$N=20,4)_{10}$

Partie entière :

$20)_{10} = 10100)_2$

Partie fractionnaire :

$0,4)_{10} = 0,0110)_2$

$N = 20,4)_{10} = 10100,0110)_2$

$N = 0,2143)_{10} = 0,001101101)_2$

Remarque : L'exemple a montré que cette conversion peut ne pas se terminer et par conséquent, lorsqu'on s'arrête, on obtient une approximation de la représentation du nombre.

III- Arithmétique binaire

L'arithmétique binaire est essentielle dans tous les ordinateurs et d'autres types de systèmes numériques. Pour comprendre le fonctionnement de ces appareils, on doit connaître les fondements de l'addition, la soustraction, la multiplication et la division.

III-1 Addition binaire

L'addition est l'opération arithmétique la plus importante dans les systèmes numériques. Les opérations de soustraction, de division et de multiplication effectuée par les ordinateurs ne sont que des variantes de l'opération d'addition, sa table est la suivante :

A	B	Somme	Retenue
0	**0**	**0**	**0**
0	**1**	**1**	**0**
1	**0**	**1**	**0**
1	**1**	**0**	**1**

On écrit les nombres à additionner les uns sous les autres, on commence par additionner les bits correspondant au plus petit poids, les 1 de retenue sont considérés comme de nouveau bits et additionnés avec ceux de la colonne de poids juste supérieur.

Exemples :

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

101110011111001+110111101010110=1100110001001111

10111,01101+11001,00111=110000,10100

Remarque : Il n'est pas nécessaire d'étudier des additions ayant plus de deux nombres binaires, car dans tous les systèmes numériques les circuits qui additionnent ne traitent pas plus de deux nombres à la fois. On additionne les deux premiers et la somme est additionnée au troisième nombre et ainsi de suite (ce n'est pas un inconvénient, puisque les machines numériques moderne peuvent réaliser une opération d'addition en moins d'une microseconde).

III-2 Soustraction binaire

La table de soustraction est la suivante :

A	B	Soustraction	Retenue
0	**0**	**0**	**0**
0	**1**	**1**	**1**
1	**0**	**1**	**0**
1	**1**	**0**	**0**

La retenue de 1 sera retranchée du chiffre de rang supérieur.

On effectue la soustraction conformément à la table ci-dessus en commençant par les bits correspondant au plus petit poids. Si le nombre à soustraire est plus grand que le soustractande, on soustrait ce dernier du nombre à soustraire et le résultat est négatif.

Exemples :

1011011-101111=101100

1001011-101111=011100

1111-0111= -1000

III-3 Multiplication binaire

La disposition des nombres à multiplier est la même en binaire qu'en décimal. La table de multiplication est particulièrement simple.

A	B	Multiplication
0	0	0
0	1	0
1	0	0
1	1	1

Exemples :

101101 * 101=11100001

1101101*1010011=10001101010111

III-4 Division binaire

La division des nombres binaires est identique à la division de nombres décimaux.

Exemples :

1011101110 / 110=1111101 le reste est 000

1001/11=011

100011000/11000=01011 le reste est 10000

L'addition binaire est l'opération arithmétique la plus importante dans les systèmes numériques. Comme nous le verrons, les opérations de soustraction, de multiplication et de division effectuées par les ordinateurs ne sont essentiellement que des variantes de l'opération d'addition.

III-5 Ecriture des nombres signés

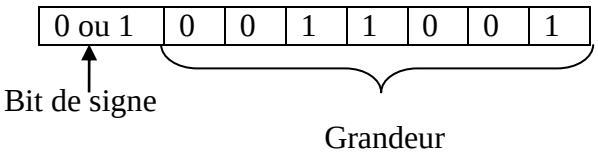
Comme les systèmes numériques traitent aussi les nombres négatifs que les nombres positifs, une certaine convention est adoptée pour représenter le signe du nombre (+ ou -).

Généralement, un autre bit appelé bit de signe est ajouté au nombre. La convention consiste à attribuer au nombre positif le bit de signe 0 et au nombre négatif le bit de signe 1.

Exemples :

0 0011001 $\Longleftrightarrow$ + $0011001)_2 = +\ 25)_{10}$

1 0011001 $\Longleftrightarrow$ - $0011001)_2 = -\ 25)_{10}$



Bien que cette notation signe-grandeur soit directe, les calculateurs numériques n'y ont généralement pas recours, en raison de la complexité des circuits qui matérialisent cette notation d'où l'utilisation dans ces machines de la notation en complément à deux pour représenter les nombres signés.

III-6 Notation en complément à 1 ou complément restreint CR(N)

En binaire, on forme le complément à 1 d'un nombre en changeant chaque ''0''par un ''1'' et chaque ''1'' par un ''0''. Autrement dit, en complémentant chaque bit du nombre.

Exemples :

Trouver le complément à 1 du nombre $N=110110110101)_2$

$$N = 110110110101)_2 \quad \Leftrightarrow \quad \overline{N} = CR(N) = 001001001010)_2$$

Soit donc $N+CR(N)=2^n-1$ ce qui donne $CR(N)= (2^n-1)-N$

Trouver le complément à 1 du nombre N=1011

$\overline{N} = 1111 - 1011 = 0100 = (2^4 - 1) - 1011$

III-7 Complément à 2 ou complément vrais CV(N)

- En binaire, trouver le complément à deux d'un nombre revient à le soustraire de la puissance immédiatement supérieure.

Exemple :

N=1011 2^4=10000

CV(1011)=10000-1011=0101

CV(1011)= 2^4-1011=0101

Soit donc CV(N)= 2^n-N=CR(N)+1

- En binaire, trouver le complément à 2 revient aussi à trouver le complément à 1 et à ajouter un 1 au résultat.

Exemple :

10011101 nombre initial

01100010 complément à 1

1

01100011 complément à 2

- Trouver le complément à deux revient aussi partant de la droite vers la gauche, à garder tous les bits jusqu'au premier ''1''rencontrer et de changer les autres bits de ''1'' en ''0'' et ''0'' en ''1''.

Exemples :

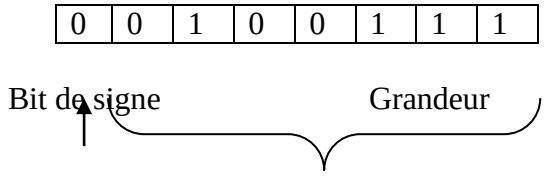
N= 1100010000 CV(N)= 0011110000

N= 11101000 CV(N)= 00011000

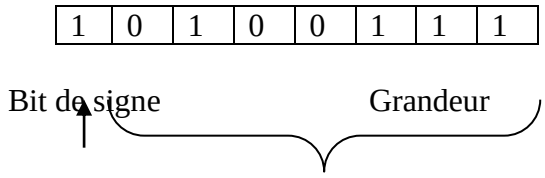
Application :

Exprimer le nombre décimal – 39 en un nombre de 8 bits en utilisant la notation signe – grandeur, le complément à 1 et le complément à 2.

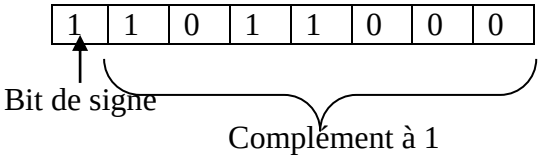
Ecrivons d'abord le nombre de 8 bits de + 39 :



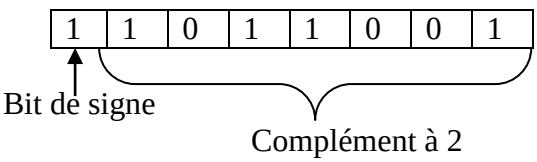
Dans la représentation signe – grandeur, on représente – 39 comme suit :



La notation en complément à 1 :



La notation en complément à 2 :



III-8 Nombres à virgule flottante

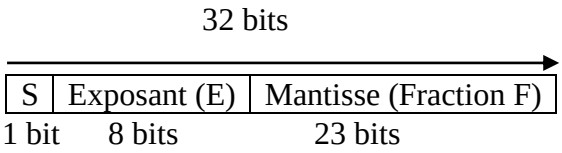
Le système des nombres à virgule flottante permet de représenter des nombres de très grande ou très petite et/ou des nombres possédants une partie entière et une partie fractionnaire sans augmenter le nombre de bits.

Exemple :

N=252415711

Le nombre à virgule flottante s'écrit : $\underbrace{0{,}252415711}_{\text{Mantisse}}\ \underbrace{10^{9}}_{\text{Exposant 9}}$

Pour un nombre binaire à virgule flottante de précision simple (32 bits) :



- **N=1011010010001**

Pour un nombre signé en complément à 2, l'échelle des valeurs est $-(2^{n-1})$ à $+(2^{n-1}-1)$. Pour un exposant de 8 bits [-128,127].

$1011010010001=1{,}011010010001\ 2^{12}$

Le nombre est positif (S=0) ; l'exposant polarisé s'obtient en ajoutant 127 :

12+127= 139 ce qui donne E= 10001011 et F =0,011010010001

Un 1 à la gauche de la virgule binaire dans l'expression de puissance de 2, il n'est pas inclus dans la mantisse. Le nombre à virgule flottante complet est donc :

S	E	F
0	10001011	01101001000100000000000

Voyant maintenant comment évaluer un nombre binaire déjà écrit en notation à virgule flottante. L'approche générale pour déterminer la valeur d'un nombre à virgule flottante est exprimée par la formule suivante :

$$N = (-1)^S \ (1+F) \ (2^{E-127})$$

Exemple :

1	10010001	10001110001000000000000

Le bit de signe est 1et l'exposant polarisé est : 10010001= 145

$N = (-1)1 \ (1+0,10001110001) \ (2^{145-127})$

$N = (-1) \ (1,10001110001) \ 2^{18}$

N = 1100011100010000000

III-9 Opérations arithmétique avec des nombres signés

Comme la notation en complément à deux est la plus couramment utilisée dans les ordinateurs et les systèmes à microprocesseurs pour la représentation des nombres binaires signés, nous allons étudier des opérations arithmétiques utilisant cette notation.

III-9-1 Soustraction ramené à l'addition

On désire effectuer l'opération : A-B

Soit A+CV(B)= A+(2^n-B) CV(B)= 2^n-B

= A-B+2^n $\Longrightarrow$ **A-B= A+CV(B)- 2^n**

- Si $A \succ B$:

On a $A - B \geq 0 \Rightarrow A + CV(B) \geq 2^n \Longrightarrow$ A-B=A+CV(B)- 2^n conduit à un dépassement de capacité.

Exemple :

9	1	1001
5	0	1011
4	10	0100
	↑	

dépassement de capacitéde2^n

La soustraction se ramène donc à une addition dans laquelle on néglige la retenue qui apparaît comme le bit de poids 2^n.

- **Si $A \prec B$**

On a :

$A - B \leq 0 \Rightarrow A + CV(B) \leq 2^n \Rightarrow A + CV(B) = A - B + 2^n \Rightarrow A + CV(B) = 2^n - (B - A) = CV(B - A)$
Il n'y a pas donc de retenue de poids 2^n. Pour avoir le résultat B-A, on prendra donc le
$CV[CV(B - A)] = 2^n - CV(B - A) = 2^n - (2^n - (B - A)) = B - A$

Exemple :

9	1	1001
−15	0	1011
−6	1	1010

1010 est le complément à 2 de -6

III-9-2 Addition en utilisant le complément à deux

Il existe quatre cas d'addition de nombres binaires signés :

1- Les deux nombres sont positifs
2- Le nombre positif est plus grand que le nombre négatif
3- Le nombre positif est plus petit que le nombre négatif
4- Les deux nombres sont négatifs

Etudiant chaque cas séparément en utilisant des exemples comportant des nombres signés de 8 bits. Les équivalences décimales sont également illustrées.

- **Cas I : Deux nombres positifs**

L'addition de deux nombres positifs est immédiate : Soit l'addition de +9 et +4.

	Addition par la table		Addition codée	
9	1001		0 1001	cumulande
+4	0100		0 0100	cumulateur
---	---		---	
13	1101	bit de signe	0 1101	

Remarquez que les bits de signe sont 0 et que celui de la somme est aussi 0, ce qui indique un nombre positif. Il faut que le cumulande et le cumulateur aient le même nombre de signe.

- **Cas II : Nombre positif et nombre négatif plus petit**

Soit l'addition de +9 et -4.

– 4 est exprimé dans la notation en complément à 2 : 1 1100

Soustraction d'après la table		Addition codée	
+9	1001	0 1001	cumulande
-4	0100	1 1100	cumulateur

-----	------	----------
+5	+ 0101	**1** 0 0101

Retenue ignorée

D'après l'addition codée, on remarque qu'il y a un report ignoré au moment de l'addition des bits de poids le plus fort avec les bits de signes d'où la somme finale de 00101, soit le nombre décimal + 5.

- **Cas III : Nombre positif et nombre négatif plus grand**

Soit l'addition de -9 et + 4.

– 9 est exprimé dans la notation en complément à 2 : 1 0111

Soustraction d'après la table		Addition codée	
-9	1001	1 0111	cumulande
+4	0100	0 0100	cumulateur
--------	------	----------	
-5	- 0101	1 1011	

Le résultat de cette addition codée est le complément à 2 du résultat final qui est 1 0101

- **Cas IV : Deux nombres négatifs**

Soit l'addition de – 9 et - 4

Addition par la table		Addition codée	
-9	- 1001	1 0111	cumulande
-4	- 0100	1 1100	cumulateur

------	------		----------
-13	-1101	**1**	1 0011

Retenue ignorée

Le résultat définitif est négatif et le résultat de l'addition codée est le complément à 2 du résultat final qui est 1 1101.

- **Cas V : Nombres égaux et opposés**

Soi -9 et +9

Soustraction d'après la table		Addition codée	
-9	-1001	1 0111	cumulande
+9	1001	0 1001	cumulateur
------	------	----------	
0	0000	1 0 0000	

Retenue ignorée

Le résultat est évidemment 0.

Remarque : Dans un ordinateur, les nombres négatifs sont stockés sous forme de complément à 2.

III-9-3 Addition en utilisant le complément à un

Les nombres négatifs sont représentés par leur complément à 1. Les quatre cas étudiés précédemment peuvent se présenter.

- **Cas I : Deux nombres positifs**

L'addition se fait dans les mêmes conditions que précédemment.

- **Cas II : Nombre positif et nombre négatif plus petit**

Soit l'addition de +12 et -5.

– 5 est exprimé dans la notation en complément à 1 : 1 1010

Soustraction d'après la table		Addition codée	
+12	1100	0 1100	cumulande
-5	0101	1 1010	cumulateur
------	------	----------	
+7	0111	1 0 0110	
		1	
		----------	
		0 0111	

Le résultat est obtenu en ajoutant la retenue de la somme des bits de poids le plus fort aux bits de signe. La retenue de cette opération est ajoutée aux bits de poids plus faible du résultat de l'addition.

- **Cas III : Nombre positif et nombre négatif plus grand**

Soit l'addition de 3 et -11.

Soustraction d'après la table		Addition codée	
+3	0011	0 0011	cumulande
-11	1011	1 0100	cumulateur
------	------	----------	
-8	0111	1 0 0111	

Le résultat de cette addition est le complément à 1 du résultat final qui est 1 1000.

- **Cas IV : Les deux nombres négatifs**

Soit l'addition de -5 et -2

Soustraction d'après la table		Addition codée	
-5	0010	1 1101	cumulande
-2	0101	1 1010	cumulateur
------	------	----------	
-7	- 0111	1 1 0111	
		1	
		----------	
		1 1000	

Le résultat obtenu est le complément à 1 du résultat final qui 1 0111)2= - (7)10