Exercice 1 :

Le principe est le même qu'en décimal

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

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

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

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

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

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

Exercice 2 :

Les nombres positifs sont toujours écrits selon la notation binaire exacte et le bit de signe est toujours 0.

a) 48 est un nombre positif, sa grandeur exacte est : 1 1 0 0 0 0

La représentation en complément à 2 sur 8 bits est : $48_{(10)} = 0\ \underbrace{0\ 1\ 1\ 0\ 0\ 0\ 0}_{\text{Grandeur}}{}_{(2)}$

Bit de signe

b) $23_{(10)} = 00010111_{(2)}$

c)

25 = 00011001
11100110 ← Complément de chaque bit
+1 ← ajouter 1 au bit de poids le plus faible
11100111 ← complément à 2 de -25

$-25_{(10)} = 11100111_{(2)}$

d)

92 = 01011100
10100011
+1
10100100

$-92_{(10)} = 10100100_{(2)}$

Exercice 3 :

a) $00101011_{(2)}$ Le bit de signe est 0 donc le nombre est positif. Les 7 autres bits représentent la grandeur exacte du nombre.

$00101011_{(2)} = +43_{(10)}$

b) $10011110_{(2)}$ Le bit de signe est 1 donc le nombre est négatif.

10011110 ← Nombre négatif original
01100001 ← Complément à 1
+1 ← ajouter 1
01100010 = $98_{(10)}$

Comme le résultat de la complémentation à 2 est +98, le nombre original doit donc être -98.
<u>La comlémentation à 2 change le signe du nombre signé.</u>

Exemples : complément à 2 de +5 est -5, complément à 2 de -17 est +17

c) $10011000_{(2)}$

10011000 ← Nombre négatif original
01100111 ← Complément à 1
+1 ← ajouter 1
01101000 = $104_{(10)}$

$10011000_{(2)} = -104_{(10)}$

d) $10000000_{(2)}$

Quand un nombre signé à 1 comme de bit de signe et que des 0 comme bits de grandeur, son équivalent décimal est -2^n, ou n est le nombre de bits de grandeur.

Exemples : 1000 = -2^3 = -8

10000 = -2^4 = -16

100000 = -2^5 = -32

Par conséquent on peut donc affirmer que l'intervalle complet des valeurs que l'on peut écrire en représentation en complément à 2 au moyen de n bits de grandeur est : -2^n à (2^n-1)

$10000000_{(2)} = -2^7 = -128_{(10)}$

Exercice 4 :

L'intervalle de valeurs décimales signées qu'on peut représenter sur un format de 6 bits est -32 à +31.

L'intervalle de valeurs décimales signées qu'on peut représenter sur un format de 8 bits est -128 à +127.

Exercice 5 :

a) $45_{(10)} = 00101101_{(2)}$

17 = 00010001
11101110
+1
-17 = 11101111

45 **-** 17 = 45 + (**-**17)

00101101
+ 11101111
$\cancel{1}$00011100

Ce report n'est pas pris en considération, le résultat est donc $00011011_{(2)} = 28_{(10)}$

Bit de signe

b) **-**15 **-** 28 =

$-15_{(10)}$ = 11110001
+
$-28_{(10)}$ = 11100100
$\cancel{1}$11010101

Le bit de signe est 1 donc le résultat est un nombre négatif.

11010101
00101010
+1
11101111 = $43_{(10)}$

Le résultat est 11010101 = $-43_{(10)}$

c) 12 **-** 42 = 12 + (**-** 42)

$+12_{(10)}$ = 00001100
+
$-42_{(10)}$ = 11010110
11100010

11100010
00011101
+1
00011110 = $30_{(10)}$

Le résultat est 11100010 = $-30_{(10)}$

Exercice 6 :

Additionneur binaire de deux nombres à deux bits (A : a_1 a_0), (B : b_1 b_0).

$$\begin{array}{r} a_1\ a_0 \\ +\ \ b_1\ b_0 \\ \hline S_2\ S_1\ S_0 \end{array}$$

1°) Table de vérité :

b_1	b_0	a_1	a_0	S_2	S_1	S_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	1	1
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	1
0	1	1	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

2°) Les équations simplifiées des sorties S_2 S_1 S_0 en utilisant les tableaux de Karnaugh

S_0 :

b_1b_0 \ a_1a_0	00	01	11	10
00	0	1	1	0
01	1	0	0	1
11	1	0	0	1
10	0	1	1	0

$$\boxed{S_0 = a_0\bar{b}_0 + \bar{a}_0 b_0 = a_0 \oplus b_0}$$

S_1 :

b_1b_0 \ a_1a_0	00	01	11	10
00	0	0	1	1
01	0	1	0	1
11	1	0	1	0
10	1	1	0	0

S_2 :

b_1b_0 \ a_1a_0	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	0	1	1	1
10	0	0	1	1

$$S_1 = \bar{b}_1\bar{b}_0 a_1 + \bar{b}_1 a_1 \bar{a}_0 + b_1 \bar{a}_1 \bar{a}_0 + b_1 \bar{b}_0 \bar{a}_1 + \bar{b}_1 b_0 \bar{a}_1 a_0 + b_1 b_0 a_1 a_0$$

$$S_1 = \bar{a}_0(a_1\bar{b}_1 + \bar{a}_1 b_1) + \bar{b}_0(a_1\bar{b}_1 + \bar{a}_1 b_1) + a_0 b_0(a_1 b_1 + \bar{a}_1\bar{b}_1)$$

$$S_1 = \bar{a}_0(a_1 \oplus b_1) + \bar{b}_0(a_1 \oplus b_1) + a_0 b_0(\overline{a_1 \oplus b_1})$$

$$S_1 = (a_1 \oplus b_1)(\bar{a}_0 + \bar{b}_0) + a_0 b_0(\overline{a_1 \oplus b_1})$$

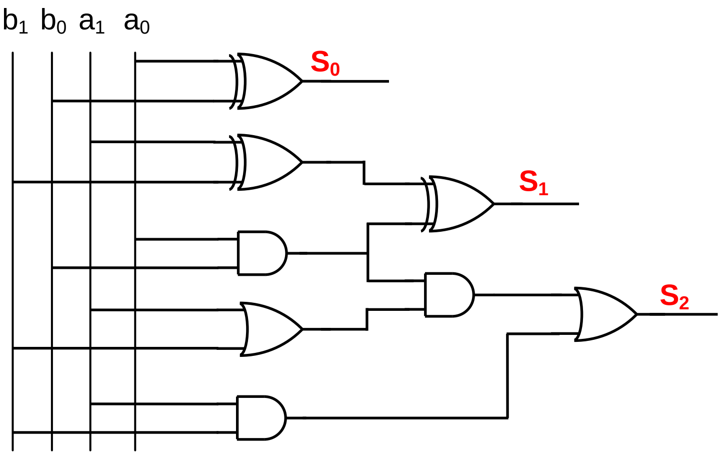
$$S_1 = (a_1 \oplus b_1)(\overline{a_0 b_0}) + a_0 b_0(\overline{a_1 \oplus b_1})$$

$$\boxed{S_1 = (a_1 \oplus b_1) \oplus (a_0 b_0)}$$

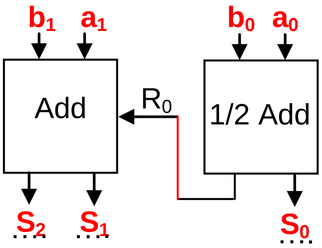
$$S_2 = a_1 b_1 + b_1 b_0 a_0 + b_0 a_1 a_0$$

$$\boxed{S_2 = a_1 b_1 + a_0 b_0 (a_1 + b_1)}$$

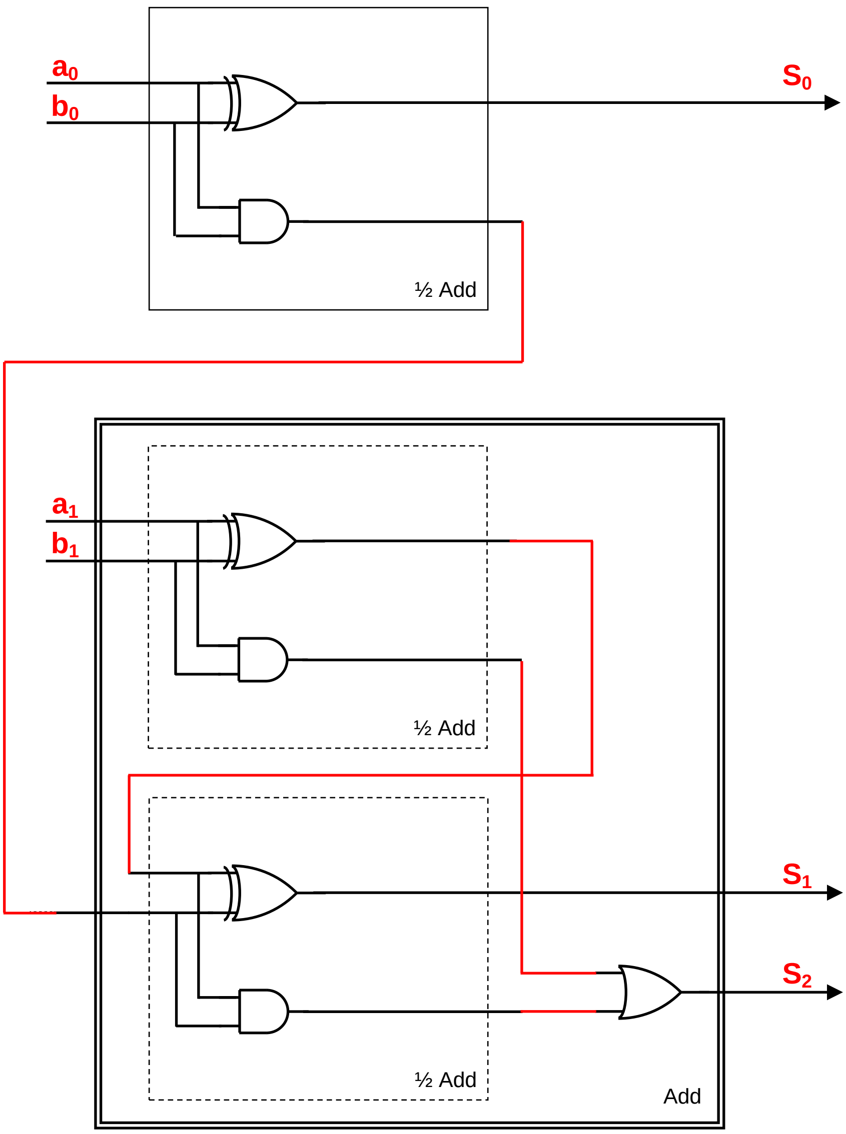
3°) Le logigramme de l'additionneur en utilisant les opérateurs : XOR, OR et AND



4°) Réalisation du même additionneur avec un demi additionneur et un additionneur complet.



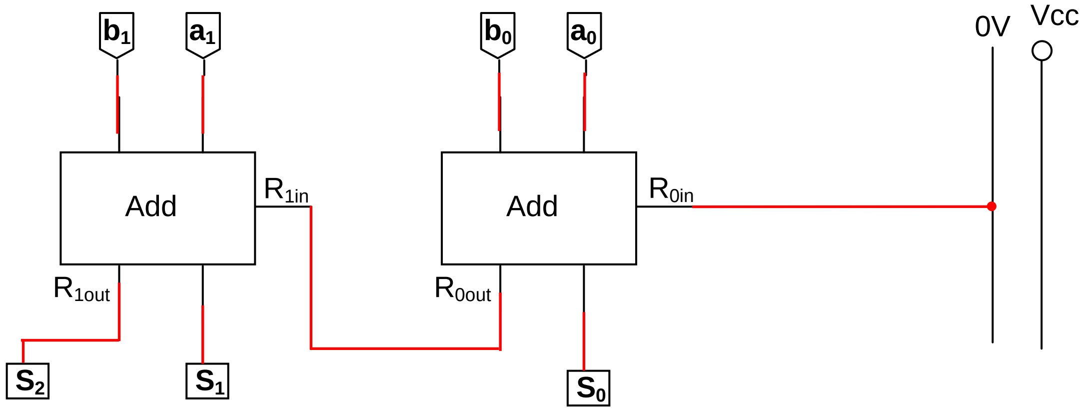
$$\begin{array}{r} a_1\ a_0 \\ +\ \ b_1\ b_0 \\ \hline S_2\ S_1\ S_0 \end{array}$$



Exercice 7 :

1°) Schéma du circuit qui permet l'addition de deux mots binaires de deux bits :
$A_{(2)} = a_1a_0$; $B_{(2)} = b_1b_0$ en utilisant deux additionneurs complets

Opération A+B



2°)

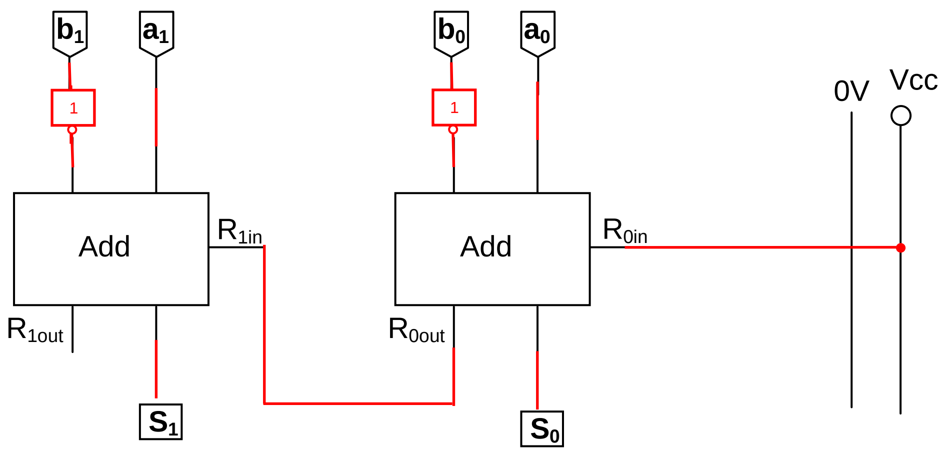
2-1) Opération de soustraction avec la représentation en complément à 2

$A - B = A + \overline{B} + 1$

2-2) L'intervalle des valeurs décimales que l'on peut écrire avec 2 bits en tenant compte du bit de signe est
-2 ,-1 0 ,1 [-2 ,1]

2-3) Schéma

Opération A - B



3) On veut maintenant exploiter les deux additionneurs complets et deux portes OU exclusif pour réaliser soit une addition soit une soustraction (résultat sur 2 bits) selon l'état logique d'une entrée notée K comme suit :

- K=0 on réalise A + B
- K=1 on réalise A - B

3-1) $b \oplus 0 = \mathbf{b}$; $b \oplus 1 = \overline{b}$

3-2) Schéma