Les opérations arithmétiques sur les nombres binaires

Site Internet : **www.gecif.net**	Type de document : **Cours**	Intercalaire :	Date :

I – Principe de l'addition binaire

I – 1 – Table d'addition élémentaire

En binaire il n'existe seulement 2 chiffres : le 0 et le 1. Un nombre binaire est un ensemble de chiffres binaires, appelés des bits. Lors de l'addition de 2 chiffres binaires (de 2 bits) il n'y a que 3 possibilités (en effet en raison de la commutativité de l'addition on a 0 + 1 = 1 + 0) :

0 + 0 =

0 + 1 =

1 + 1 =

A	B	S	R
0	0		
0	1		
1	0		
1	1		

S est la somme binaire des deux bits A et B, et R est la retenue éventuelle.

Exemples d'additions binaires :

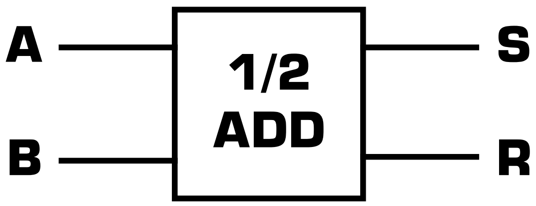
I – 2 – Le montage demi-additionneur

………

………

………

Symbole du demi-additionneur :



Equation des sortie d'après la table d'addition élémentaire :

S =

R =

Structure interne d'un demi-additionneur :

demi-additionneur

Limites du ½ additionneur :

………

………

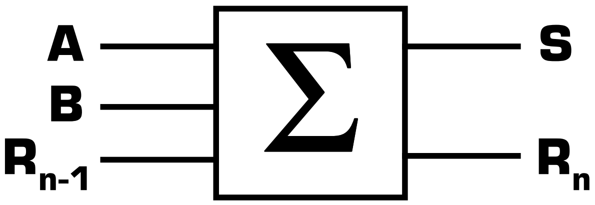
………

II – L’additionneur complet cascadable

II – 1 – Le montage additionneur complet

A la différence du ½ additionneur, l’additionneur complet dispose d’une 3ème entrée permettant de prendre en compte une éventuelle retenue. En mettant en cascade plusieurs additionneur complet il est alors possible d’effectuer l’addition (c’est-à-dire de calculer la somme) de 2 nombres binaires d’une taille quelconque.

Symbole de l’additionneur complet :



Remarque :

la lettre Σ (lettre grecque **Sigma** majuscule) au centre de l’additionneur symbolise la somme en arithmétique.

L’additionneur complet possède 3 entrées (A, B et R_{n-1}) et 2 sorties (S et R_n). A et B sont les 2 bits de rang n à additionner, S est la somme des bits A, B et R_{n-1}, R_n est la retenue éventuelle de rang n, et R_{n-1} est la retenue éventuelle du rang précédent, c’est-à-dire la retenue du rang n-1.

L’additionneur complet doit calculer la somme de 3 bits : $A + B + R_{n-1}$. Le 0 étant l’élément neutre de l’addition, lors de l’addition de 3 bits un seul cas est nouveau par rapport à l’addition de 2 bits :

1 + 1 + 1 =

Table de vérité de l’additionneur complet :

A	B	R_{n-1}	S	R_n
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Equations logiques des sorties :

S =

R_n =

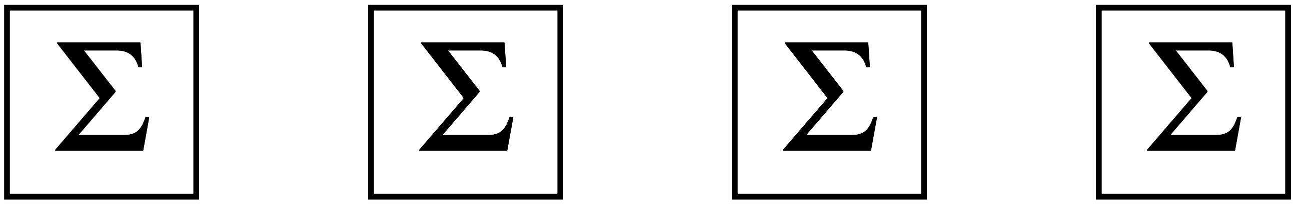
Structure interne de l’additionneur complet cascadable :

additionneur complet cascadable

II – 2 – Mise en cascade de plusieurs additionneurs complets

L'additionneur complet prenant en compte la retenue du rang précédent, il est cascadable. La mise en cascade permet d'effectuer une addition entre 2 nombres binaires d'une taille quelconque en connectant entre eux plusieurs additionneurs complet, chacun prenant en compte la retenue précédente.

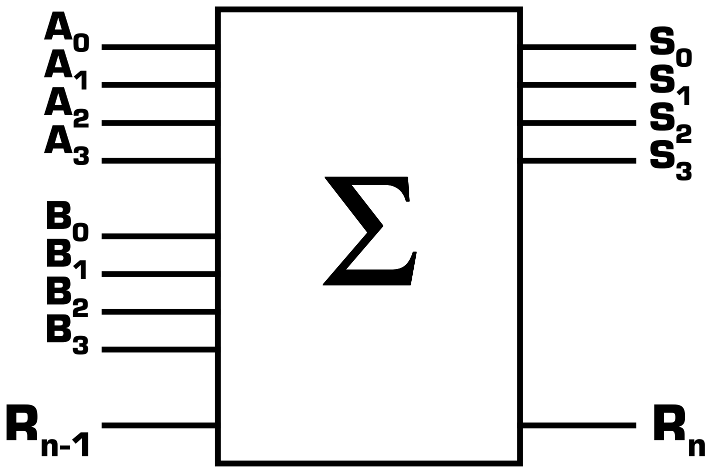
Exemple de mise en cascade de 4 additionneurs complet pour calculer la somme de deux nombres binaires de 4 bits chacun :



Remarques :

- A_3 A_2 A_1 et A_0 sont les 4 bits du 1[er] nombre binaire (A_0 est le LSB et A_3 est le MSB)
- B_3 B_2 B_1 et B_0 sont les 4 bits du 1[er] nombre binaire (B_0 est le LSB et B_3 est le MSB)
- Le résultat est composé des 5 bits S_4 S_3 S_2 S_1 et S_0
- L'entrée de retenue du premier additionneur (le poids faible) est mise à 0
- La sortie de retenue du dernier additionneur (le poids forts) constitue le 5[ème] bit du résultat
- Chaque entrée R_{n-1} est reliée à la sortie R_n de l'additionneur complet de rang inférieur
- Ce montage effectue l'addition bit à bit en tenant compte des retenues éventuelles, telle qu'on la fait à la main

Symbole générique d'un additionneur complet 4 bits :



La lettre Sigma au centre du symbole indique instantanément la fonction réalisée

III – Représentation des nombres négatifs en binaire

Rappel : on appelle **complément à 1** d'un nombre le nombre obtenu en complétant chacun de ses bits.

....

Exemple :

Limites du complément à 1 :

……

……

Exemple :

Pour représenter les nombres négatifs en binaire il y a deux solutions.

<u>III – 1 – Solution 1</u>

On ajoute un bit de signe à gauche de la représentation binaire du nombre :
- Si le bit de signe = 0 alors le nombre est **positif**
- Si le bit de signe = 0 alors le nombre est **négatif**

Exemple :

Avantage : ……

……

Limites de la représentation des nombres négatifs avec un bit de signe :

……

……

……

Exemple : pour afficher la valeur numérique d'une température on pourra utiliser un bit de signe pour coder les nombres relatifs. Dans ce cas le bit de signe représente l'état du signe moins de l'afficheur.

<u>III – 2 – Solution 2 : le complément à 2</u>

……

……

……

$$- N =$$

Avantage du complément à 2 : le complément à 2 permet de représenter des nombres signés (positifs et négatifs) sans ambiguïté. De plus il permet d'effectuer des opérations (somme et différence) sur les nombres relatifs en utilisant un simple additionneur.

Inconvénient du complément à 2 : il est plus difficile d'effectuer $\overline{N} + 1$ que de coder $\overline{N}$ ou d'ajouter simplement un bit de signe. La fonction complément à 2 ne fait pas partie des fonctions de base de la logique ou de l'arithmétique.

Exemple d'application du complément à 2 : pour effectuer la sous traction A – B on va ajouter à A le complément à 2 de B : $A - B = A + \overline{B} + 1$

Exemples numériques :

ATTENTION : tous les nombres, y compris le résultat, doivent être exprimés dans le même format (par exemple sur 4 bits).

Remarque : le complément à 2 du complément à 2 du nombre N est le nombre N lui même. La fonction réciproque du complément à 2 est donc la fonction complément à 2 elle-même. On dit alors que la fonction complément à 2 est une **involution** :

IV – Montage calculant le complément à 2 d'un nombre

On désire réaliser un montage logique fournissant en sortie le complément à 2 d'un nombre N donné sur 4 bits à l'entrée du montage :

Fonction complément à 2

Remarque : si N est sur 4 bits sont complément à 2 est sur 5 bits.

Pour réaliser la fonction complément à 2, il y a deux solutions.

IV – 1 – Solution 1

On considère que la fonction complément à deux est un montage en logique combinatoire classique (comme un transcodeur) et on recherche les équations de chacune des sorties en fonction des entrées.

Table de vérité de la fonction complément à 2 :

N	**E_0**	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
	E_1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
	E_2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
	E_3	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
$\overline{N}+1$	**S_0**																
	S_1																
	S_2																
	S_3																
	S_4																

Equations immédiate d'après la table de vérité :

S_0 =

S_4 =

Pour les sorties S1, S2 et S3, une analyse poussée de la table de vérité permet d'en dégager les équations optimisées. Ces équations contenant des ou-exclusifs, l'utilisation des tableaux de Karnaugh ne serait pas optimale :

S_1 = **S_2 =** **S_3 =**

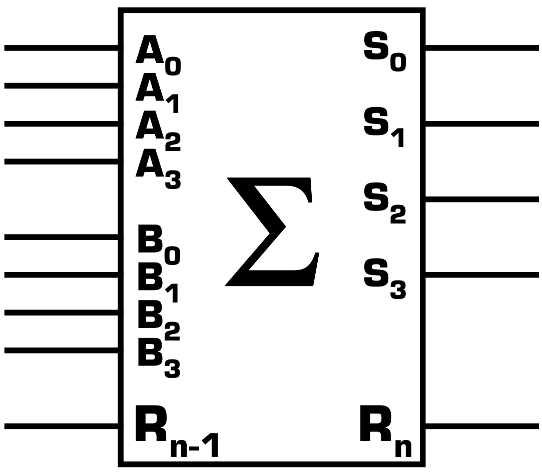
Les tableaux de Karnaugh peuvent permettre d'aboutir à ces équations optimisées, mais à condition de factoriser et de reconnaître les ou-exclusifs …

Logigramme de la fonction « Complément à 2 » :

Fonction complément à 2 d'un nombre de 4 bits

<u>IV – 2 – Solution 2</u>

Une autre solution pour calculer le complément à 2 consiste à effectuer l'opération $\overline{N} + 1$ en utilisant un additionneur 4 bits :



Utilisation d'un additionneur pour calculer le complément à 2

Comparaison des deux solutions : l'additionneur complet 4 bits est composé de 20 portes logiques en interne. Avec les 4 portes NON, la solution 2 coûte 24 portes logique au total (contre 6 portes logiques pour la solution 1). La solution 1 sera donc préférable si on doit réaliser la fonction complément à 2 « à partir de rien », alors que la solution 2 sera mise en œuvre lorsqu'on dispose déjà d'un additionneur complet 4 bits « tout fait ».