L'ADDITION BINAIRE

1. Le principe

Quand vous faites une addition en décimal, vous faites la somme des chiffres se trouvant dans une même colonne.

Si la somme est inférieure à 10, alors vous posez le résultat obtenu et passez à la colonne suivante.

Si la somme est supérieure à 10, alors vous posez le chiffre des unités et gardez en retenue le chiffre des dizaines.

Si vous faites la somme de 2 nombres, alors la retenue ne pourra être supérieure à 1.

Le principe est exactement le même en binaire. Vous faites la somme, posez le chiffre des unités, et retenez le chiffre de la seconde colonne en retenue (qu'il vaut mieux, évidemment, éviter d'appeler les « dizaines »).

Si vous faites la somme de **2 nombres**, alors il n'y a que **4 cas possibles** :

- 0 + 0 = 0, on pose 0
- 0 + 1 = 1, on pose 1
- 1 + 1 = 10, on pose 0 et on retient 1
- 1 + 1 + une retenue de 1 = 11, on pose 1 et on retient 1

Exemple : 19 + 71

Retenue				**1**	**1**	**1**	
	1	**0**	**0**	**0**	**1**	**1**	**1**
+			**1**	**0**	**0**	**1**	**1**
	1	***0***	***1***	***1***	***0***	***1***	***0***

2. **<u>Le semi-additionneur</u>**

Les résultats précédents peuvent être synthétisés dans le tableau suivant :

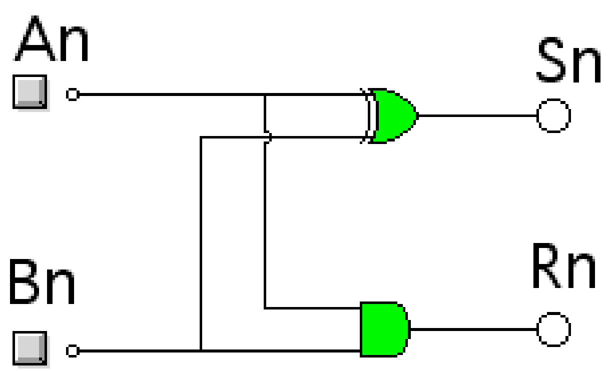
E_1	E_2	S	R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

E_1 et E_2 sont les deux chiffres additionnés.

S est la somme et R la retenue.

On reconnaît dans ce tableau, si l'on considère les colonnes E_1, E_2 et S, la table de vérité d'une porte EXOR (OU-exclusif), et si l'on regarde les colonnes E_1, E_2 et R, la table de vérité d'une porte AND (ET).

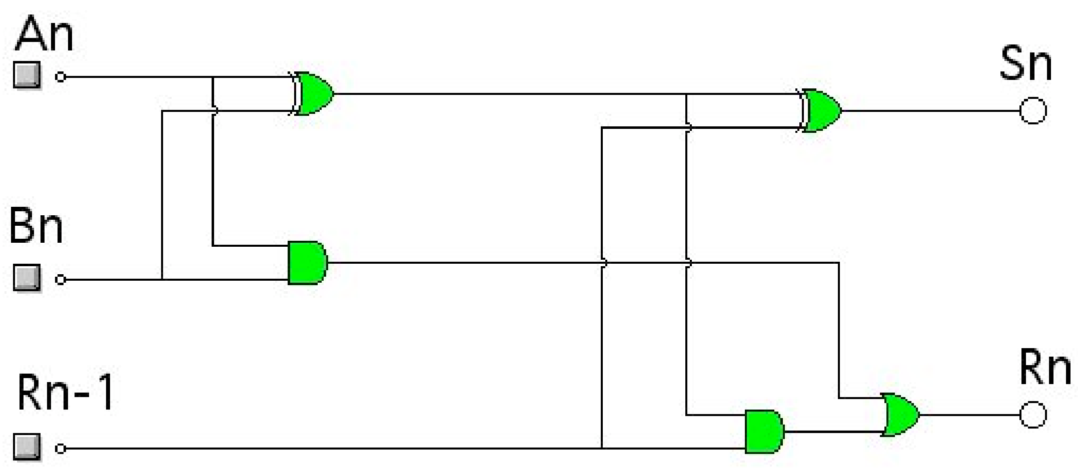
On peut donc simuler et concevoir le semi-additionneur suivant :



3. **<u>L'additionneur complet</u>**

Le montage précédent, le semi-additionneur, ne prend pas en compte une éventuelle retenue provenant de l'addition des chiffres A_{n-1} et B_{n-1}.

Le montage additionneur prend en compte cette retenue :



On obtient alors la table de vérité suivante :

A_n	B_n	R_{n-1}	S_n	R_n
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

On constate que cette table de vérité est bien conforme à ce qui a été expliqué au paragraphe 1.

On constate que l'additioneur fonctionne en **2 temps**.

Dans un premier temps, un semi-additionneur fait la somme des chiffres A_n et B_n. On obtient à la sortie de ce semi-additionneur une somme et une retenue que l'on peut qualifier d'intermédiaires, et noter S_i et R_i.

Intermédiaires, car il faut maintenant, avant d'obtenir le résultat final, prendre en compte **une éventuelle retenue** provenant de l'addition des chiffres précédents, d'où son nom R_{n-1}.

La somme intermédiaire S_i est donc additionnée à la retenue par l'intermédiaire d'une porte EXOR, à la sortie de laquelle on obtient la **somme S_n**.

Cette nouvelle addition peut donner lieu à une retenue, que j'appelle R'_i, obtenue par une porte AND à l'entrée de laquelle on a appliqué S_i et R_{n-1}.

Reste maintenant à obtenir la retenue finale, appelée R_n.
Nous avons obtenu une première retenue, appelée R_i, issue de l'addition de A_n et B_n.
Puis une deuxième retenue, appelée R'_i, issue de l'addition de la somme intermédiaire avec la retenue R_{n-1}.
Il suffit que l'une de ces retenues soit égale à 1 pour que la retenue R_n soit de 1.

Autrement dit, si la première addition avait déjà donné une retenue de 1, alors l'addition de R_{n-1} n'apportera rien de nouveau.
Si la première partie n'avait pas donné de retenue, alors il est possible que l'addition de R_{n-1} en donne une, et dans ce cas, il faudra finalement retenir 1.

S'il suffit que R_i ou R'_i vaille 1 pour que R_n vaille 1, alors il faut utiliser **une porte OU** à l'entrée de laquelle on amène R_i et R'_i.

Le montage du paragraphe 2. est appelé **semi-additionneur** car on constate que l'on associe deux semi-additionneurs pour la conception de l'additionneur complet.

A ce sujet, il est intéressant d'aborder rapidement **des critères d'ordre pratique et économique**.
Il est en effet possible de concevoir un additionneur complet d'une manière « plus simple », c'est à dire utilisant un nombre plus faible de portes que le montage que nous venons d'étudier.
C'est pourtant bien notre montage qui est utilisé dans la pratique.

Pourquoi ? Car il permet de réutiliser le montage semi-additionneur. Une fois le semi-additionneur conçu, il est alors facile de concevoir des circuits intégrés, des « puces », dont le seul rôle sera de faire des « semi-additions ». Notre montage additionneur n'utilise donc finalement que 3 puces, dont 2 identiques. Quand une puce est conçue, il est évidemment rentable de l'utiliser autant que possible.

4. Addition binaire sur 4 bits

On veut additionner un nombre A, s'écrivant $A_3A_2A_1A_0$, et un nombre B, s'écrivant $B_3B_2B_1B_0$ (les A_n et les B_n sont des chiffres qui valent bien entendu 0 ou 1).

Il y a donc 4 additions successives de 2 chiffres à réaliser.
Il faut donc **associer 4 additionneurs**.

La sortie R_{n-1} de l'additionneur n-1 est directement reliée à l'entrée portant le même nom sur l'additionneur suivant.

Seul le premier additionneur n'a pas de retenue éventuelle à prendre en compte. Cette entrée est donc supprimée.

On obtient le montage suivant :