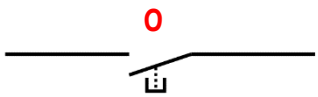
Définition : Une fonction logique est une relation entre une ou plusieurs variables d'entrées et une seule variable de sortie.

Principe :
S'exprime au moyen de deux chiffres, on ne considère que deux états :
- la grandeur physique n'existe pas : 0
- la grandeur physique existe : 1



De nombreux dispositifs électroniques, électromécanique, mécaniques, électriques, pneumatiques,
fonctionnement en tout ou rien. Ceci sous-entend qu'ils peuvent prendre 2 états.
En voici quelques exemples :
Arrêt marche, Ouvert fermé, Enclenché déclenché, Avant arrière, Vrai faux, Conduction blocage.

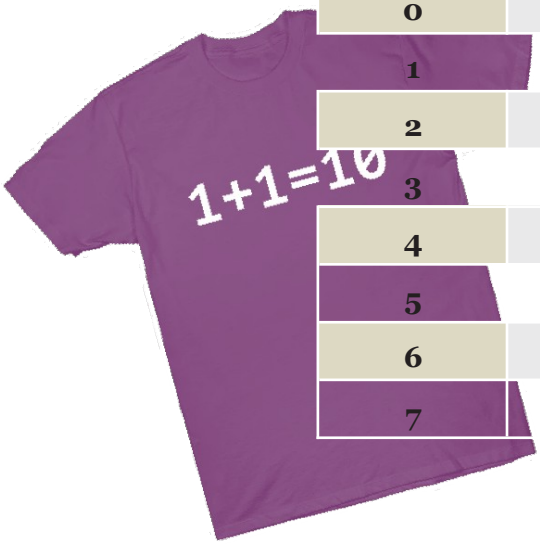
Notion de variable binaire
La variable logique est une grandeur qui peut prendre 2 valeurs qui sont repérées habituellement 0 ou 1.
Cette variable est dite binaire et se note par une lettre comme en algèbre. Exemple : a, b, x
La variable binaire est aussi appelée variable booléenne.

Table de vérité de ET		
a	b	a ET b
0	0	0
0	1	0
1	0	0
1	1	1

Table de vérité
Une fonction X peut comporter n variables.
Pour chacune de ces combinaisons, la fonction peut prendre une valeur 0 ou 1.

De la base 10 à la base 2

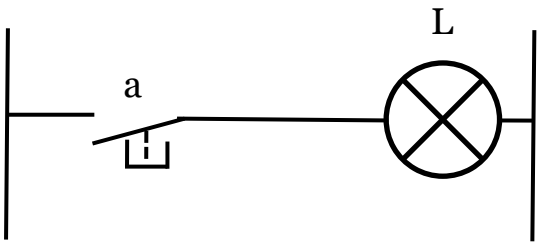
Décimal	Binaire	Décimal	Binaire
0	0000	8	1000
1	0001	9	1001
2	0010	10	1010
3	0011	11	1011
4	0100	12	1100
5	0101	13	1101
6	0110	14	1110
7	0111	15	1111



Les fonctions logiques de base (opérateurs)

1/ La fonction OUI (us: buffer)

- Schéma électrique



- Équation

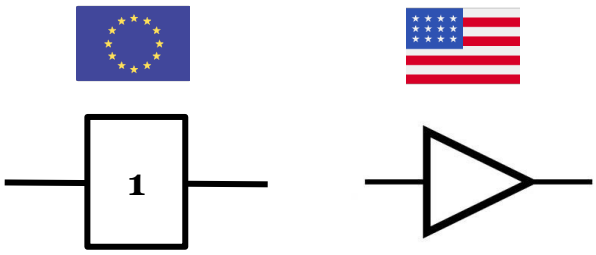
La fonction est représentée par une équation

$$L = a$$

- Table de vérité

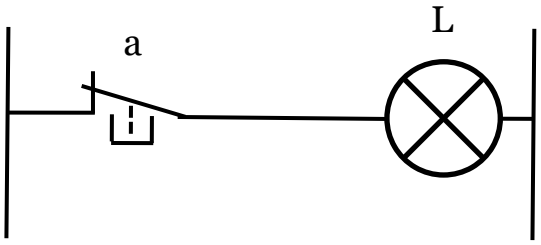
a	L
0	0
1	1

- Symbole logique



2/ La fonction NON (us: NOT)

- Schéma électrique



- Équation

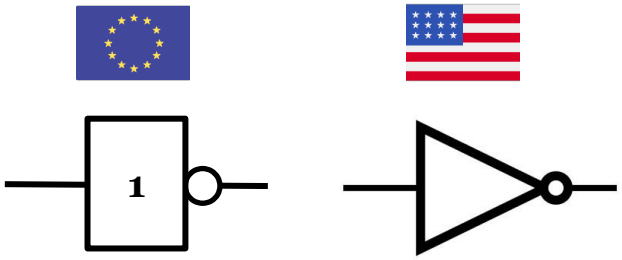
La lampe s'allume quand on n'appuie pas sur a

$$L = \overline{a}$$

- Table de vérité

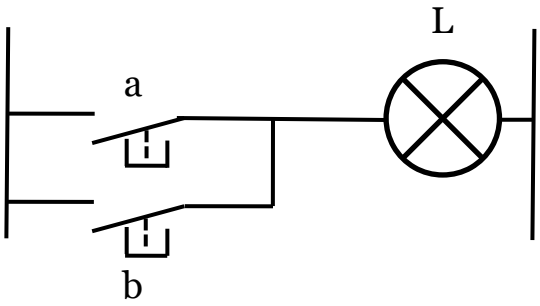
a	L
0	1
1	0

- Symbole logique



3/ La fonction OU (us: OR)

- Schéma électrique



- Équation

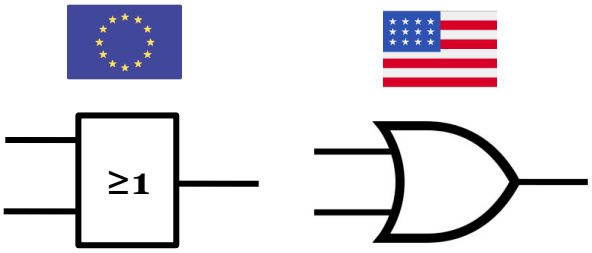
La lampe s'allume si on appuie sur a ou b

$$L = a + b$$

- Table de vérité

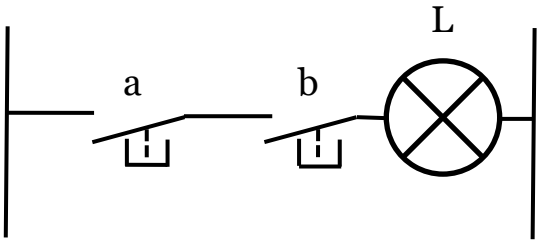
a	b	L
0	0	0
0	1	1
1	0	1
1	1	1

- Symbole logique



4/ La fonction ET (us: AND)

- Schéma électrique



- Équation

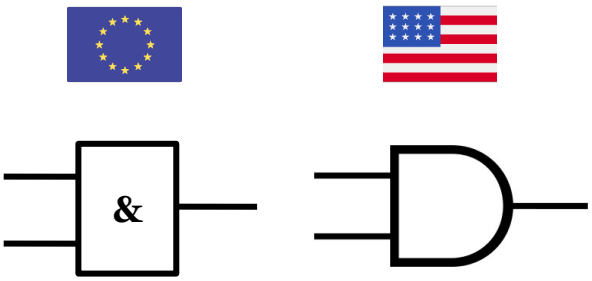
La lampe s'allume si on appuie sur a et sur b

$$L = a \cdot b$$

- Table de vérité

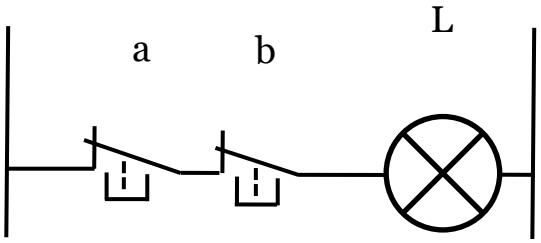
a	b	L
0	0	0
0	1	0
1	0	0
1	1	1

- Symbole logique



5/ La fonction NON OU (us: NOR)

- Schéma électrique



- Équation

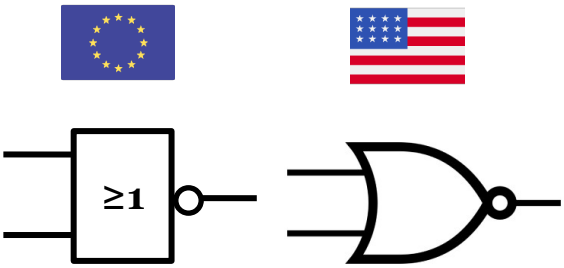
La lampe s'allume si on n'appuie pas ni sur a ou sur b

$$L = \overline{a} \cdot \overline{b} = \overline{a + b}$$

- Table de vérité

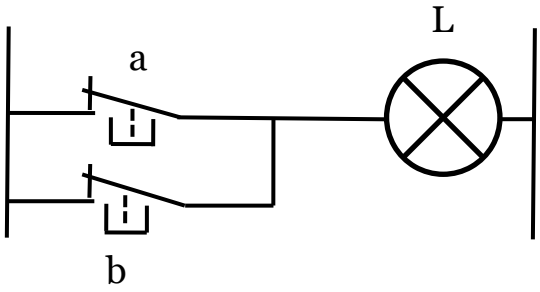
a	b	L
0	0	1
0	1	0
1	0	0
1	1	0

- Symbole logique



6/ La fonction NON ET (us: NAND)

- Schéma électrique



- Équation

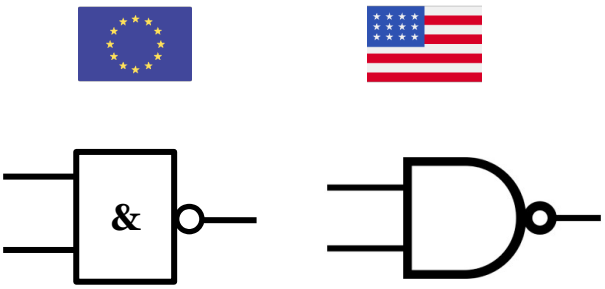
La lampe s'allume si on n'appuie pas sur a et sur b

$$L = \overline{a} + \overline{b} = \overline{a \cdot b}$$

- Table de vérité

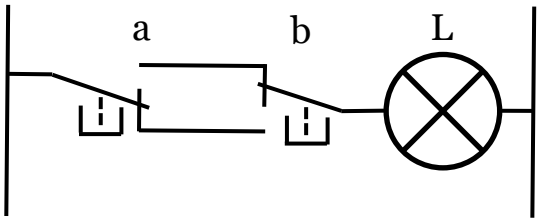
a	b	L
0	0	1
0	1	1
1	0	1
1	1	0

- Symbole logique



7/ La fonction OU EXCLUSIF (us: XOR)

- Schéma électrique



- Équation

La lampe s'allume si on appuie sur a et **pas** sur b **Ou** sur b et **pas** sur a

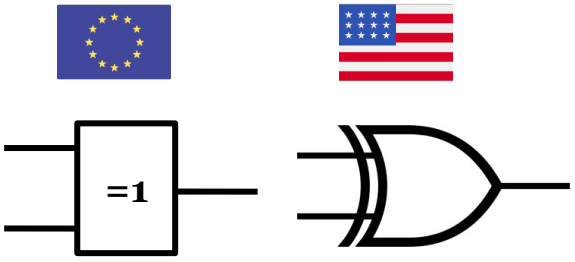
$$L = a \cdot \overline{b} + \overline{a} \cdot b$$

$$a \oplus b$$

- Table de vérité

a	b	L
0	0	0
0	1	1
1	0	1
1	1	0

- Symbole logique



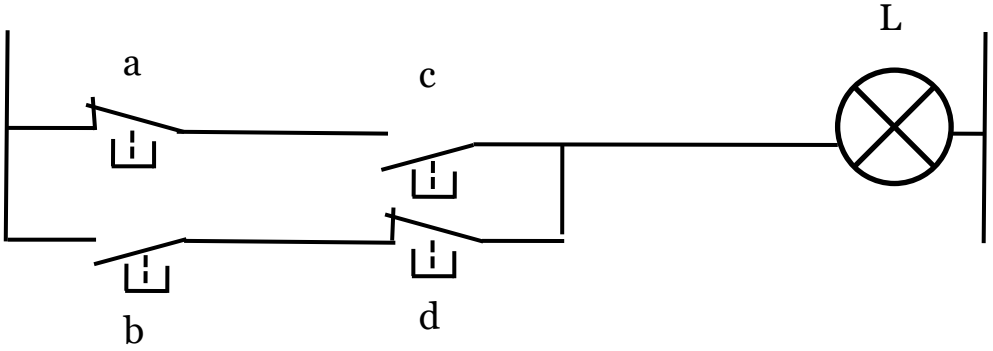
Exercices :

a) Réaliser le schéma électrique venant de cette équation logique :

$$L = (a + b).(c + d)$$

Réaliser ensuite ce schéma avec des symboles logiques

b) Déterminer l'équation logique du schéma suivant :



Système combinatoire

Un système combinatoire est un système avec N entrées et M sorties et les sorties dépendent uniquement de l'état présent des entrées (pas du temps, pas d'effet mémoire).

Sortie 1 = f1(Entrée1, Entrée2,… EntréeN)
Sortie 2 = f2(Entrée1, Entrée2,… EntréeN)
…
Sortie M = fM(Entrée1, Entrée2,… EntréeN)

Différentes représentations possibles :

- Sous forme d'équations logiques utilisant des opérateurs logiques vues précédemment
- Avec un tableau, dit table de vérité, spécifiant l'état des sorties pour toutes les combinaisons d'entrées possibles (N entrées = 2^N possibilités = 2^N lignes)
- Avec un logigramme
- Dans un langage de description (VHDL, Verilog HDL, ladder…)

Représentation externe

Description algorithmique

S=1
si a=O et b=0 et c=0
ou
si a=O et b=1 et c=1
ou
si a=1 et b=1 et c=1
sinon S=0

Table de vérité

a	b	c	S
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Équation logique : Donne la valeur d'une grandeur binaire en fonction de grandeurs également binaires

Utilise les opérateurs logiques de base : ET, OU, NON…

Exemple : $S=\overline{e1.e2}.e3+e0.\overline{e1}+e2$

Peut toujours s'écrire sous la forme d'une somme de MINTERMs (terme produit dans lequel toutes les entrées apparaissent)

$$s=\sum\prod(e_i,\bar{e}_i)$$

- **Exemple d'équation** avec des minterms pour une table de vérité avec 3 entrées (a,b et c) et 1 sortie (s)

$$S=\bar{a}.\bar{b}.\bar{c}+\bar{a}.b.c+a.b.c$$

=> Équation logique sous la forme d'une somme de minterms à partir de la table de vérité

Table de vérité

a	b	c	S
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Vers l'équation logique

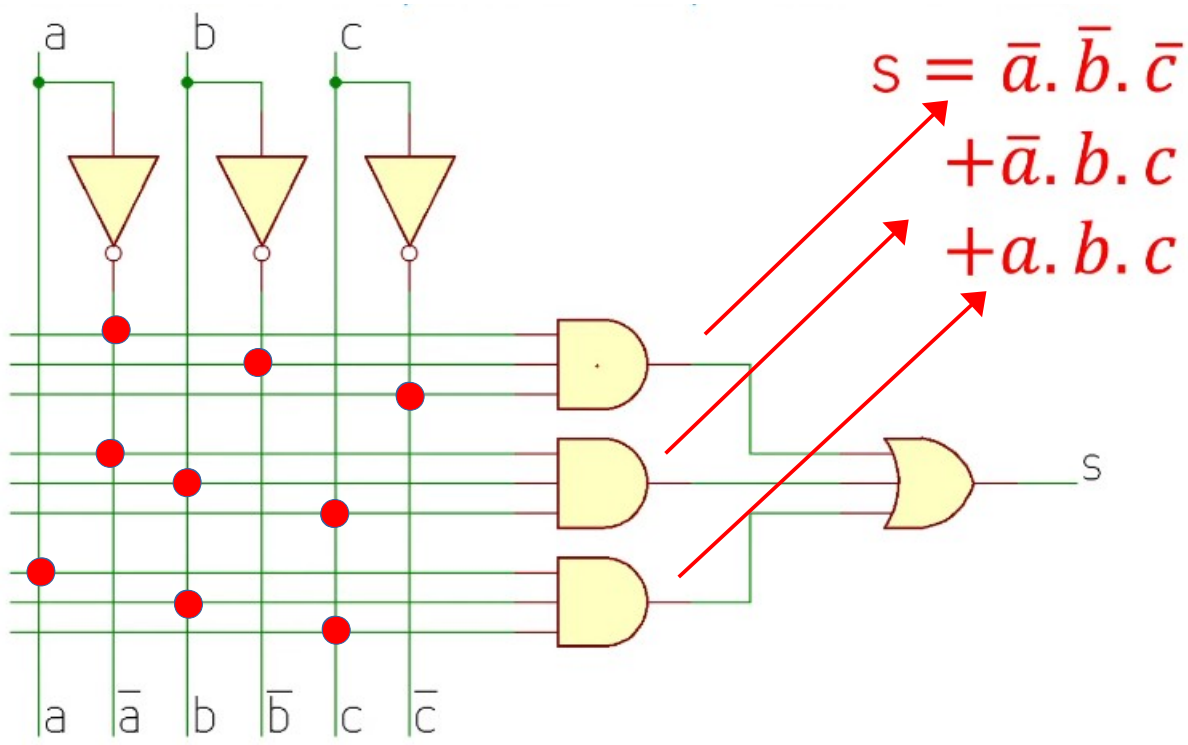
$S=1 si \bar{a}.\bar{b}.\bar{c}$

$S=1 si \bar{a}.b.c$

$S=1 si a.b.c$

$S=\bar{a}.\bar{b}.\bar{c}+\bar{a}.b.c+a.b.c$

Schéma structurel à partir de l'équation



Simplification des équations logiques : différentes méthodes possibles

- Divers algorithmes (méthode de Quine-McClustey…)
- Table de Karnaugh
- Analytique en utilisant des règles de simplification
  - Exemple de règles (liste non exhaustive)

    -Avec les éléments neutres : $a.1=a \; et \; a+0=a$

    -Avec les compléments : $a.\bar{a}=0 \; et \; a+\bar{a}=1$

    -En utilisant les théorèmes de De Morgan : $\overline{a.b}=\bar{a}+\bar{b} \; et \; \overline{a+b}=\bar{a}.\bar{b}$

  - Application de $S=\bar{a}.\bar{b}.\bar{c}+\bar{a}.b.c+a.b.c$

    factorisation : $(\bar{a}+a).b.c$

    $1.b.c=b.c$

donc => $S=\bar{a}.\bar{b}.\bar{c}+b.c$

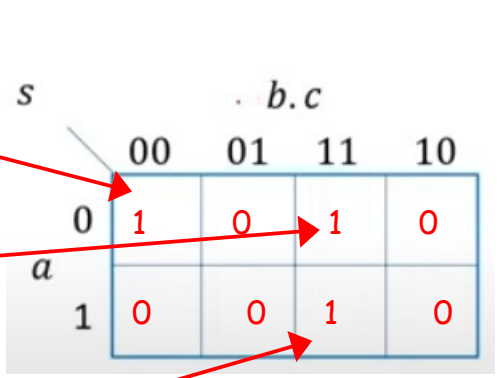
Simplification avec les tables de Karnaugh

Présentation de la table de vérité sous la forme d'un tableau dont les variables d'entrées sont ordonnées en code binaire réfléchie (code Gray)

Codage décimal	Codage binaire naturel	Codage Gray ou binaire réfléchi
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100

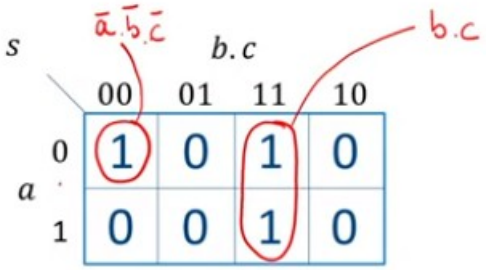
La différence principale entre les deux est le fait que le codage de Gray de deux nombres consécutifs ne diffère que d'une position.

a	b	c	S
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1



Règle de simplification :

- 2^n cases adjacentes contenant des 1 peuvent être regroupées pour donner un terme unique simplifié
- Les entrées prenant les valeurs 0 et 1 sur un regroupement disparaissent de l'équation associé au regroupent $(a+\bar{a}=1)$
- Les produits obtenus du fait de la simplification sont appelés PTERM (termes élaboré avec tout ou partie des variables d'entrées=

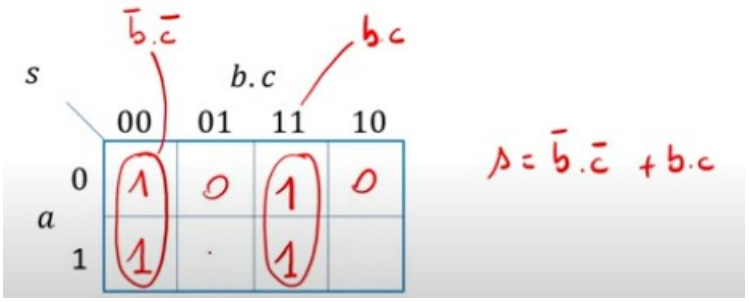


Simplification et « don't care »

Les cases dont la valeur n'est pas imposée (« don't care représenté par un X) peuvent être utilisées avec un 0 ou un 1 pour augmenter les possibilité de simplification.

Exemple :

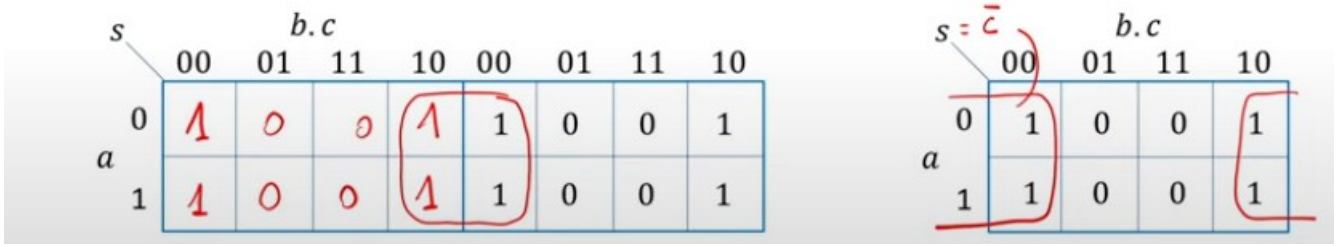
a	b	c	S
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	X
1	0	1	X
1	1	0	X
1	1	1	X



Les regroupements peuvent également se faire de manière circulaire

Exemple :

$$S=\overline{a}.\overline{b}.\overline{c}+\overline{a}.b.\overline{c}+a.\overline{b}.\overline{c}+a.b.c$$



On recopie la même table → $S=\overline{c}$

Cas particuliers

- 4 entrées, soit 16 cases maximum => pour 5 entrées, il faut dresser 2 tableaux de Karnaugh de 16 cases
- Structure en damier = OU exclusif

$$S=a\oplus b\oplus c$$