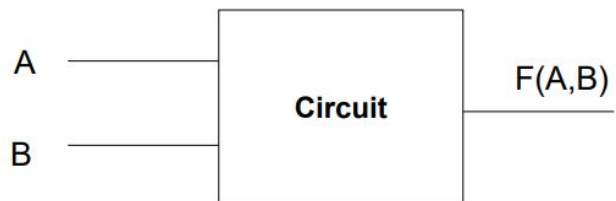
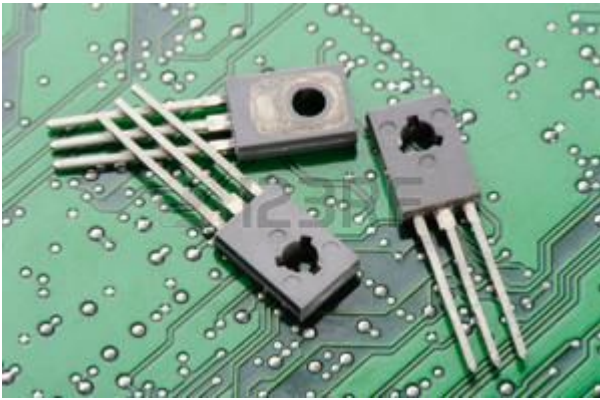


## L6 : Logique et algèbre de Boole

Une machine est composée de plusieurs circuits, eux-mêmes constitués de **transistors**, comme sur le schéma ci-dessous par exemple :



On parle de **circuits logiques binaires** : leurs entrées et sorties se caractérisent exclusivement par deux états : l'**état logique haut** et l'**état logique bas** (ou plus simplement ON et OFF). Ces transistors se comportent comme des interrupteurs.

Ces circuits logiques permettent de réaliser toutes les opérations dans les processeurs des machines (cf. le modèle d'architecture de Von Neumann).

### I. Informations binaires et variables booléennes

Une *variable booléenne* ne peut prendre que deux valeurs, notées **0** et **1**.

Ces variables peuvent servir à constituer une information binaire (oui/non, vrai/faux, allumé/éteint, ..., etc.) ou à décrire l'état physique d'un composant d'un système (alimentation d'un composant, action sur un bouton, ...)

- La valeur 0 représente l'état physique d'un composant **non alimenté**.
- La valeur 1 représente l'état physique d'un composant **alimenté**.

| Langage courant | Tension | État logique | Valeur booléenne | Jour / Nuit   |
|-----------------|---------|--------------|------------------|---|
| Marche          | 5V      | HAUT         | 1                |  |
| Arrêt           | 0V      | BAS          | 0                |  |

Les variables de type booléen sont extrêmement utilisés en informatique notamment pour l'exécution de la fameuse instruction conditionnelle **if ... else**.

## II. Algèbre de Boole

L'algèbre de Boole, ou algèbre binaire, est la partie des mathématiques qui s'intéresse aux opérations et aux fonctions sur les variables logiques. Elle fut inventée par le mathématicien britannique *George Boole*. Aujourd'hui, l'algèbre de Boole trouve de nombreuses applications en informatique et dans la conception des circuits électroniques.

**Historique :** George Boole, né le 2 novembre 1815 à Lincoln (Royaume-Uni) et mort le 8 décembre 1864 à Ballintemple (Irlande), est un logicien, mathématicien et philosophe britannique. Il est le créateur de la logique moderne, fondée sur une structure algébrique et sémantique, que l'on appelle *algèbre de Boole* en son honneur.

### 1. Opérateurs

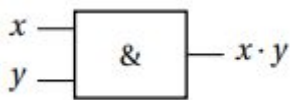
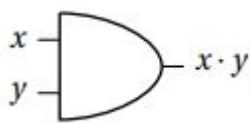
Les calculs sur les variables booléennes sont réalisés grâce à l'algèbre de *Boole* qui comporte **3 opérateurs élémentaires** :

- Opérateur ET

**Définition :** (a ET b) est **VRAI** si et seulement si (a est **VRAI**) et (b est **VRAI**)

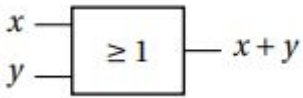
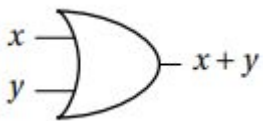
Cette opération logique est aussi notée :

- $a \cdot b$
- $a \wedge b$  (en notation algébrique)
- $a \& b$  ou  $a \&\& b$  (en C++ ou PHP)
- $a \text{ AND } b$  ( en Python)

| Fonction logique | Symbole européen  | Symbole américain   |
|------------------|---|---|
| ET (AND)         |  |  |

- Opérateur OU

**Définition :** (a OU b) est **VRAI** si et seulement si :  
cas 1 : (a est **VRAI**) ou (b est **VRAI**),  
cas 2 : si a et b sont **VRAIS**.


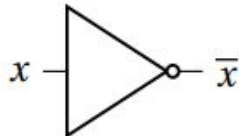
| Fonction logique | Symbole européen  | Symbole américain  |
|------------------|---|--|
| OU (OR)          |  |  |

Cette opération logique est aussi notée :

- $a + b$
- $a \vee b$  (en notation algébrique)
- $a | b$  ou  $a || b$  (en C++ ou PHP)
- $a \text{ OR } b$  ( en Python)

- Opérateur NON

**Définition :** Le contraire de « a » est **VRAI** si et seulement si a est **FAUX**.

| Fonction logique | Symbole européen  | Symbole américain  |
|------------------|---|--|
| NON (NOT)        |  |  |

L'opération logique NON est notée :

- $\bar{a}$  (se lit "a barre")
- $\neg a$  (en notation algébrique)
- $!a$  (en C ou C++)
- $\text{NOT } a$  (en Python)



## 2. Fonctions logiques et tables de vérité

**Définition :** Une **table de vérité** est un tableau qui représente des entrées (en colonne) et des états binaires (0 et 1). Le résultat, exprimé lui aussi sous forme binaire, se lit dans la dernière colonne.

### Exercice :

Compléter les tables de vérité des opérateurs logiques ci-dessous :

#### OU (OR)

| a | b | a OU b |
|---|---|--------|
| 0 | 0 |        |
| 0 | 1 |        |
| 1 | 0 |        |
| 1 | 1 |        |

#### ET (AND)

| a | b | a ET b |
|---|---|--------|
| 0 | 0 |        |
| 0 | 1 |        |
| 1 | 0 |        |
| 1 | 1 |        |

#### NON (NOT)

| a | NON a |
|---|-------|
| 0 |       |
| 1 |       |

### 3. Propriétés

- **Priorité**

Pour faciliter leur compréhension, il a été décidé que ces opérations logiques seraient soumises aux mêmes règles que les opérations mathématiques :

- La fonction ET (produit logique) est prioritaire par rapport à la fonction OU (somme logique)
- On peut placer des parenthèses pour mettre des priorités dans les opérations.

- **Associativité**

Comme avec les opérations habituelles, certaines parenthèses sont inutiles :

$$(a + b) + c = a + (b + c) = a + b + c$$

$$(a \cdot b) \cdot c = a \cdot (b \cdot c) = a \cdot b \cdot c$$

- **Commutativité**

L'ordre est sans importance :

$$a + b = b + a$$

$$a \cdot b = b \cdot a$$

- **Distributivité**

$$a \cdot (b + c) = (a \cdot b) + (a \cdot c) = a \cdot b + a \cdot c$$

Il y a distributivité de la multiplication, comme en arithmétique.

$$a + (b \cdot c) = (a + b) \cdot (a + c)$$



**En logique, il y a distributivité de l'addition (ce qui est faux en arithmétique !).**

- **Théorèmes de DE MORGAN**

☐ 1ère loi :  $\overline{a + b} = \bar{a} \cdot \bar{b}$

☐ 2ème loi :  $\overline{a \cdot b} = \bar{a} + \bar{b}$