Archi 1 – **Les portes logiques**

I – Rappel de logique

Nous avons vu les opérateurs logiques suivants :
→ l'opérateur NON a qui inverse la réponse (a : Vrai → NON a : Faux)
→ l'opérateur **a ET b** : **Vrai uniquement si tout est Vrai.**
→ l'opérateur **a NAND b** : **Faux** uniquement si tout est Vrai
→ l'opérateur **a OU b : Faux uniquement si tout est Faux.**
→ l'opérateur **a NOR b** : **Vrai** uniquement si tout est Faux
→ l'opérateur **a XOR b** : Vrai si uniquement l'un des deux est Vrai.

L'**algèbre de Boole** permet de transformer la réflexion logique en simple calcul arithmétique :

NOT a devient **1-a**, noté $\overline{\mathbf{a}}$

a AND b devient **a.b**

a NAND b devient $\overline{\mathbf{a.b}} = \overline{\mathbf{a}} + \overline{\mathbf{b}}$ (De Morgan)

a OR b devient **a+b**

a NOR b devient $\overline{\mathbf{a+b}} = \overline{\mathbf{a}} \,.\, \overline{\mathbf{b}}$ (De Morgan)

a XOR b devient $\mathbf{a.\overline{b} + \overline{a}.b}$

II – Un peu d’électricité

Un **courant électrique** est dû à un mouvement global d'électrons.

L’**intensité électrique** I permet de caractériser l’importance du courant électrique. Elle se mesure en Ampère (A)

La **tension électrique** U_{AB} entre les points A et B comme une mesure de la différence de charges électriques entre le point A et le point B. Elle se mesure en Volt.

La **résistance électrique** R en Ohm (Ω) permet de caractériser l’opposition au passage du courant avec dissipation d’énergie sous forme de chaleur.

Loi d’Ohm sur les conducteurs ohmiques : la loi d'Ohm : **U = R . I**

III – Transistor (voir site pour les détails : Hors programme)

Le transistor constitue le composant électrique élémentaire des unités informatiques.

C’est globalement un composant à trois bornes :
→ l’une des bornes A permet de commander le passage du courant entre
→ les bornes B et C : interrupteur fermé ou interrupteur ouvert.

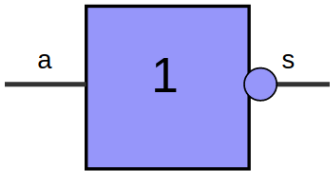
Aujourd'hui, les transistors sont de tailles microscopiques. C'est en réduisant leurs tailles qu'on augmente leur nombre et donc la vitesse de calcul des ordinateurs.

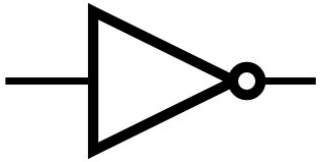
Nous atteignons aujourd'hui la taille des atomes. Il va donc falloir trouver des nouvelles façons d'augmenter cette vitesse si on veut continuer à ne pas faire mentir la loi de Moore (Puissance double tous les 2 ans environ).

IV – Porte NOT (NON)

On peut construire une porte NON en utilisant
→ un transistor et un résistor ou
→ un transistor CMOS.

Symboles :





On notera que **NON(NON(a)) = a**

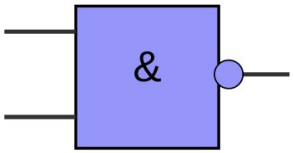
En utilisant 8*X portes logiques NON, l'UAL peut donc calculer le complémentaire de X octets en temps constant.

V – Porte NAND(NON-ET) et AND(ET)

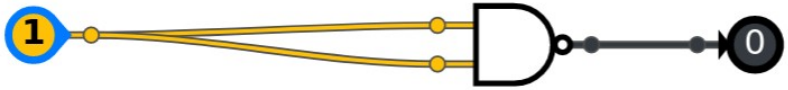
1) NAND

On peut construire une porte **NAND** en utilisant
→ deux transistors en série et un résistor ou
→ deux transistors CMOS.

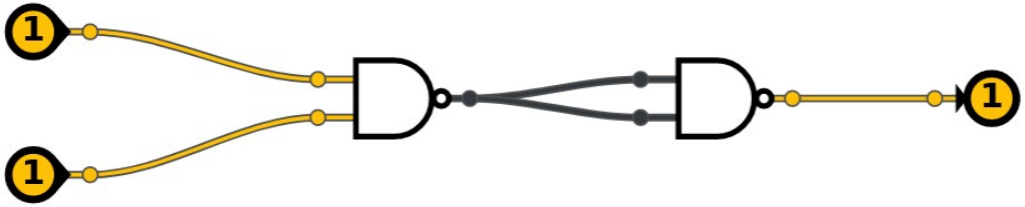
Symboles :



On peut utiliser des portes NAND pour créer toutes les autres portes.
Exemple :
→ une porte NOT



→ une porte AND



→ Une porte AND…

Remarque : c'est le rond sur la sortie qui indique une inversion (NON-ET).
Remarque : L'entrée du AND/NAND correspond à une barre droite.

2) AND

On peut construire une porte **AND** en utilisant
→ deux transistors en série et un résistor ou
→ trois transistors CMOS.

Symboles :

En utilisant des portes logiques AND, l'UAL peut savoir si :
→ un groupe d'octets est à son **maximum** (rempli de 1)
→ deux bits à la même position sont à 1.

VI – Porte NOR(NON-OU) et OR(OU) et XOR (OU exclusif)

1) NOR

On peut construire une porte **NOR** en utilisant
→ deux transistors en parallèle et un résistor ou
→ deux transistors CMOS.

Symboles :

Remarque : c'est le rond sur la sortie qui indique une inversion (NON-OU).
Remarque : L'entrée du OR/NOR correspond à une courbe.
Remarque : Le ≥ 1 fait référence à l'algèbre de Boole pour le OU.

On peut utiliser des portes NOR pour créer toutes les autres portes :
→ une porte NOT → une porte OR
→ une porte ET...

2) OR

On peut construire une porte **OR** en utilisant
→ deux transistors en parallèle et un résistor ou
→ trois transistors CMOS.

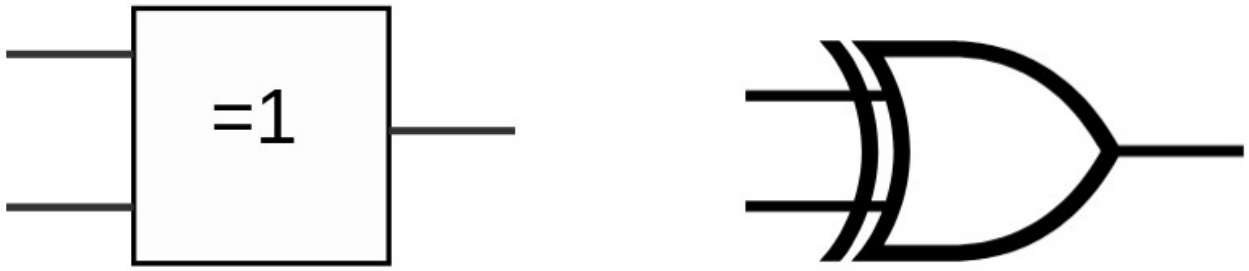
Symboles :

En utilisant des portes logiques OR, l'UAL peut savoir si :
→ un groupe d'octets est à **son minimum** (rempli de 0)
→ deux bits à la même position sont à 0.

3) XOR

On peut créer une porte **XOR** en utilisant des transistors.

Voici les symboles :



Remarque : le =1 exprime bien qu'ici on ne veut pas les deux à 1.

En utilisant des portes XOR, on peut savoir (à temps constant) :
→ si un groupe d'octets comporte un nombre impair de 1.
→ si deux bits sont **différents (!=)** (réponse 1) ou **identiques ==** (0)
→ et donc si deux octets sont identiques ou différents.

Le XOR est très utilisé dans le cadre du chiffrement symétrique (Tle NSI).

Exemple : on chiffre avant d'émettre

```
m Message   0 1 0 0 1 1 0 0 1 1   → le vrai message
c Clé       1 0 1 1 0 1 1 0 1 0
-------------------------------
m XOR c     1 1 1 1 1 0 1 0 0 1   → le message chiffré transmis
                                      sur Internet
```

On déchiffre à la réception (si on connaît la clé!)

```
mc chiffré  1 1 1 1 1 0 1 0 0 1
c Clé       1 0 1 1 0 1 1 0 1 0
-------------------------------
mc XOR c    0 1 0 0 1 1 0 0 1 1       → On retrouve le message
                                          de départ
```