1. Du transistor au micro-processeur

Une machine ne comprend que le **langage binaire**. Un langage de programmation, compréhensible par un humain, est traduit par un compilateur, un assembleur ou un interpréteur, et permet de donner des instructions à la machine. Le fonctionnement d'une machine repose sur des **circuits électroniques**. Nous chercherons dans ce cours à comprendre les liens qui existent entre ces circuits, le calcul logique et le calcul binaire.

Le transistor

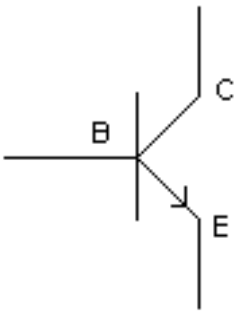
On entend souvent dire que "*un ordinateur utilise uniquement des 1 et des 0*". Cette affirmation mérite d'être précisée.

À la base de la plupart des composants d'un ordinateur, on retrouve le transistor. Ce composant électronique a été inventé fin 1947. L'invention du transistor a été un immense progrès, mais les premiers ordinateurs sont antérieurs à cette invention. En effet, ces premiers ordinateurs, par exemple le Colossus qui date de 1943, étaient conçus à base de tubes électroniques (on parle aussi de tubes à vide) qui, bien que beaucoup plus gros et beaucoup moins fiable que les transistors fonctionnent sur le même principe que ce dernier.

Un transistor

Un tube électronique

Schématiquement un transistor se représente de la manière suivante :



B : Base du transistor

C :Collecteur du transistor

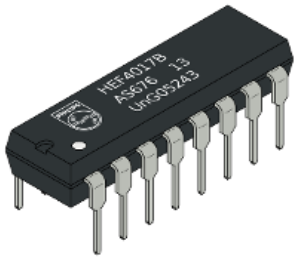
E : Emetteur du transistor

Le transistor est équivalent à un **interrupteur** fermé ou ouvert.

Lorsqu'on injecte un courant suffisant dans la base B , le transistor est dit « saturé », c-a-d qu'il se comporte comme un interrupteur fermé : le courant de puissance va pouvoir circuler du collecteur vers l'émetteur. Sinon (pas de courant dans la base) alors le transistor est dit « bloqué » , c-a-d qu'il se comporte comme un interrupteur ouvert : le courant de puissance ne peux pas circuler

Une tension électrique appliquée sur ces broches peut représenter soit 0 soit 1. Parmi ces trois broches, deux servent à la circulation du courant, la troisième commande cette circulation. C'est le même principe qu'un interrupteur commandé par la troisième broche : le courant passe ou ne passe pas entre les deux premières broches selon l'état de la troisième.

Remarquons que dans nos ordinateurs, on ne trouve plus un transistor tout seul avec ses trois broches, les transistors sont regroupés au sein de ce que l'on appelle des **circuits intégrés**. Dans un circuit intégré, les transistors sont gravés sur des plaques de **silicium**, les connexions entre les centaines de millions de transistors qui composent un circuit intégré sont, elles aussi, gravées directement dans le silicium. Le silicium est un matériau semi-conducteur dont la production pose des problèmes du point de vue écologique



Un circuit intégré

Les circuits logiques

Le transistor est l'élément de base des circuits logiques. Un circuit logique permet de réaliser une opération booléenne. Ces opérations booléennes sont directement liées à l'algèbre de Boole (Voir cours 6 et TP6 sur la représentation des booléens). Un circuit logique prend en entrée un ou des signaux électriques (chaque entrée est dans un état "haut" symbolisé par un "1" ou à un état "bas" symbolisé par un "0") et donne en sortie un ou des signaux électriques (chaque sortie est aussi dans un état "haut" ou à un état "bas").

Une porte logique est définie par la composition de ces circuits. Par exemple :

- La porte logique NOT par un circuit inverseur (composé de deux transistors) ;
- La porte logique OR par deux transistors en parallèle ;
- La porte logique AND par deux transistors en série (traduit le produit).
- La porte XOR est plus complexe à réaliser.

On trouve aussi d'autres portes logiques de base comme NOR, NAND et XNOR.

- NOR signifie NOT OR, soit : a NOR b = NOT (a OR b).
- NAND signifie NOT AND, soit : a NAND b = NOT (a AND b).
- La troisième, XNOR, signifie NOT XOR, soit : a XNOR b = NOT (a XOR b).

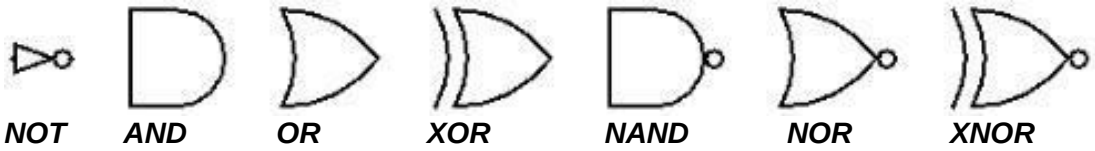
A noter que tous les circuits logiques peuvent être réalisés en utilisant uniquement des portes NAND !

Le site NandGame propose d'appréhender ce sujet en composant un ordinateur complet à partir de portes NAND.

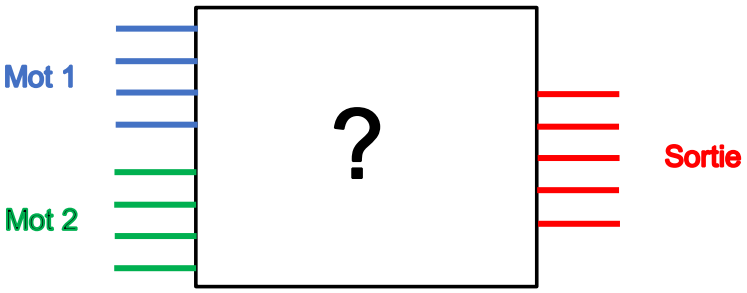
https://nandgame.com

Circuit additionneur

Les représentations graphiques des portes logiques sont présentées ci-dessous :



Un **circuit additionneur** est un circuit qui admet en entrée deux mots de n bits, et fournit en sortie le résultat de l'addition binaire des deux mots d'entrée, sur (n + 1) bits.



Exemple : additionneur de deux mots de 4 bits qui fournit en sortie un mot de 5 bits

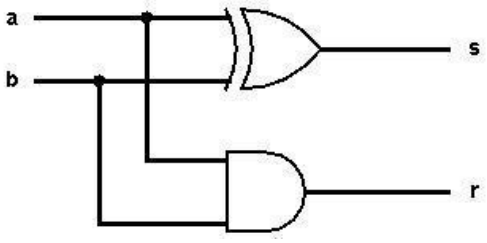
C'est un circuit complexe et pour le construire, il faut passer par une première étape en construisant un circuit plus élémentaire : un **demi-additionneur 1 bit**, c'est-à-dire un circuit à 2 entrées (les deux nombres) et 2 sorties (le résultat et la retenue) qui réalise l'addition binaire de deux nombres à 1 bit.

On associe 0 à faux et 1 à vrai. Si on note les deux entrées a et b et les deux sorties s et r (somme et retenue). Avec la notation **a + b = r s** on aura :

1 + 0 = 0 1 : la sortie s vaut 1 et la retenue r vaut 0

1 + 1 = 1 0 : s vaut 0 et r vaut 1.

La **somme** peut être réalisée avec une porte **XOR** et la **retenue** avec une porte **AND**, ainsi le demi-additionneur 1 bit peut être construit de la manière suivante :

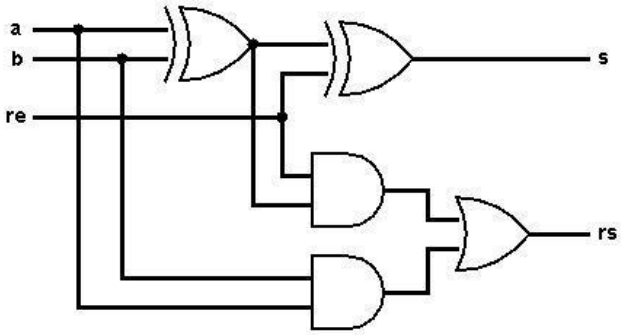


Vérifions cela dans une table de vérité :

		r	s
a	**b**	**a AND b**	**a XOR b**
1	1		
1	0		
0	1		
0	0		

Pour passer à un additionneur **complet** 1 bit, nous devons prendre en compte la retenue qui se propage et cela nécessite trois entrées **a**, **b**, **re** et deux sorties **s** et **rs**, où re est la retenue en entrée et rs la retenue en sortie.

Un circuit additionneur un bit avec retenue en entrée et retenue en sortie, donc trois entrées et deux sorties, peut se représenter à l'aide du schéma suivant :



Nous avons les résultats suivants pour s et rs :

- s = (a XOR b) XOR re
- rs = (a AND b) OR ((a XOR b) AND re)

Vérifions cela dans une table de vérité :

				s			**r**
a	**b**	**re**	**a XOR b**	**(a XOR b) XOR re**	**a AND b**	**(a XOR b) AND re**	**(a AND b) OR ((a XOR b) AND re)**
1	1	0					
1	1	1					
1	0	0					
1	0	1					
0	1	0					
0	1	1					
0	0	0					
0	0	1					

L'utilisation d'un simulateur offre la possibilité de concevoir une représentation effective des circuits mais aussi de les voir fonctionner.

Le logiciel Logisim par exemple permet de construire des circuits à l'aide de portes logiques et de les faire fonctionner en gérant les entrées et les sorties. http://www.cburch.com/logisim

Le site https://logic.ly/demo permet également de le faire directement en ligne.

Travail :

- Sur ce dernier, reproduisez les deux circuits précédents et vérifiez les tables de vérité.
- En utilisant plusieurs additionneurs 1bit, créez un additionneur 4 bits

Notes :

- Utilisez l'objet « **Toggle Switch** » pour materialiser les entrées
- Utilisez l'objet « **Light Bulb** » pour matérialiser les sorties