ARCHITECTURE Modèle d'architecture de von Neumann

On entend souvent dire qu'"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 par les Américains John Bardeen, William Shockley et Walter Brattain. L'invention du transistor a été un immense progrès, mais les premiers ordinateurs sont antérieurs à cette invention. En effet, ces premiers ordinateurs étaient conçus à base de tubes électroniques qui, bien que beaucoup plus gros et beaucoup moins fiable que les transistors fonctionnent sur le même principe.

un transistor

Autre aspect historique qu'il est important de préciser : on ne trouve plus, depuis quelque temps déjà, de transistors en tant que composant électronique discret (comme le transistor de la photo ci-dessus). Dans un ordinateur, 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 millions de transistors qui composent un circuit intégré sont, elles aussi, gravées directement dans le silicium. Les processus de ces gravures dépassent largement le cadre de ce cours, si ce sujet vous intéresse, je vous invite à visionner cette vidéo : https://www.youtube.com/watch?v=NFr-WyytNfo

un circuit intégré

un tube électronique

Il n'est pas question de nous pencher en détail sur le fonctionnement d'un transistor, mais vous devez tout de même savoir que dans un ordinateur **les transistors se comportent comme des interrupteurs : soit le transistor laisse passer le courant électrique (interrupteur fermé), soit il ne le laisse pas passer (interrupteur ouvert)**. Et c'est tout, il n'y a pas d'autre état possible pour un transistor dans un ordinateur : le courant passe ou le courant ne passe pas. Globalement l'ordinateur fonctionne uniquement avec deux états. On parle d'un **état "haut" et d'un état "bas**". On symbolise souvent l'état "haut" par le chiffre "1" et l'état "bas" par le chiffre "0", mais il faut bien avoir conscience qu'il n'y a pas dans un ordinateur des "petits 1" ou des "petits 0" qui se "baladent", c'est juste une histoire de "courant qui passe" ou de "courant qui ne passe pas". On travaille donc uniquement avec 2 chiffres, voilà pourquoi un ordinateur travaille en base 2 (en binaire) et non pas en base 10 comme dans la vie courante.

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 (Georges Boole, mathématicien Britanique 1815-1864). L'étude de l'algèbre de Boole dépasse le cadre de ce cours, vous devez juste savoir qu'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"). Il existe deux catégories de circuit logique :

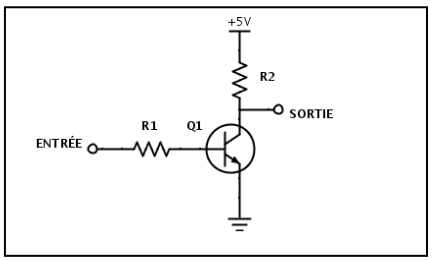
Les portes logiques

Le plus simple des circuits combinatoires est **la porte "NON"** ("NOT" en anglais) qui inverse l'état en entrée : si l'entrée de la porte est dans un état "bas" alors la sortie sera dans un état "haut" et vice versa. Si on symbolise l'état "haut" par un "1" et l'état "bas" pour un "0", on peut obtenir ce que l'on appelle la table de vérité de la porte "NON" :

E (Entrée)	S (Sortie)
1	0
0	1

Equation : S = /E

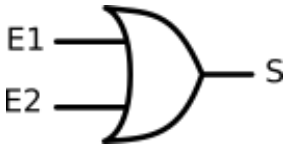
porte "NON"



La porte "OU" (or) a deux entrées (E1 et E2) et une sortie S

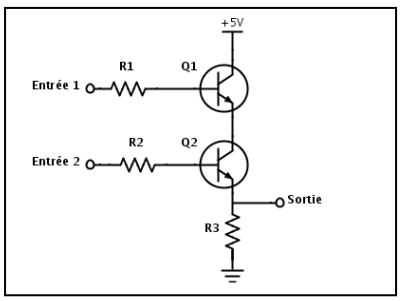
Table de vérité porte "OU" :

E1	E2	S
0	0	0
0	1	1
1	0	1
1	1	1



porte "OU"

Equation : S = E1 + E2

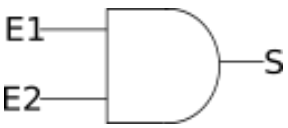


La porte "ET"

("AND") a deux entrées (E1 et E2) et une sortie S

Table de vérité porte "ET" :

E1	E2	S
0	0	0
0	1	0
1	0	0
1	1	1



porte "ET"

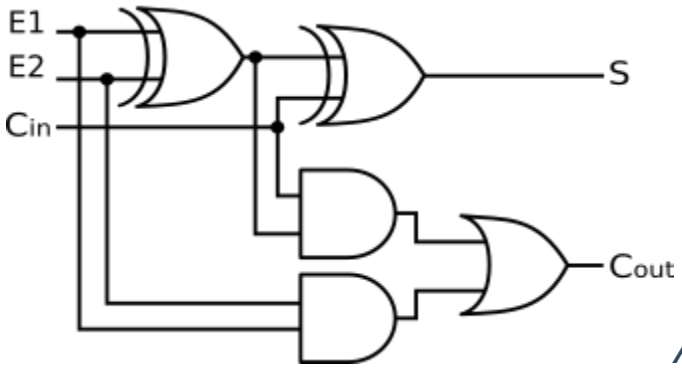
Equation : S = E1 . E2

La porte "OU EXCLUSIF" ("XOR") a deux entrées (E1 et E2) et une sortie S

Table de vérité porte "XOR" :

E1	E2	S
0	0	0
0	1	1
1	0	1
1	1	0

En combinant les portes logiques, on obtient des circuits plus complexes. Par exemple en combinant 2 portes "OU EXCLUSIF", 2 portes "ET" et une porte "OU" on obtient un additionneur :



Additionneur

Comme son nom l'indique, l'additionneur permet d'additionner 2 bits (E1 et E2) en tenant compte de la retenue entrante ("Cin" "carry in" en anglais). En sortie on obtient le résultat de l'addition (S) et la retenue sortante ("Cout"). Quelque part, on commence ici à fabriquer une partie d'un microprocesseur…

Exercice :

Établir la table de vérité de l'additionneur en complétant le tableau ci-dessous

E1	E2	Cin	Cout	S
0	0	0	0	0
0	0	1	0	1
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

En combinant plusieurs fois le type de circuit décrit ci-dessus, on obtient des additionneurs capables d'additionner des nombres sur X bits.

Une chose est très importante à bien comprendre : à la base nous avons le transistor, une combinaison de transistor (sous forme de circuit intégré) permet d'obtenir des circuits logiques, la combinaison de circuits logiques permet d'obtenir des circuits plus complexes (exemple : l'additionneur), et ainsi de suite...

Au sommet de cet édifice nous allons trouver la mémoire vive (RAM) et le microprocesseur (CPU).

La mémoire vive RAM (Random Acess Memory)

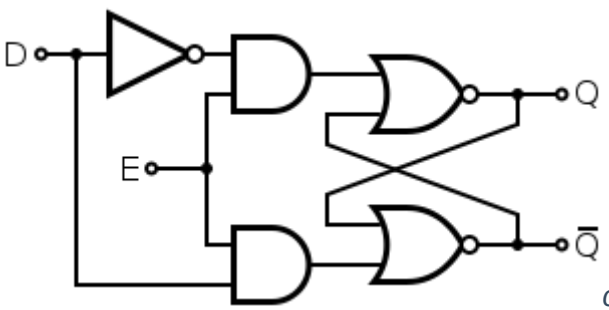
La mémoire vive permet de **stocker des données et des programmes**. Comme nous l'avons vu, l'ordinateur utilise uniquement 2 états, la mémoire va donc stocker les données sous forme de bits (0 ou 1), mais encore une fois, il ne faut pas s'imaginer que la mémoire est pleine de "petit 0" et de "petit 1", ce sont des "états électriques" qui sont stockés dans cette mémoire.

La mémoire ne gère pas les bits 1 par 1, mais 8 par 8, la mémoire gère donc des octets (rappel : 1 octet = 8 bits)

On peut se représenter la mémoire comme une série de cellules, chaque cellule étant capable de stocker 1 octet. **Chacune de ces cellules possède une adresse**. Les opérations sur la mémoire sont de 2 types : **lecture / écriture**. Une opération de lecture consiste à aller lire l'octet situé à l'adresse mémoire XXXXX (ces adresses mémoire étant bien évidemment codées en binaire) et une opération d'écriture consiste à écrire un octet donné à l'adresse mémoire YYYYY. Cette notion **d'adresse mémoire** est fondamentale.

Toujours sur l'aspect technologique, 1 bit d'une cellule est l'association d'un transistor et d'un condensateur. Un condensateur est un composant électronique qui peut être soit chargé (on stocke alors un "1"), soit déchargé (on stocke alors un "0"). Un condensateur n'est pas capable de conserver sa charge pendant très longtemps, il doit donc être alimenté électriquement parlant afin de conserver cette charge. Voilà pourquoi la mémoire vive est une mémoire volatile : **toutes les données présentes en mémoire sont perdues en cas de coupure de courant.** Pour conserver les données une fois l'ordinateur éteint, il faut faire appel à d'autres types de mémoire : les mémoires de stockage. Le disque dur est aujourd'hui la mémoire de stockage la plus utilisé (au moins dans les usages "familiaux"). Un disque dur n'a pas besoin d'alimentation électrique pour conserver les données.

Pour terminer sur cet aspect technologique, il faut noter que l'on trouve aussi des mémoires vives qui stockent l'information grâce à un circuit dit de type "**bascule".** Ce circuit est une combinaison de plusieurs portes logiques.



circuit de type bascule, permet de stocker 1 bit

Le microprocesseur CPU (Central Processing Unit)

Le microprocesseur est le "cœur" d'un ordinateur : les instructions sont exécutées au niveau du CPU. Il est schématiquement constitué de **3 parties** :

L'unité arithmétique et logique (UAL ou ALU en anglais) est chargée de l'exécution de tous les calculs que peut réaliser le microprocesseur. Nous allons retrouver dans cette UAL des circuits comme l'additionneur (voir plus haut)

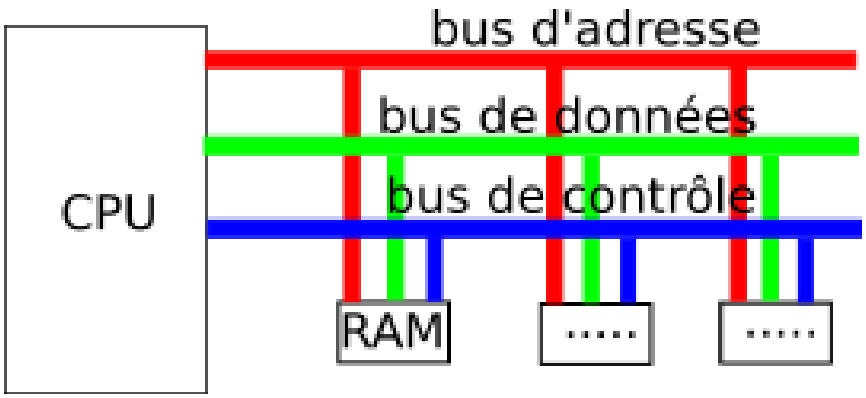
les registres (R0, R1, R2, ...) permettent de mémoriser de l'information (donnée ou instruction) au sein même du CPU. Leur nombre et leur taille sont variables en fonction du type de microprocesseur. Les registres sont aussi appelé des **accumulateurs**.

L'unité de commande permet d'exécuter les instructions (les programmes)

Le bus

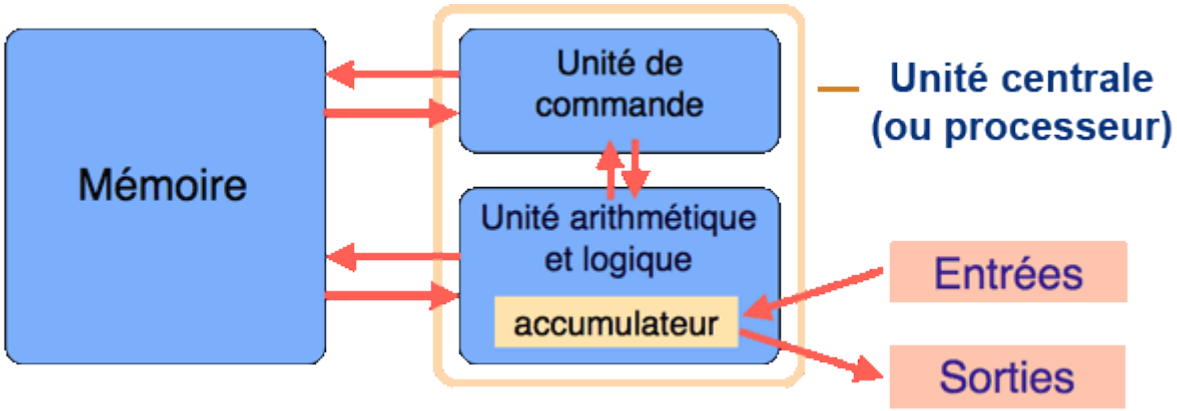
les données doivent circuler entre les différentes parties d'un ordinateur, notamment entre la mémoire vive et le CPU. Le système permettant cette circulation est appelé **bus**. Il existe, sans entrer dans les détails, 3 grands types de bus :

- Le **bus d'adresse** permet de faire circuler des adresses (par exemple l'adresse d'une donnée à aller chercher en mémoire)
- Le **bus de données** permet de faire circuler des données
- Le **bus de contrôle** permet de spécifier le type d'action (exemples : écriture d'une donnée en mémoire, lecture d'une donnée en mémoire).



Nous avons en RAM aussi bien les données (variables, etc) et les programmes ! C'est **John von Neumann** (mathématicien et physicien américano-hongrois 1903-1957) qui a eu l'idée en **1945** d'utiliser une structure de stockage unique pour les données et les instructions, voilà pourquoi on parle d'architecture de von Neumann.

Voici un schéma qui représente ce **modèle de von Neumann** :



Sur ce schéma, nous avons :

- La **mémoire** qui correspond à la RAM vue ci-dessus
- **L'unité arithmétique et logique** qui correspond à l'UAL vu ci-dessus
- **L'unité de commande** qui gère l'exécution des instructions machines À noter que cette unité de commande est aussi parfois appelée "unité de contrôle"
- **Le système "entrée-sortie"** qui permet de communiquer avec "le monde extérieur" au système CPU+RAM (clavier, souris, écran, carte graphique, disque dur...)
- Les **accumulateurs (ou registres)** qui sont une minuscule mémoire interne au processeur. l'accumulateur est un registre permettant de stocker les résultats intermédiaires lors d'un calcul)

Encore aujourd'hui, tous les ordinateurs fonctionnent sur ce principe défini par von Neumann en 1945

John von Neumann