L7 : Architecture de Von Neumann

L'**architecture des ordinateurs** décrit le principe général de fonctionnement de l'ordinateur, pour permettre à un programmeur de l'utiliser au travers du **langage machine**.

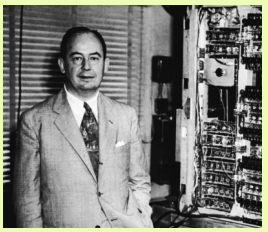
Définir une architecture, c'est définir tous les attributs d'un ordinateur qui sont visibles au programmeur lorsqu'il programme en langage machine.

Citons quatre attributs définissant une architecture :

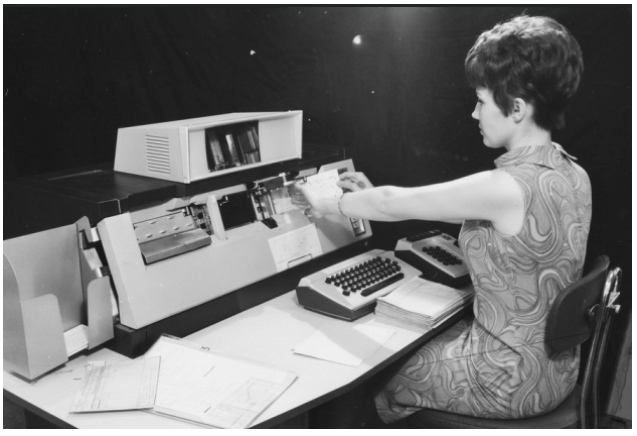
- **le jeu d'instructions qui compose le langage machine,**
- **les registres que le programmeur peut utiliser,**
- **la manière dont la mémoire de l'ordinateur est organisée,**
- **les types de données élémentaires (entiers, nombres flottants).**

I. John Von Neumann

Historique : *John von Neumann*, né le 28 décembre 1903 à Budapest et mort le 8 février 1957 à Washington, est un mathématicien et physicien américano-hongrois qui a donné son nom à l'architecture de *von Neumann* utilisée dans la quasi-totalité des ordinateurs modernes. Selon lui, la mémoire de l'ordinateur, qui servait à stocker des données, devait également stocker les programmes : c'est le concept de programme enregistré.



Au moment de la seconde guerre mondiale, dans le cadre de recherche pour l'armée américaine une équipe dirigée par le mathématicien d'origine hongroise *John von Neumann* développe le premier ordinateur basé sur l'**architecture** dite « **de von Neumann** », qui est encore celle de tous les ordinateurs actuels, et dont un des principes extrêmement innovant pour l'époque est le **stockage des données et des instructions des programmes dans une mémoire unique,** et non plus sur un support externe (ruban, carte perforée ou bande magnétique par exemple).



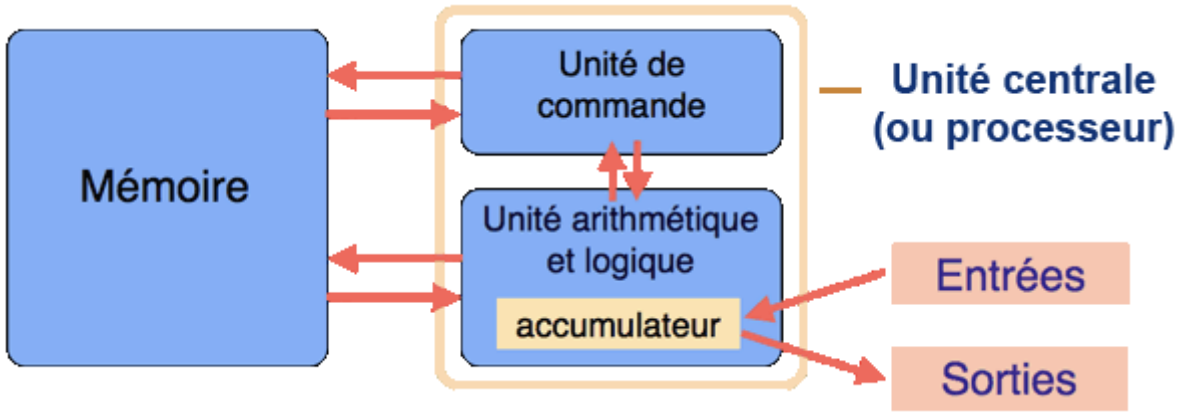
C'était aussi un des premiers ordinateurs dans l'authentique sens du mot, c'est-à-dire davantage qu'un simple calculateur électronique, une des premières machines à réaliser physiquement l'idée théorique d'une « machine universelle » formulée dix ans auparavant par le mathématicien et logicien britannique *Alan Turing*.

Cet ordinateur est capable d'additionner, soustraire, multiplier en binaire. Sa capacité mémoire est l'équivalent actuel de **5,5 ko**. Pourtant, il pèse **7850 kg**, occupe une surface **45.5m²** et nécessite des dizaines de personnes à son chevet en permanence…

Ainsi, les besoins militaires, plus particulièrement ceux liés à la cryptographie et à la conception de la bombe atomique et de la bombe à hydrogène, sont fondamentaux dans l'essor de l'ordinateur…

II. Modèle d'architecture Von Neumann (1945)

L'architecture de Von Neumann décompose l'ordinateur en 4 parties distinctes comme le montre le schéma ci-dessous :



- l'**unité arithmétique et logique** (UAL ou ALU en anglais) ou unité de traitement : son rôle est d'effectuer les opérations de base ;
- l'**unité de contrôle**, chargée du « séquençage » des opérations ;
- la **mémoire** qui contient à la fois les données et le programme qui indiquera à l'unité de contrôle quels sont les calculs à faire sur ces données ;
- les **dispositifs d'entrée-sortie**, qui permettent de communiquer avec le monde extérieur.

Tous ces éléments sont connectés à l'aide de **bus** qui assurent l'interconnexion des différentes composantes de la machine.

La première innovation de *Von Neumann* est la **séparation** nette entre l'unité de commande, qui organise le flot de séquencement des instructions, et l'unité arithmétique, chargée de l'exécution proprement dite de ces instructions.

La seconde innovation, la plus fondamentale, est l'idée du **programme enregistré** : les instructions sont enregistrées dans la mémoire selon un codage conventionnel. Un compteur ordinal contient

l'adresse de l'instruction en cours d'exécution ; il est automatiquement incrémenté après exécution de l'instruction, et explicitement modifié par les instructions de branchement.

Un emplacement de mémoire peut contenir indifféremment des instructions et des données, et une conséquence majeure (dont toute la portée n'avait probablement pas été perçue à l'époque) est qu'un programme peut être traité comme une donnée par d'autres programmes.

Pourquoi donc étudier l'architecture des ordinateurs ?

Au-delà de l'augmentation du nombre de transistors intégrés par processeur, et celle de la fréquence des processeurs, de nombreuses innovations architecturales ont été mises en place afin d'améliorer la performance des machines.

Le programmeur doit être capable de comprendre les caractéristiques de sa machine afin de concevoir des programmes plus performants.

III. Principaux composants

1. Le microprocesseur

Le microprocesseur (ou unité centrale de traitement, UCT, en anglais *Central Processing Unit*, CPU) est un composant essentiel qui exécute les instructions machine des programmes informatiques.

a. Constitution d'un processeur

Il est schématiquement constitué de 3 parties :

- l'**unité arithmétique et logique** est chargée de l'exécution de tous les calculs que peut réaliser le microprocesseur ;
- les **registres** permettent de mémoriser de l'information (donnée ou instruction) au sein même du CPU, en très petite quantité ;
- l'**unité de contrôle** permet d'exécuter les instructions (les programmes).

b. Caractéristiques d'un processeur

Le jeu d'instructions : additionner, multiplier, comparer deux nombres... Un processeur peut exécuter plusieurs dizaines, voire centaines ou milliers, d'instructions différentes.

La complexité : plus le microprocesseur contient de transistors, plus il pourra effectuer des opérations complexes.

Le nombre de bits pouvant être traités simultanément : les premiers microprocesseurs ne pouvaient pas traiter simultanément plus de 4 bits. A partir de 2007 les microprocesseurs peuvent traiter des nombres sur 64 bits. Le nombre de bits des bus, de la mémoire et du processeur permet de manipuler rapidement des grands nombres, ou des nombres d'une grande précision.

La vitesse de l'horloge : plus la vitesse de l'horloge est grande, plus le microprocesseur effectue d'instructions en une seconde.

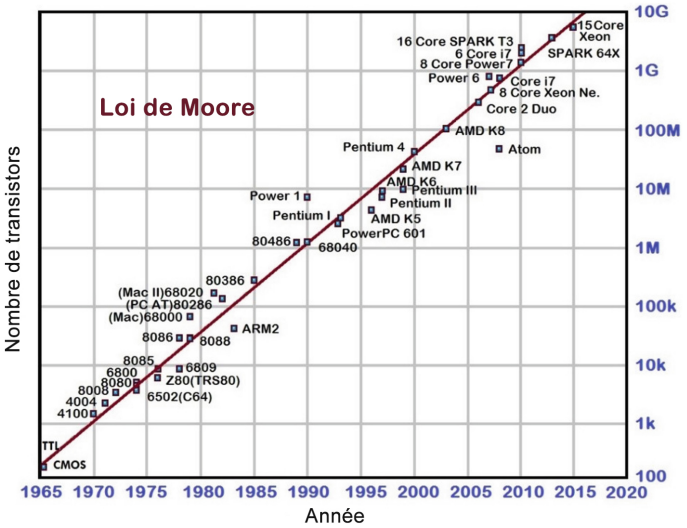
La combinaison des caractéristiques précédentes détermine la puissance du microprocesseur qui s'exprime en « millions d'instructions par seconde » (MIPS).

La puissance d'un processeur est sa capacité à traiter un grand nombre d'opérations par seconde, sur de grands nombres et en grande quantité.

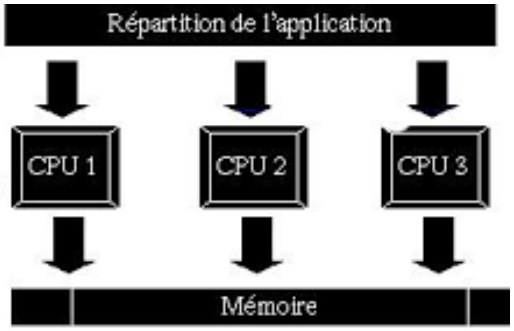
c. Monoprocesseur et multiprocesseur

Une **architecture monoprocesseur** est une architecture ne comprenant qu'un seul processeur.

L'ingénieur américain *Gordon Moore*, cofondateur de la société *Intel,* avait affirmé dès 1965 que le nombre de transistors, c'est-à-dire l'élément principal qui compose les processeurs des ordinateurs, doublerait environ tous les deux ans. Par conséquent, cela doublerait également la puissance des ordinateurs. C'est la *loi de Moore*.



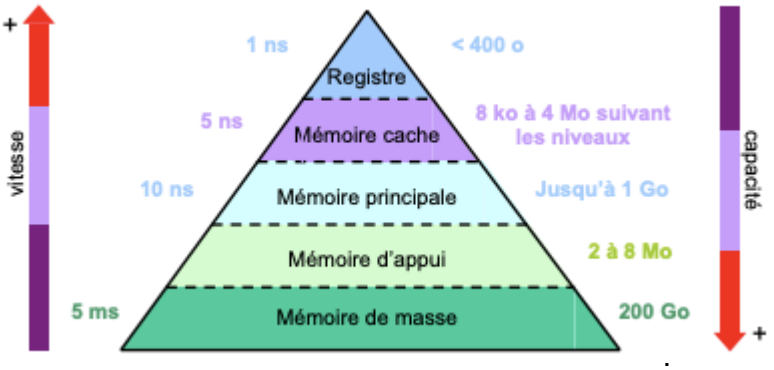
Sa loi a été vérifiée jusqu'à aujourd'hui mais il a déclaré en 1997 que cette croissance des performances des puces se heurterait aux environs de 2017 à une limite physique : celle de la taille des atomes. Une nouvelle idée permet de poursuivre cette loi : mettre plus de processeurs dans un seul ordinateur. C'est l'**architecture des ordinateurs multiprocesseurs.**



Mais avec la multiplication du nombre de processeurs, apparaissent de nouveaux problèmes tels que la synchronisation de la mémoire. On parle alors de **parallélisme**, cela consiste à exécuter simultanément des instructions sur des processeurs différents. Cela se traduit par le découpage d'un programme en plusieurs **processus** traités en parallèle afin de gagner en temps d'exécution.

2. La mémoire

La mémoire permet de stocker des données et des programmes. La mémoire se divise entre **mémoire volatile** (programmes et données en cours de fonctionnement) et **mémoire permanente** (programmes et données de base de la machine).



Dans la plupart des mémoires, les informations sont classées par adresses. En effet, chaque octet est accessible par une adresse unique.

Pour des raisons économiques, les mémoires sont en général divisées en plusieurs familles.

On distingue aujourd'hui la **mémoire dite centrale** (RAM = *Random Access Memory*) de la **mémoire de masse** (qui correspond aux périphériques tels que les disques durs).

L'information est stockée en mémoire :

- **mémoire vive** (RAM : *Random Access Memory*) accessible en lecture/écriture
- **mémoire morte** (ROM : *Read Only Memory*) accessible en lecture seule
- supports de **stockage de masse** (Disque dur, clé usb, DVD-Rom)

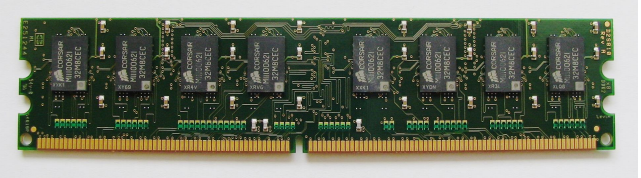
a. La mémoire de masse (ou mémoire de stockage)

Elle sert à stocker à long terme des grandes quantités d'informations.

Les technologies les plus courantes de mémoires de masse sont électromécaniques (disques durs - HDD) ou à semi-conducteurs (SSD, clés USB, …, etc.). Elles visent à obtenir une capacité de stockage élevée à faible coût et ont généralement une vitesse inférieure aux autres mémoires.

Ordres de grandeur actuels :

- capacité : **jusqu'à 14 To (HDD)**
- vitesse : **jusqu'à 500 Mo/s (SSD)**



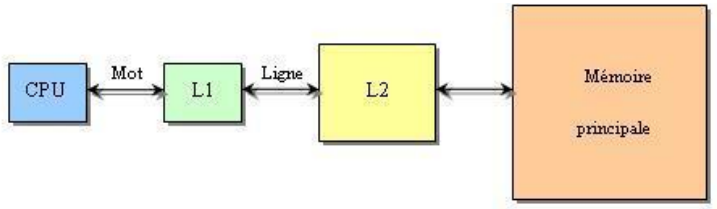
b. La mémoire vive

C'est l'espace principal de stockage du microprocesseur, mais dont le contenu disparaît lors de la mise hors tension de l'ordinateur.

Ordres de grandeur actuels :

- capacité : **jusqu'à 32 Go**
- vitesse : **jusqu'à 2 Go/s**

c. La mémoire cache



Le principe de la mémoire cache est d'insérer entre le processeur et la mémoire traditionnelle, une faible quantité de mémoire très rapide qui stocke les informations les plus récentes ou les plus souvent accédées.

La mémoire cache permet donc de conserver un court instant des informations fréquemment consultées.

Les technologies des mémoires caches visent à accélérer la vitesse des opérations de consultation. Elles ont une très grande vitesse, et un coût élevé pour une faible capacité de stockage.

Ordres de grandeur actuels :

- capacité : **quelques ko (L1) à quelques Mo (L2)**
- vitesse : **jusqu'à 5 Go/s**

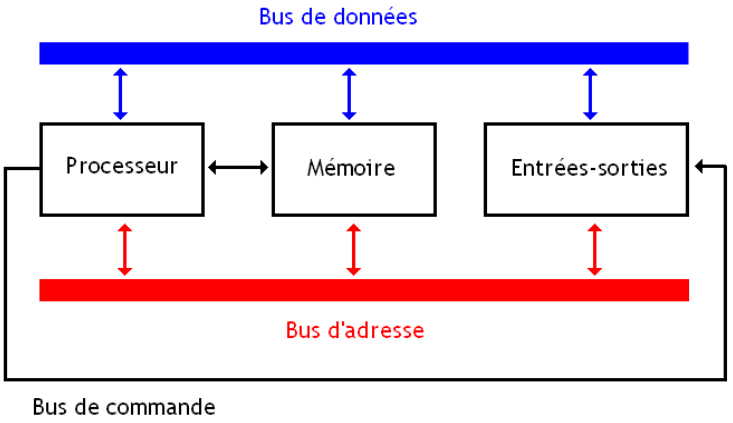
d. Le registre de processeur

Il est intégré au processeur. Ce type de mémoire est très rapide mais aussi très cher et est donc réservé à une très faible quantité de données.

Ordres de grandeur actuels :

- capacité : **quelques dizaines d'octets**
- vitesse : **jusqu'à 30 Go/s**

3. Les bus



Pour que les données circulent entre les différentes parties d'un ordinateur (mémoire, CPU et les entrées/sorties), il existe des systèmes de communication appelés bus.

Il en existe de 3 grands types :

- 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.

IV. Le langage machine

Un microprocesseur est constitué de milliards de transistors, ces transistors comparent des tensions électriques pour effectuer des calculs. Et il utilise des données qui sont stockées dans des mémoires.

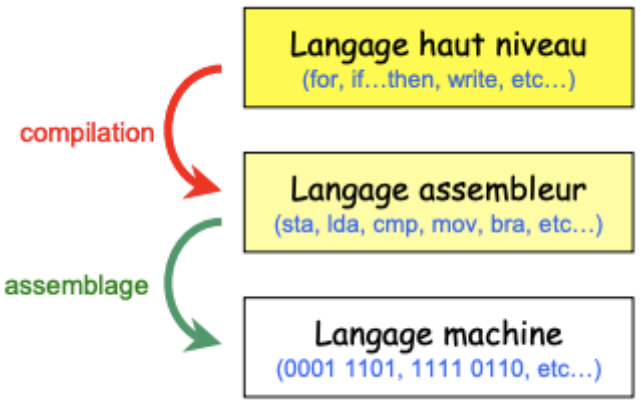
Ainsi, le CPU ne gère que des grandeurs booléennes qui sont codées en binaire. L'ensemble des instructions exécutables directement par le microprocesseur constitue ce que l'on appelle le **langage machine**. Or, les langages de programmation "évolués" dit de **haut niveau** (Python, C++, …, etc.) sont destinés à être utilisés par des humains. Il se composent d'instructions opérant sur des types de données beaucoup plus complexes que des booléens (*int, float, str, list,..., etc.*).

Il faudra donc passer par une étape de "conversion" du langage évolué vers le langage machine. Chaque instruction du langage de haut niveau est "traduit" ou "converti" en une suite d'instructions "élémentaires" du langage machine codées en binaire. C'est ce que l'on appelle la **compilation.**

Attention, il faut distinguer de l'opération de compilation, l'opération d'**interprétation** où la conversion des instructions est réalisée au fur et à mesure du déroulement du code (comme avec Python !).

V. Assembleur

Entre les langages haut niveau et le langage machine, il existe des langages intermédiaires et notamment le **langage assembleur** qui est le langage de plus bas niveau que l'on peut encore appréhender et que le processeur sait traduire en langage machine.



a. Les instructions machines

Une instruction machine est une chaîne binaire composée principalement de 2 parties :

- le champ **"code opération"** qui indique au processeur le type de traitement à réaliser. Par exemple, le code "0010 0110" donne l'ordre au CPU d'effectuer une multiplication.

- le champ **"opérandes"** indique la nature des données sur lesquelles l'opération désignée par le "code opération" doit être effectuée.

b. Exemples d'instructions

Elles sont relativement basiques (on parle d'instructions de bas niveau).
En voici quelques exemples :

- les **instructions arithmétiques** (addition, soustraction, multiplication,...).

Exemple :

```
ADD R1,R0,#128
```
Additionne la valeur contenue dans le registre R0 et le nombre 128 et range le résultat dans le registre R1.

- les **instructions de transfert de données** qui permettent de transférer une donnée d'un registre du CPU vers la mémoire vive et vice-versa.

Par exemple :

```
MOV R1, #23
MOV R1, R3
```
Place le nombre 23 dans le registre R1
Place la valeur stockée dans le registre R3 dans le registre R1

- les **instructions de saut** :

Normalement, au cours de l'exécution d'un programme, le CPU passe d'une instruction à une autre en passant d'une adresse mémoire à l'adresse mémoire immédiatement supérieure.

Ainsi en assembleur, les **instructions de saut** (appelées aussi **instructions de branchement**) permettent d'interrompre l'ordre initial sous certaines conditions en passant à une instruction située une adresse mémoire donnée.

Par exemple :
```
CMP R0, #23
BLT 78
```

La prochaine instruction à exécuter se situe à l'adresse mémoire 78 si la valeur stockée dans le registre R0 est plus grand que 23

c. Exemple de programme en assembleur

```
MOV R0,#42
ADD R1,R0,#128
HALT
```

Tester le programme ci-dessus avec le simulateur suivant : http://www.peterhigginson.co.uk/AQA/