1 Constituants d'un ordinateur

Un ordinateur contient une **unité centrale** et des **périphériques** (clavier, souris, écran, disque dur, carte graphique, carte réseau, imprimante, etc.)

L'unité centrale comprend la **carte mère** sur laquelle sont branchés les différents composants et connectée à l'alimentation électrique. On y trouve en particulier deux éléments essentiels : le **processeur** et la **mémoire**.

Voir en ligne `https://www.youtube.com/watch?v=Fk2kYo2E61A`

2 Les mémoires

Les mémoires sont caractérisées entre autres par :
- la **capacité**, représentant le volume global d'informations (en bits) que la mémoire peut stocker.
- le **temps d'accès**, correspondant à l'intervalle de temps entre la demande de lecture/écriture et la disponibilité de la donnée.
- la **non-volatilité** caractérisant l'aptitude d'une mémoire à conserver les données lorsqu'elle n'est plus alimentée électriquement.

On distinguera en particulier :
- La **mémoire vive,** ou **RAM** (Random Access Memory, mémoire à accès direct), est la **mémoire principale** du système : il s'agit d'un espace permettant de stocker de manière temporaire des données lors de l'exécution d'un programme. C'est la mémoire de travail. C'est une mémoire volatile.
- La **mémoire de stockage**, ou **mémoire de masse** (ex : disque dur, mémoire flash) : permet de stocker des données de façon permanente. Son temps d'accès est très supérieur à celui de la RAM.

Remarque : Il existe aussi de la **mémoire cache** qui accélère le fonctionnement de l'ordinateur en stockant les données utilisées le plus récemment. Elle est située proche du processeur et c'est ainsi la mémoire la plus rapide de l'ordinateur.

3 L'architecture de Von Neumann

L'architecture de Von Neumann repose sur l'association de 3 éléments : le **processeur**, la **mémoire** et les **entrées/sorties**.

3.1 Description

Le **processeur** est le cœur de l'ordinateur : **CPU (Central Processing Unit)**.

Son fonctionnement est **cadencé par l'horloge interne du système** qui rythme chaque opération élémentaire (à quelques GHz sur les processeurs modernes).

Le processeur se compose de deux éléments constitutifs principaux :

- l'**Unité Arithmétique et Logique** (UAL).
- l'**Unité de Contrôle (ou de Commande)** (UC).

C'est l'**UAL qui effectue les calculs de l'ordinateur** : opérations arithmétiques, opérations logiques et opérations de comparaison. L'UAL s'accompagne de **registres** qui sont des zones mémoires à accès rapide dans laquelle l'UAL peut lire ou écrire des données. Les données de la mémoire vive sont transférées dans ces registres pour être traitées.

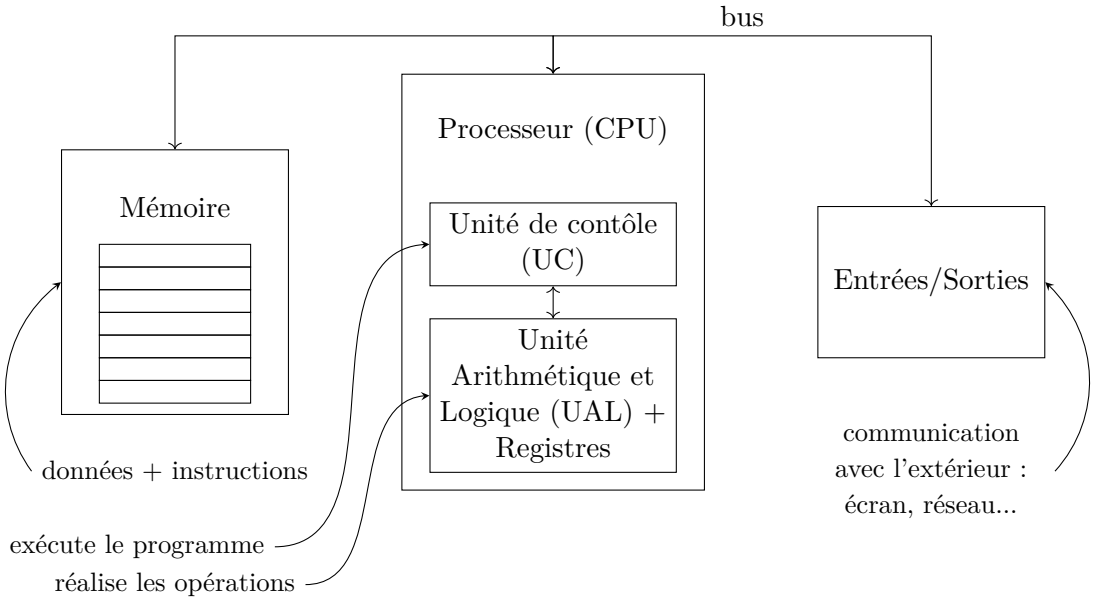
L'**Unité de Contrôle**, accompagnée d'un **séquenceur**, **organise le flot des opérations**. Elle prend ses instructions dans la mémoire. Celles-ci lui indiquent ce qu'elle doit ordonner à l'UAL et comment elle devra éventuellement agir selon les résultats que celle-ci lui fournira.

Le cycle de travail de l'UC est donc le suivant :

1. **lire une instruction (programme) de la mémoire (fetch).**
2. **décoder l'instruction (decode).**
3. **commander son exécution (execute)**.

La **mémoire** est constituée de **blocs élémentaires** (ex : blocs de 4 ou 8 octets) qui sont **identifiés par une adresse**. Par défaut, l'UC passe d'une adresse à l'adresse suivante pour lire une nouvelle instruction ; sauf si une instruction indique spécifiquement de se brancher à une adresse mémoire particulière.

Les dispositifs d'**entrées/sorties** permettent à l'ordinateur de communiquer avec l'extérieur (clavier, écran, imprimantes, réseau, etc.) : transfert d'information dans les 2 sens.



Remarque : les différents éléments de l'architecture sont reliés par des bus transportant les informations utiles (données, adresse mémoire, signal de contrôle de flux).

3.2 Particularité de l'architecture Von Neumann

Dans les premiers ordinateurs (ex : ENIAC), les différentes étapes nécessaires à l'exécution d'une tâche, le programme, étaient directement câblés dans l'unité de contrôle (ruban, cartes perforées, tableau de connexions). Un ordinateur ne pouvait effectuer qu'un seul programme à la fois.

Un grand progrès a été réalisé lorsque le programme a pu être codé et stocké dans la mémoire principale au même titre que les données à traiter : ce modèle correspond à l'architecture de Von Neumann (1945).

Un changement de programme se fait alors par une simple réécriture de la mémoire. Un emplacement de mémoire peut contenir indifféremment des instructions et des données, et une conséquence majeure est qu'**un programme peut être traité comme une donnée par dautres programmes**.

Deux articles complémentaires à lire en ligne sur Interstice :
- Interstices : mémoire et unité centrale (2019)
- Interstices : architecture de Von Neumann (2011)

3.3 Évolution

Pendant des années, pour augmenter les performances des ordinateurs, les constructeurs augmentaient la fréquence d'horloge des microprocesseurs. Plus la fréquence d'horloge du CPU est élevée, plus ce CPU est capable d'exécuter un grand nombre d'instructions machines par seconde.

La fréquence a augmenté de façon exponentielle jusque vers les années 2005-2010 où cette évolution a commencé à être nettement ralentie. Ceci est dû à une contrainte physique : plus on augmente la fréquence d'horloge d'un CPU, plus ce dernier chauffe. Il devenait difficile de refroidir le CPU, les constructeurs de microprocesseurs (principalement Intel et AMD) ont décidé alors d'arrêter la course à l'augmentation de la fréquence d'horloge et ont décidé d'adopter une nouvelle tactique.

L'idée est d'augmenter les performances en augmentant le nombre de cœurs présents sur un CPU : il y a donc plusieurs ensembles {UAL, registres et unité de contrôle}. Un cœur est donc capable d'exécuter des programmes de façon autonome. La technologie a évolué pour graver toujours plus de transistors sur une surface donnée, il est donc possible, sur une même puce, d'avoir plusieurs cœurs, alors qu'auparavant on trouvait un seul cœur dans un CPU. Cette technologie a été implémentée dans les ordinateurs grand public à partir de 2006, et même les smartphones possèdent des microprocesseurs multicœurs.

Cependant l'augmentation du nombre de cœurs n'entraîne pas obligatoirement une augmentation des performances du CPU. La conception d'applications capables de tirer profit d'un CPU multicœur demande la mise en place de certaines techniques de programmation complexes. Il faut aussi avoir conscience que les différents cœurs doivent se partager l'accès à la mémoire : quand un cœur travaille sur une certaine zone de la RAM, cette même zone n'est pas accessible aux autres cœurs.

4 Langage machine

4.1 Principes

> Un ordinateur exécute des programmes qui sont des suites d'instructions. Les instructions exécutées au niveau du CPU sont **codées en binaire**. L'ensemble des instructions reconnues par un processeur et son système de codage forment ce qu'on appelle le **langage machine** du processeur. Il est **spécifique pour chaque processeur** !

Les instructions machines sont relativement basiques.
Voici quelques exemples :

— les **instructions arithmétiques ou logiques** (addition, multiplication, comparaison, etc.).
Exemple : additionner la valeur contenue dans le registre `R1` et le nombre 789 et ranger le résultat dans le registre `R0`.

— les **instructions de transfert de données** qui permettent de transférer une donnée d'un registre vers la mémoire vive et vice-versa.
Exemple : prendre la valeur située à l'adresse mémoire `4E6A` et la placer dans le registre `R2`.
— les **instructions de rupture de séquence (branchement)**.
Normalement au cours de l'exécution d'un programme, l'UC passe d'une instruction à une autre en passant d'une adresse mémoire à l'adresse mémoire immédiatement supérieure. Les instructions de rupture de séquence permettent sous certaines conditions de passer à une instruction située une adresse mémoire donnée.
Exemple : si la valeur contenue dans le registre `R1` est strictement supérieure à 0 alors exécuter l'instruction située à l'adresse mémoire `C27F`. Dans le cas contraire, la prochaine instruction à exécuter est à l'adresse mémoire immédiatement supérieure à celle de l'instruction en cours.

Un programme en langage machine est donc une suite très longue de 1 et de 0 !
Programmer en langage machine est donc extrêmement difficile et pour pallier cette difficulté, les codes binaires sont remplacés par des symboles **mnémoniques** qui forment le langage **assembleur**.
L'assembleur est aussi le nom du logiciel qui convertit un programme assembleur en langage machine.

Exemple :
L'instruction machine "écrire la valeur immédiate 7 dans le registre 5 ", pourra s'écrire `MOV R5, #7` au lieu d'écrire `11100011101000000001000000000111`.

4.2 Exemple de jeu d'instructions

Voir en annexe quelques exemples d'instructions d'un langage assembleur que nous utiliserons en TP (simulateur online de Peter Higginson `https://www.peterhigginson.co.uk/AQA/`) :