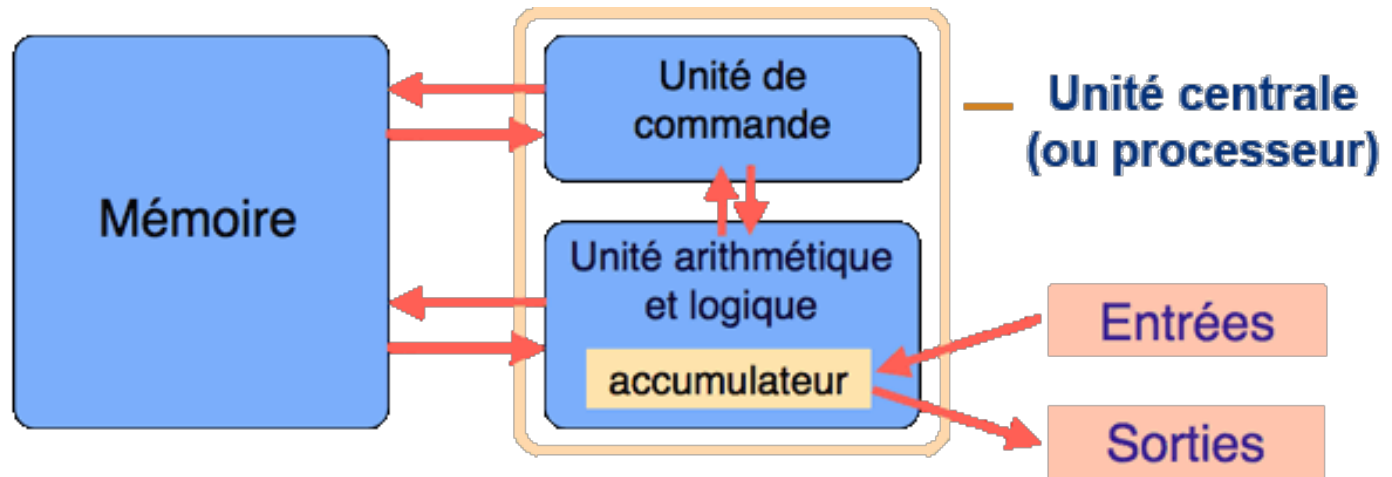


Modèle de Von Neumann - Résumé

qkzk

Modèle de Von Neumann



- **Processeur** : composé de deux unités
- **Unité de commande** : contrôle la séquence d'instructions
- **Unité arithmétique** : exécute les instructions
- **Mémoire** : contient les données et les **programmes**. mémoire *volatile* (qui s'efface après redémarrage, mais rapide) et de *masse* (stockage)
- **Entrées** : clavier, cartes perforées, etc.
- **Sorties** : affichage, imprimante, écran

Apport de von Neumann

Dans ce modèle il est le premier à utiliser **une seule mémoire pour enregistrer les programmes et leurs données**.

Il reprend d'autres concepts comme la séparation entre la mémoire et le processeur.

En traitant les *instructions* de la même manière que les *autres données*, il permet de concevoir des programmes qui créent d'autres programmes. Cette idée est à l'origine des *compilateurs* et des *interpréteurs*.

Cycle

La machine fonctionne par cycle :

- fetch
- read
- execute

Durant chaque cycle elle récupère une instruction, la décode et la réalise.

Unité de commande

Elle contrôle les instructions réalisées par la machine.

C'est elle qui récupère les instructions et les décode.

Elle s'occupe donc des parties "fetch" et "read"

Unité arithmétique et logique

Elle s'occupe de réaliser les *calculs* à effectuer.

Tous les calculs correspondent à des circuits électroniques dont on contrôle les entrées et sorties.

EX : l'additionneur 1 bit.

Mémoire

La mémoire est constituée de cellule de 1 mot (1, 2, 4 ou 8 octets) disposant d'une adresse.

On peut lire et écrire dans chaque cellule.

On y trouve à la fois les données et les programmes.

Les composants

L'ordinateur utilise seulement des 0 et des 1. Les composants fonctionnent souvent en 5V (parfois 12V, parfois 3.3V, plus rarement autre chose).

- +5V : 1 : True
- 0V : 0 : False

Le transistor



Figure 1: transistor

C'est un interrupteur contrôlable. Il dispose de 3 broches (2 entrées, 1 sortie.)

Il existe différents modèles mais, par exemple, si la base est alimentée, le courant circule entre le collecteur et l'émetteur. Sinon, il ne circule pas.

Circuit intégré

Composé de plusieurs milliers de transistors.

Opérations booléennes

En combinant les transistors on forme les portes logiques.

Mémoire vive

La mémoire vive est elle-même un circuit électronique. Tant qu'elle est alimentée, elle permet de conserver de l'information. Dès qu'elle n'est plus alimentée, l'information est perdue. Unité minimale : l'octet.

Le processeur

Les processeurs modernes comportent tous les éléments du modèles de Von Neumann :

- des registres (mémoire vive)
- Unité arithmétique et logique
- Unité de commande

Les bus

L'information circule dans des bus. Physiquement, des câbles. Il existe différents bus (au moins adresses, données, contrôle)

L'évolution des performances

A explosé depuis l'invention des premiers ordinateurs. Avant ça, les progrès étaient déjà fulgurants.

Loi de Moore (1965) :

tous les 18 mois, le nombre de transistor par processeur double

Restée valable jusqu'en 2005.

Problème de la chaleur

Depuis : insoluble problème de la dissipation de la chaleur.

La surface de contact a diminué, on ne peut plus dissiper la chaleur.

Exemple d'évolution moderne : multicoeur

Autre approche : multiplier les coeurs dans un processeur.

Coeur = UAL, registres et unité de commande.

Un coeur peut exécuter des programmes de façon autonome.

Difficulté : programmer les machines en parallèle.

Assembleur

Quelques principes

- Les instructions machines sont propres à chaque processeur (heureusement, il existe des principes communs, de grandes familles).
- Les humains programment les machines dans des langages plus haut niveau.

Langages de différents niveaux

1. langage machine : 01111111 11001010 01001000
2. assembleur : ADD, R0, R0, #3 : ajoute le contenu de R0 au nombre 3, stocke le dans R0.
3. langage haut niveau :
4. C, Rust etc. : on peut contrôler directement la mémoire (*virtuelle*) et piloter du matériel La majorité des pilotes matériels sont écrits en C (à ma connaissance).
5. langages haut niveau (python, javascript) : pas d'accès direct au matériel.

Donc, pour contrôler du matériel directement, un humain écrit en assembleur. Un programme ('l'assembleur') le compile en langage machine.

Un *compilateur* est un programme qui traduit du langage haut niveau en langage machine. Ex : gcc (linux) compile du C en langage machine.

Exemple de programme pour l'émulateur ARMLite

```
1|      MOV R1, #10           //      R1 contient 10
2|      MOV R2, #5            //      R2 contient 5
3|      ADD R3, R1, R2        //      R3 contient 5 + 10 = 15
4|      CMP R3, R1            //      Comparaison de 15 et 5
5|      BGT more              //      15 > 5 donc on va en "more", ligne 08.
6|      HALT                  //      Fin du programme (ne sera pas exécuté dans l'exer
7|                               //      Rien, on va à la ligne
8|  more:                     //      pas de code après l'étiquette, on ne fait rien
9|      ADD R3, R1, R3        //      R3 contient 15 + 10 = 25
10|     HALT                   //      Fin du programme
```