Introduction à l'informatique

Pr. Ahmed DRISSI EL MALIANI

14 décembre 2016

Information

Codage

Ordinateur

Système d'exploitation

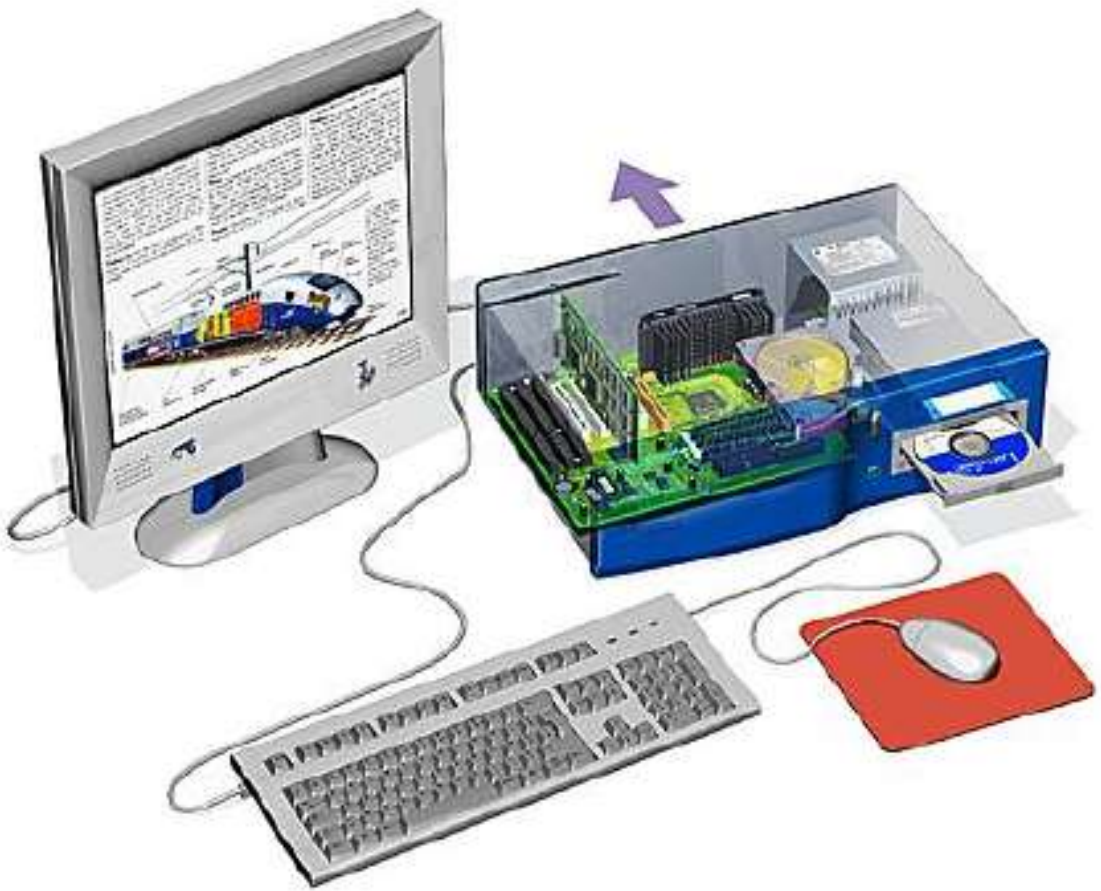
Internet

Logiciels

Une avalanche de données

L'ordinateur traite les données

C'est quoi Ordinateur ?



C'est quoi Ordinateur ?

Objectifs du cours

- Donner une vue d'ensemble de l'informatique.
- Savoir la structure d'un ordinateur et comprendre le rôle des composants de l'ordinateur.
- Comprendre les principes fonctionnement des langages de programmation.
- Maitriser le codage de l'information.
- S'initier aux principes de fonctionnement des réseaux et internet.

Contexte général

- L'informatique s'intègre de plus en plus dans la vie de chacun
- L'informatique commence à changer les règles sociétales
- Les ordinateurs s'accaparent les bureaux et les meubles

- MAIS, qu'est ce que l'**informatique** ? ? ? et qu'est ce que l'**ordinateur** ? ? ?

1. Qu'est ce que l'Informatique
2. Qu'est ce qu'un Ordinateur : coté matériel
3. Qu'est ce qu'un Ordinateur : coté logiciel
4. Codage de l'information

Mauvaises réponses

- L'informatique n'est pas la science de l'ordinateur.
  - un informaticien n'est pas obligé de savoir réparer un ordinateur en panne
  - n'est pas le meilleur pour vous dire quel carte graphique ou quel scanner acheter
- L'informatique n'est pas la science du logiciel
  - un informaticien ne sait pas toutes versions des programmes
  - n'essaie pas de régler tous les bugs des logiciels
- L'informatique (compétence d'un informaticien) n'est ni matérielle ni logicielle
  - c'est quelque chose de plus abstrait qui fait que les deux (matériel et logiciel) puissent bien fonctionner

Définition : Informatique

En anglais "**Computer Science**"

L'**Informatique**(**Information**+**Automatique**)

est la science du **traitement automatique** de l'**information**

Définition : Informatique

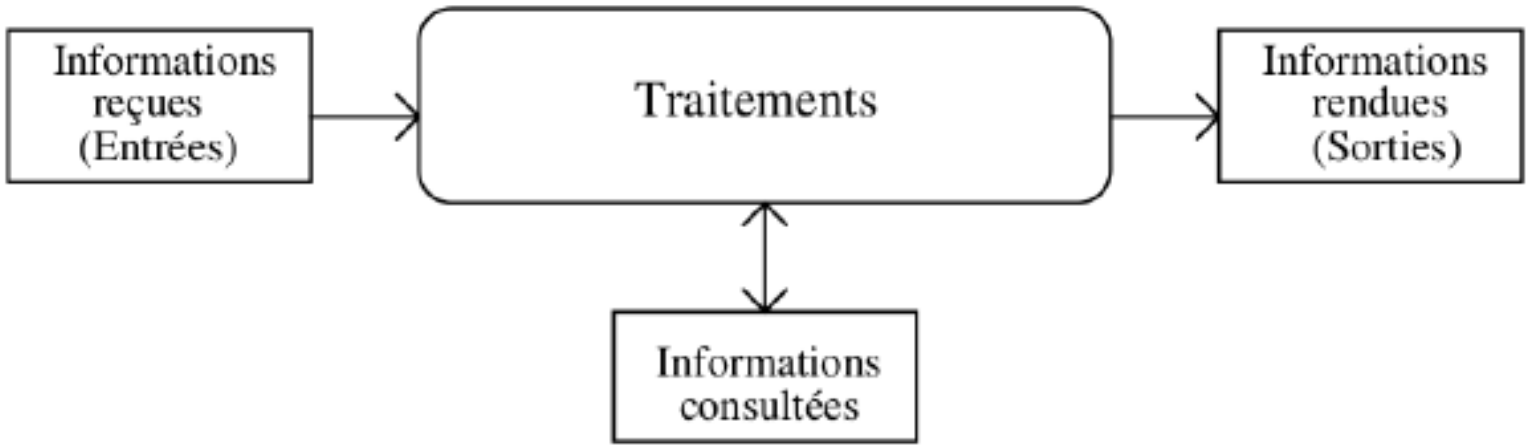
- L'informatique est l'art, la technique ou la science qui consiste à manipuler des informations à l'aide d'un outil, l'ordinateur.
- Elle a pour objectif de définir des algorithmes qui permettent de rendre plus facile un problème

Définition : l'information

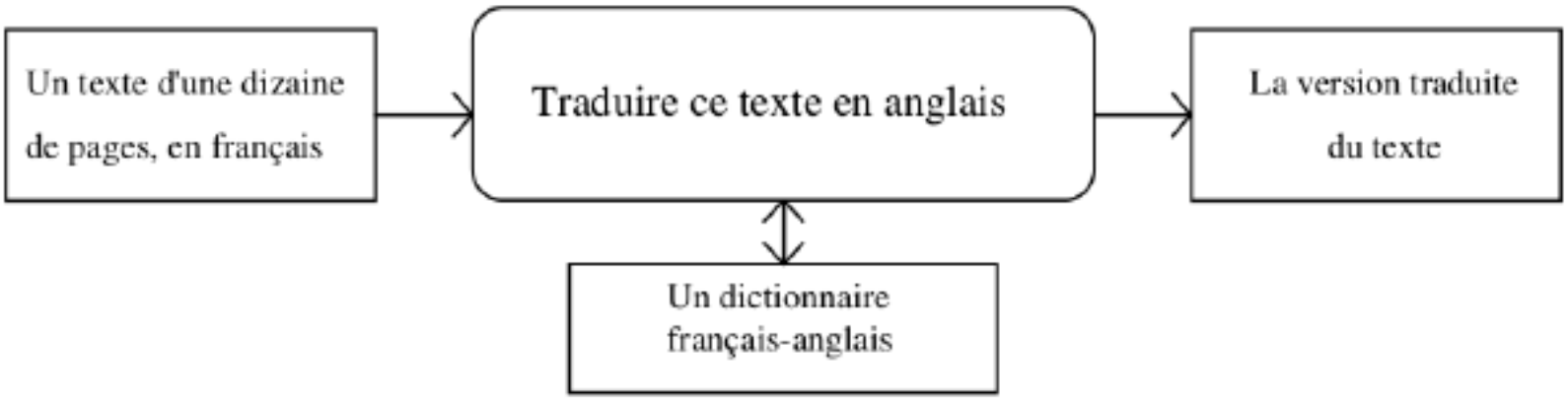
L'**information** : est un élément de connaissance humaine susceptible d'être représentée à l'aide d'un système de codage afin d'être conservée, traitée ou communiquée.

Définition : Traitement automatique de l'information

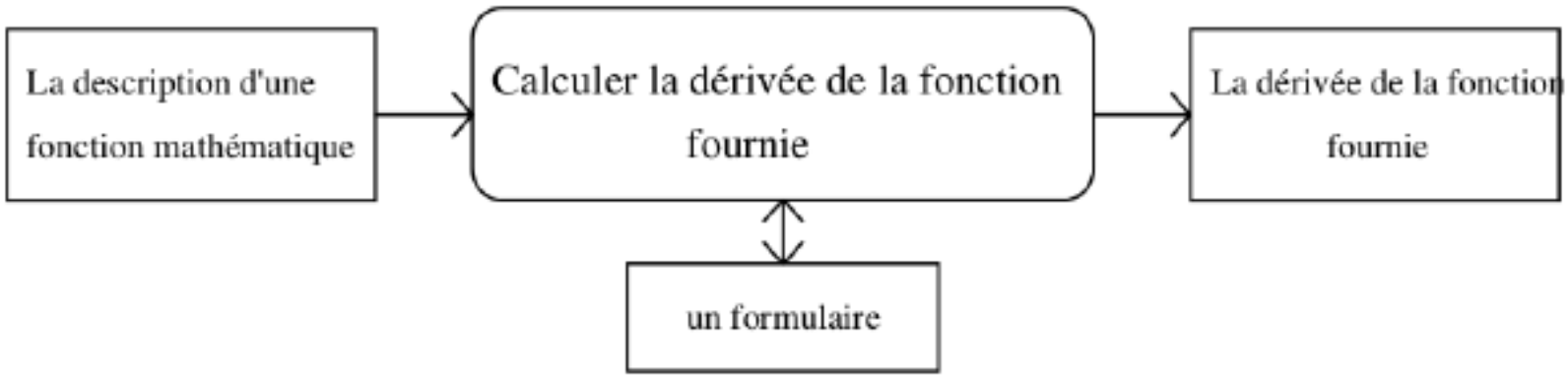
Ensemble d'opérations transformant une représentation de cette information en une autre représentation plus facile à manipuler



Exemple de traitements d'information par le cerveau humain



Exemple de traitements d'information par le cerveau humain

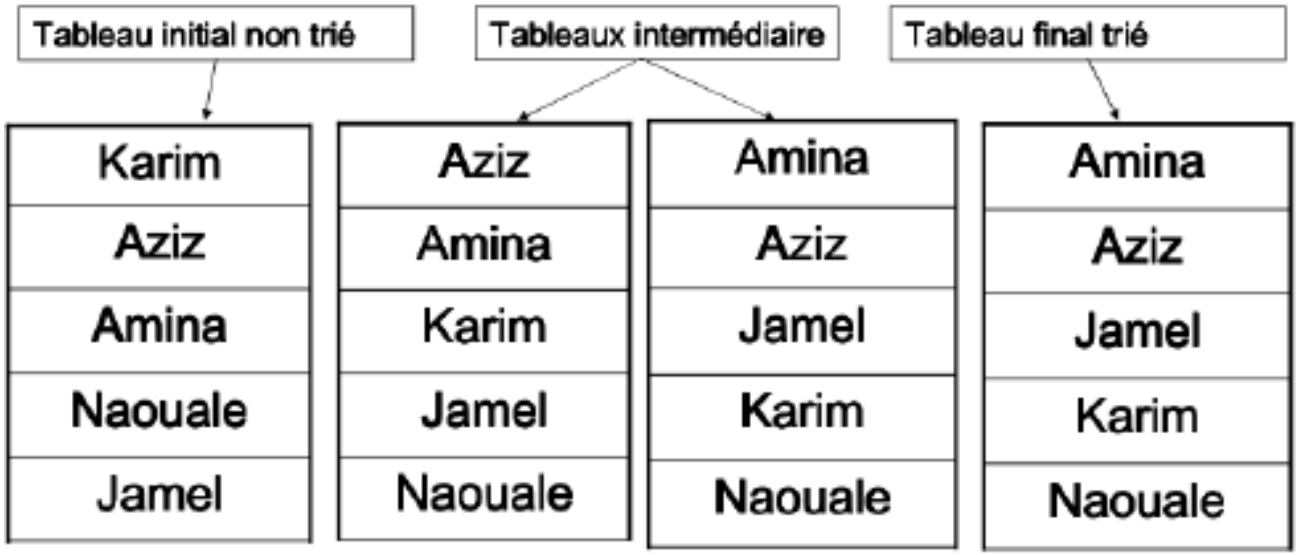


Exemple de tri par ordre alphabétique

Exercice : Trier le tableau suivant :

Karim
Aziz
Amina
Naouale
Jamel

Exemple de tri par ordre alphabétique



Actions à suivre pour aboutir à une liste de noms triée :

1. Parcourir la liste en Comparant deux noms successifs
2. Échanger leur position selon l'ordre alphabétique
3. Refaire 1 et 2 jusqu à ce que la liste soit triée

Exemples de traitements d'information

- Calcul du produit de deux nombres (3*2=6)
- Calcul de la moyenne des notes d'un module
- Entreprise : établir la fiche de paye, faire la facturation, ...

Traitement de l'information

Le **Système informatique** est le responsable du traitement automatique de l'information

Le Système informatique

le Système Informatique est composé de deux parties :

- le Matériel :
  - éléments physiques, des composants électroniques, des câbles électriques
- les Logiciels :
  - ensemble de programmes informatiques (suite d 'instructions qui dit à l'ordinateur quoi faire) : système d'exploitation, langage de programmation, ...

Le Système informatique

- Les éléments d'un système informatique :

1 Qu'est ce que l'Informatique

2 Qu'est ce qu'un Ordinateur : coté matériel

3 Qu'est ce qu'un Ordinateur : coté logiciel

4 Codage de l'information

L'ordinateur

- C'est une machine qui permet le traitement de l'information en exécutant une série d'ordres
- C'est une machine doté de mémoires à grandes capacités et de moyens de calculs extrêmement rapides

L'Ordinateur

Le mot «ordinateur» a été créé en 1955 à la demande d'IBM, et tire son étymologie du terme «ordonnateur», désignant l'autorité suprême

L'Ordinateur

- Toute machine capable de manipuler des informations peut être qualifiée d'ordinateur
  - Ordinateur de poche : Smartphone
  - Ordinateur portable : Laptop, Tablette tactile
  - Ordinateur de bureau : Personnel, Station de travail
  - Ordinateur géant : Mainframe, Superordinateur

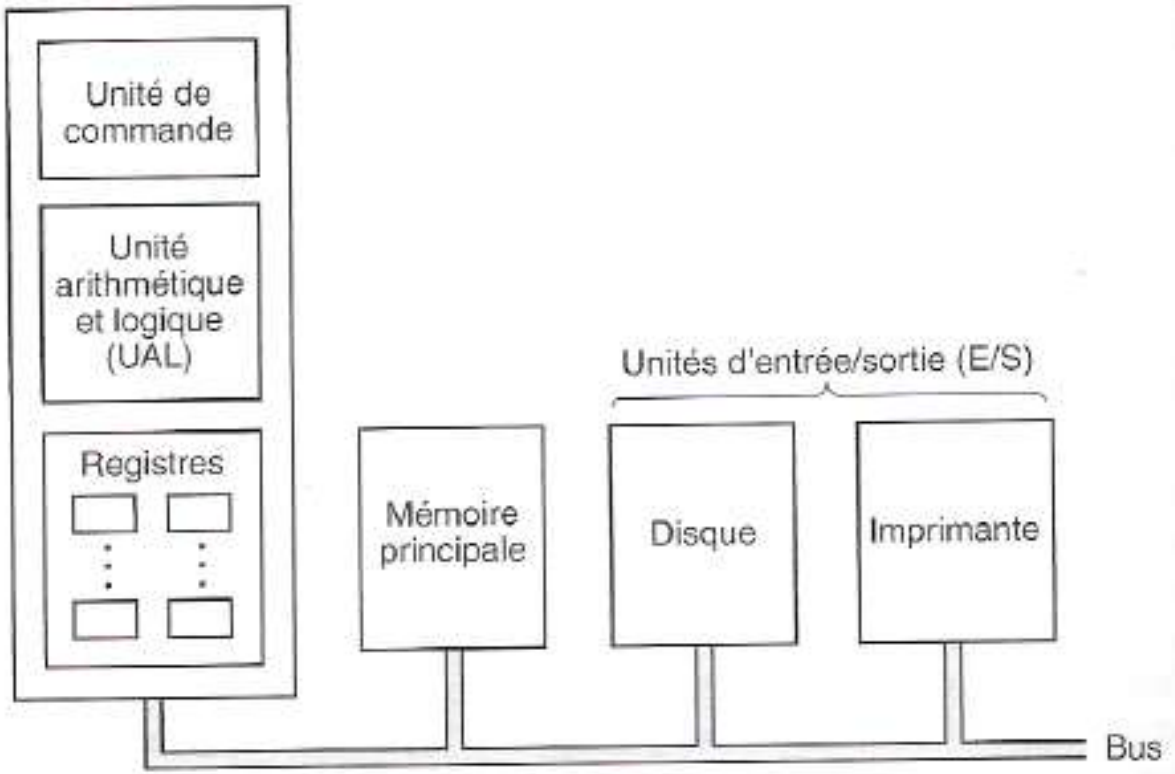
Architecture d'un ordinateur

La structure générale d'un ordinateur est constituée par quatre blocs fondamentaux :

- **Carte mère** : circuit principal de l'ordinateur
- **Unité Centrale de traitement** : processeur, Cœur de l'ordinateur
- **Mémoire centrale** : sert à stocker les données et les programmes
- **Interfaces d'entrées/sorties** : permettent de raccorder les périphériques externes d'un ordinateur.

Architecture d'un ordinateur

Ces blocs sont reliés entre eux par des bus. Un bus est un ensemble de broches qui véhicule l'information.



Carte mère

La carte mère est le circuit principal de l'ordinateur. C'est sur elle que tous les autres éléments vont venir se connecter : alimentation, processeur, Bios, mémoire, cartes d'extension, disques, clavier, souris, modem, imprimante...

Unité centrale (Central Processor Unit)

- Appelée aussi **processeur** et **CPU**, c'est le centre de calcul et de contrôle d'un ordinateur : elle constitue le « cerveau » de l'ordinateur.
- A l'intérieur de tout CPU se trouve :
  - 1/ Unité Arithmétique et Logique (UAL)
  - 2/ Registres
  - 3/ Unité de commande (UC)
  - 4/ Horloge

Unité centrale (Central Processor Unit)

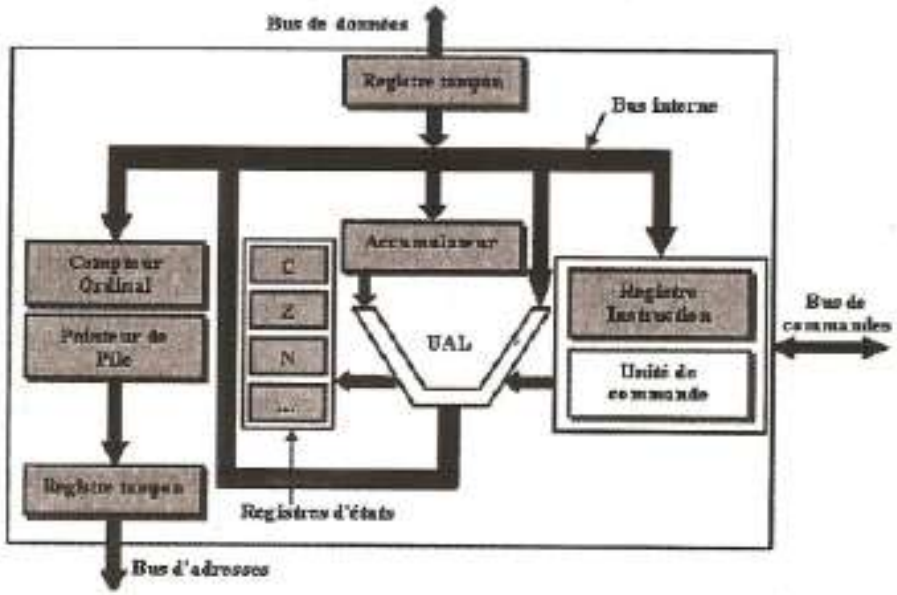
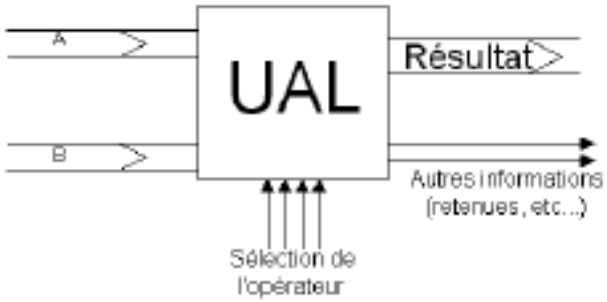


Figure 6 - Exemple d'architecture de l'unité centrale

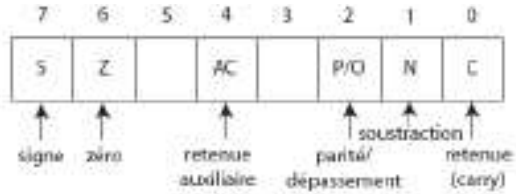
UAL

- Responsable des opérations arithmétiques et logiques (Addition, soustraction, Multiplication,...)



Registres

- Petites mémoires très rapides qui stockent localement les informations traitées par le processeur.
- Ce sont des zones de stockage temporaires qui conservent les données ou les instructions, et gèrent les adresses ainsi que les résultats des opérations.
- Chaque registre a une fonction spécifique :
  - **Registre d'état** : il est associé à l'UAL, et contient des indicateurs qui, après chaque opération, indiquent certains états particuliers tels qu'une retenue, le signe de la parité d'un résultat, etc.



  - **Compteur ordinal (PC- Program Counter)** : il est associé à l'unité de commande. Il est chargé de pointer vers la prochaine instruction à récupérer puis à exécuter.
  - **Registre instruction** : est chargé de stocker l'instruction en cours d'exécution. Il est associé à l'unité de commande.

UC

- Coordonne le travail des différents organes UAL, mémoires, périphériques, ...
- Elle effectue la recherche en mémoire de l'instruction, son décodage, son exécution et la préparation de l'instruction suivante.

Horloge

- Base de temps qui distribue régulièrement des impulsions pour synchroniser les différentes opérations élémentaires effectuées par le processeur.
- La vitesse de cette horloge est appelé fréquence (exprimée en MHz). Plus cette fréquence est élevée plus le processeur est efficace.
- Un processeur cadencé à 450 MHz possède 450 millions de cycles d'horloge par seconde.

Évolution de la vitesse des processeurs

Processeur	Année	Fréquence (MHz)
Intel 86	1981	de 4 à 8
Intel 286	1984	de 6 à 16
Intel 386	1985	de 16 à 33
Intel 486	1989	de 25 à 100
Pentium	1993	de 60 à 200
Cyrix 6x86	1996	de 133 à 150
Pentium MMX	1997	de 166 à 233
PENTIUM III	1999	500
AMD ATHLON	1999	600, 750 1000
PENTIUM IV	2000	1,4 à 2 GHz
AMD ATHLON 64	2003	2 à 2,4 GHz
PENTIUM IV	2004	2,8 à 3,4 GHz

Processeur : Exécution d'une instruction

L'unité centrale exécute chaque instruction en effectuant une série de tâches qui se résume ainsi :

1. Récupérer l'instruction dans la mémoire et la charger dans le registre instruction.
2. Modifier la valeur du compteur ordinal pour qu'il pointe vers l'adresse de la prochaine instruction.
3. Déterminer le genre d'instruction venant d'être chargée.
4. Si l'instruction utilise un mot de la mémoire, localiser son emplacement. Charger le mot, s'il y a lieu, dans un registre de l'UC.
5. Exécuter l'instruction.
6. Retourner à l'étape 1 pour effectuer l'instruction suivante.

Cette séquence d'événements est souvent appelée cycle de **chargement-décodage-exécution**. Elle est la clé du fonctionnement d'un ordinateur.

Types de processeurs

2 types :

- RISC (Reduced Istruction Set Computer) : processeur à jeu d'instructions réduit
  - les instructions sont en nombre réduit (chargement, branchement, appel sous-programme) et elle sont fréquemment utilisées.
  - Le but est d'éliminer les instructions rarement employées
  - et de consacrer les ressources matérielles à exécuter les instructions relativement simples en un cycle d'horloge
  - les autres instructions sont émulées à l'aide des séquences basées sur les instructions élémentaires.
  - Ex : UltraSPARC de Sun
- CISC (Complex Instruction Set Computer) : processeur à jeu d'instructions complexe

Types de processeurs

- CISC (Complex Instruction Set Computer) : processeur à jeu d'instructions complexe
  - Le processeur doit exécuter des tâches complexes par instruction unique.
  - Donc, pour une tâche donnée, une machine CISC exécute un petit nombre d'instructions mais chacun nécessite un plus grand nombre de cycles d'horloge
  - Ex : Pentium Intel

Types de processeurs

- Actuellement, les machines RISC et les machines CISC cohabitent
- Aucun des deux types d'architectures n'a réellement supplanté l'autre.
- Par exemple, depuis le 486, les processeurs Intel intègrent
  - un coeur RISC pour l'exécution des instructions les plus simples (et généralement les plus fréquentes)
  - les instructions plus complexes sont interprétés selon le processus CISC habituel.

Mémoire : Introduction

- La mémoire principale est la zone de l'ordinateur où sont stockés les programmes et les données.
- Les mémoires sont composées d'un ensemble de cases mémoire pouvant chacune stocker une certaine information.
- La case est une composante fondamentale car il s'agit de la plus petite unité adressable. En fait, Chaque case possède un numéro (adresse) qui permet aux programmes de la référencer. Si une mémoire comprend n cases, celles-ci se voient attribuer les adresses de 0 à n-1. Les adresses mémoire sont exprimées en binaire ou souvent en hexadécimal.

Vue logique d'une mémoire

- Les mémoires sont composées d'un ensemble de cases mémoire pouvant chacune stocker une certaine **information**.

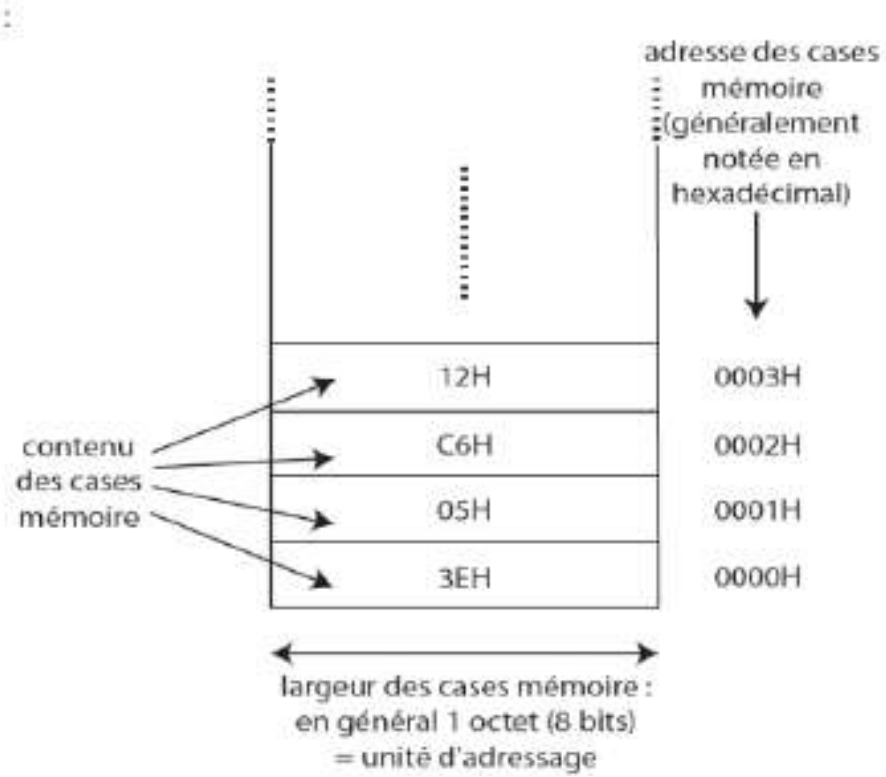
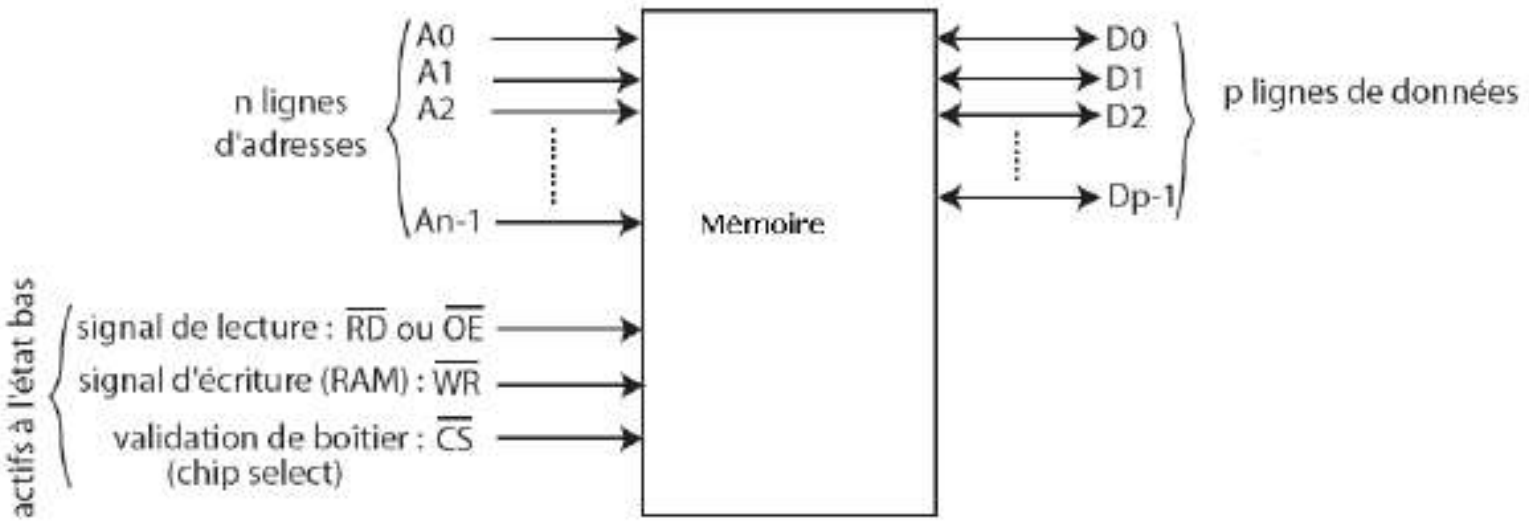
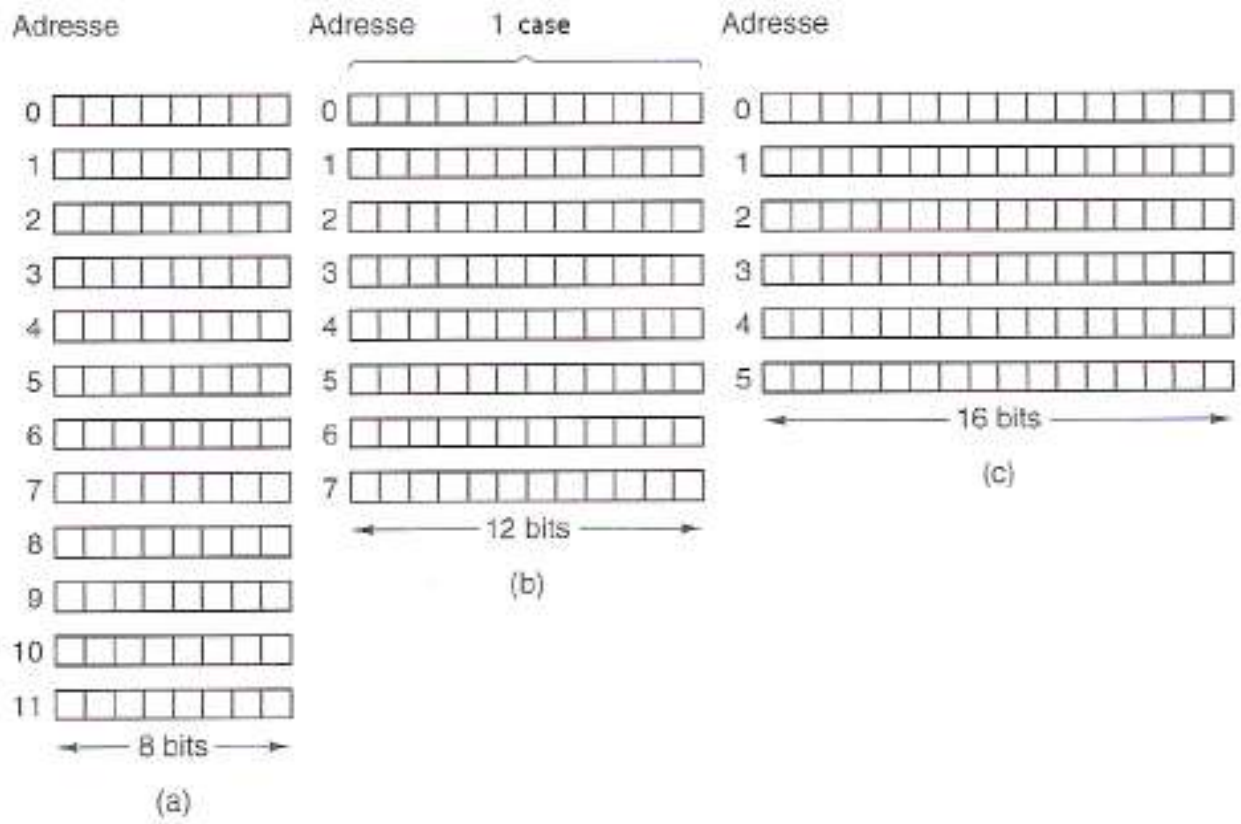


Schéma fonctionnel d'une mémoire



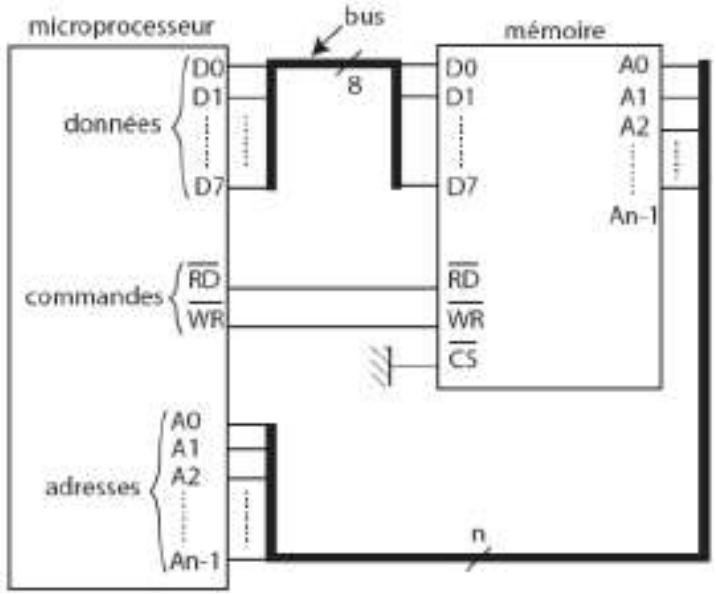
Organisation d'une mémoire

- Toutes les cases d'une mémoire contiennent le même nombre de bits appelé organisation. La figure ci-dessous illustre trois organisations possibles d'une mémoire de 96 bits.



Caractéristiques d'une mémoire

- Temps d'accès :
  - Dans le cas d'une lecture, on définit le temps d'accès à une mémoire comme le temps qui s'écoule entre l'instant où l'adresse de la case mémoire est présentée sur le bus d'adresses et celui où la mémoire place la donnée demandée sur le bus de données.



Caractéristiques d'une mémoire

- Capacité d'une mémoire
  - la capacité d'une mémoire représente le nombre total de bits qu'elle peut stocker ou plus généralement le nombre d'octets :
    - $C = 2^{N_a} \times N_d$
    - N_a :nombre de bits d'adresses.
      N_d :nombre de bits de données (organisation).
  - Soit une mémoire où : $N_a = 10$ et $N_d = 8$
    Elle contient donc : $2^{10}x8 = 8192$ bits $= 8$ K bits $= 1024$ octets $= 1$ Ko

Représentation de l'information sur la mémoire

- Sur machine toute information est représentée sous forme **binaire** (**bit**).
- le bit est L'unité de base de la théorie de l'information bit = binary digit.
- Un bit, par définition, est un composant quelconque ne pouvant se trouver que dans deux états possibles, exclusifs l'un de l'autre.

Exemples

- un fil électrique dans lequel le courant circule ou pas ;
- un aimant pouvant être polarisé «Sud» ou «Nord»
- une surface ayant soit un creux soit une bosse.
- ...

Bit

- Par convention, on appelle l'un des deux états possibles d'un tel composant 0 , et l'autre 1 .
- Un bit sera donc un espace dans lequel on pourra soit écrire 0 , soit écrire 1.
- Que faire avec de tels composants aussi élémentaires ?
- Réponse : Avec un seul, pas grand chose, mais avec plusieurs, beaucoup de choses !

Représentation en bits

- 2 bits
  - Le nombre total d'états possibles que peuvent prendre ces deux bits est de quatre : 00, 01, 10 ou 11.
    $\rightarrow$ on code 4 informations
- 3 bits
  - le nombre total d'états possibles est huit : 000, 001, 010, 011, 100, 101, 110, 111
    $\rightarrow$ on code 8 informations
- Avec n bits :
  - Pour 1 bits $2 = 2^1$, pour 2 bits $4 = 2^2$, pour 3 bits $8 = 2^3$
    Pour n bits on code 2^n **informations**

Système d'unités

- On utilise un système d'unités basés sur l'**octet** (en anglais byte) :
  - 1 Ko (Kilo-octet) = 1024 octets = 2^{10} octets.
  - 1 Mo (Méga-octet) = 1024 Ko = 2^{20} octets.
  - 1 Go (Giga-octet) = 1024 Mo = 2^{30} octets.
  - 1 To (Téra-octet) = 1024 Go = 2^{40} octets.

Types de mémoires

Un ordinateur comporte 3 types de mémoires :

- Mémoire centrale (principale)
- Mémoire cache
- Mémoire secondaire

Mémoire Centrale

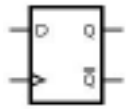
La mémoire centrale est divisée en 2 sections :

- ROM (Read Only Memory) :
  - Mémoire morte où les informations ne sont accessibles à l'utilisateur que pour la lecture
  - Permanente (conserve indéfiniment son contenu)
  - Contient des programmes spéciaux (fait par le constructeur) -> BIOS(Basic Input Output System)
- RAM (Random Access Memory) :
  - Mémoire vive où l'on peut faire toutes les modifications souhaitées
  - volatile (contenu perdu si coupure de courant)

Mémoire vive

Il existe 2 types de RAM : statique et dynamique

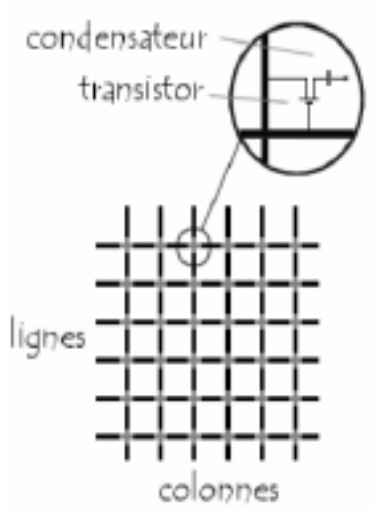
- Les mémoires vives statiques (SRAM) :
  - Utilisent des Bascules pour mémoriser chaque bit.
  - Une bascule est un dispositif qui a plusieurs entrées et une ou deux sorties (0 ou 1).



  - Les SRAM sont très rapides.
  - Les SRAM sont chères.

Mémoire vive

- Les mémoires vives dynamiques (DRAM)
  - sont composées d'une matrice de cellules, chaque cellule contenant un transistor et un condensateur.
  - Les condensateurs peuvent être chargés ou déchargés, ce qui permet de stocker des 0 et des 1.
  - La densité (nombre de bits par puce) des DRAM est plus élevée que les SRAM. La plupart des mémoires principales sont donc construites à partir de DRAM.



Mémoire vive

Il existe plusieurs types de mémoires DRAM. Nous en citons les SDRAM (Synchronous DRAM) et les SDRAM DDR (Double Data Rate).

- SDRAM : RAM hybride, mi-statique, mi-dynamique, pilotée par l'horloge du système principal.
- SDRAM DDR : Dans ce type de mémoire, le transfert des données se fait à la fois sur le front montant et sur le front descendant des impulsions d'horloge ce qui double le débit.
  Ex : une carte mère dotée de mémoire SDRAM DDR et étant cadencée à 133 MHz est équivalente en débit de données à une SDRAM à 266 MHz.

Mémoire morte

- Dans plusieurs applications (voitures, machines électroménagères, ordinateurs,...) le programme et certaines des données doivent être stockés, même lorsque l'alimentation est coupée.
- Ces derniers ne sont jamais modifiés
- Besoin, donc, d'une mémoire non effaçable, non réinscriptible, dont les données sont enregistrées lors de la fabrication $\Rightarrow$ c'est la mémoire ROM
- Dans le cas ordinateur, la ROM est un support permanent pour la conservation des informations de démarrage et les procédures d'entrée/sortie.
- types :
  - **PROM** (Programmable ROM) : elle est programmée une seule fois par l'utilisateur
  - **EPROM** (Erasable PROM) et **EEPROM** (Electrically Erasable PROM) : Elle peuvent être effacées puis reprogrammées pour un autre usage.

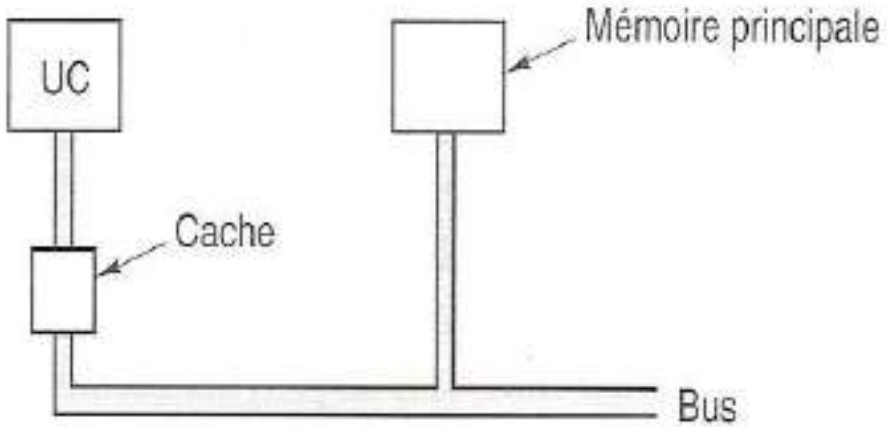
Types de mémoires

Un ordinateur comporte 3 types de mémoires :

- Mémoire centrale (principale)
- Mémoire cache
- Mémoire secondaire

Mémoire Cache

- Les unités centrales sont toujours plus rapides que les mémoires.
- Ainsi, un processeur demandant une lecture mémoire n'obtient le mot recherché que plusieurs cycles d'horloges plus tard
- Un cache (ou encore antémémoire) est une mémoire rapide de petite taille qui contient les mots mémoire les plus récemment utilisés, accélérant ainsi l'accès à ces mots



Sur le plan logique, le cache est situé entre l'UC et la mémoire principale.
Sur le plan physique, le cache peut être positionné en différents endroits.

Types de mémoires

Un ordinateur comporte 3 types de mémoires :

- Mémoire centrale (principale)
- Mémoire cache
- Mémoire secondaire

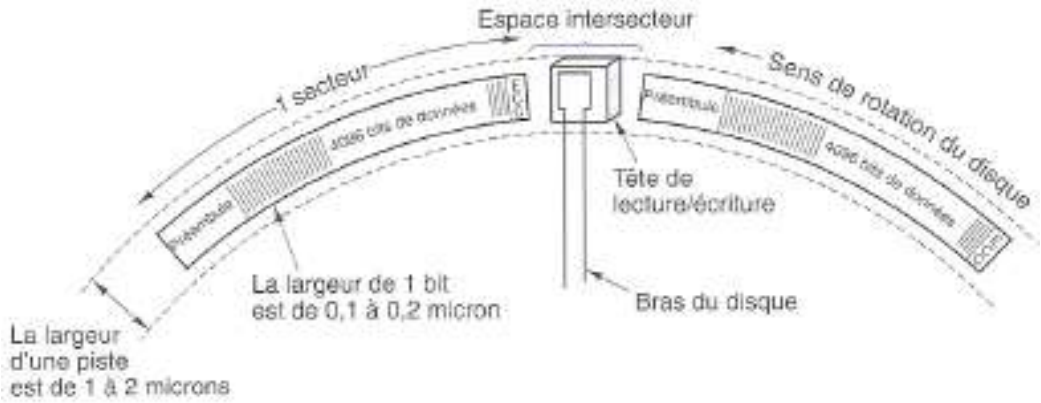
Mémoires secondaires

- On associe à la mémoire centrale d'un ordinateur des mémoires secondaires appelées mémoire de masse. Ce sont des supports qui peuvent stocker d'une manière permanente les programmes et les données
- Il existe divers types :
  - Disques magnétiques
  - CD-ROM
  - DVD
  - Blu-Ray
  - Clé-USB

Disques magnétiques

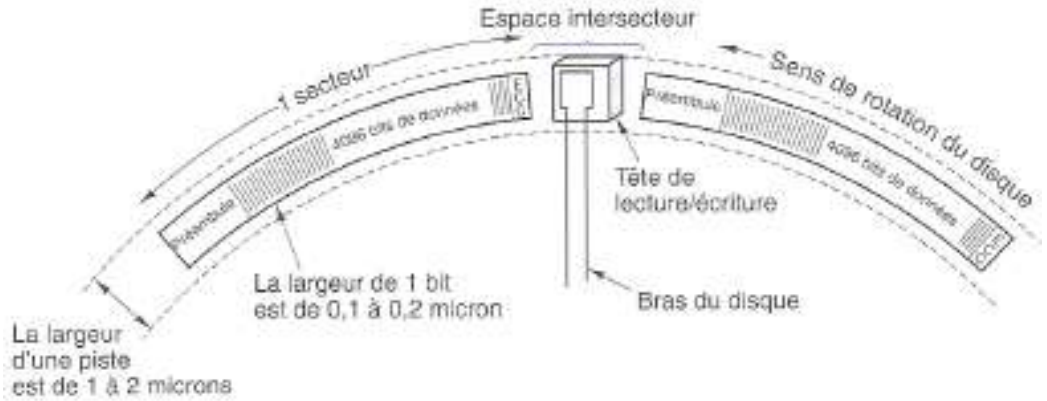
- Un disque magnétique (disque dur) est composé d'un ou de plusieurs plateaux en aluminium recouverts d'une couche de matériau magnétisable.
- Une tête de disque au dessus de la surface des plateaux.
- Lorsque la tête se positionne au dessus d'une zone magnétisée, un courant positif ou négatif circule dans la tête, ce qui permet de lire les bits préalablement stockés.
- A mesure que les plateaux tournent sous la tête, il est possible de lire ou d'écrire une séquence de bits.

- La séquence de bits circulaire pouvant être inscrites au cours d'une rotation complète d'un plateau est appelée piste. Chaque piste est divisée en un certain nombre de secteurs de longueur fixe. Le secteur contient les données

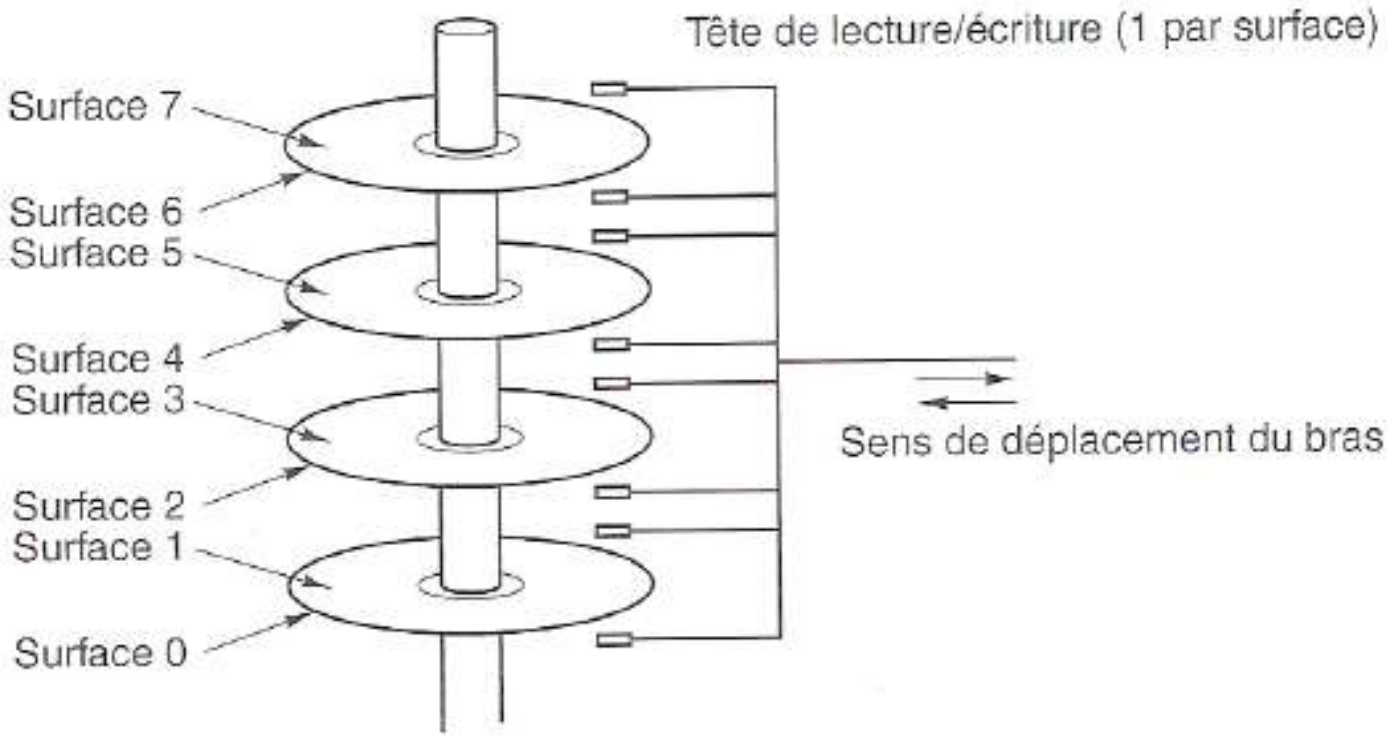


• Deux secteurs d'une piste de disque dur.

- Chaque disque dispose d'un bras mobile capable de se déplacer de façon radiale pour venir positionner la tête à une certaine distance de l'axe de rotation du plateau. A chaque distance radiale correspond une piste différente pouvant être écrite ou lue.



• Deux secteurs d'une piste de disque dur.



- Disque à quatre plateaux.

CD-ROM

- Un CD-ROM (abréviation de Compact Disc - Read Only Memory) est un disque optique utilisé pour stocker des données sous forme numérique. Ces données sont gravées une fois pour toute lors de sa fabrication. Le CD-ROM peut contenir de 650 ou 700 Mo de données.
- Les CD-ROM sont largement utilisés pour la distribution des logiciels, des livres, des films et des données de toutes sortes, ainsi que pour la réalisation de copies de sauvegarde des disques durs.
- Les disques compacts sont fabriqués à partir d'un disque en polycarbonate contenant des microcuvettes et des zones planes éclairées par une diode laser et lues par un photodétecteur.

DVD

- DVD (Digital Vaersatile Disk) C'est un disque optique numérique exploité pour la sauvegarde et le stockage de données. Les spécifications générales des DVD sont les mêmes que celles des CD. Parmi les principales différences techniques on trouve :
  - les microcuvettes qui sont plus petites que celles des CD,
  - le laser est rouge (longueur d'onde plus courte que celle utilisé pour les CD (diode laser qui émet des rayons infrarouges)).

  Les capacités d'un DVD dépendent de ses spécificités :
  - Simple face simple couche (4,7 GO) ;
  - Simple face double couche (8,5 GO) ;
  - double face simple couche (9,4 GO) ;
  - double face double couche (17 GO) ;

Blu-Ray

- Blu-Ray (abréviation officielle BD, autre dénomination B-RD (Blu-ray Disc))
- Successeur du DVD, sa dénomination provient du type de rayon laser qu'il exploite, de couleur spectrale proche du bleu (blue), et non rouge comme celui des DVD. Les lasers bleus sont de longueur d'onde plus courte.
- Les disques Blu-ray contiennent plus de données : environ 25 Go pour les simples faces et 50 Go pour les doubles faces.

Clé USB

- C'est un support fiable et très pratique qui s'utilise comme un lecteur de disque externe et qui se connecte directement sur le port USB (Universal Serial Bus).

Architecture d'un ordinateur

La structure générale d'un ordinateur est constituée par quatre blocs fondamentaux :

- **Carte mère** : circuit principal de l'ordinateur
- **Unité Centrale de traitement** : processeur, Cœur de l'ordinateur
- **Mémoire centrale** : sert à stocker les données et les programmes
- **Interfaces d'entrées/sorties** : permettent de raccorder les périphériques externes d'un ordinateur.

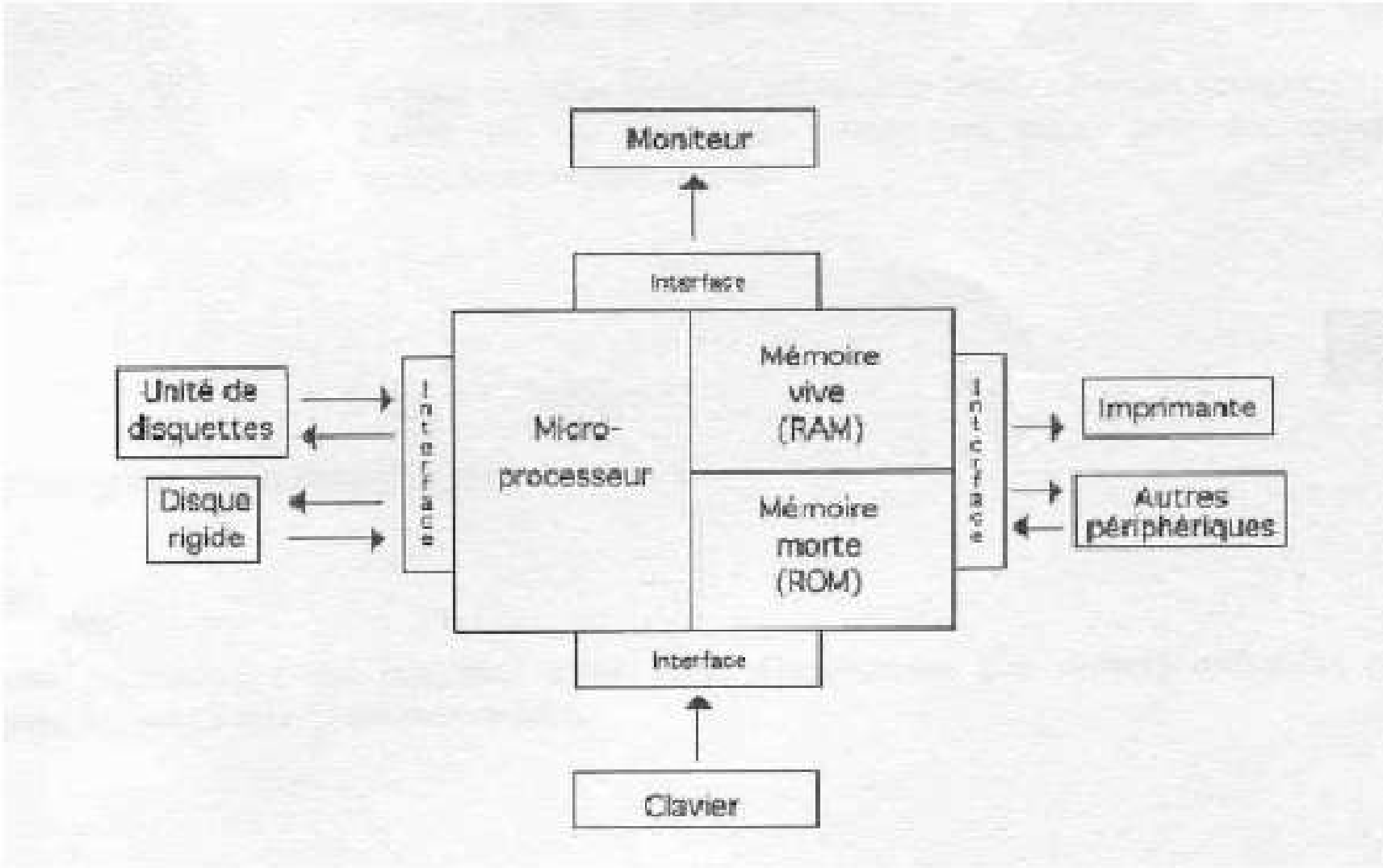
Périphériques d'Entrées/Sorties

- Ces unités permettent d'échanger les informations entre l'ordinateur et le milieu extérieur.
- Chaque appareil d'entrée/sortie est composée de deux parties :
  - Le contrôleur (interface d'entrée/sortie) : Il a pour fonction de commander le périphérique d'entrée/sortie auquel il est associé et de gérer son accès au bus. Lorsqu'un contrôleur est capable de lire ou d'écrire des données directement dans la mémoire sans l'intervention du processeur, on parle d'accès direct à la mémoire, ou DMA (Direct Memory Access).
  - Le boîtier contenant l'appareil lui-même (périphérique d'entrée/sortie), par exemple un modem.

Périphériques d'Entrées/Sorties

- Ils permettent à l'ordinateur de communiquer avec l'extérieur (utilisateur ou autre ordinateur)
  - Périphériques d'entrée : Clavier, Souris, Scanner, Ecran tactile, carte réseau, mémoires de masse, etc.
  - Périphériques de sortie : Ecran (tactile ou non), Imprimante, carte réseau, mémoires de masse, etc.

Récapitulatif : Architecture de Von Neumann



1 Qu'est ce que l'Informatique

2 Qu'est ce qu'un Ordinateur : coté matériel

3 Qu'est ce qu'un Ordinateur : coté logiciel

4 Codage de l'information

Classifications des logiciels

Les logiciels sont classés en trois catégories :

- Les systèmes d'exploitation
- Les langages de programmation
- Les logiciels d'applications

Qu'est-ce qu'un SE ?

- Les composantes d'un ordinateur et les périphériques ne sont au final qu'une collection de composantes électroniques et mécaniques.
- Pour que ces composantes réalisent une tâche donnée, un programme particulier, appelé système d'exploitation (OS pour Operating System) est nécessaire.

- Un SE est un logiciel chargé d'organiser et de contrôler le fonctionnement de l'ordinateur. C'est lui qui détermine :
  - quel programme utilisateur va être exécuté

- Un SE est un logiciel chargé d'organiser et de contrôler le fonctionnement de l'ordinateur. C'est lui qui détermine :
  - quel programme utilisateur va être exécuté
  - comment répartir la mémoire entre les différents programmes

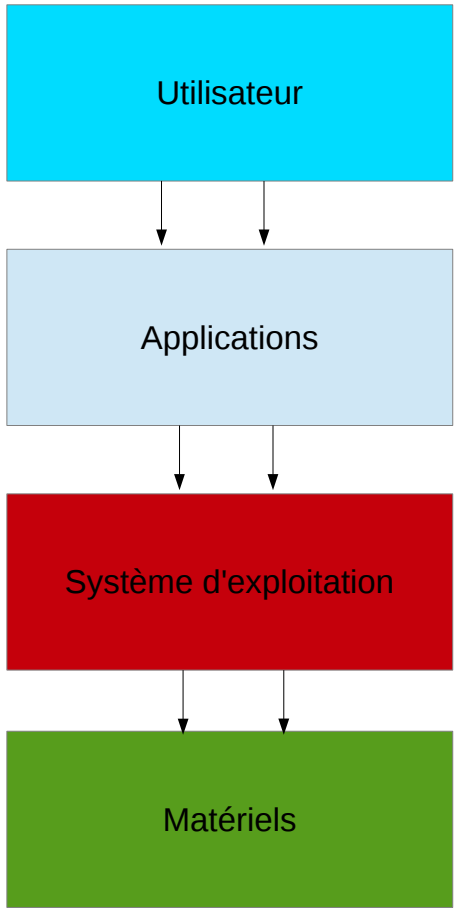
- Un SE est un logiciel chargé d'organiser et de contrôler le fonctionnement de l'ordinateur. C'est lui qui détermine :
  - quel programme utilisateur va être exécuté
  - comment répartir la mémoire entre les différents programmes
  - comment lire/enregistrer les données sur les mémoires de masse

- Un SE est un logiciel chargé d'organiser et de contrôler le fonctionnement de l'ordinateur. C'est lui qui détermine :
  - quel programme utilisateur va être exécuté
  - comment répartir la mémoire entre les différents programmes
  - comment lire/enregistrer les données sur les mémoires de masse
  - les droits de chaque utilisateur du système

- Un SE est un logiciel chargé d'organiser et de contrôler le fonctionnement de l'ordinateur. C'est lui qui détermine :
  - quel programme utilisateur va être exécuté
  - comment répartir la mémoire entre les différents programmes
  - comment lire/enregistrer les données sur les mémoires de masse
  - les droits de chaque utilisateur du système
- Le SE sert d'intermédiaire entre l'utilisateur et la machine :

- Un SE est un logiciel chargé d'organiser et de contrôler le fonctionnement de l'ordinateur. C'est lui qui détermine :
  - quel programme utilisateur va être exécuté
  - comment répartir la mémoire entre les différents programmes
  - comment lire/enregistrer les données sur les mémoires de masse
  - les droits de chaque utilisateur du système
- Le SE sert d'intermédiaire entre l'utilisateur et la machine :
  - But : rendre les machines plus faciles à utiliser et par conséquent la tâche de l'utilisateur moins fastidieuse

Qu'est-ce qu'un SE ?



Qu'est-ce qu'un SE ?

Autrement dit, un SE est :

- Un superlogiciel qui orchestre l'ordinateur.
- Le SE fonctionne exactement comme un programme ordinaire :
  - Il est exécuté par le processeur de la même manière.
  - La différence principale est sa fonction : il dirige le processeur sur l'utilisation des ressources et la manière d'exécuter les autres programmes.

Missions d'un SE

- **Permettre d'accéder au matériel de façon transparente :**

Missions d'un SE

- **Permettre d'accéder au matériel de façon transparente :**
  - un programme n'a pas à savoir s'il écrit sur un disque NTFS ou une clé USB FAT32.

Missions d'un SE

- **Permettre d'accéder au matériel de façon transparente :**
  - un programme n'a pas à savoir s'il écrit sur un disque NTFS ou une clé USB FAT32.
- **Gérer les ressources (accès aux disque, mémoire, CPU) :**

Missions d'un SE

- **Permettre d'accéder au matériel de façon transparente :**
  - un programme n'a pas à savoir s'il écrit sur un disque NTFS ou une clé USB FAT32.
- **Gérer les ressources (accès aux disque, mémoire, CPU) :**
  - optimiser l'usage de la machine
    - taux d'occupation du processeur,
    - gestion de l'énergie sur les systèmes portables,
    - etc

Missions d'un SE

- **Veiller à la sécurité des applications et des données**

Missions d'un SE

- **Veiller à la sécurité des applications et des données**
- **Fournir une qualité de service :**

Missions d'un SE

- **Veiller à la sécurité des applications et des données**
- **Fournir une qualité de service :**
  - garantir un accès, prioritaire au root (utilisateur principal),
  - temps de réponse maximal sur un SE temps réel, etc

Missions d'un SE

- **Veiller à la sécurité des applications et des données**
- **Fournir une qualité de service :**
  - garantir un accès, prioritaire au root (utilisateur principal),
  - temps de réponse maximal sur un SE temps réel, etc
- **Être robuste :**

Missions d'un SE

- **Veiller à la sécurité des applications et des données**
- **Fournir une qualité de service :**
  - garantir un accès, prioritaire au root (utilisateur principal),
  - temps de réponse maximal sur un SE temps réel, etc
- **Être robuste :**
  - tolérance à l'erreur (blocs disques défectueux, reboot sauvage, etc)
  - éviter les blocages !
  - éviter les famines !

Blocage vs Famine

- Blocage
  - Le blocage ou interblocage peut se produire quand les ressources requises pour certains programmes sont utilisées par d'autres qui en revanche attendent eux aussi des ressources utilisées par les premiers. Dans ce cas de il faut que le système d'exploitation intervienne d'autorité en retirant une ressource à l'un des processus.
- Famine
  - La famine est la situation d'un programme qui reste indéfiniment bloqué dans l'attente d'une ressource sans pour autant être en situation d'interblocage.

Types de SE

- Mono utilisateur
  - SE d'un téléphone portable
- Multi-utilisateurs :
  - protéger les données de chacun sur les supports de stockage
  - notion de droits d'accès
  - protège les utilisateurs entre eux
- Multi-taches :
  - protéger les processus les uns des autres
  - notion de protection de la mémoire
  - notion du noyau (le seul capable d'accéder à la mémoire)
- Temps réel
  - Pour développer des systèmes embarqués (vaisseaux spatiaux,...)
  - garantit un délai maximal d'exécution même si les performances ne sont pas optimales.

Le concept "noyau"

Le noyau du SE est :

- un espace mémoire protégé
- l'ensemble de programmes qui forment la base minimale du SE

Taches impotantes du noyau :

- Ordonnanceur des processus (programmes)
- Gestionnaire de la mémoire
- Gestionnaire des fichiers

Différents types de noyaux :

- Monolithique : tout (système de fichiers, pilotes, etc) est dans le noyau (Linux)
- Micro-noyau : le nécessaire (ordonnanceur, mémoire virtuelle) est dans le noyau (Mac OS)
- Hybride : tout est dans le noyau mais avec des émulateurs dans le mode utilisateurs (Windows NT)

Exemples de systèmes d'exploitation

- MS-DOS (Disk Operating System)
- Microsoft Windows : XP, 2003 Server , Vista
- UNIX-Based : IBM AIX, Hewlett Packard HPUX, and Sun Solaris
- Linux-Based : Ubuntu, Fedora, Debian
- Macintosh OS X

Le cas Linux

Pourquoi Linux ? ?

- Libre et gratuit
- Portable (MacBook, PC, smartphone,...)
- Convivial
- Académique :
  - Notion de processus, de droits d'accès, de mémoire virtuelle,...

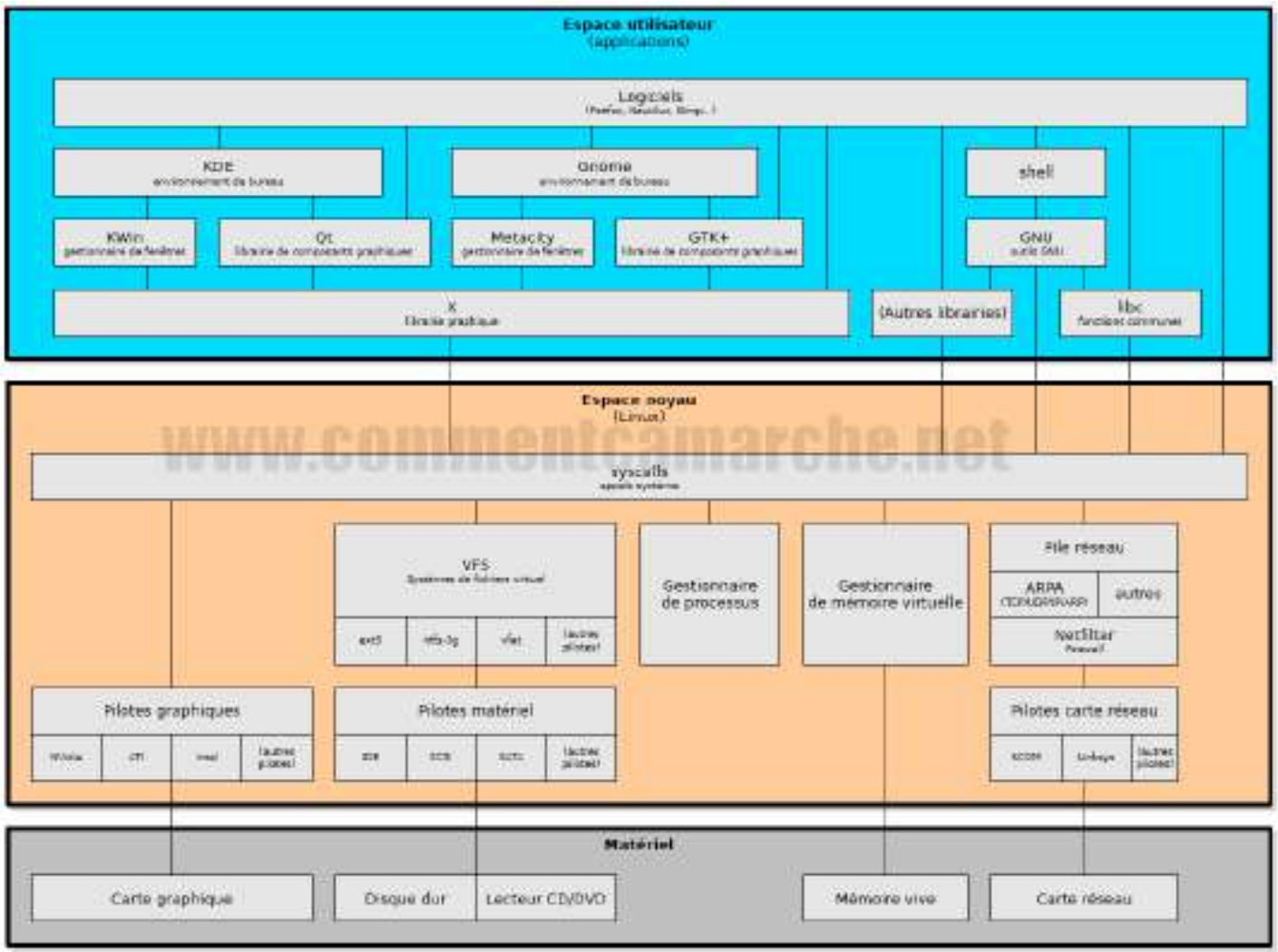
Histoire de Linux

- 1969 : UNIX (Thompson et Ritchie)
- 1973 : Réecriture en langage C
- 1977 : Début du marché.. Unix devient de plus en plus cher.
- 1984 : Création du système libre et gratuit GNU (Stallman)
- 1991 : Premier noyau Linux (Torvalds)
- 1994 : v1.0 qui donne le premier système GNU/Linux

Architecture Linux

- Noyau monolithique
- Séparation en deux espaces distincts :
  - Espace utilisateur (user land)
  - espace noyau (kernel land)

Architecture Linux



Architecture Linux

- **Partie matérielle** : fournit au système les services de base.
- **Noyau (SE)** : fournit les services communs aux programmes et les isole des particularités du matériel.
  Plus facile de les utiliser sur des machines différentes.
- **Espace utilisateur** :
  - Shell (commandes) : comme sh, who, grep,... qui font partie des configurations standards.
  - Interface graphique : comme GNOME, KDE, ...

Architecture Linux

Les programmes de couches hautes tels que shell interagissent avec le noyau via des **appels systèmes**.

Les appels système demandent au noyau de réaliser plusieurs opérations pour le programme appelant, et d'échanger les données entre le noyau et le programme.
Quelques appels systèmes classiques :

- *open*, *read*, *write* et *close* qui permettent les manipulations sur les systèmes de fichiers
- *brk*, *sbrk*, utilisés par *malloc* et *free* pour allouer et désallouer de la mémoire.

Critères de choix d'un bon système d'exploitation

Un bon système d'exploitation doit être :

- **Fiable** : le SE doit intégrer des mesures de sécurité (pannes ou pertes d'informations).
- **Commode** : abordable et assez aisé d'utilisation.
- **Adaptable** (portabilité) : les concepteurs des SE essaient de faire en sorte que leur logiciel (programme) soit portable sur le plus de matériel possible.
- **Efficace** : le SE doit répondre de manière efficace aux besoins des utilisateurs.

SE : gestion de l'ordinateur

Gestion sur 3 niveaux :

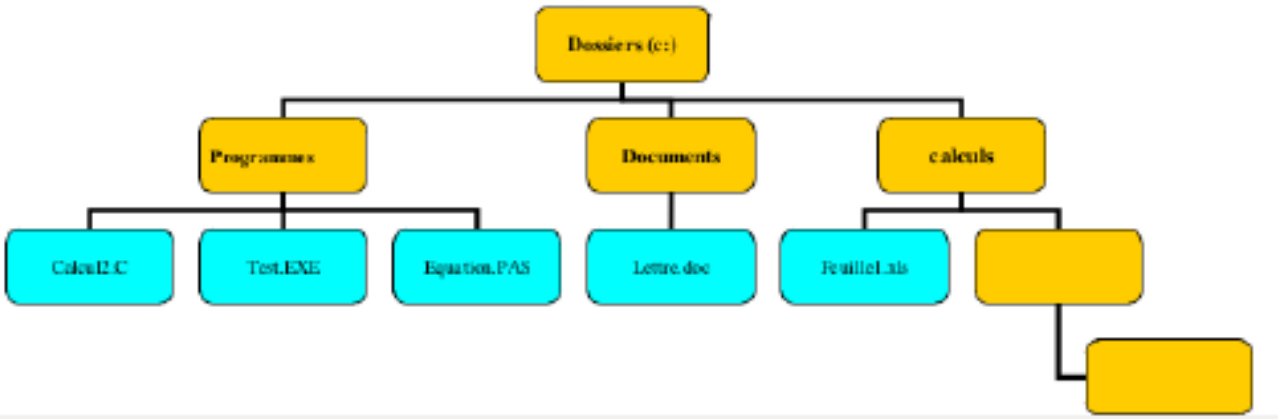
- Gestion des fichiers (et dossiers)
- Gestion de la mémoire
- Gestion des processus

Fichiers et système de gestion de fichiers

- **Fichier** :ensemble d'information défini principalement par un nom et un type,
- Les fichiers sont stockés et organisés dans une mémoire de masse (une unité de disque le plus souvent).
- Propriétés d'un fichier :
  - Nom et extension (TXT, DOC....)
  - date de création ou de mise à jour
  - Autorisation d'exploitation, taille, etc.
- Exemples :
  - Fichiers texte
  - Fichiers image
  - Fichier son
  - Document Microsoft Word
  - Feuille de calcul Microsoft Excel
  - Document html

Dossier (répertoire)

- Conteneur de fichiers et d'autres dossiers
- Les dossiers servent à classer les fichiers
- Avantages :
  - Possibilité de mettre des dossiers à l'intérieur d'autres dossiers et de créer une arborescence
  - C'est ce qui permet d'organiser (de hiérarchiser) des fichiers.



Fichiers et système de gestion de fichiers

- Un SGF est une façon de stocker les informations et de les organiser dans des fichiers
- C'est la partie la plus visible d'un système d'exploitation qui se charge de la manipulation de fichiers

Rôles d'un SGF

Un SGF a pour missions :

- **Fournir une interface conviviale pour manipuler les fichiers**
  - simplifier la gestion des fichiers pour l'utilisateur
  - l'utilisateur fournis seulement les attributs nom et extension du fichier, les autres attributs sont gérés implicitement par le SGF
- **Gérer l'organisation des fichiers sur le disque**
  - allocation de l'espace disque aux fichiers
- **Gérer l'espace libre sur le disque dur**

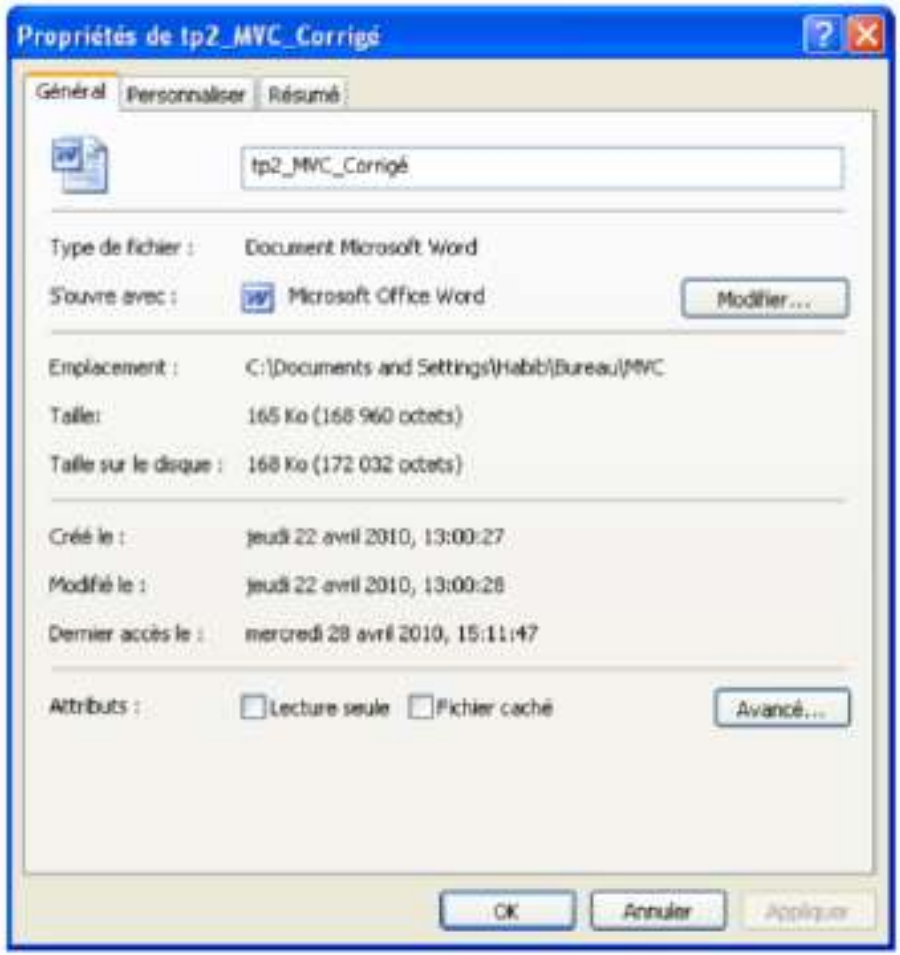
Opérations sur les dossiers et les fichiers

- Fichiers :
  - Lire le contenu
  - Écrire
  - Exécuter (pour les programmes)
  - Modifier
  - Cacher
  - Copier
- Dossiers :
  - Accéder au dossier
  - Lister son contenu
  - Créer un élément dedans
  - Supprimer un élément
  - Supprimer le dossier lui-même

Droits d'accés- DOS/Windows

- Attributs plutôt que droits d'accès :
  - Lecture seule (R)
  - Système (S)
  - Archive (A)
- Droit en exécution géré par l'extension

Droits d'accés- DOS/Windows



Droits d'accès- UNIX

- Droits d'accès
  - Lire (**R**)
  - Ecrire (**W**)
  - Exécuter (**X**)
- 3 classes d'utilisateurs :
  - Le propriétaire (Owner)
  - Le groupe d'utilisateurs (auquel appartient le propriétaire) (Group)
  - Le reste (**Other**)
- 9 informations par fichiers (**rwxrwxrwx**)

SE : gestion de l'ordinateur

Gestion sur 3 niveaux :

- Gestion des fichiers (et dossiers)
- Gestion de la mémoire
- Gestion des processus

Gestion de la mémoire

- Chaque programme qui s'exécute doit être présent en mémoire accompagné des données sur lesquelles il travaille.
- A chaque fois qu'un programme est lancé, le gestionnaire de mémoire est appelé pour 'trouver' un espace mémoire pour ce Programme.
- La plupart des systèmes permettent de lancer plusieurs programmes en parallèle

Gestion de la mémoire

- Le gestionnaire de mémoire masque la localisation physique de la mémoire (en mémoire vive, sur disque dur ou usb).
- Il présente au programme une mémoire globale uniforme dite mémoire virtuelle.
- Ainsi, tout processus croit manipuler une mémoire "logique" qui peut être étendue jusqu'aux capacités théoriques de la machine.

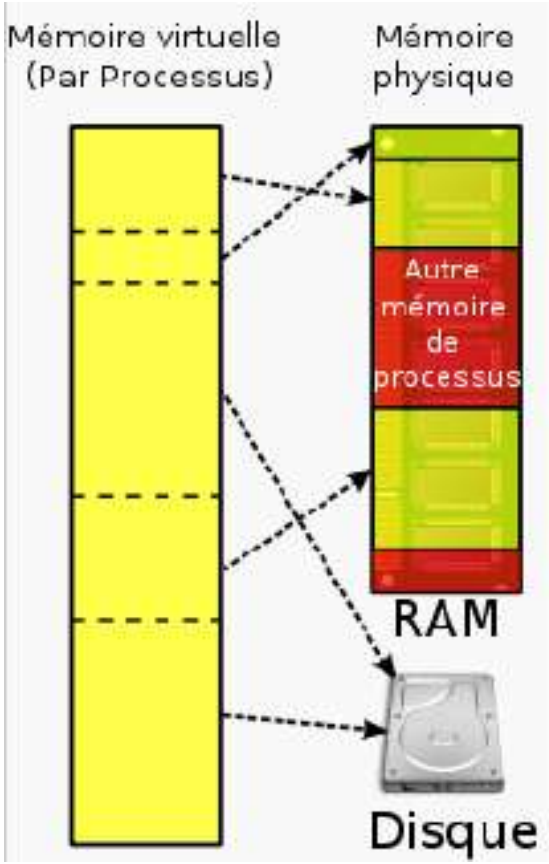
Gestion de la mémoire : Mémoire virtuelle

- Imaginons le multi-tâche sur mémoire physique
  - qu'est-ce qui se passe si un programme a besoin de plus de mémoire lors de son exécution ?
  - si un autre programme a besoin de plus de mémoire que disponible sur la machine ?
  - et si un troisième programme est là mais n'utilise pas sa mémoire ?

L'intérêt de ne pas indiquer au processus l'emplacement physique des données est de :

- Permettre au gestionnaire de mémoire de placer et déplacer à sa convenance les données en mémoire, sans affecter les processus :
  - Ces données sont fragmentées dans la mémoire vive lorsqu'un processus demande un bloc de mémoire d'une taille supérieure au plus grand bloc physique libre.
  - Le contenu de la mémoire peut aussi être migré sur les différentes mémoires secondaires.
  - La mémoire allouée mais pas encore utilisée peut être virtuellement allouée à plusieurs processus (cas du noyau linux).

Mémoire virtuelle



Buts de la mémoire virtuelle

- donner à chaque programme son espace d'adressage
  - convertir chaque accès mémoire virtuelle en un accès mémoire physique
  - l'application ne se soucie pas de quelle mémoire physique elle utilise
- garantir la protection
  - empêche une application de toucher à la mémoire d'une autre
- permettre aux programmes de voir plus de mémoire qu'il n'en existe

SE : gestion de l'ordinateur

Gestion sur 3 niveaux :

- Gestion des fichiers (et dossiers)
- Gestion de la mémoire
- Gestion des processus

Gestion des processus : Programme vs Processus

- Un programme est une suite d'instructions (un objet statique).
- Un processus est un programme en exécution (un objet dynamique) :
  - Dans un environnement monotâche la notion de processus est réduite à sa simple exécution.
  - Dans un système multitâches (ex : Linux, UNIX), plusieurs processus s'exécutent "simultanément". Ils doivent se partager l'accès au processeur.

État d'un processus

- Lors de son exécution un processus change d'état :
  - nouveau : le processus vient d'être créé
  - actif (running) : le processus est en train d'être exécuté par l'UCT
  - attente (waiting) : le processus est en train d'attendre un événement (p.ex. la fin d'une opération d'E/S)
  - prêt (ready) : le processus est en attente d'être exécuté par l'UCT
  - terminé : fin d'exécution

États et transitions entre processus

- **Nouveau :**
  - Le SE crée le processus
  - Mais ne l'exécute pas encore
- **Prêt $\rightarrow$ Exécution :**
  - Lorsque l'ordonnanceur UCT choisit un processus pour exécution
- **Exécution $\rightarrow$ Prêt :**
  - Résultat d'une interruption causée par un événement indépendant du processus
  - Il faut traiter cette interruption, donc le processus courant perd l'UCT

- **Exécution → Attente :**
  - Lorsqu'un processus fait requête d'un service du SE que le SE ne peut offrir immédiatement (interruption causée par le processus lui-même)
    - un accès à une ressource pas encore disponible
    - initie une E/S : doit attendre le résultat
    - a besoin de la réponse d'un autre processus
- **Attente → Prêt :**
  - lorsque l'événement attendu se produit
- **Terminé :**
  - Le processus n'est plus exécutable, mais ses données sont encore requises par le SE (comptabilité, etc.)

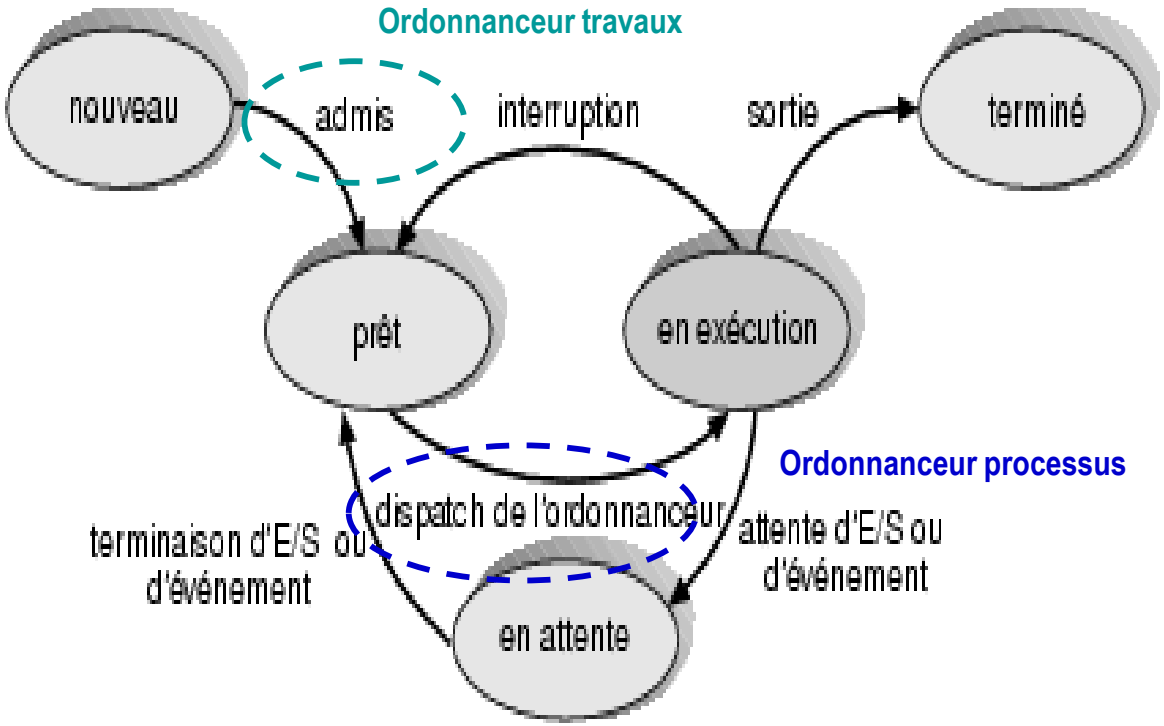
Ordonnanceur

- Le programme responsable de gérer l'utilisation des ressources de l'ordinateur
- Trois types d'ordonnanceurs :
  - **À court terme** = ordonnanceur processus :
    - sélectionne quel processus doit exécuter la transition prêt → exécution
  - **À long terme** = ordonnanceur travaux :
    - sélectionne quels processus peuvent exécuter la transition nouveau → prêt
  - **À moyen terme**= répond au manque de mémoire

Ordonnanceur

- L'ordonnanceur à court terme est exécuté très souvent
  - Il faut donc que ça aille très vite (de 1 à 1000 microsecondes)
- L'ordonnanceur à long terme est exécuté beaucoup plus rarement : il contrôle le niveau de multiprogrammation
  - décide des processus que le système peut mener en parallèle.
  - ils doivent être assez nombreux pour que le processeur soit inactif le plus rarement possible
  - mais pas trop abondants et saturer la mémoire principale du système.

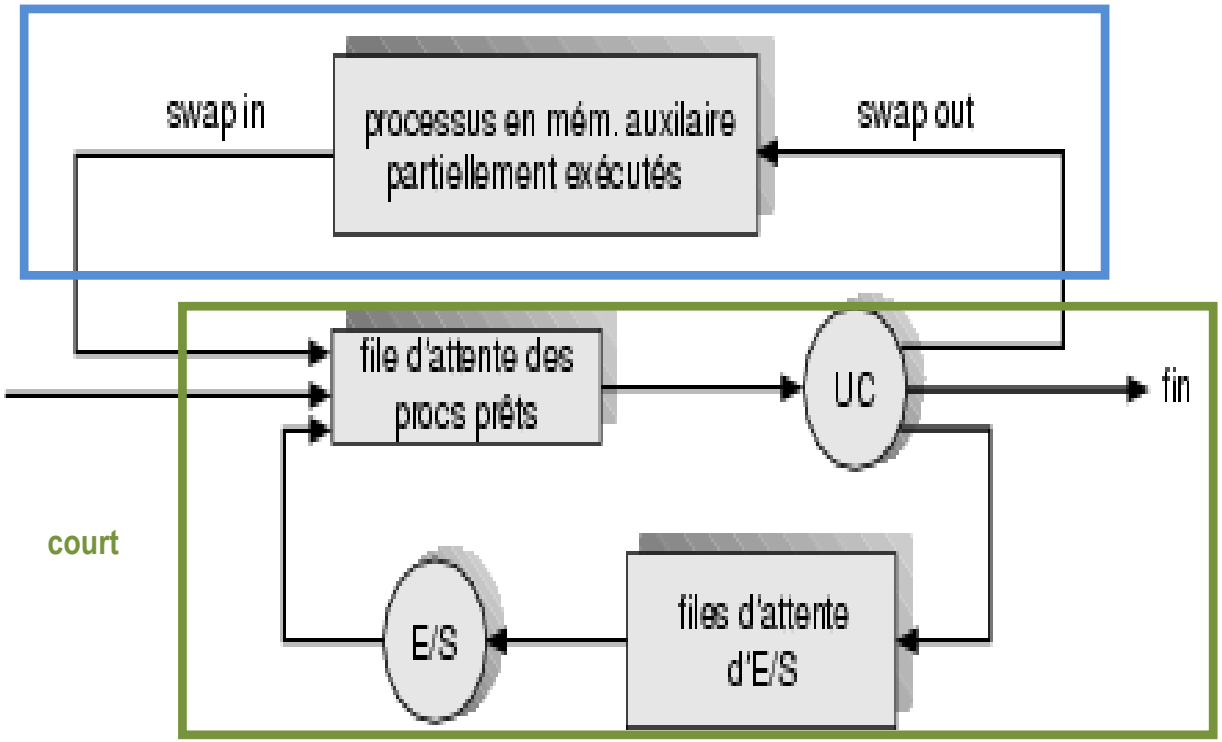
Ordonnanceurs : long terme (travaux) et court terme (processus)



Ordonnanceur à moyen terme

- Le SE est parfois obligé de suspendre des processus (pour manque de ressources). Ces processus :
  - ne seront plus en concurrence avec les autres pour des ressources
  - seront repris plus tard quand les ressources deviendront disponibles
- Ces processus sont enlevés de la mémoire centrale et mis en mémoire secondaire, pour être repris plus tard
  - va-et-vient, swap-in swap-out

Ordonnanceurs : court terme et moyen terme



Quand faire appel à l'ordonnanceur ?

- Choisir un processus parmi ceux qui sont prêts et lui donner les ressources.
- L'ordonnancement a lieu quand un processus :
  1. Se termine.
  2. Passe de l'état *actif* à *attente*.
  3. Passe de l'état *actif* à *prêt*.
  4. Passe de l'état *attente* à *prêt*.

Critères d'ordonnancement

- Il y a normalement plusieurs processus dans la file des prêt
- Quand le processeur devient disponible, lequel choisir ?
- L'idée générale est d'effectuer le choix dans l'intérêt de l'efficacité d'utilisation de la machine
- Différentes méthodes d'ordonnancement

Ordonnancement PAPS (FIFO)

Premier Arrivé Premier Servi
First In First Out

- Les processus prêt sont stockes dans une FIFO et servis par ordre d'arrivee

Processus	Temps d'exécution
P1	24
P2	3
P3	3

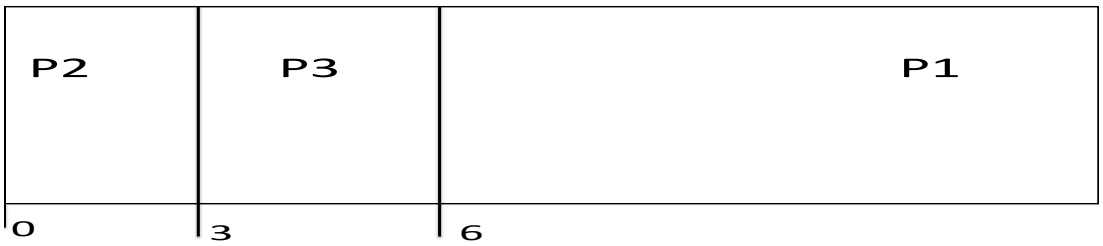
Exemple du PAPS

- Ordre d'arrivée des processus : P1, P2, P3
- Diagramme d'ordonnancement :

- Temps d'attente : P1 :0, P2 :24, P3 :27
- temps d'attente moyen : (0+24+27)/3=17

Autre exemple

- Ordre d'arrivée des processus : P2, P3, P1
- Diagramme d'ordonnancement :



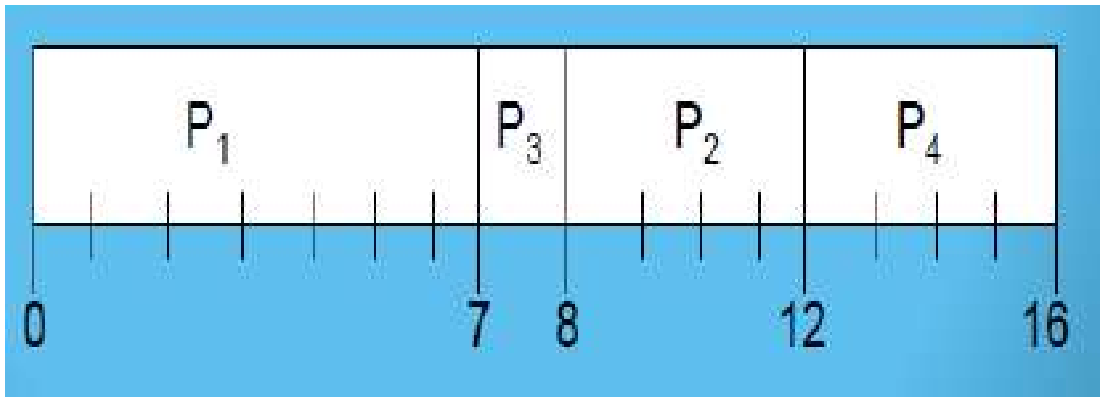
- Temps d'attente : P1 :6, P2 :0, P3 :3
- temps d'attente moyen : (6+0+3)/3=3

Ordonnancement PCTE (Plus court temps d'exécution) SJF (Shortest Job first)

- Le CPU est attribué au processus qui a le plus petit temps d'exécution (en utilisant PAPS en cas d'égalité)
- PCTER : optimal pour le temps d'attente moyen.

Exemple PCTE

Processus	Arrivée	Durée
P1	0	7
P2	2	4
P3	4	1
P4	5	4



Temps moyen d'attente= (0+6+3+7)/4=4

Qu'est ce qu'un programme ?

- Un **programme informatique** est une liste d'ordres indiquant à un ordinateur ce qu'il doit faire.
- Il se présente sous la forme d'une ou plusieurs séquences d'instructions, comportant souvent des données de base, devant être exécutées dans un certain ordre par le processeur
- Écrire un programme, c'est écrire une suite d'instructions élémentaires s'enchaînant les unes après les autres pour réaliser un traitement sur des données.

Qu'est ce qu'un algorithme

- Un **algorithme** énonce une résolution sous la forme d'une série d'opérations à effectuer.
- La mise en œuvre de l'algorithme consiste en l'écriture de ces opérations dans un **langage de programmation**

Qu'est ce qu'un langage de programmation ?

- Ce sont des logiciels qui permettent de produire d'autres logiciels
- Un langage de programmation permet d'exprimer les indications nécessaires aux traitement, de concevoir et de rédiger les programmes souhaités.
- Exemples :
  - Programmation procédurale : Pascal, C , Visual Basic,....
  - Programmation logique : Lisp, Prolog,....
  - Programmation Scientifique : Matlab, Maple...

Le langage Assembleur

- Les programmes informatiques sont codés dans la mémoire centrale et les mémoires de masse sous forme de bits. On les appels des programmes **"exécutables"** (directement prêts à être exécutés).
- Les langages de programmation les plus proches de ce codage de «bas niveau» sont appellés «langages d'assemblage» ou «Assembleurs».

Le langage Assembleur

- Programmer dans un tel langage nécessite de connaître l'architecture matérielle de l'ordinateur sur lequel il s'exécutera
- il y a presque un langage d'assemblage différent pour chaque microprocesseur.

Le langage Assembleur

Exemple : processeur x86

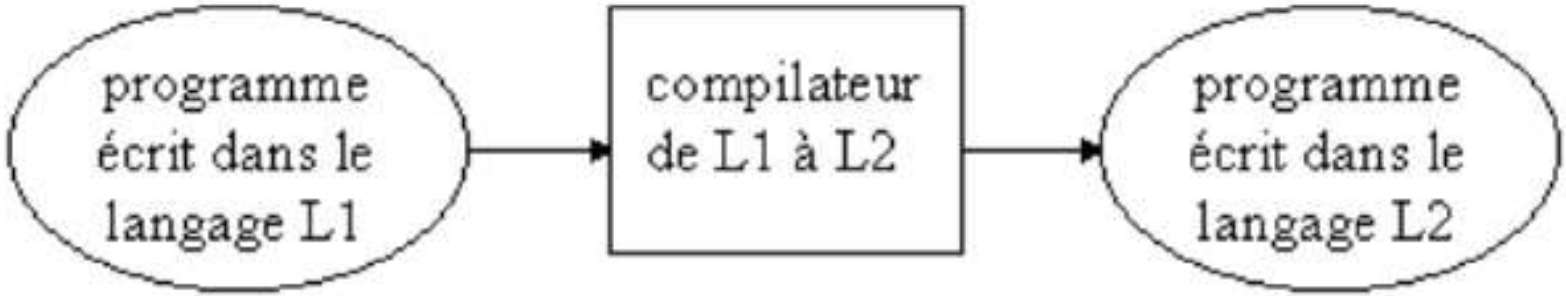
- en langage machine : `10110000 01100001`
- en langage d'assemblage : `mov %al,$0x61`
- en langage naturel : charger la valeur hexadécimale 61 dans le registre 'AL'

Langages évolués

- Les programmes écrits en Assembleur se traduisent immédiatement en programmes exécutables. Mais programmer en Assembleur n'est ni simple ni agréable.
- Des langages de programmation plus compréhensibles par l'être humain ont vu le jour.
- problème : Comment définir une correspondance, une méthode de traduction entre ce nouveau langage et un langage d'assemblage.

Langages évolués : rôle du compilateur

- Un compilateur est un logiciel capable de transformer un programme écrit dans un langage de programmation donné L1 en un programme réalisant le même traitement mais écrit dans un autre langage L2 (en général le langage machine ou le langage Assembleur)



- L'utilisation de compilateurs a permis la définition de langages de programmation de haut niveau ou évolué.
- Dans de tels langages, on peut programmer sans connaître le matériel de l'ordinateur sur lequel s'exécutera le programme pour se concentrer sur sa seule logique.
- Exemple sous Pascal :

```
BEGIN
nombre3 := nombre1 + nombre2;
nombre5 := nombre3 * nombre4;
END.
```

Notion de l'interprète

- Outil ayant pour tâche d'analyser, de traduire et d'exécuter un programme écrit dans un langage informatique
- On différencie un programme dit script, d'un programme dit compilé :
  - Un programme script est exécuté a partir du fichier source via un interpréteur de script.
  - Un programme compilé est exécuté a partir d'un bloc en langage machine issu de la traduction du fichier source.
- Le cycle d'un interprète est le suivant :
  - lire et analyser une instruction (ou expression) ;
  - si l'instruction est syntaxiquement correcte, l'exécuter (ou évaluer l'expression) ;
  - passer à l'instruction suivante.

Langages Compilés ou Interprétés

- Avantage des langages interprétés
  - facilité de programmation
    - possible d'exécuter des programmes incomplets
  - Portabilité
    - le même programme est exécutable sur n'importe quelle machine où est disponible l'interprète
    - un code compilé n'est exécutable que sur les machines compatibles avec celle où il a été produit
- Inconvénients :
  - relative lenteur par rapport au code compilé

Historique

- FORTRAN :
  - Ce fut le premier langage destiné à permettre l'écriture de programmes
  - Il était principalement destiné à faciliter le transcodage des formules mathématiques (d'où son nom FORmula TRANslation)
  - Le FORTRAN standard a été défini en 1966 (FORTRAN 66) et a été revu en 1978 (FORTRAN 77)
  - FORTRAN ne permet pas de manipuler autre chose que des nombres
  - FORTRAN ne possède pas les qualités requises pour incorporer les idées nouvelles comme :
    - la programmation structurée
    - la récursivité
    - les pointeurs

Historique

- BASIC :
  - Beginner All purpose Symbolic Instruction Code
  - Conçu au Darthmouth College (U.S.A.) en 1967 pour faciliter l'apprentissage de la programmation aux étudiants
  - Les auteurs du BASIC ont construit un langage très facile à implémenter sur les machines de l'époque
  - Depuis 1967, la syntaxe du BASIC a fortement évolué.
  - Il y a eu plusieurs versions de BASIC
  - Les dernières versions rendent le langage complexe et l'éloignent de son rôle premier qui était de fournir un langage simple destiné à résoudre de petits problèmes.

- PASCAL :
  - En même temps que naissait FORTRAN, un comité d'experts tentait de définir un nouveau langage qui, tout en restant simple et efficace, inclurait les notions importantes de structuration des algorithmes (for, while, case, etc...)
  - Il a était par la suite, adopté comme le langage fondamental pour l'enseignement, grâce a sa structuration (meilleure que FORTRAN)

Historique

- Le Langage C :
  - Le "C" fut créé en 1978 au laboratoire Bell pour implémenter le système d'exploitation UNIX
  - La force du "C" réside dans le fait qu'il allie les avantages d'un langage structuré de haut niveau comme le Pascal à l'efficacité du langage assembleur.
  - Il reste néanmoins un langage pour les programmeurs chevronnés plutôt que pour les débutants.
  - Le texte d'un programme écrit en "C" est habituellement assez difficile à comprendre

Historique

- JAVA :
  - JAVA est un langage informatique récent qui intègre les nouvelles possibilités de programmation offertes par le réseau Internet.
  - JAVA repose sur les techniques de programmation les plus modernes.
  - Il est complètement objet-orienté et inclut, entre autres, des mécanismes permettant la programmation en parallèle

1 Qu'est ce que l'Informatique

2 Qu'est ce qu'un Ordinateur : coté matériel

3 Qu'est ce qu'un Ordinateur : coté logiciel

4 Codage de l'information

Introduction

- Les informations traitées par un ordinateur peuvent être de différents types (texte, nombres, etc.)
- Mais elles sont toujours représentées et manipulées par l'ordinateur sous forme binaire.
- Toute information sera traitée comme une suite de 0 et de 1.
- L'unité d'information est le chiffre binaire (0 ou 1), que l'on appelle bit (pour binary digit, chiffre binaire).

Codage de l'information : Définition

- Codage d l'information :
  - Le codage d'une information consiste à établir une correspondance entre la représentation externe (habituelle) de l'information, et sa représentation interne dans la machine, qui est une suite de bits.
- Exemple :
  - Le nombre 35 : 35 est la représentation externe du nombre trente cinq.
  - La représentation interne de 35 sera une suite de 0 et 1 (100011).

Codage de l'information : Les étapes

Le codage de l'information s'effectue principalement en trois étapes :

- L'information sera exprimée par une suite de nombres **(Numérisation)**
- Chaque nombre est codé sous forme binaire **(suite de 0 et 1)**
- Chaque élément binaire est représenté par un **état physique**

Élément binaire ⇒ État physique

Codage de l'élément binaire par un état physique

- Charge électrique (RAM : Condensateur-transistor) : Chargé (bit 1) ou non chargé (bit 0)
- Magnétisation (Disque dur, disquette) : polarisation Nord (bit 1) ou Sud (bit 0)

Numérisation ⇒ Binaire

- On utilise la représentation binaire car elle est simple, facile à réaliser techniquement (juste **deux** états physiques).
- En plus, les opérations arithmétiques de base (addition, multiplication, etc.) sont faciles à exprimer en base 2 (noter que la table de multiplication se résume à 0x0 = 0, 1x0 = 0 et 1x1 = 1).

Système de numération

- Système de numération décrit la façon avec laquelle les nombres sont représentés.
- Un système de numération est défini par :
  - Un alphabet $\mathcal{A}$: ensemble de symboles ou chiffres,
  - Des règles d'écritures des nombres : Juxtaposition de symboles

Système de numération : Exemples

Numération Romaine :

système romain	I	V	X	L	C	D	M
valeur décimal	1	5	10	50	100	500	1000

- Lorsqu'un symbole est placé à la droite d'un symbole plus fort que lui, sa valeur s'ajoute : CCLXXI→271
- Lorsqu'un symbole est placé à la gauche d'un symbole plus fort que lui, on retranche sa valeur : CCXLIII→243
- On ne place jamais 4 symboles identique à la suite : 9 s'écrit IX et non VIIII
- Le plus grand nombre exprimable est : 3999 (MMMCMXCIX) Système inadapté au calcul

Système de numération : Exemples

Numération Décimale :

- Le système de numération le plus utilisé actuellement.
- L'alphabet est composé de dix chiffres :

$$\mathcal{A} = \{0,1,2,3,4,5,6,7,8,9\}$$

- Le nombre 10 est la base de cette numération
- C'est un système positionnel. Chaque position possède un poids.
- Par exemple, le nombre 4134 s'écrit comme :
  $4134 = 4x10^3 + 1x10^2 + 3x10^1 + 4x10^0$

Systèmes de numération : Représentation d'un nombre dans une base quelconque

- Dans un système de numération en base b, un nombre entier noté $N_{(b)}$ égal à

$$N_{(b)} = \sum_{i=0}^{n-1} a_i b^i$$

S'écrit symboliquement :

$$N_{(b)} = a_{n-1} a_{n-2} ... a_2 a_1 a_0$$

Avec :
b : base ou nombre de chiffres qu'utilise le système de numération.
a_i : chiffre de rang i.
b_i : pondération associée à a_i.
Exemple : $1852_{(10)} = 1x10^3 + 8x10^2 + 5x10^1 + 2x10^0$

Systèmes de numération : Représentation d'un nombre dans une base quelconque

- Nombres fractionnaires
  Les nombres fractionnaires sont ceux qui comportent des chiffres après la virgule.

  Dans le système décimal, on écrit par exemple :
  $12,346 = 1x10^1 + 2x10^0 + 3x10^{-1} + 4x10^{-2} + 6x10^{-3}$

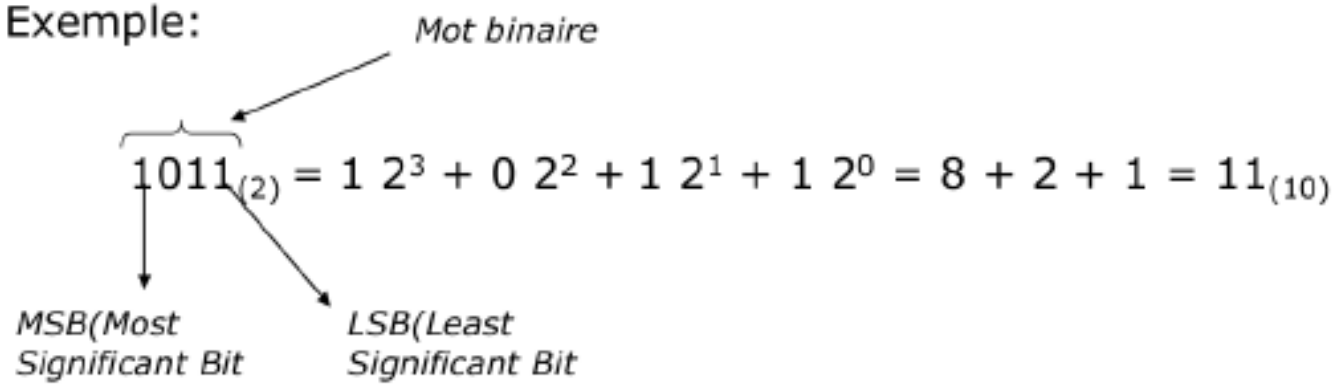
  Dans le cas général, en base b on écrit :
  $a_{n-1}...a_1a_0, a_{-1}a_{-2}...a_{-p} = a_{n-1}b^{n-1} + ... + a_0b^0 + a_{-1}b^{-1}... + a_{-p}b^{-p}$

Systèmes de numération : Représentation d'un nombre dans une base quelconque

- Bases utilisées en informatique :
  - En décimal, $b = 10$, $a_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  - En binaire, $b = 2$, $a_i \in \{0, 1\}$: 2 chiffres binaires, ou bits.
  - En hexadécimal, $b = 16$, $a_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$ (on utilise les 6 premières lettres comme des chiffres)
  - En octal : $b = 8$, $a_i \in \{0, 1, 2, 3, 4, 5, 6, 7\}$

Systèmes de numération : Représentation d'un nombre dans une base quelconque

- Système binaire :



Systèmes de numération : Représentation d'un nombre dans une base quelconque

- En binaire, avec n bits, on peut représenter 2^n informations. Exemple :
  - 1 bit, on peut représenter 2 informations possibles : 0 ou 1, soit 2 1 = 2.
  - 2 bit, on peut représenter 4 informations possibles : 00, 01, 10 ou 11, soit $2^2 = 4$.
  - 3 bit, on peut représenter $8 = 2^3$ informations possibles : 000, 001, 010...111.
  - 8 bit, on peut représenter $256 = 2^8$ informations possibles. Cet ensemble de 8 bits est appelé octet.

Systèmes de numération : Représentation d'un nombre dans une base quelconque

- Unités de mesure de l'information :
  - 1Ko (Kilo-octet) = 1024 octets = 2^{10} octets.
  - 1Mo (Méga-octet) = 1024 Ko = 2^{20} octets.
  - 1Go (Giga-octet) = 1024 Mo = 2^{30} octets.
  - 1To (Téra-octet) = 1024 Go = 2^{40} octets.

Changement de base (Transcodage)

Le transcodage (ou changement de base) est l'opération qui permet de passer de la représentation d'un nombre exprimé dans une base à la représentation du même nombre mais exprimé dans une autre base.

Changement de base : Passage d'une base b à la base 10

- L'opération est immédiate. Il suffit d'appliquer directement la relation suivante :

$$N_{(b)} = \Sigma_{i=0}^{b} a_i b^i$$

Exemples :

$AB_{(16)} = 10.16^1 + 11.16^0 = 160 + 11 = 171_{(10)}$

$10_{(2)} = 1.2^1 + 0.2^0 = 2 + 0 = 2_{(10)}$

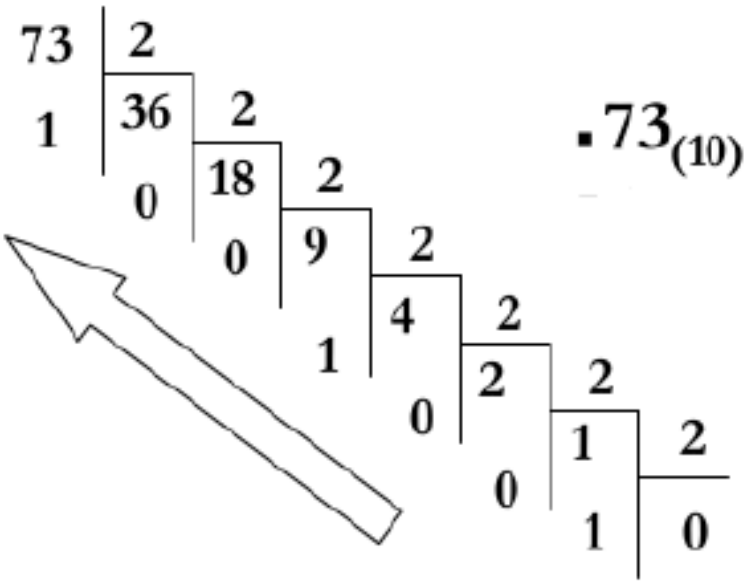
$10_{(8)} = 1.8^1 + 0.8^0 = 8 + 0 = 8_{(10)}$

Changement de base : Passage de la base 10 vers une base b

- **Nombres entiers :**
  - On procède par divisions successives.
  - On divise le nombre par la base, puis le quotient obtenu par la base, et ainsi de suite jusqu'à obtention d'un quotient nul.
  - Le nombre cherché s'écrit en plaçant les restes des divisions successives dans l'ordre inverse de leur obtention.

Changement de base : Passage de la base 10 vers une base b

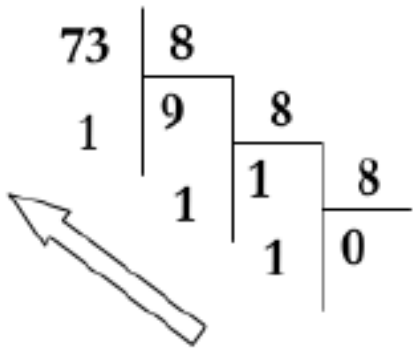
Décimale vers Binaire



$$\cdot 73_{(10)} = 1001001_{(2)}$$

Changement de base : Passage de la base 10 vers une base b

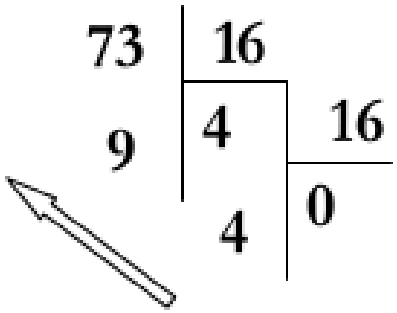
Décimale vers Octale



$$\cdot 73_{(10)} = 111_{(8)}$$

Changement de base : Passage de la base 10 vers une base b

Décimale vers Octale



$$\blacksquare 73_{(10)} = 49_{(16)}$$

Changement de base : Passage de la base 10 vers une base b

- **Nombres fractionnaires :**
  On multiplie la partie fractionnaire par la base en répétant l'opération sur la partie fractionnaire du produit jusqu'à ce qu'elle soit nulle (ou que la précision voulue soit atteinte). Pour la partie entière, on procède par divisions comme pour un entier.

  Exemple : conversion de $54,25_{(10)}$ ***en base 2***
  Partie entière : $54_{(10)} = 110110_{(2)}$ par divisions. Partie fractionnaire :

$$\begin{aligned} 0,25 \times 2 = 0,50 &\Rightarrow a_{-1} = 0 \\ 0,50 \times 2 = 1,00 &\Rightarrow a_{-2} = 1 \\ 0,00 \times 2 = 0,00 &\Rightarrow a_{-3} = 0 \end{aligned}$$

  Finalement : $54,25_{(10)} = 110110,010_{(2)}$

Changement de base : Hexadécimale $\leftrightarrows$ Binaire

Cas particulier : conversion d'un nombre hexadécimal en binaire et réciproquement

La conversion d'un nombre hexadécimal en binaire est simple car 16 combinaisons se représentent par un quartet. Une telle correspondance permet de représenter chaque digit hexadécimal par 4 bits (en complétant éventuellement par des zéros).

Exemple : $F17_{(16)} = \underbrace{1111}_{F}\ \underbrace{0001}_{1}\ \underbrace{0111}_{7}\ (2)$

La conversion inverse est immédiate. On divise le nombre binaire en « tranches de quatre» en partant de la droite. Chacun des « paquets » est ensuite converti en hexadécimal.

Tableau de correspondance des 16 premiers chiffres :

Hexadécimal	Décimal	Binaire
0	0	0 0 0 0
1	1	0 0 0 1
2	2	0 0 1 0
3	3	0 0 1 1
4	4	0 1 0 0
5	5	0 1 0 1
6	6	0 1 1 0
7	7	0 1 1 1
8	8	1 0 0 0
9	9	1 0 0 1
A	10	1 0 1 0
B	11	1 0 1 1
C	12	1 1 0 0
D	13	1 1 0 1
E	14	1 1 1 0
F	15	1 1 1 1

Changement de base : Octale $\leftrightarrows$ Binaire

Cas particulier : conversion d'un nombre octale en binaire et réciproquement

regroupement des bits en des sous ensembles de **trois bits** puis remplacer chaque groupe par le symbole correspondant dans la base 8

- Soit N un nombre représenté en base binaire par : $N = 1010011101_{(2)}$
- Représentation Octale ?

$$N = 001 \quad 010 \quad 011 \quad 101_{(2)}$$

$$= \quad 1 \quad 2 \quad 3 \quad 5_{(8)}$$

$$1010011101_{(2)} = 1235_{(8)}$$

Arithmétique binaire

Dans le cas général, en base b, les opérations arithmétiques s'effectuent avec les mêmes méthodes qu'en base 10. Une retenue ou un report apparaît lorsque l'on atteint ou dépasse la valeur b de la base.

Exemple : Table d'addition en binaire

a	b	s=a+b	r
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

En passant d'une colonne à la suivante, il peut arriver que l'on ait à additionner trois 1 :

$$1 + 1 + 1 = 11_{(2)}$$

Exercice

Effectuer les opérations suivantes en binaire :

$$\begin{array}{r} 101111 \\ + \\ 010011 \end{array}$$

$$\begin{array}{r} 11110 \\ \times \\ 1111 \end{array}$$

Codage des nombres : Entiers naturels

- Les entiers naturels (positifs ou nuls) sont codés sur un nombre d'octets fixé. On rencontre habituellement des codages sur 1, 2 ou 4 octets, etc.
- Un codage sur n bits permet de représenter tous les nombres naturels compris entre 0 et $2^n - 1$.
- Par exemple sur 1 octet, on pourra coder les nombres de 0 à $255 = 2^8 - 1$.

Codage des nombres : Entiers naturels

- On représente le nombre en base 2 et on range les bits dans les cellules binaires correspondant à leur poids binaire, de la droite vers la gauche.
- Si nécessaire, on complète à gauche par des zéros (bits de poids fort).
  - Exemple : Sur un octet, $10_{(10)}$ se code : $00001010_{(2)}$

Codage des nombres : Entiers relatifs

- Il faut ici coder le signe du nombre. Il existe principalement 3 notations :
  - Notations avec binaire signé (par signe et valeur absolue)
  - Méthode de complément à 1
  - Codage en complément à 2

Codage des entiers relatifs : Binaire signé

- Le bit le plus significatif est utilisé pour représenter le signe du nombre :
  - si le bit le plus fort = 1 alors **nombre négatif**
  - si le bit le plus fort = 0 alors **nombre positif**
- Les autres bits codent la valeur absolue du nombre
- Exemple : Sur 8 bits, codage des nombres -24 et -128 en (bs)
  - -24 est codé en binaire signé par : $10011000_{(bs)}$
  - -128 hors limite nécessite 9 bits au minimum

Codage des entiers relatifs : Binaire signé

- Inconvénient du binaire signé :
  - Deux représentations du zéro : + 0 et - 0
    Sur 4 bits : $+0 = 0000_{(bs)}$, $-0 = 1000_{(bs)}$

Exercice :
Binaire signé

- Coder 100 et -100 en binaire signé sur 8 bits
  $100_{(10)} = (01100100)_{(bs)}$
  $-100_{(10)} = (11100100)_{(bs)}$
- Décoder en décimal $(11000111)_{(bs)}$ et $(00001111)_{(bs)}$
  $(11000111)_{(bs)} = -71_{(10)}$
  $(00001111)_{(bs)} = 15_{(10)}$
- Calculer : 1-2 en binaire signé sur 8 bits

Codification des nombres : Entiers relatifs
Complément à 1

- **Complément à 1 :**
  Le complément à 1 d'un nombre binaire s'obtient en changeant chaque 0 par un 1 et chaque 1 par un 0.

1	0	1	1	0	1	**Nombre binaire initial**
0	1	0	0	1	0	**Complément à 1**

- les nombres positifs sont codés de la même manière qu'en binaire pure.

Codification des nombres : Entiers relatifs
Complément à 1

- Exemple : -24 en complément à 1 sur 8 bits
  - |-24| en binaire pur $\rightarrow 00011000_{(2)}$
  - puis on inverse les bits $\rightarrow 11100111_{(ca1)}$
- Limitation :
  - deux codages différents pour 0 (+0 et -0)
  - Sur 8 bits : $+0 = 00000000_{(ca1)}$ et $-0 = 11111111_{(ca1)}$

Exercice : Complément à 1

- Coder 100 et -100 en complément à 1 sur 8 bits
  $100_{(10)} = (01100100)_{(ca1)}$
  $-100_{(10)} = (10011011)_{(ca1)}$
- Décoder en décimal $(11000111)_{(ca1)}$ et $(00001111)_{(ca1)}$
  $(11000111)_{(ca1)} = -56_{(10)}$
  $(00001111)_{(ca1)} = 15_{(10)}$
- Calculer : 1-2 en complément à 1 sur 8 bits

Codification des nombres : Entiers relatifs
Complément à 2

- **Complément à 2 :**
  Le complément à 2 d'un nombre binaire s'obtient simplement en prenant le complément à 1 de ce nombre et en ajoutant 1 au bit de son rang de poids le plus faible.

	1	0	1	1	0	1	**Nombre binaire initial**
	0	1	0	0	1	0	**Complément à 1**
+						1	
	0	1	0	0	1	1	**Complément à 2**

- les nombres positifs sont codés de la même manière qu'en binaire pure.

Codage des nombres : Entiers relatifs
Complément à 2

- Exemple : -24 en complément à 1 sur 8 bits
  - 24 en binaire pur $\rightarrow 00011000_{(2)}$
  - -24 en complément à 1 $\rightarrow 11100111_{(ca1)}$
  - donc -24 en complément à 2 est codée par $11101000_{(ca2)}$

Codage des nombres : Entiers relatifs
Complément à 2

Un seul codage pour 0. Par exemple sur 8 bits :

- +0 est codé par $00000000_{(ca2)}$
- -0 est codé par $11111111_{(ca1)}$
- Donc -0 sera représenté par $00000000_{(ca2)}$

Codification des nombres : Entiers relatifs

- Le complément à 2 est le codage utilisé pour les nombres entiers relatifs.

Exercice : Complément à 2

- Coder 100 et -100 en complément à 2 sur 8 bits
  $+100_{(10)} = (01100100)_{(ca2)}$
  $-100_{(10)} = (10011100)_{(ca2)}$
- Décoder en décimal $(11001001)_{(ca2)}$ et $(01101101)_{(ca2)}$
  $(11001001)_{(ca2)} = -55_{(10)}$
  $(01101101)_{(ca2)} = 109_{(10)}$
- Calculer : 1-2 en complément à 2 sur 8 bits

Écriture des nombres binaires signés dans la notation en complément à 2

- **Entiers positifs ou nuls :**
  On représente le nombre en base 2 et on range les bits comme pour les entiers naturels. Cependant, la cellule de poids fort est toujours à 0 : on utilise donc n-1 bits. Le plus grand entier positif représentable sur n bits en relatif est donc $2^{n-1} - 1$.
- **Entiers négatifs :**
  Soit x un entier positif ou nul représenté en base 2 sur n-1 bits. Pour obtenir le codage d'un nombre x négatif, on code en binaire sa valeur absolue sur n-1 bits, puis on complémente à 2 tous les bits.

Remarques :

- le bit de poids fort d'un nombre négatif est toujours 1
- sur n bits, le plus grand entier positif est $2^{n-1} - 1$ (on suppose que le 0 est positif)
- sur n bits, le plus petit entier négatif est -2^{n-1} .
- $\Rightarrow$ Avec n bits on représente tous les entiers relatifs dans l'intervalle :$[-2^{n-1}, 2^{n-1} - 1]$

Exercice

- Quel intervalle de nombres décimales entiers est-il possible de représenter avec 4 bits ? Idem pour 8 bits.
- Quel intervalle de nombres décimaux relatifs signées représenter avec 4 bits ? Idem pour 8 bits.
- Réaliser la soustraction suivante : 11010010-00101010

Codage des caractères

- Les caractères, appelés *symboles alphanumériques*, incluent les lettres majuscules et minuscules, les symboles de ponctuation (& ~ , . ; # " - etc...), et les chiffres. Un texte, ou *chaîne de caractères*, sera représenté comme une suite de caractères. Le codage des caractères est fait par une table de correspondance indiquant la configuration binaire représentant chaque caractère. Les principaux codes de caractères utilisés par les ordinateurs sont:

  - **Code ASCII (American Standard Code for Information Interchange)**

    Le code ASCII représente chaque caractère sur 7 bits (on parle parfois de code **ASCII étendu**, utilisant 8 bits pour coder des caractères supplémentaires). Notons que le code ASCII original, défini pour les besoins de l'informatique en langue anglaise ne permet pas la représentation des caractère accentués (é, è, à, ù, ...), et encore moins des caractères arabes ou chinois. Pour ces langues, d'autres codages existent, utilisant 16 bits par caractères. Le tableau du code ASCII est présenté à la figure 1.

  - **Code Unicode**

    Le code Unicode attribue à chaque caractère ou symbole une valeur à 16 bits unique.

Codage des caractères

Fig 1. Jeu de caractères ASCII

Hex	Caractère	Hex	Caractère	Hex	Caractère	Hex	Caractère	Hex	Caractère	Hex	Caractère
20	Espace	30	0	40	@	50	P	60	'	70	p
21	!	31	1	41	A	51	Q	61	a	71	q
22	"	32	2	42	B	52	R	62	b	72	r
23	#	33	3	43	C	53	S	63	c	73	s
24	$	34	4	44	D	54	T	64	d	74	t
25	%	35	5	45	E	55	U	65	e	75	u
26	&	36	6	46	F	56	V	66	f	76	v
27	'	37	7	47	G	57	W	67	g	77	w
28	(	38	8	48	H	58	X	68	h	78	x
29	)	39	9	49	I	59	Y	69	i	79	y
2A	*	3A	:	4A	J	5A	Z	6A	j	7A	z
2B	+	3B	;	4B	K	5B	[	6B	k	7B	{
2C	,	3C	<	4C	L	5C	\	6C	l	7C	|
2D	-	3D	=	4D	M	5D	]	6D	m	7D	}
2E	.	3E	>	4E	N	5E	^	6E	n	7E	~
2F	/	3F	?	4F	O	5F	_	6F	o	7F	SUPPR

Codage des caractères

Hex	Nom	Signification	Hex	Nom	Signification
0	NUL	Null	10	DLE	Échappement transmission
1	SOH	Début d'en-tête	11	DC1	Commande auxiliaire 1
2	STX	Début du texte	12	DC2	Commande auxiliaire 2
3	ETX	Fin du texte	13	DC3	Commande auxiliaire 3
4	EOT	Fin de communication	14	DC4	Commande auxiliaire 4
5	ENQ	Demande	15	NAK	Acquittement négatif
6	ACK	Accusé de réception	16	SYN	Synchronisation
7	BEL	Sonnerie	17	ETB	Fin de bloc de transmission
8	BS	Retour arrière	18	CAN	Annulation
9	HT	Tabulation horizontale	19	EM	Fin de support
A	LF	Interligne	1A	SUB	Substitution
B	VT	Tabulation verticale	1B	ESC	Échappement
C	FF	Présentation de formule	1C	FS	Séparateur de fichier
D	CR	Retour début de ligne	1D	GS	Séparateur de groupe
E	SO	Hors code	1E	RS	Séparateur d'article
F	SI	En code	1F	US	Séparateur d'unité