République Algérienne Démocratique & Populaire

Ministère de L'Enseignement Supérieure et de la recherche Scientifique
Université Tahri Mohamed de Bechar
Faculté des sciences exactes
Département des Maths et Informatique

Architecture des Ordinateurs (AO)

Préparé par : *Bendjima Mostefa*
Maitres de conférences en informatique

Octobre 2020

1. Présentation générale

1.1. Introduction à la notion d'architecture des ordinateurs

C'est en 1962 que Philippe Dreyfus employé le mot informatique pour définir le traitement automatique de l'information. En fait ce mot peut correspondre à deux groupes de disciplines distinctes: l'ensemble des techniques mises en oeuvre pour l'emploi de l'ordinateur *(electronic data processing)*, une science nouvelle, qui ne nécessite pas obligatoirement l'utilisation des ordinateurs, ces derniers n'en sont qu'un outil majeur *(computer science)*.

On peut donc affirmer que l'informatique est une discipline carrefour, dont les ordinateurs actuels, les structures intellectuelles (algorithmes de calcul) et institutionnelles (organisation comptable, organisation industrielle) déterminent en majeure partie le contenu.

D'un point de vue plus général, il n'est plus à démontrer que l'informatique envahit progressivement, sous beaucoup de formes, notre vie quotidienne, tant est puissant son développement. L'informatique aujourd'hui, c'est à la fois les calculettes de poche et les ordinateurs portables, les consoles et jeux vidéo, mais aussi l'aventure spatiale, les robots industriels, les applications médicales telles que le scanner, les cartes à puce et bien d'autres applications. L'informatique à travers l'ordinateur modifie, et modifiera plus encore, l'organisation du travail et les rapports entre les individus.

Le mot architecture désigne comment est fait un ordinateur, quels sont les éléments le constituant, son mode de fonctionnement et surtout comment circule et est traité l'information par les différents constituants de l'ordinateur.

Il faut distinguer deux catégories d'architectures: matérielle et logicielle. L'architecture matérielle est relative aux différents composants, alors que l'architecture logicielle concerne surtout les fonctionnalités des systèmes d'exploitation, qui ne sont autres que des systèmes informatiques (logiciels ou programmes) permettant le fonctionnement de l'ordinateur suivant une stratégie ou architecture.

Pour comprendre le rôle et les fonctionnalités des systèmes d'exploitation, il est nécessaire d'appréhender ce qu'est une "machine", c'est-à-dire quels sont les constituants matériels d'un ordinateur, et quels en sont les principes de fonctionnement.

Un ordinateur (ou calculateur) est une machine "informatique" qui permet d'effectuer des calculs et des traitements ordonnés par un opérateur.

1.2. L'histoire de l'architecture des ordinateurs

C'est durant la période de 10000 avant J.C. que naquit le traitement rationnel de l'information. L'homme changea son mode de vie en passant du stade de chasseur à celui d'agriculteur. Dans ce nouveau mode de vie, l'homme doit disposer de semences, d'outils, d'animaux de ferme. C'est pourquoi il lui est nécessaire de mettre en place un système de troc et de ce fait, l'homme doit apprendre à s'organiser, compter, écrire : c'est l'apparition du traitement de l'information. On découvrit alors en Mésopotamie l'utilisation de boules, de jetons d'argile et de tablettes par les Sumériens qui servait dans ce système d'échange.

En arrivant au XIIIe siècle, époque de grandes effervescences intellectuelles, pour voir apparaître des systèmes de calcul plus rapides et plus automatiques.

- XIIIe: fabrication de l'Ars Magna, par Raymond Lulle : il s'agit d'une «machine logique» faite de cercles concentriques contenant des mots qui, disposés dans un certain ordre, forment des questions tandis que d'autres mots y répondent.
- XVIe: invention du codage binaire par Francis Bacon et du logarithme (à l'origine créé pour simplifier des calculs compliqués) par Napier.
- 1624: Wilhem Schickard construit une «horloge automatique calculante» à Heidelberg
- 1642: Blaise Pascal, à 19 ans, crée la «Pascaline», machine à calculer mécanique à base de roues dentées, capable de faire des additions et des soustractions, le langage PASCAL sera plus tard ainsi nommé en son honneur.
- 1673: Leibniz, grand mathématicien, améliore la Pascaline en y ajoutant la multiplication et la division ; par ailleurs, il s'intéresse beaucoup à la numérotation binaire avec laquelle il essaie de concevoir une «caractéristique universelle» dont l'objectif est de réduire toutes les opérations logiques à un calcul.
- XVIIIe: La Mettrie, philosophe disciple de Descartes, radicalise la philosophie de ce dernier et écrit *L'homme machine*, où il argumente en faveur d'une vision mécaniste du vivant (Descartes lui-même aurait construit un automate à visage humain). Les automates sont très à la mode à cette époque. L'horloger suisse Vaucansson en construit de très célèbres parmi lesquels un joueur de flûte et un canard pourvu de fonctions locomotrices et digestives, exposés à paris en 1738 : leur fonctionnement utilise un «arbre à came» (comme dans les boîtes à musique), codage binaire du mouvement. Un célèbre «joueur d'échec artificiel» parcours aussi les cours européennes à la fin du siècle (il aurait notamment battu Napoléon) avant qu'on ne démasque la supercherie : un nain caché sous la table actionnait en fait le mécanisme.
- 1805: Jacquart crée les métiers à tisser automatiques, qui utilisent des «programmes» sous forme de cartes perforées, également utilisées dans les pianos mécaniques.
- 1818: Mary Shelley publie «Frankenstein», où l'électricité donne l'étincelle de vie.
- 1822: l'ingénieur anglais Babbage fait les premiers plans de sa «machine à différences», sorte de machine à calculer mécanique utilisant les logarithmes : trop complexe pour la technologie de l'époque, elle ne sera construite d'après ces plans qu'au XXième siècle.
- 1832: invention du langage Morse.
- 1833: Babbage conçoit sa «analytical engine», encore plus performante (et compliquée) que la «machine à différence», utilisant des cartes perforées pour enchaîner l'exécution d'instructions élémentaires sur un calculateur universel (mécanique) : il passera sa vie et se ruinera à essayer en vain de construire sa machine. Il sera aidé par Lady Ada Lovelace, fille du poète Lord Byron, qui écrira les premiers «programmes» qu'aurait pu exécuter la machine (le langage de programmation ADA sera ainsi nommé pour lui rendre hommage). Cette machine aurait pourtant répondu aux besoins croissants en calcul dans la société anglaise, notamment pour l'astronomie et la navigation.

- 1854: Le logicien anglais Georges Boole publie son livre *The Mathematical Analysis of Logic*, où il définit les opérateurs logiques dits «booléens», fondés sur deux valeurs 0/1 pour coder Vrai/Faux.
- 1876: Bell invente le téléphone.
- 1884: L'ingénieur américain Hollerith dépose un brevet de machine à calculer automatique
- 1890: Hollerith commercialise des machines à calculer électriques, utilisées notamment pour traiter automatiquement les données d'un recensement aux Etats-Unis. Les besoins industriels en calcul automatique se multiplient.
- 1896: Hollerith crée une société appelée «Tabulation Machine Corporation», qui deviendra en 1924, «International Business Machine» (IBM), qui existe toujours.
- 1921: invention du mot «robot» par Karel Capek, auteur dramatique tchèque.
- 1925: Vannevar Bush, ingénieur américain, construit un calculateur analogique au MIT (Massachusetts Institute of Technology, prestigieuse école d'ingénieur américaine).
- 1927: la télévision et la radio deviennent opérationnels.
- 1931: l'allemant Konrad Zuse construit un calculateur automatique, le Z1.
- 1936: Alan Turing propose sa définition des «machines de Turing» et Church invente le «lambda-calcul», qui se révèlent avoir des capacités de calcul équivalentes.
- 1938: fondation de Hewlett Packard, société de matériels électroniques.
- 1939: John Atanassoff et Clifford Berry, son étudiant, conçoivent un prototype appelé ABC à l'université de l'Iowa, reconnu comme le premier ordinateur digital.
- 1939-1945: pendant la guerre,
    - Alan Turing travaille dans le service anglais de décryptage des messages secrets allemands (codés suivant le système appelé «Enigma») : il réalise une machine à décrypter qui contribuera à la victoire des alliés, en 1941, il construit le « Colossus » à l'université de Manchester (bientôt suivi du Mark I et du Mark II), premiers ordinateurs européens avec le Z3 de Konrad Zuse qui, pour la première fois, propose un contrôle automatique de ses opérations
    - John Von Neumann, travaille sur les calculs de balistique nécessaires au projet *Manhattan* (conception et réalisation de la première bombe atomique américaine).
- 1945: John Von Neumann écrit un rapport où il propose l'architecture interne d'un calculateur universel (ordinateur), appelée désormais «architecture de Von Neumann».
- 1946: construction de l'ENIAC à l'Université de Pennsylvanie, dernier gros calculateur électrique programmable (mais pas universel) : il fait 30 tonnes, occupe $160m^2$ et sa mémoire est constituée de 18 000 tubes à vide, sa puissance est équivalente à celle d'une petite calculette actuelle, pendant ce temps, Wallace Eckler et John Mauchly conçoivent le Binac (Binary Automatic Computer), qui opère pour la première fois «en temps réel» mais ne sera construit qu'en 1949, avec l'apport de Von Neumann.
- 1947: invention du transistor (qui peut être vu comme un interrupteur miniature).
- 1948: Claude Shannon publie sa *Théorie mathématique de l'information*, où est introduite la notion de quantité d'information d'un objet et sa mesure en bits, l'année suivante il construit la première machine à jouer aux échecs.

A partir de cette date, l'ordinateur existe et son histoire matérielle se réduit donc à l'évolution des progrès technologiques, qu'on découpe habituellement en termes de «générations». Les avancées conceptuelles les plus spectaculaires concernent, elles, principalement la conception de nouveaux langages de programmation évolués.

Première génération: les monstres

- 1949: construction de l'EDVAC, premier ordinateur construit suivant l'architecture de Von Neumann et stockant ses données sur disques magnétiques.
- 1950: Turing écrit un article dans une revue philosophique pour argumenter que le modèle des ordinateurs peut réaliser tout ce que fait l'esprit humain.
- 1952: IBM commercialise les premiers ordinateurs à lampes et à tubes à vide, IBM 650 puis IBM 701.
- 1954: premiers essais de programmation avec le langage FORTRAN (FORmula TRANslator), encore utilisé de nos jours pour le calcul scientifique.
- 1955: invention du mot «ordinateur» en France, à la demande d'IBM.
- 1956: le terme d'Intelligence Artificielle est inventé lors d'une conférence à Dartmouth, aux Etats-Unis.

Deuxième génération: intégration du transistor

- 1958: l'IBM 7044, 64 Koctets de mémoire, est le premier ordinateur intégrant des transistors; John McCarthy invente le LISP, premier langage de l'Intelligence Artificielle
- 1959: conception de COBOL (Common Business Oriented Language) : langage de programmation spécialisé pour la gestion et le domaine bancaire, encore utilisé de nos jours et du langage LISP (List Processing), adapté à des applications d'intelligence artificielle.
- 1960: conception de ALGOL (ALGOrithmic Language), langage évolué de calcul scientifique

Troisième génération: les circuits intégrés

- 1962: le terme «informatique» est créé en France par contraction de «information automatique»
- 1964: utilisation des circuits intégrés (circuits électroniques miniatures).
- 1965: le premier doctorat (thèse) en informatique est attribué à l'université de Pennsylvanie, conception du langage BASIC (Beginners' All-purposes Symbolic Instruction Code) et du langage PL/1 (Programming Language 1)
- 1969: premier essai de transfert de fichier à distance par le réseau Arpanet, ancêtre d'Internet, invention du langage PASCAL par Nicklaus Wirth
- 1971: introduction des disquettes pour l'IBM 370, conception du langage LOGO, destiné à l'initiation pédagogique aux concepts de la programmation.

Quatrième génération: les micro-ordinateurs

- 1972: conception du langage C, particulièrement adapté à la programmation et à l'utilisation de systèmes d'exploitation.
- 1973: apparition des premiers micro-ordinateurs munis d'un clavier et d'un écran, création de MICRAL, le premier micro-ordinateur français, et invention à Marseille du langage PROLOG (PROgrammation LOGique), par Alain Colmerauer.
- 1975: Bill Gates commercialise le langage BASIC et crée la société Microsoft avec Paul Allen, le premier magasin spécialisé en informatique ouvre en Californie.
- 1976: conception du langage Smalltalk, qui introduit la programmation «orientée objet».
- 1977: création de la société Apple par Steve Jobs et Steve Wozniak et commercialisation de l'Apple II, premier micro-ordinateur largement diffusé.

Cinquième génération: l'interface graphique et les réseaux

Les japonais avaient annoncé pour les années 90 l'apparition d'un nouveau type d'ordinateurs «cinquième génération» dédiés à des applications d'Intelligence Artificielle, mais ces machines d'un nouveau genre n'ont jamais vu le jour, et les évolutions majeures récentes sont plutôt à chercher du côté d'Internet.

- 1983: conception du langage ADA (en hommage à Lady Ada Lovelace), extension du langage PASCAL, pour répondre à une demande de la NASA
- 1984: Le McIntosh d'Apple introduit pour la première fois une interface graphique (menus, icônes...) et la souris, conception du langage C++, version orientée objet du langage C.
- 1992: création de Mosaïc au CERN de Genève, premier navigateur permettant de visualiser des pages Web (et donc ancêtre de Netscape).
- 1995: Windows 95 généralise l'interface graphique sur les PCs.
- …….jusqu'a ce jours.

💣 Donc, l'informatique n'est pas née d'hier, les premiers grands concepts datent de plus de 10000 ans. Cependant cette science ne s'est développée que depuis un demi siècle où l'architecture des ordinateurs a été le facteur clef de développement de l'informatique.

1.3. Définitions

1.3.1. Le mot "informatique" : il est composé des deux mots *information* et *automatique*. L'Académie française en a donné la définition suivante en 1965 : " Science du traitement rationnel, notamment à l'aide de machines automatiques, de l'information, considérée comme le support de connaissances dans les domaines scientifique, économique et social. ".

🕮 En général, l'informatique permet la manipulation, la gestion, l'organisation et le stockage de l'information.

On peut citer parmi les avantages de l'informatique :

- Calculer avec une grande précision ;
- Gain en terme de temps ;
- Aider à prendre des décisions.

1.3.2. Le mot "ordinateur": l'ordinateur signifiait au départ 'calculateur numérique électronique', mais, aujourd'hui, pourrait être définit comme *'une machine de traitement de l'nformation'*.

Un ordinateur (computer) est capable de :

- Acquérir des informations ;
- Conserver des informations ;
- Effectuer des traitements sur des informations ;
- Restituer des informations.

1.3.3. Le mot "information": généralement, on attribue le mot information le sens de données. L'information peut être numérique, texte, image, son, dessin, vidéo, etc…

📖 Pour pouvoir réaliser ces différentes opérations, un ordinateur doit posséder divers organes, tels qu'un clavier pour la saisie manuelle d'informations, une mémoire, une unité centrale de traitement et une imprimante ou un écran pour sortir les résultats, etc…

1.4. Les aspects de l'informatique

L'informatique est représentée par deux aspects distincts mais indissociables :

1.4.1. Le HARDWARE : qui est composé de l'ensemble de matériels (l'ordinateur et ses périphériques).

1.4.2. Le SOFTWARE : pour animer un ordinateur, il faut avoir des logiciels (programmes) qui trouvent place dans deux catégories :

- **Logiciel système** : qui est un ensemble de programmes chargés d'exploiter les ressources matérielles et de définir la façon dont ces derniers doit se comporter à l'égard des commandes introduites par un utilisateur. Exemples de logiciel système : le MS-DOS et Windows etc...
- **Logiciel d'application :** qui est un ensemble de programmes destinés à réaliser des taches biens définies, tels que le traitement de texte Microsoft Word et le tableur Microsoft Excel etc…

1.5. La programmation

La programmation consiste à partir d'un problème donné, à réaliser un programme dont l'exécution apporte une solution satisfaisante au problème posé. Elle consiste à écrire une suite d'instructions dans un langage compréhensible par un ordinateur.

L'activité de programmation, ou plus généralement de développement de projets, se décompose en plusieurs phases qui constituent le cycle de vie du logiciel :

- Compréhension du problème ;
- Spécification des fonctionnalités du système, on dit ce qu'on veut faire mais pas comment on veut le faire ;
- Conception des algorithmes pour résoudre le problème ;
- Programmation c'est à dire faire des programmes ;
- Tests et validation des programmes ;
- Maintenance des programmes.

1.5.1. Un algorithme : est une succession d'actions (opérations) destinées à résoudre un problème en un nombre fini d'instructions.

1.5.2. Un programme : est une suite d'instructions, écrites dans un langage donné, définissant un traitement exécutable sur un ordinateur.

1.5.3. Langage de programmation : les premiers programmes étaient écrits en langage machine (code binaire pur), puis en langage assembleur qui avait l'avantage d'utiliser des mnémoniques et des symboles. Les mnémoniques permettent de remplacer des séquences de 0 et de 1 par des caractères alphabétiques ou des noms plus faciles à mémoriser. Ensuite sont apparus des langages évolués tels que Pascal, Fortran, Prolog, Delphi etc…

Un programme écrit en langage évolué se trouve sous forme de code source. Pour pouvoir être exécuté par un ordinateur, ce code source doit subir les étapes suivantes : compilation, édition de liens et chargement dans la mémoire centrale.

- Le compilateur : est un programme qui transforme un module en code source en un module en code objet (code machine) ;
- L'éditeur de liens : est un programme s'occupe de mettre ensemble les différents modules d'un programme tels que les sous programmes, procédures, fonctions, etc…
- Le chargeur : est un programme s'occupe d'amener en mémoire centrale, depuis une mémoire auxiliaire, un programme complet et prêt à être exécuté.

1.6. Evolution réseau informatique

L'association de l'informatique et des télécommunications a conduit à l'apparition des réseaux. Tout système téléinformatique repose sur l'utilisation de un ou plusieurs réseaux.

Les principes étapes de l'évolution des réseaux informatique sont les suivantes.

- Les premiers essais de transmission de données entre deux ordinateurs ont lieu dans les années 60.
- Les années 70 sont marquées par l'accès à distance, la décentralisation des stations d'entrée/sortie (terminaux).
- A la fin des années 70 apparaissent des réseaux de terminaux (communication de circuits avec des concentrateurs, des multiplexeurs) ainsi que des réseaux d'ordinateurs (commutation de messages, paquets).
- La téléinformatique est l'un des sujets prédominants dans les années 80. Si la technologie semble à peu près au point pour permettre le développement de la téléinformatique, un problème important subsiste, celui de la normalisation.

1.6.1. Communication

Depuis les origines, l'homme a eu besoin de communiquer. Pour cela il mit au point des codes, des alphabets et des langages.

Parole, gestes de la main, signaux de fumée, document écrit etc…tout était bon pour véhiculer le message.

1.6.2. Informatique

C'est la science du traitement rationnel de l'information, notamment par machines automatiques, considérée comme support des connaissances et des communications, dans les domaines technique, économique et social.

1.6.3. Téléinformatique

C'est une discipline qui associe les techniques de l'informatique et des télécommunications dans le but de permettre à ses utilisateurs d'exploiter à distance les capacités de traitement de l'ordinateur.

1.6.4. Réseau

En toute généralité, on peut définir un réseau comme un ensemble de nœuds reliés par un ensemble de chemins. Un réseau peut être représenté par un graphe. Nous avons l'habitude d'utiliser un certain nombre de réseaux tels que les routes, le chemin de fer, les lignes aériennes, la poste, le téléphone, l'électricité, l'eau, la radio, la télévision, etc…

1.6.5. Réseau informatique

C'est un ensemble des unités de traitement de l'information, géographiquement dispersés, reliés entre eux par un ou plusieurs liens afin de permettre les échanges d'informations.

On parle aussi de réseaux :

- Homogènes: Tous les ordinateurs sont du même constructeurs: Apple - Talk.
- Hétérogènes: Les ordinateurs reliés au réseau sont de constructeurs divers: Ethernet.

💣 Généralement, quand on parle de réseau, on sous-entend un réseau informatique.

1.7. L'intérêt de l'architecture des ordinateurs

- De partager les fichiers.
- Le transfert de fichier.
- Le partage d'application : compilateur, système de gestion de base de donnée (SGBD).
- Partage d'imprimante.
- L'interaction avec les utilisateurs connectés: messagerie électronique, conférence électronique, Talk, ...
- Le transfert de donnée en générale.
- Le transfert de la parole, de la vidéo et des données.

1.8. Notion de base

- **Station de travail :** on appelle station de travail toute machine capable d'envoyer des données vers les réseaux (PC, MAC, SUN Terminale X, ...). Chaque station de travail à son propre carte interface (carte réseau).
- **Nœud :** c'est une station de travail, une imprimante, un serveur ou toute entité pouvant être adressée par un numéro unique.

  L'unicité de l'adresse est garantie par le constructeur d'une carte réseau qui donne un numéro unique ne pouvant être changé par une personne.
- **Terminal :** est un poste de travail composé d'un écran de visualisation et d'un clavier.
- **Serveur :** dépositaire centrale d'une fonction spécifique : serveur de base de donnée, de calcul, de fichier, etc ...

  Par exemple : le réseau Internet contient plusieurs serveurs :

  - Serveur WWW
  - Serveur FTP
  - Serveur telnet
  - Serveur messagerie

- **Paquet :** C'est la plus petite unité d'information pouvant être envoyée sur le réseau. Un paquet contient en général l'adresse de l'émetteur, l'adresse du récepteur et les données à transmettre.
- **Topologie :** organisation physique et logique d'un réseau. L'organisation physique concerne la façon dont les machines sont connectées (Bus, Anneau, Étoile, Maillé, Arborescence, etc...).
- **Protocole :** un protocole est l'ensemble des règles qui doivent être respectées pour réaliser un échange d'informations entre ordinateurs.
- **Multiplexage :** partage d'une support de transmission entre plusieurs liaisons.
- **Routage :** Le routage est une fonction qui consiste à acheminer les paquets d'une machine source à une machine destinataire.
- **Liaison :** une liaison est l'établissement d'une communication entre deux équipements informatique.

1.9. Informations à échanger

1.9.1. Information

Nous utilisons librement le terme information, en lui attribuant le sens de données. Les informations traitées aujourd'hui par un ordinateur ne sont pas nécessairement numériques. On peut utiliser l'ordinateur pour traiter des textes, des dessins, des images, etc….Mais, pour transmettre ces informations, il faut les coder par des 0 et des 1.

1.9.2. Codage de l'information

Les informations sont exprimées sous la forme de texte, caractères alphanumériques, caractères spéciaux, etc…

Pour les introduire dans la machine, il faut d'abord les coder sous forme binaire. Plusieurs codes ont été conçus pour réaliser cette opération de codage se basant tous sur une table de correspondance qui permet de trouver le code associé à un caractère donnée et inversement, le caractère correspondant à un code donnée.

Selon le nombre de bits associé au code, on distingue :

- le code BCD (code à 6 bits)
- le code ASCII (code à 7 bits)
- le code EBCDIC (code à 8 bits)

Le tableau suivant illustre quelques exemples de codes:

Caractère	Code BCD	Code ASCII	Code EBCDIC
0	000 000	011 0000	1111 0000
1	000 001	011 0001	1111 0001
2	000 010	011 0010	1111 0010
…	…….	……...	…….
A	010 001	100 0001	1100 0001
B	010 010	100 0010	1100 0010
…	……	……..	…….

1.10. Le réseau téléphonique pour la transmission de donnée

1.10.1. Utilisation du réseau commuté

Le réseau commuté peut transmettre des données mais à un débit de 4500 bit/s. Le réseau est accédé depuis l'ETTD via un modem normalisé pour respecter les caractéristiques du réseau.

Intérêt : Atteindre des correspondants situés n'importe où. Le coût de communication est environ égal au coût de la communication téléphonique.

1.10.2 Le RNIS

Le RNIS (Réseau Numérique à Intégration de Service) se caractérise par la distribution jusque chez l'abonner des cannaux déjà présent dans le réseau actuel.

L'usager a accès non seulement au réseau téléphonique, mais à un réseau de transmission apte à véhiculer le son, l'image et les données.

- Pour accéder aux réseaux numériques, il faut une interface standard de type X21. Pour ne pas jeter les micros munis uniquement de V24, on passe par un convertisseur X21 bis.

1.10.3. Ligne téléphonique classique (RTC)

On utilise le réseau téléphonique pour faire circuler l'information de l'entreprise. Un modem à chaque extrémité suffit pour accéder à ce réseau. C'est le moyen le plus souvent utilisé pour accéder au réseau Internet

Tarification au temps de connexion + abonnement mensuel

1.10.4. Lignes spécialisées

Vous pouvez choisir de réserver à votre seul usage un "câble" Algérie Télécom. C'est une solution séduisante mais très onéreuse pour les grandes distances. Tarification à la longueur de ligne + débit désiré.

2. La machine de Von Neumann

L'architecture d'un ordinateur est la description de ses organes fonctionnels et de leurs interconnections.

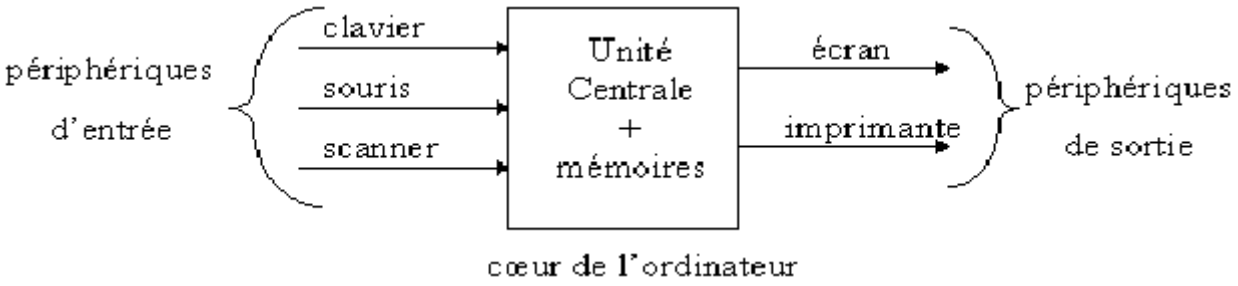


Schéma I.1. Organisation générale d'un ordinateur d'après de Von Neumann

Il convient d'abord de distinguer l'ordinateur lui-même de ses «périphériques», qui ne sont que des constituants annexes. Le coeur d'un ordinateur est constitué de trois unités :

- De l'Unité Centrale (UC), ou «microprocesseur», appelé familièrement «puce» ;
- De mémoires, parmi lesquelles on distingue plusieurs types :
    - la mémoire ROM (Read Only Memory : mémoire à accès en lecture seule) : ensemble de bits dont l'état est fixé une fois pour toute, lors de la construction de l'ordinateur. Elle sert à stocker des informations permanentes (procédures de démarrage...) ;
    - la mémoire RAM ou «mémoire vive» (Random Access Memory : mémoire à accès aléatoire) : ensemble de bits modifiables à volonté, où se trouvent stockées les données sur lesquelles travaille l'ordinateur. Il ne faut pas comprendre aléatoire dans le sens de «au hasard», mais par opposition à séquentiel ; cela signifie que l'on peut avoir accès directement à tout endroit de cette mémoire, sans avoir à la parcourir bit à bit. Cette mémoire est volatile, c'est-à-dire qu'elle ne conserve les données que tant que la machine est sous tension.
    - les mémoires secondaires ou auxiliaires : ce sont des dispositifs permettant de stocker des bits de façon stable (qui reste fixée même si on éteint la machine) tout en étant généralement modifiable. On peut inclure parmi elles les disques durs, les disquettes, les bandes magnétiques.
- De l'unité d'E/S, ou unités d'échange sont des éléments qui permettent de transférer des informations entre l'unité central et les périphériques d'E/S.

Les autres composants sont donc:

- soit des *périphériques d'entrée*, c'est-à-dire permettant à un utilisateur extérieur de *fournir des informations* (données/programmes) à la machine sous forme numérique : souris, clavier, scanner... Ces dispositifs peuvent tous être conçus comme des *numériseurs* puisqu'ils transforment un comportement (l'appui sur la touche d'un clavier, le mouvement de la souris) ou un objet (une photo analogique) en une suite de bits.
- soit des *périphériques de sortie*, c'est-à-dire permettant de visualiser ou de transmettre des données internes à l'extérieur : écran, imprimante... A l'inverse des numériseurs, ces dispositifs traduisent des suites de bits en information interprétable par les humains.

3. La machine Harvard

Le nom de cette structure vient du nom de l'université Harvard où une telle architecture a été mise en pratique pour la première fois avec le Mark I en 1944.

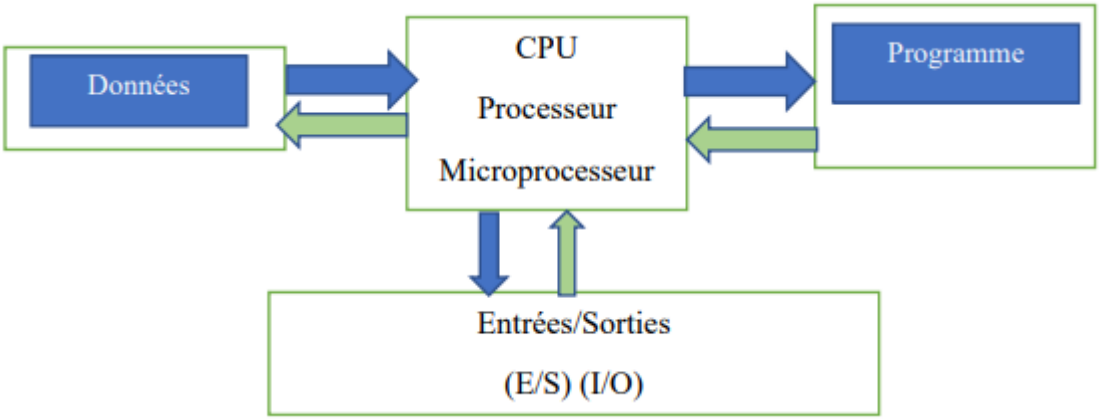


Schéma I.2. : schéma d'une machine Harvard

- Les deux mémoires peuvent avoir des caractéristiques différentes:
  - Taille du mot;
  - Timing;
  - Technologie;
  - Structure de l'adresse;
- Mémoires séparées l'une pour le programme et l'autre pour les données;
- Accés simultanés aux données et aux instructions.

4. Comparaison

	Architecture de Von Neumann	**Architecture de Harvard**
1	Mémoire partagée par les données et les instructions	Mémoires séparées pour les données et les instructions
2	Deux cycles d'horloges (un cycle pour la recherche de l'instruction et un cycle pour la recherche des données	Un seul cycle d'horloge
3	Le pipeline n'est pas possible	Le pipeline possible
4	Conception simple	Conception complexe (C.U)
5	Hardware minime	Plus de hardware
6	Moins d'espace	Plus d'espace
7	Exécution moins rapide	Exécution rapide
8	Meilleure exploitation de la mémoire	L'espace libre dans une mémoire ne peut être utilisé par l'autre mémoire

1. Schéma global d'une architecture

La configuration d'un ordinateur correspond à l'organisation adoptée pour mettre ensemble et faire fonctionner les divers éléments matériels de l'ordinateur. Les configurations possibles sont fonction de l'importance et de la finalité du système mis en œuvre.

Un ordinateur se décompose d'un ensemble d'éléments pouvant s'associer dans un boîtier, il est généralement composé:

- D'une unité centrale (processeur, carte mère, mémoire…);
- De périphériques internes (cartes de son, carte vidéo, ...);
- D'un lecteur de disquettes, d'un lecteur de CD-ROM ou de DVD-ROM;
- Eventuellement, de cartes d'extension diverses ;
- Périphérique d'entrée ;
- Périphérique de sortie.

1.1. La carte mère : la carte mère est le principal constituant de l'ordinateur. C'est sur cette carte que sont connectés les autres éléments :

- Le microprocesseur (cerveau de l'ordinateur);
- La mémoire (RAM : *Random Access Memory*, la mémoire cache);
- Le disque dur, le lecteur de CD-ROM, le lecteur de disquettes;
- Les périphériques internes : carte de son, carte vidéo.

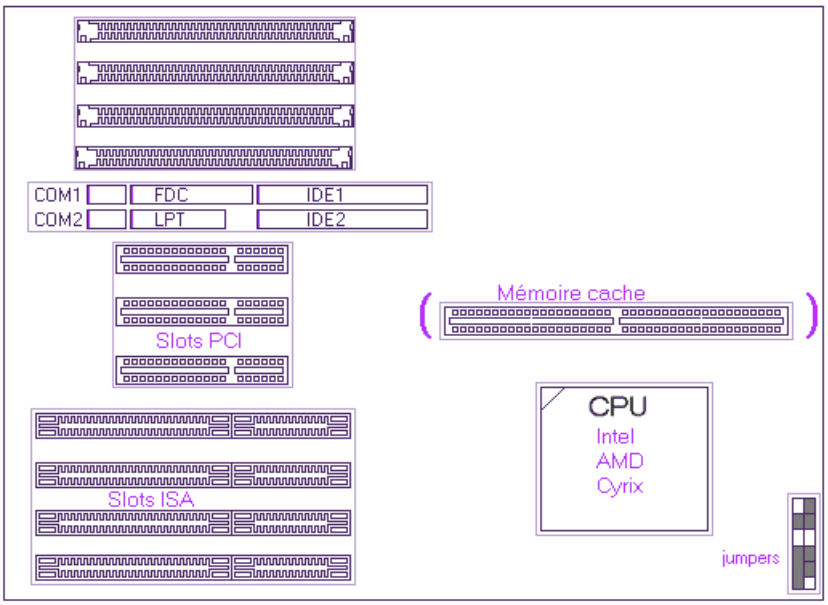


Schéma II.1. : carte mère

- **Le microprocesseur** : le processeur (CPU) est le cerveau de l'ordinateur, c'est lui qui coordonne le reste des éléments, il se charge des calculs, bref, il exécute les instructions qui ont été programmées. Toutes ces opérations sont des informations numériques. Les microprocesseurs utilisent des petits transistors pour faire des opérations de base, il y en a plusieurs millions sur un seul processeur.

  Les principaux éléments d'un microprocesseur sont :
  - Une horloge qui rythme le processeur. À chaque TOP d'horloge, le processeur effectue une instruction. Ainsi plus l'horloge a une fréquence élevée, plus le processeur effectue d'instructions par seconde (MIPS : Millions d'instruction par seconde). Par exemple un ordinateur ayant une fréquence de 100 mégahertz (MHz) effectue 100 000 000 d'instructions par seconde;
  - Une unité de gestion des bus qui gère les flux d'informations entrant et sortant;
  - Une unité d'instruction qui lit les données, les décode puis les envoie à l'unité d'exécution;
  - Une unité d'exécution accomplit les tâches données par l'unité d'instruction.

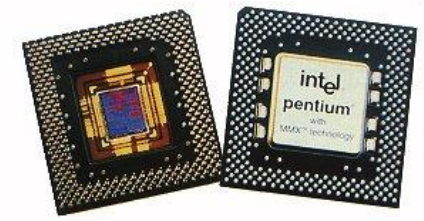


Schéma II.2: processeur

Le processeur travaille, en fait, grâce à un nombre très limité de fonctions comme des expressions logiques (ET, OU, NON, etc.), des expressions mathématiques (addition, soustraction, multiplication, etc.). Celles-ci sont directement câblées sur les circuits électroniques. Il est impossible de mettre toutes les instructions sur un processeur car celui-ci est limité par la taille de la gravure. Ainsi pour mettre plus d'instructions il faudrait un processeur ayant une très grande surface. Or le processeur est constitué de silicium et celui-ci coûte cher, et d'autre part il chauffe beaucoup. Le processeur traite donc les informations compliquées à l'aide d'instructions simples.

- **La mémoire cache** : la mémoire cache permet au processeur de se «rappeler» les opérations déjà effectuées auparavant. Elle est utilisée par le microprocesseur pour conserver temporairement des instructions élémentaires. En effet, elle stocke les opérations effectuées par le processeur, afin que celui-ci ne perde pas de temps à recalculer des calculs déjà faits précédemment. La taille de la mémoire cache est généralement de l'ordre de 512Ko ou 1Go.

- **La mémoire vive** : la mémoire vive, généralement appelée RAM (*Random Access Memory*, traduisez *mémoire à accès aléatoire*) ce qui signifie que l'on peut accéder instantanément à n'importe quelle partie de la mémoire, permet de stocker des informations pendant tout le temps de fonctionnement de l'ordinateur. Par contre, cette mémoire est détruite lors de la mise hors-tension de l'ordinateur, contrairement à une mémoire de masse comme le disque dur qui garde les informations même lorsqu'il est hors tension. La mémoire vive contient les données et les instructions des applications en cours.

- **La mémoire morte (ROM)** : mémoire permanente contenant des microprogrammes enregistrés sur des puces électroniques de la carte mère (ou *mother board*) contenant les routines de démarrage du micro-ordinateur. ROM (*Read Only Memory*, dont la traduction est *mémoire en lecture seule*) est appelée aussi parfois *mémoire non volatile*, car elle ne s'efface pas lors de la mise hors tension du système. En effet, ces informations ne peuvent être stockées sur le disque dur étant donné que les paramètres du disque (essentiels à son initialisation) font partie de ces données vitales à l'amorçage.

- **Les slots d'extension** : les slots (ou Les fentes) d'extension sont des réceptacles dans lesquels on peut enficher des cartes. Il en existe de trois types : les cartes ISA (les plus lentes fonctionnant en 16 bits), les cartes PCI (beaucoup plus rapides fonctionnant en 32 bits), et les cartes AGP (les plus rapides). Ils se branchent, grâce à des nappes, sur les broches prévues à cet effet sur la carte mère.

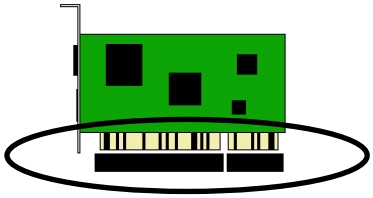


Schéma II. 3 : slot d'extension

📖 Les disques durs, CD-ROM et lecteurs de disquettes se branchent, grâce à des nappes, sur les broches prévues à cet effet sur la carte-mère. Il y en a en général au moins six:

- Les ports de communication (souris) se branchent sur les emplacements notés COM1, COM2 (parfois COM3 ...) ;
- Le port imprimante se branche sur l'emplacement noté LPT ;
- Le lecteur de disquette se branche sur l'emplacement noté FDC ("Floppy Disk controller" traduisez "Contrôleur de disquette") ;
- Les disques durs IDE, CD-ROM IDE se branchent par l'intermédiaire d'une nappe sur les emplacements notés IDE1 et IDE2.

- **Le disque dur** : le disque dur se présente sous la forme d'un boîtier hermétique à l'intérieur duquel se trouve une pile de plateaux ou disques magnétiques superposés et regroupés autour d'un même axe. Chaque disque possède deux faces : une face supérieure et une face inférieure. A chaque face est associée une tête de L/E fixée sur un bras mobile (deux têtes pour chaque disque).

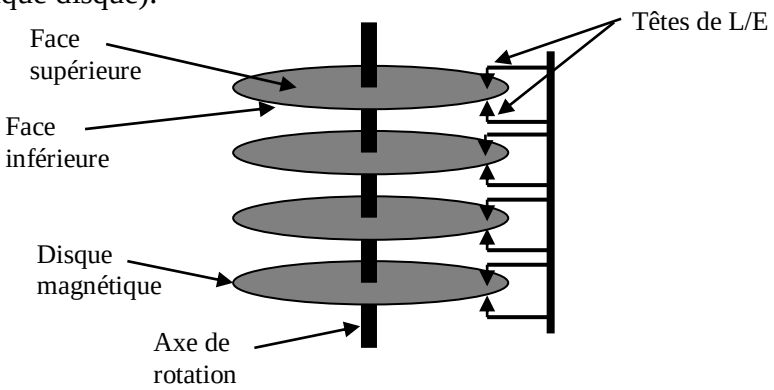


Schéma II.4. : disque dur de 4 plateaux et 8 têtes

- **La disquette** : constituée d'une rondelle de plastique souple recouverte d'un carré de plastique dur pour la protéger, la disquette à pour avantage d'être amovible, c'est-à-dire que l'on peut l'insérer et l'enlever du lecteur très facilement. Sa petite taille nous permet de la glisser dans une poche. Elle existe principalement en deux formats : 720 Ko et 1,44 Mo.

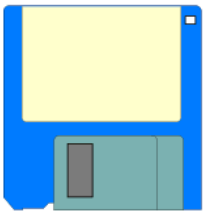


Schéma II.5 : disquette

- **Le CD-ROM** : il utilise des disques portatifs de grande capacité au format pratique, de plus en plus utilisé pour la vente de logiciels. Le CD-ROM (Compact Disc - Read Only Memory) est un disque optique de 12 cm de diamètre et de 1mm d'épaisseur, permettant de stocker des informations numériques, c'est-à-dire correspondant à 650 Mo de données informatiques ou bien jusqu'à 78 min de données audio.
  Le CD est constitué de matière plastique, recouvert d'une fine pellicule métallique d'aluminium sur une des faces. Les pistes sont gravées en spirales, ce sont en fait des alvéoles d'une profondeur de 125nm et espacées de 1,6µ. Ces alvéoles forment un code binaire, une alvéole correspond à un 0, un espace à un 1.

- **Le DVD-ROM** : le DVD-ROM (Digital Versatile Disc - Read Only Memory) est une variante du CD-ROM dont la capacité est largement plus grande. En effet, les alvéoles du DVD sont beaucoup plus petite (0,4µ et un espacement de 0.74µ), impliquant un laser avec une longueur d'onde beaucoup plus faible.
  Il existe 4 types de DVD différents :

Type de support	Capacité	Temps musical équivalent	Nombre de CD équivalent
CD	650Mo	1h18 min	1
DVD simple face simple couche	4.7Go	9 h 30	7
DVD simple face double couche	8.5Go	17 h 30	13
DVD double face simple couche	9.4Go	19 h	14
DVD double face double couche	17Go	35 h	26

- **Le modem** : le morse a été le premier codage à permettre une communication longue distance. C'est *Samuel F.B.Morse* qui l'a mis au point en 1844. Ce code est composé de points et de tirets (un langage binaire en quelque sorte). L'interpréteur était l'homme à l'époque, il fallait toutefois une bonne connaissance du code.
  De nombreux codes furent inventés dont le code d'Émile Baudot (portant d'ailleurs le nom de code *Baudot*, les anglais l'appelaient *Murray Code*).
  Le 10 mars 1876, le Dr Graham Bell met au point le téléphone, une invention révolutionnaire qui permet de faire circuler de l'information vocale dans des lignes métalliques.
  Ces lignes permirent l'essor des télescripteurs, des machines permettant de coder et décoder des caractères grâce au code Baudot (Les caractères étaient alors codés sur 5 bits, il y avait donc 32 caractères uniquement...).
  Dans les années 60, le code ASCII (American Standard Code for Information Interchange) est adopté comme standard. Il permet le codage de caractères sur 8 bits, soit 256 caractères possibles.
  Grâce aux techniques de numérisation et de modulation autour de 1962 ainsi que de l'essor des ordinateurs et des communications, le transfert de données via modem vit le jour.
  Le modem est le périphérique utilisé pour transférer des informations entre plusieurs ordinateurs (2 à la base) via les lignes téléphoniques. Les ordinateurs fonctionnent de façon digitale, ils utilisent le langage binaire, mais les modems sont analogiques. Les signaux digitaux passent d'une valeur à une autre, il n'y a pas de milieu, de moitié, c'est du Tout Ou Rien (un ou zéro).
  Le modem convertit en analogique l'information binaire provenant de l'ordinateur. Il envoie ensuite ce nouveau code dans la ligne téléphonique. On peut entendre des bruits bizarres si on le volume du son provenant du modem.
  Ainsi, le modem module les informations numériques en ondes analogiques, en sens inverse il démodule les données numériques.
  C'est pourquoi modem est l'acronyme de MOdulateur/DEModulateur.

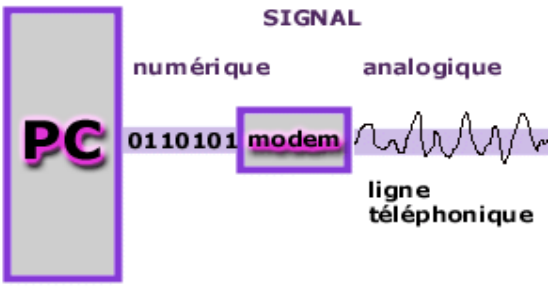


Schéma II.6: modulateur/demodulateur

- **La carte réseau** : la carte réseau est utilisée à d'interface physique entre l'ordinateur et le câble. Elle traite les données émises par l'ordinateur, elle les transfère et contrôle le flux de données entre l'ordinateur et le câble. Elle traduit aussi les données venant du câble en octets de façon que l'Unité Centrale de l'ordinateur puisse les comprenne. Enfin, la carte réseau s'insère dans un connecteur d'extensions (slot).

1.2. Périphériques d'entrée

- **Le clavier** : le clavier est le plus important périphérique d'entrée de données. Grâce à lui, il est possible de transférer des textes ou encore de donner ordre à la machine d'effectuer des opérations particulières. De la même façon que sur une machine à écrire, le clavier permet de saisir des caractères (lettres, chiffres, symboles, ...).

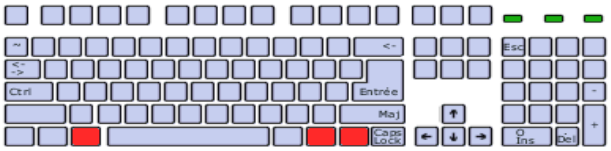


Schéma II.7: clavier

- **La souris** : le déplacement de la souris permet de déplacer un curseur sur l'écran avec lequel (en cliquant sur les boutons) on peut sélectionner, déplacer, manipuler des objets à l'écran.

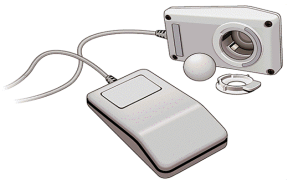


Schéma II.8 : souris

- **Le numériseur (scanner)** : Périphérique d'entrée qui permet, par balayage optique, la restitution d'une image à l'écran de l'ordinateur. Le numériseur est semblable, dans sa forme, au photocopieur, avec toutefois une importante distinction; l'image récupérée par l'appareil est transmise à l'ordinateur au lieu d'être immédiatement imprimée sur du papier.

Schéma II.9: numériseur

- **La caméra numérique** : Les caméras numériques sont des appareils photographiques qui ne contiennent pas de film. Les photos sont enregistrées sur une petite disquette au lieu de s'imprégner sur une pellicule. La photographie obtenue pourra être visionnée à partir de l'écran d'un ordinateur, ou encore d'un téléviseur. Le grand avantage de ces nouveaux appareils est leur capacité à transmettre une photo à un ordinateur, par l'intermédiaire d'un fil, pour ensuite l'intégrée à un document.

Schéma II.10 : caméra numérique

1.3. Périphériques de sortie

- **L'écran ou le moniteur** : Nous venons de voir une série de périphériques d'entrée, voyons maintenant les périphériques de sortie. Nous en retrouvons principalement deux, l'écran et l'imprimante
  L'écran, aussi appelé moniteur, affiche une image dont la netteté dépend de la résolution. Si l'image est composée de petits points, elle sera plus claire. Si les points sont plus gros, elle sera par le fait même beaucoup moins claire. Chacun de ses points s'appelle un pixel.
  Les moniteurs (écrans d'ordinateur) sont la plupart du temps des tubes cathodiques, c'est-à-dire un tube en verre dans lequel un canon à électrons émet des électrons dirigés par un champ magnétique vers un écran sur lequel il y a de petits éléments phosphorescents (luminophores) constituant des points (pixels) émettant de la lumière lorsque les électrons viennent les heurter.

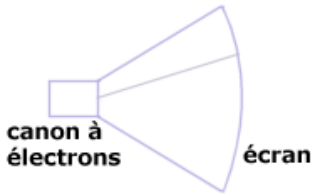


Schéma II.11 : moniteur

- **L'imprimante :** Visionner son travail à l'écran est utile mais le résultat final doit souvent se retrouver sur du papier. Il faut alors l'imprimer.
  L'imprimante permet de faire une sortie imprimée (sur papier) des données de l'ordinateur.
  Il en existe plusieurs types d'imprimantes, dont les plus courantes sont :
    - l'imprimante laser;
    - l'imprimante à jet d'encre;
    - l'imprimante à bulles d'encre;
    - l'imprimante matricielle (à aiguilles);
    - l'imprimante à marguerite.

2. Principe de fonctionnement

Un ordinateur se compose d'une mémoire central (mémoire principal), qui contient programmes et données, d'une unité central de traitement (processeur ou CPU), qui exécute un programme chargé en mémoire centrale, et d'unité d'entrée/sortie permettant l'échange d'informations avec des unités périphériques.

L'exécution d'un programme se déroule selon le modèle suivant :

- le programme et les données sont chargés en mémoire centrale.
- Les instructions du programme sont amenées une par une, séquentiellement, à l'unité de contrôle (unité de commande) qui les analyse et déclenche le traitement approprié en envoyant des signaux à l'unité arithmétique et logique. Le passage à l'instruction suivante est automatique.
- Le traitement peut nécessiter de faire appel aux unités d'entrés/sorties ou à la mémoire centrale.

Processeur = unité centrale de traitement = unité central = CPU(central processing unit)
Mémoire central = mémoire principal
Unité d'entrée/sortie = unité d'E/S = unité d'I/O = unité d'échange
Unité de contrôle = unité de commande
Unité arithmétique et logique = UAL = unité de traitement = unité de calcul

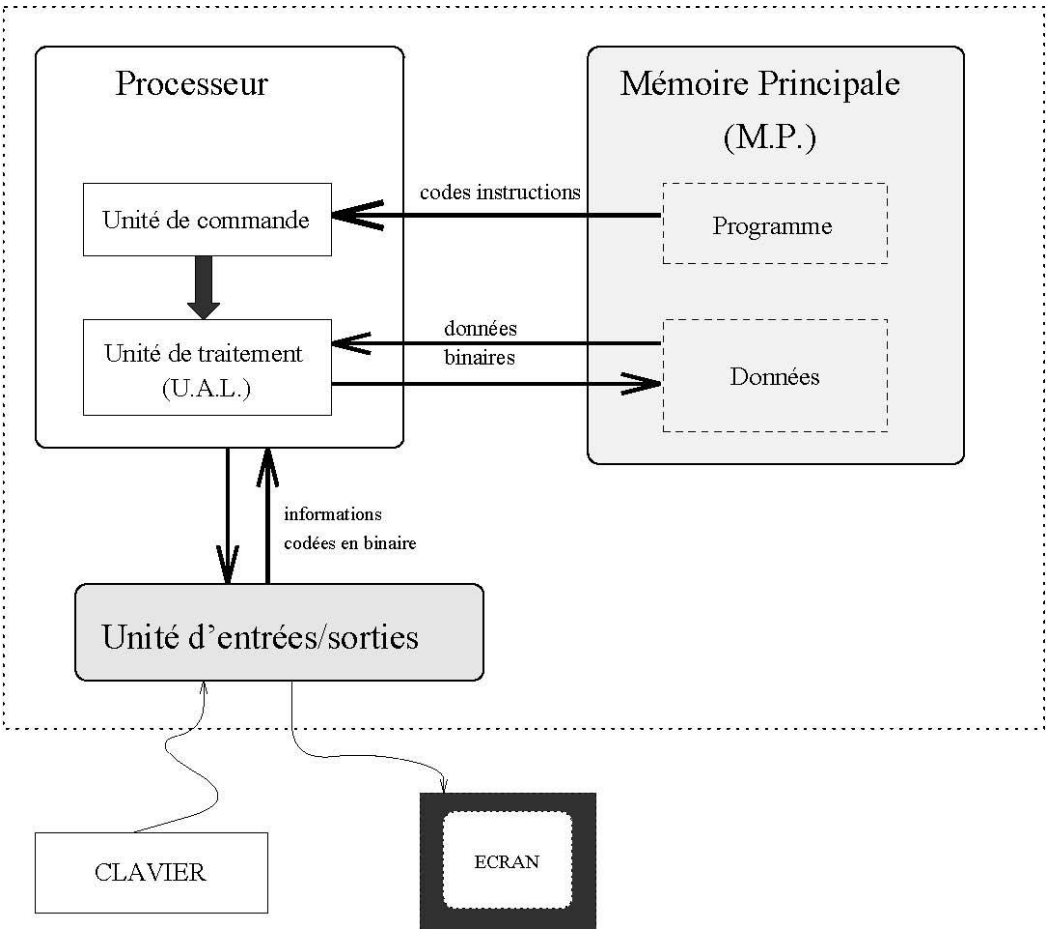


Schéma II.12 : schéma général d'un ordinateur

2.1. Unité central de traitement

L'unité centrale (CPU) est le "cerveau" de l'ordinateur. Son rôle est d'exécuter les programmes en mémoire centrale en chargeant les instructions, en les décodant et en les exécutant l'une après l'autre. L'UC est composée :

- d'une unité de commande qui charge les instructions et les décode,
- et d'une unité arithmétique et logique (UAL) qui exécute des opérations de la mémoire centrale.

- **Unité de commande :** l'unité de commande est constituée de plusieurs organes qui permettent la recherche en mémoire et le décodage d'une instruction. On trouve :
  - Le compteur ordinal qui est un registre contenant l'adresse de l'instruction à rechercher.
  - Le registre d'instruction qui reçoit l'instruction à exécuter.
  - Le décodeur de code opération qui détermine l'opération à effectuer parmi toutes celles possibles.
  - Le séquenceur qui génère les signaux de commande.
  - L'horloge qui synchronise toutes les actions de l'unité centrale.

L'Unité de commande comprend une mémoire très rapide qui lui permet de stocker des résultats temporaires ou des informations de commande. Cette mémoire est formée de quelques registres, chaque registre ayant une fonction particulière.
Le registre le plus important est le compteur ordinal (CO) qui pointe sur la prochaine instruction à exécuter. On trouve aussi le registre instruction (RI) qui contient l'instruction en cours d'exécution.
La plupart des ordinateurs contiennent également d'autres registres qui permettent aux programmeurs de stocker des résultats intermédiaires.

L'exécution d'une instruction par l'UC passe par les étapes suivantes:

1. Chargement de la prochaine instruction à exécuter depuis la mémoire jusque dans le registre instruction.
2. Modification du compteur ordinal pour qu'il pointe sur l'instruction suivante.
3. Décodage de l'instruction que l'on vient de charger.
4. Localisation dans la mémoire des éventuelles données utilisées par l'instruction.
5. Chargement des données, si nécessaire, dans les registres internes de l'unité centrale.
6. Exécution de l'instruction.
7. Stockage des résultats à leurs destinations respectives.
8. Retour à l'étape 1 pour exécuter l'instruction suivante.

- **Unité arithmétique et logique :** L'UAL réalise les opérations mathématiques et logiques nécessaires au fonctionnement de l'ordinateur et permet tout particulièrement le mode de calcul en virgule flottante. Elle possède les registres suivants :
  - Les registres arithmétiques.
  - Les registres de base et d'index (calcul d'adresse par rapport à une base ou index).
  - Les registres banalisés (ex : stockage de résultats intermédiaires).
  - Le registre d'état PSW (Program Status Word) qui indique l'état du système.

C'est l'UAL qui exécute les additions, les soustractions et toutes les opérations simples sur ses entrées, et qui produit un résultat placé dans le registre de sortie. Le contenu du registre de sortie peut alors être placé dans un autre registre avant de rejoindre, si nécessaire, la mémoire.
On peut regrouper les instructions en trois catégories :

- Registre−mémoire (2 à 3 cycles) les instructions registre−mémoire permettent de charger des mots dans des registres qui pourront, par exemple, être utilisés par d'autres instructions comme entrées de l'UAL.
- Registre−registre (1 cycle) les instructions registre−registre typiques chargent deux opérandes pris dans les registres, les placent dans les registres d'entrée de l'UAL, exécutent sur eux une certaine opération et remettent le résultat dans un registre.
- Mémoire−mémoire (plusieurs cycles) Une instruction mémoire−mémoire prend ses opérandes dans la mémoire et les place dans les registres entrées de l'UAL, exécute ensuite une opération, et place le résultat en mémoire.

2.2. Mémoire central

La mémoire centrale contient principalement deux types d'informations :

- Les instructions de différents programmes ;
- Les données nécessaires à l'exécution des programmes.

Les instructions sont stockées sous formes de code machine. On remarque qu'au niveau physique, la mémoire centrale ne contient que des bits, qui constituent l'unité élémentaire d'information. Un bit peut prendre soit la valeur 0, soit la valeur 1. Les bits sont regroupés par 8 pour constituer un caractère. Pourquoi ce nombre de 8 bits pour coder un caractère ? Car avec 8 bits il est possible de coder 2^8 = 256 informations différentes, ce qui est suffisant pour coder tous les caractères alphanumériques et tous les caractères spéciaux.

Parallèlement aux caractères, qui constituent une unité logique d'informations, la mémoire centrale est divisée physiquement en cellules. Chaque cellule correspond à un mot-mémoire et possède une adresse qui lui est propre. Ainsi les cellules peuvent être adressées séparément pour une opération de lecture ou d'écriture. La longueur d'un mot-mémoire varie d'une machine à l'autre, par exemple : 8, 16, 32, 64 bits. La valeur 32 tend à se généraliser dans la plupart des ordinateurs.

Le mot-mémoire (word) est l'unité d'information adressable, c'est-à-dire que toute opération de lecture ou d'écriture porte sur un mot-mémoire.
A chaque mot-mémoire est donc associé :

- une adresse (unique), indiquant la position en mémoire
- un contenu (instruction ou donnée)

- la capacité d'une mémoire s'exprime en fonction du nombre de mot-mémoire ainsi que du nombre de bits par mot.

  1 bit = 0 ou 1
  1 octet = 1 caractère = 1 byte = 8 bits
  1 Ko = 1024 octets
  1Mo = 1024 Ko
  1 Go = 1024 Mo

💣 Un registre est une cellule mémoire ayant une fonction particulière.

Dans la mémoire centrale on trouve deux types de registres,
- **le registre d'adresse :** qui contient l'adresse d'un mot-mémoire
- **le registre mot :** qui contient le contenu d'un mot-mémoire

💣 Un registre mot a la même taille qu'un mot-mémoire, alors qu'un registre d'adresse doit permettre d'adresser tous les mots de la mémoire.

Exemple :
Si la mémoire comporte 256 mots, le registre d'adresse doit avoir $\log_2(256) = \log_2(2^8) = 8$bits.
Un registre d'adresse de 32 bits permet d'adresser 2^8 bits permet d'adresser 2^{32} mots différents.

Les opérations possibles dans la mémoire centrale sont la lecture et l'écriture de mot-mémoire :
- **lecture :** le registre d'adresse contient l'adresse du mot à lire, le dispositif de sélection et d'accès permet de transférer le contenu de ce mot dans le registre mot.
- **Ecriture :** le registre d'adresse contient l'adresse d'un mot dans lequel on va écrire le contenu du registre mot.

Le temps nécessaire à l'écriture ou à la lecture d'un mot-mémoire est appelé le temps d'accès. Il varie entre quelques nano-secondes (1ns = 10^{-9} seconde) et quelques micro-secondes (1 μ =10^{-6}s)

2.3. Unité d'entrées/Sorties

Les unités d'E/S, ou unités d'échange sont des éléments qui permettent de transférer des informations entre l'unité central et les unités périphériques. Les unités d'E/S (I/O) les plus courantes sont :

- **le bus :** le bus est un simple câble de n lignes qui permet l'échanges et le transfert de données entre les éléments internes de l'ordinateur.
- **le DMA (Direct Memory Access):** permet à un périphérique d'accéder directement à la mémoire sans passer par le CPU, il est doté d'un registre d'adresses, d'un compteur, d'un registre de données et d'un dispositif de commande capable d'assurer le transfert.
- **le canal :** il est plus performant que le DMA, et permet à plusieurs périphériques de travailler simultanément.

Plus on s'éloigne du CPU plus les vitesses de transfert d'informations sont lentes, le CPU travaille donc plus vite que toutes les unités périphériques. Si le CPU devait lui-même s'occuper de toutes les opérations d'E/S, il passerait son temps à attendre, c'est pourquoi on utilise des processeurs spécialisés tels que les DMA et les canaux pour gérer ces E/S.

🕮 les unités périphériques se répartissent en deux classes :
- les périphériques d'entrées (clavier, sourie, etc…) ;
- les périphériques de sorties (écran, imprimante, etc…) ;
- les mémoires auxiliaires (disque dur, disquette, cassette, etc…).

💣 A chaque catégorie de périphériques est associé un contrôleur de périphériques qui s'occupe de la gestion de ces périphériques et de l'interface avec les unités d'E/S.

Le système d'exploitation et les fichiers

1. Système d'exploitation

Au cœur de l'ordinateur se trouve un ensemble de programmes que l'on nomme système d'exploitation (operating system). Ce logiciel, qui est à la base de toute exploitation de l'ordinateur, coordonne l'ensemble des taches essentielles à la bonne marche du complexe matériel et assure la gestion des ses ressources. Il facilite aussi le travail de l'utilisateur en se chargent de toutes les taches fastidieuses ou compliquées, comme le contrôle des périphériques ou le stockage et la gestion des fichiers. Il permet l'interaction directe entre l'homme et la machine, en offrant une interface convenable et en organisant le traitement et le stockage des programmes et des données.

Des exemples de systèmes d'exploitation : MS-DOS, Windows, Unix, VAX, VMS, etc…

🕮 Le système d'exploitation n'existait pas dans les machines de la première génération, où toute forme de programmation était l'affaire de l'utilisateur. Aujourd'hui, le système d'exploitation est devenu l'intermédiaire obligatoire entre l'utilisateur et la machine.

1.1. Définition: le système d'exploitation est l'ensemble des programmes qui se chargent de tous les problèmes relatifs à l'exploitation de l'ordinateur.

1.2. Rôle de système d'exploitation: le système d'exploitation a deux buts biens distincts :
- faciliter la tache de l'utilisateur en lui présentant une machine (virtuelle) plus simple à exploiter que la machine réelle et en assurant un service fiable
- assurer l'exploitation efficace et économique des ressources critiques de l'ordinateur.

2. Système d'exploitation MS-DOS

Il est conçu au début des années 80. Il signifie MicroSoft Disk Opérating (Système d'exploitation sur disque de MicroSoft) et indique le nom du programme qui nous permet de faire fonctionner notre ordinateur. Il permet de réaliser les taches suivantes :

- Gérer les fichiers et les répertoires ;
- Mettre les disques à jour ;
- Configurer le matériel ;
- Optimiser la mémoire ;
- Accélérer l'exécution des programmes.

2.1. Lancement du système d'exploitation MS-DOS: lorsqu'on met notre ordinateur sous tension, plusieurs mécanismes entrent en jeu. Différents messages très obscurs défilent sous nos yeux.
A l'allumage de l'ordinateur, celui-ci effectue automatiquement certaines taches. Tout d'abord, il vérifie le matériel mis à sa disposition, en commençant par la carte vidéo et la mémoire. Sur de nombreuses machines, l'intitulé de la carte et la mémoire disponible s'affiche. Puis le décompte de la mémoire s'enclenche. Les différents périphériques sont détectés et prêt à être utilisés, grâce à un driver (pilote) spécifique.
Enfin, l'ordinateur va lire (grâce aux informations contenue dans sa RAM), les fichiers de démarrage de notre ordinateur (IO.sys et MSDOS.sys) et les fichiers de configuration de notre système (AUTOEXEC.bat et CONFIG.sys).

💣 Tout cet enchaînement s'appelle la séquence de boot.

2.2. Description de quelques fichiers spéciaux

- **Autoexec.bat:** c'est un fichier qui s'exécute automatiquement chaque fois qu'on démarre notre ordinateur. Il contient des informations qui vont régler le fonctionnement de celui-ci.

- **Config.sys:** comme Autoexec.bat, c'est un fichier qui se lance automatiquement à chaque démarrage de l'ordinateur. Il contient des informations pour le réglage des périphériques.

- **Command.com:** contient les commandes internes que MS-DOS charge en mémoire RAM lors du démarrage du système.

- **Driver:** il se nomme pilote ou gestionnaire de périphérique. Un programme un peu particulier qui va indiquer à l'ordinateur comment utiliser le matériel mis à sa disposition.

2.3. Notions sur les fichiers

- **Les fichiers:** Le fichier est un moyen d'organiser les données pour fonctionner et pour démarrer certaines applications. Ainsi, les caractères, images, et sons sont stockés sous forme de fichiers.

- **Caractéristiques des fichiers:** les fichiers sont composés de deux parties : le nom et l'extension séparées par un point « . ». les fichiers possèdent aussi une heure et date de création ainsi que des attributs.
  Par exemple: command.com

- **Opérations sur les fichiers:**
  - La création d'un fichier
  - La consultation d'un fichier
  - Mise à jour d'un fichier (modification)
  - Suppression d'un fichier

- **Les répertoires:** un répertoire sert à regrouper un ensemble de fichiers associés lorsque leur nombre devient volumineux. Lorsqu'un répertoire est volumineux, la recherche d'un fichier devient difficile. Il est alors préférable de subdiviser ce répertoire (répertoire père) en sous répertoires c'est-à-dire en répertoire à l'intérieur du répertoire pour faciliter la recherche des fichiers.

2.4. Désignation des unités de disque: Chaque unité disque est désignée par une lettre. Les différentes désignations sont :

A : pour désigner le premier lecteur de disquette
B : pour désigner le deuxième lecteur de disquette
C : pour désigner le disque dur
D, E, F, etc… : pour désigner soit le lecteur CD-ROM et les disques amovibles, soit les unités logiques (partitions) du disque dur.

2.5. Quelques commandes de MS-DOS: MS-DOS possède deux genres de commandes : les commandes internes contenues dans le fichiers COMMAND.COM (elle sont fréquemment utilisées) et les commandes externes stockées sur disque et précisément dans le répertoire DOS.

- **La commande DIR**: La commande la plus utilisée du DOS est la commande DIR. Elle permet d'afficher à l'écran la liste des fichiers d'un disque.

Pour voir une liste de fichier, tapez DIR à la suite de l'indicatif du DOS puis appuyez sur Entrée.
C:\>DIR

Pour que l'ordinateur fasse une pause pendant le défilement de la liste tapez :
C:\>DIR / P

Pour avoir seulement une liste des noms des fichiers triés en colonne, tapez :
C:\>DIR / W

Pour voir les fichiers d'une disquette, faites suivre DIR par le nom du lecteur :
C:\>DIR A:

Utilisez le joker pour chercher des fichiers précis :
Le joker * sert à masquer un groupe de caractère dans un fichier.
Par exemple pour trouver tous les fichiers commençant par n, tapez :
DIR N*
Pour avoir les fichiers exécutables, taper
DIR *.exe

- **Lire ce que contient un fichier**: La commande à utiliser pour voir ce que contient un fichier s'appelle TYPE.

C:\>TYPE NOMFICH.EXT

Par exemple pour voire le contenu du fichier lisez.moi tapez
C:\>TYPE LISEZ.MOI
Suivi de la touche Entrée.

Pour faire une pause entre chaque écran, tapez:
C:\>TYPE LISEZ.MOI | MORE
(la barre verticale s'obtient par la combinaison [ALT]+[6]).

- **L'aide:** pour obtenir de l'aide sur une commande, il faut taper le nom de celle-ci suivi de / et d'un point d'interrogation (?).

C:\>nom de commande/ ?

Pour avoir une aide plus détaillée sur toutes les commandes MS-DOS, on tape uniquement :
C:\>HELP

Ou seulement sur une commande bien précise :
C:\>HELP nom de commande
Par exemple: C:\>HELP DIR

- **Comment changer de répertoire:** Pour passer à un autre répertoire du disque, il faut utiliser la commande CD suivie du nom du répertoire.
  C:\>CD WINDOWS

Pour passer dans la racine du disque, tapez
C:\>CD \

Pour revenir au répertoire précédent, tapez CD..
C:\>CD..

- **La touche F3:** Pour rappeler la dernière commande tapée, appuyez sur la touche F3

- **Copier un fichier:** La commande COPY sert à copier un fichier
  Elle s'utilise de cette façon :
  C:\>COPY source destination

Par exemple pour copier le fichier autoexec.bat dans le répertoire WINDOWS, tapez:
C:\>COPY AUTOEXEC.BAT C:\WINDOWS

On peut se servir du joker pour copier plusieurs fichiers en même temps.
Voir La commande DIR.

- **Supprimer un fichier:** Pour supprimer un fichier, il faut utiliser la commande DEL
  C:\>DEL source

Par exemple pour supprimer tous les fichiers commençant par un A, tapez:
C:\>DEL A*
Faites attention à ne pas supprimer n'importe quels fichiers !!

- **Renommer un fichier:** La commande REN du Dos vous permet de renommer un fichier. Le contenu et la position du fichier sur le disque ne change pas : seul son nom change.

Pour renommer le fichier AAAAA.doc en BBBBB.doc, il faut taper:
C:\>REN AAAA.doc BBBB.doc
Il faut mettre le nom actuel, suivi d'un espace et du nom futur.

Si le fichier ne se trouve pas dans le répertoire courant, il suffit de taper son chemin d'accès.
Par exemple si le fichier AAAA.doc se situe dans le répertoire C:\WORD, il faut tapez:
C:\>REN C:\WORD\AAAA.doc BBBB.doc

- **Formater une disquette:** Pour formater une disquette, il faut utiliser la commande FORMAT

Pour formater le disque A: tapez:
C:\>FORMAT A:

Pour effectuer un formatage rapide tapez:
C:\>FORMAT A:/Q

Pour faire une disquette système, tapez:
C:\>FORMAT A:/S

Vous pouvez combiner les deux: si vous voulez créer une disquette système rapidement, tapez :
C:\>FORMAT A /Q /S

- **Dupliquer des disquettes:** La commande DISKCOPY permet de dupliquer des disquettes. Vous ne pouvez pas utiliser DISKCOPY pour dupliquer deux disques de capacités différentes.
  Vous ne pouvez pas utiliser DISKCOPY avec un disque dur.
  Il suffit de taper (si vous lecteur de disquette est A):
  C:\>DISKCOPY A: A:

Après avoir chargé la première disquette, l'ordinateur vous demande d'insérer la disquette destination.

- **Editer un fichier :** Pour éditer un fichier quelconque, utilisez la commande EDIT. Pour cela tapez :

C:\>EDIT chemin\nom du fichier
Vous obtiendrez alors l'éditeur du DOS:
Pour éditer le fichier AAAA.doc du répertoire C:\WORD, tapez:
C:\>EDIT C:\WORD\AAAA.doc

2.6. Guide des commandes DOS

- CD: Cette commande permet de changer de répertoires.
- CLS: Cette commande efface l'écran, en retirant tous les messages encombrants. Cette commande est très utile.
- COPY: Cette commande permet de copier un fichier.
- DEL: Cette commande permet de supprimer des fichiers.
- DIR: Cette commande affiche la liste des fichiers et des répertoires présents sur le disque.
- DISKCOPY: Cette commande crée une copie exacte d'une disquette.
- FORMAT: Cette commande permet de formater des disques.
- MD: Cette commande permet de créer des répertoires.
- DELTREE: Cette commande permet de supprimer les répertoires, les sous répertoires et tous les fichiers.
- RD: Cette commande ne peut supprimer un répertoire que si ses répertoires et tous ses fichiers sont eux-mêmes supprimés au préalable.
- REN: Cette commande permet de renommer un fichier.
- TIME: Cette commande affiche la date et l'heure et vous permettez de les modifier.
- TYPE: Cette commande affiche le contenu d'un fichier à l'écran.
- TREE: Pour afficher l'arborescence d'un répertoire.

2.7. Les commandes à ne jamais utiliser: N'utilisez jamais les commandes du DOS suivantes:

- CTTY: Cette commande déconnecte le DOS du clavier et de l'écran.
- DEBUG: Il s'agit d'un utilitaire permettant de créer des programmes et de modifier la mémoire. Une mauvaise utilisation peut entraîner le bouleversement de votre disque dur.
- FDISK: Cette commande permet de créer des partitions. Mal utilisée, cette commande peut détruire toutes les informations de votre disque dur.
- FORMAT C: Cette commande formate votre disque dur, ne l'utilisez pas à moins d'être sur de ce que vous faites.
- RECOVER: Contrairement à ce qu'on pourrait croire, cette commande n'est pas un sauveteur. Cette commande détruit tous les fichiers et tous les répertoires de votre disque, et cela, sans de demande de confirmation.

3. Les fichiers

3.1. Introduction

Le fonctionnement de l'ordinateur pour sauvegarder un système d'information repose sur la notion de fichier. En effet, tout ce que traite un ordinateur ne peut l'être que sous forme de fichiers: programmes ou données.

- *Les programmes* : spécialement conçus et réalisés pour répondre au type du problème posé.
- *Les données* : relatives au problème et sur lesquelles vont agir les programmes pour aboutir aux résultats.

Ces données et programmes sont manipulés par la machine sous forme de fichiers. Chaque fichier est identifié par un nom et une extension.
Ainsi, on distingue deux types de fichiers:

- *Fichier programmes:* ce sont des fichiers qui contiennent les instructions du programme à exécuter. Ces instructions sont d'abord écrites dans un langage de programmation quelconque.
- *Fichier de données:* ce sont les fichiers qui regroupent les données qu'un programme peut éventuellement utiliser. Les fichiers de données sont évolutifs, c'est à dire qu'une donnée peut être: modifiée, supprimée, ajoutée, ou consultée.

Les fichiers qui nous intéressent ici, sont les fichiers de données structurées (des personnes, des ouvrages, des produits,…), crées par l'utilisateur pour la gestion d'une application donnée.

Exemple: pour la gestion d'une entreprise commerciale, les fichiers : clients, fournisseurs, produits, factures, commandes,…etc, forment une base de données. Tout ces fichiers sont reliés entre eux: un fournisseur fournit un produit, un client passe une commande pour l'achat d'un produit, il règle une facture,…etc. les fichiers précédents forment une base de données.

3.2. Concepts d'un fichier

- Indépendamment du support physique utilisé pour le stocker, un fichier est un ensemble d'informations de même nature qui décrivent des individus ou des objets possédant des caractéristiques communes.
- Un fichier peut être assimilé à un classeur regroupant des fiches.
- Chaque fiche contient un ensemble d'informations qui décrivent un individu (objet ou entité) bien précis. Une fiche forme un *enregistrement*.
- Dans un enregistrement, les informations sont écrites dans un ordre fixe et immuable. Chaque information est appelée *champ*.
- Chaque enregistrement est identifié de façon unique, par une information appelée *clé* ou identificateur.
- Un *fichier* est un ensemble de *champs* regroupés sous forme *d'enregistrements,* identifié par une *clé*.
- Un *fichier logique* est décrit par sa structure, c'est à dire, les différents champs qu'il regroupe. les enregistrements d'un fichier logique sont dits *articles* (ou *enregistrements logiques*).
- Un *fichier physique* est le résultat du stockage du fichier logique sur un support physique. Les enregistrements du fichier physique sont dits enregistrements physiques.
- Un enregistrement physique représente la quantité d'informations échangée entre la MC et l'unité de stockage.
- Le *facteur de blocage* représente le nombre des enregistrements logiques dans un enregistrement physique. $facteurdeblocage = \frac{nombred'enregistrementslogiques}{nombred'enregistrementsphysique}$

3.3. Exploitation d'un fichier

Par exploitation d'un fichier, on désigne la manière de retrouver l'emplacement d'un enregistrement sur un support physique. Cette exploitation se base sur la connaissance des paramètres suivants :

- ✓ Taux de remplissage
- ✓ Zone de débordement
- ✓ Lien de chaînage
- ✓ Code de validité

- **Le taux de remplissage:** désigne le rapport entre le nombre d'enregistrements effectivement stockés et le nombre d'emplacements disponibles pour le fichier sur le support.

$$\zeta_r = \frac{nombre\ d'enregistrements\ effectivements\ tockés}{nombre\ d'emplacements\ disponibles\ pour\ le\ fichier}$$

- **La zone de débordement:** est une zone spécifique sur le support, utilisée pour stocker les enregistrements qui n'ont pu être stockés dans la zone.
- **Un lien de chaînage:** est le contenu d'une zone de l'enregistrement qui indique l'adresse de l'enregistrement suivant (logiquement).
- **Le code de validité:** est le contenu d'une zone de l'enregistrement qui signale la présence ou l'absence logique d'un enregistrement (c'est la suppression logique d'enregistrements).

3.4. Caractéristiques d'utilisation des fichiers

Un fichier est crée pour être utilisé pour la gestion d'une application, il subit plus ou moins de manipulations (consultation, mise à jour, …). Ces manipulations déterminent les caractéristiques d'utilisation du fichier, à savoir:

- ✓ L'activité d'un fichier
- ✓ Le volume (la taille) d'un fichier
- ✓ L'accroissement d'un fichier

3.4.1 L'activité d'un fichier: l'activité d'un fichier caractérise l'ensemble des manipulations effectuées sur le fichier. Elle est définie par les quatre caractéristiques suivantes :

- ✓ taux de consultation
- ✓ fréquence de consultation
- ✓ taux de renouvellement
- ✓ stabilité du fichier

- **Le taux de consultation:** désigne le rapport entre le nombre d'enregistrements consultés (ou modifiés) et le nombre total d'enregistrements du fichier:

$$\zeta_c = \frac{nombre\ d'enregistrements\ consultés}{nombre\ total\ d'enregistrements}$$

- **La fréquence de consultation:** désigne une fréquence annuelle, c'est à dire le nombre d'accès à un enregistrement du fichier pour simple consultation ou mise à jour.
- **Le taux de renouvellement:** est relative à une période donnée. Il exprime le nombre relatif de nouveaux enregistrements qui sont insérés dans le fichier.
- **La stabilité du fichier:** *est relative à une période donnée.* Un fichier est dit stable pendant une période si le nombre d'enregistrements crées est approximativement égal au nombre d'enregistrements supprimés.

3.4.2. Le volume ou la taille d'un fichier : il désigne le nombre de caractères contenus dans le fichier. C'est une caractéristique très importante pour l'utilisation future du fichier (implantation physique, estimation du temps de manipulation du fichier,… .

3.4.3 L'accroissement d'un fichier : il désigne le nombre d'enregistrements crées par rapport à celui des enregistrements supprimés. Il est dit négatif lorsque le nombre d'enregistrements supprimés est supérieur au nombre d'enregistrements crées.

3.5. Typologie des fichiers5

On peut distinguer plusieurs types de fichiers selon :

- ✓ La nature des informations qu'il contient.
- ✓ La durée de vie
- ✓ Le type de support utilisé pour son stockage
- ✓ L'organisation des informations

3.5.1. Types de fichiers selon la nature des informations : un fichier peut contenir deux types d'informations : des données ou des programmes et selon le cas, on parle de fichier de données ou de fichier programme. Les données contenues dans un fichier de données peuvent être de différents types et on parle de fichier d'entiers, de réels, de caractères, d'étudiants, de client, …etc.

Exemple : les fichiers module et étudiant sont deux fichiers de données.

Le fichier calcule est un fichier programme qui permet de calculer la moyenne des étudiants.

3.5.2. Types de fichiers selon leur durée de vie : selon le rôle des informations contenues dans un fichier, leur utilité et importance, un fichier peut exister de façon permanente ou temporaire. Ainsi, on peut classifier les fichiers en quatre types :

- ✓ fichiers permanents
- ✓ fichiers mouvements
- ✓ fichiers de manœuvre
- ✓ fichiers intermédiaires

- **Un fichier permanent :** est un fichier dont les informations sont d'une importance vitale au sein de l'application pour laquelle il a été conçu. Son contenu ne subit pas de fréquentes modifications.
  Exemple : le fichier étudiants.

- **Un fichier mouvement :** sert à mettre à jour un fichier permanent. Il est caractérisé par une durée de vie courte.
  Exemple : on considère le cas d'un établissement scolaire qui gère ses étudiants à travers un fichier étudiants.
  Au début de chaque session, il y a une nouvelle section qui commence. Les nouveaux inscrits sont d'abord stockés dans un fichier inscrit, puis une fois leurs scolarités justifiées, ils sont ajoutés au fichier étudiant qui regroupe les informations concernant tous les étudiants de l'établissement, les nouveaux et les anciens.
  Dans cet exemple, le fichier inscrit est un fichier mouvement. Il sert à mettre à jour le fichier étudiant chaque fois qu'il y a de nouvelles inscriptions.

- **Un fichier de manœuvre :** trouve sa raison d'être lorsqu'il n'y a pas assez d'espace en MC pour contenir toutes les données nécessaires à un certain traitement. Sa durée de vie est limitée par celle du traitement qui l'a crée.

- **Un fichier intermédiaire :** contient des résultats d'un traitement donné pour être utilisés soit durant le même traitement, soit par d'autres traitements ultérieurs. Il permet l'échange de données entre programmes, contrairement au fichier de manœuvre qui ne communique ses données qu'au même traitement qui l'a généré. En outre la durée de vie d'un fichier intermédiaire n'est pas limitée par celle du traitement l'ayant crée, afin de permettre aux autres traitements de l'utiliser.

3.5.3. Type de fichiers selon le support utilisé : bien que le contenu d'un fichier reste le même quel que soit le support utilisé pour le stocker, certaines caractéristiques du fichier sont liées étroitement avec la nature de ce support et notamment le mode d'accès aux données qu'il contient. Par exemple, pour un fichier stocké sur une bande magnétique, seul un accès séquentiel peut être pratiqué, alors que sur un disque magnétique, l'accès technologies de fabrication utilisées pour les unités de stockage, en général.

3.5.4 Type de fichiers selon l'organisation des informations : l'organisation adoptée pour un fichier est une de ses caractéristiques les plus importantes, puisqu'elle permet de définir la manière d'accéder aux informations qu'il contient.

- ✓ Organisation séquentielle
- ✓ Organisation séquentielle indexée
- ✓ Organisation aléatoire (ou relative)

3.6. Opération sur les fichiers

Les opérations qu'on peut effectuer sur les fichiers sont les suivantes :

- Création
- Suppression
- Tri
- Fusion
- Eclatement

3.7. Opération sur les enregistrements

Apres la création du fichier, plusieurs opérations peuvent être effectuée sur ses enregistrements:

- *La création :* qui consiste la saisie d'un enregistrement
- *La suppression :* qui revient à effacer un ou plusieurs enregistrements. Si tous les enregistrements sont supprimés, on obtient un fichier vide.
- *La modification :* consiste à changer les valeurs d'un ou de plusieurs champs.
- *La consultation :* qui revient à lire la valeur d'un ou de plusieurs champs d'un enregistrement.

3.8. Les fichiers et le système d'exploitation

Pour exploiter facilement les fichiers (création, suppression, …), le système d'exploitation offre à l'utilisateur un certain nombre d'outils regroupés sous le nom de *système de gestion de fichiers (SGF)*. Le SGF joue le rôle d'intermédiaire entre l'utilisateur et les fichiers stockés sur le support correspondant.

En outre, le SGF permet de protéger les fichiers (contre l'écriture, la lecture, …), de les partager avec d'autres, de bien gérer l'espace de stockage,…etc.

3.9. Organisation des fichiers

Le but visé par l'organisation est de ranger les enregistrements dans un ordre bien déterminé de manière à :

- Accéder le plus rapidement possible aux enregistrements du fichier.
- Occuper le moins de place possible sur le support et ce afin d'éviter les pertes d'espace sur le support et faciliter l'insertion de nouveaux éléments.

L'organisation d'un fichier est choisie lors de sa création et elle peut être :

- Séquentielle
- Séquentielle indexée
- Aléatoire

3.9.1 Organisation séquentielle : dans une organisation séquentielle, les articles sont enregistrés dans l'ordre où ils se présentent à la saisie. Ils sont placés les uns à la suite des autres. Ainsi, un nouvel enregistrement saisi est toujours écrit à la fin du fichier.

Pour rechercher un $i^{ème}$ enregistrement, il faut d'abord parcourir les (i-1) enregistrements qui le précédent.

Caractéristiques

- les enregistrements sont écrits selon l'ordre de leur arrivée.
- l'insertion de nouveaux enregistrements se fait uniquement en fin de fichier.
- Chaque enregistrement possède un prédécesseur (sauf le premier) et un successeur (sauf le dernier).

Avantages

- Simple à appliquer
- Facile à implémenter
- Implémenter sur n'importe quel type de support (bande magnétique, disque magnétique,…..).
- Economique en espace mémoire.

Inconvénients

- elle n'est pas pratique pour les fichiers de grande taille, car elle nécessite des temps d'accès très longs.
- Pour insérer de nouveaux enregistrements au milieu du fichier, il faut copier le fichier intégralement. En effet, il faudra copier une partie du fichier jusqu'à la position où devra se faire l'insertion, enregistrer le nouvel enregistrement, puis copier la deuxième partie du fichier.

3.9.2. Organisation séquentielle indexée : comme pour l'organisation séquentielle, les enregistrements sont écrits dans l'ordre de leur arrivée. Seulement, ici à chaque fois qu'un enregistrement est écrit, une clé lui est associée. Cette clé, ainsi que l'adresse relative de l'enregistrement dans le fichier sont inscrites dans une table appelée *table d'index.*

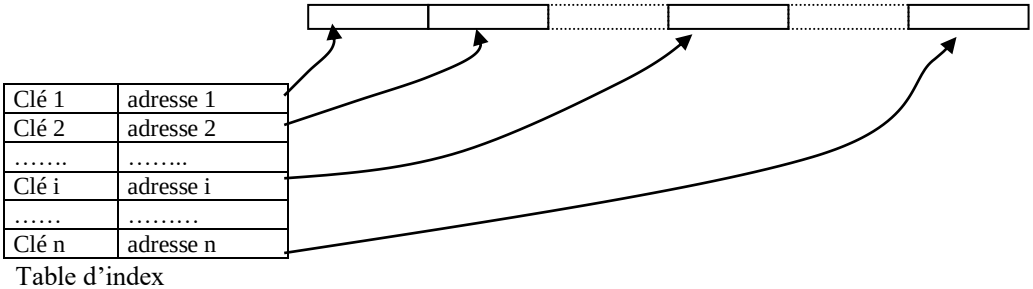


Schéma III.1 : Fichier organisé en séquentiel indexé

🕮 L'insertion d'un nouvel article se fait à la fin du fichier. A chaque insertion, la clé de l'article inséré, ainsi que son adresse dans le fichier sont ajoutés à la table d'index.

🕮 A l'ouverture du fichier, la table d'index est chargée en mémoire. Pour accéder à un enregistrement, il suffit d'utiliser sa clé pour retrouver son adresse dans la table d'index.

Caractéristiques

- Les articles sont écrits dans l'ordre de leur arrivée.
- Une table d'index est associée au fichier. Elle contient les clés des articles et leurs adresses relatives dans le fichier.
- La recherche d'un article est facilitée par la table d'index. Il suffit de connaître la clé pour connaître l'adresse à laquelle l'article est stocké.

Avantages

- La table d'index permet de retrouver plus facilement les articles dans le fichier.
- La table d'index permet d'accéder rapidement et directement (si le support de stockage le permet) aux articles.

Inconvénients

Quand le fichier est trop grand, la table d'index devient de taille importante, ce qui génère deux inconvénients :

- La table occupe un espace mémoire non négligeable.
- La manipulation de la table devient lourde.

3.9.3. Organisation aléatoire : ce mode d'organisation repose sur un principe simple :

Le fichier est organisé en pages (des pages principales et des pages de débordement). Dans une même page, les enregistrements du fichier sont écrits selon leur ordre d'arrivée. Pour déterminer le numéro de la page où devrait être écrit un enregistrement donné, un calcul est effectué sur sa clé. Ce calcul est réalisé à partir d'une fonction appelée fonction de répartition (ou fonction de randomisation).

Si la page devant contenir un nouvel enregistrement est saturée, elle sera chaînée à la page de débordement où sera écrit ce nouvel enregistrement. En général, les pages de débordement sont gérées de la même manière que les pages principales, c'est à dire qu'une fonction de répartition est appliquée pour déterminer le numéro de la page de débordement où sera écrit l'enregistrement qui a causé le débordement.

Exemple : à créer le fichier étudiant avec les enregistrements dont les clés sont : 6 , 8 , 10 , 13 , 18 , 9 , 20 , 2 et 29.

Le fichier étudiant est un fichier aléatoire organisé en 9 pages principales et 4 pages de débordement. La fonction de répartition (pour le calcul des numéros de pages) est définie comme suit :

N=clé modulo [nombre de pages principales], c'est-à-dire que le numéro de la page N est le reste de la division de la clé sur le nombre de pages principales.

Ainsi, pour écrire les enregistrements précédents, on doit appliquer cette fonction :

N_1 = 6 modulo [9] = 6 donc, 6 sera écrit dans la page numéro 6 (les pages sont numérotées à partir de 0).
N_2 = 8 modulo [9] = 8 ;
N_3 = 10 modulo [9] = 1 ;
N_4 = 13 modulo [9] = 4 ;
N_5 = 18 modulo [9] = 0 ;
N_6 = 9 modulo [9] = 0 ;
N_7 = 20 modulo [9] = 2 ;
N_8 = 2 modulo [9] = 2 ;
N_9 = 29 modulo [9] = 2 ;

On remarque, sur le schéma, que la page numéro 2 est saturée. Essayons d'insérer l'enregistrement 47.
Nous avons : 47 modulo [9] = 2, puisqu'il n'y a plus de place dans la page 2, ce nouvel enregistrement sera écrit dans une page de débordement. Pour déterminer le numéro de ctte page de débordement, on utilise toujours la fonction de répartition précédente :
Nombre de pages de débordement = 4.
D'où N = 47 modulo [4] = 3.
L'enregistrement sera écrit dans la page de débordement numéro 3. Un lien de chaînage va lier la page principale numéro 2 à l'enregistrement en page de débordement.

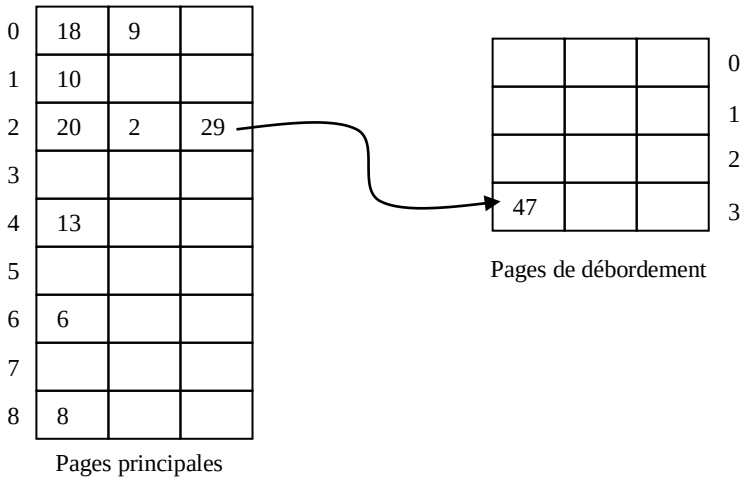


Schéma III.2. : le fichier étudiant organisé en aléatoire

- Pour obtenir une répartition équilibrée des enregistrements sur les pages et éviter des débordements fréquents, il faut bien choisir les paramètres suivants et ce dés la création du fichier :
  - Taille d'une page
  - Nombre de pages
  - Fonction de répartition

Avantages

- Le principe du calcul de l'adresse d'un enregistrement (le numéro de page) facilite son insertion dans le fichier.
- L'organisation aléatoire permet d'avoir bons temps de réponse du fait que l'accès aux enregistrements se fait par calcul d'adresse.

Inconvénients

- Elle ne permet pas le traitement efficace du fichier en séquentiel, chose due au débordement des pages principales.
- En cas d'un nombre important de débordement, une réorganisation du fichier s'impose.

3.10. Choix d'une organisation

L'organisation d'un fichier se choisit de manière à satisfaire les objectifs suivants :

- Permettre un accès rapide aux enregistrements du fichier
- Assurer un gain en place sur le support de stockage
- Faciliter l'insertion de nouveaux enregistrements.

1. Rôle du processeur

Dans les premiers temps de l'informatique, les ordinateurs étaient programmés en langage machine ou en assembleur. Pour faciliter la programmation, les concepteurs de micro-processeurs dotèrent ceux-ci d'instructions de plus en plus complexes permettant aux programmeurs de coder de manière plus concise et plus efficace les programmes.

Lorsque les premiers langages de haut niveau remplacèrent l'assembleur, cette tendance s'accentua. Les concepteurs de micro-processeurs s'efforcèrent de combler le fossé entre le langage machine et les langages de haut niveau. Des instructions proches des constructions typiques des langages de haut niveau furent ajoutées aux micro-processeurs. L'idée était de faciliter la compilation des langages de haut niveau au détriment de la complexité des micro-processeurs. On ajouta par exemple des instructions spécifiques pour les appels ou les retours de fonctions ou des instructions spéciales pour les boucles pour décrémenter un registre et faire un saut si le résultat est non nul, tout ça en une seule instruction machine.

Un élément qui favorisa cette complexification des micro-processeurs était le manque (à cause du prix) de mémoire et la lenteur de celle-ci. L'absence de beaucoup de mémoire force les programmes à être le plus compacts possible. Pour réduire le nombre d'instructions nécessaires à un programme, il faut que chaque instruction fasse plusieurs opérations élémentaires. La lenteur relative de la mémoire pousse aussi à avoir des instructions complexes. Il n'y a alors moins de codes d'instructions à lire en mémoire et le programme en est accéléré.

Comme la densité d'intégration des transistors était encore faible, les micro-processeurs possédaient très peu de registres internes. De plus, un grand nombre de registres auraient nécessité plus de bits pour coder les instructions. Pour compenser cette lacune en registres, certaines instructions étaient capables, par exemple, de charger deux valeurs en mémoire, de faire la somme et de ranger le résultat en mémoire. Il y avait de nombreux modes d'adressage et tous les modes d'adressages étaient possibles à toutes les instructions. On parle alors d'*orthogonalité* lorsque toutes les instructions peuvent utiliser tous les modes d'adressages. L'aboutissement de ce type de jeux d'instructions est celui du VAX.

La complexification des jeux d'instructions a pour effet de compliquer notablement la phase de décodage des instructions. On peux constater que sur certains micro-processeurs à jeu d'instructions complexe, la moitié des transistors sur la puce de silicium est consacrée au décodage des instructions et au contrôle de l'exécution de celles-ci.

Comme le décodage des instructions est rendu difficile par un jeu d'instructions complexes, l'exécution des instructions simples est ralentie par un décodage compliqué.

Lorsque le jeu d'instructions est complexe, la plupart des compilateurs n'utilisent pas tout le jeu d'instructions. Il se contente souvent d'un nombre réduit d'instructions. Le résultat est que les instructions les plus puissantes sont très rarement utilisées. On arrive alors au paradoxe suivant. Les instructions complexes qui ne sont pratiquement pas utilisées ralentissent les instructions simples qui sont utilisées la plupart du temps.

Des études statistiques sur des programmes tels des systèmes d'exploitation ou des applications réelles ont montré les faits suivants.

- 80 % des programmes n'utilisent que 20 % du jeu d'instructions.
- Les instructions les plus utilisées sont :
    - les instructions de chargement et de rangement,
    - les appels de sous-routines.

- Les appels de fonctions sont très gourmands en temps : sauvegarde et restitution du contexte et passage des paramètres et de la valeur de retour.
- 80 % des variables locales sont des entiers.
- 90 % des structures complexes sont des variables globales.
- La profondeur maximale d'appels imbriqués et en moyenne de 8. Une profondeur plus importante ne se rencontre que dans 1% des cas.

L'apparition de l'architecture RISC vint de la volonté de favoriser au maximum les instructions simples qui constituent la grande partie des programmes. L'idée est de supprimer les instructions complexes et les modes d'adressage élaborés afin d'augmenter la fréquence d'horloge et d'augmenter ainsi la vitesse d'exécution de toutes les instructions. La fréquence d'horloge d'un micro-processeur est souvent dictée par les instructions les plus complexes qui prennent plus de temps.

La philosophie essentielle des processeurs RISC est d'avoir un nombre important de registres. Des instructions de chargement et de rangement avec quelques modes d'adressage sont les seules à faire les échanges avec la mémoire. Toutes les autres instructions travaillent uniquement avec les registres.

L'apparition des micro-processeurs RISC est en partie due l'augmentation de la mémoire disponible sur les ordinateurs. Celle-ci n'est plus une limitation à la taille des programmes. Un autre facteur important est le fossé qui s'est creusé entre la vitesse des processeurs et celle de la mémoire. Les processeurs ont une fréquence d'horloge élevée par rapport à la vitesse de la mémoire. Chaque accès à la mémoire les pénalise. Ceci accentue le rôle d'un nombre important de registres qui évitent les accès à la mémoire. La lenteur de la mémoire est en partie compensée par la présence de caches faits de mémoires rapides. Ceux-ci sont très efficaces pour lire les instructions des programmes puisque ces accès à la mémoire se font à des cases mémoire contiguës.

1.1. Calcul de CPI (Cycle per Instruction)

En architecture d'ordinateur, instructions par cycle d'horloge (instruction par cycle ou IPC) est un terme utilisé pour décrire un aspect de la performance d'un microprocesseur: le nombre moyen d'instructions exécutées pour chaque cycle du signal d'horloge.

1.1.1 Estimation de performance

Le nombre d'instructions par seconde d'un processeur peut être déterminé en multipliant l'IPC par la fréquence d'horloge (mesuré en cycles par seconde ou Hertz) du microprocesseur en question. Le nombre d'instructions par seconde est un indicateur approximatif des performances d'un microprocesseur.

Le nombre d'instructions exécutées par cycle n'est pas une constante pour un processeur donné; il dépend de la manière dont le logiciel en cours d'exécution interagit avec le microprocesseur, et évidemment de l'ensemble de la machine, et en particulier de la hiérarchie mémoire. Cependant, certaines fonctions d'un microprocesseur ont pour effet d'accroitre les valeurs d'IPC au-dessus de la moyenne; la présence de multiples unités arithmétiques et logiques (une ALU est une unité d'un microprocesseur qui peut effectuer des opérations élémentaires d'arithmétique et logique), et un court pipeline. Par comparaison de jeu d'instructions, un jeu d'instructions simple peut amener une configuration plus élevée en termes d'IPC que l'implémentation d'un jeu d'instructions plus complexe en utilisant un microprocesseur de même technologie; cependant, le jeu d'instructions complexe peut effectuer un travail utile avec moins d'instructions.

1.1.2. Facteurs influençant l'IPC

Un même niveau d'instructions par seconde peut être obtenu avec un haut nombre d'IPC et une basse cadence d'horloge (comme l'AMD Athlon et l'Intel Core 2), ou avec un bas nombre d'IPC et une haute cadence d'horloge (comme le Intel Pentium 4 et dans une moindre mesure l'AMD Bulldozer). Les deux sont des conceptions valables, et le choix entre les deux est dicté par l'historique, les contraintes d'ingénierie, ou par la pression du marketing.

1.1.3. Rapidité de calcul

Le travail qui peut être fait avec n'importe quel ordinateur dépend de nombreux facteurs autres que la vitesse du processeur. Ces facteurs incluent la microarchitecture, l'agencement interne de la machine, la vitesse du système de support de stockage, la vitesse des autres périphériques attachés, l'efficacité du système d'exploitation, et surtout la conception globale du logiciel utilisé.

Pour les utilisateurs et acheteurs d'un ordinateur, les instructions par cycle ne sont pas une information particulièrement utile des performances de leur système. Pour avoir une information plus utile pour eux, un logiciel de benchmark est mieux indiqué. Avoir conscience de son existence est utile, en ce qu'il fournit un exemple facile à comprendre pourquoi la fréquence d'horloge n'est pas le seul facteur indiquant les performances d'un ordinateur.

1.2. Les processeurs CISC et RISC.

Les processeurs généraux actuels se répartissent en deux grandes catégories appelées CISC pour *Complex Instruction Set Computer* et RISC pour *Reduced Instruction Set Computer*. Les processeurs de ces deux catégories se distinguent par la conception de leurs jeux d'instructions. Les processeurs CISC possèdent un jeu étendu d'instructions complexes. Chacune de ces instructions peut effectuer plusieurs opérations élémentaires comme charger une valeur en mémoire, faire une opération arithmétique et ranger le résultat en mémoire. Au contraire, les processeurs RISC possèdent un jeu d'instructions réduit où chaque instruction effectue une seule opération élémentaire. Le jeu d'instructions d'un processeur RISC est plus uniforme. Toutes les instructions sont codées sur la même taille et toutes s'exécute dans le même temps (un cycle d'horloge en général). L'organisation du jeu d'instructions est souvent appelé ISA pour *Instruction Set Architecture*
La répartition des principaux processeurs dans les deux catégories est la suivante:

CISC	**RISC**
S/360 (IBM) VAX (DEC) 68xx, 680x0 (Motorola) x86, Pentium (Intel)	Alpha (DEC) PowerPC (Motorola) MIPS PA-RISC (Hewlett-Packard) SPARC

Les principales caractéristiques des processeurs RISC sont les suivantes:

Codage uniforme des instructions: Toutes les instructions sont codées avec un même nombre de bits, généralement un mot machine. L'*op-code* se trouve à la même position pour toutes les instructions. Ceci facilite le décodage des instructions.

Registres indifférenciés et nombreux: Tous les registres peuvent être utilisés dans tous les contextes. Il n'y a par exemple pas de registre spécifique pour la pile. Les processeurs séparent cependant les registres pour les valeurs flottantes des autres registres.

Limitation des accès mémoire: Les seules instructions ayant accès à la mémoire sont les instructions de chargement et de rangement. Toutes les autres instructions opèrent sur les registres. Il en résulte une utilisation intensive des registres.

Nombre réduit de modes d'adressage: Il n'y pas de mode d'adressage complexe. Les modes d'adressages possibles sont généralement immédiats, direct, indirect et relatifs. Un registre est souvent fixé à la valeur 0 afin d'obtenir certains modes d'adressages simples comme cas particulier d'autres modes d'adressage.

Nombre réduit de types de données: Les seuls types de données supportés sont les entiers de différentes tailles (8, 16, 32 et 64 bits) et des nombres flottants en simple et double précision. Certains processeurs CISC comportent des instructions pour le traitement des chaînes de caractères, des polynômes ou des complexes.

La simplification du jeu d'instructions a reporté une partie du travail sur le compilateur. Ce dernier joue un rôle essentiel dans l'optimisation du code engendré.

2. Le microprocesseur MIPS R3000

Annoncé en 1988 par la société MIPS Computer Systems, le MIPS R3000 fut le premier processeur à vocation industrielle. Il succède au MIPS R2000, dérivé des travaux de l'Université de Stanford (1985). L'une des principales caractéristiques de ce processeur est sa simplicité. En effet, la force des processeurs MIPS réside dans la relation très étroite entre l'architecture du processeur et le compilateur associé. En 1991, Le MIPS R6000 fut commercialisé à la fréquence de 66 MHz. Développé par Bipolar Integrated Technology, ce processeur, reprenait pour l'essentielle l'architecture du R3000 en introduisant certaines particularités telles qu'un cache secondaire externe, un identificateur de processus dans les caches primaires et dans le cache secondaire, une prédiction de branchement, une table de traduction d'adresses séparée de l'unité entière et implémentée sur le cache secondaire, etc...

Cependant, ce processeur réalisé en technologie ECL, ne connut pas le succès escompté. Bien que cette technologie soit potentiellement plus rapide que la technologie CMOS, les avantages de cette dernière en matière d'intégration, de consommation et de coût de revient l'imposèrent comme la plus utilisée sur le marché. Le MIPS R6000 reste à ce jour le processeur le plus connu de cette technologie. Le MIPS R4000 est la troisième génération de processeurs MIPS (1992). Il est le premier à mettre en œuvre une architecture 64 bits qui reste cependant compatible avec les générations MIPS précédentes (un mode 32 bits assure la compatibilité). Afin d'offrir une cadence de séquencer très élevée (dès sa sortie, le R4000 est annoncé à la fréquence de 100 MHz), la technique du super pipeline a été utilisée.

Destiné au marché des stations de travail, ce processeur fut initialement prévu en trois versions : R4000PC (avec cache primaire seulement), R4000SC (avec cache secondaire) et R4000MC (version destinée aux systèmes multiprocesseurs). Le MIPS R8000 est fondu par Toshiba (1994). Il vise clairement le marché des applications scientifiques et techniques (stations de travail graphiques 3-D), les serveurs de base de données, ainsi que le domaine des multiprocesseurs.

Ce processeur est le premier à mettre en œuvre une architecture super scalaire. Jusqu'à quatre instructions peuvent être séquencées en parallèle à chaque cycle grâce à ses deux unités flottantes, ses deux unités entières et ses deux unités de lecture/écriture. Par ailleurs ce processeur met en œuvre des mécanismes originaux afin d'accroître ses performances.

Commercialisé initialement à la fréquence de 75 MHz, MIPS a annoncé tout dernièrement une version à 90 MHz.

Nous décrivons dans la suite, les principales caractéristiques du processeur MIPS R3000 (pour des raisons de simplicité, tous les mécanismes matériels de gestion de la mémoire virtuelle ont été délibérément supprimés), en comparant leurs jeux d'instructions, leurs pipelines, la structure des caches et les mécanismes de gestion de la mémoire.

3. Structure externe du microprocesseur MIPS R3000

Le processeur MIPS R3000 est un processeur 32 bits industriel conçu dans les années 80. Son jeu d'instructions est de type RISC. Le MIPS R3000 est un circuit intégré de forme DIL de 40 pattes. Il est équipé d'un bus de données de 16 bits et un bus d'adresses de 20 bits et fonctionne à des fréquences diverses selon plusieurs variantes: 5, 8 ou 10 MHz, comme le montre la figure suivante:

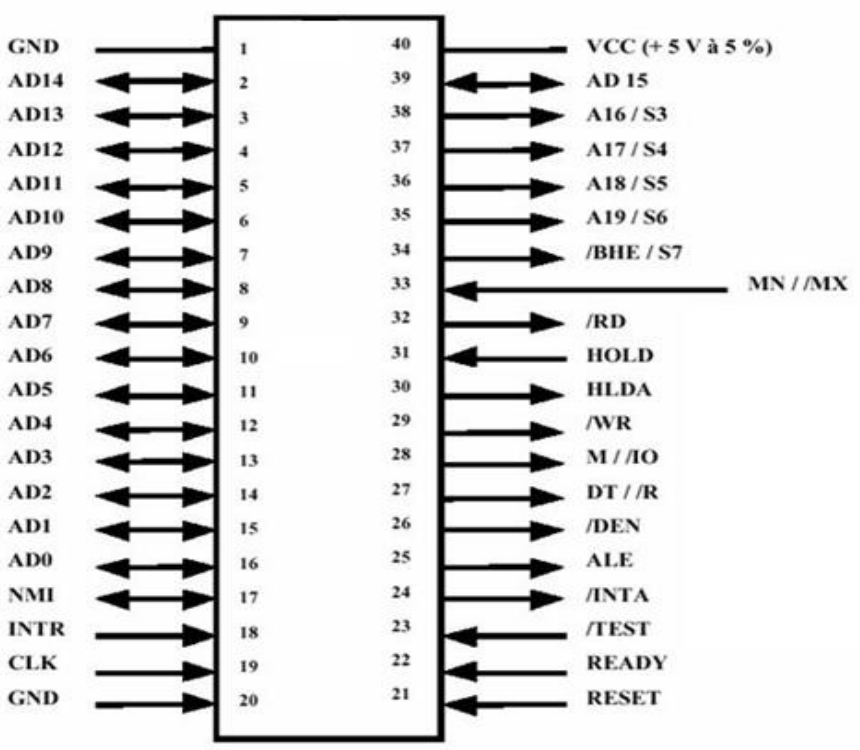


Schéma IV.1 : MIPS R3000

4. Structure interne du microprocesseur MIPS R3000

Il existe deux unités internes distinctes: l'Unité d'exécution (UE) et l'Unité d'Interfaçage avec le Bus (UIB). Le rôle de l'unité d'interfaçage avec le bus est de récupérer et stocker les informations à traiter, et d'établir les transmissions avec les bus du système.

L'unité d'exécution exécute les instructions qui lui sont transmises par l'unité d'interfaçage avec le bus. L'image ci-dessous résume les notions présentées ici. Le microprocesseur pris comme exemple est le MIPS R3000. Les processeurs actuels traitent les informations de la même façon.

Nous pouvons à présent examiner plus en détail le traitement des instructions par l'unité d'exécution et l'unité d'interfaçage avec le bus. Avec le microprocesseur MIPS 3000, le traitement des instructions se passait comme suit:

- Extraction des instructions par l'UIB
- Exécutions des instructions
- Extraction des nouvelles instructions.

Lorsque l'exécution d'une instruction est terminée, l'UE reste inactif un court instant, pendant que l'UIB extrait l'instruction suivante. Pour remédier à ce temps d'attente, le *prétraitement* ou *traitement pipeline* à été introduit dans le MIPS R3000. Pendant que l'UE exécute les informations qui lui sont transmises, l'instruction suivante est chargée dans l'UIB. Les instructions qui suivront sont placées dans une file d'attente. Lorsque l'UE a fini de traiter une instruction l'UIB lui transmet instantanément l'instruction suivante, et charge la troisième instruction en vue de la transmettre à l'UE. De cette façon, l'UE est continuellement en activité. Dans la figure suivante nous pouvons observer un schéma plus détaillé de l'UE et l'UIB. Nous y retrouvons les éléments dont il a été question précédemment.

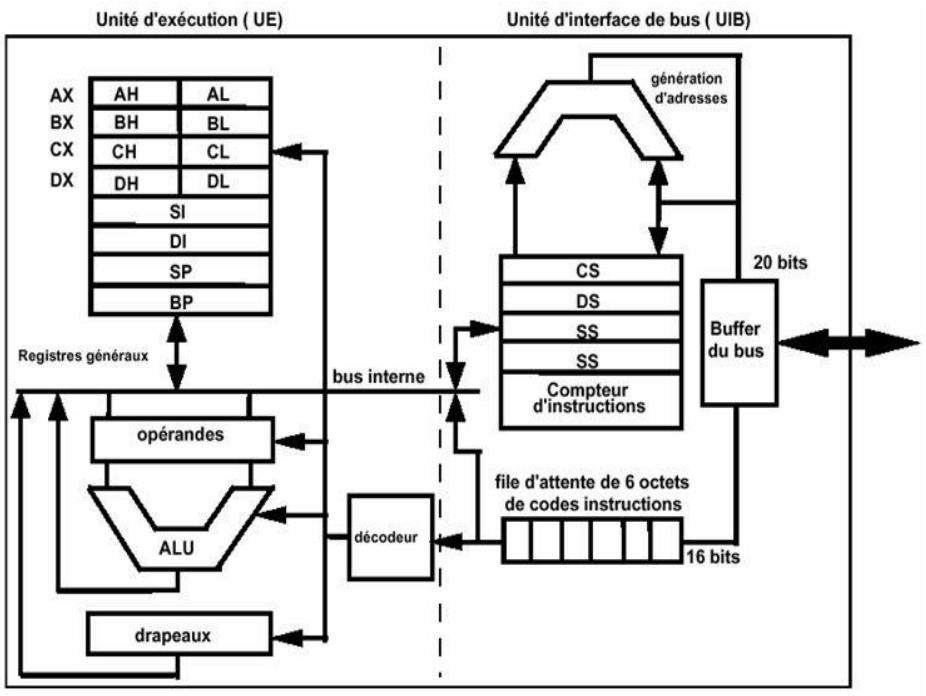


Schéma IV.2 : structure interne du MIPS R3000

Donc en conclusion on peut dire que le MIPS R3000 se compose essentiellement de deux unités la BIU qui fournit l'interface physique entre le microprocesseur et le monde extérieur, et l'EU qui comporte essentiellement l'UAL de 16 bits qui manipule les registre généraux de 16 bits aussi.

La file d'attente d'instructions contient des informations qui attendent d'être traitées par l'UE. La file d'attente est parfois appelée *capacité de traitement.* Le microprocesseur MIPS R3000 est capable de mémoriser jusqu'à six octets. Les microprocesseurs actuels sont bien entendu équipés d'une file d'attente plus rapide et plus large, c-à-d capable d'emmagasiner plus d'informations.

5. Jeu d'instructions

On peut diviser les instructions du microprocesseur MIPS R3000 en six groupes comme suit:

- Instructions de transfert de données;
- Instructions arithmétiques;
- Instructions de bits (logiques);
- Instructions de sauts de programme;
- Instructions de chaîne de caractères;
- Instructions de contrôle de processus;
- Instructions d'interruptions.

5.1. Les instructions de transfert de données

Ils sont divisés en quatre sous-groupes comme suit:

- Les instructions d'usage général ;
- Les instructions d'entrées-sorties ;
- Les instructions de transfert d'adresses;
- Les instructions d'indicateur.

5.1.1. Les instructions d'usage général

MOV: Elle permet de transférer les données (un octet ou un mot) d'un registre à un autre registre ou d'un registre à une case mémoire, sa syntaxe est comme suit: MOV destination, source.

Exemples:

```
MOV   AX, BX    ;  Transfert d'un registre de 16 bits vers un registre de 16 Bits
MOV   AH, CL    ;  Transfert d'un registre de 8 bits vers un registre de 8 bits
MOV   AX, Val1  ; Transfert du contenu d'une case mémoire 16 bits vers AX
MOV   Val2, AL  ; Transfert du contenu du AL vers une case mémoire  D'adresse Val2
```

PUSH: Elle permet d'empiler les registres du CPU sur le haut de la pile, sa syntaxe est comme suit: PUSH SOURCE.

POP: Elle permet de dépiler les registres du CPU sur le haut de la pile, sa syntaxe est comme suit: POP destination.

PUSHA: Cette instruction permet d'empiler la totalité des registres internes du microprocesseur sur la pile.

POPA: Cette instruction permet de dépiler la totalité des registres internes du microprocesseur sur la pile.

XCHG: Elle permet de commuter la source avec la destination comme suit:

XLAT: Cette instruction est utilisée pour convertir des données d'un code à un autre, en effet elle permet de placer dans l'accumulateur AL le contenu de la case mémoire adressée en adressage base+décalage (8 bits), la base étant le registre BX et le décalage étant AL lui même dans le segment DS, sa syntaxe est comme suit : XLAT tab_source.

5.1.2. Les instructions d'entrées-sorties

OUT: Elle permet de récupérer des données d'un port (donc de la périphérie) ou restituer des données à un port, dans les deux cas s'il s'agit d'envoyer ou de recevoir un octet on utilise l'accumulateur AL, s'il s'agit d'envoyer ou de recevoir un mot on utilise l'accumulateur AX, sa syntaxe est comme suit:
IN ACCUMULATEUR, DX
OUT DX, ACCUMULATEUR
DX : contient l'adresse du port.
ACCUMULATEUR : contient la donnée (à recevoir ou à emmètre).

5.1.3. Les instructions de transfert d'adresses

LEA (Load Effective Address): Elle transfert l'adresse offset (décalage) d'un opérande mémoire dans un registre de 16 bits (pointeur ou index). Cette commande a le même rôle que l'instruction MOV avec offset mais elle est plus puissante car on peut utiliser avec elle toute technique d'adressage.

Exemple:
LEA BX, TAB_VAL (c'est équivalent à MOV BX, offset TAB_VAL)
LDS: Cette instruction permet de charger le segment et l'offset d'une adresse

Exemple:
Au lieu de faire:
MOV BX, offset tab_val
MOV AX , Seg tab_val
MOV DS , AX

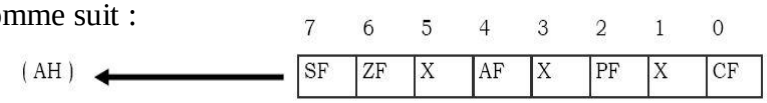
On remplace ces trois instructions par une seule:

LDS BX , tab_val ; elle charge automatiquement l'offset de tab_val dans le registre BX; et le segment dans le registre DS.

- Pour l'instruction LES : le segment est ES.

5.1.4. Les instructions d'indicateur

LAHF: Load AH from Flags : place l'octet de poids faible du registre d'état (FLAGS) dans le registre AH comme suit :



SAHF: Store AH into Flags: Place le contenu de AH dans l'octet de poids faible du registre d'état (FLAGS).

PUSHF: Permet d'empiler la totalité du registre d'état (FLAGS).

POPF: Permet de dépiler le registre d'état (FLAGS).

5.2. Instructions arithmétiques

Les instructions arithmétiques sont divisées en quatre sous-groupes:

- Addition;
- Soustraction;
- Multiplication;
- Division.

5.2.1. L'Addition

ADD (Addition): Elle permet d'additionner le contenu de la source (octet ou un mot) avec celui de la destination le résultat est mis dans la destination, sa syntaxe est comme suit: ADD Destination, source.

Exemples:

ADD AX, BX ; AX = AX + BX (addition sur 16 bits) ADD AL,BH

; AL = AL + BH (addition sur 8 bits)

ADD AL, [SI] ; AL = AL + le contenu de la case mémoire

; pointé par SI

ADD [DI], AL ; le contenu de la case mémoire pointé par DI

; Est additionnée avec AL, le résultat est mis

; dans la case mémoire pointé par DI.

ADC (Addition avec retenue): Elle permet d'additionner le contenu de la source (octet ou un mot) avec celui de la destination et la retenue (CF) le résultat est mis dans la destination, sa syntaxe est comme suit: ADC Destination, source.

Exemples:

ADC AX,BX ; AX = AX + BX + CF(addition sur 16 bits)

ADC AL,BH ; AL = AL + BH + CF(addition sur 8 bits)

ADC AL,[SI] ; AL = AL + le contenu de la case mémoire pointé par SI + CF

ADC [DI],AL ; le contenu de la case mémoire pointé par DI

INC (Incrémentation): Elle permet d'incrémenter le contenu de la destination, sa syntaxe est comme suit: INC Destination.

Exemples:

INC AX ; AX = AX + 1 (incrémentation sur 16 bits).

INC AL ; AL = AL +1 (incrémentation sur 8 bits).

INC [SI] ; [SI] = [SI] + 1 le contenu de la case mémoire pointé par SI sera incrémenter.

- On ne peut pas incrémenter une valeur immédiate.

5.2.2 Soustraction

SUB (Soustraction): Elle permet de soustraire la destination de la source (octet ou un mot) le résultat est mis dans la destination, sa syntaxe est comme suit: SUB Destination, source.

Exemples:

SUB AX,BX ; AX = AX - BX (Soustraction sur 16 bits)

SUB AL,BH ; AL = AL - BH (Soustraction sur 8 bits)

SUB AL,[SI] ; AL = AL - le contenu de la case mémoire pointé par SI

SUB [DI],AL ; le contenu de la case mémoire pointé par DI

; est soustraite de AL , le résultat est mis

; dans la case mémoire pointé par DI.

- On a les mêmes restrictions de l'instruction ADD.

SBB (Soustraction avec retenue): Elle permet de soustraire la destination de la source et la retenue (octet ou un mot) le résultat est mis dans la destination, sa syntaxe est comme suit : SBB Destination, source.

Exemples:

SBB AX,BX ; AX = AX-BX - CF (Soustraction sur 16 bits)

SBB AL,BH ; AL = AL - BH - CF(Soustraction sur 8 bits)

SBB AL,[SI] ; AL = AL - le contenu de la case mémoire

; pointé par SI - CF

SBB [DI],AL ; le contenu de la case mémoire pointé par DI

; est soustraite avec AL - CF, le résultat est

; mis dans la case mémoire pointé par DI.

- On a les mêmes restrictions de l'instruction ADD.

DEC (Décrémentation): Elle permet de décrémenter le contenu de la destination, sa syntaxe est comme suit: DEC Destination.

Exemples:

DEC AX ; AX = AX - 1 (décrémentation sur 16 bits).

DEC AL ; AL = AL -1 (décrémentation sur 8 bits).

DEC [SI] ; [SI] = [SI] - 1 le contenu de la case mémoire

; pointé par SI sera décrémenter.

- On ne peut pas décrémenter une valeur immédiate.

NEG (Négatif): Elle soustrait l'opérande destination (octet ou mot) de 0 le résultat est stocker dans la destination, donc avec cette opération on réalise le complément à deux d'un nombre, sa syntaxe est comme suit : NEG Destination.

Exemples:

NEC AX ; AX = 0 - AX

NEC AL ; AL = 0 - AL

NEC [SI] ; [SI] = 0 - [SI]

- Les indicateurs affectés par cette opération sont : AF, CF, OF, PF, SF, ZF.

CMP (Comparaison): Elle soustrait la source de la destination, qui peut être un octet ou un mot, le résultat n'est pas mis dans la destination , en effet cette instruction touche uniquement les indicateurs pour être tester avec une autre instruction ultérieure de saut conditionnel, sa syntaxe est comme suit: CMP Destination , Source.

Les indicateurs susceptibles d'être touché sont: AF, CF, OF, PF, SF, ZF.

DAS (DECIMAL Adjust for Substraction): Elle est identique à l'instruction AAA/DAA mais l'ajustement se fait en BCD pour la soustraction.

5.2.3 La multiplication:

MUL (Multiplication pour les nombres non signés): MUL effectue une multiplication non signée de l'opérande source avec l'accumulateur, sa syntaxe est comme suit: MUL Source.

- Si la source est un octet alors elle sera multipliée par l'accumulateur AL le résultat sur 16 bits sera stocké dans le registre AX.
- Si la source est un mot alors elle sera multipliée avec l'accumulateur AX le résultat de 32 bits sera stocké dans la paire des registres AX et DX.

IMUL (Multiplication pour les nombres signés): MUL effectue une multiplication signée de l'opérande source avec l'accumulateur, sa syntaxe est comme suit: IMUL Source.

- Si la source est un octet alors elle sera multipliée par l'accumulateur;
- AL le résultat sur 16 bits sera stocké dans le registre AX;
- Si la source est un mot alors elle sera multiplié avec l'accumulateur AX le résultat de 32 bits sera stocké dans la paire des registres AX et DX.
- Cette multiplication traite les données en tant que nombres signés.

AAM (ASCII Adjust for Multiplication): Comme AAA et AAS cette instruction va nous permettre de corriger le résultat d'une multiplication de deux nombres en BCD, pour corriger le résultat de l'instruction AAM divise AL par 10.

Exemple:

```
MOV AL , 6
MOV DL , 8
MUL DL
AAM
```

5.2.4 La division

DIV (Division des nombres non signés): Elle effectue une division non signée de l'accumulateur par l'opérande source, sa syntaxe est comme suit: DIV Source.

Exemples:

- Si l'opérande est un octet : alors on récupère le quotient dans le registre AL et le reste dans le registre AH;
- Si l'opérande est un mot : alors on récupère le quotient dans le registre AX et le reste dans le registre DX;

A/

```
MOV AH,00h
MOV AL,33H
MOV DL,25H
```

DIV DL ; Cela implique que AH= et AL =

B/

MOV AX,500H

MOV CX,200H

DIV CX ;Cela implique que AX= et AL =

IDIV (Division des nombres signés): Elle effectue une division signée de l'accumulateur par l'opérande source, sa syntaxe est comme suit : IDIV Source.

- Si l'opérande est un octet : alors on récupère le quotient dans le registre AL et le reste dans le registre AH;
- Si l'opérande est un mot : alors on récupère le quotient dans le registre AX et le reste dans le registre DX.

Exemples:

A/

MOV AH,00h

MOV AL,-33H

MOV DL,25H

IDIV DL ; Cela implique que AH= et AL=

B/

MOV AX,-500H

MOV CX,200H

IDIV CX ; Cela implique que AX= et AL=

AAD (ASCII Adjust for Division): Pour corriger le résultat elle va multiplier le contenu de AH par 10 et l'ajoute à celui de AL.

- Pour cette instruction il faut faire l'ajustement avant l'instruction de division.

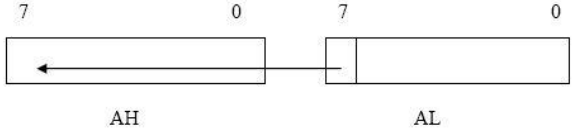
CBW (convert byte to word): Cette instruction permet de doubler la taille de l'opérande signé.

- CBW reproduit le bit 7 (bits de signe) de AL dans AH jusqu'à remplissage de ce dernier.

Exemple:

MOV AL, +96 ; AL=0110 0000

CBW ; AH=0000 0000 et AL=0110 0000



CWD (convert Word to Double): Cette instruction permet de doubler la taille de l'opérande signé.

- CWD reproduit le bit 15 (bits de signe) de AX dans DX jusqu'à remplissage de ce dernier.

Exemple:

```
MOV  AX, +260  ; AX=0000 0001 0000 0100
CWD            ; DX=0000H, AX=0104H
```

5.3. Les instructions logiques (de bits)

5.3.1. Les instructions logiques

NOT (Négation): Elle réalise la complémentation à 1 d'un nombre, sa syntaxe est comme suit: NOT Destination.

Exemple:

```
MOV AX, 500   ; AX = 0000 0101 0000 0000
NOT AX        ; AX = 1111 1010 1111 1111
```

AND (Et logique): Elle permet de faire un ET logique entre la destination et la source (octet ou un mot) le résultat est mis dans la destination, sa syntaxe est comme suit: AND Destination, source.

Exemples:

```
MOV AX , 503H     ; AX = 0000 0101 0000 0011
AND AX , 0201H    ;      0000 0101 0000 0011
                  ; AND  0000 0010 0000 0001
                  ;    = 0000 0000 0000 0001
AND AX,BX         ; AX = AX . BX (Et logique entre AX et BX)
AND  AL,BH        ; AL = AL . BH (ET logique sur 8 bits)
AND AL,[SI]       ; AL = AL AND le contenu de la case mémoire
                  ; pointé  par SI
AND [DI],AL       ; ET logique entre la case mémoire pointé par
                  ; DI et  AL , le résultat est mis dans la case
                  ; mémoire pointé par DI
```

OR (OU logique): Elle permet de faire un OU logique entre la destination et la source (octet ou un mot) le résultat est mis dans la destination, sa syntaxe est comme suit: OR Destination, source.

Exemples:

```
MOV AX , 503H     ; AX = 0000 0101 0000 0011
OR AX , 0201H     ;      0000 0101 0000 0011
                  ;  OR  0000 0010 0000 0001
                  ;  =    0000 0111 0000 0011
OR AX,BX          ; AX = AX + BX ( OU logique entre AX et BX )
```

```
OR  AL,BH          ; AL = AL + BH ( OU logique sur 8 bits )
OR AL,[SI]         ; AL = AL OU le contenu de la case mémoire
                   ; pointé  par SI
OR [DI],AL         ; OR logique entre la case mémoire pointé par
                   ; DI et  AL, le résultat est mis dans la case
                   ; mémoire pointé par DI
```

XOR (OU exclusif): Elle permet de faire un OU exclusif logique entre la destination et la source (octet ou un mot) le résultat est mis dans la destination, sa syntaxe est comme suit : XOR Destination, source.

Exemples:

```
MOV AX , 503H   ; AX = 0000 0101 0000 0011
XOR AX , 0201H  ;      0000 0101 0000 0011
                ;  XOR  0000 0010 0000 0001
                ;  =    0000 0011 0000 0010
XOR AX,BX       ; AX = AX + BX (OU exclusif  entre AX et BX )
XOR  AL,BH      ; AL = AL + BH ( OU exclusif sur 8 bits )
XOR AL,[SI]     ; AL = AL OU exclusif le contenu de la case
                ; Mémoire pointé  par SI
XOR [DI],AL     ; XOR logique entre la case mémoire pointé par
                ; DI et  AL, le résultat est mis dans la case
                ; mémoire pointé par DI
```

TEST: Elle permet de faire un ET logique entre la destination et la source (octet ou un mot) mais la destination ne sera pas touchée en effet cette instruction ne touche que les indicateurs, sa syntaxe est comme suit: TEST Destination, source.

Exemples:

```
MOV AX, 503H   ; AX = 0000 0101 0000 0011
TEST AX, 0201H ;  0000 0101 0000 0011
```

Elle va effectuer un ET logique entre le premier nombre et le second sans toucher les deux mais elle va affecter uniquement les indicateurs (Flags).

5.3.2 Les instructions de décalages

SHL: décalage logique à gauche.

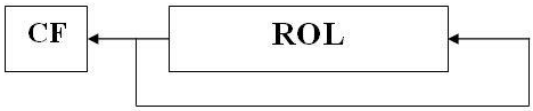
SHR : décalage logique à droite.

SAL : décalage arithmétique à gauche.

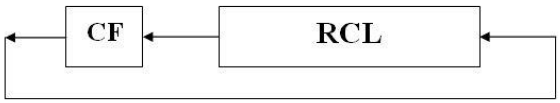
SAR : décalage arithmétique à gauche.

5.3.3 Les instructions de rotations

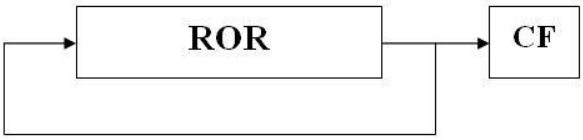
ROL : Rotation à gauche.



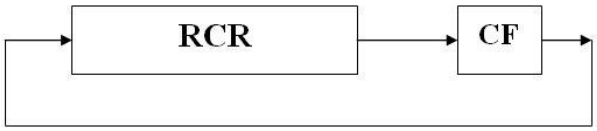
RCL : Rotation a travers la retenue à gauche.



ROR : Rotation à droite.



RCR : Rotation à travers la retenue à droite.



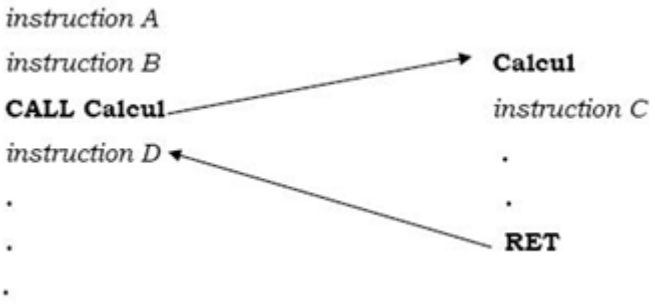
- Les instructions de rotations et de décalages arithmétiques préservent le bit de signe donc elles sont réservées aux nombres signés.

5.4. Instructions de sauts de programme: Elles permettent de faire des sauts dans l'exécution d'un programme (rupture de séquence).

- Ces instructions n'affectent pas les Flags. Dans cette catégorie on trouve toutes les instructions de branchement, de boucle et d'interruption après un branchement.

5.4.1. Branchement inconditionnel

CALL (notion de procédure): La notion de procédure en assembleur correspond à celle de fonction en langage C, ou de sous-programme dans d'autres langages.



La procédure est nommée calcul. Après l'instruction B, le processeur passe à l'instruction C de la procédure, puis continue jusqu'à rencontrer RET et revient à l'instruction D.

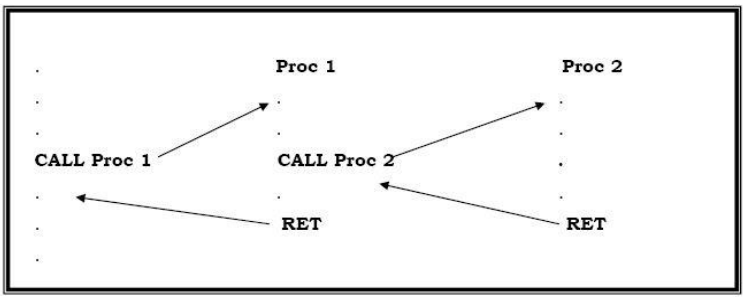
Une procédure est une suite d'instructions effectuant une action précise, qui sont regroupées par commodité et pour éviter d'avoir à les écrire à plusieurs reprises dans le programme.

Les procédures sont repérées par l'adresse de leur première instruction, à laquelle on associe une étiquette en assembleur.

L'exécution d'une procédure est déclenchée par un programme appelant. Une procédure peut elle-même appeler une autre procédure, et ainsi de suite.

CALL: L'appel d'une procédure est effectué par l'instruction CALL, CALL adresse debut procedure. L'adresse est sur 16 bits, la procédure est donc dans le même segment d'instructions. CALL est une nouvelle instruction de branchement inconditionnel. La fin d'une procédure est marquée par l'instruction RET:

RET: RET ne prend pas d'argument; le processeur passe à l'instruction placée immédiatement après le CALL. RET est aussi une instruction de branchement: le registre IP est modifié pour revenir à la valeur qu'il avait avant l'appel par CALL. Comment le processeur retrouve-t-il cette valeur? Le problème est compliqué par le fait que l'on peut avoir un nombre quelconque d'appels imbriqués, comme sur la figure suivante :



L'adresse de retour, utilisée par RET, est en fait sauvegardée sur la pile par l'instruction CALL. Lorsque le processeur exécute l'instruction RET, il dépile l'adresse sur la pile (comme POP), et la range dans IP.

L'instruction CALL effectue donc les opérations:

- Empiler la valeur de IP. A ce moment, IP pointe sur l'instruction qui suit le CALL.
- Placer dans IP l'adresse de la première instruction de la procédure (donnée en argument).

- Si la procédure appartient au même segment que le programme principal elle est dite de type NEAR sinon elle est dite de type FAR, la différence entre eux c'est que dans le premier cas le processeur doit empiler une seule valeur dans la pile c'est le registre IP mais dans le deuxième cas il faut empiler le registre IP ainsi que le registre segment CS et bien sur il les dépiler pendant le retour de la procédure.

- En général, une procédure effectue un traitement sur des données (paramètres) qui sont fournies par le programme appelant, et produit un résultat qui est transmis à ce programme.

Exemple:

Avec passage par registre; on va écrire une procédure (SOMME) qui calcule la somme de 2 nombres naturels de 16 bits.

Convenons que les entiers sont passés par les registres AX et BX, et que le résultat sera placé dans le registre AX.

La procédure s'écrit alors très simplement : SOMME PROC NEAR

```
ADD AX, BX      ; AX <- AX + BX
RET SOMME ENDP
```

et son appel, par exemple pour ajouter 6 à la variable Truc :

```
MOV AX, 6
MOV BX, Truc
CALL SOMME
MOV Truc, AX
```

Exemple:

Avec passage par la pile; cette technique met en œuvre un nouveau registre, BP (Base Pointer), qui permet de lire des valeurs sur la pile sans les dépiler ni modifier SP. Le registre BP permet un mode d'adressage indirect spécial, de la forme:

```
MOV AX, [BP+6]
```

Cette instruction charge le contenu du mot mémoire d'adresse BP+6 dans

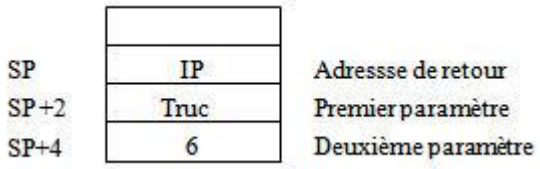
AX. Ainsi, on lira le sommet de la pile avec :

```
MOV BP, SP          ; BP pointe sur le sommet
MOV AX, [BP]        ; lit sans dépiler et le mot suivant avec:
MOV AX, [BP+2]      ; 2 car 2 octets par mot de pile.
```

L'appel de la procédure (SOMME2) avec passage par la pile est:

```
PUSH 6
PUSH Truc
CALL SOMME2
```

La procédure SOMME2 va lire la pile pour obtenir la valeur des paramètres. Pour cela, il faut bien comprendre quel est le contenu de la pile après le CALL:



Le sommet de la pile contient l'adresse de retour (ancienne valeur de IP empilée par CALL). Chaque élément de la pile occupe deux octets. La procédure SOMME2 s'écrit donc:

```
SOMME2 PROC near        ; AX <- arg1 + arg2
MOV BP, SP              ; adresse sommet pile
MOV AX, [BP+2]          ; charge argument 1
ADD AX, [BP+4]          ; ajoute argument 2
RET
SOMME2 ENDP
```

La solution avec passage par la pile parait plus lourde sur cet exemple simple. Cependant, elle est beaucoup plus souple dans le cas général que le passage par registre. Il est très facile par exemple d'ajouter deux paramètres supplémentaires sur la pile. Une procédure bien écrite modifie le moins de registres possible. En général, l'accumulateur est utilisé pour transmettre le résultat et est donc modifié. Les autres registres utilisés par la procédure seront normalement sauvegardés sur la pile. Voici une autre version de SOMME2 qui ne modifie pas la valeur contenue par BP avant l'appel:

```
SOMME2 PROC near        ; AX <- arg1 + arg2
PUSH BP                 ; sauvegarde BP
MOV BP, SP              ; adresse sommet pile
MOV AX, [BP+4]          ; charge argument 1
ADD AX, [BP+6]          ; ajoute argument 2
POP BP                  ; restaure ancien BP
RET
SOMME2 ENDP
```

Noter que les index des arguments (BP+4 et BP+6) sont modifiés car on a ajouté une valeur au sommet de la pile.

JMP (Saut inconditionnel):

Si le JMP est de type NEAR alors IP = IP + Déplacement;

Si le JMP est de type FAR alors CS et IP sont remplacés par les nouvelles valeurs obtenues à partir de l'instruction.

JMP transfert, sans condition, la commande à l'emplacement de destination. L'opérande Cible peut être obtenu à partir de l'instruction elle- même (JMP direct) ou à partir de la mémoire ou à partir d'un registre indiqué par l'instruction.

5.4.2. saut conditionnel

JC (Si retenue)

Si CF=1 alors IP = IP + déplacement

JE/JZ (Si égal/Si zéro)

Si ZF=1 alors IP = IP + déplacement

JNC (Si pas de retenue)

Si CF=0 alors IP = IP + déplacement

JNE/JNZ (Si non égal/Non zéro)

Si ZF=0 alors IP = IP + déplacement

JNO (Si pas de débordement)

Si OF=0 alors IP = IP + déplacement

JNP/JPO (Si pas de parité/Si parité impaire)

Si PF=0 alors IP = IP + déplacement

JNS (Si pas de signe)

Si SF=0 alors IP = IP + déplacement

JO (Si débordement)

Si OF=0 alors IP = IP + déplacement

JP/JPE (Si parité/Si parité paire)

Si PF=1 alors IP = IP + déplacement

JS (Si signe négatif)

Si SF=1 alors IP = IP + déplacement

5.4.3. Les instructions de boucle

LOOP (boucle): Elle décrémente le contenu de CX de 1.

Si CX est différente de zéro alors IP = IP + déplacement

Si CX = 0 l'instruction suivante est exécutée.

LOOPE/LOOPZ: Le registre CX est décrémenter de 1 automatiquement.

Si CX est différent de zéro et ZF=1 alors IP = IP + déplacement.

LOOPNE/LOOPNZ: Le registre CX est décrémenter de 1 automatiquement.

Si CX est différent de zéro et ZF=0 alors IP = IP + déplacement

5.5. Les instructions de chaînes de caractères

5.5.1. Les préfixes de répétitions

REP: Ces instructions sont utilisées avec les instructions de chaînes de caractères pour assurer la répétition de l'instruction si on veut appliquer l'instruction sur un ensemble d'informations. REP décrément automatiquement CX est test est ce qu'il est égal à zéro ou non. Si CX = 0 REP s'arrêt.

REPE/REPZ: C'est la même chose que REP c'est-à-dire elle décrément automatiquement le registre CX mais elle peut sortir de la boucle si ZF <> 0.

REPNE/REPNZ: C'est la même chose que REP c'est-à-dire elle décrément automatiquement le registre CX mais elle peut sortir de la boucle si ZF=0.

5.5.2. Les instructions MOVE-STRING

Elle déplace un élément du segment de données pointé par DS. Si l'élément à transférer est un octet on utilise: MOVB Si l'élément à transférer est un Mot on utilise: MOVW, mais dans les deux cas on n'utilise que d'opérande. SI et DI sont ensuite incrémentés de 1 (si DF=0) ou décrémentés de 1 (si DF=1) d'une manière automatique.

Exemple:

```
Donnee SEGMENT
Mess_Sour db ‘bonjour iset de nabeul'     ; message source
Donnee ENDS
Extra SEGMENT
Mes_Des db 22 dup (0)                     ; message destination
Extra ENDS
Code SEGMENT
Assume CS : Code, DS : Donnee, ES : extra
PROG PROC
MOV AX,Donnee MOV DS, AX MOV AX,Extra MOV ES, AX
LEA SI, Mess_Sour                 ; pointé le message source
LEA DI, Mess_Des                  ; pointé le message destination
MOV CX, 22                        ; nombre de caractère à transfèrer
CLD                               ; incrémentation automatique du SI et DI
```

```
REP MOVSB                ; transfert avec le préfixe REP MOV AX, 4C00H
                         ; Retour au DOS
INT 21H
PROG ENDP
CODE ENDS
END PROG
```

5.5.3. Les instructions COMPARE-STRING

Comparaison de chaîne: elle soustrait l'octet ou mot de destination (pointé par DI) de l'octet ou mot source (pointé par SI). CMPS affecte les indicateurs mais ne change pas les opérandes.

Si CMPS est utilisé avec le préfixe de répétition REPE/REPZ, elle est interprétée comme « comparer tant que la chaîne n'est pas finie (CX <>0) et que les éléments à comparer ne sont pas égaux (ZF=1).

Si CMPS est utilisé avec le préfixe de répétition REPNE/REPNZ, elle est interprétée comme « comparer tant que la chaîne n'est pas finie (CX <>0) et que les éléments à comparer ne sont pas égaux (ZF=0).

- On ne peut pas utilisé le préfixe REP avec l'instruction CMPS car cela revient à comparer uniquement les deux derniers éléments des deux chaînes.

Exemple:

```
Donnee SEGMENT
Mess_1 db ‘bonjour iset de nabeul' Mess_2 db ‘bonsoir iset de nabeul'
Donnee ENDS
Code SEGMENT
Assume CS : Code, DS : Donnee
PROG PROC
MOV AX,Donnee
MOV DS, AX
LEA SI, Mess_Sour  ; pointé le message source
LEA DI, Mess_Des   ; pointé le message destination
MO V CX, 22        ; nombre de caractère àcomparer
CLD                ; incrémentation automatique du SI et DI
REP CMPSB          ; transfert avec le préfixe
REP JCXZ  Tito     ; les deux chaînes sont identiques
MOV al, [SI-1]     ; on met dans AL le caractère
                   ; Différent
Tito : MOV AX, 4C00H  ; Retour au DOS INT 21H
```

```
PROG ENDP
CODE ENDS
END PROG
```

5.5.4. Les instructions SCAN STRING

Syntaxe :

```
SCAS chaine_destination
SCASB
SCASW
```

SCAS soustrait l'élément de la chaîne de destination (octet ou mot) adressé par DI dans le segment extra du contenu de AL (un octet) ou de AX (un mot) et agit sur les indicateurs. Ni la chaîne destination ni l'accumulateur ne change de valeur.

Exemple:

recherche de la lettre ‘a' dans une chaîne. EXTRA SEGMENT

```
Mess_Des db ‘bonjour iset de nabeul' EXTRA ENDS
Code SEGMENT
Assume CS : Code, ES : EXTRA
PROG PROC MOV AX, EXTRA MOV ES, AX
LEA DI, Mess_Des          ; pointé le message destination
MOV CX, 22                ; nombre de caractère à comparer
CLD                       ; incrémentation automatique du DI
MOV AL,'a'
REPNZ SCASB               ; transfert avec le préfixe REP JCXZ
Tito  ; les deux chaînes sont identiques
Call oui_le_a_existe      ; on met dans AL le caractère ; Différent
Tito : MOV AX, 4C00H      ; Retour au DOS
INT 21H
PROG ENDP
CODE ENDS
END PROG
```

5.5.5. Les instructions LOAD STRING (LODS) et STORE STRING (STOS)

LODS: Syntaxe :

```
LODS chaine_source
LODSB
LODSW
```

LODS transfert l'élément de chaîne (octet ou mot) adressé par SI au registre AL ou AX et remet à jour SI pour qu'il pointe vers l'élément suivant de la chaîne.

STOS: Syntaxe :

STOS chaine_destination

STOSB

STOSW

STOS transfert un octet ou un mot du registre AL ou AX vers l'élément de chaîne adressé par DI et modifie DI pour qu'il pointe vers l'emplacement suivant de la chaîne.

Exemple:

Transfert d'une chaîne source vers une chaîne destination en utilisant LODS et STOS.

```
Donnee SEGMENT
Mess_Sour db ‘bonjour iset de nabeul'     ; message source
Donnee ENDS
Extra SEGMENT
Mes_Des db 22 dup (0)                     ; message destination
Extra ENDS
Code SEGMENT
Assume CS : Code, DS : Donnee, ES : extra
PROG PROC
MOV AX,Donnee
MOV DS, AX
MOV AX,Extra
MOV ES, AX
LEA SI, Mess_Sour                 ; pointé le message source
LEA DI, Mess_Des                  ; pointé le message destination
MOV CX, 22                        ; nombre de caractère à transfèrer
CLD                               ; incrémentation automatique du SI et DI
DEBUT :    LODSB                  ; transfert avec le préfixe REP STOSB
LOOP DEBUT
MOV AX, 4C00H                     ; Retour au DOS INT 21H
PROG ENDP
CODE ENDS
END PROG
```

5.6. Les instructions de commande du processeur

Ces instructions agissent sur le processeur et ses indicateurs (Flags).

5.6.1. Indicateurs

STD: Met CF à 1; les registres d'indexation SI et/ou DI sont alors automatiquement décrémenter par les instructions de chaîne de caractère.

STI: Met IF à 1, permettant ainsi au CPU de reconnaître des demandes d'interruption masquables apparaissant sur la ligne d'entrée INTR.

5.6.2. Synchronisation

HALT: Maintient le processeur dans un état d'attente d'un RESET ou d'une interruption externe non masquable ou masquable (avec IF=1).

WAIT: Met le CPU en état d'attente tant que sa ligne de TEST n'est pas active. En effet toutes les cinq périodes d'horloge le CPU vérifie est ce que cette entrée est active ou non, si elle est active le processus exécute l'instruction suivante à WAIT.

ESC: L'instruction Escape fournit un mécanisme par lequel des coprocesseurs peuvent recevoir leurs instructions à partir de la suite d'instructions du 8086.

LOCK: Elle utilise dans les systèmes Multiprocesseur en effet elle permet le verrouillage du bus vis-à-vis des autres processeurs.

5.6.3 Sans opération

NOP (No operation): Le CPU ne fait rien on peut s'en servir pour créer des temporisations.

Exemple:

```
Tempo : MOV CX, 7FFFH          ; Effectuer une temporisation
Temp1: PUSH CX                 ; avec deux boucles imbriqués
MOV CX,7FFFH
Temp2: NOP NOP NOP NOP
LOOP Temp2
POP CX
LOOP Temp1
RET
```

1. Les interruptions spéciales

1.1. Introduction

Un système à base de microprocesseur peut comporter plusieurs périphériques tels que: un écran, une souris, une imprimante, un disque dure, un convertisseur numérique analogique, un CAN etc .. Pour dialoguer avec ces périphériques le microprocesseur a trois façons de communiquer avec ces derniers:

- En questionnant de façon continue le périphérique pour vérifier que des données peuvent être lues ou écrites (Polling).
- En l'interrompant lorsqu'un périphérique est prêt à lire ou écrire des données (interruption)
- En établissant une communication directe entre deux périphériques (DMA: Direct memory acces).

1.2. Le polling

Bien que le polling soit une façon simple d'accéder à un périphérique, le débit des données dans un périphérique est parfois beaucoup plus lent que la vitesse de fonctionnement d'un processeur, et le polling peut donc être très inefficace. On lui préfère en général la méthode par interruptions.

1.3. Le DMA

En cas de transfert de données de grande quantité entre deux périphériques, il peut être plus efficace de laisser communiquer entre eux les deux périphériques plutôt que de solliciter le processeur. Ce type de communication avec les entrées/sorties s'appelle gestion de transfert par DMA, Le processeur indique au contrôleur de DMA quels sont les périphériques qui doivent communiquer, le nombre et éventuellement l'adresse des données à transférer, puis il initie le transfert. Le processeur n'est pas sollicité durant le transfert, et une interruption signale au processeur la fin du transfert.

1.4. L'interruption

Il arrive fréquemment qu'un périphérique ou un programme nécessite une intervention du microprocesseur à cet effet il est possible de l'interrompre quelques instants, celui-ci va effectuer l'instruction demandée, est exécuter, une fois qu'elle rend le travail initial. Ce mode d'échange est principalement basé sur l'initiative du périphérique. Grâce à ce mécanisme d'interruption, le dispositif périphérique prend l'initiative de l'échange.

Une interruption est un évènement qui provoque l'arrêt du programme en cours et provoque le branchement du microprocesseur à un sous-programme particulier dit de "traitement de l'interruption".

Lorsqu'un périphérique reçoit des données pour le processeur, il envoie au processeur un signal d'interruption. Si celui-ci peut être interrompu (à condition qu'il ne soit pas en train de communiquer avec un périphérique de plus haute priorité), il stoppe la tâche en cours, sauve son état (ces registres) en mémoire (la zone pile) et appelle la routine correspondant au numéro d'interruption.

Le fonctionnement des périphériques se fait en général d'une manière asynchrone, donc plusieurs interruptions peuvent être déclenchées en même temps, il existe souvent un contrôleur d'interruptions destiné à gérer les conflits entre interruptions.

2. Interruption matérielle

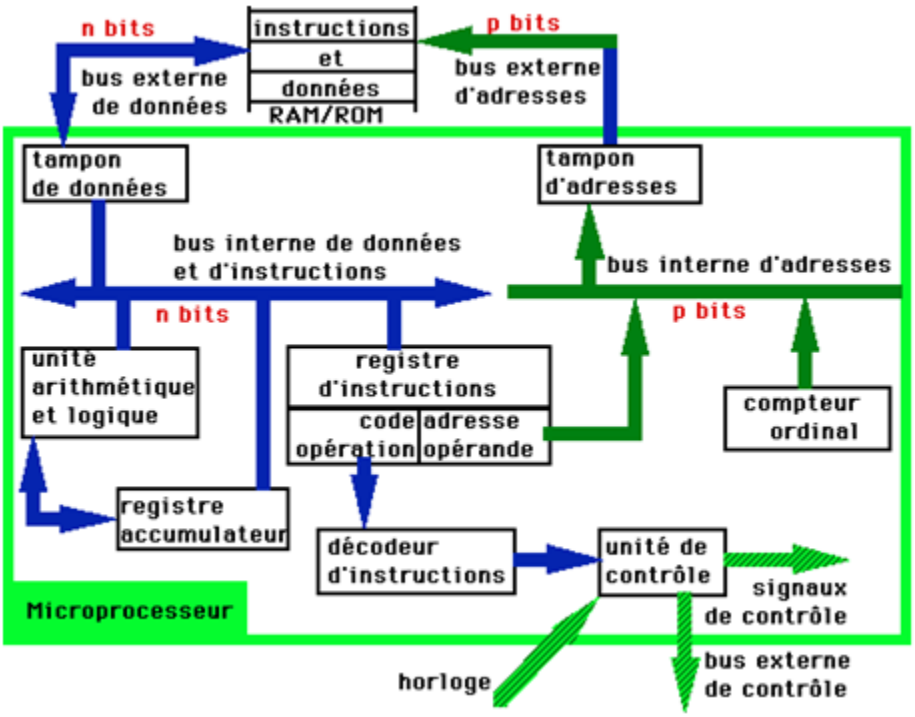
2.1. Introduction

Une interruption est signalée au processeur par un signal électrique sur une borne spéciale. Lors de la réception de ce signal, le processeur (traite) l'interruption dès la fin de l'instruction qu'il était en train d'exécuter. Le traitement de l'interruption consiste soit:

- à l'ignorer et passer normalement à l'instruction suivante: c'est possible uniquement pour certaines interruptions, nommées interruptions masquables. Il est en effet parfois nécessaire de pouvoir ignorer les interruptions pendant un certains temps, pour effectuer des traitements très urgents par exemple. Lorsque le traitement est terminé, le processeur démasque les interruptions et les prend alors en compte.
- à exécuter un *traitant d'interruption* (interrupt handler). Un traitant d'interruption est un programme qui est appelé automatiquement lorsqu'une interruption survient. L'adresse de début du traitant est donnée par la table des *vecteurs d'interruptions (voir paragraphe suivant)*. Lorsque le (programme d'interruption) traitant a effectué son travail, il exécute l'instruction spéciale IRET qui permet de reprendre l'exécution à l'endroit où elle avait été interrompue.

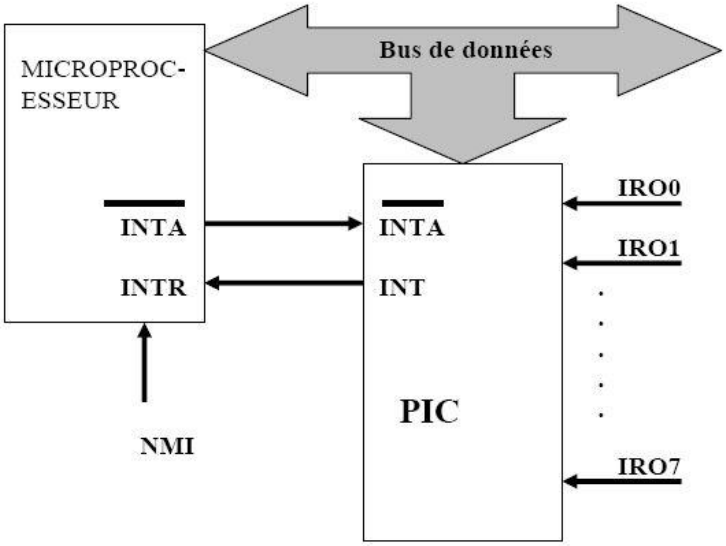
2.2. Cas du processeur MIPS R3000

Le microprocesseur MIPS R3000 possède trois lignes principales d'interruption: INTR, NMI, et RESET. Ces trois entrées permettent l'arrivée d'une demande (interruption externe) extérieur.



✓ ***Contrôleur d'interruptions dans un PC***

L'ordinateur est relié à plusieurs périphériques, mais nous venons de voir qu'il n'y avait qu'un seul signal de demande d'interruption, INTR. Le *contrôleur d'interruptions* est un circuit spécial, extérieur au processeur, dont le rôle est de distribuer et de mettre en attente les demandes d'interruptions provenant des différents périphériques.



Le contrôleur d'interruptions (PIC: Programmable Interruption Contrôler)

La figure précédente indique les connexions entre le microprocesseur et le contrôleur d'interruptions. Le contrôleur est relié aux interfaces gérant les périphériques par les bornes IRQ (*InteRrupt reQuest*). Il gère les demandes d'interruption envoyées par les périphériques, de façon à les envoyer une par une au processeur (via INTR). Il est possible de programmer le contrôleur pour affecter des priorités différentes à chaque périphérique. Avant d'envoyer l'interruption suivante, le contrôleur attend d'avoir reçu le signal INTA, indiquant que le processeur a bien traité l'interruption en cours. Une interruption est dite non masquable signifie qu'elle doit toujours être reconnue par le microprocesseur dés que le signal électrique a été déclenché.

✓ **Non masquable interruption (NMI)**

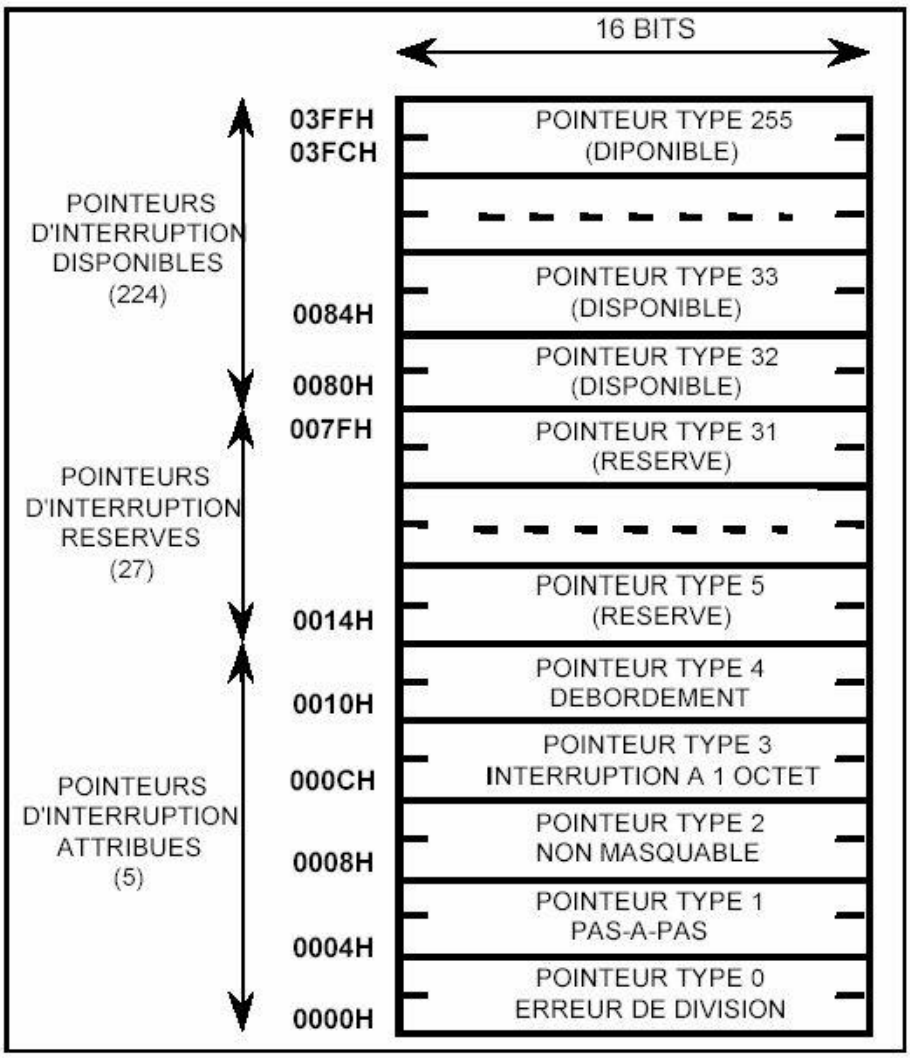
La reconnaissance de l'interruption NMI est généralement à la fin du cycle de l'instruction en cours, or les instructions du MIPS R3000 peuvent exiger un temps d'exécution plus ou moins long selon le nombre de cycles machines (comme l'instruction de multiplication) donc le temps de réponse va dépendre de l'instruction en cours. Ce temps est appelé temps de prise en charge de l'interruption.

Pendant la demande d'une interruption NMI tous les indicateurs sont empilés pour sauvegarder leurs valeurs en cours, puis l'indicateur IF (indicateur d'autorisation d'interruption: Interruption Flag) est mis à zéro (ce qui interdit toute arrivée de demande d'interruption sur la ligne INTR. Ensuite l'indicateur TF est mis à zéro (Trap Flag : indicateur pas à pas), puis le microprocesseur empile les registres CS et IP dans la pile.

Enfin le microprocesseur charge le registre IP avec la valeur à 16 bits qui se trouve à l'adresse mémoire 00008H dans le vecteur d'interruption (voir figure suivante), et le registre CS sera ensuite chargé avec la valeur de 16 bits située à l'adresse 0000AH, donc le microprocesseur maintenant pointe sur le programme d'interruption qu'il va exécuter puis revenir à son programme principal.

✓ **Le vecteur d'interruption**

Nommer aussi table d'interruption, il contient toujours les adresses des programmes d'interruptions aux quels le microprocesseur doit se brancher. Le microprocesseur se branche toujours à l'adresse 4*n et 4*n+1 pour chercher le contenu sur 16 bits du CS Si on demande l'interruption INT n.



Déroulement d'une interruption externe masquable (exemple dans un PC):

Reprenons les différents évènements liés à la réception d'une interruption masquable:

1. Un signal INT est émis par un périphérique (ou par l'interface gérant celui-ci).
2. Le contrôleur d'interruptions reçoit ce signal sur une de ses bornes IRQi. Dès que cela est possible (suivant les autres interruptions en attente de traitement), le contrôleur envoie un signal sur sa borne INT.
3. Le microprocesseur prend en compte le signal sur sa borne INTR après avoir achevé l'exécution de l'instruction en cours (ce qui peut prendre quelques cycles d'horloge). Si l'indicateur IF=0, le signal est ignoré, sinon, la demande d'interruption est acceptée.
4. Si la demande est acceptée, le microprocesseur met sa sortie INTA au niveau 0 pendant 2 cycles d'horloge, pour indiquer au contrôleur qu'il prend en compte sa demande.
5. En réponse, le contrôleur d'interruption place le numéro de l'interruption associé à la borne IRQi sur le bus de données.
6. Le processeur lit le numéro de l'interruption sur le bus de données et l'utilise pour trouver le vecteur d'interruption. Ensuite, tout se passe comme pour un appel système c'est à dire que le processeur :

(a) sauvegarde les indicateurs du registre d'état sur la pile;

(b) met l'indicateur IF à 0 (masque les interruptions suivantes);

(c) sauvegarde CS et IP sur la pile;

(d) cherche dans la table des vecteurs d'interruptions l'adresse du traitant d'interruption, qu'il charge dans CS:IP.

7. La procédure traitant l'interruption se déroule. Pendant ce temps, les interruptions sont masquées (IF=0). Si le traitement est long, on peut dans certains cas ré-autoriser les interruptions avec l'instruction STI.

8. La procédure se termine par l'instruction IRET, qui restaure CS, IP et les indicateurs à partir de la pile, ce qui permet de reprendre le programme qui avait été interrompu.

3. Interruption logiciel

Les interruptions logicielles sont semblables aux interruptions matérielles. L'unique différence réside dans le fait que les interruptions logicielles sont émises par des programmes. Les cinq premières interruptions sont définies par Intel. Les autres interruptions sont définies par le DOS et le BIOS. Ces interruptions ont une fonction définie, par exemple la lecture et l'écriture sur le disque, l'écriture des données à l'écran, etc. Contrairement à l'entrée INTR du microprocesseur, l'interruption logicielle ne peut être ni invalidé ni masquée.

Exemple d'interruption software:

Fonctions BIOS:

Int 1Ah, Fct 02h: Lecture de l'horloge temps réel BIOS (> AT):

Cette fonction permet de lire l'heure de l'horloge temps réel alimentée par batterie. Comme ce type d'horloge n'existe que sur les AT, seul ce modèle de PC soutient cette fonction.

Entrée:

AH = 02h

Sortie:

Flag Carry =0 : Tout va bien, dans ce cas

CH = Heures

CL = Minutes

DH = Secondes

Flag Carry =1 : La batterie de l'horloge est déchargée

- ✓ Toutes les indications sont fournies en format BCD.
- ✓ Le contenu des registres BX, SI, DI, BP et des registres de segment n'est pas modifié par cette fonction. Le contenu de tous les autres registres peut avoir été modifié.

Fonction DOS:

Int 21h, Fct 06h: Entrée/sortie directe de caractère:

Cette fonction permet de sortir des caractères sur le périphérique de sortie standard ou de les lire sur le périphérique d'entrée standard. Le caractère reçu ou écrit chaque fois n'est pas examiné par le système d'exploitation. Rien de particulier ne se produit donc lorsqu'un caractère Ctrl-C apparaît. Comme l'entrée et la sortie standard peuvent être redirigées sur d'autres périphériques ou vers un fichier, les caractères sortis ne doivent pas nécessairement apparaître sur l'écran ni les caractères lus provenir obligatoirement du clavier. Toutefois, lorsque l'accès se fait sur un fichier, il est impossible pour le programme d'appel de détecter si tous les caractères de ce fichier ont déjà été lus ou bien si le support (disquette, disque dur) sur lequel figure ce fichier est déjà plein. Pour l'entrée d'un caractère, la fonction n'attend pas qu'un caractère soit prêt mais revient immédiatement au programme d'appel dans tous les cas.

Entrée:

AH = 06h

DL = 0 - 254 : Sortir ce caractère

DL = 255 : Lire un caractère Sortie :

Pour la sortie de caractères : aucune

Pour l'entrée de caractères :

Flag Zéro =1 : aucun caractère n'est prêt

Flag Zéro =0 : Le caractère entré figure dans le registre AL.

Lorsque des codes étendus sont lus, le code 0 est tout d'abord renvoyé dans le registre AL. La fonction doit être appelée à nouveau pour lire le code étendu lui-même.

- Le caractère de code ASCII 255 ne peut être sorti à l'aide de cette fonction puisqu'il est interprété comme d'entrée d'un caractère.

- Le contenu des registres AH, BX, CX, DX, SI, DI, BP, CS, DS, SS, ES et du registre de flags n'est pas modifié par cette fonction.

Exemple de programme: lecture et affichage d'un caractère surl'écran:

```
CODE SEGMENT ASSSUME CS :CODE PROG PROC
DEBUT :
MOV  AH,06H          ; préparer l'interruption de lecture d'un
MOV  DL,255          ; caractère
INT 21H
```

```
JZ DEBUT            ; si le bit ZF = 0 => aucune touche n'est appuyée
MOV DL,AL           ; afficher le caractère appuyer .
INT 21H
MOV AX,4C00H        ; retour au DOS
INT 21H PROG ENDP
CODE ENDS
END PROG
```

Int 21h, Fct 09h Sortie d'une chaîne de caractères

Cette fonction permet de sortir une chaîne de caractères sur le périphérique de sortie standard. Comme ce périphérique standard peut être redirigé sur un autre périphérique ou vers un fichier, il n'y a aucune garantie que la chaîne de caractères apparaisse sur l'écran. Si la sortie est redirigée sur un fichier, le programme d'appel n'a aucune possibilité de détecter si le support (disquette, disque dur) sur lequel figure le fichier est déjà plein, autrement dit s'il est encore possible d'écrire la chaîne de caractères dans le fichier.

Entrée:

AH = 09h

DS = Adresse de segment de la chaîne de caractères DX = Adresse d'offset de la chaîne de caractères.

Sortie: Aucune

Exemple: affichage d'un message sur l'écran:

```
DONNEE SEGEMENT
MESSAGE DB BONJOUR ISET DE NABEUL $',13,10
DONNEE ENDS
CODE SEGMENT
ASSUME   CS : CODE , DS : DONNEE
PROG PROC
MOV AX , DONNEE MOV DS , AX
LEA DX , MESSAGE   ; Pointé le message par DX MOV AH , 09H
INT 21H            ; afficher le message
MOV AX,4C00H       ; Retour au DOS
INT 21H
PROG ENDP CODE ENDS END PROG
```

La chaîne de caractères doit être stockée dans la mémoire sous forme d'une séquence d'octets correspondant aux codes ASCII des caractères composant la chaîne. La fin de la chaîne de caractères doit être signalée au DOS à l'aide d'un caractère "$" (code ASCII 36).

- Si la chaîne de caractères contient des codes de commande comme Backspace, Carriage Return ou Line Feed, ceux-ci seront traités comme tels.
- Seul le contenu du registre AL est modifié par l'appel de cette fonction.

📖 L'appel de cette fonction agit sur toute une série de paramètres de la souris qui peuvent être fixés à l'aide des différentes fonctions de la souris:

- Le curseur de la souris est placé au centre de l'écran puis caché. Il apparaîtra désormais en mode graphique sous forme d'une flèche, alors qu'il sera représenté en mode de texte sous forme d'une case de texte en inversion vidéo. Le curseur de la souris est systématiquement affiché dans la page 0 de l'écran, indépendamment du mode. La zone de déplacement sélectionnée est la totalité de l'écran.
- Les gestionnaires d'événements installés par un programme, ils sont désactivés.

Int 33h, Fct 01h Afficher le curseur de la souris sur l'écran

Après appel de cette fonction, le curseur de la souris devient visible sur l'écran et suit désormais les déplacements de la souris sur l'écran.

Entrée:

AX = 0001h

Sortie:

Aucune

📖 L'appel de cette fonction entraîne l'incrémentation d'un compteur interne, en fonction duquel le curseur de la souris est ou non affiché sur l'écran. Si ce compteur contient la valeur 0, le curseur de la souris sera affiché sur l'écran, alors que la valeur -1 a pour effet de le faire disparaître de l'écran. Lorsque la fonction 00h (Reset) est appelée, ce compteur est tout d'abord fixé sur -1, pour prendre ensuite la valeur 0 lors du premier appel de cette fonction, ce qui se traduit par la réapparition du curseur sur l'écran.

Int 33h, Fct 03h Lire la position de la souris et l'état des boutons de la souris:

L'appel de cette fonction fournit la position actuelle de la souris et l'état des différents boutons de la souris.

Entrée:

AX = 0003h

Sortie:

BX = Etat des boutons de la Souris (à partir de 1.0) Bit Signification

0 = bouton gauche de la souris appuyé

1 = bouton droit de la souris appuyé

2 = bouton central de la souris appuyé

3-15 = Aucune (0)

CX = Position horizontale de la Souris (à partir de 1.0) DX = Position verticale de la souris

Les ordonnées renvoyées dans les registres CX et DX ne se rapportent pas à l'écran physique mais aux positions en points écran dans l'écran virtuel de la souris.

📖 Si la souris ne possède que deux boutons, les informations fournies sur le bouton central sont dépourvues de signification.