Programme de cours Introduction à l'ordinateur

Chapitre 1 : Historique de l'ordinateur

Aujourd'hui, l'ordinateur s'accapare nos modes de travail, envahit nos maisons, s'intègre dans les objets les plus quotidiens ; et, il est à l'origine de nouveaux modes de sociabilité et de l'informatique. Pourtant, l'ordinateur lui-même demeure pour beaucoup une énigme.

Ce cours se veut en partie, une réponse à ceux qui se demandent quels sont les fondements de l'informatique. L'informatique dont il sera question ici est une discipline scientifique qui, en tant que telle, a ses propres questions, ses propres problèmes, et dispose pour les aborder d'outils et de méthodes spécifiques. De cette discipline, on abordera les fondements théoriques ainsi que quelques réalisations pratiques, mais on insistera plus sur les concepts que sur la technique.

I- Qu'est-ce qu'un ordinateur ?

A la demande d'IBM en 1955, l'ordinateur fut créé. Il se définit comme une instance matérielle, concrète, d'une machine de Turing universelle. Il est donc capable, dans la limite de ses capacités en espace mémoire et en vitesse de calcul, d'exécuter n'importe quel algorithme qu'on lui fournit sous forme de programme, sur n'importe quelle donnée discrète, qu'on lui fournit également. Il se distingue ainsi fondamentalement d'une simple machine à calculer par sa capacité à enchaîner plusieurs opérations en suivant des instructions.

Autrement, l'ordinateur est une machine électronique programmable capable de réaliser des calculs logiques sur des nombres binaires.

Un ordinateur se compose de (02) deux parties :

- **Une partie matérielle ou hardware** : Le fonctionnement d'un ordinateur est basé sur une architecture matérielle (processeur, support de stockage, interfaces utilisateurs, connexion, . . .) dont le fonctionnement est soumis aux lois de la physique.
- **Une partie logicielle ou software** : L'ordinateur est capable de remplir des tâches différentes selon les instructions qui lui sont adressées. Ces instructions, rédigées sous forme de programmes par les informaticiens, sont traitées en fin de course par le matériel de l'ordinateur.

- **Remarque** : La plupart du temps, l'informaticien n'a pas à interagir directement avec le matériel. Pour traiter avec les composants, tous les ordinateurs disposent d'une couche logicielle appelée système d'exploitation. Cette couche est en charge de faire la passerelle entre l'informaticien, ses outils, les programmes qu'il développe et, les composants et leur fonctionnement.

1- Notion de bit

Tout le monde a entendu dire que les ordinateurs « *ne fonctionnent qu'avec des 0 et des 1* ». Qu'est-ce que cela signifie exactement et où, dans l'ordinateur, sont-ils cachés? Pour le comprendre, il faut cette fois partir des composants matériels qui constituent un ordinateur et aborder quelques notions élémentaires de la théorie de l'information.

✓ Définition

L'unité de base de la théorie de l'information est le bit, contraction de binary digit, qui signifie en anglais nombre binaire. Un bit, par définition, est un composant quelconque ne pouvant se trouver que dans deux états possibles, exclusifs l'un de l'autre. Par convention, pour s'abstraire de toutes contingences matérielles, on appelle l'un des deux états possibles d'un tel composant 0, et l'autre 1.

✓ Tableau de conversion

Nom	Symbole	
kilooctet	ko	2^{10}=1 024 octets
mégaoctet	Mo	2^{20}=1 048 576 octets
gigaoctet	Go	2^{30}=1 073 741 824 octets
téraoctet	To	2^{40}= 1 099 511 627 776 octets
pétaoctet	Po	2^{50}=1 125 899 906 842 624 octets
exaoctet	Eo	2^{60}=1 152 921 504 606 846 976 octets
zettaoctet	Zo	2^{70}=1 180 591 620 717 411 303 424 octets

yottaoctet	Yo	2^{80}=1 208 925 819 614 629 174 706 octets

2- Principaux composants d'un ordinateur

La figure ci-dessous présente l'environnement d'un ordinateur :

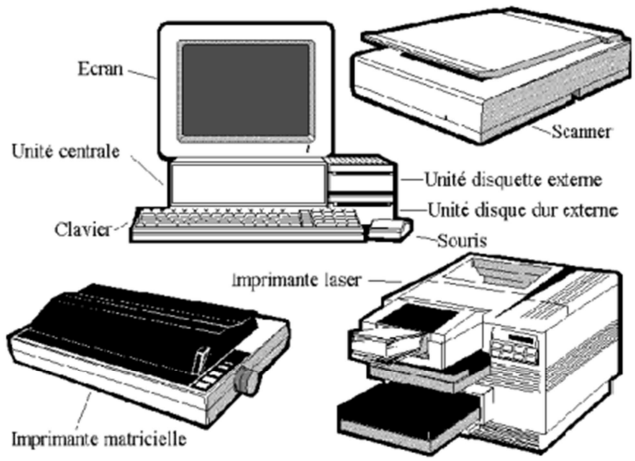


Fig1. Environnement d'un ordinateur

Cependant, il est important de distinguer un ordinateur de ses périphériques qui ne sont que des constituants connexes. Le cœur de l'ordinateur se compose d'une unité centrale, d'un microprocesseur, de mémoires de différents types, parmi lesquelles on distingue plusieurs types :

- ala mémoire ROM ou « mémoire morte» (Read Only Memory : mémoire à accès en lecture seule). Elle sert à stocker des informations permanentes (procédures de démarrage...) ;
- la mémoire RAM ou « mémoire vive» (Random Access Memory : mémoire à accès aléatoire). Cette mémoire est volatile, c'est-à-dire qu'elle ne conserve les données que tant que la machine est sous tension. La mémoire vive des meilleurs ordinateurs actuels atteint 1Giga-octet.
- 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, les clés USB... La capacité des disques durs actuels se compte en Giga-octets.

Les autres composants sont donc :

- 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, appareil photo numérique, caméscope numérique... Ces dispositifs peuvent tous être conçus comme des numériseurs puisqu'ils transforment un comportement en une suite de bits.
- des périphériques de sortie, c'est-à-dire permettant de visualiser ou de transmettre des données internes à l'extérieur : écran, imprimante, IPod, vidéoprojecteur... A l'inverse des numériseurs, ces dispositifs traduisent des suites de bits en information interprétable par les humains.

3- Architecture d'un ordinateur

L'architecture nous permet de décrire l'organisation interne de l'ordinateur. Si Turing peut être considère comme le père de l'informatique théorique, l'homme à l'origine de la conception des ordinateurs actuels est John Von Neumann.

En 1945, il a écrit les principes de la réalisation d'une machine universelle (ordinateur). Ces principes sont depuis connus sous le nom «d'architecture de Von Neumann», et sont ceux encore utilisés pour la conception des ordinateurs actuels. Le schéma général de l'architecture de Von Neumann est :

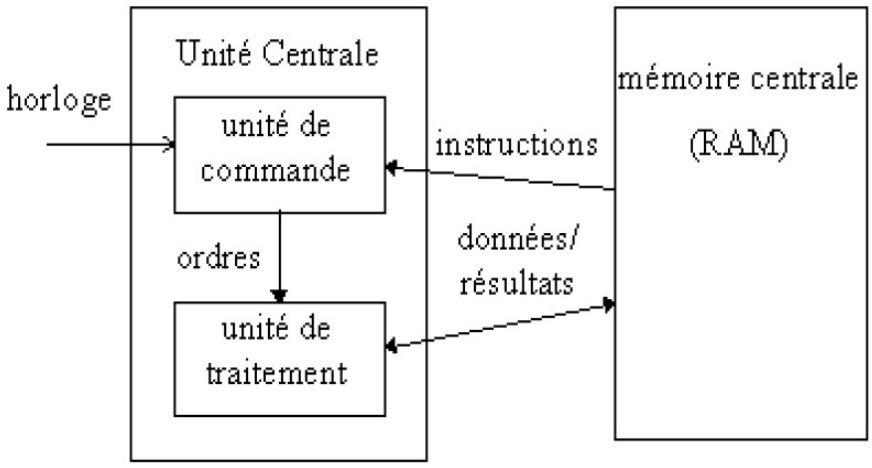


Fig2. Architecture de Von Neumann

Les deux innovations majeures introduites par Von Neumann par rapports aux calculateurs sont l'intégration :

- d'une «unité de commande» qui donne les ordres et synchronise les opérations ;
- d'une mémoire centrale interne permettant de stocker aussi bien des données que des programmes.

Pour bien comprendre comment fonctionne un ordinateur, il nous faut comprendre chaque composant de cette architecture.

✓ La mémoire centrale (RAM)

La mémoire vive d'un ordinateur est composée d'un ensemble de «*mots mémoires*», qui sont des suites de bits de taille fixe. Chaque mot mémoire est identifié par son «adresse» et celle-ci est indispensable pour référencer chaque mot mémoire et ainsi retrouver ce qui a été préalablement stocké dans l'un d'eux. L'adresse est simplement un code, donc une autre suite de bits.

Une particularité fondamentale de la mémoire centrale, dans l'architecture de Von Neumann, c'est qu'elle sert à stocker aussi bien des bits codant des données que des bits codant des traitements, des instructions (nous y reviendrons). Cette capacité à coder avec des 0 ou des 1 aussi bien des données que des traitements, est le fondement de l'informatique.

✓ L'unité de commande

Elle est composée de deux «registres» et joue un peu le rôle de la tête de lecture des machines. Un registre est simplement une petite unité de mémoire vive d'accès rapide.
Les deux registres de l'unité de commande sont :

- le compteur ordinal (ou CO) : ce registre sert à stocker en permanence l'adresse où se trouve en mémoire centrale interne l'instruction en cours d'exécution (instruction courante»).
- le registre d'instruction (ou RI) : il sert à stocker en permanence l'instruction en cours d'exécution (instruction courante»).

✓ L'horloge

L'horloge de l'unité centrale est un métronome électronique qui lance des signaux à intervalles de temps réguliers. Ces signaux d'horloge donnent la cadence à laquelle travaille l'ordinateur et permettent à l'ensemble des composants de l'unité centrale de se synchroniser. Plus les signaux sont rapprochés, plus l'ordinateur est rapide. La fréquence de l'horloge se compte en nombre de signaux par secondes, dont l'unité de mesure est le Hertz ou le Mega-Hertz MH (avec 1MH = 106 Hertz). Voici l'ordre de grandeur de la vitesse des ordinateurs ces dernières années :

- en 1985 : de 5 à 8 Mega-Hertz
- en 1990 : environ 20 Mega-Hertz
- en 1995 : environ 200 Mega-Hertz
- en 2001 sont sorties les premières puces cadencées à 1 Giga-Hertz (soit 109 Hetz)

✓ L'unité de traitement

L'unité de traitement est le composant qui exécute les calculs et est composé :

- de trois registres, servant respectivement à stocker les données (que nous notons donnée 1 et donnée 2) d'une opération arithmétique et son résultat : leur taille est celle d'un mot mémoire (2 octets dans notre exemple) ;
- de l'Unité Arithmétique et Logique (UAL) capable, quand on lui fournit le code d'une opération arithmétique à exécuter, de prendre les contenus des deux premiers registres (ceux contenant les données 1 et 2) et de remplir le troisième registre avec le résultat de cette opération. L'UAL est ainsi la «machine à calculer» de l'ordinateur. Elle est simplement constituée de circuits électroniques câblés une fois pour toute pour transformer des 1 en 0 ou des 0 en 1 (c'est-à-dire en fait pour actionner des interrupteurs faisant passer ou non du courant) de façon à ce que les bits du registre résultat correspondent bien au codage du résultat du calcul qui lui est demandé. Elle ne sait faire que des opérations élémentaires (dans notre exemple : simplement les 4 opérations arithmétiques de base).

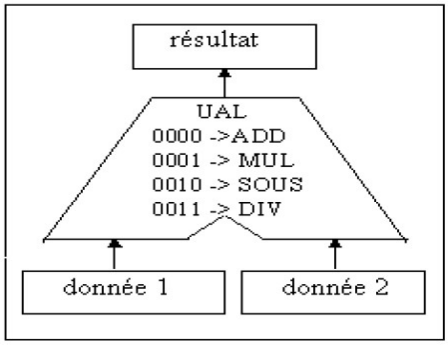
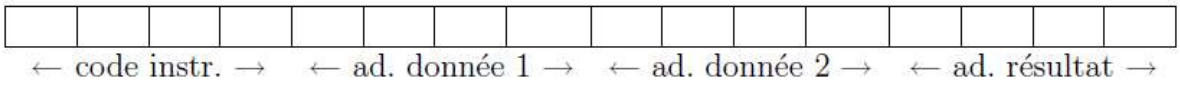


Fig3. Structure de l'unité de traitement



- la première, code instr., est le code de l'opération à effectuer. Dans notre exemple, nous nous contenterons des 4 opérations arithmétiques de base et nous nous fixons la convention suivante : 0000 code l'addition, 0001 la multiplication, 0010 la soustraction et 0011 la division. Il reste des codes disponibles pour d'autres opérations possibles.
- la deuxième et la troisième partie, notées ad. donnée 1 et ad. donnée 2 contiennent l'adresse en mémoire où se trouvent stockées respectivement la première et la deuxième donnée (dans cet ordre) sur lesquelles l'opération arithmétique doit être effectuée ;
- la quatrième, notée ad. résultat, est l'adresse en mémoire où doit être stocké le résultat de l'opération.

Ainsi, par exemple, «0011 1001 0011 0001» est l'instruction d'effectuer une division (code 0011) entre le nombre stocké dans le mot mémoire d'adresse 9 (code 1001) et celui stocké à l'adresse 3 (code 0011) et de stocker le résultat dans le mot mémoire d'adresse 1 (code 0001).

✓ Les bus

Les flèches reliant les composants entre eux sont des ensembles de fils permettant de transporter plusieurs bits en parallèle. On les appelle des bus.

Dans le schéma de la figure 2, figurent (03) trois bus dont les noms désignent le type de données qu'ils transportent. Le bus «***ordres***» sert à transmettre les demandes d'exécution d'opérations de l'unité de commande vers l'unité de traitement. Le bus «***instructions***» fait transiter les instructions élémentaires des mots mémoire vers le registre d'instruction de l'unité de

commande, et le bus «données/résultats» fait circuler (dans les deux sens) le contenu des mots mémoires entre la mémoire et les différents registres de l'unité de traitement.
Ces différents bus peuvent contenir un nombre de fils différent. Le nombre de fils du bus de données/résultats détermine la capacité du microprocesseur. Par exemple, un «microprocesseur 32 bits» (ordre de grandeur des puces actuelles) contient donc un bus données/résultats composé de 32 fils.

II- Systèmes à puce et mémoires

Dans un ordinateur, un système à puce désigné dans la littérature scientifique par le terme anglais « *system on a chip* » en abrégé SoC, est un système complet embarqué sur une seule puce, c'est-à-dire un circuit intégré, pouvant comprendre de la mémoire, un ou plusieurs microprocesseurs, des périphériques d'interface, ou tout autre composant nécessaire à la réalisation de la fonction attendue. On peut intégrer de la logique, de la mémoire (statique, dynamique, flash, ROM, PROM, EPROM, EEPROM), des dispositifs (capteurs) mécaniques, chimiques ou biologiques ou des circuits radio.

Nous nous intéressons ici aux mémoires, aux disques durs et aux microprocesseurs dans l'ordinateur.

1- Le microprocesseur

Un microprocesseur est un circuit intégré complexe. Il résulte de l'intégration sur une puce de fonctions logiques combinatoires (logiques et/ou arithmétique) et séquentielles (registres, compteur, etc…). Il est capable d'interpréter et d'exécuter les instructions d'un programme.

Le concept de microprocesseur a été créé par la Société Intel spécialisée dans la conception et la fabrication de puces mémoire depuis sa création en 1968. À la demande de deux de ses clients, fabricants de calculatrices et de terminaux, Intel étudia une unité de calcul implémentée sur une seule puce. Ceci donna naissance, en 1971, au premier microprocesseur, le 4004, qui était une unité de calcul 4 bits fonctionnant à 108 kHz. Il résultait de l'intégration d'environ 2300 transistors.

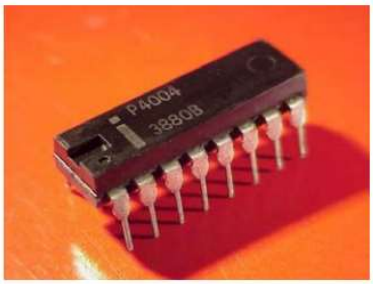


Fig4. Premier microprocesseur, le 4004

- **Remarque** : Les informations traitées par un microprocesseur sont de différents types (nombres, instructions, images, vidéo, etc…) mais elles sont toujours représentées sous un format binaire. Seul le codage changera suivant les différents types de données à traiter.

2- Le disque dur

Le disque dur est indispensable au fonctionnement de l'ordinateur et se présente sous la forme d'un boîtier en alliage d'aluminium d'environ 1.5 cm d'épaisseur, 10 cm de large et 15 cm de long. Il est généralement fixé dans un berceau placé sur la partie antérieure du châssis de l'ordinateur. Il est relié à la carte mère par un large ruban de connexion et reçoit son alimentation électrique par une fiche à fils. Ce boitier ne doit jamais être ouvert : qu'une simple poussière microscopique pénètre à l'intérieur et c'est la mort du disque dur. Dans le vocabulaire anglo-saxon, il est appelé Hard Disk Drive.

✓ Structure d'un disque dur

Un disque dur est constitué de plusieurs disques rigides (ou plateaux) en aluminium, verre ou céramique. Ces disques sont entraînés en rotation à une vitesse fixe. Les vitesses les plus courantes sont 5400 tr/mn, 7200 tr/mn, 10000 tr/mn et 15000 tr/mn. Des têtes de lecture situées de chaque côté d'un plateau, à quelques nanomètres de sa surface, viennent lire ou écrire les données. Toutes les têtes de tous les plateaux se déplacent en même temps.

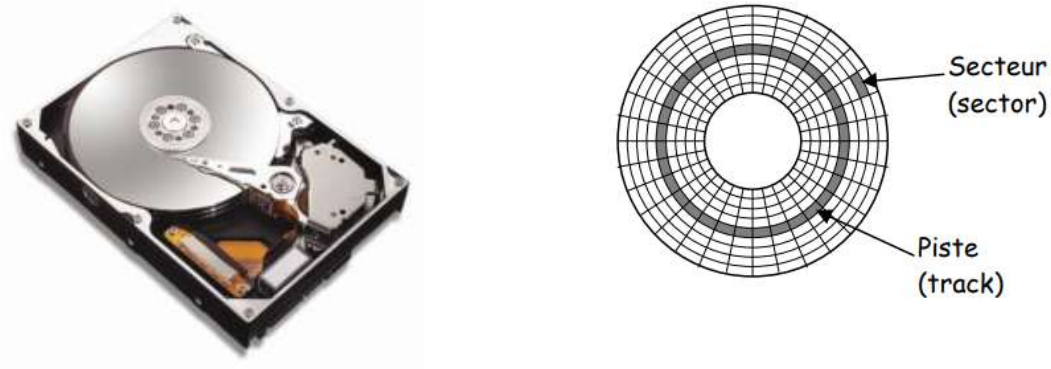


Fig5. Le disque dur

Les surfaces des disques sont divisées en pistes concentriques et en secteurs. L'ensemble des données situées sur une même piste de plateaux différents (c'est-à-dire à la verticale les unes des autres) est appelé cylindre. Le nombre de cylindres est égal au nombre de pistes sur une face d'un disque.

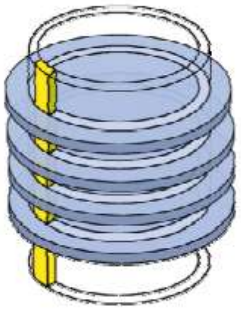


Fig6. Schéma d'un cylindre de disque dur

- **Remarque** : Le constructeur fournit généralement les caractéristiques CHS (Cylinders Heads Sectors). La taille d'un secteur étant de 512 octets, on peut alors calculer la capacité du disque dur :

$$\boldsymbol{CAPACITE\ (octets) = Nbre_{têtes} \times Nbre_{cylindres} \times Nbre_{secteurs} \times Taille_{Secteurs}}$$

3- Les mémoires

Une mémoire est un circuit à semi-conducteur permettant d'enregistrer, de conserver et de restituer des informations (instructions et variables). C'est cette capacité de mémorisation qui explique la polyvalence des systèmes numériques et leur adaptabilité à de nombreuses situations. Les informations peuvent être écrites ou lues.

Il y a écriture lorsqu'on enregistre des informations en mémoire, et lecture lorsqu'on récupère des informations précédemment enregistrées.

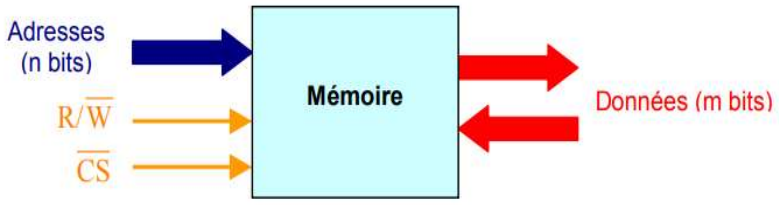
✓ Organisation d'une mémoire

Une mémoire peut être représentée comme une armoire de rangement constituée de différents tiroirs. Chaque tiroir représente une case mémoire qui peut contenir un seul élément (des données). Le nombre de cases mémoires pouvant être très élevé, celles-ci sont identifiées par un numéro. Ce numéro est appelé adresse. Chaque donnée devient alors accessible grâce à son adresse.

Adresse	Case mémoire
7 = 111	
6 = 110	
5 = 101	
4 = 100	
3 = 011	
2 = 010	
1 = 001	
0 = 000	0001 1010

Avec une adresse de n bits, on peut alors référencer 2^n cases mémoires. Chaque case est remplie par un mot de données (sa longueur m est toujours une puissance de 2).

On peut donc schématiser un circuit mémoire par la figure suivante où l'on peut distinguer :



- les entrées d'adresses
- les entrées de données
- les sorties de données
- les entrées de commandes :
  - une entrée de sélection de lecture ou d'écriture (R/W)
  - une entrée de sélection du circuit (CS)

- **Remarque :** Une opération de lecture ou d'écriture de la mémoire suit toujours le même cycle :
  1. sélection de l'adresse
  2. choix de l'opération à effectuer (R/W)

3. sélection de la mémoire (CS = 0)
4. lecture ou écriture la donnée

Les mémoires des tous premiers ordinateurs étaient magnétiques. Les mémoires sont maintenant des composants électroniques à base de transistors. Il existe deux types de mémoires qui se distinguent par leur technique de fabrication : les mémoires dynamiques et les mémoires statiques. Il s'agit dans les deux cas de mémoires volatiles qui nécessitent une alimentation pour conserver leur contenu. Nous pouvons citer :

- EPROM, EEPROM, les mémoires flash, DRAM, SRAM,….

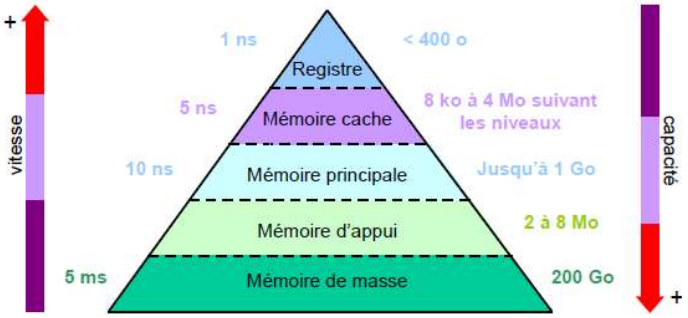
Mémoire dynamique	Mémoire statique
Grande densité d'intégration	Petite densité d'intégration
Bon marché	Chère
Lente	Rapide
Mécanisme de rafraîchissent	

✓ **Caractéristiques d'une mémoire**

- **La capacité :** c'est le nombre total de bits que contient la mémoire. Elle s'exprime aussi souvent en octet.
- **Le format des données :** c'est le nombre de bits que l'on peut mémoriser par case mémoire. On dit aussi que c'est la largeur du mot mémorisable.
- **Le temps d'accès :** c'est le temps qui s'écoule entre l'instant où a été lancée une opération de lecture/écriture en mémoire et l'instant où la première information est disponible sur le bus de données.
- **Le temps de cycle :** il représente l'intervalle minimum qui doit séparer deux demandes successives de lecture ou d'écriture.
- **Le débit :** c'est le nombre maximum d'informations lues ou écrites par seconde.
- **Volatilité :** elle caractérise la permanence des informations dans la mémoire. L'information stockée est volatile si elle risque d'être altérée par un défaut d'alimentation électrique et non volatile dans le cas contraire.

✓ **Notion de hiérarchie mémoire**

Une mémoire idéale serait une mémoire de grande capacité, capable de stocker un maximum d'informations et possédant un temps d'accès très faible afin de pouvoir travailler rapidement sur ces informations. Mais il se trouve que les mémoires de grande capacité sont souvent très lente et que les mémoires rapides sont très chères. Et pourtant, la vitesse d'accès à la mémoire conditionne dans une large mesure les performances d'un système. Afin d'obtenir le meilleur compromis coût-performance, on définit donc une hiérarchie mémoire. On utilise des mémoires de faible capacité mais très rapide pour stocker les informations dont le microprocesseur se sert le plus et on utilise des mémoires de capacité importante mais beaucoup plus lente pour stocker les informations dont le microprocesseur se sert le moins. Ainsi, plus on s'éloigne du microprocesseur et plus la capacité et le temps d'accès des mémoires vont augmenter.



- **Les registres** sont les éléments de mémoire les plus rapides. Ils sont situés au niveau du processeur et servent au stockage des opérandes et des résultats intermédiaires.
- **La mémoire cache** est une mémoire rapide de faible capacité destinée à accélérer l'accès à la mémoire centrale en stockant les données les plus utilisées.
- **La mémoire principale** est l'organe principal de rangement des informations. Elle contient les programmes (instructions et données) et est plus lente que les deux mémoires précédentes.
- **La mémoire d'appui** sert de mémoire intermédiaire entre la mémoire centrale et les mémoires de masse. Elle joue le même rôle que la mémoire cache.
- **La mémoire de masse** est une mémoire périphérique de grande capacité utilisée pour le stockage permanent ou la sauvegarde des informations. Elle utilise pour cela des supports magnétiques (disque dur, ZIP) ou optiques (CDROM, DVDROM).

III- Historique de l'ordinateur

L'homme a toujours eu besoin de compter, c'est qui est à l'origine de l'ordinateur qui est une machine automatique de traitement de l'information, obéissant à des programmes formés par des suites d'opérations arithmétiques et logiques.

- Le **bouclier** : Tout commence avec le bouclier (un type d'abaque). Cette invention, la machine à calculer, très utilisée par les chinois en 500 av. J.-C. était déjà utilisée par les babyloniens en 3000 av. J.-C. Il permettait de réaliser des additions, soustraction, multiplications et divisions.
- La **règle à calcul** : Les logarithmes, inventés en 1614 par l'écossais John Neper (1550-1617) permirent, en 1620, l'invention de la règle à calcul par William Oughtred (1574-1660). Cette règle permet, de manière analogique, d'effectuer des multiplications, des divisions, et des opérations plus complexes telle que les racines (carrées et cubiques), les calculs trigonométriques… Cette règle a été utilisée jusque dans les années 70 par les scientifiques.
- L'**horloge à calculer** : Wilhelm Schickard (1592-1635) inventa la première machine à calculer en 1623. **L'horloge à calculer** permettait les additions et les soustractions. Elle fut détruite dans un incendie.+++
- La **Pascaline** : Blaise Pascal (1623-1662), pour aider son père percepteur des impôts, invente en 1642, la Pascaline, une machine permettant les additions et les soustractions.

L'évolution remarquable que les mathématiques connurent jusqu'à la Seconde Guerre Mondiale ainsi que l'invention du premier tube à vide (la diode) par John Fleming en 1904, permirent de faire naître rapidement l'ordinateur électronique.

- Le **Z1** : Le Z1, premier ordinateur mécanique programmable, est créé en 1938 par Konrad Zuse (1910-1995). Sa conception fût réalisée dans le salon des parents de Zuse.
- L'**ABC** : John Atanasoof et Clifford Berry construisirent l'ABC (Atanasoff-Berry Computer), **le premier calculateur binaire à lampes**. Cet ordinateur est le premier à utiliser l'algèbre de Boole (nous y reviendrons au chapitre 4).
- L'**ENIAC** : John William Mauchly (1907-1980) et John Eckert (1919-1995) mirent au point l'ENIAC en 1946, **le premier calculateur électronique**. Cadencé à 100 kHz, il permettait de faire 330 multiplications par seconde. Il servit à la conception de la bombe H. https://fr.wikiversity.org/wiki/Logique_de_base/Introduction

Chapitre 2 : Système de numération

La création de la numération est un des faits les plus marquants de l'histoire de l'humanité. Si la plupart des civilisations ont adopté le système décimal, c'est qu'il a toujours été naturel de compter sur ses doigts. L'utilisation des phalanges et des articulations permit également d'améliorer ce simple procédé connu de tous. On utilise les " systèmes de numération" pour compter des objets et les représenter par des nombres.

I- Système de numération

Trois notions interviennent dans un système de numération :

- la **base B** du système, c'est un nombre entier quelconque.
- les **digits** du système sont des caractères tous différents et représentent chacun un élément de la base; il y en a donc **B** au total.
- le **poids** du digit selon son rang.

➢ **Exemple d'écriture d'un nombre A à 4 chiffres dans la base B** :

$(A)_B = a_3a_2a_1a_0$

$\forall i,\ a_i \in B,\ (A)_B = a_0B^0 + a_1B^1 + a_2B^2 + a_3B^3$;

Le poids du digit $a_i = B^i$

- **Généralisation** :

Soit un nombre $N = (a_{n-1}\ a_{n-2}\ a_{n-3} \ldots a_1a_0)_b$

Pour avoir une représentation du nombre N dans la base b, il suffit d'effectuer le calcul suivant :

$$(N)_b = a_{n-1} * b^{n-1} + \ldots + a_1 * b^1 + a_0 * b^0$$

La formule générale est donnée par l'expression suivante :

$$(N)_b = \sum_{i=0}^{n-1} ai * b^i$$

1. Système décimal

Dans la base 10 représentant le système décimal, il y a dix digits allant de 0 à 9 appelés chiffre. Soit le nombre 1234 représenté dans la base 10, on a :

$(1234)_{10}= 4x10^0 + 3x10^1 + 2x10^2 + 1x10^3$

$=4 + 30 + 200 + 1000$

La base B est 10 et le poids :

- du premier digit est 10^0 =1 (Unité)
- du deuxième digit est 10^1 =10 (Dizaine)
- du troisième digit est10^2 =100 (Centaine)
- du quatrième digit est 10^3 =1000 (Milliers)

2. Système binaire

Dans ce système, la base vaut 2, et il y a donc 2 digits 0 et 1 appelés dans ce cas « BIT » (Binary digIT).

Par exemple, le nombre 1011 exprimé en binaire signifie:

$(1011)_2 = 1x2^0 + 1x2^1 + 0x2^2 + 1x2^3$

$= 1 + 2 + 8$

3. Système octal

Dans ce système, la base vaut 8 et il y a 8 digits allant de 0 à 7.

Par exemple: le nombre 275 exprimé en octal :

$(275)_8 = 5x8^0 + 7x8^1 + 2x8^2$

$= 5 + 56 + 128$

4. Système hexadécimal

Dans ce système, la base vaut 16 et il y a 16 digits : 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E et F. Les dix premiers digits de 0 à 9 sont les chiffres du système décimal et les digits de 10 à 15 sont les premières lettres majuscules de l'alphabet.

Exemple, le nombre BAC exprimé en hexadécimal :

$(BAC)_{16} = Cx16^0 + Ax16^1 + Bx16^2$

$= 12 + 10x16 +11x256$

$= 12 + 160 + 2816$

II- Conversion ou changement de base

1. Conversion octal → binaire (binaire → octal)

La base octale correspond à la base 8 et on sait aussi que $8 = 2^3$. Pour une conversion **octale vers une base binaire**, il faut faire correspondre à chaque digit d'un nombre exprimé en octal un ensemble de 3 bits du même nombre exprimé en binaire.

Par exemple, la conversion de $(763)_8$ en base 2 donne :

$(763)_8 = (111)\ (110)\ (011)$

$= (111110011)_2$

- **Remarque** : Chaque digit a été éclaté sur 3 bits.

La conversion inverse de **binaire vers octal**, se fait de la même façon, en regroupant le nombre binaire par ensembles de 3 bits à partir de la droite.

Par exemple :

$(010111011101)_2 = (2735)_8$

2. Conversion hexadécimal → binaire (binaire → hexadécimal)

La base hexadécimale correspond à la base 16 et on remarque que $16 = 2^4$. On fera donc correspondre à chaque digit d'un nombre hexadécimal, 4 bits du nombre binaire correspondant.

Par exemple :

$(A28)_{16} = (101000101000)_2$

La conversion inverse de binaire à hexadécimal, se fait en regroupant le nombre binaire par ensembles de 4 bits à partir de la droite.

Par exemple:

$(101110011101001)_2 = (0101)\ (1100)\ (1110)\ (1001)$

$= (5CE9)_{16}$

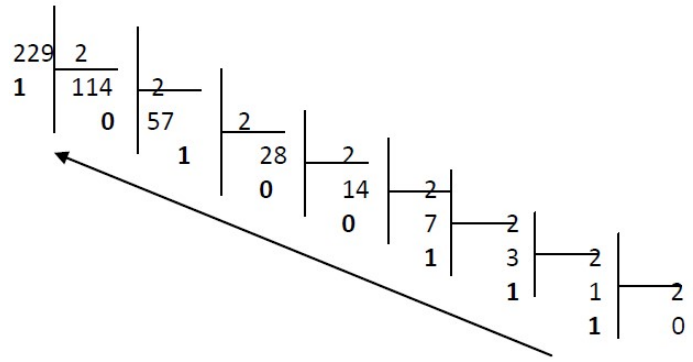
- **Remarque** : L'expression hexadécimale d'un nombre binaire est très utilisée pour interpréter des résultats fournis par un "microprocesseur".

3. Conversion décimale vers binaire, décimal vers octal ou décimal vers hexadécimal

La conversion de l'expression décimale d'un nombre en son expression binaire, octale ou hexadécimale repose sur la recherche des multiples des puissances successives de la base (2,8 ou 16 selon le cas) que contient ce nombre.

La méthode pratique consiste à effectuer des divisions successives: du nombre par la base, puis du quotient obtenu par la base, puis du nouveau quotient par la base,… jusqu'à ce que le quotient devienne nul. L'expression cherchée est constituée par l'ensemble des restes successifs des divisions, lu à l'envers.

Soit le nombre 229 écrit en base 10 à convertir en binaire. On a :



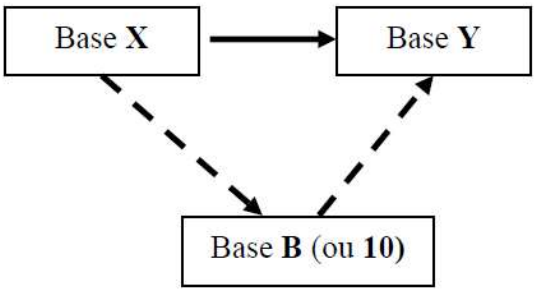
$(229)_{10} = (11100101)_2$

La même méthode sera applicable pour les conversions :

- décimal → octal (des divisions successives par 8)
- décimal → hexadécimal (des divisions successives par 16).

4. Conversion d'une base X vers base Y

Si $X = B^m$ et $Y = B^n$, alors convertir le nombre de la base X (B^m) vers B puis de la base B vers la base Y (B^n). Sinon, convertir de la base X vers la base **10** puis de la base **10** vers la base Y.



III- Représentation des nombres

On distingue deux catégories de codes: les "codes numériques" qui permettent seulement le codage des nombres, et les "codes alphanumériques" qui permettent le codage d'une information quelconque (ensembles de lettres, de chiffres et de symboles).

1. Entiers non signés

Tous les bits du nombre représentent un poids binaire. La valeur résultante est donc forcément entière et positive.

Sur n bits, distinguant 2^n valeurs distinctes et on peut alors coder les valeurs 0 à 2^{n-1}.

Poids	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	b7	b6	b5	b4	b3	b2	b1	b0

2. Entiers signés

La taille des mémoires des processeurs étant limitée, cette représentation facilite l'utilisation des nombres négatifs. Il s'agit alors de distinguer le signe de la valeur (+ ou -) par un bit supplémentaire.

Poids		2^6	2^5	2^4	2^3	2^2	2^1	2^0
	s	b6	b5	b4	b3	b2	b1	b0

Le bit le plus à gauche est réquisitionné pour représenter le bit de signe :

0 : nombre positif, 1 : nombre négatif.

- **NB :** Avec cette représentation, la valeur nulle ne peut prendre que deux (02) +0 ou -0, ce qui entraîne des problèmes de calcul et à conduire à la représentation en mode « complément à 2 » sur une taille (nombre de bits) dépendant du processeur utilisé (8 bits, 16 bits, 32 bits, ...).

3. Complément à 2

Pour avoir la représentation d'un nombre négatif en complément à 2, on **complémente** tous les bits et on ajoute **1**.

4. Nombres fractionnaires

Soit une base b associée à b symboles $\{S_0, S_1, S_2, ..., S_{b-1}\}$. Un nombre positif N dans un système de base b s'écrit sous la forme polynomiale :

$$N = a_{n-1} \cdot b^{n-1} + a_{n-2} \cdot b^{n-2} + \cdots + a_1 \cdot b^1 + a_0 \cdot b^0 + a_{-1} \cdot b^{-1} + a_{-2} \cdot b^{-2} + \cdots + a_{-m+1} \cdot b^{-m+1} + a_{-m} \cdot b^{-m}$$

La représentation simple de position est la suivante:

$$(a_{n-1}a_{n-2}a_1a_0, a_{-1}a_{-2}a_{-m+1}a_{-m})$$

a_i est le chiffre de rang i (a_i appartient à un ensemble de b symboles)
a_{n-1} est le chiffre le plus significatif
a_{-m} est le chiffre le moins significatif
$(a_{n-1}a_{n-2}...a_0)$ partie entière
$(a_{-1}a_{-2}...a_{-m})$ partie fractionnaire (<1)

✓ **<u>Méthode</u>** :

On multiplie la partie fractionnaire par la base en répétant l'opération sur la partie fractionnaire du produit jusqu'à ce qu'elle soit nulle (ou que la précision voulue soit atteinte).

Pour la partie entière, on procède par divisions comme pour un entier.

- **Exemple** : conversion de $(54,25)_{10}$ en base 2
  - Partie entière : $(54)_{10} = (110110)_2$ par divisions.
  - Partie fractionnaire :

$0,25 \times 2 = 0,50 \rightarrow a_{-1} = 0$
$0,50 \times 2 = 1,00 \rightarrow a_{-2} = 1$
$0,00 \times 2 = 0,00 \rightarrow a_{-3} = 0$

$(54,25)_{10} = (110110,01)_2$

5. Représentation des nombres réels

Le codage en complément à deux sur *n* bits ne permet de représenter qu'un intervalle de 2^n valeurs. Pour un grand nombre d'applications, cet intervalle de valeurs est trop restreint. La représentation à virgule flottante (*floating-point*) a été introduite pour répondre à ce besoin.

Pour des mots de 32 bits,

- la représentation en ***complément à deux*** permet de coder un intervalle de 2^{32} ***valeurs***
- tandis que la représentation à ***virgule flottante*** permet de coder un intervalle d'environ 2^{255} ***valeurs***.

La représentation en virgule flottante a été normalisée (norme IEEE 754)

Signe	*Exposant*	*Fraction*
S	*e*	*f*

Nombre de bits	Taille de *s*	Taille de *f*	Taille de *e*	*Emin*	*Emax*
32(simple précision)	1	23	8	- 126	127
64 (double précision)	1	52	11	-1022	1023

Représentation des nombres à virgule flottante dans la norme IEEE 754

Dans cette représentation, la valeur d'un nombre sur **32 bits** est donnée par l'expression :

$$(-1)^s \times \left(1 + \sum_{i=1}^{23} f_i 2^{-i}\right) \times 2^{e-E_m}$$

Où, *fi* correspond au $i^{ème}$ bit de la fraction *f*.

- **Exemple** : n = 32 bits

A = **1** 10000100 01010000000000000000000

S= 1(-), e=132-127=5, f= $2^{-2}+2^{-4}$ = 0,25+0,0625 = 0,3125

A= **-** 1,3125x2^5

Chapitre 3 : Opérations arithmétiques et logiques

Les quatre opérations arithmétiques fondamentales (addition, soustraction, multiplication et division) peuvent évidemment s'effectuer non seulement dans le système décimal mais aussi dans les autres systèmes numériques et en particulier dans le système binaire; les règles du système décimal seront valables pour ces opérations.

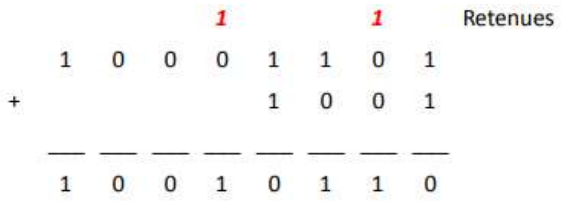
I- Opérations arithmétiques

1- Addition en binaire

- **Règles d'addition**

  0 + 0 = 0
  0 + 1 = 1
  1 + 0 = 1
  1 + 1 = 0 avec retenue de 1

- **Exemple**



- **Applications**
  - Faire en binaire l'addition des nombres suivants :
    101001 + 101 ; 11101010 + 10101 ; 1101011 + 10111 + 101

2- Soustraction en binaire

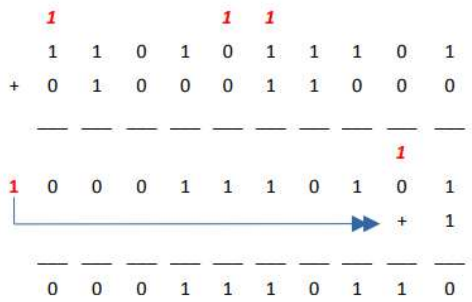
- **Règles de la soustraction**

  0 – 0 = 0
  0 – 1 = 1 et on retient 1
  1 – 0 = 1
  1 – 1 = 0

- **1ère Méthode : Complément à 1 et addition**

Le complément à 1 d'un nombre binaire est la valeur obtenue en inversant tous les bits de ce nombre (en permutant les 0 par des 1 et inversement). La soustraction consiste à ajouter au nombre binaire (le diminuende) le complément à 1 de l'autre nombre (le diminuteur, nombre à soustraire).

- **Exemple** : Soustrayons les nombres 1101011101 et 1011100111
  Le complément à 1 de 1011100111 est 0100011000



- **2eme Méthode : Complément à 2**

Le complément à 2 d'un nombre binaire est la valeur obtenue en faisant le complément à 1 + (1).

La soustraction par complément à 2 consiste à complémenter le diminuteur ensuite à additionner les deux nombres (diminuende + diminuteur complémenté à 2).

- **NB** : Les deux nombres doivent avoir le même nombre de bits. S'il y a un bit de dépassement, il faut l'ignorer.
  Il est à noter également que lorsque le résultat d'une soustraction est négatif, le nombre résultant est le complément à 2 de la valeur positive.
  Le premier bit est le bit de signe. Il est à 0 pour les nombres positifs et à 1 pour les nombres négatifs.

3- Multiplication binaire

- **Règles de la multiplication**

  0 * 0 = 0

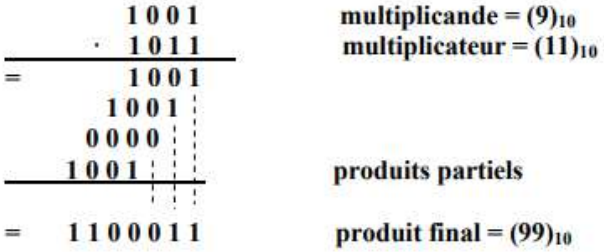
  0 * 1 = 0

  1 * 0 = 0

  1 * 1 = 1

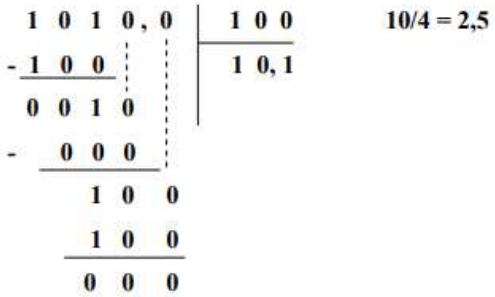
➢ **NB :** L'opération s'effectue comme en base 10

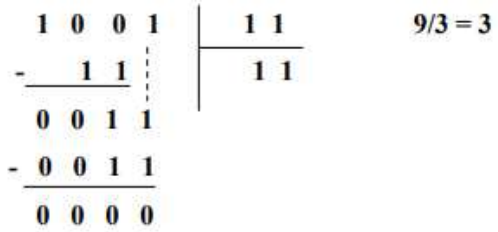
- **Exemple**:



4- Division binaire

La division s'effectue de la même manière qu'en base 10.





II- Exercices : Opérations arithmétiques en binaire

a- Effectuer les additions suivantes :

1100 + 0011=……

1111 + 0101=……..……..

10101010 + 00110011= ……..

11001101 + 11100011= ……..

b- Effectuer les soustractions suivantes:

1111 - 0101= ……..……

1100 - 0011= ……..……

10101010 - 00110011= ……..…….

11001101 - 01100011= ……..……

c- Multiplier les nombres suivants par 2, 6, 8 :

00001100, 00010101, 10101000

d- Diviser les nombres suivants par 2,4, 8 :

11000000, 01010000, 11001100

Chapitre 4 : Algèbre de Boole et Tableau de Karnaugh

Les machines numériques sont constituées d'un ensemble de circuits électroniques. Chaque circuit fournit une fonction logique bien déterminée (addition, comparaison,….). Pour concevoir et réaliser ce circuit on doit avoir un modèle mathématique de la fonction réalisée par ce circuit. Ce modèle doit prendre en considération le système binaire et le modèle mathématique utilisé est celui de Boole.

I- Algèbre de Boole

L'algèbre de Boole ou calcul booléen est un ensemble de règles utilisées pour simplifier les expressions logiques sans pour autant changer leur fonctionnalité. Elle fut lancée en 1854 par le mathématicien britannique George Boole. Elle utilise des techniques algébriques pour traiter les expressions à deux, trois ou quatre variables du calcul.

Aujourd'hui, l'algèbre de Boole trouve de nombreuses applications en informatique et dans la conception des circuits électroniques.

1. Variable logique (booléenne)

Une variable logique est une variable qui peut prendre soit la valeur 0 ou 1. Généralement, elle est exprimée par un seul caractère alphabétique en majuscule (A, B, S, …).

- **Exemple** : Une lampe : allumée L = 1, éteinte L = 0.

2. Fonction logique

C'est une fonction qui relie N variables logiques avec un ensemble d'opérateurs logiques de base. Dans l'Algèbre de Boole, il existe trois opérateurs de base :

NON, ET, OU.

La valeur d'une fonction logique est égale à 1 ou 0 selon les valeurs des variables logiques. Si une fonction logique possède N variables logiques, on a :

- 2^n combinaisons
- la fonction possède 2^n valeurs.
- les 2^n combinaisons sont représentées dans une table qui s'appelle table de vérité.

- **Exemple : Fonction logique** $F(A,B,C)=\overline{A}.\overline{B}.C+\overline{A}.B.C+A.\overline{B}.C+A.B.C$

La fonction possède 3 variables, on a 2^3 combinaisons. Ce qui donne :

$$F(0,0,0)=\overline{0}.\overline{0}.0+\overline{0}.0.0+0.\overline{0}.0+0.0.0=0$$
$$F(0,0,1)=\overline{0}.\overline{0}.1+\overline{0}.0.1+0.\overline{0}.1+0.0.1=1$$
$$F(0,1,0)=\overline{0}.\overline{1}.0+\overline{0}.1.0+0.\overline{1}.0+0.1.0=0$$
$$F(0,1,1)=\overline{0}.\overline{1}.1+\overline{0}.1.1+0.\overline{1}.1+0.1.1=1$$
$$F(1,0,0)=\overline{1}.\overline{0}.0+\overline{1}.0.0+1.\overline{0}.0+1.0.0=0$$
$$F(1,0,1)=\overline{1}.\overline{0}.1+\overline{1}.0.1+1.\overline{0}.1+1.0.1=1$$
$$F(1,1,0)=\overline{1}.\overline{1}.0+\overline{1}.1.0+1.\overline{1}.0+1.1.0=0$$
$$F(1,1,1)=\overline{1}.\overline{1}.1+\overline{1}.1.1+1.\overline{1}.1+1.1.1=1$$

A	B	C		F
0	0	0		0
0	0	1		1
0	1	0		0
0	1	1		1
1	0	0		0
1	0	1		1
1	1	0		0
1	1	1		1

Table de vérité

3. Opérateurs logiques de base

- **NON (négation)** est un opérateur unaire (une seule variable) qui a pour rôle d'inverser la valeur d'une variable.

F(A)= Non A = $\overline{A}$
(lire : A barre)

A	$\overline{A}$
0	1
1	0

- **ET (AND)** est un opérateur binaire (deux variables), a pour rôle de réaliser le produit logique entre deux variables booléennes.

- Le ET est défini par : F(A,B)= A . B

A	B	A . B
0	0	0
0	1	0
1	0	0
1	1	1

- **OU (OR)** est un opérateur binaire (deux variables), a pour rôle de réaliser la somme logique entre deux variables logiques.

Le OU est défini par F(A,B)= A + B (il ne faut pas confondre avec la somme arithmétique)

A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

- **<u>Remarque</u> :**
  - Dans la définition des opérateurs ET, OU, nous avons juste donné la définition de base avec deux variables logiques.
  - L'opérateur ET peut réaliser le produit de plusieurs variables logique.
    L'opérateur OU peut aussi réaliser la somme logique de plusieurs variables logiques.
  - Dans une expression on peut aussi utiliser les parenthèses.
  - Pour évaluer une expression logique (fonction logique), il faut :
    - commencer par évaluer les sous expressions entre les parenthèses.
    - Puis le complément (NON), ensuite le produit logique (ET) et enfin la somme logique (OU).

4. Lois fondamentales de l'Algèbre de Boole

$(A.B).C = A.(B.C) = A.B.C$	Associativité
$A.B = B.A$	Commutativité
$A.A = A$	Idempotence
$A.1 = A$	Elément neutre
$A.0 = 0$	Elément absorbant

$(A+B)+C = A+(B+C) = A+B+C$	Associativité
$A+B = B+A$	Commutativité
$A+A = A$	Idempotence
$A+0 = A$	Elément neutre
$A+1 = 1$	Elément absorbant

$A.(B+C) = (A.B)+(A.C)$ Distributivité du ET sur le OU
$A+(B.C) = (A+B).(A+C)$ Distributivité du OU sur le ET

•Autres relations utiles

$A + (A . B) = A$
$A . (A + B) = A$
$(A + B) . (A + \overline{B}) = A$
$A + \overline{A} . B = A + B$

$\overline{\overline{A}} = A$
$\overline{A} + A = 1$
$\overline{A} . A = 0$

Théorème de DE-MORGANE

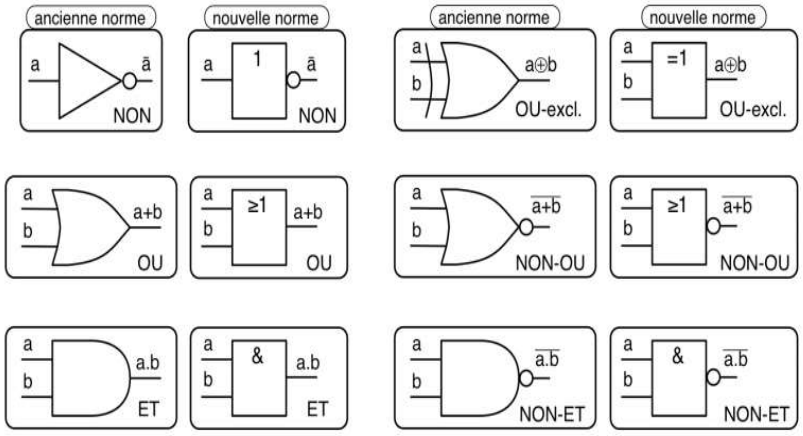
$\overline{A+B} = \overline{A} . \overline{B}$
$\overline{A.B} = \overline{A} + \overline{B}$

OU exclusif (XOR)

$A \oplus B = \overline{A}.B + A.\overline{B}$

5. Portes logiques

Une porte logique est un circuit électronique élémentaire qui permet de réaliser la fonction d'un opérateur logique de base. Différentes normes ont été définies pour représenter les portes logiques.



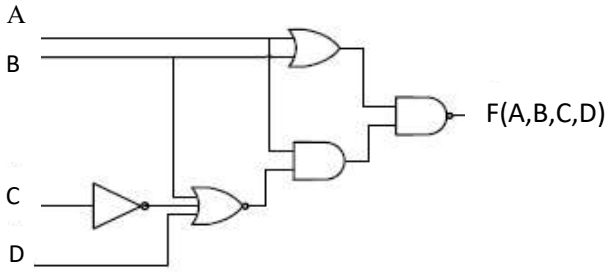
Symboles des portes logiques.

- **Schéma d'un circuit logique**

C'est la traduction de la fonction logique en un schéma électronique. Le principe consiste à remplacer chaque opérateur logique par la porte logique qui lui correspond.

- **Exemple :**

$$F(A,B,C,D) = \overline{(A+B).\overline{(B+\overline{C}+D)}.A}$$



Les fonctions et les circuits logiques sont souvent définis à partir d'expressions algébriques. Afin de réduire la complexité de ces circuits et fonctions, il est important de les simplifier. Plusieurs méthodes existent pour la simplification :

- la méthode algébrique,

- les méthodes graphiques (Ex : table de Karnaugh)

II- Méthodes de simplifications

1. Méthode algébrique

Le principe consiste à appliquer les règles de l'algèbre de Boole afin d'éliminer des variables ou des termes.

- **Exemple :**

$$\begin{aligned} ABC + AB\overline{C} + A\overline{B}CD &= AB(C + \overline{C}) + A\overline{B}CD \\ &= AB + A\overline{B}CD \\ &= A(B + \overline{B}(CD)) \\ &= A(B + CD) \\ &= AB + ACD \end{aligned}$$

- **Exercice** :

1- Démontrer la proposition suivante :

$$A.B + B.C + A.C + A.\overline{B}.\overline{C} + \overline{A}.B.\overline{C} + \overline{A}.\overline{B}.C = A + B + C$$

= AC+ A**BC**+ AB+ABC+BC+ABC

= A(C+**BC**) **+B(A+AC)+ C(B+AB)**

=A(C+B)+ B(A+C)+ C(B+A)

=AC+AB +BA+BC+ CB +CA

=AC+ CA +AB+BA+BC+CB

= C(A+A) + A(B+B) + B(C+C)

= C +A+B

2- Donner la forme simplifiée de la fonction suivante :

$$F(A,B,C,D) = \overline{A}BCD + A\overline{B}CD + AB\overline{C}D + ABC\overline{D} + ABCD$$

=ABCD+ ABCD+ ABCD+ ABCD+ ABCD

= ABCD+ ABCD+ ABCD+ ABC(D+ D)

= ABCD+ ABCD+ ABCD+ ABC

= ABCD+ ABC+ ABCD+ ABCD

=BC(AD+A) + ABCD+ ABCD

=BC(D+A)+ ABCD+ ABCD

=BCD+ABC+ ABCD+ ABCD

= BCD+ ABCD+ABC+ ABCD

=CD(B+AB)+ AB(C+CD)

=CD(B+A)+ AB(C+D)

=ABCD+ ABCD+ ABCD+ ABCD+ ABCD

=CD(AB+AB)+ AB(CD+CD)+ ABCD

= CD(A⊕B)+ AB(C⊕ D)+ABCD

2. Simplification par la table de Karnaugh

1- Les termes adjacents

Soit l'expression suivante : $A \cdot B + A \cdot \overline{B}$

Les deux termes possèdent les mêmes variables, seul l'état de la variable B qui change. Si on applique les règles de simplification, on obtient :

$$AB + A\overline{B} = A(B + \overline{B}) = A$$

Ces termes sont alors dites **adjacents**.

2- Description de la table de Karnaugh

La méthode de Karnaugh se base sur la règle précédente des termes adjacents. Elle consiste à mettre en évidence par une méthode graphique (un tableau) tous les termes qui sont adjacents.

La méthode peut s'appliquer aux fonctions logiques de 2, 3, 4, 5 et 6 variables.

Un tableau de Karnaugh comportent 2^n cases (N est le nombre de variables).

- **Exemple de représentation graphique**

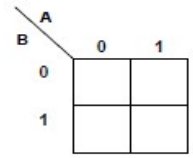
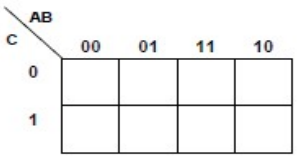
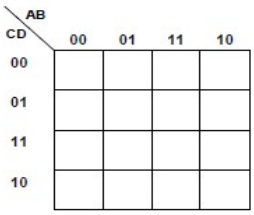


Tableau à 2 variables



Tableaux à 3 variables

Tableau à 4 variables



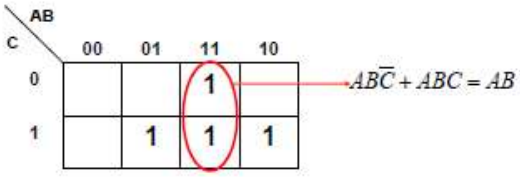
3- Méthode de simplification

L'idée de base est d'essayer de regrouper les cases adjacentes qui comportent des 1, c'est-à-dire rassembler les termes adjacents.

Essayer de faire des regroupements avec le maximum de cases (16, 8, 4 ou 2).

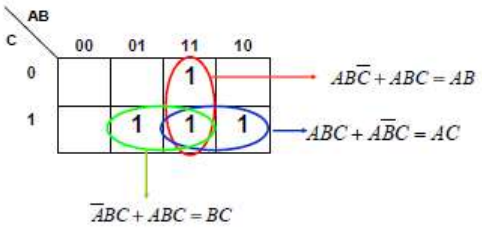
- **Application** :

Soit la fonction à 3 variables suivante à simplifier : $A\bar{B}\bar{C} + ABC = AB$



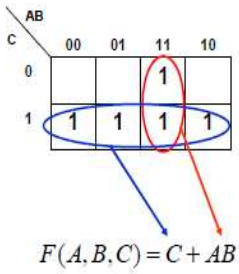
Puisqu'il existe encore des cases qui sont en dehors d'un regroupement, on refait la même procédure pour former des regroupements.

Il est important de rappeler qu'une case peut appartenir à plusieurs regroupements.



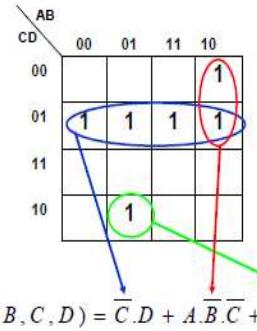
$$F(A,B,C) = AB + AC + BC$$

Exemple 1 : 3 variables



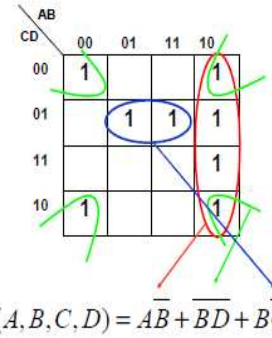
$$F(A,B,C) = C + AB$$

Exemple 2 : 4 variables



$$F(A,B,C,D) = \overline{C}.D + A.\overline{B}.\overline{C} + \overline{A}.B.C.\overline{D}$$

Exemple 3 : 4 variables



$$F(A,B,C,D) = A\overline{B} + \overline{BD} + B\overline{CD}$$

III- Exercice

Trouver la forme simplifiée des fonctions à partir des deux tableaux?