

Introduction aux systèmes d'information

Sommaire

A. Petite histoire de l'informatique	3
B. Principes de base	8
C. Le microprocesseur	20
D. La mémoire	30
E. L'ordinateur	43
F. Les systèmes d'exploitation	59
G. Les logiciels.....	63
H. Les réseaux	64
I. La programmation	89
J. Notes	95

NOTA : ce support est une compilation de différentes sources

A. Petite histoire de l'informatique

Ceci est une traduction (aimablement autorisée par l'auteur) de *A Very Brief History of Computer Science*, texte écrit en 1995 (et revu en 1999) par Jeffrey Shallit pour ses étudiants de l'Université de Waterloo (Canada).

La plupart des notices biographiques obtenues en cliquant sur les noms cités font partie de l'excellente base de données MacTutor History of Mathematics archive, développée et gérée par John O'Connor et Edmund Robertson à l'Université de St Andrews (Ecosse).

Avant 1900

Les machines à calculer sont utilisées depuis des milliers d'années : on trouvait probablement des abaques à Babylone en 3000 avant notre ère. Les Grecs ont fabriqué des calculateurs analogiques très perfectionnés. En 1901, au large de l'île d'Antikythera, on a découvert une épave dans laquelle se trouvait, encroûté de sel, un assemblage d'engrenages rouillés (le mécanisme d'Antikythera), daté d'environ 80 avant notre ère, que l'on a reconstruit : il servait à prédire les mouvements des astres.

L'Ecossais John Napier (1550-1617), l'inventeur des logarithmes, fabriqua vers 1610 les règles de Napier pour simplifier la multiplication.

En 1641, Blaise Pascal (1623-1662) construisit une machine à additionner. Un travail analogue fut réalisé par Gottfried Wilhelm Leibniz (1646-1716), qui préconisa l'utilisation du système binaire pour les calculs. On a récemment découvert que Wilhelm Schickard (1592-1635), professeur à l'Université de Tübingen, avait construit une machine de ce genre vers 1623 ou 1624 (avant Pascal et Leibniz), qu'il décrivit brièvement dans deux lettres à Johannes Kepler. Malheureusement, la machine brûla dans un incendie, et Schickard lui-même mourut de la peste bubonique en 1635, durant la Guerre de Trente Ans.

Joseph-Marie Jacquard (1752-1834) inventa un métier à tisser dont les motifs étaient indiqués par des cartons perforés. Charles Babbage (1792-1871) construisit deux machines : la machine différentielle (exposée au Science Museum de Londres) et la machine analytique, beaucoup plus ambitieuse (un précurseur de l'ordinateur), mais aucune des deux ne marchait correctement. (Babbage, que l'un de ses biographes traite de « génie irascible », était un peu bizarre. On ignore généralement qu'il est l'inventeur de la dendrochronologie, ou datation des arbres; il ne poursuivit pas ses recherches à ce sujet. Devenu vieux, il consacra une grande partie de son temps à persécuter les joueurs d'orgue de Barbarie.)

Une amie de Babbage, Ada Byron, comtesse de Lovelace (1815-1852), est parfois considérée comme le premier programmeur de l'Histoire, en raison d'un rapport qu'elle écrivit sur la machine de Babbage. (Le langage de programmation Ada a été nommé en son honneur.)

L'économiste et logicien anglais William Jevons (1835-1882) construisit en 1869 une machine à résoudre des problèmes de logique : « la première machine suffisamment puissante pour résoudre un problème compliqué plus rapidement qu'à la main » (Martin Gardner). La machine se trouve actuellement au Museum of the History of Science d'Oxford.

Le statisticien américain Herman Hollerith (1860-1929) inventa la carte perforée moderne pour l'utiliser dans une machine destinée à analyser les résultats du recensement de 1890.

1900 - 1939: l'avancée mathématique

L'étude des machines à calculer se poursuivait. On construisit des machines destinées à une utilisation particulière: ainsi, en 1919, le lieutenant d'infanterie E. Carissan (1880-1925) conçut et réalisa une merveilleuse machine à factoriser les entiers. L'Espagnol Leonardo Torres y Quevedo (1852-1936) construisit plusieurs machines électromécaniques, dont l'une qui jouait des fins de parties d'échecs.

En 1928, le mathématicien David Hilbert (1862-1943) posa trois questions au Congrès International des Mathématiciens : (1) Les mathématiques sont-elles **complètes**? (tout énoncé mathématique peut-il être soit prouvé, soit réfuté ?) (2) Les mathématiques sont-elles **cohérentes**? (peut-on être sûr que des raisonnements valides ne conduiront pas à des absurdités ?) (3) Les mathématiques sont-elles **décidables**? (existe-t-il un algorithme pouvant dire de n'importe quel énoncé mathématique s'il est vrai ou faux ?) Cette dernière question est connue sous le nom de *Entscheidungsproblem*.

En 1931, Kurt Gödel (1906-1978) répondit à deux de ces questions. Il démontra que tout système formel suffisamment puissant est soit incohérent, soit incomplet. De plus, si un système d'axiomes est cohérent, cette cohérence ne peut être prouvée en n'utilisant que les axiomes. La troisième question restait ouverte, en remplaçant « vrai » par « prouvable » (existe-t-il un algorithme pour dire si une assertion peut être prouvée ?)

En 1936, Alan Turing (1912-1954) résolut l'*Entscheidungsproblem* en construisant un modèle formel de calculateur - la machine de Turing - et en prouvant qu'une telle machine ne pouvait pas résoudre certains problèmes, en particulier le problème d'arrêt : étant donné un programme, peut-on dire s'il termine pour n'importe quelle valeur des données ?

Les années 1940 : la guerre fait naître l'ordinateur électronique

La complication des calculs balistiques, durant la seconde guerre mondiale, aiguillonna le développement de l'ordinateur électronique. En 1944, à Harvard, Howard Aiken (1900-1973) construisit le calculateur électromécanique Mark I, avec l'aide d'IBM.

Le décryptage militaire conduisit aussi à des projets d'ordinateur. Alan Turing, en Angleterre, travaillait à décoder la machine allemande Enigma; les Anglais construisirent un calculateur, le Colossus, pour aider au décryptage.

En 1939, à l'Université d'Iowa, John Atanasoff (1904-1995) et Clifford Berry conçurent et réalisèrent l'ABC, un calculateur électronique pour résoudre des systèmes d'équations linéaires, mais il ne fonctionna jamais correctement.

Atanasoff discuta de son invention avec John Mauchly (1907-1980), qui, plus tard, avec John Eckert (1919-1995), conçut et réalisa l' ENIAC, un calculateur électronique destiné à l'origine aux calculs balistiques. On ne sait pas très bien quelles idées Atanasoff transmit à Mauchly; le mérite d'avoir inventé le premier ordinateur revient-il à Atanasoff ou à Mauchly et Eckert ? Ce fut le sujet de batailles juridiques, c'est encore celui d'un débat historique. L'ENIAC fut construit à l'Université de Pennsylvanie, et terminé en 1946.

En 1944, Mauchly, Eckert, et John von Neumann (1903-1957) travaillaient à la conception d'un ordinateur électronique, l'EDVAC. Le premier rapport de Von Neumann sur l'EDVAC eut beaucoup d'influence; on y trouve de nombreuses idées encore utilisées dans les ordinateurs les plus modernes, dont une routine de tri par fusion. Eckert et Mauchly reprirent ces idées pour construire l'UNIVAC.

Pendant ce temps, en Allemagne, Konrad Zuse (1910-1995) construisait le premier calculateur programmable universel (non spécialisé), le Z3 (1941).

En 1945, Vannevar Bush publia *As We May Think*, un article étonnamment prophétique sur le traitement de l'information, et ses effets sur la société dans les temps à venir.

En Angleterre, Maurice Wilkes (né en 1913) construisit l'EDSAC (à partir de l'EDVAC). F. Williams (né en 1911) et son équipe construisirent le Manchester Mark I, dont une version fut opérationnelle dès juin 1948. Certains considèrent cette machine comme le premier ordinateur à programme en mémoire (architecture dite de Von Neumann).

L'invention du transistor en 1947 par John Bardeen, Walter Brattain et William Shockley transforma l'ordinateur, et permit la révolution du microprocesseur. Pour cette découverte, ils reçurent le Prix Nobel de Physique en 1956. (Par la suite, Shockley se rendit célèbre pour ses points de vue racistes.)

Jay Forrester (né en 1918) inventa vers 1949 la mémoire à noyau magnétique.

Les années 50

Grace Hopper (1906-1992) inventa la notion de compilateur (1951). (Quelques années plus tôt, elle avait trouvé le premier bug de l'histoire de l'informatique, une phalène entrée dans le Mark II de Harvard.)

John Backus et son équipe écrivirent le premier compilateur FORTRAN en avril 1957. LISP (List Processing), un langage de traitement de listes pour l'intelligence artificielle, fut inventé par John McCarthy vers 1958. Alan Perlis, John Backus, Peter Naur et leurs associés développèrent Algol (Algorithmic Language) en 1959. Jack Kilby (Texas Instruments) et Robert Noyce (Fairchild Semiconductor) inventèrent les circuits intégrés en 1959.

Edsger Dijkstra (1930-2002) trouva un algorithme efficace pour résoudre le problème des plus courts chemins dans un graphe, à titre de démonstration pour l'ARMAC en 1956. Il trouva aussi un algorithme efficace de recherche d'un arbre recouvrant de poids minimal, afin de minimiser le câblage du X1. (Dijkstra est célèbre pour ses déclarations caustiques et péremptoires; voir par exemple son avis sur quelques langages de programmation).

Dans un célèbre article de la revue *Mind*, en 1950, Alan Turing décrivit le test de Turing, l'une des premières avancées en intelligence artificielle. Il proposait une définition de la « pensée » ou de la « conscience » relative à un jeu : un examinateur pose des questions par écrit à un interlocuteur situé dans la pièce voisine, et doit décider, au vu des réponses, si son interlocuteur est une machine ou un être humain.

S'il est incapable de répondre, on peut raisonnablement dire que l'ordinateur « pense ». En 1952, Alan Turing fut arrêté pour outrage aux bonnes moeurs après qu'une plainte pour cambriolage eut révélé sa liaison avec Arnold Murray. L'homosexualité affichée était tabou dans l'Angleterre des années 1950, et on obligea Turing à suivre un « traitement » hormonal qui le rendit impuissant et lui fit pousser des seins. Le 7 juin 1954, Turing se suicida en mangeant une pomme enrobée de cyanure.

Les années 1960

Dans les années 1960, l'informatique devint une discipline à part entière. Le premier département d'informatique fut créé en 1962 à l'Université de Purdue; le premier Ph.D. d'informatique fut délivré à Richard Wexelblat par l'Université de Pennsylvanie, en décembre 1965.

Il y eut une percée dans les systèmes d'exploitation. Fred Brooks (IBM) conçut System/360, une série d'ordinateurs de tailles variées, avec la même architecture et le même ensemble d'instructions. Edsger Dijkstra, à Eindhoven, conçut le système multiprogramme THE.

De nombreux langages de programmation virent le jour, tels que BASIC, développé vers 1964 par John Kemeny (1926-1992) et Thomas Kurtz (né en 1928).

Les années 1960 virent émerger la théorie des automates et des langages formels : on peut notamment citer Noam Chomsky (qui se fit plus tard remarquer par la théorie suivant laquelle le langage est « câblé » dans le cerveau, et pour sa critique de la politique étrangère des Etats-Unis) et Michael Rabin.

On commença aussi à utiliser des méthodes formelles pour prouver la correction des programmes. Les travaux de Tony Hoare (l'inventeur de Quicksort) jouèrent un rôle important.

Vers la fin de la décennie, on commença à construire ARPAnet, le précurseur d'Internet. Ted Hoff (né en 1937) et Federico Faggin (Intel) concurent le premier microprocesseur en 1969-1971.

Donald Knuth (né en 1938), auteur du traité *The Art of Computer Programming*, posa des fondements mathématiques rigoureux pour l'analyse des algorithmes.

Les années 1970

Les travaux d'Edgar Codd (1924-2003) sur les bases de données relationnelles permirent une avancée majeure dans la théorie des bases de données. Codd reçut le Turing Award en 1961. Le système d'exploitation Unix fut développé aux Bell Laboratories par Ken Thompson (né en 1943) et Dennis Ritchie (né en 1941). Brian Kernighan et Ritchie développèrent C, un important langage de programmation.

On vit apparaître de nouveaux langages, tels que Pascal (inventé par Niklaus Wirth) et Ada (réalisé par une équipe dirigée par Jean Ichbiah). La première architecture RISC fut commencée par John Cocke en 1975, chez IBM. Vers cette époque, des projets analogues démarrèrent à Berkeley et Stanford. Les années 1970 virent aussi naître les super-ordinateurs. Seymour Cray (né en 1925) conçut le CRAY-1, qui apparut en mars 1976; il pouvait exécuter

160 millions d'opérations par seconde. Le Cray XMP sortit en 1982. Cray Research (à présent repris par Silicon Graphics) continue à construire des ordinateurs géants.

Il y eut aussi des progrès importants en algorithmique et en théorie de la complexité. En 1971, Steve Cook publia son article fondamental sur la NP-complétude, et, peu après, Richard Karp montra que de nombreux problèmes combinatoires naturels étaient NP-complets.

Whit Diffie et Martin Hellman publièrent un article fondant la théorie de cryptographie à clef publique; le système de cryptage RSA fut inventé par Ronald Rivest, Adi Shamir, et Leonard Adleman.

En 1979, trois étudiants de Caroline du Nord développèrent un serveur de nouvelles distribué qui finalement devint Usenet.

Les années 1980

Cette décennie vit apparaître le micro-ordinateur personnel, grâce à Steve Wozniak et Steve Jobs, fondateurs de Apple Computer. Les premiers virus informatiques apparurent en 1981 (leur nom est dû à Leonard Adleman).

En 1981, l'Osborne I, le premier ordinateur vraiment portable, fut mis sur le marché. En 1984, Apple commercialisa le Macintosh. En 1987, l'US National Science Foundation démarra NSFnet, qui devait devenir une partie de l'Internet actuel.

Les années 1990 et au-delà

On continue à développer des ordinateurs parallèles.

L'informatique biologique, avec les récents travaux de Leonard Adleman sur l'utilisation de l'ADN comme calculateur non déterministe, ouvre de grandes perspectives. Le projet Génome Humain cherche à séquencer tout l'ADN d'un individu.

Peter Shor découvre que l'on peut efficacement factoriser des entiers sur un ordinateur quantique (théorique), ce qui ouvre la voie à la programmation quantique.

Les autoroutes de l'information relient de plus en plus les ordinateurs du monde entier.

Les ordinateurs sont de plus en plus petits; naissance de la nano-technologie.

B. Principes de base

Présentation du binaire

Vers la fin des années 30, Claude Shannon démontra qu'à l'aide de « contacteurs » (interrupteurs) fermés pour « vrai » et ouverts pour « faux » il était possible d'effectuer des opérations logiques en associant le nombre 1 pour « vrai » et 0 pour « faux ».

Ce codage de l'information est nommé **base binaire**. C'est avec ce codage que fonctionnent les ordinateurs. Il consiste à utiliser deux états (représentés par les chiffres 0 et 1) pour coder les informations.

L'homme calcule depuis 2000 ans avant Jésus-Christ avec 10 chiffres (0, 1, 2, 3, 4, 5, 6, 7, 8, 9), on parle alors de base décimale (ou base 10). Toutefois dans des civilisations plus anciennes ou pour certaines applications actuelles d'autres bases de calcul ont et sont toujours utilisées :

- base sexagésimale (60), utilisée par les Sumériens. Cette base est également utilisée dans le système horaire actuel, pour les minutes et les secondes ;
- base vicésimale (20), utilisée par les Mayas ;
- base duodécimale (12), utilisée par les anglo-saxons dans leur système monétaire jusqu'en 1960 : un « pound » représentait vingt « shilling » et un « shilling » représentait douze « pences ». Le système d'heure actuel fonctionne également sur douze heures (notamment dans la notation anglo-saxonne) ;
- base quinaire (5), utilisée par les Mayas ;
- base binaire (2), utilisée par l'ensemble des technologies numériques.

Bit

Le terme **bit** (*b* avec une minuscule dans les notations) signifie « **binary digit** », c'est-à-dire 0 ou 1 en numérotation binaire. Il s'agit de la plus petite unité d'information manipulable par une machine numérique. Il est possible de représenter physiquement cette information binaire :

- par un signal électrique ou magnétique, qui, au-delà d'un certain seuil, correspond à la valeur 1 ;
- par des aspérités géométriques dans une surface ;
- grâce à des bistables, c'est-à-dire des composants électroniques qui ont deux états d'équilibre (l'un correspond à l'état 1, l'autre à 0).

Avec un bit il est ainsi possible d'obtenir deux états : soit 1, soit 0. Grâce à 2 bits, il est possible d'obtenir quatre états différents (2^2) :

0	0
0	1
1	0
1	1

Avec 3 bits, il est possible d'obtenir huit états différents (2^3) :

Valeur binaire sur 3 bits	Valeur décimale
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Pour un groupe de n bits, il est possible de représenter 2^n valeurs.

Poids des bits

Dans un nombre binaire, la valeur d'un bit, appelée **poids**, dépend de la position du bit en partant de la droite. A la manière des dizaines, des centaines et des milliers pour un nombre décimal, le poids d'un bit croît d'une puissance de deux en allant de la droite vers la gauche comme le montre le tableau suivant :

Nombre binaire	1	1	1	1	1	1	1	1
Poids	$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$

Conversions

Pour convertir un mot binaire en nombre décimal, il suffit de multiplier la valeur de chaque bit par son poids, puis d'additionner chaque résultat. Ainsi, le mot binaire 0101 vaut en décimal :

$$\begin{aligned}
 & 2^3 \times 0 + 2^2 \times 1 + 2^1 \times 0 + 2^0 \times 1 \\
 & = 8 \times 0 + 4 \times 1 + 2 \times 0 + 1 \times 1 \\
 & = 5
 \end{aligned}$$

Octet

L'**octet** (en anglais *byte* ou *B* avec une majuscule dans les notations) est une unité d'information composée de 8 bits. Il permet par exemple de stocker un caractère, tel qu'une lettre ou un chiffre.

Ce regroupement de nombres par série de 8 permet une lisibilité plus grande, au même titre que l'on apprécie, en base décimale, de regrouper les nombres par trois pour pouvoir distinguer les milliers. Le nombre « 1 256 245 » est par exemple plus lisible que « 1256245 ».

Une unité d'information composée de 16 bits est généralement appelée **mot** (en anglais *word*).

Une unité d'information de 32 bits de longueur est appelée **mot double** (en anglais *double word*, d'où l'appellation *dword*).

Pour un octet, le plus petit nombre est 0 (représenté par huit zéros 00000000), et le plus grand est 255 (représenté par huit chiffres « un » 11111111), ce qui représente 256 possibilités de valeurs différentes.

$2^7 = 128$	$2^6 = 64$	$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$
0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1

KiloOctets, MégaOctets

Longtemps l'informatique s'est singularisée par l'utilisation de différentes valeurs pour les unités du système international. Ainsi beaucoup d'informaticiens ont appris que 1 kilooctet valait 1024 octets. Or, depuis décembre 1998, l'organisme international *IEC* a statué sur la question (<http://physics.nist.gov/cuu/Units/binary.html>). Voici donc les unités standardisées :

- Un kilooctet (ko ou kB) = 1000 octets
- Un Mégaoctet (Mo ou MB) = 1 000 ko = 1 000 000 octets
- Un Gigaoctet (Go ou GB) = 1 000 Mo = 1 000 000 000 octets
- Un Téraoctet (To) = 1 000 Go = 1 000 000 000 000 octets

Attention ! De nombreux logiciels (parfois même certains systèmes d'exploitation) utilisent toujours la notation antérieure à 1998 pour laquelle :

- Un kilooctet (ko) = 2^{10} octets = 1024 octets
Un Mégaoctet (Mo) = 2^{20} octets = 1 024 ko = 1 048 576 octets
Un Gigaoctet (Go) = 2^{30} octets = 1 024 Mo = 1 073 741 824 octets
Un Téraoctet (To) = 2^{40} octets = 1 024 Go = 1 099 511 627 776 octets



L'IEC a également défini le kilo binaire (kibi), le méga binaire (Mébi), le giga binaire (Gibi), le tera binaire (Tebi).

Voici leurs définitions :

- Un kibioctet (kio ou kiB) vaut $2^{10} = 1024$ octets
- Un Mébioroctet (Mio ou MiB) vaut $2^{20} = 1 048 576$ octets
- Un Gibioctet (Gio ou GiB) vaut $2^{30} = 1 073 741 824$ octets
- Un Tébioroctet (Tio ou TiB) vaut $2^{40} = 1 099 511 627 776$ octets

Il est également utile de noter que la communauté internationale dans son ensemble utilise préférentiellement le nom de « byte » plutôt que le terme « octet » purement francophone. Cela donne les notations suivantes pour kilobyte, mégabyte, gigabyte et terabyte : *kB, MB, GB, TB*



Notez l'utilisation d'un *B* majuscule pour différencier *Byte* et *bit*.

Opérations en binaire

Les opérations arithmétiques simples telles que l'addition, la soustraction et la multiplication sont faciles à effectuer en binaire.

Addition binaire

L'addition en binaire se fait avec les mêmes règles qu'en décimale :

On commence à additionner les bits de poids faible (les bits de droite) puis on a des retenues lorsque la somme de deux bits de même poids dépasse la valeur de l'unité la plus grande (dans le cas du binaire : 1), cette retenue est reportée sur le bit de poids plus fort suivant...

Par exemple :

$$\begin{array}{r} 01101 \\ +01110 \\ \hline 11011 \end{array}$$

Multiplication binaire

La table de multiplication en binaire est très simple :

- $0 \times 0 = 0$
- $0 \times 1 = 0$
- $1 \times 0 = 0$
- $1 \times 1 = 1$

La multiplication se fait en formant un produit partiel pour chaque digit du multiplicateur (seuls les bits non nuls donneront un résultat non nul). Lorsque le bit du multiplicateur est nul, le produit partiel est nul, lorsqu'il vaut un, le produit partiel est constitué du multiplicande décalé du nombre de positions égal au poids du bit du multiplicateur.

Par exemple :

0101 multiplicande

x 0010 multiplicateur

0000
0101
0000

01010

Système hexadécimal

Les nombres binaires étant de plus en plus longs, il a fallu introduire une nouvelle base : la **base hexadécimale**.

La base hexadécimale consiste à compter sur une base 16, c'est pourquoi au-delà des 10 premiers chiffres on a décidé d'ajouter les 6 premières lettres : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Base décimale	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Base hexadécimale	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Base binaire	000	000	001	001	010	010	011	011	100	100	101	101	110	110	111	111

Un exemple

Le nombre 27 (en base 10) vaut en base 16 : $1*16^1 + 11*16^0 = 1*16^1 + B*16^0$
c'est-à-dire 1B en base 16.

Le nombre FB3 (en base 16) vaut en base 10 : $F*16^2 + B*16^1 + 3*16^0 = 3840 + 176 + 3 = 4019$

Pour convertir un octet en hexadécimale, on le partage en 2 groupes de 4 bits, qui correspondent chacun à un chiffre hexadécimal.

2	A	D	5
0010	1010	1101	0101

Représentation d'un nombre dans un ordinateur

On appelle représentation (ou codification) d'un nombre la façon selon laquelle il est décrit sous forme binaire. La représentation des nombres sur un ordinateur est indispensable pour que celui-ci puisse les stocker, les manipuler. Toutefois le problème est qu'un nombre mathématique peut être infini (aussi grand que l'on veut), mais la représentation d'un nombre dans un ordinateur doit être faite sur un nombre de bits prédéfini. Il s'agit donc de prédefinir un nombre de bits et la manière de les utiliser pour que ceux-ci servent le plus efficacement possible à représenter l'entité. Ainsi il serait idiot de coder un caractère sur 16 bits (65536 possibilités) alors qu'on en utilise généralement moins de 256...

Représentation d'un entier naturel

Un entier naturel est un entier positif ou nul. Le choix à faire (c'est-à-dire le nombre de bits à utiliser) dépend de la fourchette des nombres que l'on désire utiliser. Pour coder des nombres entiers naturels compris entre 0 et 255, il nous suffira de 8 bits (un octet) car $2^8=256$. D'une manière générale un codage sur n bits pourra permettre de représenter des nombres entiers naturels compris entre 0 et 2^n-1 .

Pour représenter un nombre entier naturel après avoir défini le nombre de bits sur lequel on le code, il suffit de ranger chaque bit dans la cellule binaire correspondant à son poids binaire de la droite vers la gauche, puis on « remplit » les bits non utilisés par des zéros.

Représentation d'un entier relatif

Un entier relatif est un entier pouvant être négatif. Il faut donc coder le nombre de telle façon que l'on puisse savoir s'il s'agit d'un nombre positif ou d'un nombre négatif, et il faut de plus que les règles d'addition soient conservées. L'astuce consiste à utiliser un codage que l'on appelle *complément à deux*.

- **un entier relatif positif ou nul** sera représenté en binaire (base 2) comme un entier naturel, à la seule différence que le bit de poids fort (le bit situé à l'extrême gauche) représente le signe. Il faut donc s'assurer pour un entier positif ou nul qu'il est à zéro (0 correspond à un signe positif, 1 à un signe négatif). Ainsi si on code un entier naturel sur 4 bits, le nombre le plus grand sera 0111 (c'est-à-dire 7 en base décimale).

D'une manière générale le plus grand entier relatif positif codé sur n bits sera $2^{n-1}-1$.

- **un entier relatif négatif** grâce au codage en complément à deux.

Principe du *complément à deux* :

Soit à représenter un nombre négatif.

- Prenons son opposé (son équivalent en positif)
- On le représente en base 2 sur $n-1$ bits
- On complémentise chaque bit (on inverse, c'est-à-dire que l'on remplace les zéros par des 1 et vice-versa)
- On ajoute 1

On remarquera qu'en ajoutant le nombre et son complément à deux on obtient 0...

Voyons maintenant cela sur un exemple :

On désire coder la valeur -5 sur 8 bits. Il suffit :

- d'écrire 5 en binaire : 00000101
- de complémenter à 1 : 11111010
- d'ajouter 1 : 11111011
- la représentation binaire de -5 sur 8 bits est 11111011

Remarques:

Le bit de poids fort est 1, on a donc bien un nombre négatif.

Si on ajoute 5 et -5 (00000101 et 11111011) on obtient 0 (avec une retenue de 1...).

Représentation d'un nombre réel

Il s'agit d'aller représenter un nombre binaire à virgule (par exemple 101,01 qui ne se lit pas *cent un virgule zéro un* puisque c'est un nombre binaire mais 5,25 en décimale) sous la forme 1,XXXX... * 2^n (c'est-à-dire dans notre exemple 1,0101 * 2^2). La norme *IEEE* définit la façon de coder un nombre réel.

Cette norme se propose de coder le nombre sur 32 bits et définit trois composantes :

- le signe est représenté par un seul bit, le bit de poids fort (celui le plus à gauche)

- l'exposant est codé sur les 8 bits consécutifs au signe
- la mantisse (les bits situés après la virgule) sur les 23 bits restants

Ainsi le codage se fait sous la forme suivante :

seeeeeeeeemmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm

- le **s** représente le bit relatif au signe
- les **e** représentent les bits relatifs à l'exposant
- les **m** représentent les bits relatifs à la mantisse

Certaines conditions sont toutefois à respecter pour les exposants :

- l'exposant 00000000 est interdit
- l'exposant 11111111 est interdit. On s'en sert toutefois pour signaler des erreurs, on appelle alors cette configuration du nombre *Nan*, ce qui signifie *Not a number*
- Il faut rajouter 127 (01111111) à l'exposant pour une conversion de décimal vers un nombre réel binaire. Les exposants peuvent ainsi aller de -254 à 255

La formule d'expression des nombres réels est ainsi la suivante:

$$(-1)^S * 2^{(E - 127)} * (1 + F)$$

où:

- S est le bit de signe et l'on comprend alors pourquoi 0 est positif ($-1^{0}=1$).
- E est l'exposant auquel on doit bien ajouter 127 pour obtenir son équivalent codé.
- F est la partie fractionnaire, la seule que l'on exprime et qui est ajoutée à 1 pour effectuer le calcul.

Voyons ce codage sur un exemple :

Soit à coder la valeur 525,5.

- 525,5 est positif donc le 1er bit sera 0.
- Sa représentation en base 2 est la suivante : 1000001101,1
- En normalisant, on trouve : 1,0000011011* 2^9
- On ajoute 127 à l'exposant qui vaut 9 ce qui donne 136, soit en base 2 : 10001000
- La mantisse est composée de la partie décimale de 525,5 en base 2 normalisée, c'est-à-dire 0000011011.
- Comme la mantisse doit occuper 23 bits, il est nécessaire d'ajouter des zéros pour la compléter :

0000011011000000000000000

- La représentation du nombre 525,5 en binaire avec la norme IEEE est donc :

0 1000 1000 00001101100000000000000

0100 0100 0000 0011 0110 0000 0000 (4403600 en hexadécimal)

Voici un autre exemple avec un réel négatif :

Soit à coder la valeur -0,625.

- Le bit s vaut 1 car 0,625 est négatif
- 0,625 s'écrit en base 2 de la façon suivante : 0,101
- On souhaite l'écrire sous la forme 1.01×2^{-1}
- Par conséquent l'exposant vaut 1111110 car $127 - 1 = 126$ (soit 1111110 en binaire)
- la mantisse est 01000000000000000000000000000000 (seuls les chiffres après la virgule sont représentés, le nombre entier étant toujours égal à 1)

- La représentation du nombre 0,625 en binaire avec la norme IEEE est :

1 1111 1110 01000000000000000000000000000000

1111 1111 0010 0000 0000 0000 0000 0000 (FF 20 00 00 en hexadécimal)

Le codage des informations

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 codage binaire en quelque sorte...). Il permit d'effectuer des communications beaucoup plus rapides que ne le permettait le système de courrier de l'époque aux Etats-Unis : le Pony Express. L'interpréteur était l'homme à l'époque, il fallait donc 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 en revanche *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. Pour l'anecdote, la Chambre des représentants a décidé que l'invention du téléphone revenait à Antonio Meucci. Ce dernier avait en effet déposé une demande de brevet en 1871, mais n'avait pas pu financer celle-ci au-delà de 1874.

Ces lignes permirent l'essor des téléscripteurs, 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.

Qu'est-ce que le code ASCII ?

La mémoire de l'ordinateur conserve toutes les données sous forme numérique. Il n'existe pas de méthode pour stocker directement les caractères. Chaque caractère possède donc son équivalent en code numérique : c'est le **code ASCII** (*American Standard Code for Information Interchange* - traduisez « Code Americain Standard pour l'Echange d'Informations »). Le code ASCII de base représentait les caractères sur 7 bits (c'est-à-dire 128 caractères possibles, de 0 à 127).

- Les codes 0 à 31 ne sont pas des caractères. On les appelle *caractères de contrôle* car ils permettent de faire des actions telles que :
 - retour à la ligne (CR)
 - Bip sonore (BEL)
- Les codes 65 à 90 représentent les majuscules
- Les codes 97 à 122 représentent les minuscules

(Il suffit de modifier le 6^{ème} bit pour passer de majuscules à minuscules, c'est-à-dire ajouter 32 au code ASCII en base décimale.)

Table des caractères ASCII

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STH	ETH	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	CD2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	!	"	#	\$	%	&	'	()	*	+	,	-	.	/	
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	-
6	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	}	~	DEL	
8	€	□	,	f	„	…	†	‡	^	%o	Š	<	Œ	□	Ž	□
9	□	'	'	"	"	▪	—	—	~	TM	š	>	œ	□	ž	Ý
A	ı	φ	£	*	¥	!	§	“	”	©	¤	«	”	-	®	-
B	°	±	²	³	·	µ	¶	·	·	¹	º	»	¼	½	¾	½
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	Þ
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
F	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

Table des caractères ASCII Étendue

Le code ASCII a été mis au point pour la langue anglaise, il ne contient donc pas de caractères accentués, ni de caractères spécifiques à une langue. Pour coder ce type de caractère il faut recourir à un autre code. Le code ASCII a donc été étendu à 8 bits (un octet) pour pouvoir coder plus de caractères (on parle d'ailleurs de code ASCII étendu...).

Ce code attribue les valeurs 0 à 255 (donc codées sur 8 bits, soit 1 octet) aux lettres majuscules et minuscules, aux chiffres, aux marques de ponctuation et aux autres symboles (caractères accentués dans le cas du code *iso-latin1*).

Le code ASCII étendu n'est pas unique et dépend fortement de la plateforme utilisée.

Les deux jeux de caractères ASCII étendus les plus couramment utilisés sont :

- Le code ASCII étendu OEM, c'est-à-dire celui qui équipait les premières machines de type IBM PC

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	ç	ü	é	â	ä	à	å	ç	ê	ë	è	í	î	ì	ã	ß
9	É	æ	Œ	ô	ö	ò	û	ÿ	ö	ü	ç	£	¥	®	ƒ	
A	á	í	ó	ú	ñ	Ñ	œ	ç	‐	‐	½	¾	í	«	»	
B	▀	▀	▀	▀	▀	▀	▀	▀	▀	▀	▀	▀	▀	▀	▀	▀
C	└	└	└	└	‐	‐	‐	‐	‐	‐	‐	‐	‐	‐	‐	‐
D	„	„	„	„	„	„	„	„	„	„	„	„	„	„	„	„
E	α	β	Γ	Π	Σ	σ	μ	τ	δ	θ	Ω	δ	ω	ø	€	ø
F	≡	±	≥	≤	ƒ	J	÷	≈	°	-	-	√	n	z	█	

- Le code ASCII étendu ANSI, utilisé par les systèmes d'exploitation récents

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	□	□	,	f	„	…	†	‡	^	‰	ſ	⟨	Œ	□	□	□
9	□	‘	’	„	„	▪	—	—	“	„	ſ	⟩	œ	□	□	ÿ
A	ı	ö	£	¤	¥	ı	ſ	“	©	™	«	¬	—	®	—	—
B	°	±	²	³	‘	μ	¶	·	·	°	»	¼	½	¾	¸	¸
C	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Í	Í
D	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	Þ
E	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	í	í
F	ë	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ÿ	þ	ÿ

Le code EBCDIC

Le code *EBCDIC* (*Extended Binary-Coded Decimal Interchange Code*), développé par IBM, permet de coder des caractères sur 8 bits. Bien que largement répandu sur les machines IBM, il n'a pas eu le succès qu'a connu le code ASCII.

Unicode

Le code *Unicode* est un système de codage des caractères sur 16 bits mis au point en 1991. Le système Unicode permet de représenter n'importe quel caractère par un code sur 16 bits, indépendamment de tout système d'exploitation ou langage de programmation.

Il regroupe ainsi la quasi-totalité des alphabets existants (arabe, arménien, cyrillique, grec, hébreu, latin, ...) et est compatible avec le code ASCII.

L'ensemble des codes Unicode est disponible sur le site <http://www.unicode.org>.

Le codage Base64

Le principe du codage **Base 64** consiste à utiliser des caractères US-ASCII (caractères non accentués) pour coder tout type de données codé sur 8 bits.

Les protocoles de courrier électronique ont en effet été prévus à l'origine pour transporter des messages en texte seulement. Or, étant donné la diversité des systèmes de courrier électronique, l'échange de données binaires se traduit la plupart du temps par des transformations du contenu rendant illisible le document original.

Le format Base64, utilisé massivement dans les échanges de courrier électronique, permet ainsi de transmettre n'importe quel document binaire (application, vidéo, fichier audio, etc.) en pièce jointe d'un courrier électronique en les codant à l'aide de caractères classiques.

Le codage Base64 provoque une augmentation globale de 33% du volume des données à encoder.

Principe de codage Base64

Le principe du codage Base64 consiste à utiliser 4 caractères imprimables (au format US-ASCII) pour coder un groupe de 3 octets quelconques (3×8 bits = 24 bits).

Le codage Base64 utilise un alphabet de 64 caractères imprimables classiques pour représenter une donnée de 6 bits. Les 64 symboles de cet alphabet sont choisis pour être universellement lisibles et pour ne pas posséder de signification dans les principaux protocoles de messagerie (en particulier SMTP).

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

123456789+/.

En parcourant les données binaires de gauche à droite, des groupes de 24 bits sont créés en concaténant des blocs de 3 données de 8 bits. Chaque groupe de 24 bits est ensuite divisé en 4 groupes de 6 bits, correspondant à 4 caractères de l'alphabet Base64.

L'encodage Base64 est prévu pour des données formant un multiple de 24 bits. Ainsi, si le volume des données à coder ne forment pas un multiple de 24 bits, le résultat du codage Base64 doit être complété par 0 à 3 caractères « = » afin d'obtenir un multiple de 24 bits. Ce 65ème caractère ne peut ainsi être présent qu'à la fin des données encodées.

Par ailleurs, afin de garantir une compatibilité avec l'ensemble des systèmes de messagerie, les données Base64 sont formatées avec des retours à la ligne afin que chaque ligne ne dépasse pas 76 caractères.

Qu'est-ce qu'un fichier?

Un **fichier** est une suite d'informations binaires, c'est-à-dire une suite de 0 et de 1. Ce fichier peut être stocké pour garder une trace de ces informations. Un fichier texte est un fichier composé de caractères stockés sous la forme d'octets.

Ce fichier est enregistré sur le disque dur sous la forme "**nom_du_fichier.ext**".

".ext" représente l'extension c'est un moyen de reconnaître le type de programme avec lequel ce fichier peut être ouvert (**attention** cela ne garantit pas le type de fichier: lorsque l'on change l'extension on ne change pas le type de fichier!).

La longueur du nom et de l'extension peut varier suivant le système d'exploitation :

- 8 caractères pour le nom et 3 pour l'extension sous DOS et Windows 3.1
- 256 caractères pour le nom et l'extension sous Windows 95, 98 et NT
- 256 sous les systèmes Unix

Ainsi, sous DOS ou Windows 3.1, un fichier provenant de Windows 9x aura un nom tronqué comportant les 6 premiers caractères du nom suivi de ~x où x représente un chiffre qui est incrémenté à chaque fois qu'un fichier porte le même nom. C'est-à-dire que si un fichier nommé "fichie~1" existe déjà il nommera le suivant "fichie~2".

De plus, un fichier contient un en-tête qui permet de stocker des informations supplémentaires, comme le type de fichier et surtout la taille du fichier. Il contient aussi un caractère de fin de fichier signalant que les informations situées au-delà de ce caractère ne font plus partie du même fichier.

Quoi de plus idiot que de mettre dans l'en-tête du fichier la taille du fichier puisqu'on la connaît me direz-vous?

Voici deux exemples qui vous démontreront son utilité

Les fichiers corrompus

Il vous est forcément déjà arrivé de télécharger un fichier sur Internet et que le navigateur plante ou bien que le serveur qui héberge ce fichier coupe la communication. Si ce fichier est un fichier texte, il ne vous manquera que la fin du texte, par contre si celui-ci est un fichier binaire (un programme exécutable par exemple) son exécution pourrait très bien être dangereuse car il manque des informations. Le système d'exploitation compare donc sa taille réelle à la taille indiquée dans l'en-tête pour vérifier la validité du fichier. On parle généralement d'intégrité. En réalité ce contrôle est réalisé à l'aide d'un algorithme plus performant appelé CRC (contrôle de redondance cyclique).

Infection par un virus

Lorsqu'un fichier est infecté par un virus, ce dernier y ajoute des lignes de code. Ainsi, l'information concernant la taille du fichier située dans l'en-tête ne correspondra plus (à moins que le virus ne soit programmé de manière à modifier l'en-tête), il pourra donc être repéré.

Qu'est-ce qu'un répertoire?

Un **répertoire** (appelé également *dossier* ou *folder* en anglais) est un objet informatique pouvant contenir des fichiers.

Imaginez une grande commode qui contient des tiroirs dans lesquels pourraient se trouver des fichiers ainsi que d'autres tiroirs. Un répertoire peut en effet contenir :

- des fichiers ;
- d'autres répertoires.

Relations relatives

Si l'on reprend notre exemple de la commode, la plus grande entité contenant d'autres entités est la commode : elle ne peut pas se trouver dans un tiroir !

Dans le cas de l'informatique, on appelle cette entité **le répertoire racine** (appelé parfois tout simplement « *racine* » en anglais *root directory*) : il s'agit de l'entité de plus bas niveau, car elle peut contenir des fichiers ou des répertoire mais ne peut pas se trouver elle-même dans un répertoire !

On la note "\\" (dans le monde Windows) ou "/" (dans le monde UNIX / Linux). La racine est unique sous les système UNIX et il en existe une par partition sous les systèmes Microsoft Windows.

Un répertoire qui en contient un autre est dit "répertoire parent". Lorsque d'un répertoire on veut aller au répertoire parent, celui-ci est désigné par ".." sur la plupart des systèmes (on tapera donc "cd .." sous DOS ou sous UNIX pour accéder à un répertoire parent).

Voyons la représentation d'un système de répertoires sous Windows 95:



Dans cet exemple *répertoire2* est parent de *répertoire20* et de *répertoire21*. "*répertoire2*" vis-à-vis de "*répertoire20*" sera ainsi notée "...". La racine (d:\) vis-à-vis de "*répertoire20*" sera quant à elle notée "..\.." car deux relations de parenté les séparent.

Notion de chemin

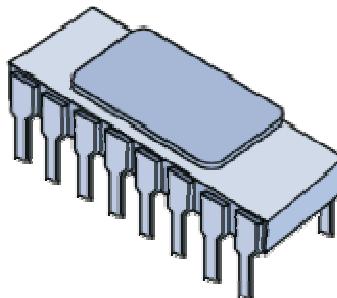
On appelle «*chemin*» (en anglais *path*) la succession des répertoires en partant de la racine pour atteindre un fichier. Sous les systèmes Windows un chemin sera de la forme *x:\repertoire1\repertoire2* tandis que sous un système Unix il sera noté */repertoire1/repertoire2/*.

C. Le microprocesseur

Présentation

Le **processeur** (CPU, pour *Central Processing Unit*, soit *Unité Centrale de Traitement*) est le cerveau de l'ordinateur. Il permet de manipuler des informations numériques, c'est-à-dire des informations codées sous forme binaire, et d'exécuter les instructions stockées en mémoire.

Le premier **microprocesseur** (Intel 4004) a été inventé en 1971. Il s'agissait d'une unité de calcul de 4 bits, cadencé à 108 kHz. Depuis, la puissance des microprocesseurs augmente exponentiellement. Quels sont donc ces petits morceaux de silicium qui dirigent nos ordinateurs?



Fonctionnement

Le **processeur** (noté **CPU**, pour *Central Processing Unit*) est un circuit électronique cadencé au rythme d'une horloge interne, grâce à un cristal de quartz qui, soumis à un courant électrique, envoie des impulsions, appelées « **top** ». La **fréquence d'horloge** (appelée également **cycle**, correspondant au nombre d'impulsions par seconde, s'exprime en Hertz (Hz). Ainsi, un ordinateur à 200 MHz possède une horloge envoyant 200 000 000 de battements par seconde. La fréquence d'horloge est généralement un multiple de la fréquence du système (*FSB, Front-Side Bus*), c'est-à-dire un multiple de la fréquence de la carte mère

A chaque top d'horloge le processeur exécute une action, correspondant à une instruction ou une partie d'instruction. L'indicateur appelé **CPI** (*Cycles Par Instruction*) permet de représenter le nombre moyen de cycles d'horloge nécessaire à l'exécution d'une instruction sur un microprocesseur. La puissance du processeur peut ainsi être caractérisée par le nombre d'instructions qu'il est capable de traiter par seconde. L'unité utilisée est le **MIPS** (*Millions d'Instructions Par Seconde*) correspondant à la fréquence du processeur que divise le **CPI**.

Instruction

Une **instruction** est l'opération élémentaire que le processeur peut accomplir. Les instructions sont stockées dans la mémoire principale, en vue d'être traitée par le processeur. Une instruction est composée de deux champs :

- le **code opération**, représentant l'action que le processeur doit accomplir ;
- le **code opérande**, définissant les paramètres de l'action. Le code opérande dépend de l'opération. Il peut s'agir d'une donnée ou bien d'une adresse mémoire.

Code opération	Champ opérande
----------------	----------------

Le nombre d'octets d'une instruction est variable selon le type de donnée (l'ordre de grandeur est de 1 à 4 octets).

Les instructions peuvent être classées en catégories dont les principales sont :

- **Accès à la mémoire** : des accès à la mémoire ou transferts de données entre registres.
- **Opérations arithmétiques** : opérations telles que les additions, soustractions, divisions ou multiplication.
- **Opérations logiques** : opérations ET, OU, NON, NON exclusif, etc.
- **Contrôle** : contrôles de séquence, branchements conditionnels, etc.

Registres

Lorsque le processeur exécute des instructions, les données sont temporairement stockées dans de petites mémoires rapides de 8, 16, 32 ou 64 bits que l'on appelle **registres**. Suivant le type de processeur le nombre global de registres peut varier d'une dizaine à plusieurs centaines.

Les registres principaux sont :

- **le registre accumulateur (ACC)**, stockant les résultats des opérations arithmétiques et logiques ;
- **le registre d'état (PSW, Processor Status Word)**, permettant de stocker des indicateurs sur l'état du système (retenue, dépassement, etc.) ;
- **le registre instruction (RI)**, contenant l'instruction en cours de traitement ;
- **le compteur ordinal (CO ou PC pour Program Counter)**, contenant l'adresse de la prochaine instruction à traiter ;
- **le registre tampon**, stockant temporairement une donnée provenant de la mémoire.

Mémoire cache

La **mémoire cache** (également appelée *antémémoire* ou *mémoire tampon*) est une mémoire rapide permettant de réduire les délais d'attente des informations stockées en mémoire vive. En effet, la mémoire centrale de l'ordinateur possède une vitesse bien moins importante que le processeur. Il existe néanmoins des mémoires beaucoup plus rapides, mais dont le coût est très élevé. La solution consiste donc à inclure ce type de mémoire rapide à proximité du processeur et d'y stocker temporairement les principales données devant être traitées par le processeur. Les ordinateurs récents possèdent plusieurs niveaux de mémoire cache :

- **La mémoire cache de premier niveau** (appelée **L1 Cache**, pour **Level 1 Cache**) est directement intégrée dans le processeur. Elle se subdivise en 2 parties :
 - La première est le cache d'instructions, qui contient les instructions issues de la mémoire vive décodées lors de passage dans les pipelines.
 - La seconde est le cache de données, qui contient des données issues de la mémoire vive et les données récemment utilisées lors des opérations du processeur.

Les caches du premier niveau sont très rapides d'accès. Leur délai d'accès tend à s'approcher de celui des registres internes aux processeurs.

- La **mémoire cache de second niveau** (appelée **L2 Cache**, pour **Level 2 Cache**) est située au niveau du boîtier contenant le processeur (dans la puce). Le cache de second niveau vient s'intercaler entre le processeur avec son cache interne et la mémoire vive. Il est plus rapide d'accès que cette dernière mais moins rapide que le cache de premier niveau.
- La **mémoire cache de troisième niveau** (appelée **L3 Cache**, pour **Level 3 Cache**) est située au niveau de la carte mère.

Tous ces niveaux de cache permettent de réduire les temps de latence des différentes mémoires lors du traitement et du transfert des informations. Pendant que le processeur travaille, le contrôleur de cache de premier niveau peut s'interfacer avec celui de second niveau pour faire des transferts d'informations sans bloquer le processeur. De même, le cache de second niveau est interfacé avec celui de la mémoire vive(cache de troisième niveau), pour permettre des transferts sans bloquer le fonctionnement normal du processeur.

Signaux de commande

Les **signaux de commande** sont des signaux électriques permettant d'orchestrer les différentes unités du processeur participant à l'exécution d'une instruction. Les signaux de commandes sont distribués grâce à un élément appelé *séquenceur*. Le signal *Read / Write*, en français *lecture / écriture*, permet par exemple de signaler à la mémoire que le processeur désire lire ou écrire une information.

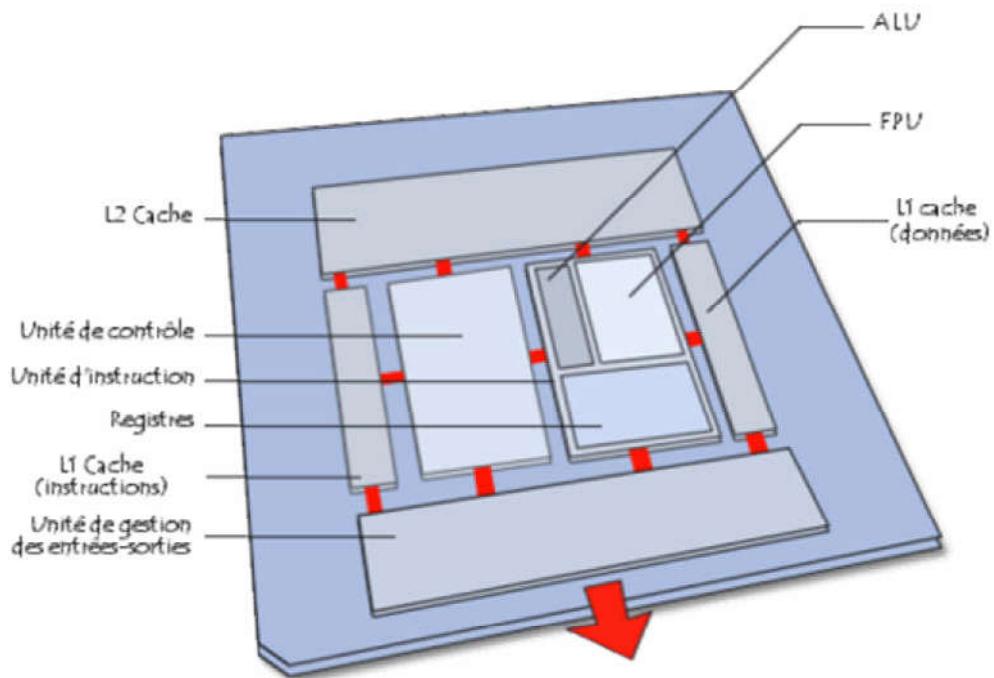
Unités fonctionnelles

Le processeur est constitué d'un ensemble d'unités fonctionnelles reliées entre elles. L'architecture d'un microprocesseur est très variable d'une architecture à une autre, cependant les principaux éléments d'un microprocesseur sont les suivants :

- Une **unité d'instruction** (ou *unité de commande*, en anglais *control unit*) qui lit les données arrivant, les décode puis les envoie à l'unité d'exécution ; L'unité d'instruction est notamment constituée des éléments suivants :
 - **séquenceur** (ou *bloc logique de commande*) chargé de synchroniser l'exécution des instructions au rythme d'une horloge. Il est ainsi chargé de l'envoi des signaux de commande ;
 - **compteur ordinal** contenant l'adresse de l'instruction en cours ;
 - **registre d'instruction** contenant l'instruction suivante.
- Une **unité d'exécution** (ou *unité de traitement*), qui accomplit les tâches que lui a données l'unité d'instruction. L'unité d'exécution est notamment composée des éléments suivants :
 - **L'unité arithmétique et logique** (notée **UAL** ou en anglais **ALU** pour *Arithmetical and Logical Unit*). L'UAL assure les fonctions basiques de calcul arithmétique et les opérations logiques (ET, OU, Ou exclusif, etc.) ;
 - **L'unité de virgule flottante** (notée **FPU**, pour *Floating Point Unit*), qui accomplit les calculs complexes non entiers que ne peut réaliser l'unité arithmétique et logique.

- Le **registre d'état** ;
- Le **registre accumulateur**.
- Une **unité de gestion des bus** (ou *unité d'entrées-sorties*), qui gère les flux d'informations entrant et sortant, en interface avec la mémoire vive du système ;

Le schéma ci-dessous donne une représentation simplifiée des éléments constituant le processeur (l'organisation physique des éléments ne correspond pas à la réalité) :



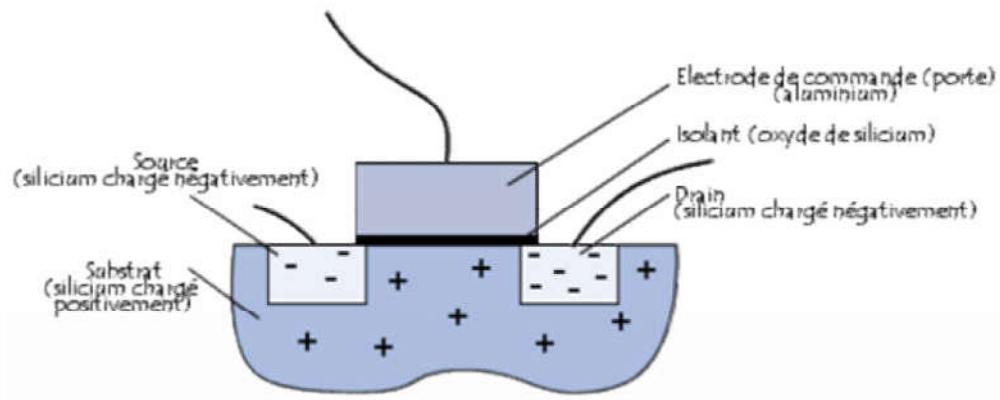
Transistor

Pour effectuer le traitement de l'information, le microprocesseur possède un ensemble d'instructions, appelé « **jeu d'instructions** », réalisées grâce à des circuits électroniques. Plus exactement, le jeu d'instructions est réalisé à l'aide de semi-conducteurs, « petits interrupteurs » utilisant l'**effet transistor**, découvert en 1947 par *John Barden, Walter H. Brattain et William Shockley* qui reçurent le prix Nobel en 1956 pour cette découverte.

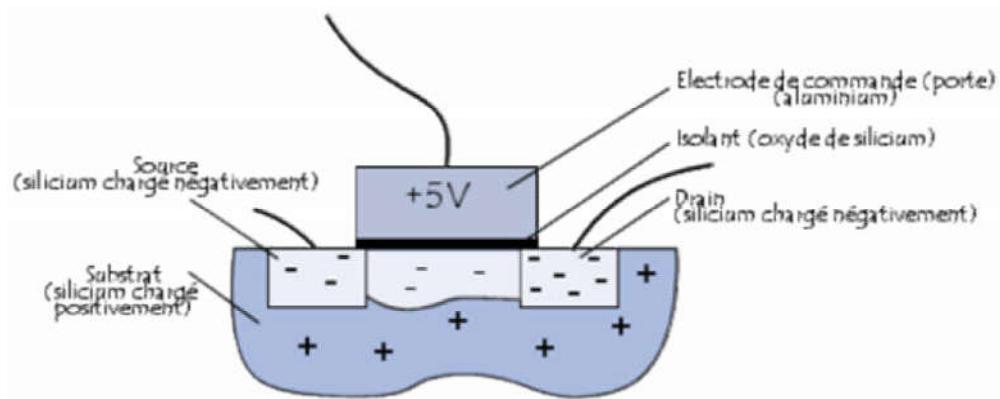
Un **transistor** (contraction de *transfer resistor*, en français *résistance de transfert*) est un composant électronique semi-conducteur, possédant trois électrodes, capable de modifier le courant qui le traverse à l'aide d'une de ses électrodes (appelée électrode de commande). On parle ainsi de « composant actif », par opposition aux « composants passifs », tels que la résistance ou le condensateur, ne possédant que deux électrodes (on parle de « bipolaire »).

Le transistor MOS (*métal, oxyde, silicium*) est le type de transistor majoritairement utilisé pour la conception de circuits intégrés. Le transistor MOS est composé de deux zones

chargées négativement, appelées respectivement **source** (possédant un potentiel quasi-nul) et **drain** (possédant un potentiel de 5V), séparées par une région chargée positivement, appelée **substrat** (en anglais *substrate*). Le substrat est surmonté d'une électrode de commande, appelée **porte** (en anglais *gate*, parfois également appelée *grille*), permettant d'appliquer une tension sur le substrat.



Lorsqu'aucune tension n'est appliquée à l'électrode de commande, le substrat chargé positivement agit telle une barrière et empêche les électrons d'aller de la source vers le drain. En revanche, lorsqu'une tension est appliquée à la porte, les charges positives du substrat sont repoussées et il s'établit un canal de communication, chargé négativement, reliant la source au drain.



Le transistor agit donc globalement comme un interrupteur programmable grâce à l'électrode de commande. Lorsqu'une tension est appliquée à l'électrode de commande, il agit comme un interrupteur fermé, dans le cas contraire comme un interrupteur ouvert.

Circuits intégrés

Assemblés, les transistors peuvent constituer des circuits logiques, qui, assemblés à leur tour, constituent des processeurs. Le premier circuit intégré date de 1958 et a été mis au point par la société *Texas Instruments*.

Les transistors MOS sont ainsi réalisés dans des tranches de silicium (appelées *wafer*, traduisez *gaufres*), obtenues après des traitements successifs. Ces tranches de silicium sont alors découpées en éléments rectangulaires, constituant ce que l'on appelle un « **circuit** ». Les circuits sont ensuite placés dans des boîtiers comportant des connecteurs d'entrée-sortie, le tout constituant un « **circuit intégré** ». La finesse de la gravure, exprimée en microns (micromètres, notés μm), définit le nombre de transistors par unité de surface. Il peut ainsi exister jusqu'à plusieurs millions de transistors sur un seul processeur.

La **loi de Moore**, édictée en 1965 par Gordon E. Moore, cofondateur de la société Intel, prévoyait que les performances des processeurs (par extension le nombre de transistors intégrés sur silicium) doubleraient tous les 12 mois. Cette loi a été révisée en 1975, portant le nombre de mois à 18. La loi de Moore se vérifie encore aujourd'hui.

Dans la mesure où le boîtier rectangulaire possède des broches d'entrée-sortie ressemblant à des pattes, le terme de « **puce électronique** » est couramment employé pour désigner les circuits intégrés.

Familles

Chaque type de processeur possède son propre jeu d'instruction. On distingue ainsi les familles de processeurs suivants, possédant chacun un jeu d'instruction qui leur est propre :

- 80x86 : le « x » représente la famille. On parle ainsi de 386, 486, 586, 686, etc.
- ARM
- IA-64
- MIPS
- Motorola 6800
- PowerPC
- SPARC
- ...

Cela explique qu'un programme réalisé pour un type de processeur ne puisse fonctionner directement sur un système possédant un autre type de processeur, à moins d'une traduction des instructions, appelée

émulation. Le terme « **émulateur** » est utilisé pour désigner le programme réalisant cette traduction.

Jeu d'instruction

On appelle **jeu d'instructions** l'ensemble des opérations élémentaires qu'un processeur peut accomplir. Le jeu d'instruction d'un processeur détermine ainsi son architecture, sachant qu'une même architecture peut aboutir à des implémentations différentes selon les constructeurs.

Le processeur travaille effectivement grâce à un nombre limité de fonctions, directement câblées sur les circuits électroniques. La plupart des opérations peuvent être réalisées à l'aide de fonctions basiques. Certaines architectures incluent néanmoins des fonctions évoluées courante dans le processeur.

Architecture CISC

L'architecture **CISC** (*Complex Instruction Set Computer*, soit « *ordinateur à jeu d'instruction complexe* ») consiste à câbler dans le processeur des instructions complexes, difficiles à créer à partir des instructions de base.

L'architecture **CISC** est utilisée en particulier par les processeurs de type 80x86. Ce type d'architecture possède un coût élevé dû aux fonctions évoluées imprimées sur le silicium.

D'autre part, les instructions sont de longueurs variables et peuvent parfois nécessiter plus d'un cycle d'horloge. Or, un processeur basé sur l'architecture CISC ne peut traiter qu'une instruction à la fois, d'où un temps d'exécution conséquent.

Architecture RISC

Un processeur utilisant la technologie **RISC** (*Reduced Instruction Set Computer*, soit « *ordinateur à jeu d'instructions réduit* ») n'a pas de fonctions évoluées câblées.

Les programmes doivent ainsi être traduits en instructions simples, ce qui entraîne un développement plus difficile et/ou un compilateur plus puissant. Une telle architecture possède un coût de fabrication réduit par rapport aux processeurs CISC. De plus, les instructions, simples par nature, sont exécutées en un seul cycle d'horloge, ce qui rend l'exécution des programmes plus rapide qu'avec des processeurs basés sur une architecture CISC. Enfin, de tels processeurs sont capables de traiter plusieurs instructions simultanément en les traitant en parallèle.

Améliorations technologiques

Au cours des années, les constructeurs de microprocesseurs (appelés *fondeurs*), ont mis au point un certain nombre d'améliorations permettant d'optimiser le fonctionnement du processeur.

Le parallélisme

Le **parallélisme** consiste à exécuter simultanément, sur des processeurs différents, des instructions relatives à un même programme. Cela se traduit par le découpage d'un programme en plusieurs processus traités en parallèle afin de gagner en temps d'exécution.

Ce type de technologie nécessite toutefois une synchronisation et une communication entre les différents processus, à la manière du découpage des tâches dans une entreprise : le travail est divisé en petits processus distincts, traités par des services différents. Le fonctionnement d'une telle entreprise peut être très perturbé lorsque la communication entre les services ne fonctionne pas correctement.

Le pipeline

Le **pipeline** (ou *pipelining*) est une technologie visant à permettre une plus grande vitesse d'exécution des instructions en parallélisant des étapes.

Pour comprendre le mécanisme du pipeline, il est nécessaire au préalable de comprendre les phases d'exécution d'une instruction. Les phases d'exécution d'une instruction pour un processeur contenant un pipeline « classique » à 5 étages sont les suivantes :

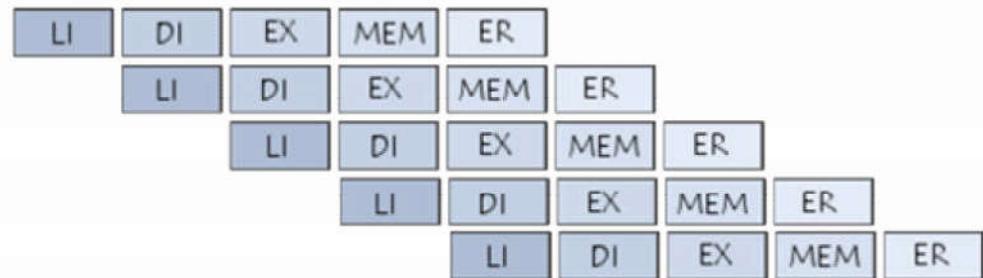
- **LI** : (*Lecture de l'Instruction* (en anglais *FETCH instruction*) depuis le cache ;
- **DI** : (*Décodage de l'Instruction* (*DECODE instruction*) et recherche des opérandes (Registre ou valeurs immédiate);

- **EX** : *Exécution de l'Instruction (EXECute instruction)* (si ADD, on fait la somme, si SUB, on fait la soustraction, etc.);
- **MEM** : *Accès mémoire (MEMory access)*, écriture dans la mémoire si nécessaire ou chargement depuis la mémoire ;
- **ER** : *Ecriture (Write instruction)* de la valeur calculée dans les registres.

Les instructions sont organisées en file d'attente dans la mémoire, et sont chargées les unes après les autres.

Grâce au pipeline, le traitement des instructions nécessite au maximum les cinq étapes précédentes. Dans la mesure où l'ordre de ces étapes est invariable (LI, DI, EX, MEM et ER), il est possible de créer dans le processeur un certain nombre de circuits spécialisés pour chacune de ces phases.

L'objectif du pipeline est d'être capable de réaliser chaque étape en parallèle avec les étapes amont et aval, c'est-à-dire de pouvoir lire une instruction (LI) lorsque la précédente est en cours de décodage (DI), que celle d'avant est en cours d'exécution (EX), que celle située encore précédemment accède à la mémoire (MEM) et enfin que la première de la série est déjà en cours d'écriture dans les registres (ER).



Il faut compter en général 1 à 2 cycles d'horloge (rarement plus) pour chaque phase du pipeline, soit 10 cycles d'horloge maximum par instruction. Pour deux instructions, 12 cycles d'horloge maximum seront nécessaires ($10+2=12$ au lieu de $10*2=20$), car la précédente instruction était déjà dans le pipeline. Les deux instructions sont donc en traitement dans le processeur, avec un décalage d'un ou deux cycles d'horloge). Pour 3 instructions, 14 cycles d'horloge seront ainsi nécessaires, etc.

Le principe du pipeline est ainsi comparable avec une chaîne de production de voitures. La voiture passe d'un poste de travail à un autre en suivant la chaîne de montage et sort complètement assemblée à la sortie du bâtiment. Pour bien comprendre le principe, il est nécessaire de regarder la chaîne dans son ensemble, et non pas véhicule par véhicule. Il faut ainsi 3 heures pour faire une voiture, mais pourtant une voiture est produite toute les minutes !

Il faut noter toutefois qu'il existe différents types de pipelines, de 2 à 40 étages, mais le principe reste le même.

Technologie superscalaire

La technologie **superscalaire** (en anglais *superscaling*) consiste à disposer plusieurs unités de traitement en parallèle afin de pouvoir traiter plusieurs instructions par cycle.

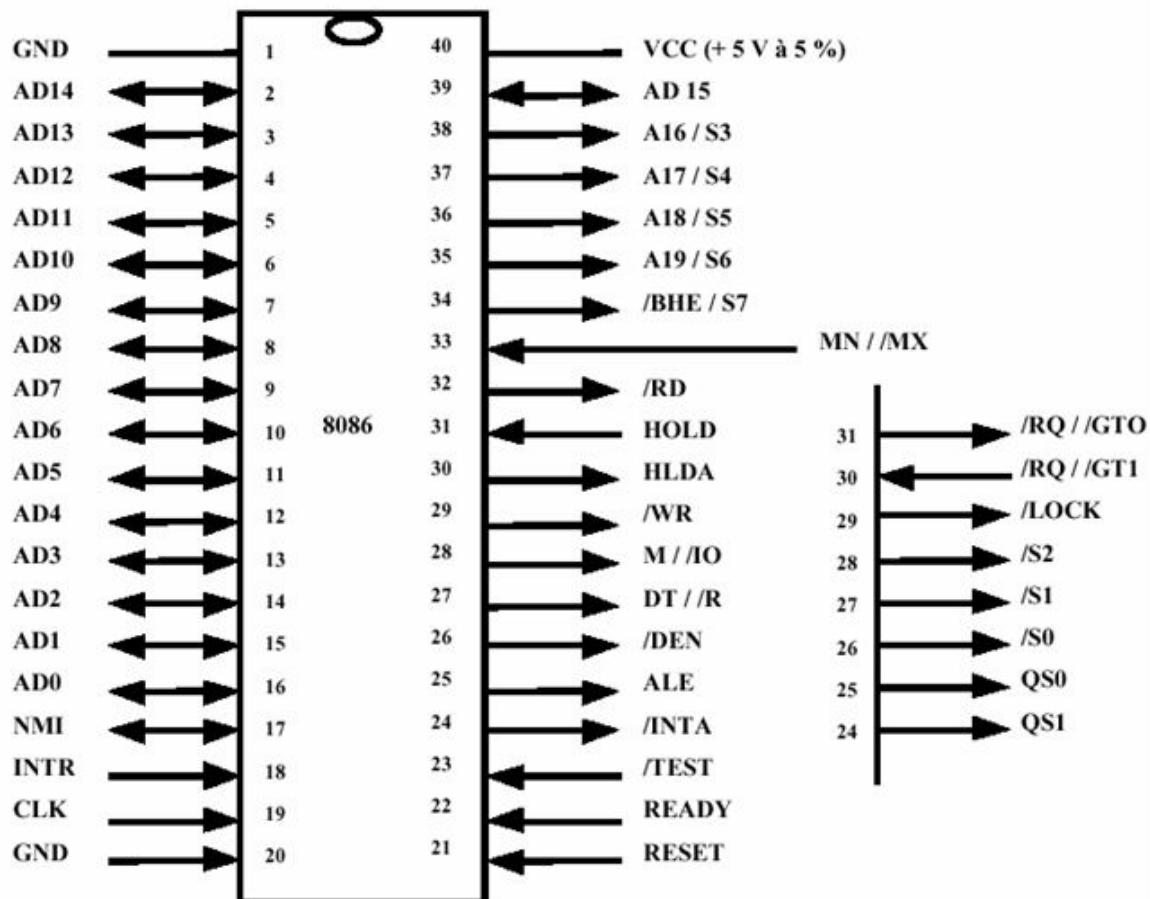
HyperThreading

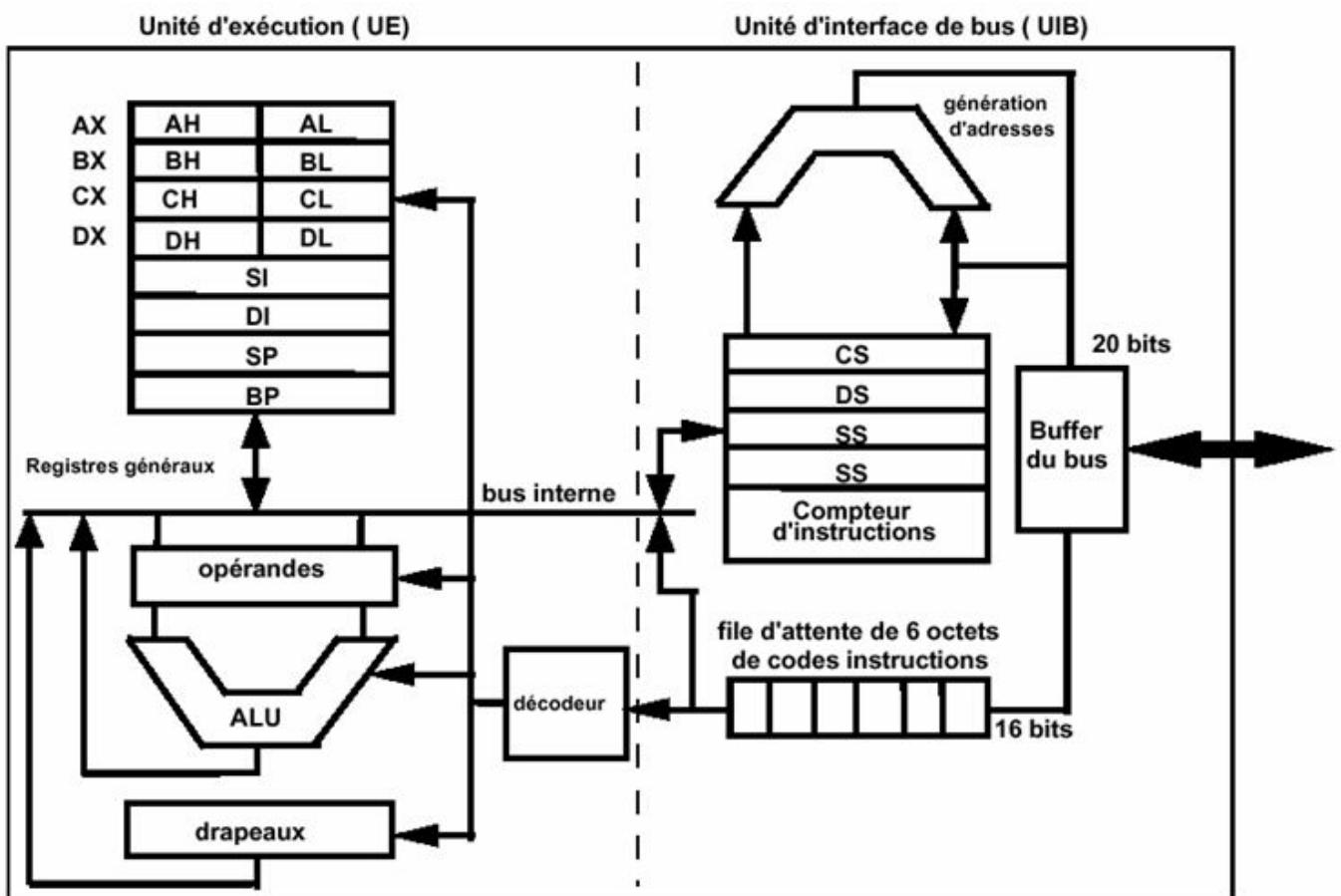
La technologie **HyperThreading** (ou *Hyper-Threading*, noté *HT*, traduisez *HyperFlots* ou *HyperFlux*) consiste à définir deux processeurs logiques au sein d'un processeur physique. Ainsi, le système reconnaît deux processeurs physiques et se comporte en système multitâche en envoyant deux threads simultanés, on parle alors de **SMT** (*Simultaneous Multi Threading*). Cette « supercherie » permet d'utiliser au mieux les ressources du processeur en garantissant que des données lui sont envoyées en masse.

Multi-cœur

Un **processeur multi-cœur** est tout simplement un processeur composé non pas de 1 mais de 2 (ou 4 ou 8) unités de calcul. Ainsi, pour un processeur bi-cœur (ou DualCore) le processeur dispose à fréquence d'horloge égale d'une puissance de calcul deux fois plus importante. Pour autant, le gain n'est pas systématiquement visible. En effet, il faut que les logiciels et les systèmes d'exploitation sachent gérer correctement ces processeurs afin qu'un gain significatif soit perceptible. Ainsi, sous Windows, seul Vista exploite correctement ces processeurs. Dans ce cas, la version 64 bits est conseillée.

Exemple sur le 8086 INTEL





.386

code segment use16

assume cs:code, ds:code, ss:code

org 100h ;offsets décalés de 100h = 256

debut :

mov ah, 09h ;fonction n°9 : écrire une chaîne à l'écran

mov dx, offset message ;mettre l'offset de la chaîne dans DX
int 21h ;écrire la chaîne à l'écran en appelant l'interruption 21h

ret ;rendre la main au DOS

message db "Bonjour, monde !", '\$' ;définition du message

code ends

end debut

D. La mémoire

Rôle de la mémoire

On appelle « **mémoire** » tout composant électronique capable de stocker temporairement des données. On distingue ainsi deux grandes catégories de mémoires :

- la **mémoire centrale** (appelée également *mémoire interne*) permettant de mémoriser temporairement les données lors de l'exécution des programmes. La mémoire centrale est réalisée à l'aide de micro-conducteurs, c'est-à-dire des circuits électroniques spécialisés rapides. La mémoire centrale correspond à ce que l'on appelle la mémoire vive.
- la **mémoire de masse** (appelée également *mémoire physique* ou *mémoire externe*) permettant de stocker des informations à long terme, y compris lors de l'arrêt de l'ordinateur. La mémoire de masse correspond aux dispositifs de stockage magnétiques, tels que le disque dur, aux dispositifs de stockage optique, correspondant par exemple aux CD-ROM ou aux DVD-ROM, ainsi qu'aux mémoires mortes.

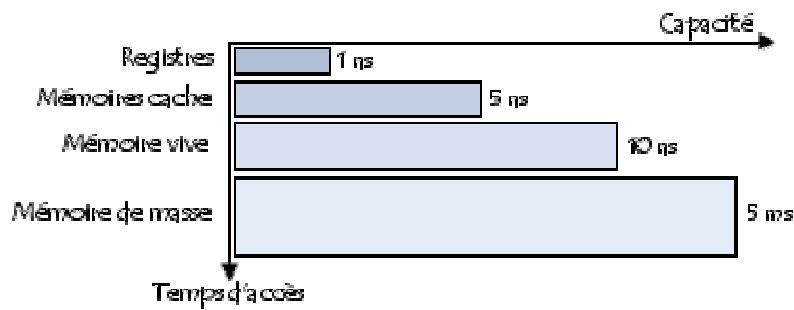
Caractéristiques techniques

Les principales caractéristiques d'une mémoire sont les suivantes :

- La **capacité**, représentant le volume global d'informations (en bits) que la mémoire peut stocker ;
- Le **temps d'accès**, correspondant à l'intervalle de temps entre la demande de lecture/écriture et la disponibilité de la donnée ;
- Le **temps de cycle**, représentant l'intervalle de temps minimum entre deux accès successifs ;
- Le **débit**, définissant le volume d'information échangé par unité de temps, exprimé en bits par seconde ;
- La **non volatilité** caractérisant l'aptitude d'une mémoire à conserver les données lorsqu'elle n'est plus alimentée électriquement.

Ainsi, la mémoire idéale possède une grande capacité avec des temps d'accès et temps de cycle très restreints, un débit élevé et est non volatile.

Néanmoins les mémoires rapides sont également les plus onéreuses. C'est la raison pour laquelle des mémoires utilisant différentes technologies sont utilisées dans un ordinateur, interfacées les unes avec les autres et organisées de façon hiérarchique.



Les mémoires les plus rapides sont situées en faible quantité à proximité du processeur et les mémoires de masse, moins rapides, servent à stocker les informations de manière permanente.

Types de mémoires

Mémoire vive

La **mémoire vive**, généralement appelée **RAM** (*Random Access Memory*, traduisez *mémoire à accès direct*), est la mémoire principale du système, c'est-à-dire qu'il s'agit d'un espace permettant de stocker de manière temporaire des données lors de l'exécution d'un programme.

En effet, contrairement au stockage de données sur une mémoire de masse telle que le disque dur, la mémoire vive est volatile, c'est-à-dire qu'elle permet uniquement de stocker des données tant qu'elle est alimentée électriquement. Ainsi, à chaque fois que l'ordinateur est éteint, toutes les données présentes en mémoire sont irrémédiablement effacées.

Mémoire morte

La **mémoire morte**, appelée **ROM** pour *Read Only Memory* (traduisez *mémoire en lecture seule*) est un type de mémoire permettant de conserver les informations qui y sont contenues même lorsque la mémoire n'est plus alimentée électriquement. A la base ce type de mémoire ne peut être accédée qu'en lecture. Toutefois il est désormais possible d'enregistrer des informations dans certaines mémoires de type *ROM*.

Mémoire flash

La **mémoire flash** est un compromis entre les mémoires de type RAM et les mémoires mortes. En effet, la mémoire Flash possède la non-volatilité des mémoires mortes tout en pouvant facilement être accessible en lecture ou en écriture. En contrepartie les temps d'accès des mémoires flash sont plus importants que ceux de la mémoire vive.

Types de mémoires vives

On distingue généralement deux grandes catégories de mémoires vives :

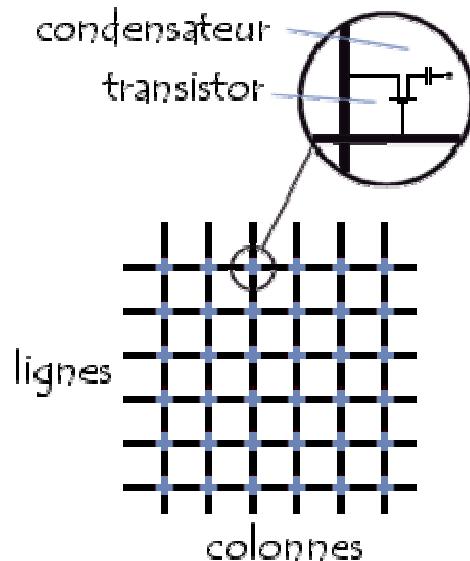
- Les **mémoires dynamiques (DRAM, Dynamic Random Access Module)**, peu coûteuses. Elles sont principalement utilisées pour la mémoire centrale de l'ordinateur ;
- Les **mémoires statiques (SRAM, Static Random Access Module)**, rapides et onéreuses. Les SRAM sont notamment utilisées pour les mémoires cache du [processeur](#) ;

Fonctionnement de la mémoire vive

La mémoire vive est constituée de centaines de milliers de petits condensateurs emmagasinant des charges. Lorsqu'il est chargé, l'état logique du condensateur est égal à 1, dans le cas contraire il est à 0, ce qui signifie que chaque condensateur représente un [bit](#) de la mémoire.

Etant donné que les condensateurs se déchargent, il faut constamment les recharger (le terme exact *estraîcher*, en anglais *refresh*) à un intervalle de temps régulier appelé **cycle de rafraîchissement**. Les mémoires DRAM nécessitent par exemple des cycles de rafraîchissement d'environ 15 nanosecondes (ns).

Chaque condensateur est couplé à un transistor (de type *MOS*) permettant de « récupérer » ou de modifier l'état du condensateur. Ces transistors sont rangés sous forme de tableau (matrice), c'est-à-dire que l'on accède à une *case mémoire* (aussi appelée *point mémoire*) par une ligne et une colonne.



Chaque point mémoire est donc caractérisé par une adresse, correspondant à un numéro de ligne (en anglais *row*) et un numéro de colonne (en anglais *column*). Or cet accès n'est pas instantané et s'effectue pendant un délai appelé **temps de latence**. Par conséquent l'accès à une donnée en mémoire dure un temps égal au temps de cycle auquel il faut ajouter le temps de latence.

Ainsi, pour une mémoire de type DRAM, le temps d'accès est de 60 nanosecondes (35ns de délai de cycle et 25 ns de temps de latence). Sur un ordinateur, le temps de cycle correspond à l'inverse de la fréquence de l'horloge, par exemple pour un ordinateur cadencé à 200 MHz, le temps de cycle est de 5 ns ($1/(200*10^6)$).

Par conséquent un ordinateur ayant une fréquence élevée et utilisant des mémoires dont le temps d'accès est beaucoup plus long que le temps de cycle du processeur doit effectuer des **cycles d'attente** (en anglais *wait state*) pour accéder à la mémoire. Dans le cas d'un ordinateur cadencé à 200 MHz utilisant des mémoires de types DRAM (dont le temps d'accès est de 60ns), il y a 11 cycles d'attente pour un cycle de transfert. Les performances de l'ordinateur sont d'autant diminuées qu'il y a de cycles d'attentes, il est donc conseillé d'utiliser des mémoires plus rapides.

Formats de barrettes de mémoire vive

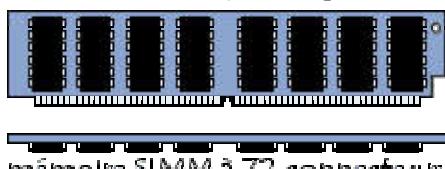
Il existe de nombreux types de mémoires vives. Celles-ci se présentent toutes sous la forme de barrettes de mémoire enfichables sur la carte-mère.

Les premières mémoires se présentaient sous la forme de puces appelées *DIP (Dual Inline Package)*. Désormais les mémoires se trouvent généralement sous la forme de barrettes, c'est-à-dire des cartes enfichables dans des connecteurs prévus à cet effet. On distingue habituellement trois types de barrettes de RAM :

- les barrettes au format **SIMM** (*Single Inline Memory Module*) : il s'agit de circuits imprimés dont une des faces possède des puces de mémoire. Il existe deux types de barrettes SIMM, selon le nombre de connecteurs :
 - Les barrettes SIMM à 30 connecteurs (dont les dimensions sont 89x13mm) sont des mémoires 8 bits qui équipaient les premières générations de PC (286, 386).

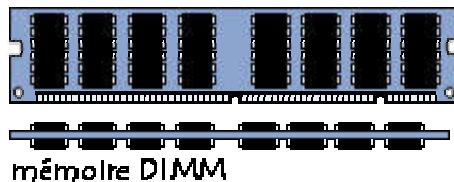


- Les barrettes SIMM à 72 connecteurs (dont les dimensions sont 108x25mm) sont des mémoires capables de gérer 32 bits de données simultanément. Ces mémoires équipent des PC allant du 386DX aux premiers Pentium. Sur ces derniers le processeur travaille avec un bus de données d'une largeur de 64 bits, c'est la raison pour laquelle il faut absolument équiper ces ordinateurs de deux barrettes SIMM. Il n'est pas possible d'installer des barrettes 30 broches sur des emplacements à 72 connecteurs dans la mesure où un détrompeur (encoche au centre des connecteurs) en empêche l'enfichage.



- les barrettes au format **DIMM** (*Dual Inline Memory Module*) sont des mémoires 64 bits, ce qui explique pourquoi il n'est pas nécessaire de les appairer. Les barrettes DIMM possèdent des puces de mémoire de part et d'autre du circuit imprimé et ont également 84 connecteurs de chaque côté, ce qui les dote d'un total de 168 broches. En plus de leurs

dimensions plus grandes que les barrettes SIMM (130x25mm) ces barrettes possèdent un second détrompeur pour éviter la confusion.



Il peut être intéressant de noter que les connecteurs DIMM ont été améliorés afin de faciliter leur insertion grâce à des leviers situés de part et d'autre du connecteur. Il existe en outre des modules de plus petite taille, appelés **SO DIMM** (*Small Outline DIMM*), destinés aux ordinateurs portables. Les barrettes *SO DIMM* comportent uniquement 144 broches pour les mémoires 64 bits et 77 pour les mémoires 32 bits.

- les barrettes au format **RIMM** (*Rambus Inline Memory Module*, appelées également *RD-RAM* ou *DRD-RAM*) sont des mémoires 64 bits développée par la société Rambus. Elles possèdent 184 broches. Ces barrettes possèdent deux encoches de repérage (détrompeurs), évitant tout risque de confusion avec les modules précédents.

Compte tenu de leur vitesse de transfert élevée, les barrettes RIMM possèdent un film thermique chargé d'améliorer la dissipation de la chaleur.

Comme dans le cas des DIMM, il existe des modules de plus petite taille, appelés **SO RIMM** (*Small Outline RIMM*), destinés aux ordinateurs portables. Les barrettes *SO RIMM* comportent uniquement 160 broches.

DRAM PM

La **DRAM** (*Dynamic RAM*, RAM dynamique) est le type de mémoire le plus répandu au début du millénaire. Il s'agit d'une mémoire dont les transistors sont rangés dans une matrice selon des lignes et des colonnes. Un transistor, couplé à un condensateur donne l'information d'un bit. 1 octet comprenant 8 bits, une barrette de mémoire DRAM de 256 Mo contiendra donc $256 * 2^{10} * 2^{10} = 256 * 1024 * 1024 = 268\ 435\ 456$ octets = $268\ 435\ 456 * 8 = 2\ 147\ 483\ 648$ bits = 2 147 483 648 transistors. Une barrette de 256Mo possède ainsi en réalité une capacité de 268 435 456 octets, soit 268 Mo ! Ce sont des mémoires dont le temps d'accès est de 60 ns.

D'autre part, les accès mémoire se font généralement sur des données rangées consécutivement en mémoire. Ainsi le mode d'accès en *rafale* (**burst mode**) permet d'accéder aux trois données consécutives à la première sans temps de latence supplémentaire. Dans ce mode en rafale, le temps d'accès à la première donnée est égal au temps de cycle auquel il faut ajouter le temps de latence, et le temps d'accès aux trois autres données est uniquement égal aux temps de cycle, on note donc sous la forme X-Y-Y-Y les quatre temps d'accès, par exemple la notation 5-3-3-3 indique une mémoire pour laquelle 5 cycles d'horloge sont nécessaires pour accéder à la première donnée et 3 pour les suivantes.

DRAM FPM

Pour accélérer les accès à la DRAM, il existe une technique, appelée **pagination** consistant à accéder à des données situées sur une même colonne en modifiant uniquement l'adresse de la ligne, ce qui permet d'éviter la répétition du numéro de colonne entre la lecture de chacune des lignes. On parle alors de **DRAM FPM** (*Fast Page Mode*). La FPM permet d'obtenir des temps d'accès de l'ordre de 70 à 80 nanosecondes pour une fréquence de fonctionnement pouvant aller de 25 à 33 Mhz.

DRAM EDO

La **DRAM EDO** (*Extended Data Out*, soit *Sortie des données améliorée* parfois également appelé "*hyper-page*") est apparue en 1995. La technique utilisée avec ce type de mémoire consiste à adresser la colonne suivante pendant la lecture des données d'une colonne. Cela crée un chevauchement des accès permettant de gagner du temps sur chaque cycle. Le temps d'accès à la mémoire EDO est donc d'environ 50 à 60 nanosecondes pour une fréquence de fonctionnement allant de 33 à 66 Mhz.

Ainsi, la RAM EDO, lorsqu'elle est utilisée en mode rafale permet d'obtenir des cycles de la forme 5-2-2-2, soit un gain de 4 cycles sur l'accès à 4 données. Dans la mesure où la mémoire EDO n'acceptait pas des fréquences supérieures à 66 Mhz, elle a disparu au bénéfice de la SDRAM.

SDRAM

La **SDRAM** (*Synchronous DRAM*, traduisez *RAM synchrone*), apparue en 1997, permet une lecture des données synchronisée avec le bus de la carte-mère, contrairement aux mémoires EDO et FPM (qualifiées d'*asynchrones*) possédant leur propre horloge. La SDRAM permet donc de s'affranchir des temps d'attente dus à la synchronisation avec la carte-mère. Celle-ci permet d'obtenir un cycle en mode rafale de la forme 5-1-1-1, c'est-à-dire un gain de 3 cycles par rapport à la RAM EDO. De cette façon la SDRAM est capable de fonctionner avec une cadence allant jusqu'à 150 Mhz, lui permettant d'obtenir des temps d'accès d'environ 10 ns.

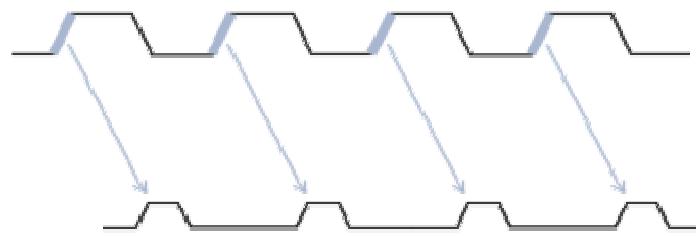
DR-SDRAM (Rambus DRAM)

La **DR-SDRAM** (*Direct Rambus DRAM* ou encore *RDRAM*) est un type de mémoire permettant de transférer les données sur un bus de 16 bits de largeur à une cadence de 800Mhz, ce qui lui confère une bande passante de 1,6 Go/s. Comme la SDRAM, ce type de mémoire est synchronisé avec l'horloge du bus pour améliorer les échanges de données. En contrepartie, la mémoire RAMBUS est une technologie propriétaire, ce qui signifie que toute entreprise désirant construire des barrettes de RAM selon cette technologie doit reverser des droits (royalties) aux sociétés RAMBUS et Intel.

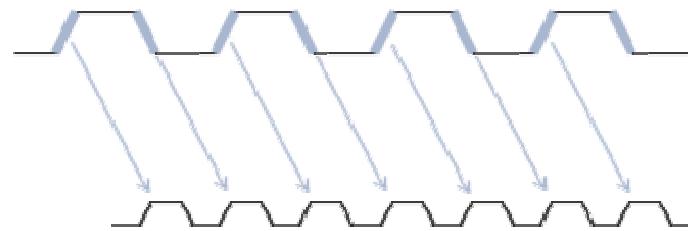
DDR-SDRAM

La **DDR-SDRAM** (*Double Data Rate SDRAM*) est une mémoire basée sur la technologie SDRAM, permettant de doubler le taux de transfert de la SDRAM à fréquence égale.

La lecture ou l'écriture de données en mémoire est réalisé sur la base d'une horloge. Les mémoires DRAM standard utilisent une méthode appelé **SDR** (*Single Data Rate*) consistant à lire ou à écrire une donnée à chaque front montant.



La DDR permet de doubler la fréquence des lectures/écritures, avec une horloge cadencée à la même fréquence, en envoyant les données à chaque front montant, ainsi qu'à chaque front descendant.

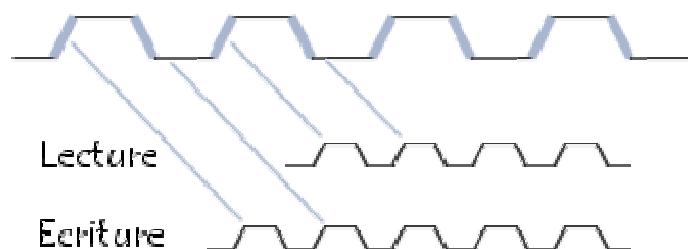


Les mémoires DDR possèdent généralement une appellation commerciale du type PCXXXX où «XXXX» représente le débit en Mo/s.

DDR2-SDRAM

La mémoire DDR2 (ou DDR-II) permet d'atteindre des débits deux fois plus élevés que la DDR à fréquence externe égale.

On parle de QDR (*Quadruple Data Rate* ou *quad-pumped*) pour désigner la méthode de lecture et d'écriture utilisée. La mémoire DDR2 utilise en effet deux canaux séparés pour la lecture et pour l'écriture, si bien qu'elle est capable d'envoyer ou de recevoir deux fois plus de données que la DDR.



La DDR2 possède également un plus grand nombre de connecteurs que la DDR classique (240 pour la DDR2 contre 184 pour la DDR).

DDR3-SDRAM

Le DDR3 SDRAM améliore les performances par rapport au DDR2, mais surtout diminue la consommation électrique. En effet, celle-ci est de 40 % inférieure, en particulier grâce à une baisse du voltage utilisé, une finesse de gravure accrue. Si le débit théorique de ces barrettes peut dépasser les 10 Go/s, les temps de latence sont restés dans les mêmes ordres de grandeur que ceux des DDR2.

Les barrettes DDR3 ont 240 connecteurs comme les DDR2 mais ne sont absolument pas compatibles (des détrompeurs empêchent l'insertion).

Tableau récapitulatif

Le tableau ci-dessous donne la correspondance entre la fréquence de la [carte-mère](#) (FSB), celle de la mémoire (RAM) et son débit :

Mémoire	Appellation	Fréquence E/S	Fréquence mémoire	Débit
DDR200	PC1600	200 MHz	100 MHz	1,6 Go/s
DDR266	PC2100	266 MHz	133 MHz	2,1 Go/s
DDR333	PC2700	333 MHz	166 MHz	2,7 Go/s
DDR400	PC3200	400 MHz	200 MHz	3,2 Go/s
DDR433	PC3500	433 MHz	217 MHz	3,5 Go/s
DDR466	PC3700	466 MHz	233 MHz	3,7 Go/s
DDR500	PC4000	500 MHz	250 MHz	4 Go/s
DDR533	PC4200	533 MHz	266 MHz	4,2 Go/s
DDR538	PC4300	538 MHz	269 MHz	4,3 Go/s
DDR550	PC4400	550 MHz	275 MHz	4,4 Go/s
DDR2-400	PC2-3200	400 MHz	100 MHz	3,2 Go/s
DDR2-533	PC2-4300	533 MHz	133 MHz	4,3 Go/s
DDR2-667	PC2-5300	667 MHz	167 MHz	5,3 Go/s
DDR2-675	PC2-5400	675 MHz	172,5 MHz	5,4 Go/s
DDR2-800	PC2-6400	800 MHz	200 MHz	6,4 Go/s
DDR2-1066	PC2-8500	533 MHz	266 MHz	8,5 Go/s
DDR2-1100	PC2-8800	560 MHz	280 MHz	8,8 Go/s
DDR2-1200	PC2-9600	600 MHz	300 MHz	9,6 Go/s
DDR3-800	PC3-6400	400 MHz	100 MHz	6,4 Go/s
DDR3-1066	PC3-8500	533 MHz	133 MHz	8,5 Go/s

DDR3-1333	PC3-10600	666 MHz	166 MHz	10,7 Go/s
DDR3-1600	PC3-12800	800 MHz	200 MHz	12,8 Go/s

Synchronisation (timings)

Il n'est pas rare de voir des notations du type 3-2-2-2 ou 2-3-3-2 pour décrire le paramétrage de la mémoire vive. Cette suite de quatre chiffres décrit la synchronisation de la mémoire (en anglais *timing*), c'est-à-dire la succession de cycles d'horloge nécessaires pour accéder à une donnée stockée en mémoire vive. Ces quatre chiffres correspondent généralement, dans l'ordre, aux valeurs suivantes :

- **CAS delay** ou **CAS latency** (CAS signifiant *Column Address Strobe*) : il s'agit du nombre de cycles d'horloge s'écoulant entre l'envoi de la commande de lecture et l'arrivée effective de la donnée. Autrement dit, il s'agit du temps d'accès à une colonne.
- **RAS Precharge Time** (noté *tRP*, RAS signifiant *Row Address Strobe*) : il s'agit du nombre de cycles d'horloge entre deux instructions RAS, c'est-à-dire entre deux accès à une ligne. opération.
- **RAS to CAS delay** (noté parfois *tRCD*) : il s'agit du nombre de cycles d'horloge correspondant au temps d'accès d'une ligne à une colonne.
- **RAS active time** (noté parfois *tRAS*) : il s'agit du nombre de cycles d'horloge correspondant au temps d'accès à une ligne.

Les cartes mémoires sont équipées d'un dispositif appelé **SPD** (*Serial Presence Detect*), permettant au [BIOS](#) de connaître les valeurs nominales de réglage définies par le fabricant. Il s'agit d'une [EEPROM](#) dont les données seront chargées par le BIOS si l'utilisateur choisi le réglage « auto ».

La correction d'erreurs

Certaines mémoires possèdent des mécanismes permettant de pallier les erreurs afin de garantir l'intégrité des données qu'elles contiennent. Ce type de mémoire est généralement utilisé sur des systèmes travaillant sur des données critiques, c'est la raison pour laquelle on trouve ce type de mémoire dans les serveurs.

Bit de parité

Les barrettes avec bit de parité permettent de s'assurer que les données contenues dans la mémoire sont bien celles que l'on désire. Pour ce faire, un des bits de chaque octet stocké en mémoire sert à conserver la somme des bits de données.

Le bit de parité vaut 1 lorsque la somme des bits de données est impaire et 0 dans le cas contraire.

De cette façon les barrettes avec bit de parité permettent de vérifier l'intégrité des données mais ne permettent pas de corriger les erreurs. De plus pour 9 Mo de mémoire, seulement 8 serviront à stocker des données, dans la mesure où le dernier mégaoctet conservera les bits de parité.

Barrettes ECC

Les barrettes de mémoire ECC (*Error Correction Coding*) sont des mémoires possédant plusieurs bits dédiés à la correction d'erreur (on les appelle ainsi *bits de contrôle*). Ces barrettes, utilisées principalement dans les serveurs, permettent de détecter les erreurs et de les corriger.

Dual Channel

Certains contrôleurs mémoire proposent un double canal (en anglais *Dual Channel*) pour la mémoire. Il s'agit d'exploiter les modules de mémoire par paire afin de cumuler la bande passante et ainsi exploiter au maximum les capacités du système. Il est essentiel, lors de l'utilisation du Dual Channel, d'utiliser des barrettes identiques par paire (fréquence, capacité et préférentiellement de même marque).

La mémoire morte (ROM)

Il existe un type de mémoire permettant de stocker des données en l'absence de courant électrique, il s'agit de la ***ROM*** (*Read Only Memory*, dont la traduction littérale est *mémoire en lecture seule*) appelée **mémoire morte**, parfois *mémoire non volatile* car elle ne s'efface pas lors de la mise hors tension du système.

Ce type de mémoire permet notamment de conserver les données nécessaires au démarrage de l'ordinateur. 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.

Différentes mémoires de type *ROM* contiennent des données indispensables au démarrage, c'est-à-dire :

- Le **BIOS** est un programme permettant de piloter les interfaces d'entrée-sortie principales du système, d'où le nom de *BIOS ROM* donné parfois à la puce de mémoire morte de la carte-mère qui l'héberge.
- Le **chargeur d'amorce**: un programme permettant de charger le système d'exploitation en mémoire (vive) et de le lancer. Celui-ci cherche généralement le système d'exploitation sur le lecteur de disquette, puis sur le disque dur, ce qui permet de pouvoir lancer le système d'exploitation à partir d'une disquette système en cas de dysfonctionnement du système installé sur le disque dur.
- Le **Setup CMOS**, c'est l'écran disponible à l'allumage de l'ordinateur permettant de modifier les paramètres du système (souvent appelé *BIOS à tort...*).
- Le **Power-On Self Test (POST)**, programme exécuté automatiquement à l'amorçage du système permettant de faire un test du système (c'est pour cela par exemple que vous voyez le système "compter" la RAM au démarrage).

Etant donné que les ROM sont beaucoup plus lentes que les mémoires de types **RAM** (une ROM a un temps d'accès de l'ordre de 150 ns tandis qu'une mémoire de type SDRAM a un temps d'accès d'environ 10 ns), les instructions contenues dans la ROM sont parfois copiées en RAM au démarrage, on parle alors *deshadowing* (en français cela pourrait se traduire par *ombrage*, mais on parle généralement de *mémoire fantôme*).

Les types de ROM

Les ROM ont petit à petit évolué de *mémoires mortes figées* à des mémoires programmables, puis reprogrammables.

ROM

Les premières ROM étaient fabriquées à l'aide d'un procédé inscrivant directement les données binaires dans une plaque de silicium grâce à un masque. Ce procédé est maintenant obsolète.

PROM

Les **PROM** (*Programmable Read Only Memory*) ont été mises au point à la fin des années 70 par la firme *Texas Instruments*. Ces mémoires sont des puces constituées de milliers de fusibles (ou bien de diodes) pouvant être "grillés" grâce à un appareil appelé « *programmateur de ROM* », appliquant une forte tension (12V) aux cases mémoire devant être marquées. Les fusibles ainsi grillés correspondent à des 0, les autres à des 1.

EPROM

Les **EPROM** (*Erasable Programmable Read Only Memory*) sont des PROM pouvant être effacées. Ces puces possèdent une vitre permettant de laisser passer des rayons ultra-violets. Lorsque la puce est en présence de rayons ultra-violets d'une certaine longueur d'onde, les fusibles sont reconstitués, c'est-à-dire que tous les bits de la mémoire sont à nouveau à 1. C'est pour cette raison que l'on qualifie ce type de PROM d'*effaçable*.

EEPROM

Les **EEPROM** (*Electrically Erasable Read Only Memory*) sont aussi des PROM effaçables, mais contrairement aux EPROM, celles-ci peuvent être effacées par un simple courant électrique, c'est-à-dire qu'elles peuvent être effacées même lorsqu'elles sont en position dans l'ordinateur.

Il existe une variante de ces mémoires appelée **mémoires flash** (également *ROM Flash* ou *Flash EPROM*). Contrairement aux EEPROM classiques, utilisant 2 à 3 transistors par bit à mémoriser, la Flash EPROM utilise un seul transistor. D'autre part l'EEPROM peut-être écrite et lue mot par mot, alors que la Flash ne peut être effacée que par pages (la taille des pages étant en constante diminution).

Enfin la densité de la mémoire Flash est plus importante, ce qui permet la réalisation de puces contenant plusieurs centaines de Mégoctets. Des EEPROM sont ainsi préférentiellement utilisées pour la mémorisation de données de configuration et la mémoire Flash pour du code programmable (programmes informatiques).

On qualifie de flashage l'action consistant à reprogrammer une EEPROM.

Il existe de nombreux types de mémoire vive (RAM). Celles-ci se présentent sous la forme de barrettes que l'on enfiche sur la carte mère.

Les différents formats de mémoires :

- les barrettes mémoires de format **SIMM** (Single In-line Memory Module) :

Ce type de mémoire était utilisé avec les anciens systèmes : les barrettes SIMM à 30 connecteurs (8 bits) équipaien les PC 286 et 386, et les barrettes SIMM à 72 connecteurs (32 bits) équipaien les PC 386DX, 486 et les premiers Pentium.

- les barrettes mémoires de format **DIMM** (Dual In-line Memory Module) :

Ce type de mémoire est actuellement utilisé dans nos PC. Il s'agit de mémoires 64 bits. Leur dimension est de 130x25mm. Contrairement aux mémoires de type SIMM, les mémoires DIMM possèdent des puces de mémoire de part et d'autre de la barrette.

Les différents types de mémoires :

- **DRAM EDO** (Extended Data Out) :

Ce type de mémoire est apparu en 1995. La principale caractéristique de ce type de mémoire est sa capacité à adresser la colonne suivante pendant la lecture des données d'une colonne. Le temps d'accès à la mémoire EDO est de 50 à 60 nanosecondes pour une fréquence allant de 33 à 66 MHz. N'étant pas capable de supporter des fréquences supérieures à 66 MHz, ce type de mémoire a disparu au profit de la mémoire SDRAM.

- **SDRAM** (Synchronous Dynamic RAM) :

Ce type de mémoire est apparu en 1997 pour remplacer la mémoire EDO. La SDRAM permet, contrairement à la mémoire EDO, une lecture des données synchronisée. Celle-ci est capable de fonctionner à des fréquences allant jusqu'à 150 MHz avec des temps d'accès d'environ 10 nanosecondes.

- **DDR-SDRAM** (Double Data Rate SDRAM) :

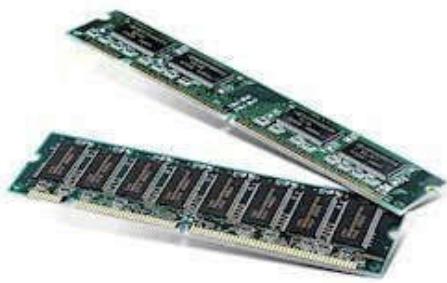
Ce type de mémoire est basé sur la technologie de la mémoire SDRAM, qui permet de doubler le taux de transfert de la SDRAM à fréquence égale.

- **DR-SDRAM** (Rambus DRAM) :

Ce type de mémoire est capable de fonctionner à une fréquence de 800 MHz, ce qui lui confère une bande passante de 1,6 Go/s. Ce type de mémoire a été conçu par les sociétés RAMBUS et Intel. Il s'agit d'un type de mémoire propriétaire.

Quel type de mémoire pour mon processeur ?

- **SDRAM** : ce type de mémoire existe dans différentes fréquences (66, 100 et 133 MHz). Nous les utilisons avec les processeurs Pentium 1, 2 et 3, les Intel Celeron, les processeurs AMD K6 et Duron. Les SDRAM 133 MHz sont disponibles en 128, 256 et 512 Mb.



- **DDR-SDRAM** : ce type de mémoire existe dans différentes fréquences (266, 333 et 400 MHz). Nous les utilisons avec les processeurs Intel Pentium 4 et les processeurs AMD Athlon.



- **Rambus** : ce type de mémoire n'est utilisée qu'avec les processeur Intel Pentium 4



E. L'ordinateur

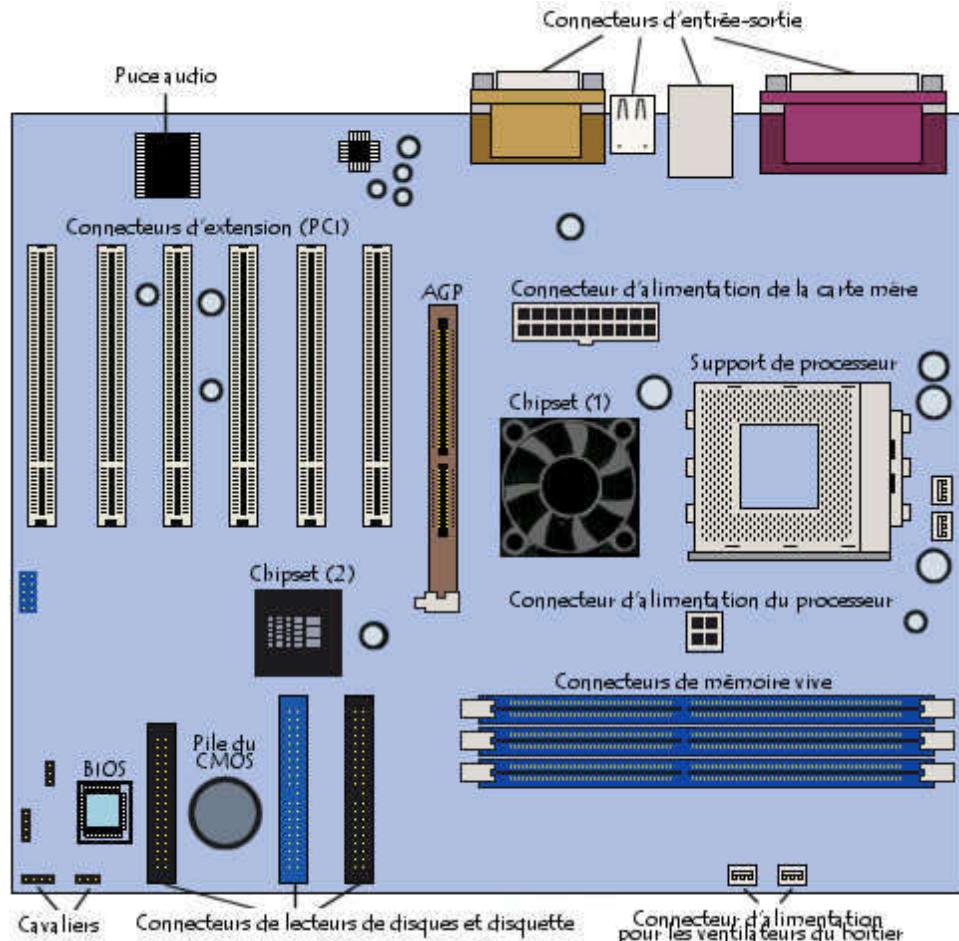
Familles d'ordinateurs

On distingue généralement plusieurs familles d'ordinateurs selon leur format :

- Les **mainframes** (en français *ordinateurs centraux*), ordinateurs possédant une grande puissance de calcul, des capacités d'entrée-sortie gigantesques et un haut niveau de fiabilité. Les mainframes sont utilisés dans de grandes entreprises pour effectuer des opérations lourdes de calcul ou de traitement de données volumineuses. Les mainframes sont généralement utilisés dans des architectures centralisées, dont ils sont le cœur.
- Les **ordinateurs personnels**, parmi lesquels on distingue :
 - Les **ordinateurs de bureau** (en anglais *desktop computers*), composés d'un boîtier renfermant une carte mère et permettant de raccorder les différents périphériques tels que l'écran .
 - Les **ordinateurs portables** (en anglais *laptop* ou *notebooks*), composé d'un boîtier intégrant un écran dépliable, un clavier et un grand nombre de périphériques incorporés.
 - Les **tablettes PC** (en anglais *tablet PC*, également appelées *ardoises électroniques*), composées d'un boîtier intégrant un écran tactile ainsi qu'un certain nombre de périphériques incorporés.
 - Les **centres multimédia** (*Media Center*), représentant une plate-forme matérielle, destinée à une utilisation dans le salon pour le pilotage des éléments hifi (chaîne hifi, téléviseur, platine DVD, etc.).
 - Les **assistant personnels** (appelés **PDA**, pour *Personal digital Assistant*, ou encore *handheld*, littéralement «tenu dans la main»), parfois encore qualifiés d'organiseur (en anglais *organizer*) ou d'agenda électronique, sont des ordinateurs de poche proposant des fonctionnalités liées à l'organisation personnelle. Ils peuvent être dotés des fonctions d'un téléphone portable. On parle alors souvent dans ce cas de **smartphone**.
 - Enfin, les **netbooks** sont des ordinateurs portables dotés d'un écran de petite dimension (généralement 12") et dont on a remplacé le disque dur par de la mémoire flash, afin de réduire la consommation électrique (et le coût).

Présentation de la carte mère

L'élément constitutif principal de l'ordinateur est la **carte mère** (en anglais « *mainboard* » ou « *motherboard* », parfois abrégé en « *mobo* »). La carte mère est le socle permettant la connexion de l'ensemble des éléments essentiels de l'ordinateur.



Comme son nom l'indique, la carte mère est une carte maîtresse, prenant la forme d'un grand circuit imprimé possédant notamment des connecteurs pour les cartes d'extension, les barrettes de mémoires, le processeur, etc.

Caractéristiques

Il existe plusieurs façons de caractériser une carte mère, notamment selon les caractéristiques suivantes :

- le facteur d'encombrement,
- le chipset,
- le type de support de processeur,
- les connecteurs d'entrée-sortie.

Facteur d'encombrement d'une carte mère

On désigne généralement par le terme « **facteur d'encombrement** » (ou *facteur de forme*, en anglais *form factor*), la géométrie, les dimensions, l'agencement et les caractéristiques électriques de la carte mère. Afin de fournir des cartes mères pouvant s'adapter dans différents boîtiers de marques différentes, des standards ont été mis au point :

- **AT baby / AT full format** est un format utilisé sur les premiers ordinateurs PC du type 386 ou 486. Ce format a été remplacé par le format ATX possédant une forme plus propice à la circulation de l'air et rendant l'accès aux composants plus pratique ;
- **ATX** : Le format ATX est une évolution du format Baby-AT. Il s'agit d'un format étudié pour améliorer l'ergonomie. Ainsi la disposition des connecteurs sur une carte mère ATX est prévue de manière à optimiser le branchement des périphériques (les connecteurs IDE sont par exemple situés du côté des disques). D'autre part, les composants de la carte mère sont orientés parallèlement, de manière à permettre une meilleure évacuation de la chaleur ;
 - **ATX standard** : Le format ATX standard présente des dimensions classiques de 305x244 mm. Il propose un connecteur AGP et 6 connecteurs PCI.
 - **micro-ATX** : Le format microATX est une évolution du format ATX, permettant d'en garder les principaux avantages tout en proposant un format de plus petite dimension (244x244 mm), avec un coût réduit. Le format micro-ATX propose un connecteur AGP et 3 connecteurs PCI.
 - **Flex-ATX** : Le format FlexATX est une extension du format microATX afin d'offrir une certaine flexibilité aux constructeurs pour le design de leurs ordinateurs. Il propose un connecteur AGP et 2 connecteurs PCI.
 - **mini-ATX** : Le format miniATX est un format compact alternatif au format microATX (284x208 mm), proposant un connecteur AGP et 4 connecteurs PCI au lieu des 3 du format microATX. Il est principalement destiné aux ordinateurs de type mini-PC (barebone).
- **BTX** : Le format BTX (*Balanced Technology eXtended*), porté par la société Intel, est un format prévu pour apporter quelques améliorations de l'agencement des composants afin d'optimiser la circulation de l'air et de permettre une optimisation acoustique et thermique. Les différents connecteurs (connecteurs de mémoire, connecteurs d'extension) sont ainsi alignés parallèlement, dans le sens de circulation de l'air. Par ailleurs le microprocesseur est situé à l'avant du boîtier au niveau des entrées d'aération, où l'air est le plus frais. Le connecteur d'alimentation BTX est le même que celui des alimentations ATX. Le standard BTX définit trois formats :
 - **BTX standard**, présentant des dimensions standard de 325x267 mm ;
 - **micro-BTX**, de dimensions réduites (264x267 mm) ;
 - **pico-BTX**, de dimensions extrêmement réduites (203x267 mm).
- **ITX** : Le format ITX (*Information Technology eXtended*), porté par la société Via, est un format extrêmement compact prévu pour des configurations exigües telles que les mini-PC. Il existe deux principaux formats ITX :
 - **mini-ITX**, avec des dimensions minuscules (170x170 mm) est un emplacement PCI ;

- **nano-ITX**, avec des dimensions extrêmement minuscules (120x120 mm) et un emplacement miniPCI.

Ainsi, du choix d'une carte mère (et de son facteur de forme) dépend le choix du boîtier. Le tableau ci-dessous récapitule les caractéristiques des différents facteurs de forme :

Facteur de forme	Dimensions	Emplacements
ATX	305 mm x 244 mm	AGP / 6 PCI
microATX	244 mm x 244 mm	AGP / 3 PCI
FlexATX	229 mm x 191 mm	AGP / 2 PCI
Mini ATX	284 mm x 208 mm	AGP / 4 PCI
Mini ITX	170 mm x 170 mm	1 PCI
Nano ITX	120 mm x 120 mm	1 MiniPCI
BTX	325 mm x 267 mm	7
microBTX	264 mm x 267 mm	4
picoBTX	203 mm x 267 mm	1

Composants intégrés

La carte mère contient un certain nombre d'éléments embarqués, c'est-à-dire intégrés sur son circuit imprimé :

- Le chipset, circuit qui contrôle la majorité des ressources (interface de bus du processeur, mémoire cache et mémoire vive, slots d'extension,...),
- L'horloge et la pile du CMOS,
- Le BIOS,
- Le bus système et les bus d'extension.

En outre, les cartes mères récentes embarquent généralement un certain nombre de périphériques multimédia et réseau pouvant être désactivés :

- carte réseau intégrée ;
- carte graphique intégrée ;
- carte son intégrée ;
- contrôleurs de disques durs évolués.

Le chipset

Le **chipset** (traduisez *jeu de composants* ou *jeu de circuits*) est un circuit électronique chargé de coordonner les échanges de données entre les divers composants de l'ordinateur (processeur, mémoire...). Dans la mesure où le chipset est intégré à la carte mère, il est important de choisir une carte mère intégrant un chipset récent afin de maximiser les possibilités d'évolutivité de l'ordinateur.

Certains chipsets intègrent parfois une puce graphique ou une puce audio, ce qui signifie qu'il n'est pas nécessaire d'installer une carte graphique ou une carte son. Il est toutefois parfois conseillé de les désactiver (lorsque cela est possible) dans le setup du BIOS et d'installer des cartes d'extension de qualité dans les emplacements prévus à cet effet.

L'horloge et la pile du CMOS

L'horloge temps réel (notée **RTC**, pour *Real Time Clock*) est un circuit chargé de la synchronisation des signaux du système. Elle est constituée d'un cristal qui, en vibrant, donne des impulsions (appelés *tops d'horloge*) afin de cadencer le système. On appelle *fréquence de l'horloge* (exprimée en *MHz*) le nombre de vibrations du cristal par seconde, c'est-à-dire le nombre de *tops d'horloge* émis par seconde. Plus la fréquence est élevée, plus le système peut traiter d'informations.

Lorsque l'ordinateur est mis hors tension, l'alimentation cesse de fournir du courant à la carte mère. Or, lorsque l'ordinateur est rebranché, le système est toujours à l'heure. Un circuit électronique, appelé *CMOS*(*Complementary Metal-Oxyde Semiconductor*, parfois appelé *BIOS CMOS*), conserve en effet certaines informations sur le système, telles que l'heure, la date système et quelques paramètres essentiels du système.

Le CMOS est continuellement alimenté par une pile (au format *pile bouton*) ou une batterie située sur la carte mère. Ainsi, les informations sur le matériel installé dans l'ordinateur (comme par exemple le nombre de pistes, de secteurs de chaque disque dur) sont conservées dans le CMOS. Dans la mesure où le CMOS est une mémoire lente, certains systèmes recopient parfois le contenu du CMOS dans la RAM (mémoire rapide), le terme de « *memory shadow* » est employé pour décrire ce processus de copie en mémoire vive.

Le « *complémentary metal-oxyde semiconductor* », est une technologie de fabrication de transistors, précédée de bien d'autres, telles que la *TTL* (« *Transistor-transistor-logique* »), la *TTLS* (*TTL Schottky*) (plus rapide), ou encore le *NMOS* (canal négatif) et le *PMOS* (canal positif).

Le CMOS a permis de mettre des canaux complémentaires sur une même puce. Par rapport à la TTL ou TTLS, le CMOS est beaucoup moins rapide, mais a consommé en revanche infiniment moins d'énergie, d'où son emploi dans les horloges d'ordinateurs, qui sont alimentées par des piles. Le terme de CMOS est parfois utilisé à tort pour désigner l'horloge des ordinateurs.

Lorsque l'heure du système est régulièrement réinitialisée, ou que l'horloge prend du retard, il suffit généralement d'en changer la pile !

Le BIOS

Le BIOS (*Basic Input/Output System*) est le programme basique servant d'interface entre le système d'exploitation et la carte mère. Le BIOS est stocké dans une *ROM* (mémoire morte, c'est-à-dire une mémoire en lecture seule), ainsi il utilise les données contenues dans le *CMOS* pour connaître la configuration matérielle du système.

Il est possible de configurer le BIOS grâce à une interface (nommée *BIOS setup*, traduisez *configuration du BIOS*) accessible au démarrage de l'ordinateur par simple pression d'une touche (généralement la touche *Suppr*). En réalité le setup du BIOS sert uniquement d'interface pour la configuration, les données sont stockées dans le *CMOS*. Pour plus d'informations n'hésitez pas à vous reporter au manuel de votre carte mère).

Le support de processeur

Le processeur (aussi appelé *microprocesseur*) est le cerveau de l'ordinateur. Il exécute les instructions des programmes grâce à un jeu d'instructions. Le processeur est caractérisé par sa fréquence, c'est-à-dire la cadence à laquelle il exécute les instructions. Ainsi, un processeur cadencé à 800 MHz effectuera grossièrement 800 millions d'opérations par seconde.

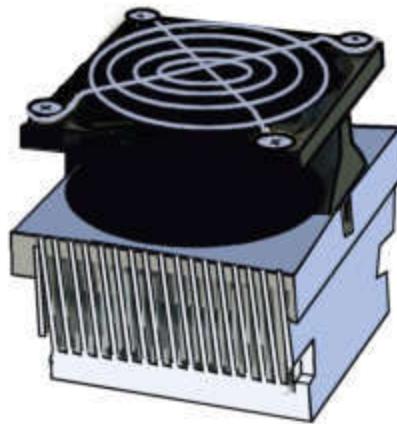
La carte mère possède un emplacement (parfois plusieurs dans le cas de cartes mères multi-processeurs) pour accueillir le processeur, appelé **support de processeur**. On distingue deux catégories de supports :

- **Slot** (en français *fente*) : il s'agit d'un connecteur rectangulaire dans lequel on enfiche le processeur verticalement
- **Socket** (en français *embase*) : il s'agit d'un connecteur carré possédant un grand nombre de petits connecteurs sur lequel le processeur vient directement s'enficher

Au sein de ces deux grandes familles, il existe des versions différentes du support, selon le type de processeur. Il est essentiel, quel que soit le support, de brancher délicatement le processeur afin de ne pas endommager aucune de ses broches (il en compte plusieurs centaines). Afin de faciliter son insertion, un support appelé

ZIF (*Zero Insertion Force*, traduisez *force d'insertion nulle*) a été créé. Les supports ZIF possèdent une petite manette, qui, lorsqu'elle est levée, permet l'insertion du processeur sans aucune pression et, lorsqu'elle est rabaisée, maintient le processeur sur son support.

Le processeur possède généralement un détrompeur, matérialisé par un coin tronqué ou une marque de couleur, devant être aligné avec la marque correspondante sur le support.



Dans la mesure où le processeur rayonne thermiquement, il est nécessaire d'en dissiper la chaleur pour éviter que ses circuits ne fondent. C'est la raison pour laquelle il est généralement surmonté d'**undissipateur thermique** (appelé parfois *refroidisseur* ou *radiateur*), composé d'un métal ayant une bonne conduction thermique (cuivre ou aluminium), chargé d'augmenter la surface d'échange thermique du microprocesseur. Le dissipateur thermique comporte une base en contact avec le processeur et des ailettes afin d'augmenter la surface d'échange thermique. Un ventilateur accompagne généralement le dissipateur pour améliorer la circulation de l'air autour du dissipateur et améliorer l'échange de chaleur. Le terme « **ventirad** » est ainsi parfois utilisé pour désigner l'ensemble *Ventilateur + Radiateur*. C'est le ventilateur du boîtier qui est chargé d'extraire l'air chaud du boîtier et permettre à l'air frais provenant de l'extérieur d'y entrer. Pour éviter les bruits liés au ventilateur et améliorer la dissipation de chaleur, il est également possible d'utiliser un système de refroidissement à eau (dit **watercooling**).

Les connecteurs de mémoire vive

La mémoire vive (*RAM* pour *Random Access Memory*) permet de stocker des informations pendant tout le temps de fonctionnement de l'ordinateur, son contenu est par contre détruit dès lors que l'ordinateur est éteint ou redémarré, contrairement à une mémoire de masse telle que le disque dur, capable de garder les informations même lorsqu'il est hors tension. On parle de « volatilité » pour désigner ce phénomène.

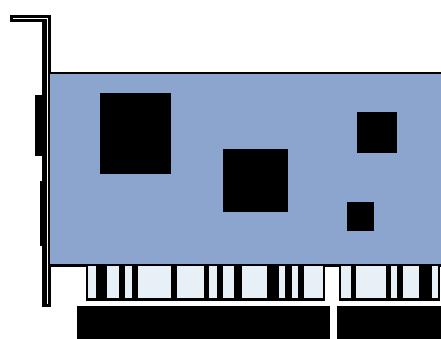
Pourquoi alors utiliser de la mémoire vive alors que les disques durs reviennent moins chers à capacité égale ? La réponse est que la mémoire vive est extrêmement rapide par comparaison aux périphériques de stockage de masse tels que le disque dur. Elle possède en effet un temps de réponse de l'ordre de quelques dizaines de nanosecondes (environ 70 pour la DRAM, 60 pour la RAM EDO, et 10 pour la SDRAM voire 6 ns sur les SDRAM DDR) contre quelques millisecondes pour le disque dur.

La mémoire vive se présente sous la forme de barrettes qui se branchent sur les connecteurs de la carte mère.

Les connecteurs d'extension

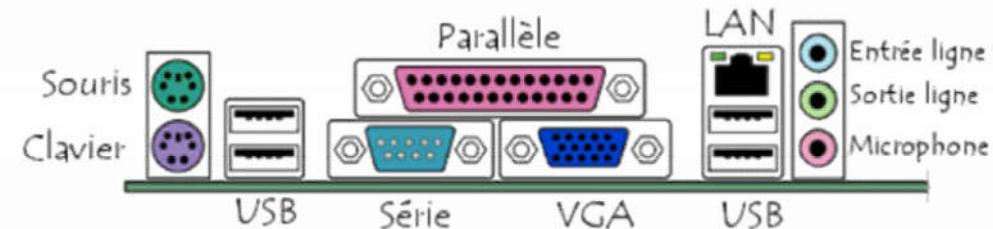
Les **connecteurs d'extension** (en anglais **slots**) sont des réceptacles dans lesquels il est possible d'insérer des cartes d'extension, c'est-à-dire des cartes offrant de nouvelles fonctionnalités ou de meilleures performances à l'ordinateur. Il existe plusieurs sortes de connecteurs :

- Connecteur ISA (*Industry Standard Architecture*) : permettant de connecter des cartes ISA, les plus lentes fonctionnant en 16-bit
- Connecteur VLB (*Vesa Local Bus*) : Bus servant autrefois à connecter des cartes graphiques
- Connecteur PCI (*Peripheral Component InterConnect*) : permettant de connecter des cartes PCI, beaucoup plus rapides que les cartes ISA et fonctionnant en 32-bit
- Connecteur AGP (*Accelerated Graphic Port*) : un connecteur rapide pour carte graphique.
- Connecteur PCI Express (*Peripheral Component InterConnect Express*) : architecture de bus plus rapide que les bus AGP et PCI.
- Connecteur AMR (*Audio Modem Riser*) : ce type de connecteur permet de brancher des mini-cartes sur les PC en étant équipés



Les connecteurs d'entrée-sortie

La carte mère possède un certain nombre de connecteurs d'entrées-sorties regroupés sur le « **panneau arrière** ».



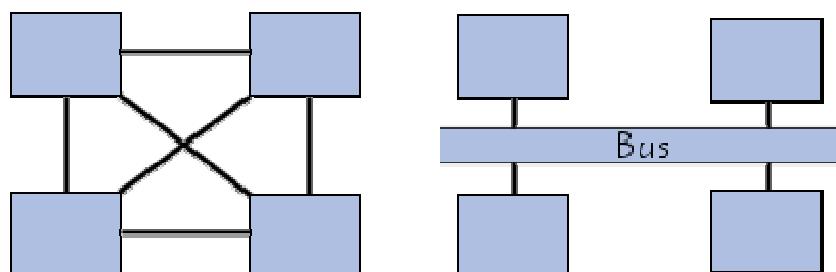
La plupart des cartes mères proposent les connecteurs suivants :

- Port série, permettant de connecter de vieux périphériques ;
- Port parallèle, permettant notamment de connecter de vieilles imprimantes ;
- Ports USB (1.1, bas débit, ou 2.0, haut débit), permettant de connecter des périphériques plus récents ;
- **Connecteur RJ45** (appelés *LAN* ou *port ethernet*) permettant de connecter l'ordinateur à un réseau. Il correspond à une *reseau.php3* carte réseau intégrée à la carte mère ;
- **Connecteur VGA** (appelé *SUB-D15*), permettant de connecter un écran. Ce connecteur correspond à *lagraphique.php3* carte graphique intégrée ;
- **Prises audio** (*entrée Line-In, sortie Line-Out et microphone*), permettant de connecter des enceintes acoustiques ou une chaîne hi fi, ainsi qu'un microphone. Ce connecteur correspond à *son.php3* carte son intégrée.

Introduction à la notion de bus

On appelle **bus**, en informatique, un ensemble de liaisons physiques (câbles, pistes de circuits imprimés, etc.) pouvant être exploitées en commun par plusieurs éléments matériels afin de communiquer.

Les bus ont pour but de réduire le nombre de « voies » nécessaires à la communication des différents composants, en mutualisant les communications sur une seule voie de données. C'est la raison pour laquelle la métaphore d'« autoroute de données » est parfois utilisée.



Dans le cas où la ligne sert uniquement à la communication de deux composants matériels, on parle de **port matériel** (port série, port parallèle, etc.).

Caractéristiques d'un bus

Un bus est caractérisé par le volume d'informations transmises simultanément. Ce volume, exprimé en **bits**, correspond au nombre de lignes physiques sur lesquelles les données sont envoyées de manière simultanée. Une nappe de 32 fils permet ainsi de transmettre 32 bits en parallèle. On parle ainsi de «**largeur**» pour désigner le nombre de bits qu'un bus peut transmettre simultanément.

D'autre part, la vitesse du bus est également définie par sa **fréquence** (exprimée en Hertz), c'est-à-dire le nombre de paquets de données envoyés ou reçus par seconde. On parle de **cycle** pour désigner chaque envoi ou réception de données.

De cette façon, il est possible de connaître le **débit** maximal du bus (ou *taux de transfert maximal*), c'est-à-dire la quantité de données qu'il peut transporter par unité de temps, en multipliant sa largeur par sa fréquence. Un bus d'une largeur de 16 bits, cadencé à une fréquence de 133 MHz possède donc un débit égal à :

$$16 * 133.10^6 = 2128 * 10^6 \text{ bit/s},$$

$$\text{soit } 2128 * 10^6 / 8 = 266 * 10^6 \text{ octets/s}$$

$$\text{soit } 266 * 10^6 / 1000 = 266 * 10^3 \text{ Ko/s}$$

$$\text{soit } 259.7 * 10^3 / 1000 = 266 \text{ Mo/s}$$

Sous-ensembles de bus

En réalité chaque bus est généralement constitué de 50 à 100 lignes physiques distinctes, classées en trois sous-ensembles fonctionnels :

- **Le bus d'adresses** (appelé parfois *bus d'adressage* ou *bus mémoire*) transporte les adresses mémoire auxquelles le processeur souhaite accéder pour lire ou écrire une donnée. Il s'agit d'un bus unidirectionnel.
- **Le bus de données** véhicule les instructions en provenance ou à destination du processeur. Il s'agit d'un bus bidirectionnel.
- **Le bus de contrôle** (parfois *bus de commandes*) transporte les ordres et les signaux de synchronisation en provenance de l'unité de commande et à destination de l'ensemble des composants matériels. Il s'agit d'un bus directionnel dans la mesure où il transmet également les signaux de réponse des éléments matériels.

Les principaux bus

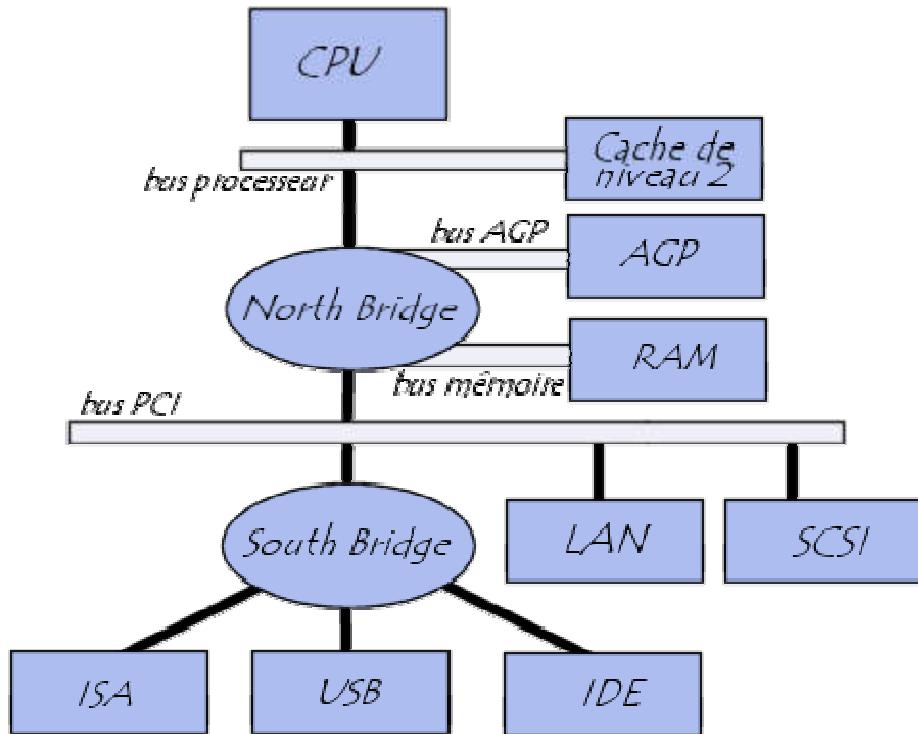
On distingue généralement sur un ordinateur deux principaux bus :

- **le bus système** (appelé aussi *bus interne*, en anglais *internal bus* ou *front-side bus*, noté *FSB*). Le bus système permet au processeur de communiquer avec la mémoire centrale du système (mémoire vive ou RAM).
- **le bus d'extension** (parfois appelé *bus d'entrée/sortie*) permet aux divers composants de la carte-mère (USB, série, parallèle, cartes branchées sur les connecteurs PCI, disques durs, lecteurs et graveurs de CD-ROM, etc.) de communiquer entre eux mais il permet surtout l'ajout de nouveaux périphériques grâce aux connecteurs d'extension (appelés **slots**) connectés sur le bus d'entrées-sorties.

Le chipset

On appelle **chipset** (en français *jeu de composants*) l'élément chargé d'aiguiller les informations entre les différents bus de l'ordinateur afin de permettre à tous les éléments constitutifs de l'ordinateur de communiquer entre eux. Le **chipset** était originalement composé d'un grand nombre de composants électroniques, ce qui explique son nom. Il est généralement composé de deux éléments :

- Le **NorthBridge** (**Pont Nord** ou **Northern Bridge**, appelé également *contrôleur mémoire*) est chargé de contrôler les échanges entre le processeur et la mémoire vive, c'est la raison pour laquelle il est situé géographiquement proche du processeur. Il est parfois appelé **GMCH**, pour *Graphic and Memory Controller Hub*.
- Le **SouthBridge** (**Pont Sud** ou **Southern Bridge**, appelé également *contrôleur d'entrée-sortie ou contrôleur d'extension*) gère les communications avec les périphériques d'entrée-sortie. Le pont sud est également appelé **ICH** (*I/O Controller Hub*). On parle généralement de **bridge** (en français *pont*) pour désigner un élément d'interconnexion entre deux bus.



Il est intéressant de noter que, pour communiquer, deux bus ont besoin d'avoir la même largeur. Cela explique pourquoi les barrettes de mémoire vive doivent parfois être appariées sur certains systèmes (par exemple sur les premiers Pentium, dont la largeur du bus processeur était de 64 bits, il était nécessaire d'installer des barrettes mémoire d'une largeur de 32 bits par paire).

Voici un tableau récapitulant les caractéristiques des principaux bus :

Norme	Largeur du bus (bits)	Vitesse du bus (MHz)	Bandé passante (Mo/sec)
ISA 8-bit	8	8.3	7.9
ISA 16-bit	16	8.3	15.9
EISA	32	8.3	31.8
VLB	32	33	127.2
PCI 32-bit	32	33	127.2
PCI 64-bit 2.1	64	66	508.6
AGP	32	66	254.3
AGP(x2 Mode)	32	66x2	528
AGP(x4 Mode)	32	66x4	1056

AGP(x8 Mode)	32	66x8	2112
ATA33	16	33	33
ATA100	16	50	100
ATA133	16	66	133
Serial ATA (S-ATA)	1		180
Serial ATA II (S-ATA2)	2		380
USB	1		1.5
USB 2.0	1		60
Firewire	1		100
Firewire 2	1		200
SCSI-1	8	4.77	5
SCSI-2 - Fast	8	10	10
SCSI-2 - Wide	16	10	20
SCSI-2 - Fast Wide 32 bits	32	10	40
SCSI-3 - Ultra	8	20	20
SCSI-3 - Ultra Wide	16	20	40
SCSI-3 - Ultra 2	8	40	40
SCSI-3 - Ultra 2 Wide	16	40	80
SCSI-3 - Ultra 160 (Ultra 3)	16	80	160
SCSI-3 - Ultra 320 (Ultra 4)	16	80 DDR	320
SCSI-3 - Ultra 640 (Ultra 5)	16	80 QDR	640

Présentation du BIOS

Le **BIOS** (« *Basic Input/Output System* » traduisez « Système de gestion élémentaire des entrées/sorties ») est un composant essentiel de l'ordinateur, permettant le contrôle des éléments matériels. Il s'agit d'un petit logiciel dont une partie est dans une ROM (mémoire morte, c'est-à-dire une mémoire qui ne peut pas être modifiée), et une autre partie est dans un EEPROM (mémoire modifiable par impulsions électriques, d'où le terme flasher pour désigner l'action de modifier l'EEPROM).

Le POST

Lorsque le système est mis sous-tension ou réamorcé (Reset), le BIOS fait l'inventaire du matériel présent dans l'ordinateur et effectue un test (appelé **POST**, pour "*Power-On Self Test*") afin de vérifier son bon fonctionnement.

- Effectuer un test du processeur (CPU)
- Vérifier le BIOS
- Vérifier la configuration du CMOS
- Initialiser le timer (l'horloge interne)
- Initialiser le contrôleur DMA
- Vérifier la mémoire vive et la mémoire cache
- Installer toutes les fonctions du BIOS
- Vérifier toutes les configurations (clavier, disquettes, disques durs ...)

Si jamais le POST rencontre une erreur, il va essayer de continuer le démarrage de l'ordinateur. Toutefois si l'erreur est grave, le BIOS va arrêter le système et :

- afficher un message à l'écran si possible (le matériel d'affichage n'étant pas forcément encore initialisée ou bien pouvant être défaillant) ;
- émettre un signal sonore, sous forme d'une séquence de bips (*beeps* en anglais) permettant de diagnostiquer l'origine de la panne ;
- envoyer un code (appelé code *POST*) sur le port série de l'ordinateur, pouvant être récupéré à l'aide d'un matériel spécifique de diagnostic.

Si tout est correct, le BIOS émettra généralement un bip bref, signalant qu'il n'y a pas d'erreur.

Signification des bips pour les BIOS Award récents		
Nb de bips	Signification	Résolution du problème
1 bip court	Le PC démarre normalement	
2 bips courts	Problème CMOS	Réinitialiser le CMOS en enlevant la pile du BIOS et en la remettant ou en déplaçant le cavalier JP4
1 bip long / 1 bip court	Problème de carte-mère ou de mémoire vive	Enficher correctement les modules de mémoire vive, tester sa RAM ou les changer
1 bip long / 2 bips courts	Problème lié à la carte graphique	Vérifier que la carte graphique est bien enfichée. Eventuellement, tester avec une autre carte vidéo
1 bip long / 3	Problème lié au clavier	Vérifier que le clavier est bien enfiché et qu'aucune

bips courts		touche n'est enfoncee. Eventuellement, tester avec un autre clavier
1 bip long / 9 bips courts	Problème du BIOS	Le BIOS est invalide, flasher le BIOS avec une version plus récente
3 bips	Problème dans les 64 premiers Ko de la RAM	La mémoire vive contient des erreurs. Essayer de la réinsérer correctement ou en changer
4 bips	Problème de rafraîchissement	La mémoire vive n'est pas rafraîchie correctement. Remettre des valeurs de rafraîchissement correctes dans le BIOS ou faire un reset du BIOS.
5 bips	Problème de processeur	Vérifier que le processeur est correctement branché, que son ventilateur fonctionne. Eventuellement, en changer.
6 bips	Problème lié au clavier	Vérifier que le clavier est bien enfoncé et qu'aucune touche n'est enfoncee. Eventuellement, tester avec un autre clavier
8 bips	Problème lié à la carte graphique	Vérifier que la carte graphique est bien enfoncée. Eventuellement, tester avec une autre carte vidéo
Bips longs incessants	Problème de mémoire vive	Enficher correctement les modules de mémoire vive, tester sa RAM ou les changer
Bips courts incessants	Problème d'alimentation	Vérifier que tous les câbles d'alimentation sont bien reliés à la carte mère, tester avec une autre alimentation ou bien en changer

Signification des bips pour les BIOS AMI (AMIBIOS)		
Nb de bips	Signification	Résolution du problème
1	Refresh failure(<i>erreur lors du rafraîchissement de la mémoire</i>)	La mémoire vive n'est pas rafraîchie correctement. Remettre des valeurs de rafraîchissement correctes dans le BIOS ou faire un reset du BIOS. Enficher correctement les modules de mémoire vive ou les changer.
2	Parity Error(<i>erreur de parité</i>)	Enficher correctement les modules de mémoire vive ou les changer. Tester sa mémoire vive.
3	Base 64K RAM failure(<i>erreur dans les 64 premiers Ko de la mémoire vive</i>)	Enficher correctement les modules de mémoire vive ou les changer. Eventuellement, flasher le BIOS.
4	System timer not operational	La carte mère doit être envoyée en réparation
5	Processor Error(<i>erreur du processeur</i>)	Vérifier que le processeur est correctement branché, que son ventilateur fonctionne. Eventuellement, en changer.
6	Gate A20 failure(<i>échec clavier</i>)	Vérifier que le clavier est bien enfoncé et qu'aucune touche n'est enfoncee. Eventuellement, tester avec un autre clavier.
7	Processor exception interrupt error(<i>erreur d'interruption du</i>	La carte mère doit être envoyée en réparation

	<i>processeur)</i>	
8	Display memory read/write failure(<i>erreur de mémoire vidéo</i>)	Vérifier que la carte graphique est bien enfichée. Eventuellement, tester avec une autre carte vidéo.
9	ROM checksum error(<i>erreur de la somme de contrôle de la mémoire morte</i>)	La puce du BIOS doit être changée ou flashée.
10	CMOS shutdown register read/write error(<i>erreur de lecture/écriture lors de l'enregistrement dans le CMOS</i>)	La carte mère doit être envoyée en réparation
11	Cache memory problem(<i>problème de mémoire cache</i>)	Vérifier que le processeur est correctement branché, que son ventilateur fonctionne. Eventuellement, en changer. Enficher correctement les modules de mémoire vive ou les changer

Signification des bips pour les BIOS Phoenix		
Nb de bips	Signification	Résolution du problème
1-3-1-1	DRAM Refresh error(<i>erreur lors du rafraîchissement de la mémoire</i>)	Enficher correctement les modules de mémoire vive ou les changer
1-2-2-3	ROM checksum error(<i>erreur de la somme de contrôle de la mémoire morte</i>)	Enficher correctement les modules de mémoire vive ou les changer
1-3-1-3	Keyboard Controller Error(<i>erreur du contrôleur de clavier</i>)	Enficher correctement le clavier ou le changer
1-3-4-1	RAM error(<i>erreur dans la mémoire</i>)	Enficher correctement les modules de mémoire vive ou les changer
1-3-4-3	RAM error(<i>erreur dans la mémoire</i>)	Enficher correctement les modules de mémoire vive ou les changer
1-4-1-1	RAM error(<i>erreur dans la mémoire</i>)	Enficher correctement les modules de mémoire vive ou les changer
2-2-3-1	Unexpected interrupt(<i>interruption inattendue</i>)	

Pour le BIOS Award, seules les erreurs relatives à la vidéo font l'objet de signaux sonores, les autres erreurs sont envoyées sous forme de codes *POST* et sont affichées à l'écran.

Ainsi un long bip, suivi de deux bips courts indique une erreur due aux périphériques vidéo (carte graphique). Dans ce cas il est nécessaire d'essayer d'enficher correctement la carte vidéo voire d'en changer. Tout autre bip indique une erreur due à la mémoire.

Voici la liste des codes POST et de la signification des bips pour les 3 principaux constructeurs de BIOS :

- **Phoenix - Phoenix BIOS POST code**
- **AMIBIOS - AMIBIOS POST code**
- **Award - BIOS Award POST code**

Le setup du BIOS

La plupart des BIOS ont un « setup » (programme de configuration) qui permet de modifier la configuration basique du système. Ce type d'information est stockée dans une mémoire autoalimentée (à l'aide d'une pile) afin que l'information soit conservée même lorsque le système est hors tension (la mémoire vive est réinitialisée à chaque redémarrage).

Il existe de nombreux BIOS dans chaque machine :

- Le BIOS de la carte mère
- Le BIOS qui contrôle le clavier
- Le BIOS de la carte vidéo
- et éventuellement
 - Le BIOS de contrôleurs SCSI qui permettent de booter sur le périphérique SCSI, qui communique alors avec le DOS sans pilote supplémentaire
 - (Le BIOS de cartes réseau qui permettent de booter sur le réseau)

Lorsque le système est mis sous tension, le BIOS affiche un message de copyright à l'écran, puis il effectue les tests de diagnostics et d'initialisation. Lorsque tous les tests ont été effectués, le BIOS affiche un message invitant l'utilisateur à appuyer sur une ou plusieurs touches afin d'entrer dans le setup du BIOS.

Selon la marque du BIOS il peut s'agir de la touche *F2*, de la touche *F10*, de la touche *DEL* (sur les claviers français : "Suppr"), ou bien d'une des séquences de touche suivantes :

- Ctrl+Alt+S
- Ctrl+Alt+Esc
- Ctrl+Alt+Ins

Sur les BIOS Award le message suivant est affiché lors du *POST* :

TO ENTER SETUP BEFORE BOOT PRESS CTRL-ALT-ESC OR DEL KEY

Ce message signifie « *PRESSEZ "CTRL-ALT-ESC" ou la touche "DEL" pour entrer dans le "SETUP" avant le démarrage du PC* »

Réinitialiser le BIOS

Dans la mesure où le setup du BIOS permet de modifier des paramètres matériels, il peut arriver que le système devienne instable, voire ne redémarre plus. Ainsi, lorsque cela arrive, il devient nécessaire d'annuler les modifications apportées au BIOS et de remettre les paramètres par défaut.

Si l'ordinateur démarre et que l'accès au setup du BIOS est possible, celui-ci offre généralement la possibilité de rétablir les paramètres par défaut. Sur les BIOS de type *PhoenixBIOS*, l'appui sur la touche *F9* permet de rétablir les paramètres par défaut du constructeur. Sur les BIOS de type *AwardBIOS* l'appui sur la touche *F5* rétablit les paramètres précédents, l'appui sur *F6* rétablit les valeurs

par défaut du BIOS Award, enfin la touche *F7* permet de rétablir les paramètres par défaut fournis par le constructeur de la carte mère.

Si l'accès au BIOS est impossible par la procédure standard, la plupart des cartes mères sont dotées d'un cavalier (jumper) leur permettant de rétablir les valeurs par défaut. Il suffit de changer la position du cavalier, et de le laisser maintenu dans cette nouvelle position pendant une dizaine de secondes.

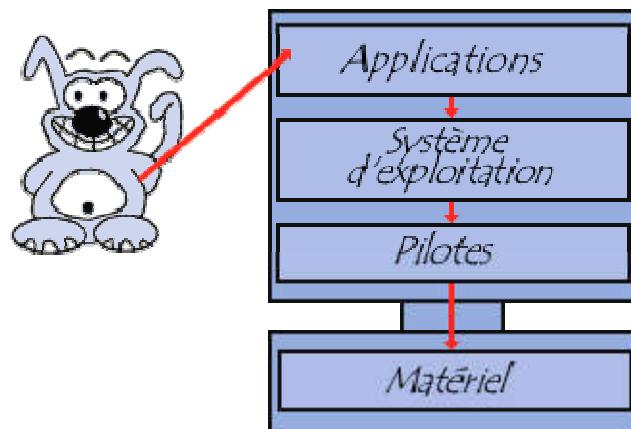
Il est fortement conseillé de procéder à ces manipulations en ayant préalablement mis l'ordinateur hors tension. Pour toutes ces manipulations référez-vous au manuel fourni avec votre carte mère !

F. Les systèmes d'exploitation

Description du système d'exploitation

Pour qu'un ordinateur soit capable de faire fonctionner un **programme informatique** (appelé parfois *application* ou *logiciel*), la machine doit être en mesure d'effectuer un certain nombre d'opérations préparatoires afin d'assurer les échanges entre le processeur, la mémoire, et les ressources physiques (périphériques).

Le **système d'exploitation** (noté *SE* ou *OS*, abréviation du terme anglais *Operating System*), est chargé d'assurer la liaison entre les ressources matérielles, l'utilisateur et les applications (traitement de texte, jeu vidéo, ...). Ainsi lorsqu'un programme désire accéder à une ressource matérielle, il ne lui est pas nécessaire d'envoyer des informations spécifiques au périphérique, il lui suffit d'envoyer les informations au système d'exploitation, qui se charge de les transmettre au périphérique concerné via son pilote. En l'absence de pilotes il faudrait que chaque programme reconnaîsse et prenne en compte la communication avec chaque type de périphérique !



Le système d'exploitation permet ainsi de "dissocier" les programmes et le matériel, afin notamment de simplifier la gestion des ressources et offrir à l'utilisateur une interface homme-machine (notée «IHM») simplifiée afin de lui permettre de s'affranchir de la complexité de la machine physique.

Rôles du système d'exploitation

Les rôles du système d'exploitation sont divers :

- **Gestion du processeur** : le système d'exploitation est chargé de gérer l'allocation du processeur entre les différents programmes grâce à un **algorithme d'ordonnancement**. Le type d'ordonnanceur est totalement dépendant du système d'exploitation, en fonction de l'objectif visé.
- **Gestion de la mémoire vive** : le système d'exploitation est chargé de gérer l'espace mémoire alloué à chaque application et, le cas échéant, à chaque usager. En cas d'insuffisance de mémoire physique, le système d'exploitation peut créer une zone mémoire sur le disque dur, appelée «**mémoire virtuelle**». La mémoire virtuelle permet de faire fonctionner des applications nécessitant plus de mémoire qu'il n'y a de mémoire vive disponible sur le système. En contrepartie cette mémoire est beaucoup plus lente.

- **Gestion des entrées/sorties** : le système d'exploitation permet d'unifier et de contrôler l'accès des programmes aux ressources matérielles par l'intermédiaire des pilotes (appelés également gestionnaires de périphériques ou gestionnaires d'entrée/sortie).
- **Gestion de l'exécution des applications** : le système d'exploitation est chargé de la bonne exécution des applications en leur affectant les ressources nécessaires à leur bon fonctionnement. Il permet à ce titre de «tuer» une application ne répondant plus correctement.
- **Gestion des droits** : le système d'exploitation est chargé de la sécurité liée à l'exécution des programmes en garantissant que les ressources ne sont utilisées que par les programmes et utilisateurs possédant les droits adéquats.
- **Gestion des fichiers** : le système d'exploitation gère la lecture et l'écriture dans le système de fichiers et les droits d'accès aux fichiers par les utilisateurs et les applications.
- **Gestion des informations** : le système d'exploitation fournit un certain nombre d'indicateurs permettant de diagnostiquer le bon fonctionnement de la machine.

Composantes du système d'exploitation

Le système d'exploitation est composé d'un ensemble de logiciels permettant de gérer les interactions avec le matériel. Parmi cet ensemble de logiciels on distingue généralement les éléments suivants :

- **Le noyau** (en anglais **kernel**) représentant les fonctions fondamentales du système d'exploitation telles que la gestion de la mémoire, des processus, des fichiers, des entrées-sorties principales, et des fonctionnalités de communication.
- **L'interpréteur de commande** (en anglais **shell**, traduisez «*coquille*» par opposition au noyau) permettant la communication avec le système d'exploitation par l'intermédiaire d'un langage de commandes, afin de permettre à l'utilisateur de piloter les périphériques en ignorant tout des caractéristiques du matériel qu'il utilise, de la gestion des adresses physiques, etc.
- **Le système de fichiers** (en anglais «*file system*», noté *FS*), permettant d'enregistrer les fichiers dans une arborescence.

Systèmes multitâches

Un système d'exploitation est dit «**multi-tâche**» (en anglais *multithreaded*) lorsque plusieurs «**tâches**» (également appelées *processus*) peuvent être exécutées simultanément.

Les applications sont composées en séquence d'instructions que l'on appelle «**processus légers**» (en anglais «*threads*»). Ces threads seront tour à tour actifs, en attente, suspendus ou détruits, suivant la priorité qui leur est associée ou bien exécutés séquentiellement.

Un système est dit **préemptif** lorsqu'il possède un **ordonnanceur** (aussi appelé *planificateur*), qui répartit, selon des critères de priorité, le temps machine entre les différents processus qui en font la demande.

Le système est dit à **temps partagé** lorsqu'un quota de temps est alloué à chaque processus par l'ordonnanceur. C'est notamment le cas des systèmes multi-utilisateurs qui permettent à plusieurs utilisateurs d'utiliser simultanément sur une même machine des applications différentes ou bien similaires : le système est alors dit «**système transactionnel**». Pour ce faire, le système alloue à chaque utilisateur une tranche de temps.

Systèmes multi-processeurs

Le **multiprocessing** est une technique consistant à faire fonctionner plusieurs processeurs en parallèle afin d'obtenir une puissance de calcul plus importante que celle obtenue avec un

processeur haut de gamme ou bien afin d'augmenter la disponibilité du système (en cas de panne d'un processeur).

On appelle **SMP** (*Symmetric Multiprocessing* ou *Symmetric Multiprocessor*) une architecture dans laquelle tous les processeurs accèdent à un espace mémoire partagé.

Un système multiprocesseur doit donc être capable de gérer le partage de la mémoire entre plusieurs processeurs mais également de distribuer la charge de travail.

Systèmes embarqués

Les **systèmes embarqués** sont des systèmes d'exploitation prévus pour fonctionner sur des machines de petite taille, telles que des PDA (*personal digital assistants* ou en français *assistants numériques personnels*) ou des appareils électroniques autonomes (sondes spatiales, robot, ordinateur de bord de véhicule, etc.), possédant une autonomie réduite. Ainsi, une caractéristique essentielle des systèmes embarqués est leur gestion avancée de l'énergie et leur capacité à fonctionner avec des ressources limitées.

Les principaux systèmes embarqués «grand public» pour assistants numériques personnels sont :

- PalmOS
- Windows CE / Windows Mobile / Window Smartphone

Systèmes temps réel

Les **systèmes temps réel** (*real time systems*), essentiellement utilisés dans l'industrie, sont des systèmes dont l'objectif est de fonctionner dans un environnement contraint temporellement. Un système temps réel doit ainsi fonctionner de manière fiable selon des contraintes temporelles spécifiques, c'est-à-dire qu'il doit être capable de délivrer un traitement correct des informations reçues à des intervalles de temps bien définis (réguliers ou non).

Voici quelques exemples de systèmes d'exploitation temps réel :

- OS-9 ;
- RTLinux (RealTime Linux) ;
- QNX ;
- VxWorks.

Les types de systèmes d'exploitation

On distingue plusieurs types de systèmes d'exploitation, selon qu'ils sont capables de gérer simultanément des informations d'une longueur de 16 bits, 32 bits, 64 bits ou plus.

Système	Codage	Mono-utilisateur	Multi-utilisateur	Mono-tâche	Multitâche
DOS	16 bits	X		X	
Windows3.1	16/32 bits	X			non préemptif
Windows95/98/Me	32 bits	X			coopératif
WindowsNT/2000	32 bits			X	préemptif
WindowsXP	32/64 bits			X	préemptif
Unix / Linux	32/64 bits			X	préemptif
MAC/OS X	32 bits			X	préemptif
VMS	32 bits			X	préemptif

Variables d'environnement

Une **variable d'environnement** est une valeur dynamique, chargée en mémoire, pouvant être utilisée par plusieurs processus fonctionnant simultanément. Sur la plupart des systèmes

d'exploitation, les emplacements de certaines librairies, voire des principaux exécutables du système peuvent avoir un emplacement différent selon l'installation.

Ainsi, grâce aux variables d'environnement, il est possible, à partir d'un programme, de faire référence à un emplacement en s'appuyant sur les variables d'environnement définissant ces données.

Sous Windows

Sous Windows, les variables d'environnement sont entourées du caractère « % ». Ainsi, pour afficher la valeur d'une variable d'environnement, il suffit de taper une commande du type : echo %NOM_DE_LA_VARIABLE%

G. Les logiciels

H. Les réseaux

Qu'est-ce qu'un réseau?

Le terme générique « **réseau** » définit un ensemble d'entités (objets, personnes, etc.) interconnectées les unes avec les autres. Un réseau permet ainsi de faire circuler des éléments matériels ou immatériels entre chacune de ces entités selon des règles bien définies.

- **réseau** (en anglais *network*) : Ensemble des ordinateurs et périphériques connectés les uns aux autres. Notons que deux ordinateurs connectés ensemble constituent à eux seuls un réseau minimal.
- **mise en réseau** (en anglais *networking*) : Mise en oeuvre des outils et des tâches permettant de relier des ordinateurs afin qu'ils puissent partager des ressources en réseau.

Selon le type d'entité concernée, le terme utilisé sera ainsi différent :

- **réseau de transport**: ensemble d'infrastructures et de disposition permettant de transporter des personnes et des biens entre plusieurs zones géographiques
- **réseau téléphonique**: infrastructure permettant de faire circuler la voix entre plusieurs postes téléphoniques
- **réseau de neurones**: ensemble de cellules interconnectées entre-elles
- **réseau de malfaiteurs**: ensemble d'escrocs qui sont en contact les uns avec les autres (un escroc en cache généralement un autre!)
- **réseau informatique**: ensemble d'ordinateurs reliés entre eux grâce à des lignes physiques et échangeant des informations sous forme de données numériques (valeurs binaires, c'est-à-dire codées sous forme de signaux pouvant prendre deux valeurs : 0 et 1)

Les présents articles s'intéressent bien évidemment aux réseaux informatiques.

Il n'existe pas un seul type de réseau, car historiquement il existe des types d'ordinateurs différents, communiquant selon des langages divers et variés. Par ailleurs ceci est également dû à l'hétérogénéité des supports physiques de transmission les reliant, que ce soit au niveau du transfert de données (circulation de données sous forme d'impulsions électriques, de lumière ou d'ondes électromagnétiques) ou bien au niveau du type de support (câble coaxial, paires torsadées, fibre optique, etc.).

Les différents chapitres s'attachent à décrire les caractéristiques des supports physiques de transmission, ainsi que la manière dont les données transitent sur le réseau.

Comment sont organisés les chapitres relatifs aux réseaux

La section réseau du site [CommentÇaMarche] est divisée en plusieurs chapitres :

- Le chapitre Initiation aux réseaux décrit ce qu'est un réseau et les différents types de réseaux qui existent
- Le chapitre Transmission de données traite de la façon selon laquelle les données sont transmises sur le support
- Le chapitre Equipements réseau décrit les différents type d'équipements présents au niveau d'une entreprise et permettant d'interconnecter les ordinateurs
- Le chapitre Protocoles explique comment l'information circule (au niveau logique) sur les réseaux, et en particulier sur Internet
- Le chapitre Technologies énumère les différents moyens physiques qui existent pour faire transiter des informations

Dans la section **Pratique**, le chapitre **Internet utile** donne des informations permettant d'apprendre à utiliser Internet!

Intérêt d'un réseau

Un ordinateur est une machine permettant de manipuler des données. L'homme, en tant qu'être communiquant, a rapidement compris l'intérêt qu'il pouvait y avoir à relier ces ordinateurs entre-eux afin de pouvoir échanger des informations.

Un réseau informatique peut servir plusieurs buts distincts :

- Le partage de ressources (fichiers, applications ou matériels, connexion à internet, etc.)
- La communication entre personnes (courrier électronique, discussion en direct, etc.)
- La communication entre processus (entre des ordinateurs industriels par exemple)
- La garantie de l'unicité et de l'universalité de l'accès à l'information (bases de données en réseau)
- Le jeu vidéo multijoueurs

Les réseaux permettent aussi de standardiser les applications, on parle généralement de groupware pour qualifier les outils permettant à plusieurs personnes de travailler en réseau. Par exemple la messagerie électronique et les agendas de groupe permettent de communiquer plus efficacement et plus rapidement. Voici un aperçu des avantages qu'offrent de tels systèmes :

- Diminution des coûts grâce aux partages des données et des périphériques,
- Standardisation des applications,
- Accès aux données en temps utile,
- Communication et organisation plus efficace.

Aujourd'hui, avec internet, on assiste à une unification des réseaux. Ainsi, les intérêts de la mise en place d'un réseau sont multiples, que ce soit pour une entreprise ou un particulier.

Similitudes entre types de réseaux

Les différents types de réseaux ont généralement les points suivant en commun :

- **Serveurs** : ordinateurs qui fournissent des ressources partagées aux utilisateurs par un serveur de réseau
- **Clients** : ordinateurs qui accèdent aux ressources partagées fournies par un serveur de réseau
- **Support de connexion** : conditionne la façon dont les ordinateurs sont reliés entre eux.
- **Données partagées** : fichiers accessibles sur les serveurs du réseau
- **Imprimantes et autres périphériques partagés** : fichiers, imprimantes ou autres éléments utilisés par les usagers du réseau
- **Ressources diverses** : autres ressources fournies par le serveur

Les différents types de réseau

On distingue généralement les deux types de réseaux suivants :

- Les réseaux poste à poste (peer to peer / égal à égal)
- Réseaux organisés autour de serveurs (Client/Serveur)

Ces deux types de réseau ont des capacités différentes. Le type de réseau à installer dépend des critères suivants :

- Taille de l'entreprise
- Niveau de sécurité nécessaire
- Type d'activité
- Niveau de compétence d'administration disponible
- Volume du trafic sur le réseau
- Besoins des utilisateurs du réseau
- Budget alloué au fonctionnement du réseau (pas seulement l'achat mais aussi l'entretien et la maintenance)

Que signifie le terme « topologie »

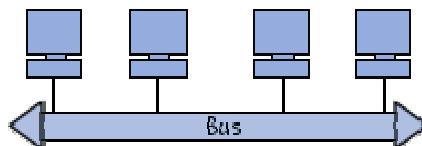
Un réseau informatique est constitué d'ordinateurs reliés entre eux grâce à des lignes de communication (câbles réseaux, etc.) et des éléments matériels (cartes réseau, ainsi que d'autres équipements permettant d'assurer la bonne circulation des données). L'arrangement physique, c'est-à-dire la configuration spatiale du réseau est appelé **topologie physique**. On distingue généralement les topologies suivantes :

- Topologie en bus
- Topologie en étoile
- Topologie en anneau
- Topologie en arbre
- Topologie maillée

La **topologie logique**, par opposition à la topologie physique, représente la façon dont les données transitent dans les lignes de communication. Les topologies logiques les plus courantes sont Ethernet, Token Ring et FDDI.

Topologie en bus

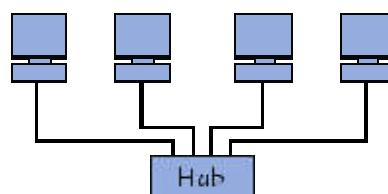
Une **topologie en bus** est l'organisation la plus simple d'un réseau. En effet, dans une topologie en bus tous les ordinateurs sont reliés à une même ligne de transmission par l'intermédiaire de câble, généralement coaxial. Le mot « bus » désigne la ligne physique qui relie les machines du réseau.



Cette topologie a pour avantage d'être facile à mettre en oeuvre et de posséder un fonctionnement simple. En revanche, elle est extrêmement vulnérable étant donné que si l'une des connexions est défectueuse, l'ensemble du réseau en est affecté.

Topologie en étoile

Dans une **topologie en étoile**, les ordinateurs du réseau sont reliés à un système matériel central appelé **concentrateur** (en anglais *hub*, littéralement *moyen de roue*). Il s'agit d'une boîte comprenant un certain nombre de jonctions auxquelles il est possible de raccorder les câbles réseau en provenance des ordinateurs. Celui-ci a pour rôle d'assurer la communication entre les différentes jonctions.

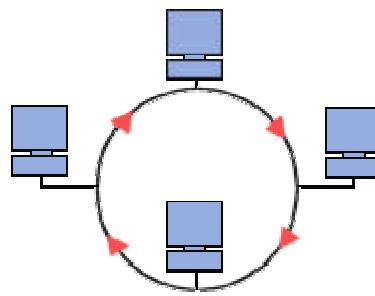


Contrairement aux réseaux construits sur une topologie en bus, les réseaux suivant une topologie en étoile sont beaucoup moins vulnérables car une des connexions peut être débranchée sans paralyser le reste du réseau. Le point névralgique de ce réseau est le concentrateur, car sans lui plus aucune communication entre les ordinateurs du réseau n'est possible.

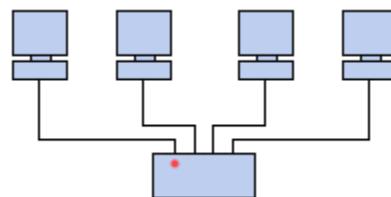
En revanche, un réseau à topologie en étoile est plus onéreux qu'un réseau à topologie en bus car un matériel supplémentaire est nécessaire (le hub).

Topologie en anneau

Dans un réseau possédant une **topologie en anneau**, les ordinateurs sont situés sur une boucle et communiquent chacun à leur tour.



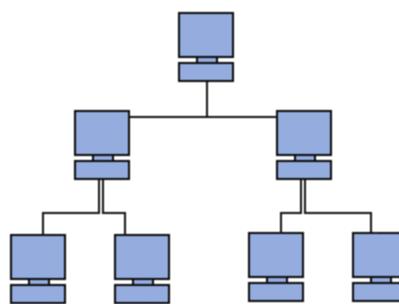
En réalité, dans une topologie anneau, les ordinateurs ne sont pas reliés en boucle, mais sont reliés à un **répartiteur** (appelé *MAU, Multistation Access Unit*) qui va gérer la communication entre les ordinateurs qui lui sont reliés en impartissant à chacun d'entre-eux un temps de parole.



Les deux principales topologies logiques utilisant cette topologie physique sont Token ring (anneau à jeton) et FDDI.

Topologie en arbre

Aussi connu sous le nom de *topologie hiérarchique*, le réseau est divisé en niveaux. Le sommet, le haut niveau, est connectée à plusieurs nœuds de niveau inférieur, dans la hiérarchie. Ces nœuds peuvent être eux-mêmes connectés à plusieurs nœuds de niveau inférieur. Le tout dessine alors un arbre, ou une arborescence.

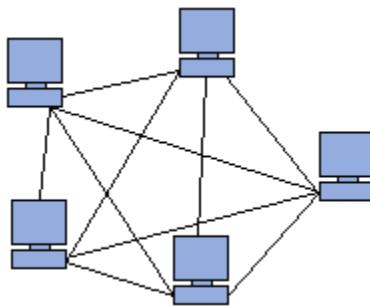


Topologie maillée

Une topologie maillée, est une évolution de la topologie en étoile, elle correspond à plusieurs liaisons point à point. Une unité réseau peut avoir (1,N) connexions point à point vers plusieurs autres unités. Chaque terminal est relié à tous les autres. L'inconvénient est le nombre de liaisons nécessaires qui devient très élevé.

Cette topologie se rencontre dans les grands réseaux de distribution (Exemple : Internet). L'information peut parcourir le réseau suivant des itinéraires divers, sous le contrôle de puissants superviseurs de réseau, ou grâce à des méthodes de routage réparties. L'armée utilise également cette topologie, ainsi, en cas de rupture d'un lien, l'information peut quand même être acheminée.

Elle existe aussi dans le cas de couverture Wi-Fi. On parle alors bien souvent de topologie mesh mais ne concerne que les routeurs WiFi.



Les différents types de réseaux

On distingue différents types de réseaux (privés) selon leur taille (en terme de nombre de machines), leur vitesse de transfert des données ainsi que leur étendue. Les réseaux privés sont des réseaux appartenant à une même organisation. On fait généralement trois catégories de réseaux :

- LAN (local area network)
- MAN (metropolitan area network)
- WAN (wide area network)

Il existe deux autres types de réseaux : les TAN (Tiny Area Network) identiques aux LAN mais moins étendus (2 à 3 machines) et les CAN (Campus Area Network) identiques au MAN (avec une bande passante maximale entre tous les LAN du réseau).

Les LAN

LAN signifie *Local Area Network* (en français *Réseau Local*). Il s'agit d'un ensemble d'ordinateurs appartenant à une même organisation et reliés entre eux dans une petite aire géographique par un réseau, souvent à l'aide d'une même technologie (la plus répandue étant Ethernet)..

Un réseau local est donc un réseau sous sa forme la plus simple. La vitesse de transfert de données d'un réseau local peut s'échelonner entre 10 Mbps (pour un réseau ethernet par exemple) et 1 Gbps (en FDDI ou Gigabit Ethernet par exemple). La taille d'un réseau local peut atteindre jusqu'à 100 voire 1000 utilisateurs.

En élargissant le contexte de la définition aux services qu'apportent le réseau local, il est possible de distinguer deux modes de fonctionnement :

- dans un environnement d'"égal à égal" (en anglais *peer to peer*), dans lequel il n'y a pas d'ordinateur central et chaque ordinateur a un rôle similaire
- dans un environnement "client/serveur", dans lequel un ordinateur central fournit des services réseau aux utilisateurs

Les MAN

Les **MAN** (*Metropolitan Area Network*) interconnectent plusieurs LAN géographiquement proches (au maximum quelques dizaines de km) à des débits importants. Ainsi un MAN permet à deux nœuds distants de communiquer comme si ils faisaient partie d'un même réseau local.

Un MAN est formé de commutateurs ou de routeurs interconnectés par des liens hauts débits (en général en fibre optique).

Les WAN

Un **WAN** (Wide Area Network ou réseau étendu) interconnecte plusieurs LANs à travers de grandes distances géographiques.

Les débits disponibles sur un WAN résultent d'un arbitrage avec le coût des liaisons (qui augmente avec la distance) et peuvent être faibles.

Les WAN fonctionnent grâce à des routeurs qui permettent de "choisir" le trajet le plus approprié pour atteindre un noeud du réseau.

Le plus connu des WAN est Internet.

Le concept de réseau privé virtuel

Les réseaux locaux d'entreprise (LAN ou RLE) sont des réseaux internes à une organisation, c'est-à-dire que les liaisons entre machines appartiennent à l'organisation. Ces réseaux sont de plus en plus souvent reliés à Internet par l'intermédiaire d'équipements d'interconnexion. Il arrive ainsi souvent que des entreprises éprouvent le besoin de communiquer avec des filiales, des clients ou même du personnel géographiquement éloignées via internet.

Pour autant, les données transmises sur Internet sont beaucoup plus vulnérables que lorsqu'elles circulent sur un réseau interne à une organisation car le chemin emprunté n'est pas défini à l'avance, ce qui signifie que les données empruntent une infrastructure réseau publique appartenant à différents opérateurs. Ainsi il n'est pas impossible que sur le chemin parcouru, le réseau soit écouté par un utilisateur indiscret ou même détourné. Il n'est donc pas concevable de transmettre dans de telles conditions des informations sensibles pour l'organisation ou l'entreprise.

La première solution pour répondre à ce besoin de communication sécurisé consiste à relier les réseaux distants à l'aide de liaisons spécialisées. Toutefois la plupart des entreprises ne peuvent pas se permettre de relier deux réseaux locaux distants par une ligne spécialisée, il est parfois nécessaire d'utiliser Internet comme support de transmission.

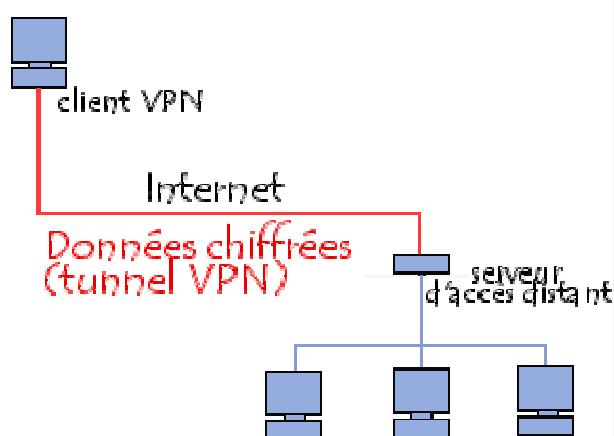
Un bon compromis consiste à utiliser Internet comme support de transmission en utilisant un protocole d'"encapsulation" (en anglais *tunneling*, d'où l'utilisation impropre parfois du terme "tunnelisation"), c'est-à-dire encapsulant les données à transmettre de façon chiffrée. On parle alors de **réseau privé virtuel** (noté **RPV** ou **VPN**, acronyme de *Virtual Private Network*) pour désigner le réseau ainsi artificiellement créé.

Ce réseau est dit *virtuel* car il relie deux réseaux "physiques" (réseaux locaux) par une liaison non fiable (Internet), et *privé* car seuls les ordinateurs des réseaux locaux de part et d'autre du VPN peuvent "voir" les données.

Le système de *VPN* permet donc d'obtenir une liaison sécurisée à moindre coût, si ce n'est la mise en oeuvre des équipements terminaux. En contrepartie il ne permet pas d'assurer une qualité de service comparable à une ligne louée dans la mesure où le réseau physique est public et donc non garanti.

Fonctionnement d'un VPN

Un réseau privé virtuel repose sur un protocole, appelé **protocole de tunnelisation** (*tunneling*), c'est-à-dire un protocole permettant aux données passant d'une extrémité du VPN à l'autre d'être sécurisées par des algorithmes de cryptographie.



Le terme de "tunnel" est utilisé pour symboliser le fait qu'entre l'entrée et la sortie du VPN les données sont chiffrées (cryptées) et donc incompréhensible pour toute personne située entre les deux extrémités du VPN, comme si les données passaient dans un tunnel. Dans le cas d'un VPN établi entre deux machines, on appelle *client VPN* l'élément permettant de chiffrer et de déchiffrer les données du côté utilisateur (client) et *serveur VPN* (ou plus généralement **serveur d'accès distant**) l'élément chiffrant et déchiffrant les données du côté de l'organisation.

De cette façon, lorsqu'un utilisateur nécessite d'accéder au réseau privé virtuel, sa requête va être transmise en clair au système passerelle, qui va se connecter au réseau distant par l'intermédiaire d'une infrastructure de réseau public, puis va transmettre la requête de façon chiffrée. L'ordinateur distant va alors fournir les données au serveur VPN de son réseau local qui va transmettre la réponse de façon chiffrée. A réception sur le client VPN de l'utilisateur, les données seront déchiffrées, puis transmises à l'utilisateur ...

Les protocoles de tunnelisation

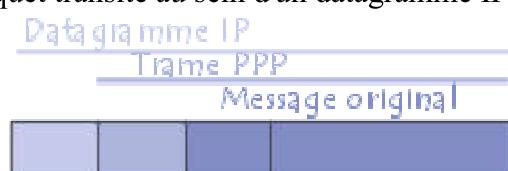
Les principaux protocoles de tunneling sont les suivants :

- **PPTP** (*Point-to-Point Tunneling Protocol*) est un protocole de niveau 2 développé par Microsoft, 3Com, Ascend, US Robotics et ECI Telematics.
- **L2F** (*Layer Two Forwarding*) est un protocole de niveau 2 développé par Cisco, Northern Telecom et Shiva. Il est désormais quasi-obsolète
- **L2TP** (*Layer Two Tunneling Protocol*) est l'aboutissement des travaux de l'IETF (RFC 2661) pour faire converger les fonctionnalités de *PPTP* et *L2F*. Il s'agit ainsi d'un protocole de niveau 2 s'appuyant sur PPP.
- **IPSec** est un protocole de niveau 3, issu des travaux de l'IETF, permettant de transporter des données chiffrées pour les réseaux IP.

Le protocole PPTP

Le principe du protocole PPTP (*Point To Point Tunneling Protocol*) est de créer des trames sous le protocole PPP et de les encapsuler dans un datagramme IP.

Ainsi, dans ce mode de connexion, les machines distantes des deux réseaux locaux sont connectés par une connexion point à point (comprenant un système de chiffrement et d'authentification, et le paquet transite au sein d'un datagramme IP).



De cette façon, les données du réseau local (ainsi que les adresses des machines présentes dans l'en-tête du message) sont encapsulées dans un message PPP, qui est lui-même encapsulé dans un message IP.

Le protocole L2TP

Le protocole L2TP est un protocole standard de tunnelisation (standardisé dans un RFC) très proche de PPTP. Ainsi le protocole L2TP encapsule des trames protocole PPP, encapsulant elles-mêmes d'autres protocoles (tels que IP, IPX ou encore NetBIOS).

Le protocole IPSec

IPSec est un protocole défini par l'IETF permettant de sécuriser les échanges au niveau de la couche réseau. Il s'agit en fait d'un protocole apportant des améliorations au niveau de la sécurité au protocole IP afin de garantir la confidentialité, l'intégrité et l'authentification des échanges.

Le protocole IPSec est basé sur trois modules :

- *IP Authentication Header (AH)* concernant l'intégrité, l'authentification et la protection contre le rejet des paquets à encapsuler
- *Encapsulating Security Payload (ESP)* définissant le chiffrement de paquets. ESP fournit la confidentialité, l'intégrité, l'authentification et la protection contre le rejet.
- *Security Association (SA)* définissant l'échange des clés et des paramètres de sécurité. Les SA rassemblent ainsi l'ensemble des informations sur le traitement à appliquer aux paquets IP (les protocoles AH et/ou ESP, mode tunnel ou transport, les algo de sécurité utilisés par les protocoles, les clés utilisées,...). L'échange des clés se fait soit de manière manuelle soit avec le protocole d'échange IKE (la plupart du temps), qui permet aux deux parties de s'entendre sur les SA.

Qu'est-ce qu'Internet?

Aux débuts de l'informatique des ordinateurs ont été mis au point, dès qu'ils furent aptes à fonctionner seuls, des personnes eurent l'idée de les relier entre eux afin qu'ils puissent échanger des données, c'est le concept de réseau. Il a donc fallu mettre au point des liaisons physiques entre les ordinateurs pour que l'information puisse circuler, mais aussi un langage de communication pour qu'il puisse y avoir un réel échange, on a décidé de nommer ce langage: *protocole*.

Sur Internet, de nombreux protocoles sont utilisés, ils font partie d'une suite de protocoles qui s'appelle TCP/IP. TCP/IP est basé sur le repérage de chaque ordinateur par une adresse appelée *adresse IP* qui permet d'acheminer les données à la bonne adresse. Puis on a associé à ces adresses des noms de domaine pour permettre de s'en souvenir plus facilement.

Des réseaux hétérogènes (de natures différentes) se sont développés aux quatre coins du globe; des personnes décidèrent donc de relier ces réseaux entre eux (des universités par exemple, ou l'armée). Les protocoles ont donc évolué pour permettre la communication de tous ces réseaux pour former le réseau des réseaux, formant petit à petit une gigantesque toile d'araignée (en anglais « web ») formant le réseau le plus vaste, puisque contenant tous les réseaux, que l'on appelle **Internet**! Sur Internet il existe différents protocoles (langages entre les ordinateurs) qui permettent de faire différentes choses :

- IRC: discuter en direct
- HTTP: regarder des pages web
- FTP: transférer des fichiers
- et bien d'autres choses

On assigne à chacun d'entre eux un numéro (le port) qui est transmis lors de la communication (la transmission est effectuée par petits paquets d'informations). Ainsi, il est possible de savoir à quel programme correspond chaque petit paquet :

- les paquets HTTP arrivent sur le port 80 et sont transmis au navigateur internet à partir duquel la page a été appelée
- les paquets IRC arrivent sur le port 6667 (ou un autre situé généralement autour de 7000) et sont transmis à un client IRC tel que mIRC (ou autre)

Se connecter à Internet

La carte réseau est l'élément de l'ordinateur qui permet de se connecter à un réseau par des lignes spécialement prévues pour faire transiter des informations numériques.

Le modem permet, lui, de se connecter à un réseau par l'intermédiaire des lignes téléphoniques, qui ne sont pas prévues à cet effet à l'origine mais qui restent le moyen de communication le plus répandu.

A la carte réseau est associée une adresse IP, permettant de caractériser l'ordinateur sur le réseau.

La connexion par l'intermédiaire d'un modem est totalement différente. En effet, un modem permet d'établir une communication entre deux ordinateurs par l'intermédiaire d'une ligne téléphonique. Vous pouvez toutefois avoir accès à un réseau (donc par extension à Internet) en contactant un ordinateur relié ("d'un côté") à une ou plusieurs lignes téléphoniques (pour recevoir l'appel) et ("de l'autre côté") à un réseau par l'intermédiaire d'une carte réseau.

Cet ordinateur appartient généralement à votre fournisseur d'accès internet (FAI). Lorsqu'il vous connecte par son intermédiaire, il prête une adresse IP que l'ordinateur gardera le temps de la connexion. A chaque connexion il attribue arbitrairement une des adresses IP libres qu'il possède. S'il est en mesure de fournir la même adresse à chaque connexion, on parle alors d'« adresse IP fixe ».

Qu'est-ce qu'un protocole?

Un **protocole** est une méthode standard qui permet la communication entre des processus (s'exécutant éventuellement sur différentes machines), c'est-à-dire un ensemble de règles et de procédures à respecter pour émettre et recevoir des données sur un réseau. Il en existe plusieurs selon ce que l'on attend de la communication. Certains protocoles seront par exemple spécialisés dans l'échange de fichiers (le FTP), d'autres pourront servir à gérer simplement l'état de la transmission et des erreurs (c'est le cas du protocole ICMP), ...

Sur Internet, les protocoles utilisés font partie d'une suite de protocoles, c'est-à-dire un ensemble de protocoles reliés entre-eux. Cette suite de protocole s'appelle TCP/IP.

Elle contient, entre autres, les protocoles suivants :

- HTTP
- FTP
- ARP
- ICMP
- IP
- TCP
- UDP
- SMTP
- Telnet
- NNTP

Protocoles orientés et non orientés connexion

On classe généralement les protocoles en deux catégories selon le niveau de contrôle des données que l'on désire :

- **Les protocoles orientés connexion:** Il s'agit des protocoles opérant un contrôle de transmission des données **pendant** une communication établie entre deux machines. dans un tel schéma, la machine réceptrice envoie des accusés de réception lors de la communication, ainsi la machine émettrice est garante de la validité des données qu'elle envoie. Les données sont ainsi envoyées sous forme de flot.TCP est un protocole orienté connexion
- **Les protocoles non orientés connexion:** Il s'agit d'un mode de communication dans lequel la machine émettrice envoie des données sans prévenir la machine réceptrice, et la machine réceptrice reçoit les données sans envoyer d'avis de réception à la première. Les données sont ainsi envoyées sous forme de blocs (datagrammes). UDP est un protocole non orienté connexion

Protocole et implémentation

Un protocole définit uniquement la façon par laquelle les machines doivent communiquer, c'est-à-dire la forme et la séquence des données à échanger. Un protocole ne définit pas la manière de programmer un logiciel de telle manière à ce qu'il soit compatible avec le protocole. On appelle ainsi **implémentation** la traduction d'un protocole en langage informatique.

Les spécifications des protocoles ne sont jamais exhaustives, aussi il est courant que les implémentations soient l'objet d'une certaine interprétation des spécifications, ce qui conduit parfois à des spécificités de certaines implémentations ou pire à des incompatibilités ou des failles de sécurité !

Qu'est-ce qu'une adresse IP

Sur Internet, les ordinateurs communiquent entre eux grâce au protocole IP (*Internet Protocol*), qui utilise des adresses numériques, appelées **adresses IP**, composées de 4 nombres entiers (4 octets) entre 0 et 255 et notées sous la forme xxx.xxx.xxx.xxx. Par exemple, 194.153.205.26 est une adresse IP donnée sous une forme technique.

Ces adresses servent aux ordinateurs du réseau pour communiquer entre-eux, ainsi chaque ordinateur d'un réseau possède une adresse IP unique sur ce réseau.

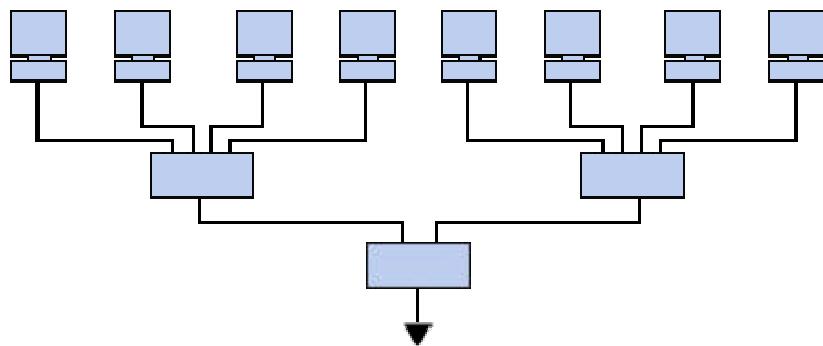
C'est l'ICANN (*Internet Corporation for Assigned Names and Numbers*, remplaçant l'IANA, *Internet Assigned Numbers Agency*, depuis 1998) qui est chargée d'attribuer des adresses IP publiques, c'est-à-dire les adresses IP des ordinateurs directement connectés sur le réseau public internet.

Déchiffrement d'une adresse IP

Une **adresse IP** est une adresse 32 bits, généralement notée sous forme de 4 nombres entiers séparés par des points. On distingue en fait deux parties dans l'adresse IP :

- une partie des nombres à gauche désigne le réseau est est appelée **ID de réseau** (en anglais *netID*),
- Les nombres de droite désignent les ordinateurs de ce réseau est est appelée **ID d'hôte** (en anglais *host-ID*).

Soit l'exemple ci-dessous :



Notons le réseau de gauche *194.28.12.0*. Il contient les ordinateurs suivants :

- 194.28.12.1 à 194.28.12.4

Notons celui de droite *178.12.0.0*. Il comprend les ordinateurs suivants :

- 178.12.77.1 à 178.12.77.6

Dans le cas ci-dessus, les réseaux sont notés *194.28.12* et *178.12.77*, puis on numérote incrémentalement chacun des ordinateurs le constituant.

Imaginons un réseau noté *58.0.0.0*. Les ordinateurs de ce réseau pourront avoir les adresses IP allant de *58.0.0.1* à *58.255.255.254*. Il s'agit donc d'attribuer les numéros de telle façon qu'il y ait une organisation dans la hiérarchie des ordinateurs et des serveurs.

Ainsi, plus le nombre de bits réservé au réseau est petit, plus celui-ci peut contenir d'ordinateurs.

En effet, un réseau noté *102.0.0.0* peut contenir des ordinateurs dont l'adresse IP peut varier entre *102.0.0.1* et *102.255.255.254* ($256*256*256-2=16777214$ possibilités), tandis qu'un réseau noté *194.26* ne pourra contenir que des ordinateurs dont l'adresse IP sera comprise entre *194.26.0.1* et *194.26.255.254* ($256*256-2=65534$ possibilités), c'est la notion de **classe d'adresse IP**.

Adresses particulières

Lorsque l'on annule la partie host-id, c'est-à-dire lorsque l'on remplace les bits réservés aux machines du réseau par des zéros (par exemple *194.28.12.0*), on obtient ce que l'on appelle **l'adresse réseau**. Cette adresse ne peut être attribuée à aucun des ordinateurs du réseau.

Lorsque la partie netid est annulée, c'est-à-dire lorsque les bits réservés au réseau sont remplacés par des zéros, on obtient l'**adresse machine**. Cette adresse représente la machine spécifiée par le host-ID qui se trouve sur le réseau courant.

Lorsque tous les bits de la partie host-id sont à 1, l'adresse obtenue est appellée l'**adresse de diffusion**(en anglais **broadcast**). Il s'agit d'une adresse spécifique, permettant d'envoyer un message à toutes les machines situées sur le réseau spécifié par le *netID*.

Enfin, l'adresse **127.0.0.1** est appelée **adresse de rebouclage** (en anglais **loopback**), car elle désigne la **machine locale** (en anglais *localhost*).

Les classes de réseaux (obsolète)

Dans le système de définition des réseau ip originel les adresses IP étaient réparties en classes, selon le nombre d'octets qui représentent le réseau, lui même déterminé par les premiers bits de l'adresse ip:

Aujourd'hui ce système a été remplacé par le CIDR au milieu des années 90 .

On avait à cette époque 3 classes pour les adresses unicast, une classe pour les adresses multidestinataires (multicast), la classe D et une classe E non utilisée:

Classe A

Dans une adresse IP de classe A, le premier octet représente le réseau.

Le bit de poids fort (le premier bit, celui de gauche) est à zéro, ce qui signifie qu'il y a 2^7 (0000000 à 0111111) possibilités de réseaux, soit 128 possibilités. Toutefois, le réseau 0 (bits valant 0000000) n'existe pas et le nombre 127 est réservé pour désigner votre machine.

Les réseaux disponibles en classe A sont donc les réseaux allant de **1.0.0.0** à **126.0.0.0** (les derniers octets sont des zéros ce qui indique qu'il s'agit bien de réseaux et non d'ordinateurs !)

Les trois octets de droite représentent les ordinateurs du réseaux, le réseau peut donc contenir un nombre d'ordinateur égal à :

$$2^{24}-2 = 16777214$$
 ordinateurs.

Une adresse IP de classe A, en binaire, ressemble à ceci :

0 xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx

Réseau Ordinateurs

Classe B

Dans une adresse IP de classe B, les deux premiers octets représentent le réseau.

Les deux premiers bits sont 1 et 0, ce qui signifie qu'il y a 2^{14} (10 000000 00000000 à 10 111111 11111111) possibilités de réseaux, soit 16384 réseaux possibles. Les réseaux disponibles en classe B sont donc les réseaux allant de **128.0.0.0** à **191.255.0.0**

Les deux octets de droite représentent les ordinateurs du réseau. Le réseau peut donc contenir un nombre d'ordinateurs égal à :

$$2^{16}-2 = 65534$$
 ordinateurs.

Une adresse IP de classe B, en binaire, ressemble à ceci :

10 xxxxxx xxxxxxxx xxxxxxxx xxxxxxxx

Réseau

Ordinateurs

Classe C

Dans une adresse IP de classe C, les trois premiers octets représentent le réseau. Les trois premiers bits sont 1,1 et 0, ce qui signifie qu'il y a 2^{21} possibilités de réseaux, c'est-à-dire 2097152. Les réseaux disponibles en classe C sont donc les réseaux allant de **192.0.0.0 à 223.255.255.0**

L'octet de droite représente les ordinateurs du réseau, le réseau peut donc contenir: $2^8 - 2^1 = 254$ ordinateurs.

Une adresse IP de classe C, en binaire, ressemble à ceci :

110 xxxxx xxxxxxxx xxxxxxxx xxxxxxxx

Réseau

Ordinateurs

Attribution des adresses IP

Le but de la division des adresses IP en trois classes A,B et C est de faciliter la recherche d'un ordinateur sur le réseau. En effet avec cette notation il est possible de rechercher dans un premier temps le réseau que l'on désire atteindre puis de chercher un ordinateur sur celui-ci. Ainsi, l'attribution des adresses IP se fait selon la taille du réseau.

Classe	Nombre de réseaux possibles	Nombre d'ordinateurs maxi sur chacun
A	126	16777214
B	16384	65534
C	2097152	254

Les adresses de classe A sont réservées aux très grands réseaux, tandis que l'on attribuera les adresses de classe C à des petits réseaux d'entreprise par exemple

Adresses IP réservées

Il arrive fréquemment dans une entreprise ou une organisation qu'un seul ordinateur soit relié à internet, c'est par son intermédiaire que les autres ordinateurs du réseau accèdent à internet (on parle généralement de proxy ou de passerelle).

Dans ce cas de figure, seul l'ordinateur relié à internet a besoin de réserver une adresse IP auprès de l'ICANN. Toutefois, les autres ordinateurs ont tout de même besoin d'une adresse IP pour pouvoir communiquer ensemble en interne.

Ainsi, l'ICANN a réservé une poignée d'adresses dans chaque classe pour permettre d'affecter une adresse IP aux ordinateurs d'un réseau local relié à internet sans risquer de créer des conflits d'adresses IP sur le réseau des réseaux. Il s'agit des adresses suivantes :

- Adresses IP privées de classe A : 10.0.0.1 à 10.255.255.254, permettant la création de vastes réseaux privés comprenant des milliers d'ordinateurs.
- Adresses IP privées de classe B : 172.16.0.1 à 172.31.255.254, permettant de créer des réseaux privés de taille moyenne.
- Adresses IP privées de classe C : 192.168.0.1 à 192.168.255.254, pour la mise en place de petits réseaux privés.

Masques de sous-réseau

Pour comprendre ce qu'est un masque, il peut-être intéressant de consulter la section « assebleur » qui parle du masquage en binaire

En résumé, on fabrique un masque contenant des 1 aux emplacements des bits que l'on désire conserver, et des 0 pour ceux que l'on veut annuler. Une fois ce masque créé, il suffit de faire un ET logique entre la valeur que l'on désire masquer et le masque afin de garder intacte la partie que l'on désire et annuler le reste.

Ainsi, un **masque réseau** (en anglais *netmask*) se présente sous la forme de 4 octets séparés par des points (comme une adresse IP), il comprend (dans sa notation binaire) des zéros aux niveau des bits de l'adresse IP que l'on veut annuler (et des 1 au niveau de ceux que l'on désire conserver).

Interet d'un masque de sous-réseau

Le premier intérêt d'un masque de sous-réseau est de permettre d'identifier simplement le réseau associé à une adresse IP.

En effet, le réseau est déterminé par un certain nombre d'octets de l'adresse IP (1 octet pour les adresses de classe A, 2 pour les adresses de classe B, et 3 octets pour la classe C). Or, un réseau est noté en prenant le nombre d'octets qui le caractérise, puis en complétant avec des 0. Le réseau associé à l'adresse 34.56.123.12 est par exemple 34.0.0.0, car il s'agit d'une adresse IP de classe A.

Pour connaître l'adresse du réseau associé à l'adresse IP 34.56.123.12, il suffit donc d'appliquer un masque dont le premier octet ne comporte que des 1 (soit 255 en notation décimale), puis des 0 sur les octets suivants.

Le masque est: 1111111.00000000.00000000.00000000

Le masque associé à l'adresse IP 34.208.123.12 est donc 255.0.0.0.

La valeur binaire de 34.208.123.12 est: 00100010.11010000.0111011.00001100

Un ET logique entre l'adresse IP et le masque donne ainsi le résultat suivant :

00100010.11010000.0111011.00001100

ET

1111111.00000000.00000000.00000000

=

00100010.00000000.00000000.00000000

Soit 34.0.0.0. Il s'agit bien du réseau associé à l'adresse 34.208.123.12

En généralisant, il est possible d'obtenir les masques correspondant à chaque classe d'adresse :

- Pour une adresse de **Classe A**, seul le premier octet doit être conservé. Le masque possède la forme suivante **1111111.00000000.00000000.00000000**, c'est-à-dire **255.0.0.0** en notation décimale ;
- Pour une adresse de **Classe B**, les deux premiers octets doivent être conservé, ce qui donne le masque suivant **1111111.1111111.00000000.00000000**, correspondant à **255.255.0.0** en notation décimale ;
- Pour une adresse de **Classe C**, avec le même raisonnement, le masque possédera la forme suivante **1111111.1111111.1111111.00000000**, c'est-à-dire **255.255.255.0** en notation décimale

Création de sous-réseaux

Reprenons l'exemple du réseau 34.0.0.0, et supposons que l'on désire que les deux premiers bits du deuxième octet permettent de désigner le réseau.

Le masque à appliquer sera alors :

1111111.11000000.00000000.00000000

C'est-à-dire 255.192.0.0

Si on applique ce masque, à l'adresse 34.208.123.12 on obtient :
34.192.0.0

En réalité il y a 4 cas de figures possibles pour le résultat du masquage d'une adresse IP d'un ordinateur du réseau 34.0.0.0

- Soit les deux premiers bits du deuxième octet sont **00**, auquel cas le résultat du masquage est **34.0.0.0**
- Soit les deux premiers bits du deuxième octet sont **01**, auquel cas le résultat du masquage est **34.64.0.0**
- Soit les deux premiers bits du deuxième octet sont **10**, auquel cas le résultat du masquage est **34.128.0.0**
- Soit les deux premiers bits du deuxième octet sont **11**, auquel cas le résultat du masquage est **34.192.0.0**

Ce masquage divise donc un réseau de classe A (pouvant admettre 16 777 214 ordinateurs) en 4 sous-réseaux - d'où le nom de *masque de sous-réseau* - pouvant admettre 2^{22} ordinateurs, c'est-à-dire 4 194 304 ordinateurs.

Le nombre de sous-réseaux dépend du nombre de bits attribués en plus au réseau (ici 2). Le nombre de sous-réseaux est donc :

Nombre de bits	Nombre de sous-réseaux
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8 (impossible pour une classe C)	256

Qu'appelle-t-on DNS ?

Chaque ordinateur directement connecté à internet possède au moins une adresse IP propre. Cependant, les utilisateurs ne veulent pas travailler avec des adresses numériques du genre *194.153.205.26* mais avec un nom de domaine ou des adresses plus explicites (appelées adresses FQDN) du type [www.commentcamarche.net].

Ainsi, il est possible d'associer des noms en langage courant aux adresses numériques grâce à un système appelé **DNS** (*Domain Name System*).

On appelle *résolution de noms de domaines* (ou *résolution d'adresses*) la corrélation entre les adresses IP et le nom de domaine associé.

Noms d'hôtes

Aux origines de TCP/IP, étant donné que les réseaux étaient très peu étendus ou autrement dit que le nombre d'ordinateurs connectés à un même réseau était faible, les administrateurs réseau créaient des fichiers appelés *tables de conversion manuelle*. Ces tables de conversion manuelle étaient des fichiers séquentiels, généralement nommés *hosts* ou *hosts.txt*, associant sur chaque ligne l'adresse IP de la machine et le nom littéral associé, appelé *nom d'hôte*.

Introduction au Domain Name System

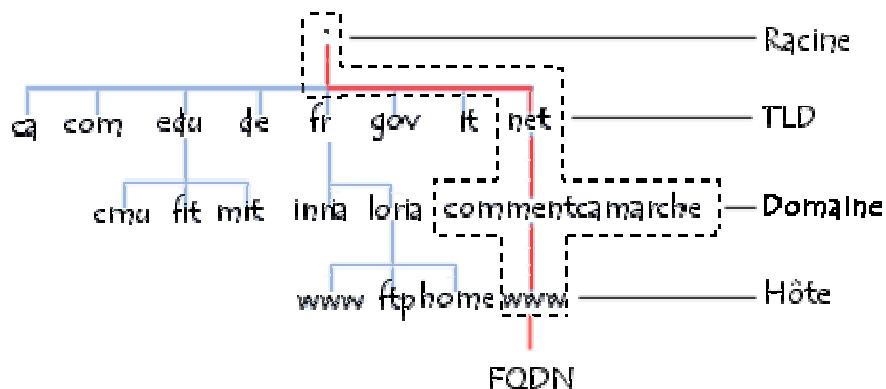
Le système précédent de tables de conversion nécessitait néanmoins la mise à jour manuelle des tables de tous les ordinateurs en cas d'ajout ou de modification d'un nom de machine. Ainsi, avec l'explosion de la taille des réseaux, et de leur interconnexion, il a fallu mettre en place un système de gestion des noms hiérarchisé et plus facilement administrable. Le système nommé **Domain Name System (DNS)**, traduisez *Système de nom de domaine*, a été mis au point en novembre 1983 par Paul Mockapetris (RFC 882 et RFC 883), puis révisé en 1987 dans les RFCs 1034 et 1035. Le DNS a fait l'objet depuis de nombreuses RFCs.

Ce système propose :

- un **espace de noms** hiérarchique permettant de garantir l'unicité d'un nom dans une structure arborescente, à la manière des systèmes de fichiers d'Unix.
- un système de **serveurs distribués** permettant de rendre disponible l'espace de noms.
- un système de **clients** permettant de « résoudre » les noms de domaines, c'est-à-dire interroger les serveurs afin de connaître l'adresse IP correspondant à un nom.

L'espace de noms

La structuration du système DNS s'appuie sur une structure arborescente dans laquelle sont définis des domaines de niveau supérieur (appelés **TLD**, pour *Top Level Domains*), rattachés à un noeud racine représenté par un point.



On appelle « **nom de domaine** » chaque noeud de l'arbre. Chaque noeud possède une étiquette (en anglais « *label* ») d'une longueur maximale de 63 caractères.

L'ensemble des noms de domaine constitue ainsi un arbre inversé où chaque noeud est séparé du suivant par un point (« . »).

L'extrémité d'une branche est appelée **hôte**, et correspond à une machine ou une entité du réseau. Le nom d'hôte qui lui est attribué doit être unique dans le domaine considéré, ou le cas échéant dans le sous-domaine. A titre d'exemple le serveur web d'un domaine porte ainsi généralement le nom *www*.

Le mot « **domaine** » correspond formellement au suffixe d'un nom de domaine, c'est-à-dire l'ensemble des étiquettes de noeuds d'une arborescence, à l'exception de l'hôte.

Le nom absolu correspondant à l'ensemble des étiquettes des noeuds d'une arborescence, séparées par des points, et terminé par un point final, est appelé **adresse FQDN** (*Fully Qualified Domain Name*, soit *Nom de Domaine Totalement Qualifié*). La profondeur maximale de l'arborescence est de 127 niveaux et la longueur maximale d'un nom FQDN est de 255 caractères. L'adresse FQDN permet de repérer de façon unique une machine sur le réseau des réseaux. Ainsi *www.commentcamarche.net.* représente une adresse FQDN.

Les serveurs de noms

Les machines appelées *serveurs de nom de domaine* permettent d'établir la correspondance entre le nom de domaine et l'adresse IP des machines d'un réseau.

Chaque domaine possède un serveur de noms de domaines, appelé « *serveur de noms primaire* » (*primary domain name server*), ainsi qu'un serveur de noms secondaire (*secondary*

domaine name server), permettant de prendre le relais du serveur de noms primaire en cas d'indisponibilité.

Chaque serveur de nom est déclaré dans à un serveur de nom de domaine de niveau immédiatement supérieur, ce qui permet implicitement une délégation d'autorité sur les domaines. Le système de nom est une architecture distribuée, où chaque entité est responsable de la gestion de son nom de domaine. Il n'existe donc pas d'organisme ayant à charge la gestion de l'ensemble des noms de domaines.

Les serveurs correspondant aux domaines de plus haut niveau (TLD) sont appelés « **serveurs de noms racine** ». Il en existe treize, répartis sur la planète, possédant les noms « a.root-servers.net » à « m.root-servers.net ».

Un serveur de noms définit une zone, c'est-à-dire un ensemble de domaines sur lequel le serveur a autorité. Le système de *noms de domaine* est transparent pour l'utilisateur, néanmoins il ne faut pas oublier les points suivants :

- Chaque ordinateur doit être configuré avec l'adresse d'une machine capable de transformer n'importe quel nom en une adresse IP. Cette machine est appelée Domain Name Server. Pas de panique: lorsque vous vous connectez à internet, le fournisseur d'accès va automatiquement modifier vos paramètres réseau pour vous mettre à disposition ces serveurs de noms.
- L'adresse IP d'un second *Domain Name Server* (secondary Domain Name Server) doit également être définie : le serveur de noms secondaire peut relayer le serveur de noms primaire en cas de dysfonctionnement.

Le serveur le plus répandu s'appelle **BIND** (*Berkeley Internet Name Domain*). Il s'agit d'un logiciel libre disponible sous les systèmes UNIX, développé initialement par l'université de Berkeley en Californie et désormais maintenu par l'*ISC* (*Internet Systems Consortium*).

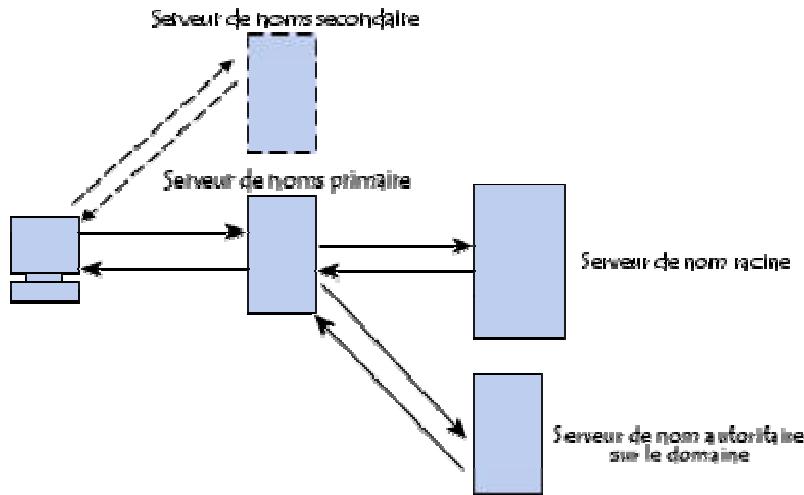
Résolution de noms de domaine

Le mécanisme consistant à trouver l'adresse IP correspondant au nom d'un hôte est appelé « **résolution de nom de domaine** ». L'application permettant de réaliser cette opération (généralement intégrée au système d'exploitation) est appelée « **résolveur** » (en anglais « *resolver* »).

Lorsqu'une application souhaite se connecter à un hôte connu par son nom de domaine (par exemple « www.commentcamarche.net »), celle-ci va interroger un serveur de noms défini dans sa configuration réseau. Chaque machine connectée au réseau possède en effet dans sa configuration les adresses IP de deux serveurs de noms de son fournisseur d'accès.

Une requête est ainsi envoyée au premier serveur de noms (appelé « serveur de nom primaire »). Si celui-ci possède l'enregistrement dans son cache, il l'envoie à l'application, dans le cas contraire il interroge un serveur racine (dans notre cas un serveur racine correspondant au TLD « .net »). Le serveur de nom racine renvoie une liste de serveurs de noms faisant autorité sur le domaine (dans le cas présent les adresses IP des serveurs de noms primaire et secondaire de *commentcamarche.net*).

Le serveur de noms primaire faisant autorité sur le domaine va alors être interrogé et retourner l'enregistrement correspondant à l'hôte sur le domaine (dans notre cas *www*).



Types d'enregistrements

Un DNS est une base de données répartie contenant des enregistrements, appelés **RR** (*Resource Records*), concernant les noms de domaines. Seules sont concernées par la lecture des informations ci-dessous les personnes responsables de l'administration d'un domaine, le fonctionnement des serveurs de noms étant totalement transparent pour les utilisateurs.

En raison du système de cache permettant au système DNS d'être réparti, les enregistrements de chaque domaine possèdent une durée de vie, appelée **TTL** (*Time To Live*, traduisez *espérance de vie*), permettant aux serveurs intermédiaires de connaître la date de péremption des informations et ainsi savoir s'il est nécessaire ou non de la vérifier.

D'une manière générale, un enregistrement DNS comporte les informations suivantes :

Nom de domaine (FQDN)	TTL	Type	Classe	RData
www.commentcamarche.net.	3600	A	IN	163.5.255.85

- **Nom de domaine** : le nom de domaine doit être un nom FQDN, c'est-à-dire être terminé par un point. Si le point est omis, le nom de domaine est relatif, c'est-à-dire que le nom de domaine principal suffixera le domaine saisi ;
- **Type** : une valeur sur 16 bits spécifiant le type de ressource décrit par l'enregistrement. Le type de ressource peut être un des suivants :
 - **A** : il s'agit du type de base établissant la correspondance entre un nom canonique et une adresse IP. Par ailleurs il peut exister plusieurs enregistrements A, correspondant aux différentes machines du réseau (serveurs).
 - **CNAME** (*Canonical Name*) : il permet de faire correspondre un alias au nom canonique. Il est particulièrement utile pour fournir des noms alternatifs correspondant aux différents services d'une même machine.

- **HINFO** : il s'agit d'un champ uniquement descriptif permettant de décrire notamment le matériel (CPU) et le système d'exploitation (OS) d'un hôte. Il est généralement conseillé de ne pas le renseigner afin de ne pas fournir d'éléments d'informations pouvant se révéler utiles pour des pirates informatiques.
- **MX (Mail eXchange)** : correspond au serveur de gestion du courrier. Lorsqu'un utilisateur envoie un courrier électronique à une adresse (utilisateur@domaine), le serveur de courrier sortant interroge le serveur de nom ayant autorité sur le domaine afin d'obtenir l'enregistrement MX. Il peut exister plusieurs MX par domaine, afin de fournir une redondance en cas de panne du serveur de messagerie principal. Ainsi l'enregistrement MX permet de définir une priorité avec une valeur pouvant aller de 0 à 65 535 :
 - **NS** : correspond au serveur de noms ayant autorité sur le domaine.
 - **PTR** : un pointeur vers une autre partie de l'espace de noms de domaines.
 - **SOA (Start Of Authority)** : le champ SOA permet de décrire le serveur de nom ayant autorité sur la zone, ainsi que l'adresse électronique du contact technique (dont le caractère « @ » est remplacé par un point).
- **Classe** : la classe peut être soit **IN** (correspondant aux protocoles d'internet, il s'agit donc du système utilisé dans notre cas), soit **CH** (pour le système chaotique) ;
- **RDATA** : il s'agit des données correspondant à l'enregistrement. Voici les informations attendues selon le type d'enregistrement :
 - A : une adresse IP sur 32 bits ;
 - CNAME : un nom de domaine ;
 - MX : une valeur de priorité sur 16 bits, suivi d'un nom d'hôte ;
 - NS : un nom d'hôte ;
 - PTR : un nom de domaine ;
 - SOA : plusieurs champs.

Domaines de haut niveau

Il existe deux catégories de **TLD** (*Top Level Domain*, soit *domaines de plus haut niveau*) :

- Les domaines dits « génériques », appelés **gTLD** (*generic TLD*). Les gTLD sont des noms de domaines génériques de niveau supérieur proposant une classification selon le secteur d'activité. Ainsi chaque gTLD possède ses propres règles d'accès :
 - gTLD historiques :
 - **.arpa** correspond aux machines issues du réseau originel ;
 - **.com** correspondait initialement aux entreprises à vocation commerciale. Désormais ce TLD est devenu le « TLD par défaut » et l'acquisition de domaines possédant cette extension est possible, y compris par des particuliers.
 - **.edu** correspond aux organismes éducatifs ;
 - **.gov** correspond aux organismes gouvernementaux ;
 - **.int** correspond aux organisations internationales ;
 - **.mil** correspond aux organismes militaires ;

- **.net** correspondait initialement aux organismes ayant trait aux réseaux. Ce TLD est devenu depuis quelques années un TLD courant. L'acquisition de domaines possédant cette extension est possible, y compris par des particuliers.
- **.org** correspond habituellement aux entreprises à but non lucratif.</gras>
- nouveaux gTLD introduits en novembre 2000 par l'ICANN :
 - **.aero** correspond à l'industrie aéronautique ;
 - **.biz (business)** correspondant aux entreprises commerciales ;
 - **.museum** correspond aux musées ;
 - **.name** correspond aux noms de personnes ou aux noms de personnages imaginaires ;
 - **.info** correspond aux organisations ayant trait à l'information ;
 - **.coop** correspondant aux coopératives ;
 - **.pro** correspondant aux professions libérales.</gras>
- gTLD spéciaux :
 - **.arpa** correspond aux infrastructures de gestion du réseau. Le gTLD arpa sert ainsi à la résolution inverse des machines du réseau, permettant de trouver le nom correspondant à une adresse IP.
- Les domaines dits «nationaux », appelés **ccTLD** (country code TLD). Les ccTLD correspondent aux différents pays et leurs noms correspondent aux abréviations des noms de pays définies par la norme ISO 3166.

L'utilité des ports

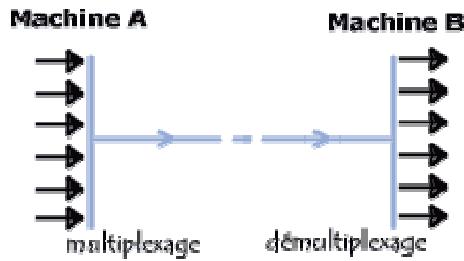
De nombreux programmes TCP/IP peuvent être exécutés simultanément sur Internet (vous pouvez par exemple ouvrir plusieurs navigateurs simultanément ou bien naviguer sur des pages HTML tout en téléchargeant un fichier par FTP). Chacun de ces programmes travaille avec un protocole, toutefois l'ordinateur doit pouvoir distinguer les différentes sources de données.

Ainsi, pour faciliter ce processus, chacune de ces applications se voit attribuer une adresse unique sur la machine, codée sur 16 bits: **un port** (la combinaison *adresse IP + port* est alors une adresse unique au monde, elle est appelée socket).

L'adresse IP sert donc à identifier de façon unique un ordinateur sur le réseau tandis que le numéro de port indique l'application à laquelle les données sont destinées. De cette manière, lorsque l'ordinateur reçoit des informations destinées à un port, les données sont envoyées vers l'application correspondante. S'il s'agit d'une requête à destination de l'application, l'application est appelée application **serveur**. S'il s'agit d'une réponse, on parle alors d'application **cliente**.

La fonction de multiplexage

Le processus qui consiste à pouvoir faire transiter sur une connexion des informations provenant de diverses applications s'appelle le multiplexage. De la même façon le fait d'arriver à mettre en parallèle (donc répartir sur les diverses applications) le flux de données s'appelle le **démultiplexage**.



Ces opérations sont réalisées grâce au port, c'est-à-dire un numéro associé à un type d'application, qui, combiné à une adresse IP, permet de déterminer de façon unique une application qui tourne sur une machine donnée.

Assignations par défaut

Il existe des milliers de ports (ceux-ci sont codés sur 16 bits, il y a donc 65536 possibilités), c'est pourquoi une assignation standard a été mise au point par l'**IANA** (*Internet Assigned Numbers Authority*), afin d'aider à la configuration des réseaux.

- Les ports 0 à 1023 sont les «**ports reconnus**» ou réservés («**Well Known Ports**»). Ils sont, de manière générale, réservés aux processus système (démêmes) ou aux programmes exécutés par des utilisateurs privilégiés. Un administrateur réseau peut néanmoins lier des services aux ports de son choix.
- Les ports 1024 à 49151 sont appelés «**ports enregistrés**» («**Registered Ports**»).
- Les ports 49152 à 65535 sont les «**ports dynamiques et/ou privés**» («**Dynamic and/or Private Ports**»).

Voici certains des ports reconnus les plus couramment utilisés :

Port Service ou Application	
21	FTP
23	Telnet
25	SMTP
53	Domain Name System
63	Whois
70	Gopher
79	Finger
80	HTTP
110	POP3
119	NNTP

Ainsi, un serveur (un ordinateur que l'on contacte et qui propose des services tels que FTP, Telnet, ...) possède des numéros de port fixes auxquels l'administrateur réseau a associé des services. Ainsi, les ports d'un serveur sont généralement compris entre 0 et 1023 (fourchette de valeurs associées à des services connus).

Du côté du client, le port est choisi aléatoirement parmi ceux disponibles par le système

d'exploitation. Ainsi, les ports du client ne seront jamais compris entre 0 et 1023 car cet intervalle de valeurs représente les *ports connus*.

Qu'est-ce qu'une URL?

Une **URL** (*Uniform Resource Locator*) est un format de nommage universel pour désigner une ressource sur Internet. Il s'agit d'une chaîne de caractères ASCII imprimables qui se décompose en cinq parties :

- **Le nom du protocole** : c'est-à-dire en quelque sorte le langage utilisé pour communiquer sur le réseau. Le protocole le plus largement utilisé est le protocole HTTP (*HyperText Transfer Protocol*), le protocole permettant d'échanger des pages Web au format HTML. De nombreux autres protocoles sont toutefois utilisables (FTP, News, Mailto, Gopher, ...)
- **Identifiant et mot de passe** : permet de spécifier les paramètres d'accès à un serveur sécurisé. Cette option est déconseillée car le mot de passe est visible dans l'URL
- **Le nom du serveur** : Il s'agit d'un nom de domaine de l'ordinateur hébergeant la ressource demandée. Notez qu'il est possible d'utiliser l'adresse IP du serveur, ce qui rend par contre l'URL moins lisible.
- **Le numéro de port** : il s'agit d'un numéro associé à un service permettant au serveur de savoir quel type de ressource est demandée. Le port associé par défaut au protocole est le port numéro 80. Ainsi, lorsque le service Web du serveur est associé au numéro de port 80, le numéro de port est facultatif
- **Le chemin d'accès à la ressource** : Cette dernière partie permet au serveur de connaître l'emplacement auquel la ressource est située, c'est-à-dire de manière générale l'emplacement (répertoire) et le nom du fichier demandé

Une URL a donc la structure suivante :

Protocole	Mot de passe (facultatif)	Nom du serveur	Port(facultatif si 80)	Chemin
http://	user:password@	www.commentcamarche.net	:80	/glossair/glossair.php3

Les protocoles suivant peuvent par exemple être utilisés par l'intermédiaire de l'URL :

- http, pour la consultation de pages web
- ftp, pour la consultation de sites FTP
- telnet, pour la connexion à un terminal distant
- mailto, pour l'envoi d'un courrier électronique
- wais
- gopher

Le nom de fichier dans l'URL peut être suivi d'un point d'interrogation puis de données au format ASCII, il s'agit de données supplémentaires envoyées en paramètre d'une application sur le serveur (un script CGI par exemple). L'URL ressemblera alors à une chaîne de caractères comme celle-ci :

<http://www.commentcamarche.net/forum/index.php3?cat=1&page=2>

Le codage d'une URL

Etant donné que l'URL est un moyen d'envoyer des informations à travers Internet (pour envoyer des données à un script CGI par exemple), il est nécessaire de pouvoir envoyer des caractères spéciaux, or les URL ne peuvent pas contenir de caractères spéciaux. De plus, certains caractères sont réservés car ils ont une signification (le slash permet de spécifier un sous-répertoires, les caractères & et ? servent à l'envoi de données par formulaires...). Enfin les URL peuvent être inclus dans un document HTML, ce qui rend difficile l'insertion de caractères tels que < ou > dans l'URL.

C'est pourquoi un codage est nécessaire ! Le codage consiste à remplacer les caractères spéciaux par le caractère % (devenant lui aussi un caractère spécial) suivi du code ASCII du caractère à coder en notation hexadécimale.

I. La programmation

Langage informatique

On appelle « **langage informatique** » un langage destiné à décrire l'ensemble des actions consécutives qu'un ordinateur doit exécuter. Un langage informatique est ainsi une façon pratique pour nous (humains) de donner des instructions à un ordinateur.

A contrario, le terme « langage naturel » représente les possibilités d'expression partagé par un groupe d'individus (par exemple l'anglais ou le français).

Les langages servant aux ordinateurs à communiquer entre eux n'ont rien à voir avec des langages informatiques, on parle dans ce cas de protocoles de communication, ce sont deux notions totalement différentes. Un langage informatique est rigoureux :

À CHAQUE instruction correspond UNE action du processeur.

Le langage utilisé par le processeur est appelé **langage machine**. Il s'agit des données telles qu'elles arrivent au processeur, constituées d'une suite de 0 et de 1 (données binaire).

Le langage machine n'est ainsi pas compréhensible par l'être humain, c'est pourquoi des langages intermédiaires, compréhensibles par l'homme, ont été mis au point. Le code écrit dans ce type de langage est transformé en langage machine pour être exploitable par le processeur.

L'assembleur est le premier langage informatique qui ait été utilisé. Celui-ci est très proche du langage machine mais reste compréhensible pour des développeurs. Toutefois, un tel langage est tellement proche du langage machine qu'il dépend étroitement du type de processeur utilisé (chaque type de processeur peut avoir son propre langage machine). Ainsi, un programme développé pour une machine ne pourra pas être *porté* sur un autre type de machine. Le terme « **portabilité** » désigne l'aptitude d'un programme informatique à être utilisé sur des machines de types différents. Pour pouvoir utiliser un programme informatique écrit en assembleur sur un autre type de machine, il sera parfois nécessaire de réécrire entièrement le programme !

Un langage informatique a donc plusieurs avantages :

- il est plus facilement compréhensible que le langage machine ;
- il permet une plus grande portabilité, c'est-à-dire une plus grande facilité d'adaptation sur des machines de types différents ;

Langages impératifs et fonctionnels

On distingue habituellement deux grandes familles de langages de programmation, selon la manière de laquelle les instructions sont traitées :

- les langages impératifs ;
- les langages fonctionnels.

Langage impératif

Un langage impératif organise le programme sous forme d'une série d'instructions, regroupées par blocs et comprenant des sauts conditionnels permettant de revenir à un bloc d'instructions si la condition est réalisée. Il s'agit historiquement des premiers langages, même si de nombreux langages modernes utilisent toujours ce principe de fonctionnement.

Les langages impératifs structurés souffrent néanmoins d'un manque de souplesse étant donné le caractère séquentiel des instructions.

Langage fonctionnel

Un **langage fonctionnel** (parfois appelé *langage procédural*) est un langage dans lequel le programme est construit par fonctions, retournant un nouvel état en sortie et prenant en entrée

la sortie d'autres fonctions. Lorsque la fonction s'appelle elle-même, on parle alors de récursivité.

Interprétation et compilation

Les langages informatiques peuvent grossièrement se classer en deux catégories :

- les **langages interprétés**
- les **langages compilés**.

Langage interprété

Un langage informatique est par définition différent du langage machine. Il faut donc le traduire pour le rendre intelligible du point de vue du processeur. Un programme écrit dans un langage interprété a besoin d'un programme auxiliaire (l'interpréteur) pour traduire au fur et à mesure les instructions du programme.

Langage compilé

Un programme écrit dans un langage dit « **compilé** » va être traduit une fois pour toutes par un programme annexe, appelé **compilateur**, afin de générer un nouveau fichier qui sera autonome, c'est-à-dire qui n'aura plus besoin d'un programme autre que lui pour s'exécuter; on dit d'ailleurs que ce fichier est **exécutable**.

Un programme écrit dans un langage compilé a comme avantage de ne plus avoir besoin, une fois compilé, de programme annexe pour s'exécuter. De plus, la traduction étant faite une fois pour toute, il est plus rapide à l'exécution.

Toutefois il est moins souple qu'un programme écrit avec un langage interprété car à chaque modification du fichier source (fichier intelligible par l'homme: celui qui va être compilé) il faudra recompiler le programme pour que les modifications prennent effet.

D'autre part, un programme compilé a pour avantage de garantir la sécurité du code source. En effet, un langage interprété, étant directement intelligible (lisible), permet à n'importe qui de connaître les secrets de fabrication d'un programme et donc de copier le code voire de le modifier. Il y a donc risque de non-respect des droits d'auteur. D'autre part, certaines applications sécurisées nécessitent la confidentialité du code pour éviter le piratage (transaction bancaire, paiement en ligne, communications sécurisées, ...).

Langages intermédiaires

Certains langages appartiennent en quelque sorte aux deux catégories (LISP, Java, Python, ..) car le programme écrit avec ces langages peut dans certaines conditions subir une phase de compilation intermédiaire vers un fichier écrit dans un langage qui n'est pas intelligible (donc différent du fichier source) et non exécutable (nécessité d'un interpréteur). Les applets Java, petits programmes insérés parfois dans les pages Web, sont des fichiers qui sont compilés mais que l'on ne peut exécuter qu'à partir d'un navigateur internet (ce sont des fichiers dont l'extension est .class).

Quelques exemples de langages couramment utilisés

Voici une liste non exhaustive de langages informatiques existants :

Langage	Domaine d'application principal	Compilé/interprété
ADA	Le temps réel	Langage compilé
BASIC	Programmation basique à but éducatif	Langage interprété
C	Programmation système	Langage compilé
C++	Programmation système objet	Langage compilé
Cobol	Gestion	Langage compilé
Fortran	Calcul	Langage compilé
Java	Programmation orientée internet	Langage intermédiaire
MATLAB	Calcul mathématique	Langage interprété

Mathematica	Calcul mathématique	Langage interprété
LISP	Intelligence artificielle	Langage intermédiaire
Pascal	Enseignement	Langage compilé
PHP	Développement de sites web dynamiques	Langage interprété
Prolog	Intelligence artificielle	Langage interprété
Perl	Traitement de chaînes de caractères	Langage interprété

Qu'est-ce qu'un programme informatique?

Un programme informatique est une succession d'instructions exécutable par l'ordinateur. Toutefois, l'ordinateur ne sait manipuler que du binaire, c'est-à-dire une succession de 0 et de 1. Il est donc nécessaire d'utiliser un langage de programmation pour écrire de façon lisible, c'est-à-dire avec des instructions compréhensibles par l'humain car proches de son langage, les instructions à exécuter par l'ordinateur.

Ainsi, ces programmes sont traduits en langage machine (en binaire) par un compilateur. La façon d'écrire un programme est intimement liée au langage de programmation que l'on a choisi car il en existe énormément. De plus, le compilateur devra correspondre au langage choisi: à chaque langage de programmation son compilateur (exception faite des langages interprétés...).

D'une façon générale, le programme est un simple fichier texte (écrit avec un traitement de texte ou un éditeur de texte), que l'on appelle **fichier source**.

Le fichier source contient les lignes de programmes que l'on appelle **code source**. Ce fichier source une fois terminé doit être compilé. La compilation se déroule en deux étapes :



- le compilateur transforme le code source en code objet, et le sauvegarde dans un **fichier objet**, c'est-à-dire qu'il traduit le fichier source en langage machine (certains compilateurs créent aussi un fichier en assembleur, un langage proche du langage machine car possédant des fonctions très simples, mais lisibles)
 - le compilateur fait ensuite appel à un **éditeur de liens** (en anglais **linker** ou **binder**) qui permet d'intégrer dans le fichier final tous les éléments annexes (fonctions ou librairies) auquel le programme fait référence mais qui ne sont pas stockés dans le fichier source. Puis il crée un **fichier exécutable** qui contient tout ce dont il a besoin pour fonctionner de façon autonome, (sous les systèmes d'exploitation Microsoft Windows ou MS-Dos le fichier ainsi créé possède l'extension .exe)

A quoi ressemble un programme informatique?

L'allure d'un programme dépend du type de langage utilisé pour faire le programme... Toutefois, à peu près tous les langages de programmation sont basés sur le même principe : Le programme est constitué d'une suite d'instructions que la machine doit exécuter. Celle-ci exécute les instructions au fur et à mesure qu'elle lit le fichier (donc de haut en bas) jusqu'à ce qu'elle rencontre une instruction (appelée parfois instruction de branchement) qui lui indique d'aller un endroit précis du programme. Il s'agit donc d'une sorte de jeu de piste dans lequel la machine doit suivre le fil conducteur et exécuter les instructions qu'elle rencontre jusqu'à ce qu'elle arrive à la fin du programme et celui-ci s'arrête.

La notion de variable

Dans la plupart des langages, on travaille généralement sur des variables, c'est-à-dire que l'on associe à un nom un contenu. On pourra ainsi appeler une variable "toto" et y stocker le chiffre 8.

Type de données

Certains langages acceptent que l'on associe à un nom de variable n'importe quel type de donnée (c'est-à-dire aussi bien un nombre entier qu'un caractère), on appelle ces langages des **langages non typés**.

En fait comme vous pourrez le voir dans le chapitre représentation des données, le type de donnée conditionne le nombre d'octets sur laquelle la donnée est codée, c'est-à-dire l'occupation en mémoire de cette donnée ainsi que le format dans lequel elle est représentée. C'est la raison pour laquelle les langages évolués (Le C, le Java) sont des **langages typés**, cela signifie qu'à une variable est associé non seulement un nom mais aussi un type de donnée qu'il faudra préciser lorsque l'on déclarera la variable, c'est-à-dire que lorsque l'on écrira le nom de la variable pour la première fois il faudra préciser au compilateur quelle sorte de données celle-ci va pouvoir contenir (la façon de déclarer la variable dépendra du langage).

Syntaxe

Les langages demandent une syntaxe rigoureuse, on ne peut donc pas écrire les choses de la manière dont on le souhaite.

Ainsi, certains langages sont **case sensitive** (en français "sensibles à la casse"), cela signifie qu'un nom ne comportant que des minuscules ne sera pas considéré comme équivalent au même nom comprenant des majuscules. Ainsi la variable nommée "Toto" sera une variable différente de la variable "toto".

Les noms de variables admettent généralement une longueur maximale (qui dépend du langage) et un jeu de caractères réduit, parmi lesquels on retrouve généralement les caractères suivants :

abcdefghijklmnoprstuvwxyz
ABCDEFGHIJKLMNPQRSTUVWXYZ
1234567890_

Ainsi, un espace (" ") est en réalité un caractère à part entière, appelé *caractère spécial*. Il est ainsi rare qu'un langage accepte des caractères spéciaux dans un nom de variable !

Mots réservés

Dans la plupart des langages, il existe une poignée de mots que l'on ne peut pas attribuer aux noms de variables, on les appelle **mots réservés** (en anglais *reserved words*). Ceux-ci seront explicités dans chaque chapitre correspondant à un langage spécifique.

Les constantes

Les constantes sont des données dont la valeur ne peut être modifiée. On les définit généralement en début de programme. La valeur que la constante contient peut être de tout type, suivant ce que le langage autorise.

Les commentaires

Il est généralement bon de pouvoir ajouter dans un programme des lignes de texte qui ne seront pas prises en compte par le compilateur. Ces lignes de textes sont généralement précédées (ou encadrées) par des instructions spéciales qui signaleront au compilateur de les ignorer.

Les commentaires servent à clarifier un programme en donnant des explications. Ils serviront si jamais une autre personne essaie de comprendre le fonctionnement du programme en lisant le fichier source, ou bien à la personne qui l'a créé si jamais il relit le fichier source quelques années après l'avoir écrit...

A quoi ressemble une instruction?

L'instruction est l'élément clé de l'ordinateur car c'est elle qui permet de spécifier au processeur l'action à effectuer. Les instructions à effectuer sont indiquées dans le fichier source et l'ordinateur passe d'une instruction à une autre en suivant les instructions indiquées de haut en bas (car la lecture d'un fichier se fait de haut en bas).

Une instruction est généralement composée de deux éléments :

- l'opérateur: action à effectuer par le processeur
- la ou les opérandes: une ou plusieurs données sur lequel on va effectuer l'opération

opérateur***opérande(s)***

Les types d'opérateurs

On distingue généralement deux ou trois types d'opérateurs :

- Les opérateurs unaires: ce sont des opérateurs qui n'admettent qu'une seule opérande
- Les opérateurs binaires: ce sont des opérateurs qui, contrairement à ce que l'on pourrait croire, ne travaillent pas sur des opérandes binaires, mais admettent deux opérandes (binaire désigne donc le nombre d'opérandes manipulées, l'addition, souvent notée +, est donc un opérateur binaire)
- Les opérateurs ternaires: ce sont des opérateurs qui admettent trois opérandes (les opérateurs conditionnels sont par exemple des opérateurs ternaires)

Les opérateurs peuvent être aussi répartis selon plusieurs catégories selon le type d'action que leur exécution déclenche :

- les opérateurs arithmétiques
- les opérateurs de comparaison
- les opérateurs logiques
- les opérateurs de bits
- les opérateurs d'affectation
- les opérateurs conditionnels
- les opérateurs séquentiels
- ...

Les priorités des opérateurs

Dans chaque langage il existe généralement des priorités d'évaluation des opérateurs, afin que l'ordinateur sache dans quel sens évaluer les opérateurs lorsque plusieurs d'entre eux sont présents dans une même expression

API

Une **API** (*Application Programmable Interface*, traduisez « *interface de programmation* » ou « *interface pour l'accès programmé aux applications* ») est un ensemble de fonctions permettant d'accéder aux services d'une application, par l'intermédiaire d'un langage de programmation.

Une API permet de fournir un certain niveau d'abstraction au développeur, c'est-à-dire qu'elle lui masque la complexité de l'accès à un système ou à une application en proposant un jeu de fonctions standard dont seuls les paramètres et les valeurs rentrées sont connus.

Ainsi, par analogie avec une voiture, le conducteur n'a pas à connaître le fonctionnement mécanique du moteur d'un véhicule pour pouvoir le conduire. Seule une interface composée

d'un volant, de pédales (accélérateur, embrayage, frein), de manettes (clignotants, phares, boîte de vitesse) et de boutons (warning, anti-brouillard, klaxon, etc.) lui sont accessibles : il s'agit d'une certaine façon de l'interface proposée à l'utilisateur.

Grâce aux API, un développeur n'a donc pas à se soucier de la façon dont une application distante fonctionne, ni de la manière dont les fonctions ont été implémentées pour pouvoir l'utiliser dans un programme. Une API peut être disponible pour un langage particulier ou bien être disponible pour plusieurs langages de programmation.

Génie logiciel

Le terme **génie logiciel** (en anglais *software engineering*) désigne l'ensemble des méthodes, des techniques et outils concourant à la production d'un logiciel, au-delà de la seule activité de programmation.

Le choix du terme « génie » fait directement référence à celui du génie civil, désignant l'art de la construction. En effet, pour construire un ouvrage architecturale, le seul fait de poser brique et ciment ne suffit pas. La construction d'un bâtiment est un tout, comprenant des activités de conception architecturale, de maçonnerie, de plomberie et d'électricité devant être coordonnées afin d'obtenir une maîtrise des délais et des budgets.

Le génie logiciel comporte donc des aspects de gestion de projet afin de produire un logiciel dans les délais prévus, avec un budget maîtrisé et donnant satisfaction au client (notion de qualité).