Structure et Technologie des Ordinateurs

A. Oumnad

aoumnad@menara.ma

Plan du cours

I INTRODUCTION

I.1 ARCHITECTURE FONCTIONNELLE

La figure 1.1 illustre la structure générale d'un ordinateur. Tous les périphériques représentés ne sont pas nécessaires au fonctionnement de l'ordinateur, et on peut, bien entendu, brancher d'autres périphériques tels que modems, scanners, traceurs, lecteurs de bandes magnétiques ...

Les logiciels (suite d'instructions exécutables par le processeur) ne sont pas représentés sur le schéma, mais il faut bien se rendre compte qu'un ordinateur sans logiciels et complètement inutile. Les notions de matériel (*Hardware*) et de logiciel (*software*) sont indissociables, l'un ne peut fonctionner sans l'autre.

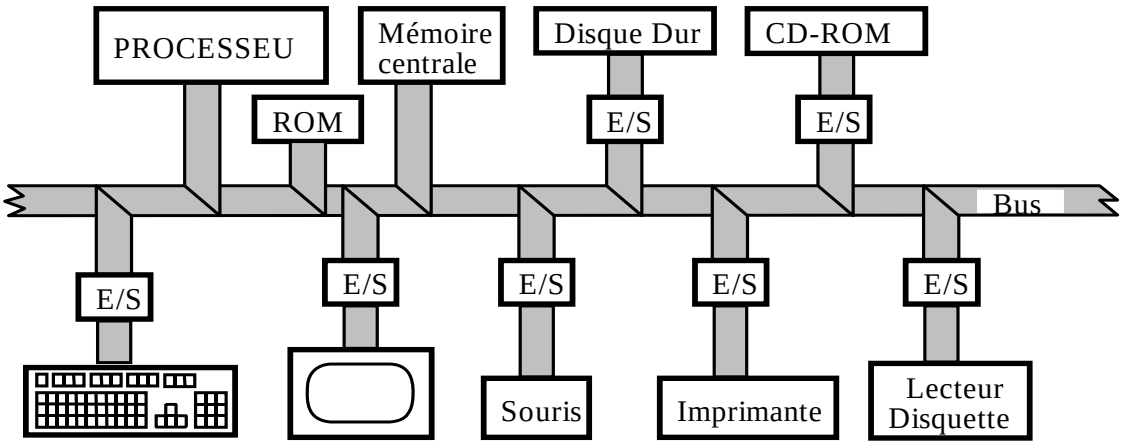


Fig. I.1 : Structure générale d'un ordinateur.

I.2 LE PROCESSEUR

Le processeur ou unité centrale (CPU : Central processing unit*)* est en quelque sorte le "cerveau" de l'ordinateur. C'est lui qui va chercher dans la mémoire centrale la suite des instructions constituant un programme et les exécute. Ces instructions peuvent être des opérations de calcul ou des opérations d'accès aux périphériques pour y déposer ou d'en ramener des données.

I.3 LA MEMOIRE CENTRALE

La mémoire centrale dite aussi mémoire de travail ou mémoire vive ou mémoire principale ou tout simplement mémoire RAM, est une mémoire volatile, c'est à dire que son contenu disparaît à l'extinction de l'ordinateur. Cette mémoire à lecture écriture contient la suite des instructions constituant le programme en cours d'exécution, elle contient aussi éventuellement les données nécessaires à l'exécution du programme. C'est aussi dans cette mémoire que seront stockés les résultats intermédiaires et définitifs de l'exécution du programme en cours.

I.4 LE DISQUE DUR

Le disque dur est une mémoire magnétique à lecture écriture non volatile à très grande capacité de stockage. C'est dans le disque que sont stockés tous les logiciels et les donnés contenus dans l'ordinateur. Parmi ces programmes, on trouve le très important système d'exploitation qui supervise tout ce qui se passe dans l'ordinateur.

I.5 LES DISQUETTES ET LES CD-ROM

Les disquettes et le CD-ROM sont des unités de stockage portables. Les disquettes sont des disques magnétiques souples réinscriptibles qui fonctionnent de la même façon que les disques dur sauf qu'ils ont une capacité de stockage plus faible, 1,44 Mo pour le format 3 pouces ¼ qui est le plus utilisé de nos jours (1997). Les CD-ROM (compact disk*)* ou disque optique compact sont des disques à très grande capacité de stockage (700 Mo) mais qui ne sont pas

réinscriptibles (ROM : Read Only Memory). L'écriture et la lecture de ces disques se font par un rayon laser, d'où l'appellation disques laser ou lecteur laser.

I.6 LE CLAVIER LA SOURIS ET L'ECRAN

Le clavier la souris et l'écran sont les périphériques qui permettent les échanges entre l'homme et la machine. Sur beaucoup de systèmes, le clavier est considéré comme l'entrée standard alors que l'écran constitue la sortie standard.

I.7 LE BUS

Le bus est le chemin par lequel transitent toutes les informations entre le processeur et les autres composantes de l'ordinateur. Il peut y avoir aussi des échanges de données entre deux composantes via le bus sans passer par le processeur.

I.8 LES UNITES D'ENTRES SORTIES (E/S)

Les unités d'entrées/sorties réalisent l'interfaçage entre le processeur et les périphériques extérieurs qui peuvent être très variés. Il est évident que la procédure pour écrire sur un disque magnétique n'est pas la même que celle qui consiste à afficher les données sur un écran. C'est le rôle de l'unité d'E/S de réaliser le "formatage" des données que lui communique le processeur avant de les envoyer vers le périphérique spécifié.

I.9 BIOS ET SYSTEME D'EXPLOITATION

Pour l'instant, on peut se contenter de dire que le fonctionnement d'un ordinateur consiste en un processeur qui exécute, instruction après instruction, un programme qui est stocké dans la mémoire centrale. Mais comme on l'a dit précédemment, tous les programmes contenus dans l'ordinateur sont stockés dans le disque dur. Donc pour exécuter un programme, il faut le charger du disque dur vers la mémoire. C'est le système d'exploitation qui s'occupe de cette tache. Or, le système d'exploitation lui même est un programme particulier qui se trouve dans le disque dur quand la machine est éteinte. Donc au démarrage de l'ordinateur, il faut un autre programme qui sait accéder au disque, qui connaît la position où est stocké le système d'exploitation, et qui le charge dans la mémoire centrale et demande au processeur de l'exécuter. Ce programme est appelé BIOS, il est stocké dans un mémoire non volatile qu'on appelle mémoire morte ou ROM qui est une mémoire électronique dont le fonctionnement est similaire à celui de la mémoire centrale mais elle est accessible seulement en lecture. Au démarrage, un mécanisme d'initialisation matériel aiguille le processeur vers cette mémoire provoquant l'exécution du bios qui charge le système d'exploitation et lui passe la main.

I.10 ARCHITECTURE MATERIELLE

Les ordinateurs de bureau les plus répandus sont constitués d'une carte mère qui constitue l'élément central sur lequel on vient brancher les éléments périphériques. La carte mère comporte toujours les éléments suivants :

- Un support pour placer le processeur
- Un ou plusieurs circuits électroniques appelés Chipset
- Une mémoire non volatile contenant un programme important appelé BIOS
- Des supports pour placer la mémoire centrale (RAM)
- Des connecteurs pour placer les cartes d'extension (PCI)
- Un bus permettant d'interconnecter le processeur avec le reste des éléments
- Un (ou +) connecteur pour brancher les disques durs et les lecteurs de CD (IDE)
- Une horloge temps réel et sa pile
- Plusieurs connecteurs pour brancher les éléments externes comme l'écran, le clavier, la souris, l'imprimante, les hauts parleurs . . .

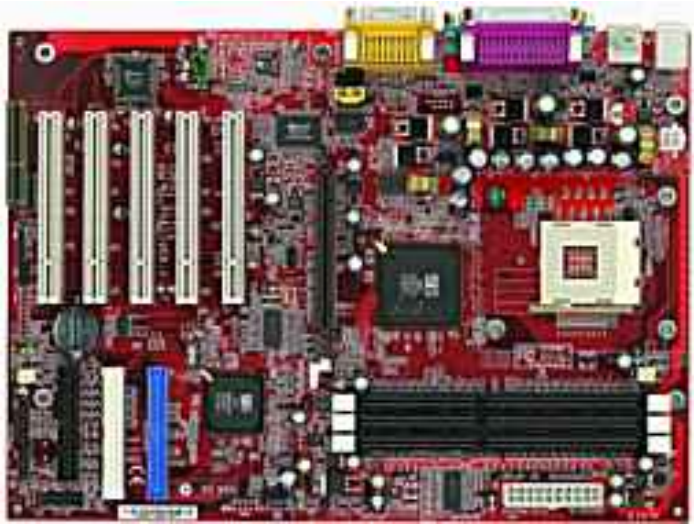


Fig. I.2 : Photos d'une carte mère

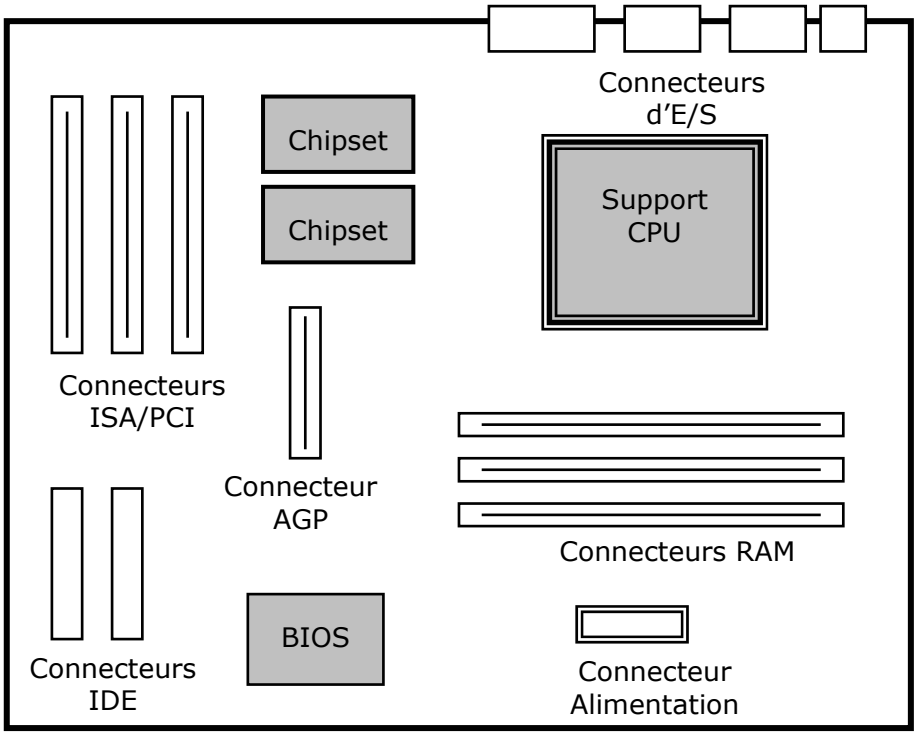


Fig. I.3 : synoptique d'une carte mère

Un des éléments important que l'on trouve sur la carte mère est le chipset. C'est L'ensemble de l'électronique qui est chargé de coordonner les échanges de données entre les divers composants de l'ordinateur. Jadis répartis sur beaucoup de circuits, le chipset est de plus en plus intégré de sorte que toutes les fonctions fondamentale on été regroupées dans un ou deux circuits intéfrés.

II ARCHITECTURE DU PROCESSEUR

Comme on l'a déjà précisé, c'est le processeur qui exécute les instructions constituants le programme. Cette tâche nécessite plusieurs interventions qui sont classées en général dans deux blocs différents. Le premier bloc **constitue l'unité de contrôle** qui va amener, une par une les instructions à partir de la mémoire centrale, analyse chaque instruction, charge éventuellement les donnée nécessaires à son exécution dans les registres et déclenche son exécution par le deuxième bloc appelé **Unité Arithmétique et Logique** *(ALU)* en lui envoyant les signaux électriques correspondant à l'opération désirée. La figure 2.1 illustre l'architecture simplifiée d'un processeur.

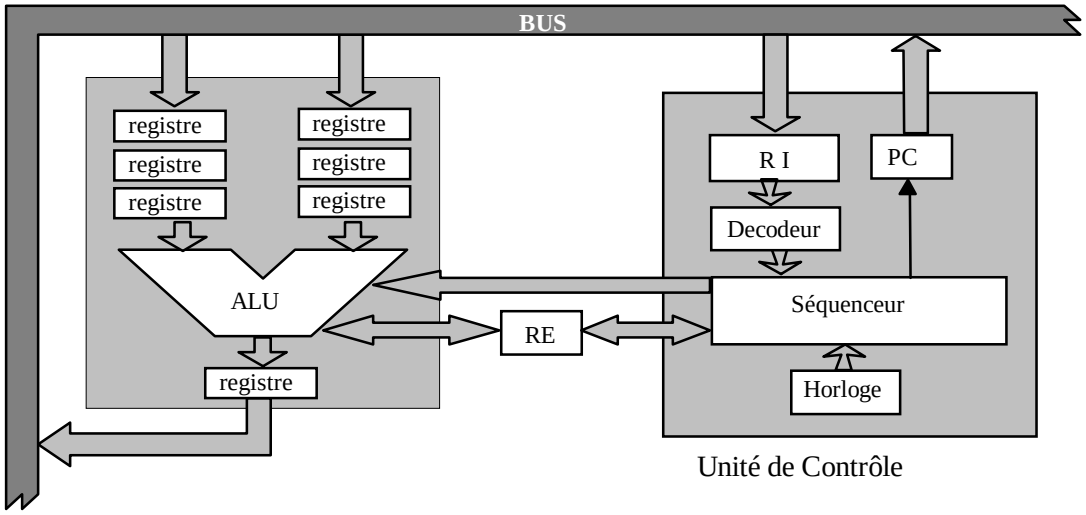


Fig. II.1 : Architecture simplifiée d'un processeur

II.1 L'UNITE DE CONTROLE

L'unité de contrôle est l'unité qui supervise le déroulement de toutes les opérations, elle contient principalement :

- Une horloge qui va permettre la synchronisation des opérations.
- Un séquenceur qui génère les signaux de commande et gère le séquencement des opérations.
- Un registre d'instruction RI où est stockée l'instruction en cours d'exécution.
- Un registre dit compteur ordinal CO *(PC : program counter)* qui contient l'adresse mémoire où est stockée la prochaine instruction à charger. Au début de l'exécution d'un programme, le CO est initialisé par le système d'exploitation à l'adresse mémoire où est stockée la première instruction du programme. Le compteur ordinal est incrémenté chaque fois qu'une instruction est chargée dans le CPU.
- Un décodeur qui va "décoder" l'instruction contenue dans RI et générer les signaux correspondant et les communiquer aux autres unités (séquenceur).

II.2 L'UNITE ARITHMETIQUE ET LOGIQUE

L'ALU contient tous les circuits électroniques qui réalisent effectivement les opérations désirées. Ces opérations sont principalement l'addition, la soustraction, la multiplication, la division ainsi que les opérations logiques comme la négation, ET, OU, OUX et les opération de décalage de bits. Le schéma de la figure 2.2 montre un exemple (74LS382) d'ALU. Les nombres A et B constituent les deux opérandes. Le nombre C constitue le code de la fonction à réaliser. Le nombre S est le résultat de l'opération. Re et Rs sont les retenues

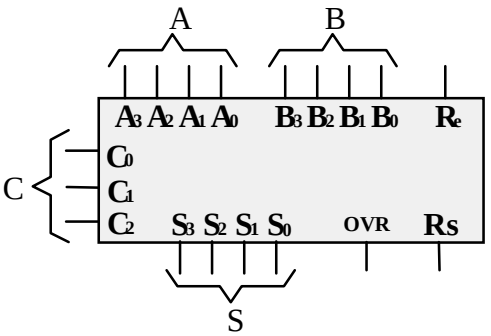


Fig. II.2 : exemple d'UAL

entrante et sortante. OVR indique qu'il y a un dépassement. Le tableau ci-dessous résume le fonctionnement de cette ALU.

$C_2C_1C_0$	Opération réalisée	
0 0 0	S = 0000	
0 0 1	S = B moins A	Opérations
0 1 0	S = A moins B	
0 1 1	S = A plus B	Arithmétiques
1 0 0	S = A + B	Opérations
1 0 1	S = A ⊕ B	
1 1 0	S = A . B	Logiques
1 1 1	S = 1111	

II.3 LES REGISTRES

Les registres sont des unités de mémorisation d'un mot machine dont la taille dépend des processeurs, on trouve des registres de 8 bits, 16 bit et 32 bits. Certains registres servent à stocker les codes d'instruction, d'autres servent à stocker les donnés constituant les opérandes et les résultats des opérations. D'autres servent à stocker les adresses permettant d'accéder aux opérandes sur la mémoire centrale.

- PC : *(program counter)* ou Compteur Ordinal CO. C'est lui qui contient l'adresse de la prochaine instruction à exécuter.

- RI : Registre d'instruction

- GPR : Ce sont des registres de travail, (general propose register) ils permettent de sauvegarder les résultats intermédiaires de calcul.

- ACC : Accumulateur. La majorité des instructions d'un CPU qui possède un accumulateur s'exécutent soit sur l'accumulateur seul soit entre l'accumulateur et un deuxième opérande. Le résultat de l'opération est stocké sur l'accumulateur. Certain CPU possèdent plusieurs accumulateurs.

- XR : Registre d'index. Ce registre peut être utilisé comme un registre GPR. En plus on dispose d'instructions permettant de l'incrémenter ou de le décrémenter. En plus il a une fonction spéciale qui d'une grande utilité dans la manipulation des tableaux de données. Il est en effet utilisé comme paramètre pour calculer les adresse suivant une forme particulière d'adressage appelée **adressage indexé**.

- RE : Registre d'état appelé aussi registre de condition. Il est formé de plusieurs bits appelés drapeaux qui sont positionné par l'ALU après chaque opération. On dispose d'un jeux d'instruction conditionnées par l'état de différents drapeaux. Par exemple le bit indicateur Z indique quand il est positionné que le résultat de l'opération est égal à Zéro. Le bit C indique que l'opération éxécutée à produit une retenue. Lz bit O indique un dépassement de capacité dans l'ACC est que le résultat est faux. Le bit N indique que le résultat est négatif . . .

- SP : pointeur de pile (Stack Pointer). Ce registre pointe (contient l'adresse) sur une zone mémoire qu'on appelle pile (LIFO) dans laquelle le processeur (ou le programmeur) peut sauvegarder momentanément des données sans être obliger de gérer le problème d'adressage.

II.4 LE DEROULEMENT D'UNE INSTRUCTION

Le déroulement d'une instruction se fait en deux cycles, un cycle de recherche *(fetch cycle)* et un cycle d'exécution.

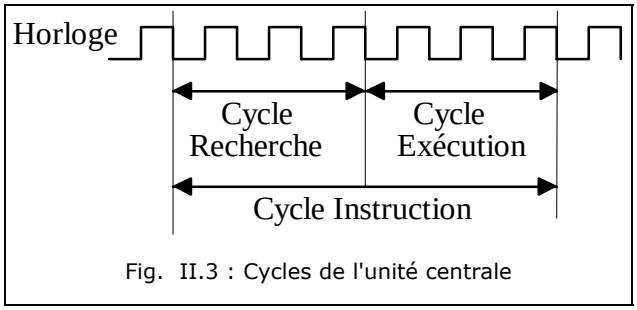


Fig. II.3 : Cycles de l'unité centrale

Pendant le cycle de **recherche**, l'instruction est chargée de la mémoire vers le registre d'instruction RI

Pendant le cycle **d'exécution**,

- L'instruction est analysée par le décodeur pour déterminer la nature de l'opération ainsi que le nombre et le type des opérandes.
- Ces informations sont communiquées au séquenceur qui charge les opérandes dans les registres approprié et envoi à l'ALU les signaux de commandes lui indiquant l'opération à effectuer.
- L'ALU exécute l'opération et positionne les indicateurs du registre d'état

Le temps d'exécution d'une instruction dépend du type de l'opération à exécuter. Un cycle d'instruction peut s'étendre sur plusieurs cycles machines ou périodes d'horloge. Le terme **cycle CPU** est utilisé pour indiquer le temps d'exécution de l'instruction la plus courte.

II.5 STRUCTURE D'UNE INSTRUCTION

Les ordinateurs sont capables de faire un certain nombre d'opérations simple, par exemple additionner deux nombres, tester le signe d'une valeur numérique, copier le contenu d'un registre dans un autre, stocker le contenu d'un registre dans une mémoire ou charger le contenu d'une mémoire dans un registre. Une instruction machine doit donc contenir un code opération et un ensemble d'adresses. Si on exclut les opérations entre registres, une instruction ressemblerait à CO ad1 ad2 ad3 :

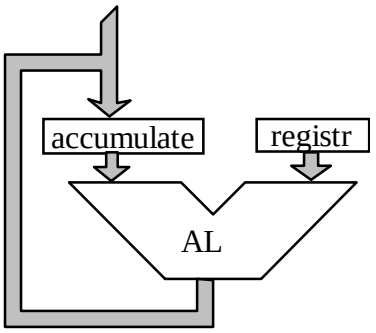


Fig. II.4 : ALU avec accumulateur

CO : Code Opération

Ad1 et ad2 : adresses des opérandes

Ad3 : adresse ou doit être estoqué le résultat

Pour simplifier les programmes, les concepteurs de microprocesseurs ont opté pour une architecture avec des instructions à **une** adresse. Ceci est rendu possible par l'utilisation d'un registre spécial appelé **accumulateur.** Les opérations se font entre l'accumulateur et un autre opérande dont l'adresse est précisée. Le résultat de l'opération est stocké dans l'accumulateur. L'instruction sera donc constituée seulement de deux champs CO | AD. Mais comme les processeurs avec un seul accumulateur sont un peu limités, on multiplie le nombre d'accumulateurs mais on a de nouveau des instructions à 3 champs puisqu'il faut préciser l'accumulateur utilisé CO Ax AD

Pour une machine 8 bits, le code opération est un nombre de 8 bits (1 octet) qu'on représente le plus souvent en Hexadécimal (2 chiffres). Le champ adresse quant à lui, peut être constitué de plusieurs octets, cela dépend de la taille mémoire. On aura par exemple des instructions du genre 3A 24 5678 (tous les nombres sont en hexadécimal)

3A = 0011 1010 = code de l'Opération

24 = 0010 0100 = code de l'accumulateur utilisé

5678 = 0101 0110 0111 1000 = adresse du deuxième opérande

Pour faciliter la lecture des programmes par « l'homme », on associe une mnémonique à chaque code opération, des noms pour les registres et on adopte une sorte de grammaire pour l'écriture des instructions, obtient des instructions du genre :

ADD AX, [3456] additionner le contenu de l'accumulateur AX avec le contenu de la case mémoire d'adresse 3456 avec résultat dans l'accumulateur

ADD [1258],BX additionner le contenu de l'accumulateur BX avec le contenu de la case mémoire d'adresse 1258 avec résultat dans la case mémoire

III LES MEMOIRES

Une mémoire est un dispositif capable d'enregistrer, de conserver et de restituer des informations codées en binaire dans un ordinateur.

III.1 HIERARCHIE DES MEMOIRES PAR PERFORMANCE

Les mémoires d'un ordinateur se répartissent en plusieurs niveaux caractérisés par leur temps d'accès et leur capacité. La figure 3.1 illustre cette hiérarchie. Plus on s'éloigne du CPU, plus le temps d'accès aux mémoires augmente ainsi que leur capacité alors que le coût par bit diminue.

- Les éléments situés dans l'unité centrale sont les registres, ils sont très rapides et servent principalement au stockage des opérandes et des résultats intermédiaires.
- La mémoire cache est une mémoire rapide de faible capacité. La mémoire cache est le plus souvent intégrée au processeur pour qu'elle soit la plus rapide possible. On peut avoir une partie sur le processeur (On chip cache) et une partie hors du processeur.
- La mémoire centrale et l'organe principal de rangement des informations utilisées par le CPU. C'est une mémoire à semi-conducteurs, son temps d'accès est beaucoup plus grand que celui des registres et de la mémoire cache.
- Les mémoires de masse sont des mémoires périphériques de grande capacité et de coût relativement faible. Elles servent d'éléments de stockage permanent et utilisent pour cela des supports magnétiques (disques, bandes) et des support optiques (disques optiques).

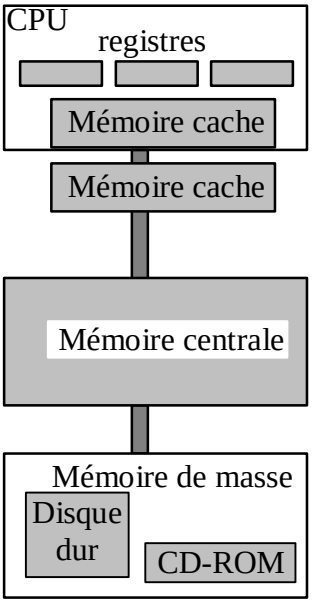


Fig. III.1 : hiérarchie des mémoires

Registres	1 - 2 ns	32 - 512 octets
On-chip cache	3 - 10 ns	1 - 256 Ko
Off-chip cache (SRAM)	5 - 50 ns	64 - 256 Ko
Mémoire centrale (DRAM)	50 - 250 ns	1 Mo - 1 Go
Mémoire secondaire (disk)	5 - 20 ms	100 Mo - 1 To
Mémoire tertiaire (CD-ROM)	100 - 500 ms	600 Mo - qq Go

tab. III-1 : Quelques ordres de grandeurs

III.2 LES MEMOIRES A SEMI-CONDUCTEURS

Il existe de nombreuses variétés de mémoire à semi-conducteurs. Cette diversité vient du fait que la mémoire idéale à grande capacité, consommant peu d'énergie, de vitesse élevée, gardant son information en cas de coupure d'alimentation, n'existe pas. Les différentes catégories de mémoire que nous rencontrerons sont des compromis sur quelques paramètres indispensables, nous allons les classer en fonction de leurs utilisations.

III.2.1 Mémoire vive ou RAM

La mémoire vive est une mémoire dans laquelle on peut écrire ou lire une information. En anglais on la désigne sous le sigle RAM (Random Access Memory), mémoire à accès aléatoire, cela signifie qu'après avoir lu ou écrit dans une position mémoire, on peut lire ou écrire dans une autre position quelconque. Ceci par opposition avec le s mémoire à accès séquentiel (série), dans lesquels après avoir lu ou écrit dans une position mémoire, la prochaine opération de lecture/écriture ne peut porter que sur la position mémoire immédiatement voisine. Remarquons que la nomenclature RWM *(read write memory)* aurait été plus appropriée.

Le contenu d'une mémoire vive s'efface quand la tension d'alimentation disparaît, d'où la qualification de mémoire **volatile**.

On distingue les RAMs statiques et les RAMs dynamiques :

- Le taux d'intégration des RAM statique est assez faible et leur prix de revient (au Mbits) reste relativement élevé, par contre, leur temps d'accès est faible. Elles sont utilisées dans

les mémoires caches (interne et externe)
- Le taux d'intégration des RAM dynamique est élevé et leur prix de revient (au Mbits) est plus faible mais leur temps d'accès est assez élevé. Elles sont utilisées dans la mémoire centrale.

III.2.2 Mémoire Morte ou ROM

L'utilisateur ne peut que lire le contenu de cette mémoire. Elle est inscrite par le constructeur au moment de la fabrication selon les spécifications du client. On utilise ce genre de mémoire quand l'information qu'on y enregistre est une information figée qui n'est pas susceptible de subir un changement, comme par exemple les valeurs de la fonction sinus pour les angles compris entre 0 et 90°. S'il arrive malgré tout qu'on soit obligé de changer le contenu, il faut commander un autre boîtier au constructeur, ce qui demande beaucoup de temps (plusieurs semaines).

L'utilisation des ROM ne devient intéressante que si le nombre de boîtiers identiques est grand (plusieurs milliers), compte tenu du coût de développement initial (masque du contenu de la mémoire). Le gros avantage des mémoires ROM est de conserver leur contenu après une coupure d'alimentation, elle fait partie des mémoires non volatiles.

III.2.3 Mémoire morte programmable ou PROM

Lorsque l'information que l'on désire enregistrer dans une mémoire non volatile est susceptible de varier de temps en temps (comme un programme qu'on met au point par exemple), l'utilisation des ROM ne convient plus. On utilise alors des mémoires PROM programmable par l'utilisateur au moyen d'un dispositif adéquat appelé programmateur de PROM. Si après inscription et utilisation, le contenu s'avère inexact, on jette le boîtier et on en reprogramme un autre. L'opération prend quelques minutes.

Comme les mémoires ROM, le contenu des PROM ne s'efface pas après coupure d'alimentation.

III.2.4 Mémoire morte reprogrammable ou EPROM

Avec les PROM, pour changer le contenu, il faut jeter le boîtier est reprogrammer un nouveau. Cela peut devenir gênant du point de vue financier si les modifications deviennent trop fréquentes. Les mémoires EPROM (Electrically Programmable Read Only Memory) appelées aussi ROM effaçables, constitue une solution à ce problème. Quand on veut changer le contenu d'un boîtier, on n'est pas obligé de le jeter, on peut effacer son contenu en l'exposant aux rayons ultraviolets à travers une fenêtre de quartz placée sur le boîtier, puis enregistrer électriquement les nouvelles informations en appliquant des tensions plus élevées que la tension d'alimentation normale. L'effacement par ultraviolets dure une vingtaine de minutes. Comme pour les mémoires ROM, l'intégrité de l'information est conservée après disparition de l'alimentation.

III.2.5 Mémoire morte efaçable électriquement ou EEPROM

Ces mémoire non volatiles présentent l'avantage d'être inscriptible électriquement et effaçable électriquement d'où leur nom EEPROM (Electrically erasable programmable Read Only Memory) . Cela permet de gagner du temps car l'effacement électrique prend beaucoup mois de temps que l'effacement par ultraviolets.

III.2.6 Mémoire FLASH

Les mémoires flash sont des EEPROM à accès rapide. L'accès en lecture est comparable à celui des RAMs (<= 100 ns). L'accès en écriture est plus long (<= 10 µs). On distingue des variantes à accès parallèle et d'autres à accès série. Sur les ordinateurs, elles sont utilisées surtout pour le stockage du bios. Ailleurs, ces mémoires sont utilisées dans beaucoup d'applications et sont promues à un avenir très prometteur. Les cartes à puces en sont fournies et elles remplacent déjà les Disques durs sur certains ordinateurs portables.

III.3 TECHNOLOGIES DES MEMOIRES

L'unité de base de stockage et la cellule qui permet de stocker un bit. Sa structure dépend du type de mémoire. Le regroupement de 8 cellules donne une case mémoire qui peut stocker un octet.

III.3.1 Cellule statique d'une mémoire vive

Dans ce cas, l'information est stockée dans une bascule comme une bascule D par exemple. Comme on le sait une fois la sortie de la bascule est dans un état, elle y restera tant qu'en ne vient pas la changer en mettant le bit à enregistrer sur l'entrée D et en envoyant un coup d'horloge sur son entrée horloge. Les mémoires ainsi construites sont appelées les RAM **Statiques (SRAM)**. Toutes les bascules (D, R-S, J-K) avec ou sans horloge peuvent servir de point de mémorisation, mais pour des raisons d'encombrement, on utilise des bascules bistables constituées de 6 transistors MOS ou de 2 transistors bipolaires. Les MOS sont plus utilisés du fait de leur facilité d'intégration et de leur faible consommation. Même cette solution reste trop encombrante ce qui fait qu'en général les RAM statiques n'ont pas une très grande capacité. la figure **Erreur ! Source du renvoi introuvable.** illustre le principe de fonctionnement d'une cellule SRAM

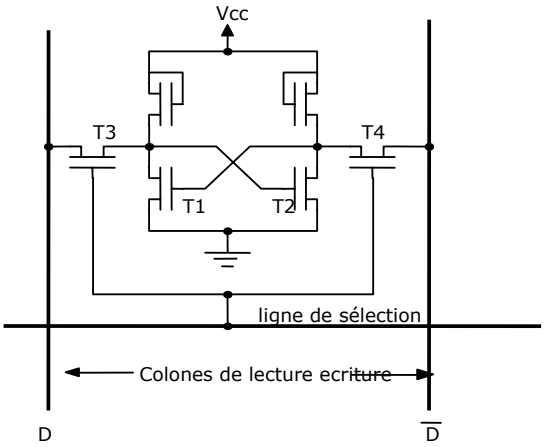


Fig. III.2 : cellule d'une RAM statique

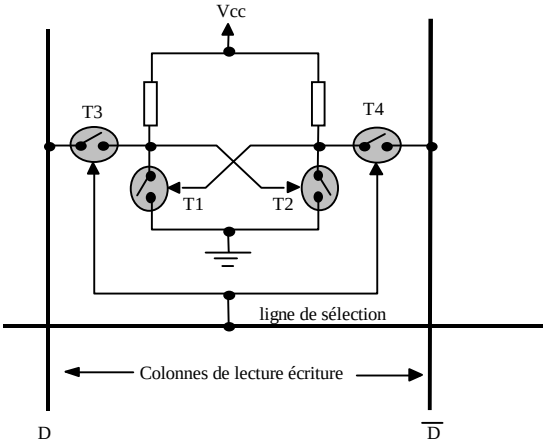


Fig. III.3 : schéma général d'une cellule SRAM

Quelque soit le type de mémoire, les cellules son organisées en matrice XY. Une cellule est repérée par son numéro de ligne et son numéro de colonne qui constituent ce qu'on appelle **l'adresse** de la cellule. L'exemple de Fig. III.4 illustre l'exemple d'une mémoire 16 bits, organisée en 4 lignes et 4 colonnes. En utilisant des décodeurs, on a besoin de deux bits d'adresse A_1A_0 Pour sélectionner une ligne, et de deux bits d'adresse A_3A_2 pour sélectionner une colonne, soit une adresse globale de 4 bits. Donc en général pour une mémoire de capacité N bits, il faut n bits d'adresses tels que $N=2^n$.

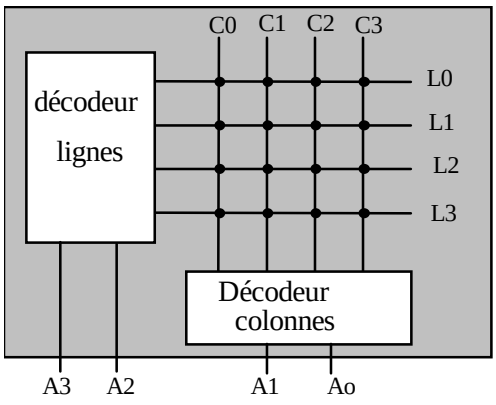


Fig. III.4 : structure matricielle

Un décodeur est un circuit numérique qui a *n* entrées d'adresse et $N = 2^n$ sorties. Les entrées d'adresse permettent de sélectionner une seule sortie. Selon la nature du décodeur utilisé, la sortie sélectionnée passe à l'état logique *"1"* ou *"0"*, toutes les autres sorties sont dans l'état logique contraire. La figure Fig. III.5 montre la convention de dessin pour faire la différence entre les deux types de décodeur.

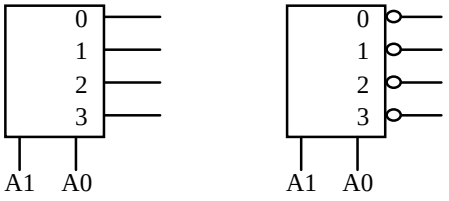


Fig. III.5 : convention de dessin pour décodeur

Le schéma de la figure Fig. III.6 illustre l'exemple d'une RAM statique 16 bits organisée en matrice 4 x 4. Si on applique une adresse $A_3A_2A_1A_0$ = 0110. A_1A_0 = 10 $\Rightarrow$ La sortie 2 décodeur colonne est mise à "1" ce qui rend T_7 et T_8 conducteurs, on a accès à toutes les cellules de la (double) colonne n° 2 (2, 6, 10 et 14). Or,. A_3A_2 = 01 $\Rightarrow$ La ligne 1 est mise à "1", seul le contenu de la cellule 6 est connectée à la double colonne n° 2 qui l'achemine vers la sortie à travers les transistors T_7 et T_8. L'écriture se fait de la même façon en utilisant les lignes $D \text{ et } \overline{D}$ comme entrées. En fait, en utilise une seule entrée de lecture/ecriture grace au circuit illustré sur Fig. III.7 qui utilise des circuits à logique 3 états pour contrôler la lecture et l'écriture. Pour écrire, on fait W=1, R=0, les buffers d'écriture sont validés, alors que l'ampli de lecture est déconnecté. Pour la lecture on fait R=1, W=0, les buffers d'écriture son déconnectés, et l'ampli de lecture est validé.

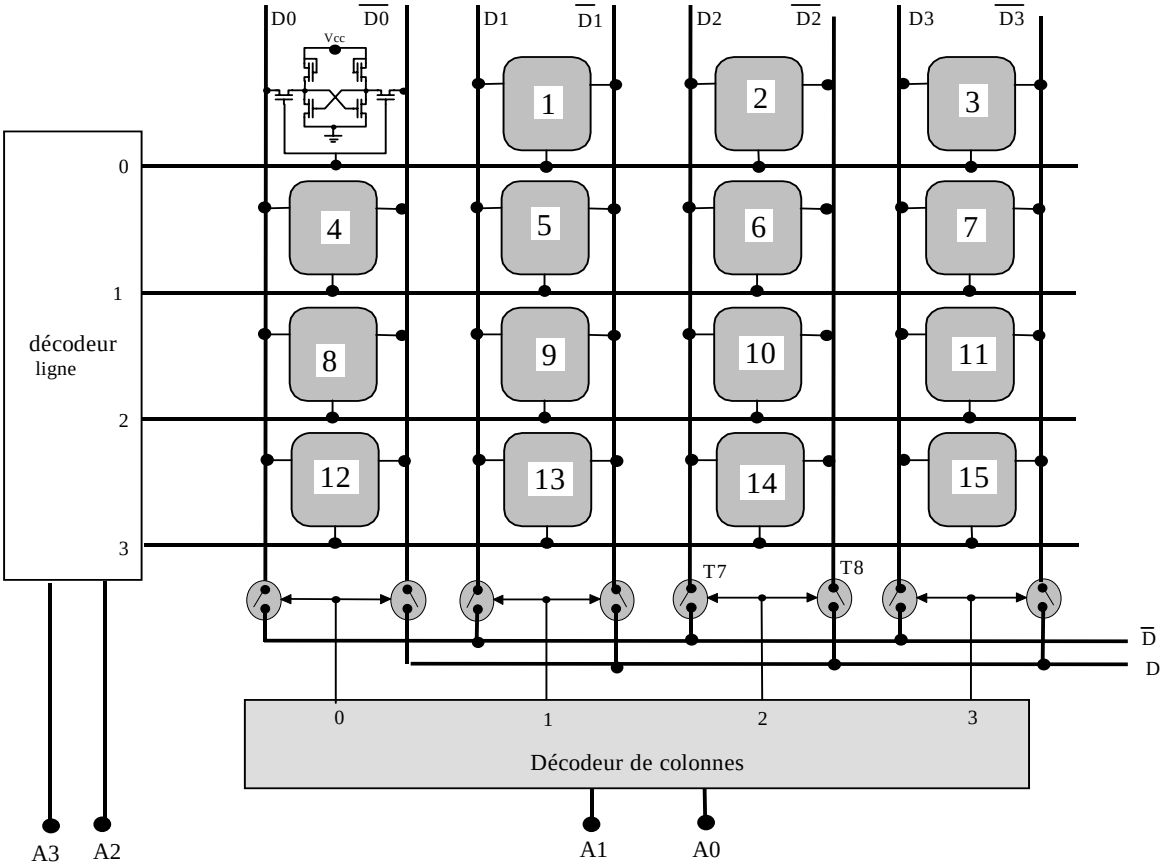


Fig. III.6 : RAM statique 16 bits organisée en matrice 4 x 4

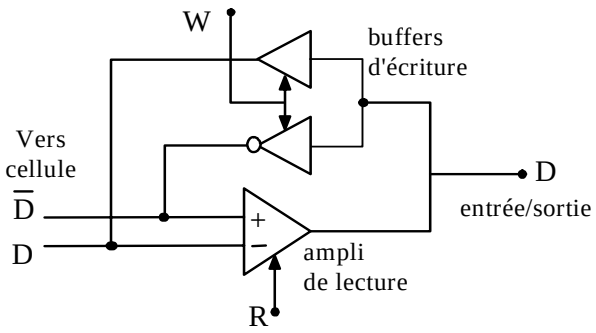


Fig. III.7 : circuit de lecture écriture d'une RAM statique

III.3.2 Cellule dynamique d'une mémoire vive

L'information est stockée dans une capacité de structure qui en fait la capacité parasite grille-substrat d'un transistor MOS. Le "1" logique correspond à la capacité chargée et le "0" logique correspond à la capacité déchargée. La cellule de mémorisation de base peut alors être réalisée de façon simplifiée par rapport à celle des mémoires statiques comme cela est illustré sure la figure Fig. III.9. Cependant, si cette structure occupe peut de place, elle n'a par contre pas d'état stable car la capacité à tendance à se décharger dans la résistance de fuite associée a la capacité. Il faut donc constamment rafraîchir la mémoire, pour cela on lit la cellule à intervalle régulier (quelques millisecondes) et on réinscrit son contenu. Pour cette raison, la mémoire est dite **dynamique**. Les mémoires dynamiques sont environ 4 fois plus denses que les mémoires statiques de même technologies mais plus délicates d'utilisation.

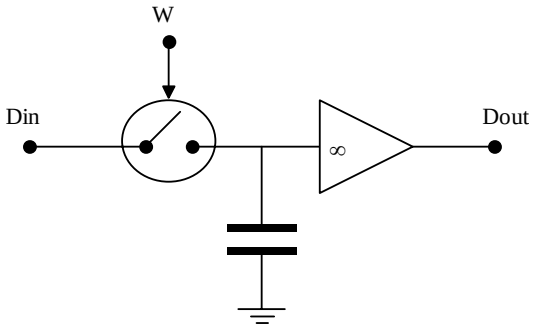


Fig. III.8 : principe d'une cellule DRAM

- Ecriture : W=1, $\Rightarrow T_1$ conduit
  Si $D_{in} = 0 \Rightarrow$ C est déchargée
  Si $D_{in} = 1 \Rightarrow$ C est chargée

- Lecture : W=0, R=1, $\Rightarrow T_3$ conduit
  - Si C chargée $\Rightarrow T_2$ conduit $\Rightarrow D_{out} = 0$
  - Si C déchargée $\Rightarrow T_2$ bloqué $\Rightarrow D_{out} = 1$

On retrouve donc en sortie l'information d'entrée complémentée.

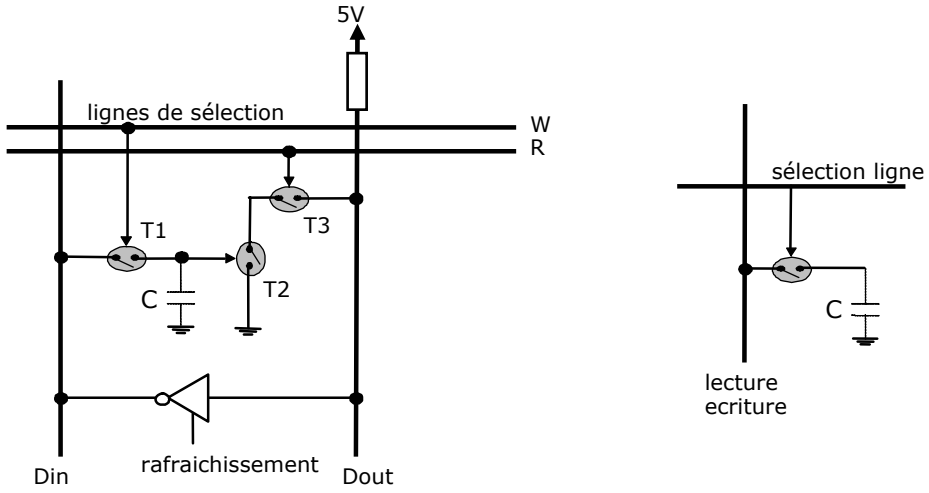


Fig. III.9 : Cellules de mémorisation d'une RAM dynamique

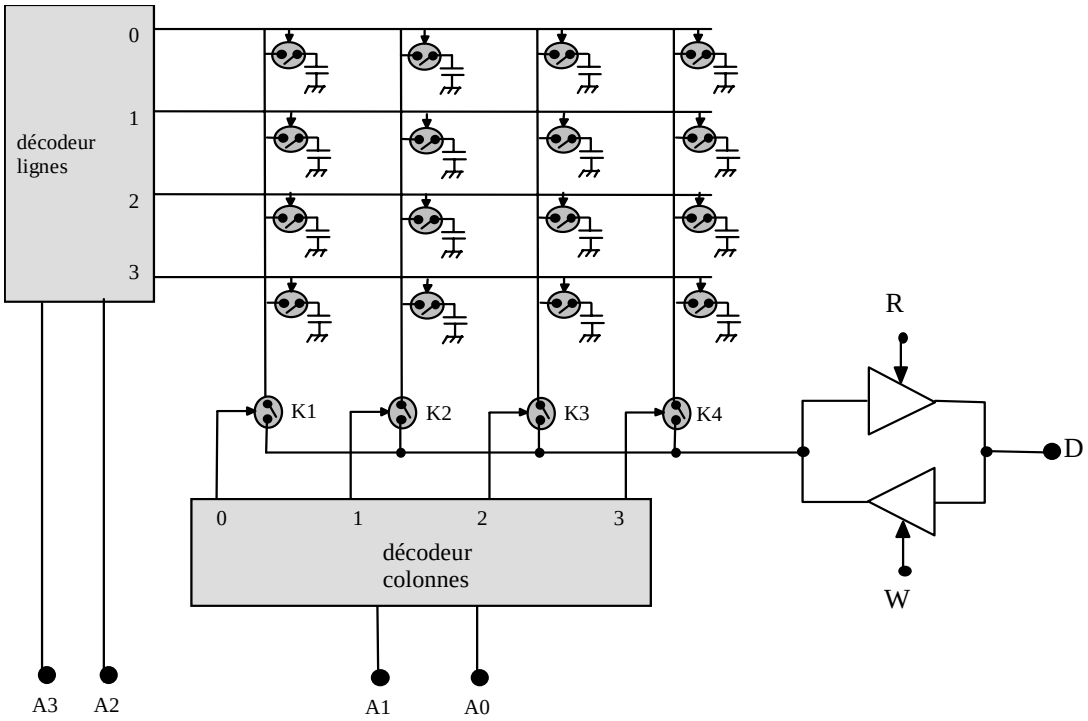


Fig. III.10 : RAM Dynamique organisée en matrice 4 x 4 avec son circuit de lecture écriture

III.3.3 Cellule d'une mémoire ROM

Il s'agit essentiellement de présence ou d'absence d'une connexion entre une ligne et une colonne. Cette connexion peut être une métallisation (court-circuit), une diode ou un transistor MOS.

Pour lire le contenu cellule (i,j), on met la colonne j à 0 et on lit la sortie D sur la ligne i.

- Si présence de connexion $\Rightarrow$ D = 0
- Si absence de connexion $\Rightarrow$ D = 1

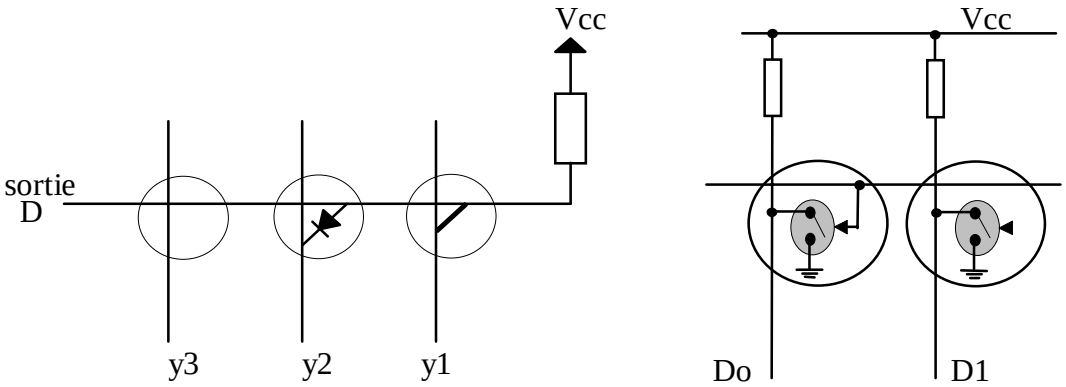


Fig. III.11 : Cellule d'une mémoire ROM

En technologie MOS, le point de connexion est un transistor MOS avec ou sans grille selon si on désire mémoriser un 0 ou un 1. Pour lire le contenu cellule (i,j), on met la ligne i à 1 et on lit la sortie D sur la colonne j.

- Si MOS avec grille, il conduit $\Rightarrow D_j = 0$
- Si MOS sans grille, il ne conduit pas $\Rightarrow$ D = 1

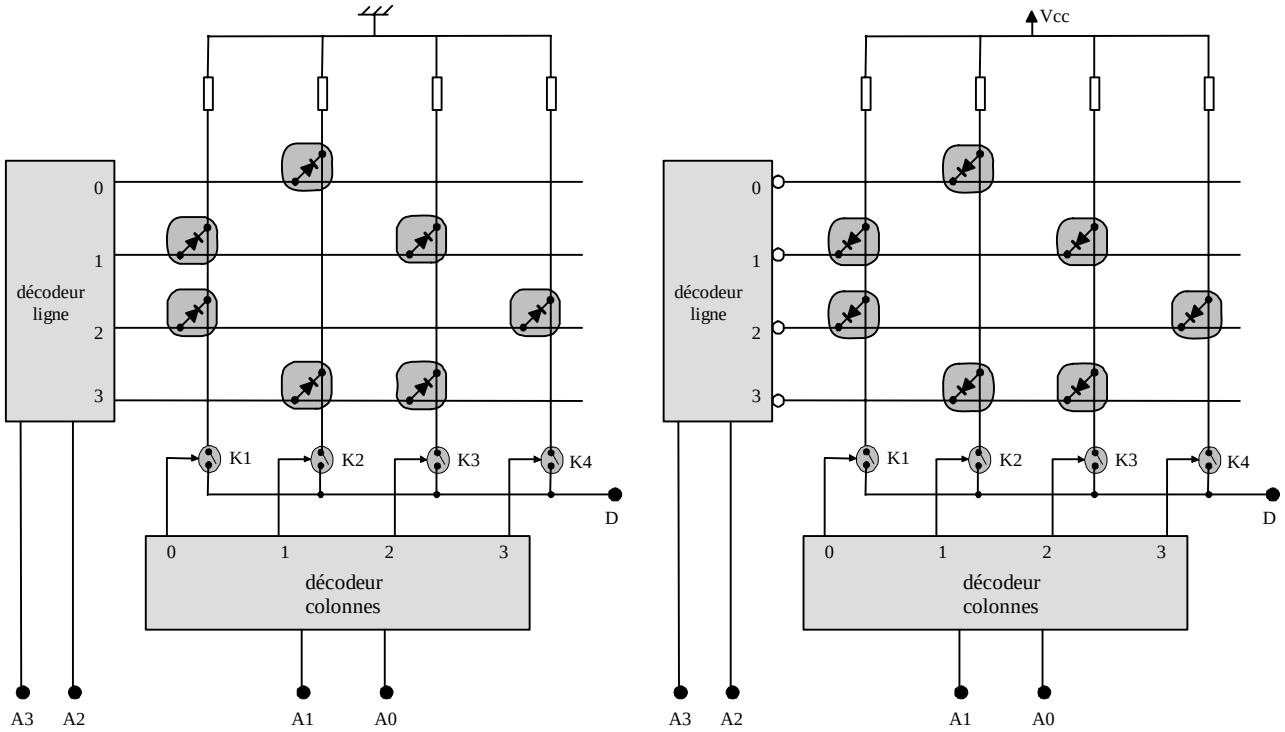


Fig. III.12 : ROM 16 bits (décodeur ligne actif : (a) niveau haut, (b) niveau bas)

III.3.4 Cellule d'une mémoire PROM

La connexion est remplacée par un micro fusible que l'utilisateur peut laisser intacte ou détruire selon s'il veut mémoriser un 0 ou un 1. Le fusible peut être détruit par le passage d'un courant très supérieur au courant normal de lecture. Dans le cas ou le fusible est constitué qu'une diode, celle ci peut être détruite par claquage en lui appliquant une tension inverse importante. On utilise aussi des transistor bipolaires dont on détruit la jonction B-E ou des MOS dont on détruit l'oxyde.

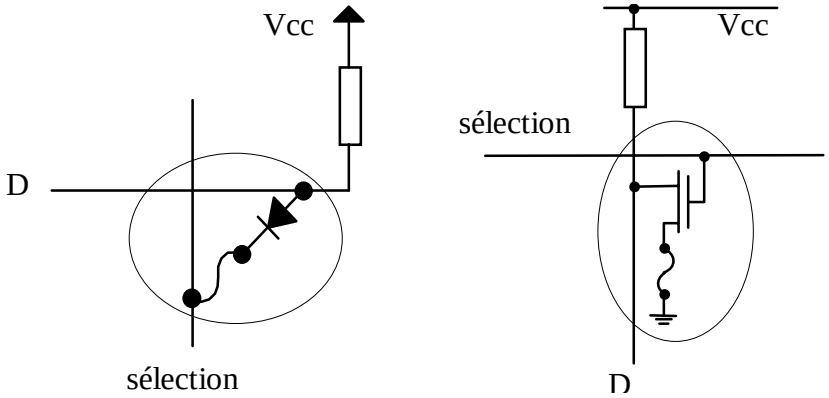


Fig. III.13 : Cellule d'une PROM

III.3.5 Cellule d'une mémoire EPROM et EEPROM

Le point de connexion est constitué d'un transistor Spécial (FAMOS : Floating avalanche injection MOS.) qui a une grille au silicium polycristallin complètement isolée.

L'oxyde est de 1000 Å environ entre le drain et la grille flottante, les électrons peuvent alors voyager entre le drain et la grille isolée à travers la couche d'oxyde sous l'effet d'un champ électrique issu d'une tension (10 à 30 V) entre le drain est la grille de contrôle. Une fois l'impulsion terminée, les électrons restent piégés grâce à l'isolement de la grille. Si la charge de la grille est supérieure à la tension de seuil, on aura rendu le MOS conducteur et mémoriser un *"0"*. L'effacement de la mémoire est obtenu par rayonnement ultra violet (2537 Å)

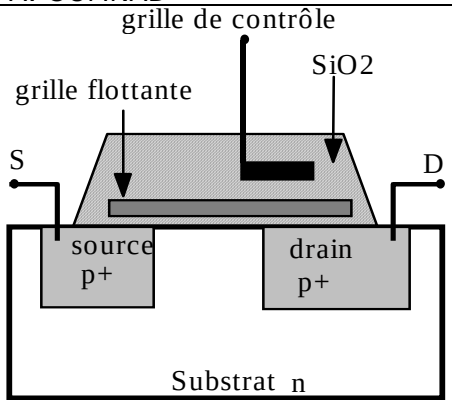


Fig. III.14 : transistor FAMOS

d'intensité importante provoquant un photo-courant entre le substrat et la grille et déchargeant celle-ci. Après effacement, tous les bits sont à "1".

Les EEPROMs utilisent une technologie semblable à l'EPROM avec la propriété d'être effaçable électriquement. En fait, on peut réécrire dans la mémoire avec une impulsion électrique sans être obligé de l'effacer. Ceci est rendu possible car la zone (tunnel) isolant la grille et le drain a une épaisseur très mince (50 à 200 Å contre 1000 pour l'EEPROM) ce qui rend possible le déplacement des électrons dans les deux sens grâce au mécanisme de Fowler-Nordheim.

Le développement des EEPROMs a ouvert un champ d'utilisation très important car on a enfin des mémoires électroniques non volatiles. Elles ne sont pas aussi rapides que les RAM, mais en tout cas, bien plus rapides et surtout moins encombrantes que les mémoires magnétiques. Les plus rapides sont appelées mémoires flash. Elles remplacent très avantageusement les disquettes et les cartes magnétiques, mais il faut attendre encore un peu pour arriver à la capacité des disques durs.

III.4 ORGANISATION PAR MOT

Dans les mémoires que nous venons de voir, on peut adresser un bit à la fois. Dans la pratique, on désire adresser des mots de plusieurs bits, comme des octets par exemple. Pour faciliter le dessin, la figure Fig. III.15 montre une mémoire de 16 mots de 4 bits chacun. Elle est obtenue par association de 4 matrices de 16 bits. Toutes les matrices reçoivent la même adresse ligne et colonne. Quand on écrit un mot, chaque bit est stocké dans une matrice. Les circuits de lecture écriture ne sont pas représentés.

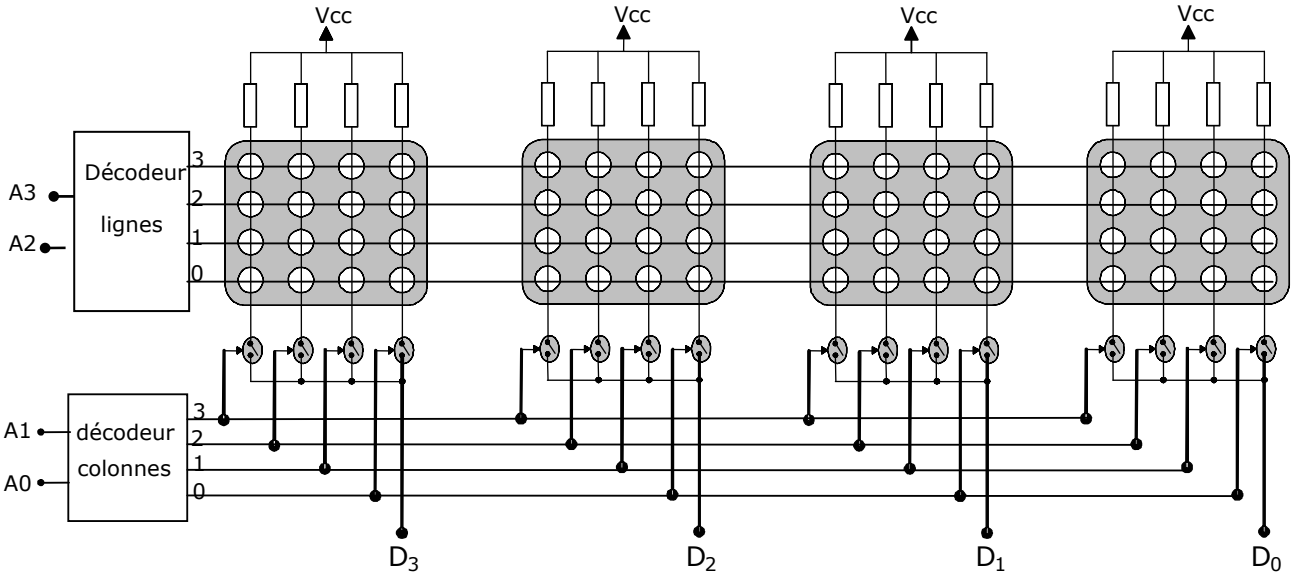


Fig. III.15 : mémoire de 16 demi-octets

Pour obtenir une mémoire organisée en octets, il suffit de prendre 8 matrices.

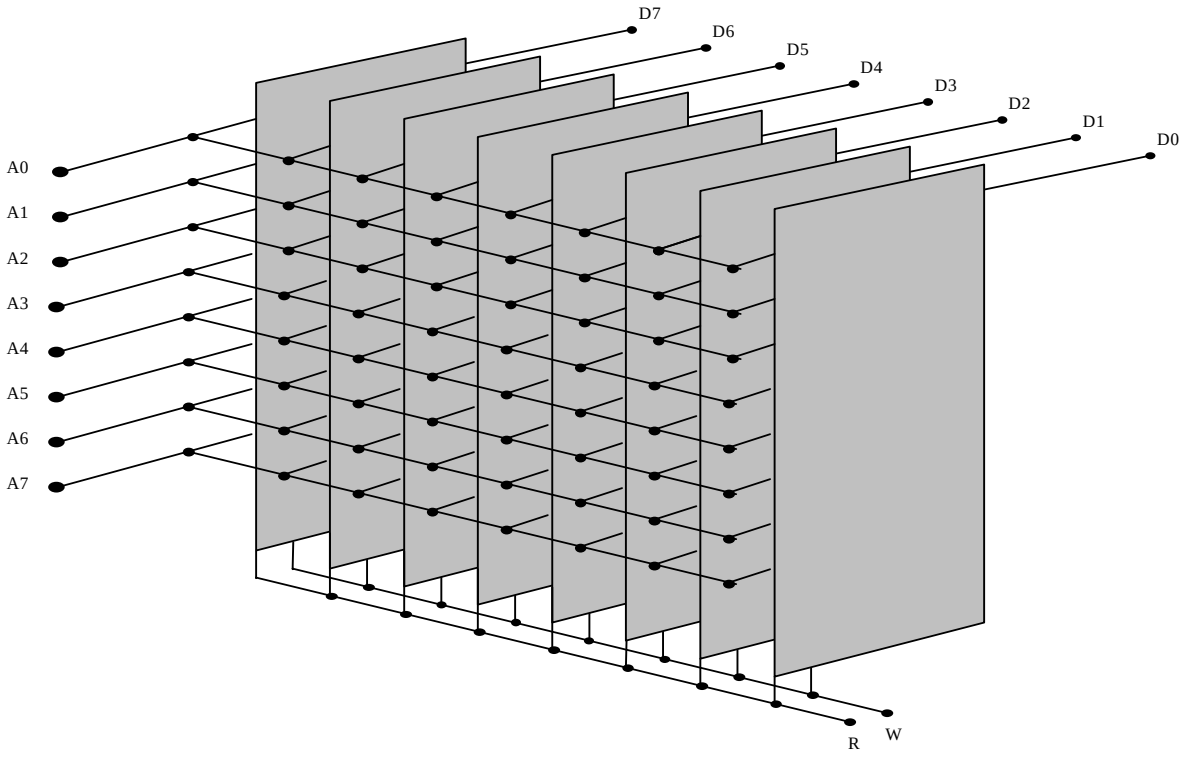


Fig. III.16 : mémoire 256 x 8

III.4.1 Capacité d'une mémoire

Pour éviter toute confusion lors de la détermination de la taille d'une mémoire, se rappeler que :

- Le nombre de bits du BUS DE DONNEES détermine la TAILLE DES MOTS que l'on peut mémoriser dans la mémoire.
- Le nombre de bits du BUS D'ADRESSE détermine CAPACITE, c'est à dire le NOMBRE DE MOTS que la mémoire peut stocker.

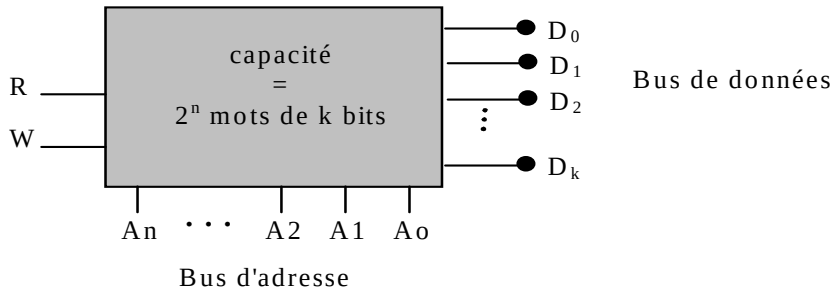


Fig. III.17 : présentation externe d'une mémoire

III.4.2 Entrée de sélection de boîtier

Beaucoup de circuits électroniques sont munis de cette entrée. Quand elle est validée, elle permet au circuit de fonctionner correctement. Si elle n'est pas validée, le circuit est complètement déconnecté. Ceci est très utile quand il s'agit de connecter plusieurs circuits en parallèle sur un même bus. L'adressage doit être fait de telle sorte qu'il n'y a jamais plus d'un circuit sélectionné.

III.4.3 Augmentation de capacité mémoire par association de plusieurs boîtiers

Réalisons une mémoire de 4 Mo à l'aide de 4 boîtiers mémoires de 1 Mo chacun. Une mémoire de 1 Mo possède 20 entrées adresse (voir tableau ci-dessous), A_0...A_{19}. Or, pour adresser 4 Mo il faut 22 entrées adresse, A_0...A_{21}. Les deux bits de plus fort poids permettent de sélectionner un boîtier, les 20 autres permettent d'adresser un octet à l'intérieur du boîtier.

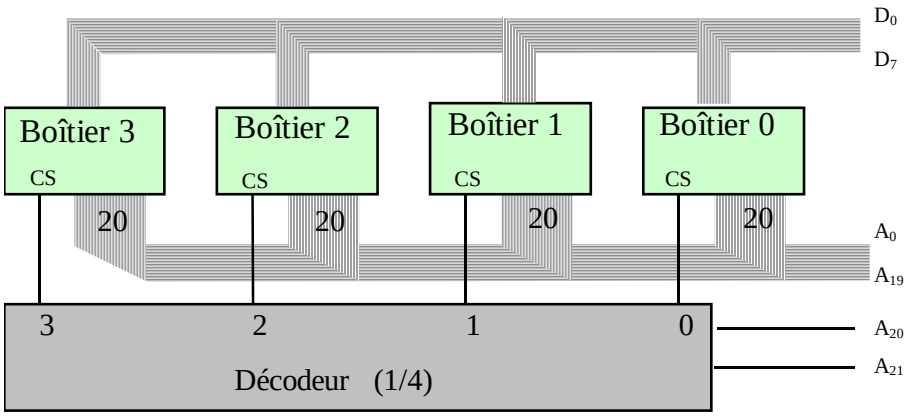


Fig. III.18 : association de boîtiers mémoire

Position	Adresse (Hexa)	Adresse (Dec)
1er boîtier	000000→0FFFFF	0→1048575
2ème boîtier	100000→1FFFFF	1048576→2097151
3ème boîtier	200000→2FFFFF	2097152→3145727
4ème boîtier	300000→3FFFFF	3145728→41194304

tab. III-2 : répartition des adresses entre boîtiers

Nb bits adresse	Capacité	
10	1024	1 ko
11	2048	2 ko
12	4096	4 ko
13	8192	8 ko
14	16384	16 ko
15	32768	32 ko
16	65536	64 ko
17	131072	128 ko
18	262144	256 ko
19	524288	512 ko
20	1048576	1 Mo
21	2097152	2 Mo
22	4194304	4 Mo
23	8388608	8 Mo
24	16777216	16 Mo
25	33554432	32 Mo
26	67108864	64 Mo

tab. III-3 : capacité en fonction du nombre de bits d'adresse

III.5 CYCLE DE LECTURE

Les cycles de lecture écriture ne sont pas les même pour toutes les mémoires. Le cycle de lecture représenté sur la figure 3.13 est un cycle général qui représente les opérations à effectuer pour réaliser une opération de lecture.

1) L'UC envoie l'adresse (de la case mémoire que l'on désire lire)
2) L'UC envoie le signal de sélection de boîtier CS.
3) L'UC envoie le signal RE *(Read Enable)* pour informer la mémoire qu'on désire réaliser une lecture.
4) Au bout d'un certain temps que l'on définit comme le temps d'accès, les données se présentent sur le bus de données qui était en mode haute impédance
5) Après lecture des données, L'UC ramène les signaux CS et RE à leur position de repos. Un court instant après, les sorties repassent en haute impédance et le bus d'adresse est libéré pour une éventuelle nouvelle utilisation.

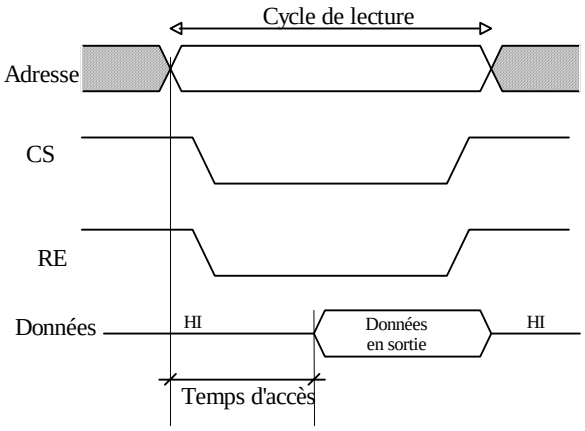


Fig. III.19 : Cycle de lecture

Remarque :

Le positionnement de l'adresse revient à positionner plusieurs bits d'adresse. Pour ne pas alourdir le dessin, on a coutume de représenter deux signaux complémentaires avec un point d'intersection qui matérialise l'instant de changement des signaux. La zone hachurée précise que la valeur de l'adresse n'a aucune importance.

III.5.1 Cycle d'écriture

Comme pour le cycle de lecture, l'UC :

1) Envoie l'adresse
2) Envoie CS
3) Place la donnée sur le bus de données
4) Envoie WE
5) Ramène WE à sa position de repos après une temporisation qui dépend du type de mémoire.
6) Désélectionne le boîtier en ramenant CS à sa position de repos.

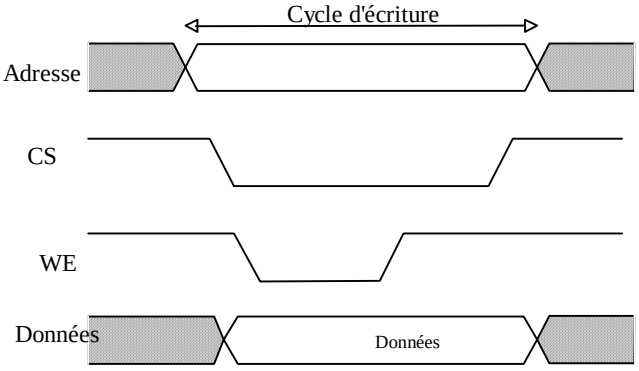


Fig. III.20 : Cycle d'écriture

III.5.2 Les barrettes SIM et DIM

Les barrettes SIM et DIM sont des petites barrettes enfichables portant des RAMs dynamiques qu'on utilise au niveau de la mémoire centrale. Ces barrettes on eu beaucoup de succès car elles prennent très peu de place sur la carte mère et sont très facile à placer. Le tableau ci-dessous montre quelques valeurs à titre indicatif (jusqu'à 1997):

Barrette	capacité par barrette	temps de cycle
SIM à 30 pins	256k, 1Mo , 4 Mo	60ns - 70 ns
SIM à 72 pins	4Mo, 16 Mo	60 ns - 70 ns
DIM 168 pins	32 Mo, 64 Mo, 128 Mo	10 ns - 70 ns

III.6 MEMOIRES MAGNETIQUES

Le principe des mémoires magnétiques est analogue à celui utilisé sur les bandes des magnétophones. Il exploite l'aimantation rémanente créée sur une couche mince de matériaux ferromagnétique. Le plus souvent, c'est de l'oxyde de fer déposé sur un support souple (disquette) ou sur un support rigide (disque dur). La couche magnétique est constituée de micro domaines magnétiques qu'on appelle cellules, chacune peut être magnétisée dans un sens ou dans le sens opposé, ce qui correspond à la valeur 0 ou 1.

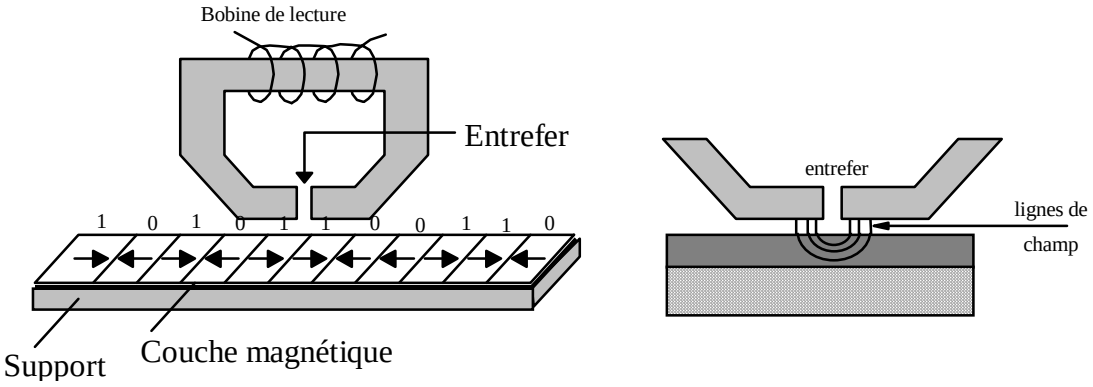


Fig. III.21 : tête de lecture/écriture magnétique

Pour magnétiser les cellules, on utilise une tête de lecture écriture constituée par l'entrefer d'un aimant sur lequel est enroulée une bobine électrique.

Ecriture : On fait passer un courant dans la bobine ce qui a pour effet de créer un champ magnétique au voisinage de l'entrefer, les lignes de champ traversent la couche magnétique transformant la cellule en dessous de l'entrefer en petit aimant qui subsistera même après suppression du courant de la bobine et ceci grâce à la rémanence de l'oxyde de fer. Le sens du courant dans la bobine définit le sens d'orientation du champ dans l'entrefer et donc l'orientation de l'aimantation de la cellule et donc la valeur 0 ou 1 du bit enregistré.

Lecture : Chaque cellule aimantée dans un sens ou dans l'autre est un petit aimant. Quand elle défile sous la tête de lecture, elle induit un courant électrique dans la bobine. Suivant le sens du courant induit, on détermine la valeur 0 ou 1 du bit lu.

III.6.1 Les disquettes

Pour les disquettes, on utilise un disque souple comme support pour la couche magnétique. Lors du formatage, la surface du disque est partagée en plusieurs pistes concentriques. Chaque piste est subdivisée en plusieurs secteurs. Les format de disquettes utilisés de nos jours (1997) sont les 5"¼ et 3"½ et c'est le 3"½ qui s'impose largement avec un capacité après formatage de 1.44 Mo.

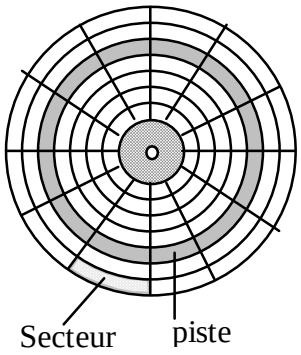


Fig. III.22 : pistes et secteurs d'une disquette

Le lecteur de disquettes de se compose de 4 éléments :
- Le moteur de rotation du disque
- Deux tête de lecture écriture combinées, une pour chaque face.
- Un moteur de translation des têtes.
- L'électronique de contrôle.

Le moteur de rotation ne fonctionne qu'au moment de l'accès à la disquette. La vitesse de rotation est de 300 tr/mn pour les disquettes 3" ½ .

Les têtes de lectures sont posées sur la surface de la disquette. Elle peuvent effectuer un déplacement radial et atteindre ainsi toutes les pistes de la disquettes.

Dimension (pouces)	5 ¼	5 ¼	3 ½	3 ½
Capacité (octets)	360 k	1.2 M	720 k	1.44 M
Nombre de pistes	40	80	80	80
Secteurs/piste	9	15	9	18
Nombre de têtes	2	2	2	2
Vitesse de rotation	300	360	300	300
Débit (kbits/s)	250	500	250	500

En général on enregistre 512 Octets par secteur, la capacité d'une disquette est donc :

Capacité = NP x NS x CS x NT

NP : Nombre de piste
NS : Nombre de secteur par piste
CS : Capacité d'un secteur
NT : Nombre de tête

III.6.2 Les disques durs

Un disque dur est constitué d'un ensemble de plateaux en aluminium recouverts d'une fine couche magnétique. Comme pour les disquettes, la surface de chaque plateau est subdivisée en plusieurs pistes concentriques. Chaque piste est subdivisée en plusieurs secteurs d'une capacité de 512 octets en général. Les têtes de lecture écriture (une par face) sont placées sur un bras mobile leur permettant un déplacement radial pour atteindre toutes les pistes. Toutes les têtes se déplacent d'une façon solidaire, c'est pour ça qu'on parle de cylindre à la place de pistes quand il s'agit de disque dur. En effet quand une tête est placée sur une piste, les autres têtes sont placées sur des pistes de même rang et on peut accéder à toutes les données enregistrées sur un cylindre constitué de toutes les pistes superposées.

Le disque (les plateaux) tourne à une vitesse supérieure à celle des disquettes (actuellement 7000 tours/min au lieu de 360 pour une disquette), il est maintenu à cette vitesse tant que l'ordinateur est alimenté. A la différence des disquettes, les têtes de lecture écriture ne posent pas sur le support mais planent sur un coussin d'air à une distance infime du plateau, dite **hauteur de vol**. Celle-ci est de l'ordre de 0,2 à 1 µ. (un centième d'un cheveu humain). En effet, vu la vitesse de rotation élevée, si les têtes touchent les plateaux, cela risque d'abîmer la couche magnétique. Le " flottage " de la tête est assuré par un coussin d'air créé par la rotation du disque. Il faut donc s'assurer que lors de l'arrêt de celui-ci, la tête ne se trouve pas au-dessus d'une zone de données, mais au-dessus d'une zone spéciale, dite **zone d'atterrissage** *(landing zone)*. La surface du disque doit être absolument propre sous peine de provoquer un atterrissage de la tête. C'est pour cela que les disques sont montés en salle blanche dans un boîtier absolument étanche, l'infiltration de la moindre poussière pourrait provoquer un "headcrash" à savoir un écrasement d'une tête sur un plateau (une poussière = 20 µ, un cheveu = 70µ).

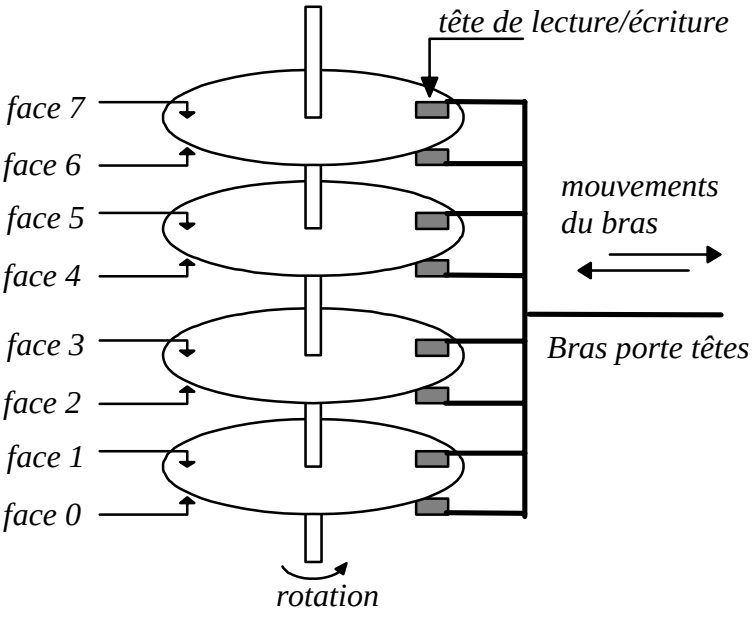


Fig. III.23 : Disque dur à 4 plateaux

Les disques durs sont caractérisés par leur capacité et par leur temps d'accès. De nos jours, le temps d'accès varie entre 8 ms et 13 ms. La capacité des disques couramment livrés sur les machines grand public comme les PCs varie entre 1 Go et 20 Go. La capacité d'un disque dur

se calcule de la même façon qu'une disquette sauf qu'ici on parle de cylindre au lieu de piste, (NC = nombre de cylindres)

Capacité = NC x NS x CS x NT

Un disque qui a 10 plateaux (20 surfaces = 20 têtes) subdivisés en 2000 cylindres de 100 secteurs (de 512 octets) chacun a une capacité de 2000 x 100 x 0.5k x 20 = 2 Go.

III.7 LES INTERFACES DE GESTION DE DISQUES DURS

L'interface de gestion du disque dur permet au processeur d'échanger des données avec le disque sans se préoccuper de la façon don les données sont enregistrées sur celui-ci.

III.7.1 Interface IDE (et ses variantes)

L'interface IDE (Integrated Device Equipement) est devenue le standard en vigueur (1997) dans la gestion des disques durs sur les PCs. Cette interface utilise un câble de 40 fils pour gérer 2 disques, il faut toutefois veiller à configurer un disque en maître et le deuxième en esclave pour ne pas avoir de conflit d'adresse. Les disques IDE sont munis de jumpers prévus à cet effet.

Le contrôleur IDE utilise une technique astucieuse pour stocker plus d'information sur un disque. En se basant sur la constatation que les secteurs des pistes situées sur le bord des plateaux on une surface plus grande que les secteurs des pistes intérieures, alors qu'ils servent à stocker la même quantité d'information (512 octets), l'interface IDE découpent les pistes extérieures en un nombre plus grand de secteurs afin d'avoir une surface de secteur homogène sur tout le disque. Les paramètres (cylindre/secteur/têtes) déclaré lors de la configuration du PC ne correspondent pas à la réalité au niveau du disque. C'est le contrôleur IDE qui fait la conversion entre les deux formats.

Les cartes mère PC commercialisées de nos jours ont une interface IDE intégrée et offrent 4 connecteur pouvant piloter deux disques chacun. On peut donc brancher jusqu'à 8 disques durs.

III.7.2 Interface SCSI

Le standard SCSI (Small Computer System Interface) est une interface intelligente qui n'est pas destinée seulement à la gestion des disques. On peut y connecter des périphériques SCSI de tout type comme des streamer, des scanners, des imprimantes...

Un contrôleur SCSI peut gérer 8 périphériques (lui-même inclus), ce qui fait qu'on peut brancher 7 périphériques par contrôleur SCSI. Tous les périphériques sont branchés sur le même câble de 50 fils.

Des variantes de l'interface SCSI on fait leur apparition sur le marché comme le fast SCSI, le wide SCSI et l'ultra SCSI et des versions combinées comme le FW et UW capables de gérer (15+1) périphériques par contrôleur. Ceci sachant qu'on peut installer autant de contrôleurs SCSI qu'on a de slots disponibles. Le bus SCSI peut fonctionner en synchrone et en asynchrone.

Avec les disques SCSI, on n'est pas obligé de déclarer les paramètres (cylindre/secteur/tête) lors de la configuration de l'ordinateur, le contrôleur lui-même détecte et reconnaît le disque dur.

Un aspect intéressant est de pouvoir pour des raisons de sécurité utiliser deux disques pour stocker la même information et pratiquer ce qu'on appelle du miroring entre le deux. Toute écriture se fait simultanément sur les deux disques. Lors de la lecture, le contrôleur envoie une requête, et lit sur le disque qui répond le premier. En cas de panne d'un disque, le deuxième continue à fonctionner normalement.

Avec les nouvelle variantes du standard SCSI, on peut atteindre des taux de transfert de 80 Mb/s, à condition toutefois d'utiliser des disques capables de pratiquer ces débits (problème mécanique de balayage des pistes). Il est difficile de mesurer les performances de ce standard, on peut toutefois constater que sur les machines PC, on réalise un gain de performance de l'ordre de 2.5 quand on passe du Standard IDE au standard SCSI.

III.8 LES MEMOIRE OPTIQUES

Il s'agit de disques qu'on peut lire et écrire à l'aide d'un rayon laser.

III.8.1 Nomenclature

Le CD-ROM

Le CD-ROM (Compact Disk Read Only Memory) est un disque optique à lecture seule. Les données y sont inscrites par moulage lors de sa fabrication.

Le CD-R

Le CD-R *(Compact Disk Recordable)* est un disque de type WORM *(Write Once Read many).* Le CD vierge est enregistré (une fois) par l'utilisateur à l'aide d'un graveur et peut ensuite être lu sur un lecteur de CD-ROM classique.

Le CD-RW

Le CD-RW *(Compact Disk ReWritable)* est un CD réinscriptible que l'on peut utiliser un peu comme une disquette.

III.8.2 Le CD-ROM

Les CD se présentent sous la forme d'un disque de 12 cm et de 1,2 mm d'épaisseur percé d'un trou de 15 mm de diamètre. Le disque est réalisé dans une matière plastique transparente (polycarbonate) sur laquelle les données sont inscrites par moulage lors de la fabrication.

L'élaboration d'un CD-ROM commence par l'élaboration d'un disque "mère" qui est percé de minuscules trous de l'ordre du micron provoqué par échauffement local à l'aide d'un laser à haute énergie. A partir de ce disque mère, on réalise une matrice père (père *stamper* = moule) qui sert à la fabrication par pressage de très nombreux disques.

Après le moulage, La face gravée et recouverte d'une très fine couche réfléchissante en aluminium puis d'une couche un peu plus épaisse d'un vernis protecteur sur lequel on imprime l'étiquette du CD.

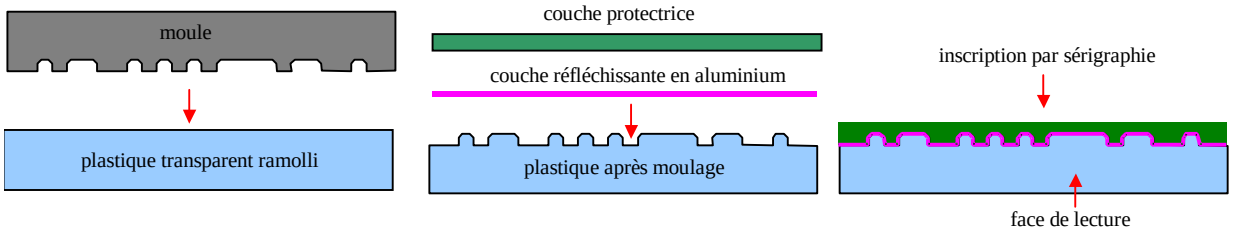


Fig. III.24 : étapes de fabrication d'un CD-ROM

Fig. III.25 : coupe et vue de face d'un CD-ROM

Les informations sont stockées sous la forme de pits (cuvettes ou creux) et de lands (plats) à la surface du disque. La profondeur des pits est de l'ordre de 0.12 µm. Les pits et les lands sont ordonnés le long de la seule et unique spirale constituant la piste du CD. Cette spirale qui commence au bord intérieur du CD et finit au bord extérieur a plus de 6 km de longueur.

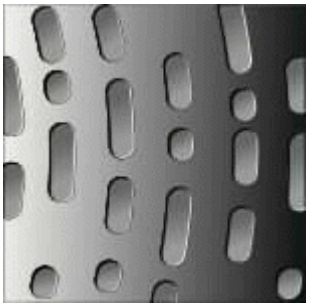


Fig. III.26 : vue de face

III.8.3 Principe de lecture

Pendant la lecture d'un CD-ROM, un détecteur reçoit et mesure l'énergie d'un rayon laser de faible puissance réfléchi sur la couche d'aluminium. Les creux et les plats entraînent une différence de réflexivité qui sont mises en valeur par le détecteur pour la reconnaissance des 1 et des 0.

La distance focale du rayon laser de lecture est ajustée pour que le rayon focalise exactement sur les plats.

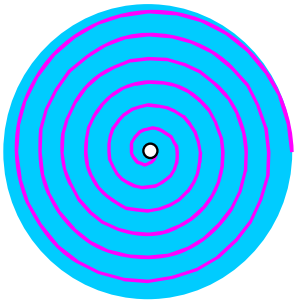


Fig. III.27 : piste en spirale d'un CD

- Lors de la lecture d'un land, la lumière émise par le laser est réfléchie en totalité, elle est captée par un photo-détecteur qui délivre un signal électrique important.

- Lors de la lecture d'un pit, on peut noter sur Fig. III.29 que le rayon laser est réfléchi en partie par la surface du disque et en partie par le fond du pit. En fait, la moitié de l'énergie lumineuse est réfléchie par la surface du disque. La lumière réfléchie par le fond du pit parcourt une distance supérieure à la distance parcourue par la lumière réfléchie par la surface. L'écart entre les deux trajets correspond exactement à la moitié de la longueur d'onde du rayon Laser utilisé. Dans ces conditions, un phénomène physique appelé **interférence destructive** se produit. Les deux rayons lumineux s'annulent car ils sont en opposition de phase. Aucune lumière n'est réfléchie, la photodiode ne capte pas d'information lumineuse, et délivre un signal nul ou quasiment nul.
  On notera aussi qu'au niveau du pit, le rayon est défocalisé et une partie de la lumière est dispersée et ne sera de toute façon pas arrivée sur le photodétecteur.

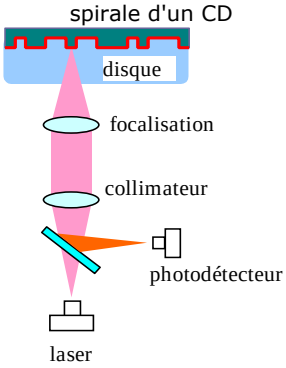


Fig. III.28 : otique

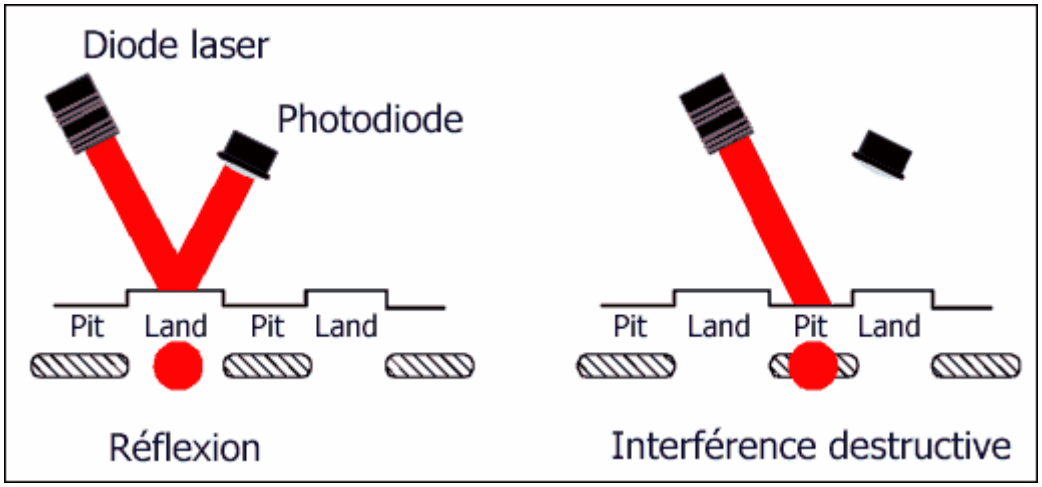


Fig. III.29 : lecture d'un CD-ROM

III.8.4 Codage de l'information

Au lieux d'utiliser les creux et les plats pour coder les 1 et les 0, on préfère utiliser les transitions creux-plat pour coder le 1. On peut vérifier sur la figure ci-dessous que cette technique est de loin meilleure à cause des différences de réflexion qui peuvent exister entre un disque est un autre ou une zone propre et une zone sale d'un même disque.

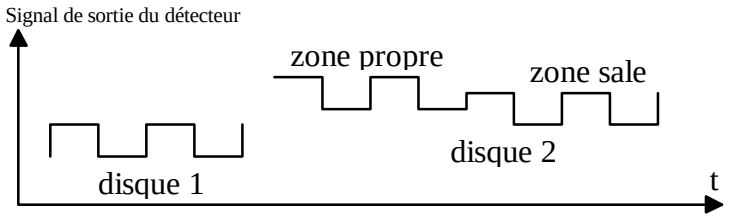


Fig. III.30 : exemple de signaux lus sur un CD

- Les "1" logiques sont codés par une transition
- Les "0" logiques sont codés par un manque de transition

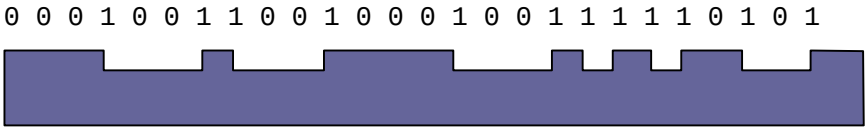


Fig. III.31 : codage de l'information sur un CD

III.8.5 Vitesse de rotation

La vitesse de rotation du disque est telle que la vitesse de déplacement de la tête de lecture sur la piste soit constante (CLV) ce qui donne une vitesse de rotation variable qui diminue lorsque la tête se dirige vers l'extérieur du disque (210 à 539 tr/mn pour les 1er lecteurs).

Pour augmenter le débit de transfert (150 ko/s à l'origine), on augmente la vitesse de rotation, on a ainsi fait les lecteurs 4X, 8X, 12X, 24X, 32X, 48X, 52X. Mais pour des vitesses angulaires variant de 5000 à 12000 tr/mn, des problèmes critiques apparaissent et certains lecteurs récents sont revenus à une vitesse angulaire fixe (CAV) ou à un système mixte

III.8.6 Le standard

Le standard de CD-ROM le plus répandu est celui du livre jaune apparu en 1985 qui permet de stocker 650 Mo et qui a donné naissance à deux modes :

- Le mode 1 constitue le standard ISO 9660 proposé en 1985 sous la nomination High Sierra. Il décrit description logique du disque avec un système de gestion de fichier arborescent similaire à celui de MS-DOS ou UNIX. Le système a été amélioré en 1988 afin d'être indépendant de la plateforme sur laquelle le CD-ROM doit être lu.

- Le mode 2 dit CD-ROM XA *(eXtended Architecture)* proposé en 1988 ajoute la possibilité de faire des CD multisessions et d'entrelacer sur la même piste des segments de données, de son et d'images.

III.8.7 Le CD-R

Un CD-R est constitué de 3 couches :

- une couche de plastique transparent constituant la face avant du disque
- une couche constituée d'un colorant organique sensible à la lumière (cyanine ou phtalocyanine)
- Une fine couche métallisée très réfléchissante en or ou en argent
- une couche protectrice constituant la face arrière du disque

Pendant la phase de gravage, le faisceau laser est focalisé sur la couche organique, sa puissance est ajustée à une valeur importante, la température de l'ordre de 250 °C provoque des réactions chimiques dans le colorant, qui devient opaque. les zones ainsi *"brulées"* vont jouer le rôle des pits d'un CD-ROM et les zones non "*brulées"* sont les équivalents des *lands.*

Fig. III.32 : coupe d'un CD-R

A la lecture, la puissance du laser est ajustée à une valeur plus faible :

- Les zones opaques empêchent la lumière d'arriver sur la couche réfléchissante, aucune lumière n'est réfléchit. Le photodétecteur ne délivre aucun signal électrique

- Les zones transparentes, laisse passer la lumière qui se réfléchit sur la couche métallique et revient vers le photo-détecteur qui délivre un signal électrique important.

Le CD-R est compatible avec le CD-ROM, il peut être lu sur un lecteur de CD-ROM normal

III.8.8 Le CD-RW

La technologie de CD-RW est quasiment la même que celle du CD-R. Le colorant organique est remplacé par un matériau qui est constitué d'un mélange de métaux (argent indium, tellure, antimoine).

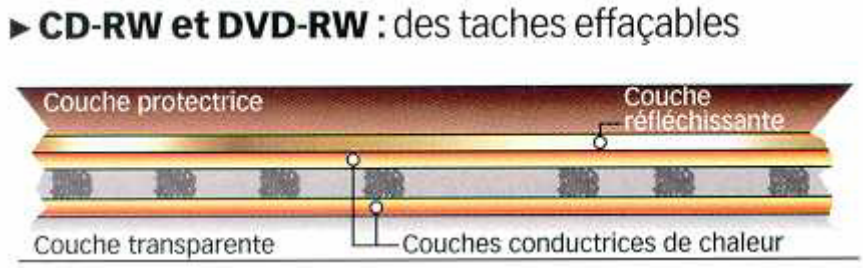


Fig. III.33 : coupe d'un CD-RW

Lors de l'enregistrement, le laser chauffe l'alliage au-delà de la température de fusion (Pwrite), soit plus de 600°C. Puis, pendant un temps très court, la puissance du laser est baissée de façon à atteindre une température inférieure à 200°C (Pbias). Ainsi traitée, la zone devient amorphe ou non cristalline qui a la caractéristique d'être opaque et se comportera comme un pit. Les zones restées cristallines correspondent aux lands.

Lors de la phase de réécriture ou d'effacement, la puissance du laser est ajustée à une valeur plus faible, l'alliage est chauffé un peu au delà de 200°C, La matière subit un " revenu

", qui homogénéise la disposition des cristaux et les oriente uniformément et les ramène à une structure cristalline transparente.

III.8.9 Le DVD

Lorsque le CD-ROM est apparu, sa capacité de stockage paraissait très largement suffisante pour les besoins de l'époque. Or le développement du multimédia a conduit les fabricants à développer un support plus performant. Ainsi, en septembre 1995, plusieurs compagnies se sont regroupées pour proposer le standard DVD *(Digital Video Disc ou Digital Versatile Disc)* .

Le DVD utilise la même technologie que le CD-ROM. Les améliorations hardware et software permettent de stocker plus de données, de les lire plus rapidement tout en étant compatible avec le standard CD-ROM : un lecteur DVD peut lire tous les formats de CD-ROMs.

Quatre variantes son disponible :

- Simple face simple couche
- simple face double couches
- double face simple couche
- double faces double couches

	DVD-ROM	CD-ROM
Diamètre	12cm (et 8 cm)	12 cm
Epaisseur	2 x 0.6 mm	1.2 mm
Capacité	4.7 à 17 Go	650 Mo
Espace interpiste	0.74 µ	1.6 µ
longueur cellule	0.4µ	0.83 µ
vitesse linéaire	4 m/s	1.2 m/s
longueur d'onde	650 et 635 nm	780 nm
Modulation	EFM plus 8 à 16	EFM 8 à 14
compression vidéo	MPEG-2	MPEG-1
compression audio	Dolby AC-3 5.1 canaux	MPEG-1 à 2 canaux

tab. III-4 : comparaison CD-ROM et DVD

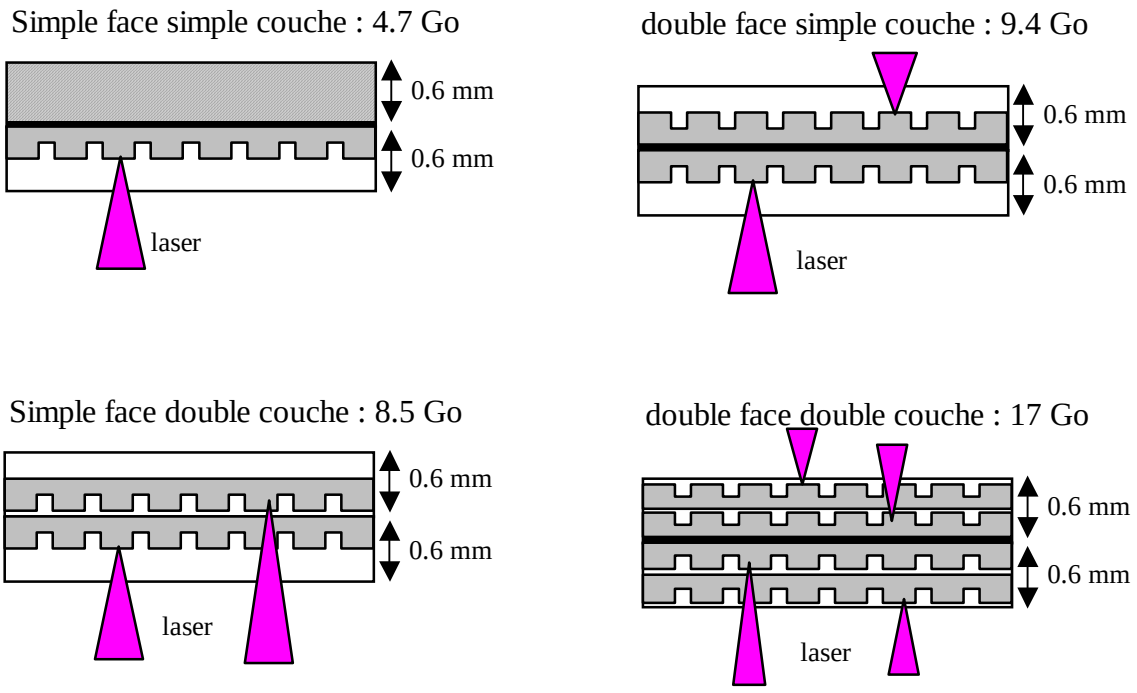


Fig. III.34 : les 4 variantes de DVD

Sur les versions double couches, la première couche est semi-transparente et le rayon laser focalise soit sur la première soit sur la deuxième.

La compression MPEG-2 permet de stocker plus de 2h de vidéo (133 mn) sur un DVD simple face simple couche ce qui est suffisant pour la plupart des films du marché. Plusieurs pistes audio sont disponibles ce qui permet par exemple de proposer des films avec plusieurs langues.

IV LA MEMOIRE CACHE

L'utilisation de *mémoire cache* s'appuie sur le *principe de localité*. Celui-ci est basé sur le fait que les instructions d'un programme au cours de son exécution doivent être proches les unes des autres. On a donc tout à gagner à amener des blocs de mots consécutifs dans une mémoire à accès rapide de telle sorte que les accès aux instructions se fassent avec un temps de réponse minimal. Ces blocs sont désignés par le terme de *lignes*.

adresse	donné
45890	XX
●●●	

Lorsque le processeur tente d'accéder à une information (instruction ou donnée) :

- En *lecture* : si l'information se trouve dans le cache (*hit*), le processeur y accède sans état d'attente, sinon (*miss*) le cache est chargé avec un *bloc* d'informations en mémoire.
- En *écriture* : si l'information référencée est dans le cache, l'écriture est effectuée dans le cache *et* dans la mémoire. Dans le cas contraire l'écriture n'est effectuée que dans la mémoire. (il existe en fait plusieurs autres *politiques* de mise à jour du cache)

La réalisation des caches utilise le principe de mémoire associative, ou mémoire adressable par son contenu *(Content Addressable Memory)*. La mémoire cache est partagée en deux zones

- La "mémoire adresse" contient l'adresse de l'information,
- La "mémoire de contenu" contient l'information elle-même.

En effet, supposons que nous disposons d'un cache de 1 ko. Une donnée XX se trouvant à la position 45890 de la RAM a été transporté dans le cache à la position 0003. Comment ensuite le processeur peut il accéder à cette donnée en ne disposant que de son adresse dans la RAM (45980). L'idée consiste à écrire dans le cache la donnée plus son adresse dans la RAM afin de pouvoir la localiser ultérieurement.

La mémoire cache est gérée par contrôleur spécifique pour alléger le processeur de cette tache. Pour définir si une information se trouve dans le cache, le contrôleur effectue une comparaison SIMULTANEE de l'adresse émise par l'unité d'adresses avec TOUTES les adresses des informations rangées dans le cache à l'aide d'un comparateur parallèle. Au cas où l'adresse s'y trouve (*hit*), la donnée est lue dans le cache à la position correspondante. Dans le cas contraire (*miss*), un nouveau bloc d'information, appelé *ligne*, est chargé avec partir de la mémoire centrale vers le cache, mais ou peut-on le placer ?

IV.1.1 Gestion des remplacements

Le choix de la ligne à remplacer dans un cache suit la règle dite de *la moins récemment utilisée (LRU : Least Recent Used)*. Cette politique de remplacement nécessite un mécanisme matériel qui s'apparente à celui mis en oeuvre dans une pile FIFO. Chaque bloc (ligne) est référencé par un numéro stocké dans une liste de classement. Lors d'un *hit* le numéro de bloc est amené en tête de liste. Lors d'un *miss* le bloc chargé à partir de la RAM remplacera le bloc dont le numéro se trouve en fin de liste, son numéro est ensuite placé en début de liste.

V LES BUS

Le bus est une structure d'interconnexion qui permet de raccorder le processeur à la mémoire et autres périphériques d'entrée sortie. Le bus est constitué de 3 ensemble de lignes :

- Les lignes d'adresse → Bus d'adresse.
- Les lignes de données → Bus de données.
- Les ligne de commande ou de contrôle → Bus de contrôle.

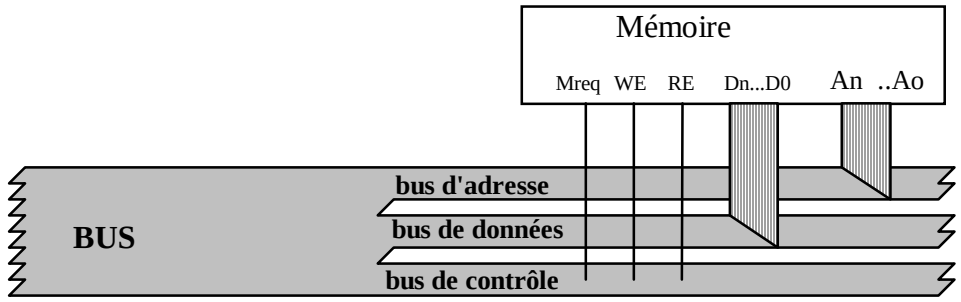


Fig. V.1 : Connexion d'une mémoire à un bus

Pour l'exemple de la figure Fig. V.1, quand le processeur veut écrire un octet X à la position P de la mémoire, il place l'adresse P sur le bus d'adresse, la donnée X sur le bus de donnée puis active la ligne de contrôle WE.
Les bus sont divisés en deux classes distinctes selon la technique adoptée pour le cadencement des échanges de données à travers le bus. On distingue les bus synchrones et les bus asynchrones.

V.1 BUS SYNCHRONE

Les bus synchrones disposent d'une ligne horloge spécifique dont la fréquence n'est pas forcément égale à l'horloge du processeur. Toute opération d'échange à travers le bus synchrone se fait en un nombre entier de périodes d'horloge. La période de cette horloge est appelée cycle du bus. Prenons l'exemple de la figure xx qui illustre une opération de lecture en mémoire. Une horloge de 4 Mhz définit un cycle du bus de 250 ns. L'opération de lecture nécessite 3 cycles de bus, soit 750 ns.

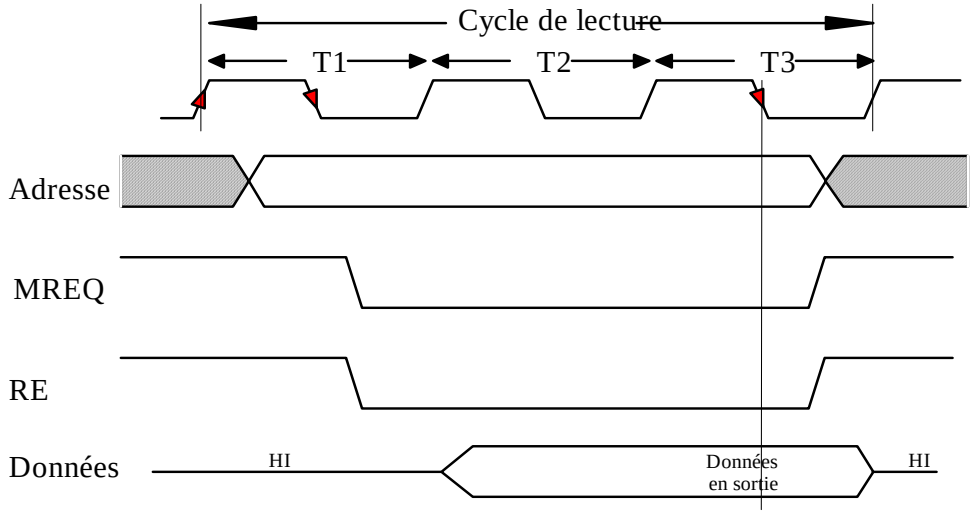


Fig. V.2 : Séquence de lecture sur un bus Synchrone

Au front montant du cycle T1, Le CPU place l'adresse sur le bus d'adresse. Celle-ci ne se stabilise qu'au bout d'un certain retard.
Au front descendant de T1, le CPU active MREQ et RE, le premier indique que c'est un accès

mémoire et non un accès E/S qui réalisé. Le second définit une action de lecture.
Rien ne se passe sur le bus à partir de ce moment pour laisser à la mémoire le temps de décoder l'adresse et de présenter la donnée demandée sur le bus de données. Pendant le front descendant de T3, le CPU capte et enregistre l'information présente sur le bus de données dans un de ses registres internes puis remet les signaux MREQ et RE à leurs positions de repos. Une nouvelle opération de lecture/écriture peut commencer (si nécessaire) au front montant du prochain cycle de bus (T4).
Si la mémoire utilisée était plus rapide, on aurait pu concevoir un cycle de lecture seulement en deux cycle de bus T1 + T2 :

- Front montant de T1 : Appliquer l'adresse
- Front descendant de T1 : Activer les lignes de contrôle MREQ et RE
- Front descendant de T2 : Capter la donnée et désactiver les lignes de contrôle.

Ce type de bus, bien que facile à mettre en œuvre, présente des limitations car un cycle de lecture ou d'écriture doit être définit en fonction des circuits les plus lents. Les circuit plus rapide ne seront pas utilisé au mieux de leurs performances. Ce problème se pause aussi si on essaye d'améliorer les performances d'une machine en remplaçants des composants par d'autres composants plus rapides.

V.2 BUS ASYNCHRONE

Les bus asynchrones ne disposent pas d'horloge pilote du bus. Un cycle de lecture/écriture prend le temps qu'il faut pour réaliser l'opération demandée. Ce type de bus fonctionne selon une technique d'échange dite *"Full handshake"* ou requête et accusé de réception. A la place de l'horloge, les bus asynchrones utilise deux lignes de synchronisation MSYN et SSYN.
Avec la ligne MSYN *(Master SYNchronisation)*, le CPU (Maître) informe un périphérique qu'il désire réaliser un échange. Après réception de ce signal, le périphérique réalise l'opération demandée à sa vitesse propre puis active le signal SSYN *(Slave SYNchronisation)* pour informer le CPU que l'opération est terminée.

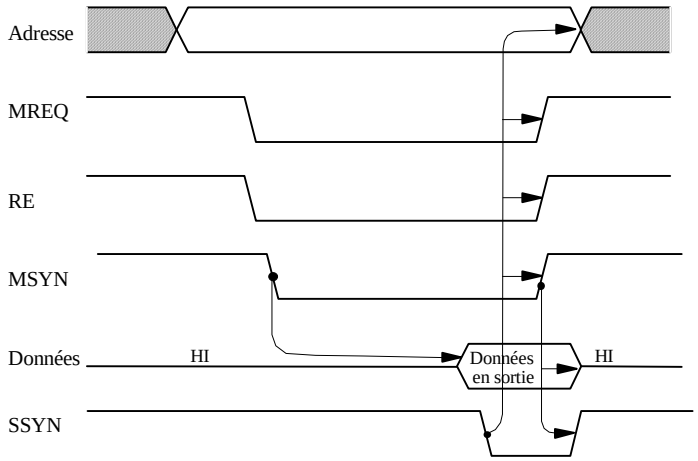


Fig. V.3 : séquence de lecture sur un bus asynchrone

Bien que l'avantage du bus asynchrone soit très net, la plupart des ordinateurs utilisent des bus synchrones. Car la mise au point d'un ordinateur autour d'un bus asynchrone est plus complexe et plus coûteuse. Les bus synchrones sont plus facile à mettre en oeuvre, et peuvent aboutir à de très bonnes performances (meilleures que les bus asynchrones) si on veille à utiliser des circuits qui ne présentent pas une grande dispersion des caractéristiques temporelles et de bien ajuster les cycles de bus sur les timing des circuits.

V.3 LES BUS D'EXTENSION DU PC

Les périphériques d'un ordinateur ne sont pas reliés directement au bus du processeur (bus système). Il sont reliés sur un bus d'extension qui est géré par un contrôleur spécifique.

V.3.1 Le bus ISA : 10 années de bons et loyaux services.

ISA (Industry Standard Architecture) : Bus de référence, apparu en 1979, il équipe le PC depuis ses débuts, de "bus PC" il est devenu en **1987 le "bus ISA"**. Interface commune à tous les PC qui permet à n'importe quelle carte d'extension de prendre place dans n'importe quel PC architecturé autour de ce bus.

Au départ il travaillait su**r 8 bits à une vitesse de 4,77 MHz** avec :

- 20 bits pour les adresses, (soit 1 Mo de mémoire adressable)

- 8 bits pour les données, vitesse de transfert théorique de 4,7 Mo/s en réalité de 1 Mo/s
- 8 lignes d'interruptions, (IRQ) moyen pour prévenir le système de l'appel d'un périphérique afin d'éviter les conflits.
- 4 DMA (Direct MemoryAccess), circuit qui effectue les transferts d'information direct entre un périphérique et la mémoire sans passer par le processeur.

Avec l'arrivé du PC AT, on est passé d'une communication 8 bits à 16 bits (un second connecteur est ajouté à la suite du premier pour fournir un accès aux 8 bits supplémentaires. La fréquence est portée à 8 MHz, l'adressage sur 24 bits permet d'accéder à 16 Mo de mémoire, on dispose de 16 IRQ. La capacité de transfert théorique est de 16Mo/s.

Aujourd'hui certaines cartes mères possèdent toujours ce bus de 16 bits à 8 MHz qui s'avère souvent inadapté aux périphériques modernes en particulier les cartes graphiques mais aussi les cartes réseaux et les contrôleurs disques.

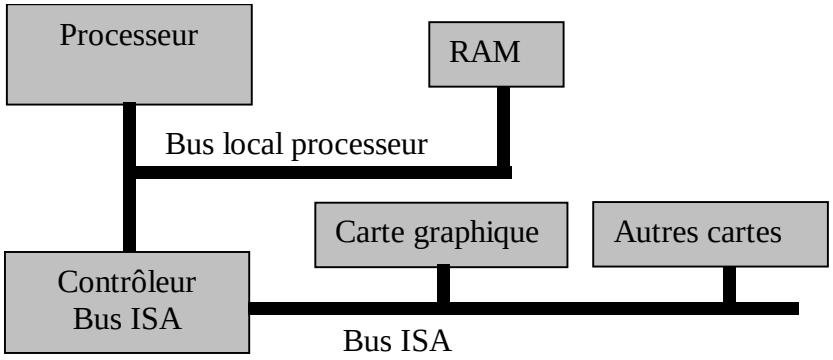


Fig. V.4 : Configuration Bus ISA

V.3.2 Le pari raté du tout 32 bits.

Le bus MCA : 1987 IBM avec les PS/2 propose un nouveau bus le MCA **(Micro Channel Architecture)** travaillant sur 32 bits à une vitesse de 8 MHz. A l'origine le taux de transfert de 20 Mo/s a été porté à 160 Mo/s (mode Burst mais hors norme) La configuration des cartes est automatisée mais ce bus est incompatible avec les cartes ISA existantes ce qui va entraver son développement.

Le Bus EISA : 1988 Les fabricants de clones voulant un bus 32 bits gardant la compatibilité se regroupent autour de Compaq et proposent alors la norme EISA. **(Extended Industry Standard Architecture)**. Ce bus double les contacts du bus ISA il garde une fréquence de 8 MHz. L'utilisateur dispose alors de 32 bits d'adresse (4 Go) et les capacités de transfert se font à 8 Mo/s en mode 8 bits 16 Mo/s en mode 16 bits, et 32 Mo/s en 32 bits. De plus il présente des innovations majeures comme le Bus Mastering qui permet aux cartes d'un serveur de communiquer entre elles sans passer par le processeur. Des mécanismes d'arbitrage sont intégrés pour éviter les conflits entre cartes d'autant plus que les lignes d'interruption peuvent être partagées. Les interruptions sont configurées par logiciel. Le bus EISA accepte aussi bien les cartes 8, 16, 32 bits, mais ce bus qui est conçu pour supporter ces trois format de données différents et assurer les liaisons entre cartes différentes est d'une technologie complexe et chère ce qui va le limiter son utilisation.

V.3.3 Le bus VLB (VESA Local Bus).

A l'époque des 486 et Pentium et l'arrivée des interfaces graphiques (windows), le bus ISA est devenu insuffisant à la gestion des graphiques. Mais tous les périphériques n'ont encore pas besoin de communications rapides sur 32 bits. Quelques solutions sont proposées dont le VESA local bus qui propose de mettre la carte graphique sur le bus local du processeur et de garder les autres cartes sur le bus ISA.

Alors que le bus ISA travaille en 8 MHz, une carte graphique connectée sur en bus local peut fonctionner à 25 ou 33 MHz en parfait synchronisation avec le processeur. Au départ il n'y avait aucune norme, c'est pourquoi l'association VESA (Vidéo Electronics Standards, Architecture) a été créée en 1989 pour définir des standards graphiques et a proposé en juin

1992 le **VLB** (Vesa Local Bus).

Du point de vue matériel, le bus VLB se présente sous forme d'un slot supplémentaire souvent de couleur brun dans le prolongement du connecteur ISA. Les autres connecteurs ISA 16 bits étaient conservés. Ce bus était surtout réservé à la carte graphique, mais comme il était possible d'avoir 3 connecteurs VESA, on pouvait aussi mettre une carte d'interface pour disque dur et les sorties séries et parallèle.

Ce bus, très prometteur au départ, avait des limites de performances notamment pour la charge limitée à 3 cartes et les perspectives de développement très limitées.

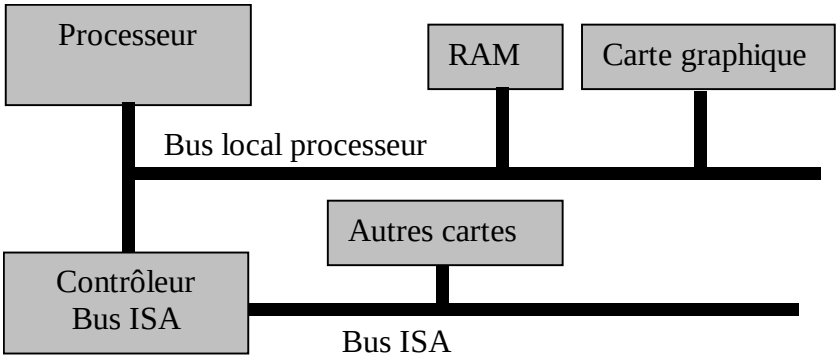


Fig. V.5 : Configuration avec bus local VLB

V.3.4 Le bus PCI (Peripheral Component Interconnect)

Issu d'un projet INTEL lancé en 91, pour rendre plus performant les PC équipé du Pentium. Il permet le transfert des données des cartes vers la mémoire en sollicitant très peu le processeur. A l'origine il gère 32 bits d'adresses et de données à une vitesse de 33 MHz soit une vitesse de transfert d'informations à 33 x 32 / 8 = 132 Mo en théorie. Sa **deuxième version (PCI 2.0)** pour processeur Pentium voit sa largeur étendue à 64 bits pour une fréquence maintenue à 33 MHz soit une vitesse de transfert de 230 Mo/s. La version actuelle **PCI 2.1 adopte une fréquence de 66 MHz (460 Mo/s).** Ce bus ne se substitue cependant pas au bus ISA qui reste l'architecture de base, c'est un complément qui cohabite par l'intermédiaire d'un pont (Host Bridge) et propose jusqu'à 5 connecteurs blancs. Un de ses grands atouts c'est l'auto configuration des cartes utilisant ce bus ce qui permet d'imposer le **Plug and Play**.

Le bus PCI peut lire et écrire directement en mémoire sans passer par le microprocesseur grâce au Host Bridge. De plus les cartes PCI peuvent aussi échanger directement des données entre elles. On désigne cela sous le nom de Busmastering. Cependant sur les 4 emplacements prévus pour les cartes PCI, deux seulement gèrent le Busmastering (généralement ceux qui sont à côté des connecteurs ISA).

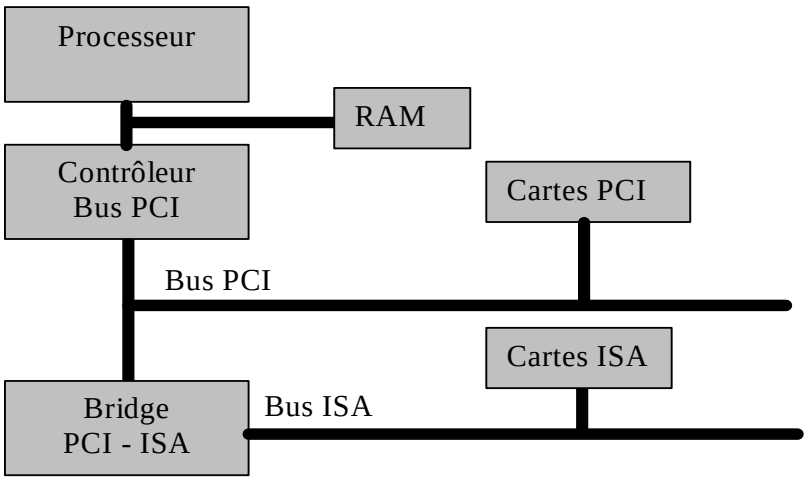


Fig. V.6 : Configuration avec bus PCI

V.3.5 Le bus AGP (Accelerated Graphics Port)

A cause de la limitation de ses spécifications, le bus PCI ne peut assumer des applications utilisant par exemple des graphiques 3D en temps réel. l'AGP (Accelerated Graphics Port), a été **conçu dans le but d'offrir des performances graphiques de haut niveau** (grande bande passante, gros débit) à un prix abordable. Il supporte des échanges allant jusqu'à 528 Mo/sec. Les spécifications d'Intel prévoient trois modes de fonctionnement, 1 x à 264 Mo/s, 2 x à 500 Mo/s et 4 x à 1 Go/s. La gestion de la mémoire est améliorée, possibilité d'adresser directement la mémoire (bénéfice important pour la 3D en temps réel). De plus les textures peuvent être stockées dans la mémoire système plutôt que dans la mémoire de la carte graphique.

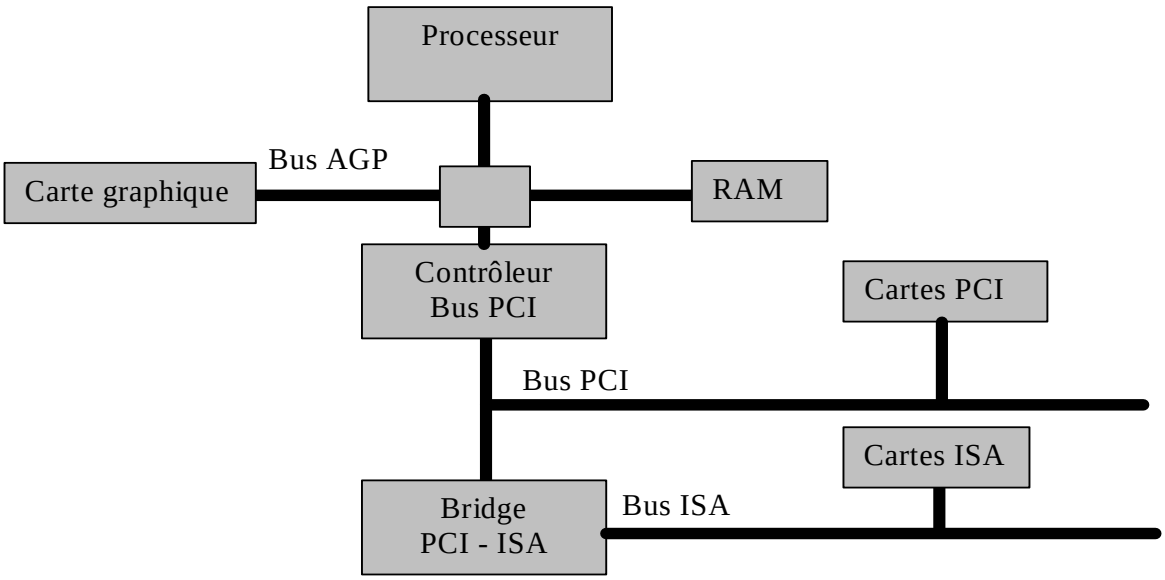


Fig. V.7 : configuration avec bus AGP, PCI et ISA

VI REPRESENTATION DE L'INFORMATION

VI.1 REPRESENTATION DES CARACTERES

L'ordinateur doit pouvoir codifier un jeu de caractères contenant les lettres majuscules et minuscules, les chiffres de 0 à 9, les signe de ponctuation tels que l'espace et la virgule, le signes mathématiques, ainsi que certains caractères un peu plus élaborés. Les caractères sont généralement codés en binaire sur 7 ou 8 bits. Le tableau donnant la correspondance entre chaque caractère et le nombre binaire qui lui a été affecté constitue le code. Le problème est que chaque pays à son propre jeu de caractères, plusieurs codes sont alors utilisés ce qui rend un peut difficile les échanges d'information. Ceci devrait disparaître avec l'arrivé du code INICOD englobant les caractères et signes de tous les pays du globe.

VI.1.1 Code ASCII

Jusqu'à aujourd'hui, le code le plus utilisé pour échanger les informations entre différents pays est le code **ASCII** (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange). Ce code qui a vu le jour aux USA vers 1967 codifie chaque caractère sur 7 bits, ce qui permet de coder 128 caractères différents de 0 à 127. Les 32 premiers caractères sont des caractères de contrôle (non affichables), le caractère de code ASCII 7 est nommé BEL, il permet d'obtenir un beep sur le haut parleur de l'ordinateur.

Binaire			0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
	Hexal		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	0	Déc	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
001	1	16+	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
010	2	32+	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
011	3	48+	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
100	4	64+	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
101	5	80+	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
110	6	96+	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
111	7	112+	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Tableau VI.1 : code ASCII

VI.1.2 Code ISO-*8859*

Afin de pouvoir coder les différents caractères nationaux comme le **é** (France) ou le **ü** (Allemagne), l'association des constructeurs européens d'ordinateur ECMA (*European Computer Manufacturer's Association*) a mis au point une série de 10 tables de caractères codées sur 8 bits ce qui permet de coder 256 caractères. Les 128 premiers caractères sont codés en code ASCII et les 128 autres diffèrent d'un pays à l'autre. Ces nouveaux codes ont été standardisés par l'organisme de standardisation international (ISO) sous la nomenclature ISO8859. Les tableaux ci dessous illustrent ces codes à partir de 160 car les caractères 128 jusqu'à 159 correspondent à des caractères de contrôle peu utilisés.

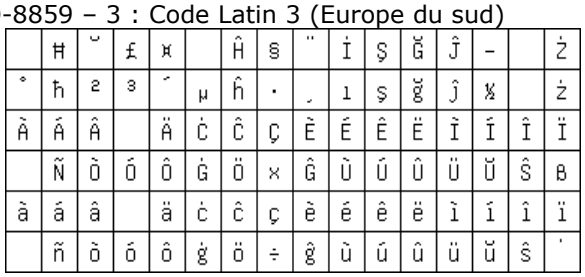
ISO-8859 – 1 : Code Latin 1 (Europe de l'ouest)

160		¡	¢	£	¤	¥	¦	§	¨	©	ª	«	¬	-	®	¯
176	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
192	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
208	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
224	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
240	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

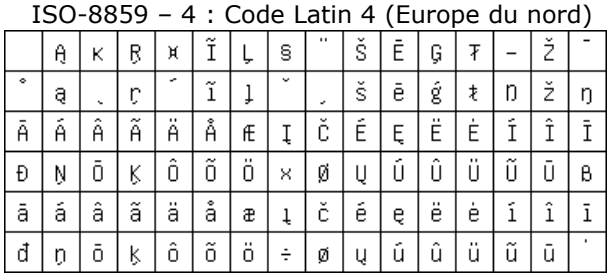
ISO-8859 – 2 : Code Latin 2 (Europe de l'est)

	Ą	˘	Ł	¤	Ľ	Ś	§	¨	Š	Ş	Ť	Ź	-	Ž	Ż
°	ą	˛	ł	´	ľ	ś	ˇ	¸	š	ş	ť	ź	˝	ž	ż
Ŕ	Á	Â	Ă	Ä	Ĺ	Ć	Ç	Č	É	Ę	Ë	Ě	Í	Î	Ď
Đ	Ń	Ň	Ó	Ô	Ő	Ö	×	Ř	Ů	Ú	Ű	Ü	Ý	Ţ	ß
ŕ	á	â	ă	ä	ĺ	ć	ç	č	é	ę	ë	ě	í	î	ď
đ	ń	ň	ó	ô	ő	ö	÷	ř	ů	ú	ű	ü	ý	ţ	˙

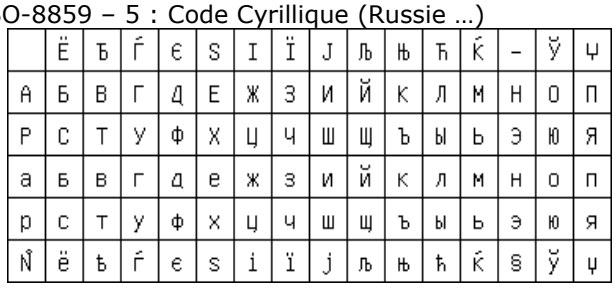
ISO-8859 – 3 : Code Latin 3 (Europe du sud)



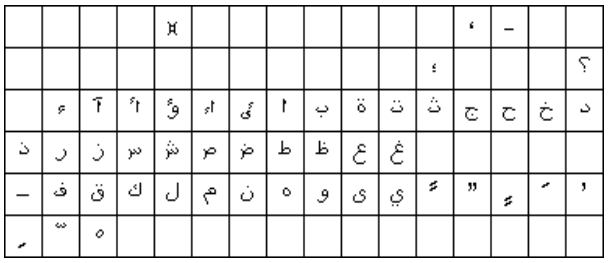
ISO-8859 – 4 : Code Latin 4 (Europe du nord)



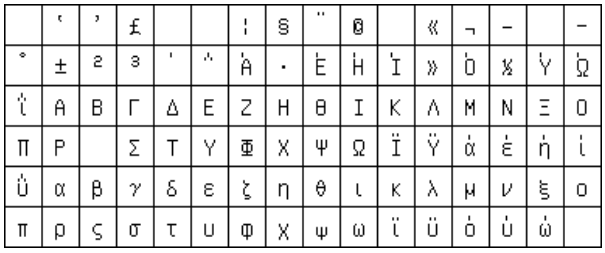
ISO-8859 – 5 : Code Cyrillique (Russie …)



ISO-8859 – 6 : Arabe



ISO-8859 – 7 : Grec



ISO-8859 – 8 : Hébreux

ISO-8859 –9 : Code Latin 5 (Turque)

ISO-8859 – 10 : Code Latin 6 (Pays baltique)

VI.2 REPRESENTATION DES NOMBRES ENTIERS

Les nombres entiers sont représentés en **binaire naturel**. Chaque chiffre est pondéré par un poids qui est un une puissance de 2. Le nombre de bits utilisé dépend des machines et aussi des langages de programmation utilisés

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

1	0	0	1	0	0	0	1	= 128+16+1=145

VI.2.1 La représentation des nombres entiers signés

Les nombres entiers signés sont les nombres entiers relatifs. Il y a les nombres positifs et les nombres négatifs.

Les nombre positifs : La représentation des nombres positifs se fait en binaire naturel.

Les nombres négatifs : Plusieurs méthodes existent pour représenter les nombres négatifs. La plus utilisée est la méthode du complément. Cette méthode consiste à représenter le nombre négatif (-n) par un nombre qui, ajouté au nombre positif (+n) donne 0. Cette méthode

fait que la représentation d'un nombre négatif va dépendre du nombre de bits utilisé.

Considérons une machine qui travaille sur 4 bits, Cette machine peut représenter 16 (2^4) nombres non signé comme représenté sur le tableau. Si on fait l'opération 12 + 6 on obtient 3 car notre machine n'a que 4 bits et ne peut représenter le 5ème bit dont le poids est égal à 16. On peut donc dire que notre machine travaille modulo 16, elle donne le reste par rapport à 16. Dans ce cas, 16 est représenté par 0. On peut vérifier que 8+8=9+7+…=16 = 0

12	1	1	0	0
+7	0	1	1	1
=3	0	0	1	1

0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

On peut donc conclure que pour cette machine, l'opposé (négatif) d'un nombre n est son complément m par rapport à 16 : n + m = 16 = 0
Ex : L'opposé de 2 est 14, donc -2 = 1110

+n		-n		
0000	+0			
0001	+1	-1	15	1111
0010	+2	-2	14	1110
0011	+3	-3	13	1101
0100	+4	-4	12	1100
0101	+5	-5	11	1011
0110	+6	-6	10	1010
0111	+7	-7	9	1001
		-8	8	1000

Une méthode rapide pour obtenir la représentation d'un nombre négatif est la méthode du **complément à 2** :

1. On représente le nombre positif en binaire naturel (sur le nombre de bits de la machine)
2. On fait le complément à 1 : on complémente tous les bits 1 par 1
3. on ajoute 1 au résultat ce qui donne le complément à 2

Exemple :

Donner la représentation de -6 sur 4 bits et sur 8 bits

Sur 4 bits :

```
6       →   0110
cà1     →   1001
            +  1
cà2     →   1010
```

Sur 8 bits :

```
6       →   00000110
cà1     →   11111001
            +      1
cà2     →   11111010
```

VI.3 REPRESENTATION DES NOMBRES REELS

Comme pour les autres types d'information, pour représenter un nombre réel, on dispose d'un nombre fini de bits et on peut imaginer différentes façons de représenter un réel avec cet ensemble de bits. Les deux méthodes de représentation les plus connue sont la représentation en virgule fixe et la représentation en virgule flottante. C'est cette dernière qui est la plus utilisée.

VI.3.1 La représentation en virgule fixe

il s'agit de la représentation habituelle comme on le fait sur papier sauf pour le signe et la virgule. Pour le signe, on réserve un bit, en général le bit de gauche. La virgule quant à elle n'est pas représenté, on travaille avec une virgule implicite située à une place bien déterminée.

Exemple :

- Représentation sur 9 bits
- bit de gauche réservé au signe (0 = positif, 1 = négatif)
- les autres bits représentent la valeur absolue
- la virgule est placée à gauche du $4^{ème}$ bit à partir de la droite

| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | $= + (1100,0111)_2 = (12,4375)_{10}$

Rappelons que les valeurs de pondération pour le système binaire sont :

$$\ldots 128\ 64\ 32\ 16\ 8\ 4\ 2\ 1\ ,\ \frac{1}{2}\ \frac{1}{4}\ \frac{1}{8}\ \frac{1}{16}\ \frac{1}{32}\ \frac{1}{64}\ \frac{1}{128}\ \ldots$$

Rappelons aussi qu'une méthode systématique pour convertir un nombre réel de décimal à binaire consiste à convertir la partie entière par divisions successives et la partie fractionnelle par multiplications successives, on arrête la multiplication quand on a un résultat nul ou quand on estime qu'on a suffisamment de chiffre après la virgule.

0.4375 x 2 = 0.875 → 0
0.875 x 2 = 1.75 → 1
0.75 x 2 = 1 .5 → 1
0.5 x 2 = 1 .0 → 1 ⇨ 0.4375 = 0,0111

Cette représentation n'est pas très utilisée car elle a une faible précision à cause des chiffres perdus à droite de la virgule. On peut vérifier qu'avec la représentation de l'exemple précèdent, 011000111 représente aussi bien le nombre +12,4375 que le nombre +12,4999. De même, les deux nombres +0.5 et 0.56 seront tous les deux représenté par 000001000.

Un autre problème de cette représentation est qu'elle ne permet pas de représenter les nombres très grands.

VI.3.2 La représentation en virgule flottante

Les scientifiques ont depuis longtemps appris à s'accommoder avec les problème de la virgule fixe en utilisant la représentation scientifique dans laquelle un nombre est représenté par le produit d'une mantisse et d'une puissance de 10 (décimal) ou d'une puissance de 2 (binaire). La représentation des nombres sur un ordinateur selon cette méthode est désignée par représentation en virgule flottante.

$R = - 4,463876\ 10^{23}$ (décimal)
$R = + 5\ 10^{-56}$ (décimal)
$R = - 1,1001110\ 2^{110011}$ (binaire)
$R = + 1,11110\ 2^{-10011}$ (binaire)

Dans les ordinateurs, la représentation se fait évidement en binaire, il faut maintenant se

mettre d'accord comment répartir le nombre de bits dont en dispose pour représenter :

- Le signe du nombre
- la mantisse
- l'exposant signé

Pour pouvoir échanger des documents, il est indispensable que tout le monde utilise la même représentation.

VI.3.2.1 La norme IEEE-754

C'est la norme recommandée par *the Institute of Electrical and Electronics Engineers*, elle est utilisée sur la quasi-totalité des ordinateurs pour représenter les nombres réels.

Chaque nombre est représenté par :

- 1 bit pour le signe
- N_e bits pour l'exposant signé
- N_m bits pour la valeur absolue de la mantisse

Signe	Exposant	Mantisse

On distingue plusieurs variantes dont les plus utilisée sont :

	Nombre de bits	Signe	Exposant	Mantisse
Simple Précision	32	1	8	23
Double précision	64	1	11	52

VI.3.2.2 IEEE-754 simple précision

1	8	23
Signe	Exposant	Mantisse

Signe :

Le bit tout à fait à gauche est réservé au signe : 0→positif, 1 → négatif

L'exposant :

L'exposant est représenté sur 8 bits ce qui permet de représenter 256 nombres différents allant de 0 = 00000000 à 255 = 11111111.

Les deux combinaisons d'extrémité c'est-à-dire 0 et 255 sont réservés pour représenter des nombres particuliers (que nous allons voir un peu plus loin). Il reste donc 254 combinaisons allant de 00000001 à 11111110. Comme l'exposant est un nombre signé, les 254 combinaisons permettent de représenter les nombres allant de -126 à +127. Ce n'est pas la représentation en complément à 2 qui est utilisée mais la représentation biaisée. Pour représenter un exposant de l'intervalle [-126,+127] on lui ajoute un biais (décalage) suffisamment grand pour l'amener dans l'intervalle [1,254] afin qu'en puisse le représenter en binaire naturel. La valeur du Biais est donc B=127.

Exposant E	Exposant biaisé E_B=E+B	
127	254	11111110
126	253	
-125	2	00000010
-126	1	00000001

La mantisse :

On dispose de 23 bits pour représenter la mantisse. Avant cela il faut la normaliser ou la

cadrer c'est-à-dire la ramener dans l'intervalle [1 , 2[autrement dit il faut avoir :

$1 \leq$ mantisse < base de numération

La normalisation de la mantisse revient à déplacer la virgule ce qui va forcément modifier l'exposant. Chaque fois qu'on déplace la virgule d'une position à gauche il faut augmenter l'exposant de 1 et inversement.

R = 1100111,110111 = **1,**100111110111 2^6
R = 1100111,110111 2^{-3} = **1,**100111110111 2^3
R = 0,0011101 = **1,**1101 2^{-3}

En résumé il faut que la mantisse soit sous la forme 1,xxxxxx, Avec cette convention, on a pas besoin de stocker le 1 ce qui nous ferait perdre une position pour rien, on ne stocke alors que la partie à droite de la virgule

On peut maintenant essayer de voir quelques exemples :

- **R = + 110011,0101111**
  - On commence par normaliser ➔ R = 1,100110101111 2^5
  - On détermine ensuite la représentation biaisée de l'exposant ➔ E_B = 5+127= 132 = 10000100

R ←→ | 0 | 10000100 | 10011010111100000000000 |

- R ←→ | 1 | 11100111 | 11100111001110011100111 |
  - c'est un nombre négatif
  - E_B = 11100111 = 228 ⇨ E = 231 – 127 = 104
  - R = - 1,11100111001110011100111 $2^{104} \approx$ - 1,9 $2^{104} = \approx$ - 3,86 10^{31}

Revenons maintenant aux 2 combinaisons de l'exposant non encore utilisé soit 00000000 et 11111111

- E_B = 00000000 est utilisé pour représenter les nombres voisins de 0 c.à.d inférieurs à 2^{-126} qui est le plus petit nombre qu'on peut représenter par le système décrit ci-dessus soit :

2^{-126} = 0 00000001 00000000000000000000000

Si on note m la partie fractionnelle de la mantisse, on distingue deux situations :

  - m = 0 ⇨ le nombre représenté R = 0
  - m ≠ 0, ⇨ c'est un nombre dénormalisé car on va utiliser un biais B=126 et une mantisse de la forme 0,xxxxx. Ceci va nous permettre de représenter les nombre compris entre 0 et 2^{-126} , il est vrai avec une précision moindre puisqu'on peut avoir des mantisse de la forme 0,00000000xxxx, mais il vaux mieux les représenter avec moins de précision que ne pas les représenter du tout
  -

- E_B = 11111111 cette combinaison est utilisé pour indiquer un résultat qui n'est pas un nombre réel représentable comme les nombres qui dépassent le plus grand nombre que l'on peur représenter (infini) ou quand on utilise une opération arithmétique non définie comme 0/0 ou log(-1) on parle alors d'un pseudo nombre appelé NaN *(Not a Number)*.
  - m = 0 ⇨ R = ∞
  - m ≠ 0, ⇨ R = NaN

Récapitulation

E_B	m	Nombre représenté

11111111	≠ 0	NaN
	= 0	∞
11111110 ↑ 00000001	quelconque	$(-1)^S \times 1,m \times 2^{E_B-127}$
00000000	≠ 0	$(-1)^S \times 0,m \times 2^{-126}$
	= 0	0

Le plus petit nombre (positif) représentable

0 00000000 00000000000000000000001 = 0,00000000000000000000001 2^{-126} = 2^{-149} ≈ 1,4 10^{-45}

Le plus petit nombre (positif) normalisé

0 00000001 00000000000000000000000 = 1,0 2^{1-127} = 2^{-126} ≈ 1,1755 10^{-38}

Le plus grand nombre (positif) représentable

0 11111110 11111111111111111111111 = 1,11111111111111111111111 $2^{254-127}$
= $(2^{24} - 1) \times 2^{104}$ = 3,40282346638528859811704183484516925440 ≈ 3,4 10^{38}

VI.3.2.3 IEEE-754 double précision

La représentation en double précision se fait façon tout à fait similaire. La représentation se fait 64 bits avec 11 bits pour l'exposant et 52 pour la mantisse

1	11	52
Signe	Exposant	Mantisse

Pour les nombre normalisés (E_B = -1022 à 1023) on utilise un biais B=1023.
Pour 0, les nombres non normalisé, ∞ et NaN on utilise exactement le même principe utilisé pour la simple précision

E_B	m	Nombre représenté
11111111111	≠ 0	NaN
	= 0	∞
11111111110 ↑ 00000000001	quelconque	$(-1)^S \times 1,m \times 2^{E_B-1023}$
00000000000	≠ 0	$(-1)^S \times 0,m \times 2^{-1022}$
	= 0	0

VI.3.3 Conversion base 10 vers VF Simple Précision

Pour déterminer la représentation en virgule flottante d'un nombre, il faut réaliser les étapes suivantes

1) Convertir la partie entière en binaire par divisions successives, on obtient N1 bits
2) Convertir la partie fractionnaire en binaire par multiplications successives, on obtient N2 bits. On arrête la multiplication quand la partie fractionnaire du résultat est nulle ou quand le nombre de bits N2 obtenu est tel que N1+ N2 = 23+1. Si N1 est nul, il faut compter N2 à partir du premier bit non nul.
3) Normaliser le résultat binaire sous forme 1,xxxx..... 2^E. Chaque fois qu'on déplace la virgule à gauche, il faut incrémenter l'exposant et inversement.
4) Biaiser l'exposant en lui ajoutant le biais.
5) Convertir l'exposant biaisé en binaire naturel
6) Ecrire le résultat sous forme : | s | E_B | m |

Exemple :
Déterminer la représentation en VF-SP du nombre R = 25,34
1) On commence par convertir 25 en binaire en trouve 11001 (N1=5)
2) On convertit 0,34 en binaire, on s'arrête à N2=19 bits ou si on tombe sur résultat nul.
   On obtient 0101100110011001100 R = 11001, 0101011100001010001
3) On normalise le résultat ce qui donne R = 1,10010101011100001010001 2^4
4) L'exposant biaisé est E_B = 4 + 127 = 131 qu'on convertit en binaire : 10000011
5) La représentation de 25,34 en virgule flottante est donc 0 10000011 10010101011100001010001
   soit $(41CABB51)_H$
   Les langages évolué comme le C calcule avec plus de 23 chiffre après la virgule et arrondissent le dernier chiffre, ne soyez pas étoné si une machine vous dit que $(25.34)_{10}$ → $(41CABB52)_H$

VI.3.4 Conversion virgule flottante SP vers décimal

Pour convertir un nombre représenté en virgule flottante vers la base 10, on réalise les étapes suivantes

1) Le bit le plus à gauche est le bit de signe : 1➔ négatif , 0➔ positif
2) Convertir l'exposant (biaisé) en décimal
3) Retrancher le biais du résultat pour obtenir la vraie valeur E de l'exposant
4) Convertir la mantisse en décimal. Attention la mantisse est un nombre fractionnaire, les poids sont constitués des puissances négatives de 2.
5) Le résultat en décimal est : s 1,m 2^E

Exemples :

R = 1 10000011 10101011111000000000000
Signe = 1 ➔ nombre négatif
E_B = 10000011 = 131 ⇨ E = 131 – 127 = 4
Mantisse = 101010111110 = 1/2+1/8+1/32+1/128+1/256+1/512+1/1024+1/2048 = 0.6713867
R = - 1.6713867 * 2^4 ≈ - 26.74422

VI.3.5 Addition et soustraction en VF

Pour réaliser la somme de deux nombres signés R = A + B, on procède selon le signes des opérandes :

1) Si A et B sont du même signe (somme)
   - Le signe de R est le même que celui de A et B : s(R) = s(A) = s(B)
   - Cadrer un des deux nombres pour avoir le même exposant
   - Sommer les mantisses
   - Normaliser

2) Si A et B sont de signes différents (soustraction)
   - Cadrer un des deux nombres pour avoir le même exposant
   - Soustraire la plus faible mantisse de la plus forte.
     0 – 1 = 1 → 1 0 – 1 – 1 = 0 → 1 1 – 1 – 1 = 1 → 1
   - Normaliser
   - Déterminer le signe m(+) > m(-) ⇒ s = 0
     m(+) < m(-) ⇒ s = 1

VI.3.6 Multiplication en VF

Pour réaliser le produit de deux nombres signés R = A × B, on procède comme suit :
- Multiplier les mantisses
- Additionner les exposant
- Normaliser le résultat
- Définire le signe : si A et B de même signe ⇒ s = 0, si A et B de signes différents ⇒ s = 1.

VII RETOUR SUR QUELQUE ASPECTS FONDAMENTAUX

VII.1 LE CHIPSET

Pratiquement toutes les fonctions de l'ordinateur comme la gestion du bus, des interruptions, des périphériques d'E/S ont été intégrés dans un ou deux composants appelés Chipset. Le but recherché est d'améliorer les débits de données entre le processeur, la mémoire centrale, la mémoire vidéo le disque dur et autres périphériques.

Le chipset est généralement composé d'au moins deux puces distinctes: Le NorthBridge (PONT NORD)et Le SouthBridge (PONT SUD).

Le Pond Nord est la partie la plus proche du processeur donc celle qui fonctionne à la fréquence la plus élevée et qui contrôlera les éléments les plus rapides du PC. Ainsi il gère les échanges avec la mémoire, le bus AGP (pour les cartes graphiques).

Le pond Sud est relié au pont nord par le bus système, il gère les entées/sorties : Contrôleur IDE (pour les disques durs et CD-ROM) , port USB , port Parallèle , ports Série , port PS2 (pour souris et clavier) et Bus PCI pour les cartes d'extension additionnelles.

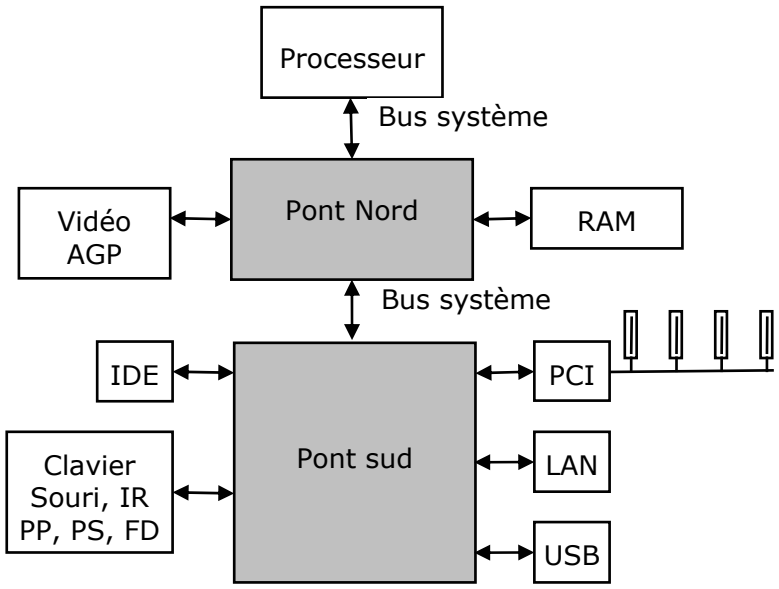


Fig. VII.1 : schéma simplifié d'une chipset de type PC

VII.2 LE CONTROLEUR DMA

Quand un périphérique (disque dur) a besoin d'échanger une grande quantité d'information avec la mémoire centrale, le transfert doit normalement être réalisé par le processeur en exécutant un programme qui réalise le transfert mot par mot ce qui monopolise le processeur et les bus d'adresse et de données.

Pour décharger le processeur de cette tache, le Chipset comporte des circuits dédiés pouvant réaliser ces transferts beaucoup plus rapidement tout en permettant au processeur de continuer à travailler. Ces circuits sont les contrôleurs DMA (DMAC : Direct Memory Access Controler).

Lorsqu'un DMAC reçoit une demande de transfert d'un périphérique (ligne de contrôle DREQ), il transmet une requête "Bus Request" au processeur pour demande de prendre possession du Bus (ligne HOLD), le processeur renvoie un acquittement "Bus acknowledge" (ligne HLDA). Le DMAC devient alors maître du bus, il le signale au périphérique (ligne DREQi) et commence le transfert de données en supervisant des cycles de lecture/écriture à travers le Bus.

Dans la pratique, les machines disposent de plusieurs contrôleurs DMA ayant plusieurs canaux chacun. Les canaux sont spécialisés réalisant chacun une tache spécifique comme :

- Rafraîchissement de la mémoire dynamique
- Echange avec le disque dur
- Echanges avec le lecteur de disquette
- . . .

VII.3 LA GESTION DES INTERRUPTIONS

Une interruption est un évènement asynchrone conduisant le processeur à délaisser le programme qu'il est en train d'exécuter et d'aller exécuter le programme correspondant à l'interruption qu'on appelle vecteur d'interruption. Une interruption peut être matérielle ou logicielle.

Une interruption matérielle est provoquée par l'application d'une impulsion sur une ligne d'interruption du processeur. Celui-ci en possède plusieurs, chacune réservée à une tache particulière, la plus connue est la ligne RESET qui permet de réinitialiser le processeur.

Comme le processeur ne peut pas avoir beaucoup de lignes d'interruption physiques, on a introduit la possibilité de demander une interruption au processeur à l'aide d'une instruction (INT) que l'on insère dans les programmes utilisateur. L'instruction d'interruption possède au moins un paramètre obligatoire qui est le numéro d'interruption permettant au processeur d'aller exécuter le vecteur d'interruption approprié.

La famille des processeurs Intel 80x86 dispose de 256 interruptions logicielles différentes. Les vecteurs d'interruption font partie, soit du bios (interruptions du bios) soit du DOS (interruptions du dos)

Les interruptions sont hiérarchisées, certaines sont plus prioritaires que d'autres. Lors de l'exécution d'une interruption "*importante*", les interruptions mois prioritaire sont masquées *"de sortent à ce qu'elles ne viennent pas déranger".* Les interruptions importantes ne sont pas masquables (Interruptions NMI)

Pour gérer tout ça, le Chipset dispose contient un contrôleur d'interruption qui est un circuit dédié à la gestion des interruptions. son rôle consiste essentiellement à :

- Prendre en compte les interruptions lorsqu'elles arrivent
- Etablir les priorités entre les demandes
- autoriser le masquage de certaines interruptions
- gérer les numéros d'interruption

VII.4 LA MEMOIRE VIRTUELLE

Lorsque la mémoire était chère, on ne pouvait pas se permettre de disposer d'une mémoire suffisamment importante capable de mémoriser un gros programme dans sa totalité. *(Les premiers PC avaient une RAM de 64 ko)* Maintenant que la mémoire est moins chère, le problème reste posé car les programmes sont de plus en plus gros et avec le traitement multitâche, on a plusieurs programmes et pilotes de matériels qui se partagent la mémoire.

La solution de la mémoire virtuelle consiste à utiliser la mémoire de masse comme une extension de la RAM. Mais on ne peut pas travailler directement dans la mémoire de masse à cause de sont temps de réponse exorbitant. La solution consiste à travailler dans la RAM mais de ne charger dedans que la partie du programme dont on a immédiatement besoin.

Pour faciliter la gestion, et rendre les choses transparentes pour l'utilisateur, ce dernier disposait d'un espace mémoire très important qu'il utilise sans se poser de question. En réalité, cette mémoire n'existe pas physiquement puisque la RAM dont on dispose a une capacité beaucoup plus faible et le système se charge de la compléter avec de l'espace dans le disque dur. On dispose donc de deux espaces mémoire, une mémoire **virtuelle** utilisée par le programmeur et une mémoire physique où sont réellement stockées les informations. Un contrôleur de mémoire virtuelle (matériel+ logiciel) se charge de faire la correspondance entre les deux mémoires. Quand le processeur (programmeur) fournit une adresse virtuelle, le contrôleur l'utilise pour déterminer l'adresse physique où est stockée l'information.

Par exemple un processeur avec un bus d'adresse de 32 bits peut adresser un espace mémoire de 4Go. Si la mémoire réelle est seulement de 128 Mo, le programmeur ne doit pas s'en soucier, des mécanismes transparents gérés par le système d'exploitation assurent la correspondance entre les adresses virtuelles et les adresses réelles (RAM + DD) donnant ainsi la possibilité d'exécuter un programme de près de 4Go.

VII.4.1 La pagination *(swapping)*

Pour faciliter la gestion de la mémoire virtuelle, celle-ci est découpée en pages (blocs de 4ko) de taille fixe. Chaque page est identifiée par un numéro de page VPN *(Virtual Page Number).* La mémoire physique est aussi divisée en page de la même taille, chaque page est identifiée par un numéro de page physique PFN *(Page Frame Number)*. La tache du contrôleur

de mémoire virtuelle et de réaliser la correspondance entre les VPN et les PFN.

A l'exécution d'un programme, seulement les première pages le constituants sont chargées en mémoire, au fur et à mesure que l'exécution avance, d'autres pages sont chargées et viennent remplacer les pages qui ne sont plus utilisées, c'est le mécanisme de pagination.

Parmi les questions auxquelles est confronté un système de gestion de mémoire virtuelle, on peut citer :

- Comment savoir si une page se trouve déjà en mémoire et où se trouve-t-elle exactement
- Comment convertir les VPN en PFN
- Quelle page remplacer pour faire place à une nouvelle page.
- Comment savoir si la page sortant a été modifiée afin de mettre à jour la copie restée sur le disque.
- . . .

Un mécanisme essentiel aidant à répondre à ces questions est **la table des pages** gérée par le système d'exploitation. Cette page contient un enregistrement pour chaque page virtuelle. Chaque enregistrement contient le n° de la page virtuelle, un bit P qui indique si elle présente ou non dans la RAM, un champ qui indique son adresse physique dans le disque dur, un autre champ pour son adresse physique dans la RAM (si P=1) ainsi que d'autres attributs comme un bit D *(Dirty)* qui est placé à 1 si la page mémoire est modifiée, un bit U/S pour indiquer si la page appartient au système ou à un utilisateur, un bit bits R/W utilisé pour la protection …

Le mécanisme de transformation d'adresse est assez simple, le processeur envoie une adresse sous la forme ***page:offset*** qui indique l'adresse du début de la page et l'adresse de la donnée au sein de la page, le contrôleur de mémoire virtuelle en déduit le VPN, localise l'enregistrement correspondant dans la table des pages, en sort avec le PFN qui lui permet de localiser la page physique soit dans la RAM soit dans le disque dur. L'offset de l'adresse virtuelle sera utilisé tel quel dans l'adresse physique car les pages on la même taille.

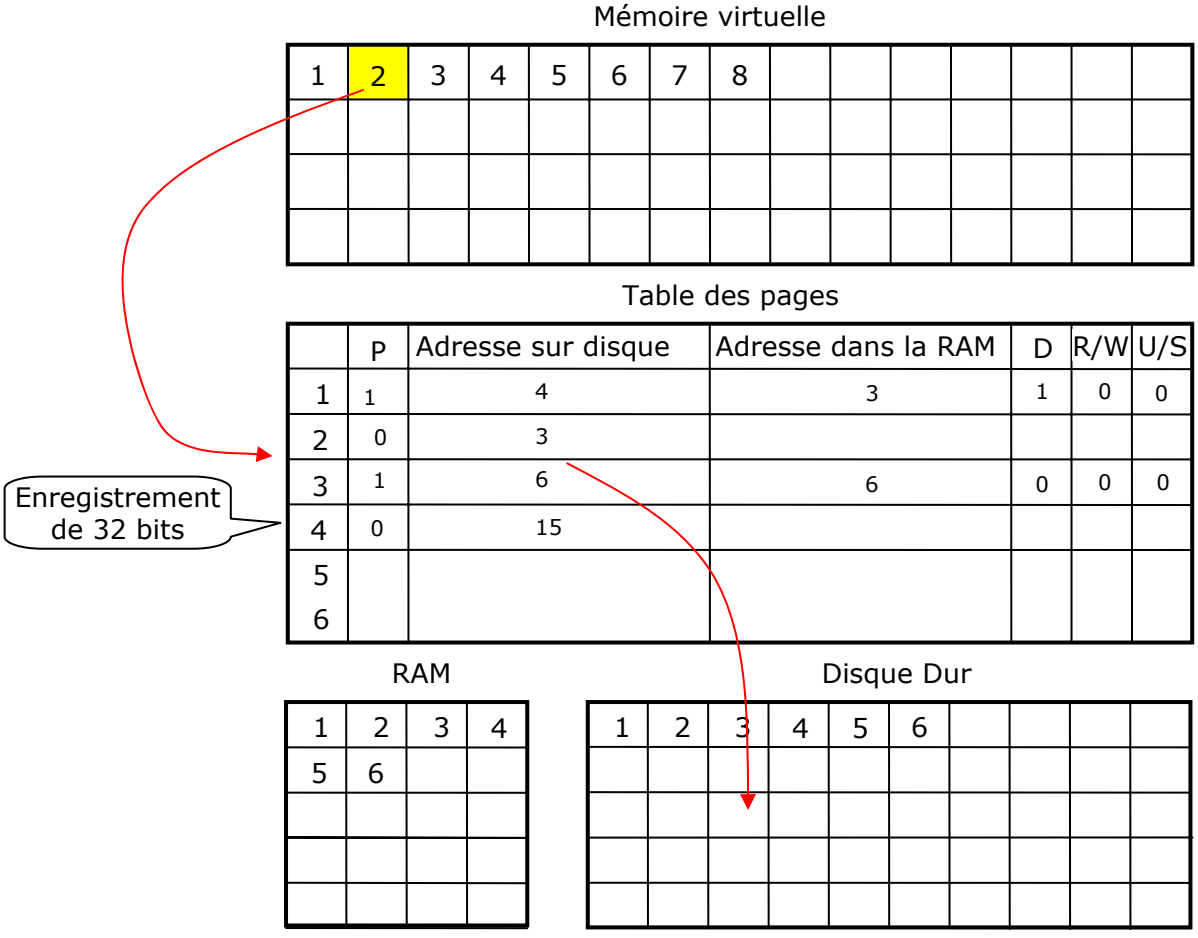


Fig. VII.2 : Gestion de la mémoire virtuelle

VII.4.2 Mécanisme de remplacement

Lorsqu'on veut accéder à une page et on trouve que celle-ci n'est pas dans la RAM (P=0), on dit qu'il y a un **défaut de page**. Il faut alors chercher la page en disque dur et la charger dans la mémoire. Si toutes les pages de la mémoire sont utilisées, il faudra en remplacer une.

Parmi les stratégies adoptées pour le choix des pages à remplacer, la plus efficace est celle qui consiste à remplacer la page la moins récemment utilisée *(LRU : Less Recently Used)*. Pour localiser une page LRU, plusieurs techniques peuvent être envisagées, l'une d'elle consiste à jour une petite table (qu'on peut appeler table de classement) contenant les numéros des pages résidant dans la RAM et qui fonctionne comme une pile. Chaque fois qu'une page est utilisée, son numéro est placé en tête de la table. Ainsi la page dont le numéro se trouve en fin de pile est la page la plus ancienne et la meilleure candidate pour être remplacée.

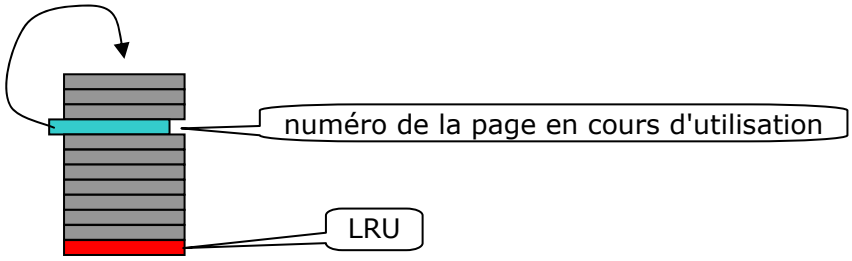


Fig. VII.3 : table de classement

VIII LES ENTREES SORTIES

Nous allons voir dans ce chapitre comment l'ordinateur communique avec les périphériques qui sont des dispositifs matériels qui vont permettre d'assurer l'entrée et la sortie d'information entre l'ordinateur et le monde extérieur ou bien de conserver ces informations de façon permanente. L'environnement extérieur peut être de 2 type :

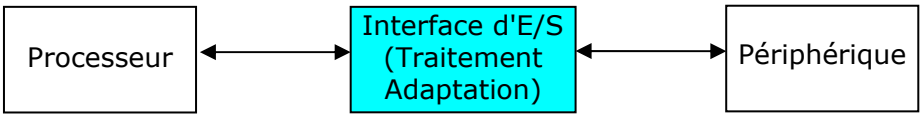


Fig. VIII.1 : interface d'entrée sortie

- Un environnement numérique : affichage, clavier, souris, Disque dur, Scanner, Imprimantes ...

- Un environnement analogique : Audio (microphone, haut-parleurs), Signaux issus de capteurs de température, d'humidité, de pression, de position ou tout autre information analogique. Comme l'ordinateur ne sait traiter que des informations numériques, la communication avec le monde analogique nécessite une conversion de l'information analogique en information numérique.

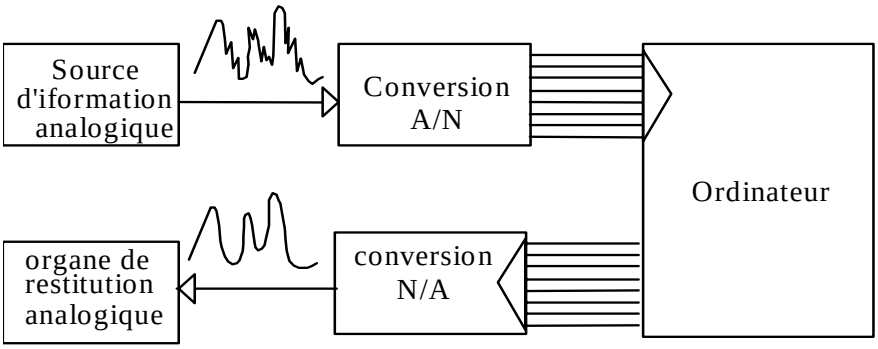


Fig. VIII.2 : Interfaçage analogique

Les signaux utilisés par les périphériques diffèrent sur de nombreuses caractéristiques. Le signal utilisé pour piloter un écran ne ressemble en rien à celui d'un haut parleur. Il faut donc des interfaces qui adaptent les signaux du processeur avec ceux des périphériques. On les appelle interface d'E/S, leur rôle et de recevoir l'information venant du processeur et de la transformer avant de l'envoyer au périphérique.

Les Interfaces d'E/S sont relié au processeur par l'intermédiaire du bus, Pour choisir une interface parmi plusieurs, il faut réaliser un décodage d'adresse. Chaque contrôleur dispose d'une ou plusieurs adresses. Il est possible d'avoir les adresses dans l'espace mémoire ou dans un dans un espace séparé. Intel a choisi de séparer les espaces en utilisant la ligne M/IO du bus de contrôle. Si M/IO=1, on accède à la mémoire, si M/IO=0 on accède à un registre d'un contrôleur d'E/S. ceci se traduit au niveau de la programmation par l'utilisation de deux instruction de transfert différentes, L'instruction MOV pour la mémoire et les instruction IN/OUT pour les E/S.

VIII.1 LES TECHNIQUES DE GESTION

L'ordinateur dispose de plusieurs périphériques et ceux-ci peuvent émettre des requêtes vers le processeur, comment gérer globalement tous les périphériques et éviter les conflits ? Plusieurs solutions existent.

VIII.1.1 La scrutation

Chaque périphérique dispose d'un indicateur (bit dans un registre d'état) que le processeur vient tester régulièrement. Si cet indicateur est positionné, le processeur comprend que le périphérique en question désire réaliser un échange d'information. Par exemple si on a appuyé sur une touche du clavier, celui-ci mémorise la touche dans sa mémoire tampon et positionne son indicateur pour l'informer qu'un caractère attend d'être lu.

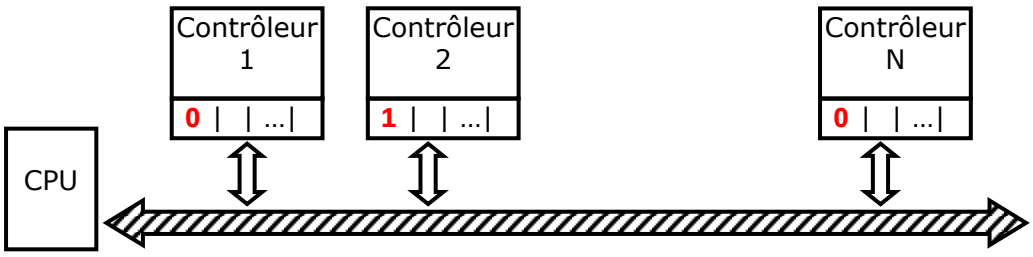


Fig. VIII.3 : gestion par scrutation

Cette technique nécessite que le processeur exécute en permanence une boucle de scrutation qui visite sans arrêt tous les contrôleur d'E/S. l'inconvénient de cette technique est qu'elle peut avoir un certain retard entre le moment ou le contrôleur positionne son indicateur est le moment où son tour de scrutation arrive. En plus, le temps d'exécution est partagé entre le programme de scrutation et les autres programmes entrain de s'exécuter sur le CPU.

VIII.1.2 Les Interruptions

Une autre méthode consiste à utiliser la technique des interruptions : une ligne d'interruption relie le processeur et le contrôleur. Celui-ci enverra un signal au processeur par cette ligne uniquement s'il a quelque chose à lui communiquer. le processeur peut alors soit lui répondre immédiatement soit le faire attendre un peu s'il est en train de faire quelque chose d'important (c'est un peu comme quand le téléphone sonne).

Avec une seule ligne d'interruption, le problème qui se pose pour le processeur est de savoir qui a déclanché l'interruption. Plusieurs solutions peuvent être envisagées, la plus utilisée consiste à utiliser un contrôleur d'interruption qui sera relié à tout les périphériques et qui aidera le processeur à gérer les interruption. C'est ce que fait le PC en utilisant le contrôleur d'interruption 8259.

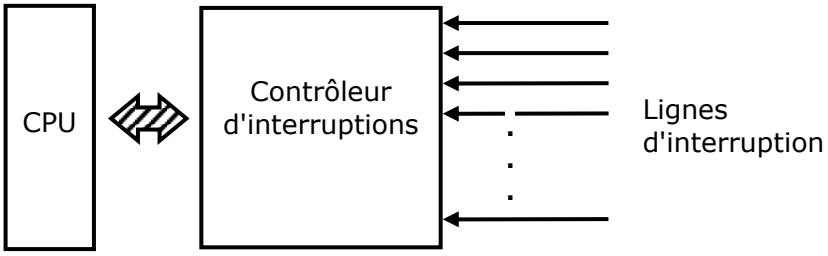


Fig. VIII.4 : gestion à l'aide d'un contrôleur d'interruptions

Quand le contrôleur reçoit une interruption venant d'un périphérique il envoie un octet vers le CPU pour l'informer de quelle interruption s'agit-il. Il réalise aussi d'autre tache comme la gestion des priorités entre les différentes demandes, le masquage de certaine interruption …

VIII.2 ENTREES/SORTIES PARALLELES

En communication parallèle, on utilise un bus de 8 lignes pour transmettre les bits constituant les caractères. Les 8 bits sont transmis simultanément 1 sur chaque ligne, c'est pour ça qu'on parle de transmission parallèle.

Les circuits qui permettent l'interfaçage parallèle avec le monde extérieur sont les PIA (Parallel *Interface Adapter*) .

Le PIA apparaît au processeur comme une (ou plusieurs) positions mémoire. Pour envoyer un caractère vers un organe externe, la procédure peut être résumée de la manière suivante, le processeur adresse le PIA auquel l'organe est connecté, ensuite place le caractère sur le bus de donnée. En réalité la procédure est légèrement plus compliquée, car le PIA contient souvent

plusieurs bus (ports) de sortie qui sont programmable en entrée ou en sortie, de ce fait on trouvera aussi quelques registres de contrôle permettant la configuration désirée du PIA avant de procéder à une E/S.
Les standards de communication parallèle les plus répandus sont le standard IEEE-488 qui est le plus souvent destinée à relier des appareils de mesure et le standard CENTRONICS essentiellement destiné à relier des imprimantes.

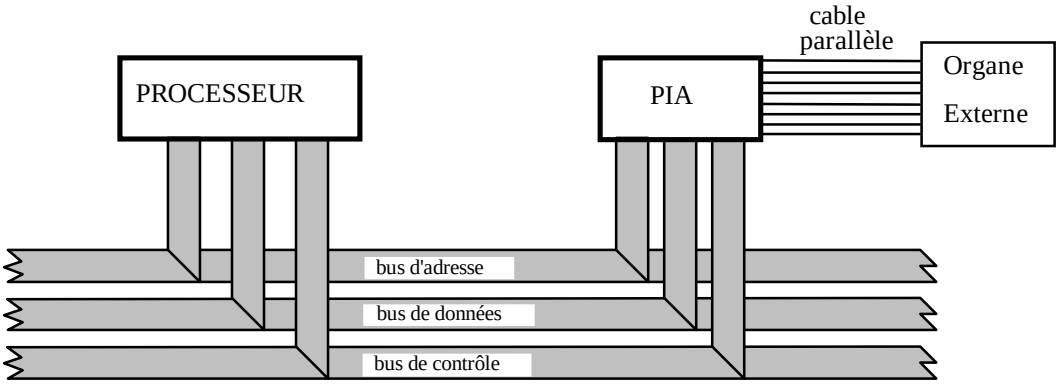


Fig. VIII.5 : Entrée/sortie parallèle

VIII.2.1 Standard CENTRONICS SPP (Standard Parallel port)

Le standard Centronics définit la nature des signaux, des connecteurs à utiliser pour relier une imprimante à un ordinateur, ainsi que le protocole de communication entre l'ordinateur et l'imprimante.

- Les signaux sont des signaux TTL . *("0" ≈ 0V et "1" ≈ 4V)*
- Coté ordinateur, on a un connecteur DB25 femelle
- Coté imprimante on a un connecteur Centronics à 36 broches.

Broche	Signal	Sens	Description
1	nSTROBE	→	L'ordinateur informe l'imprimante qu'un caractère est disponible sur les lignes de données.
2	Data 0	→	Ces signaux représentent les 8 bits de données constituants le caractère transmis. C'est le DATA port
3	Data 1	→	
4	Data 2	→	
5	Data 3	→	
6	Data 4	→	
7	Data 5	→	
8	Data 6	→	
9	Data 7	→	
10	nACK	←	L'imprimante informe l'ordinateur que la donnée a été lue et qu'elle est prête à en recevoir une autre .
11	nBUSY	←	L'imprimante étant plus lente que l'ordinateur, elle l'informe qu'elle est occupée et qu'elle ne peut pas recevoir de donnée.
12	PE	←	L'imprimante informe l'ordinateur qu'elle n'a plus de papier
13	SLCT	←	Imprimante prête = *on line*
14	nAUTOFEED	→	utilisé sur certaines imprimantes pour avancer le papier d'une ligne
15	nERROR	←	Problème de tout genre
16	nINIT	→	Sortie d'initialisation pour réinitialiser l'imprimante
17	nSLCTIN	→	Permet de mettre l'imprimante on-line(0) / off- line(1)
18-25	GND		Masses logiques

tab. VIII-1 : assignation du port parallèle

VIII.2.1.1 Cycle d'envoi d'un octet vers l'imprimante

Le schéma ci-contre illustre la synchronisation entre l'ordinateur et l'imprimante. L'ordinateur place un caractère sur les lignes de données puis envoie une impulsion STROBE. Dès que l'imprimante reçoit le STROBE, elle active la sortie BUSY (affin de pouvoir travailler à son rythme) et va lire le caractère, l'imprime ou le stocke dans sa mémoire. Quand elle a terminé, elle envoie une impulsion sur ACKNLG et remet le BUSY à 0.

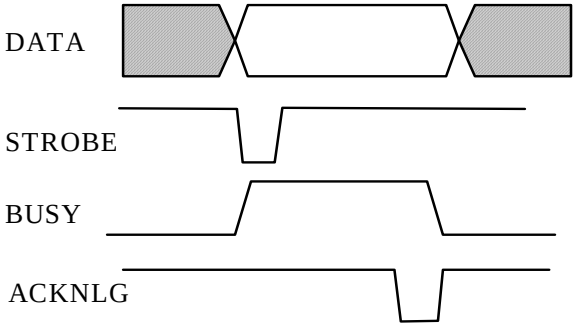
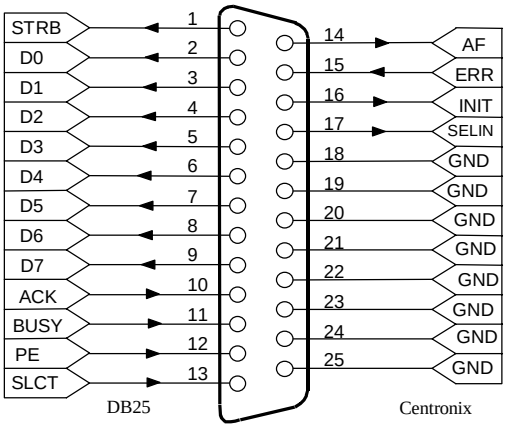
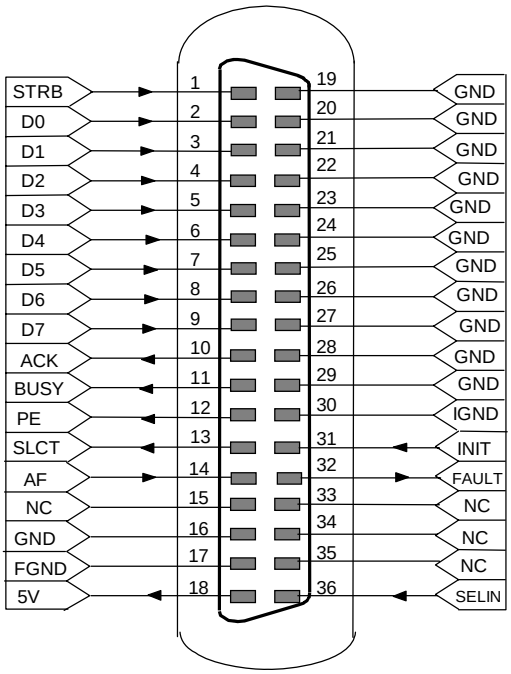


Fig. VIII.6 : Handshake PC / imprimante



DB 25 Femelle
sur ordinateur



Centronics sur imprimante

Fig. VIII.7 : Les connecteurs du standard Centronics

VIII.2.1.2 Les ports (registres) et leurs adresses

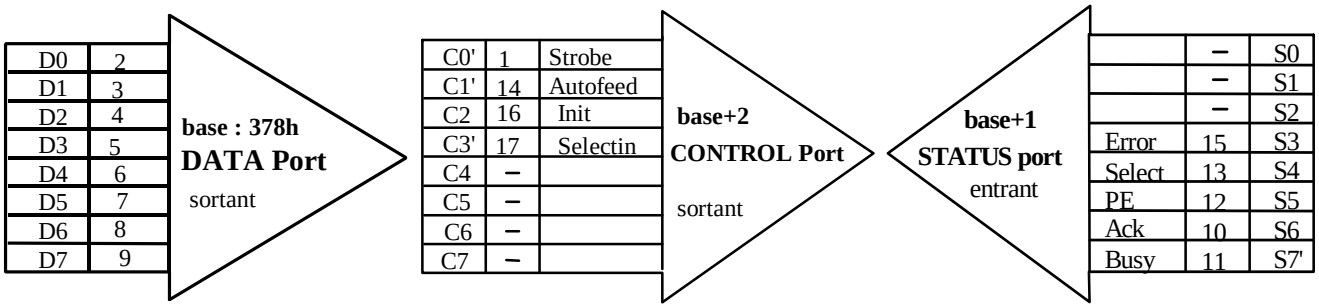


Fig. VIII.8: les registres du port parallèle

Les signaux (ou broches) du port parallèle sont accessibles par programmation grâce à 3 registres appelés communément DATA port, STATUS port et CONTROL port.

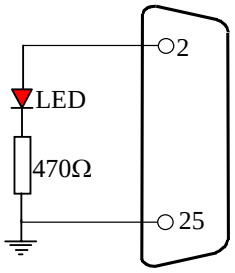
L'adresse du DATA port est par défaut 378h (pour le port LPT1) qu'on l'on désigne par adresse de base. L'adresse du STATUS port est base+1, celle du CONTROL port est base+2.

Attention, les broches 1, 11, 14 et 17 sont inversées matériellement, elles sont repérées par (') sur le dessin. Si on place 5V sur la broche 11 et on lit le STATUS port, le 8ème bit S7' sera

égal à 0 et non à 1, l'utilisateur en déduira que ce 0 correspond à un niveau HAUT sur l'entrée Busy. De la même façon, si on écrit $(0)_{10} = (00000000)_2$ sur le CONTROL port, on pourra mesurer 5V sur les sorties 1, 14 et 17 et 0V sur la sortie 16.

voici un petit programme qui fait clignoter une LED connectée à la sortie 2 (D0)

```
#include <stdio.h>
#include <dos.h>
#include <conio.h>
int main()
{
    while(!kbhit){
      outport(0X378, 0);
      delay(500);
      outport(0X378, 1);
      delay(500);
    }
}
```



VIII.2.2 Le Standard ECP (Extendes capbilities Port)

Le standard ECP permet d'utiliser le port parallèle en multimode. Parmi les modes possibles on trouve : mode SPP *(Standard parallel port*), mode byte dit aussi mode PS2 qui fonctionne comme un port SPP bidirectionnel, mode EPP (*Enhanced parallel mode*) et bien sur le mode ECP proprement dit.

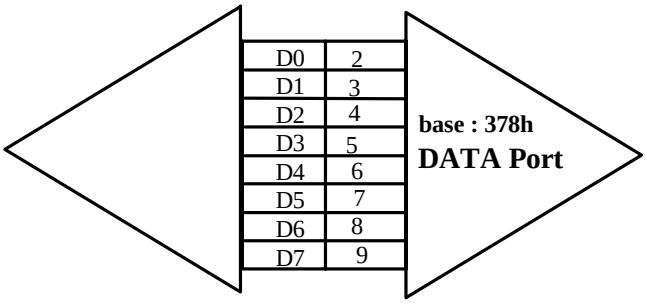
Ce standard utilise le même connecteur DB25 que le SPP mais l'affectation des pins est différente :

Pin	désignation ECP	Sens	Description
1	nHostClk	→	Impulsion négative envoyée par le Host pour indiquer qu'un octet est prêt (en sortie) sur le bus de donnée
2-9	D0 – D7	←→	Bus de données BIDIRECTIONNEL
10	nPerifClk	←	Impulsion négative envoyée par le Périphérique pour indiquer qu'un octet est prêt (en entrée) sur le bus de donnée
11	PerifAck	←	- En mode forward : Accusé de réception du périphérique - En mode reverse : H → On reçoit une donnée L → On reçoit une commande
12	nAckReverse	←	Accusé de réception d'une commande
13	X-Flag	←	Drapeau d'extensibilité
14	HostAck	→	- En mode forward : H → cycle de donnée L → cycle de commande - En mode reverse : Accusé de réception du host
15	nPerifRequest	←	mis à 0 par le périf pour indiquer qu'une donnée est disponible
16	nReverseRequest	→	Le host la met à 0 pour demander le passage en mode reverse
17	1284	→	Le host indique le mode de transfert du standard 1284
18-25	GND	GND	Masse électrique

tab. VIII-2 : affectation des pins pour le mode ECP

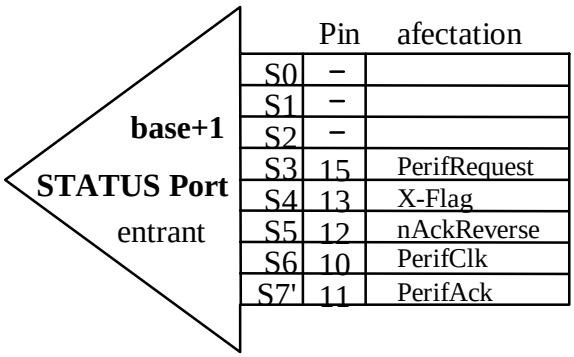
VIII.2.2.1 Le DATA Port

C'est le même port avec le mode SPP sauf qu'il est bidirectionnel. Voir CONTROL port pour voir comment l'utiliser en entrée



VIII.2.2.2 Le STATUS Port

Le STATUS port est un port unidirectionnel. Il est accessible à l'adresse **base+1**. Les bits S0, S1 et S2 ne sont pas utilisés. Les bits S3 à S7 permettent de lire l'état des pins 15, 13, 12, 10 et 11 du connecteur DB25.



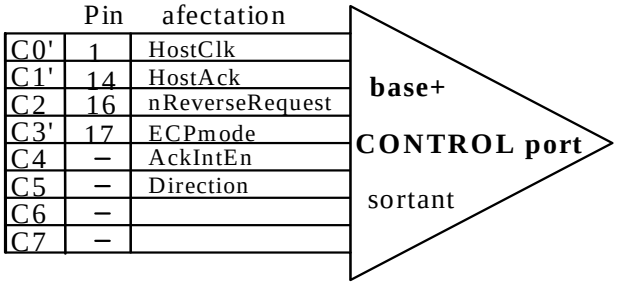
	Pin	afectation
S0	–	
S1	–	
S2	–	
S3	15	PerifRequest
S4	13	X-Flag
S5	12	nAckReverse
S6	10	PerifClk
S7'	11	PerifAck

VIII.2.2.3 Le CONTROL Port

Le CONTROL port est accessible à l'adresse base+2. Les bits C0 à C3 permettent d'accéder aux pins 1, 14, 16 et 17 du connecteur DB25. Les drivers de sorties fonctionnent en collecteur ouvert en mode SPP et en push-pull dans tous les autres modes.

La différence avec le mode SPP c'est qu'on a rajouté deux bits de control C4 et C5. ils ne sont pas reliés à des broches sur le connecteur. Le plus important est le bit C5, c'est le bit qui définit la direction du DATA port. Le sens des communications est définit comme suit :

C5 = 0 : DATA port fonction en sortie
C5 = 1 : DATA port fonction en entrée



	Pin	afectation
C0'	1	HostClk
C1'	14	HostAck
C2	16	nReverseRequest
C3'	17	ECPmode
C4	–	AckIntEn
C5	–	Direction
C6	–	
C7	–	

VIII.2.2.4 Important

- Par défaut, le PP est configuré en SPP, on peut le modifier en accèdent au setup du PC
- Si le DATA port est configuré en SORTIE, sa lecture retourne la valeur qui a été écrite dedans auparavant
- Si le DATA port est configuré en ENTREE, sa lecture retourne la valeur présentes sur les broche 2 à 9 du connecteur
- La lecture du CONTROL port retourne la valeur qui a été écrite dedans auparavant.

VIII.3 ENTREES/SORTIES SERIE

Ici les 8 bits constituant le caractère à transmettre sont envoyés l'un après l'autre. On dit qu'ils sont envoyés en série. Un seul fil suffit pour la communication mais le temps de transmission sera plus long.

Il existe deux types de communication série, la communication synchrone et la communication Asynchrone. C'est cette dernière qui est la plus répandue et on ne parlera que de celle ci dans ce cours.

Le circuit qui permet de faire l'interface entre le processeur et les équipements extérieurs est un ACIA (Asynchronous Communication Interface Adapter*).* Il construit autour d'un registre à décalage qui permet la conversion des informations parallèle qui arrivent du processeur sur le bus de données en informations série qui sont envoyé vers les équipements extérieurs. Là aussi l'ACIA apparaît au processeur comme une position mémoire.

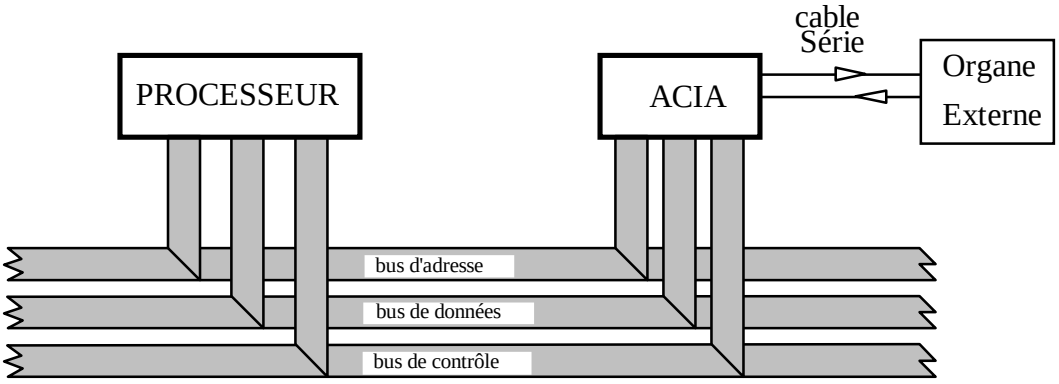


Fig. VIII.9 : Interface Série

En général un ACIA dispose de deux registres à décalage, un pour émettre les informations vers le monde extérieur et l'autre pour lire les informations venant de l'extérieur. Les deux registres sont cadencés par la même horloge qui fixe la vitesse de transmission.

Le registre de transmission est un registre à entrée parallèle sortie série. Il est d'abord chargé en parallèle par les 8 bits constituant le caractère à transmettre, puis il passe en mode décalage et reçoit 8 coups d'horloges et le tours et joué.

Dans le sens entrant, le registre d'entrée est d'abord chargé en série en recevant 8 coups d'horloges, puis son contenu est lu en parallèle par le processeur.

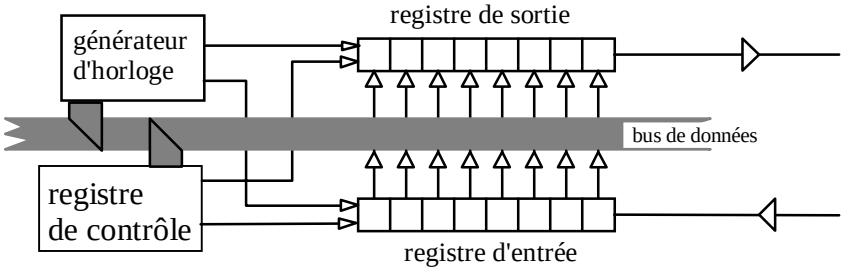


Fig. VIII.10 : Structure Simplifiée d'une ACIA

VIII.3.1 Le standard de communication série RS*232*-C

La liaison RS232C est la plus connue et la plus utilisée pour l'interconnexion des différentes parties d'un ensemble informatique. La norme a été publiée par l'EIA *(*Electronic Industries Association*)* en 1969.

RS : Recommended *Standard,* 232 : Numéro de la norme, C : 3ème version

- La norme fixe les caractéristiques physiques du connecteur. C'est un connecteur 25 broches. Le vrai nom du connecteur est V24 ou Sub-D bien qu'aujourd'hui, tout le monde

l'appelle DB25. Bien après la naissance de la norme RS232, on s'est aperçu que seuls quelques broches du connecteur étaient utilisées, un nouveau connecteur a vu alors le jour. C'est un connecteur 9 broches communément appelé DB9.

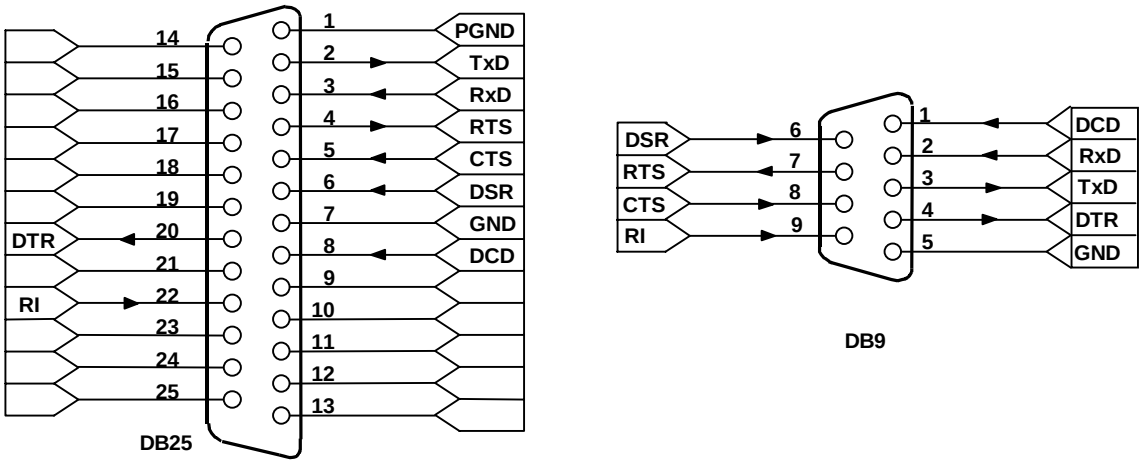


Fig. VIII.11 : Brochage des connecteurs du standard RS232C

- La norme fixe aussi les caractéristiques électriques des signaux :
  Niveau logique "1" $\rightarrow$ -12*V* {-5 , -13}
  Niveau logique "0" $\rightarrow$ +12*V* {+5, +13}

- Elle fixe les fonctions de chaque broche comme suit :

DB25	DB9	Signal	Sens	Signification	
1	-	FGND		Frame Ground	Masse châssis
2	3	TxD	-->	Transmit Data	Emission de donnée
3	2	RxD	<--	Receive Data	Réception de donnée
4	7	RTS	-->	Request To Send	Demande à émettre
5	8	CTS	<--	Clear To Send	Prêt à émettre
6	6	DSR	<--	Data Set Ready	Poste de données prêt
7	5	GND		Signal Ground	Masse signal
8	1	DCD	<--	Data Carrier Detect	Détection de porteuse
20	4	DTR	-->	Data Terminal Ready	Terminal de données prêt
22	9	RI	<--	Ring indicator	Indicateur de sonnerie

tab. VIII-3 : affectation des signaux RS232 sur les connecteurs

Pour comprendre la signification des signaux de la RS232, il faut rappeler qu'à l'origine, la norme a été définie pour connecter deux ordinateurs distants par l'intermédiaire d'une ligne téléphonique. Le réseau téléphonique n'accepte que des signaux analogiques dans la bande de fréquence [300 - 3400] Hz . Il faut donc utiliser deux modems, un à coté de chaque machine pour faire la conversion des informations numériques des ordinateurs en signaux analogique pouvant voyager sur la ligne téléphonique et vice versa.

Les modems de nos jours utilisent des techniques de communication sophistiquées pour permettre des débits importants. Des modems 56 kbit/s sont courants de nos jours (1999). Nous allons illustrer le principe en disant que le modem utilise une modulation de fréquence pour transformer le signal numérique en signal analogique sinusoïdal de fréquence f1 pour le niveau logique "1" et de fréquence f2 pour le niveau logique "0".

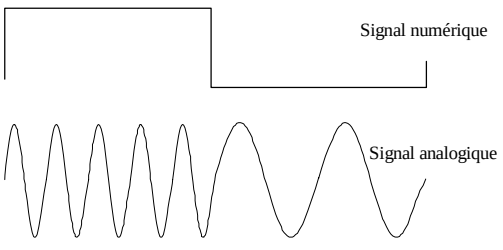


Fig. VIII.12 : Modulation bifréquence

Revenons à la signification des signaux et analysons les différentes étapes des différentes

étapes d'une communication entre des ordinateurs A et B. les ordinateurs sont appelés terminal de données (*DTE : Data Terminal Equipement*) et les modems sont appelés les postes de donnés ou postes de communication (*DSE : Data Set Equipment ou DCE : Device Communication Equipment*).

Supposons que la machine B avec l'aide de son modem compose le numéro de téléphone de la machine A

- Le modem A reçoit un signal sonnerie sur la ligne téléphonique, il en informe la machine A par la broche 22:RI *(Ring indicator)*

- Si la machine A est prête, elle le fait savoir à son modem par la broche 20:DTR *(Data Terminal Ready)* lui demandant ainsi de décrocher.

- Le modem A décroche, Si la liaison téléphonique s'établit correctement, les deux modems en informent leurs machines respectives par la broche 6:DSR *(Data Set Ready)* Poste de données prêt.

- Si la machine A a quelque chose à émettre, elle en informe son modem par la broche 4:RTS *(Request to send)* demande à émettre.

- Le modem A vérifie que la ligne téléphonique est libre en vérifiant l'état de la broche 8:DCD *(Data carrier detect)* Détection de porteuse, pour s'assurer qu'il n'est pas en train de recevoir quelque chose venant de B. il envoie un signal téléphonique signifiant RTS au modem B et répond à sa machine A par la broche 5:CTS *(Clear to send)* Prêt à émettre pour l'informer que la ligne est préparée, que l'autre bout et informé et qu'il était prêt à émettre tout ce que la machine A lui confie vers l'autre modem.

- Remarque : Dès que le modem B reçoit le signal téléphonique signifiant RTS émis par le modem A, il place sa broche 8:DCD à l'état bas. Comme ça, s'il reçoit un RTS de la machine B, il ne lui répond pas un CTS à mois que les deux modems peuvent faire du full-duplex (ligne téléphonique à 4 fils) auquel cas, un modem qui reçoit RTS peut répondre CTS sans se soucier de l'état de DCD.

- La machine A transmet les informations sur sa broche 2:TxD *(Transmit Data)* le modem A les transforment en signal téléphonique et les transmet vers le modem B. Ce dernier les reconvertit en numérique et les communique à la machine B sur la broche 3:RxD *(Receive Data)* .

- Quand la machine A n'a plus rien à émettre, elle remet sa ligne RTS à zéros. Le modem A le fait savoir au modem B qui replace sa ligne DCD au niveau haut pour dire à son ordinateur "*je suis en train de recevoir*". la ligne est ainsi prête à fonctionner dans l'autre sens (ceci bien sur dans le cas du half-duplex)

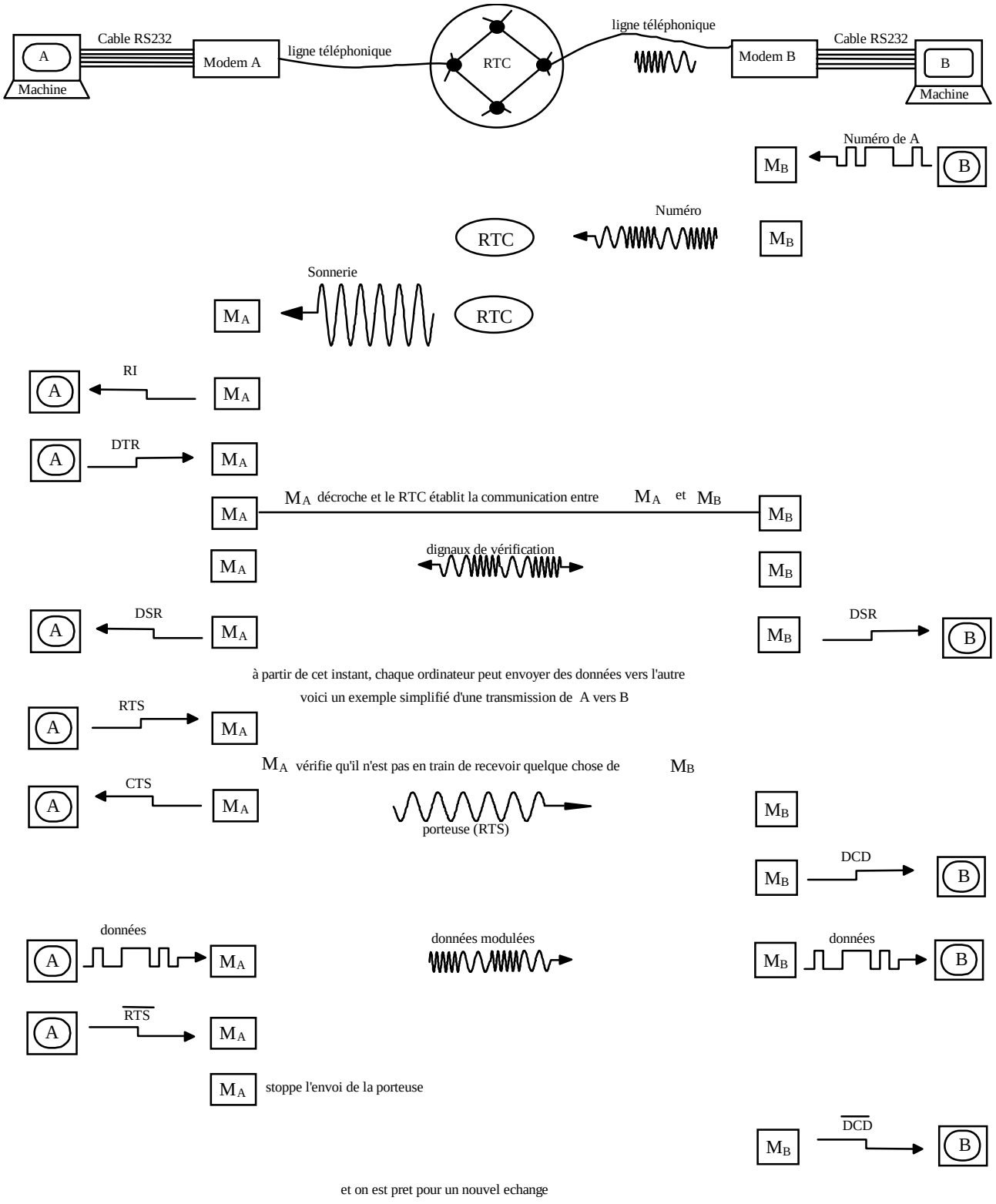


Fig. VIII.13 : Connexion entre deux ordinateurs distants à l'aide d'une ligne RTC

Le câble RS232 reliant chaque machine à son modem est un câble *"pin to pin"*, voir figure ci-dessous

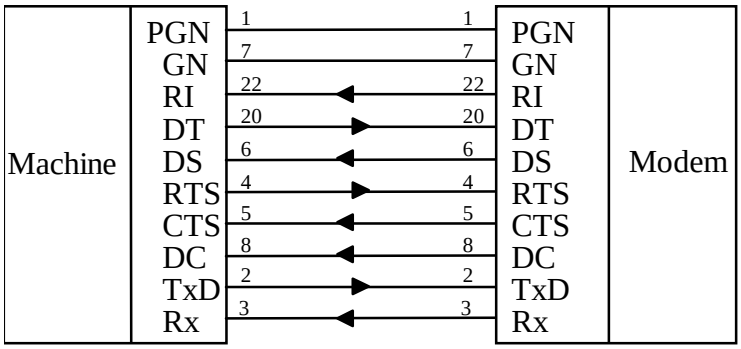


Fig. VIII.14 : Câble PC / modem

VIII.3.1.1 La RS*232C* en liaison directe sans modem

Il arrive que les appareils à relier se trouvent dans le même local ou suffisamment proche pour l'utilisation de modem ne soit plus obligatoire. La liaison se fait alors par connexion directe des deux machines à l'aide d'un câble appelé **Null-modem**.
Le câble est fait d'une façon particulière pour que tout se passe comme si les modems étaient là.

- La première chose à faire et de relier les masses électriques 7 - 7. Les masses Châssis peuvent être reliées pour plus de sécurité 1 - 1. Mais on s'en passe souvent pour économiser un fil.
- La machine A émet les donnés sur TxD et la machine B les reçoit sur RxD. Il faut donc croiser le TxD avec le RxD
- Chaque machine, croyant qu'elle est branchée à un modem, lui envoie un signal DTR pour lui dire qu'elle prête et lui demander de préparer la ligne. Ce signal sera envoyé sur la broche DSR de l'autre machine qui croit que son modem lui répond que la ligne est prête et tout est bon.
- La ligne DCD permettait à un modem d'informer sa machine que la machine éloignée avait émis un RTS. Ceci peut être réalisé ainsi :
- Quand une machine émet un RTS, elle ne peut transmettre les données que lorsque le modem lui répond un CTS pour lui dire qu'il est prêt à émettre ces données. On peut faire de sorte que la machine reçoit son propre RTS en guise de CTS en bouclant la sortie 4 sur l'entrée 5.

On obtient finalement le câble représenté ci-dessous.

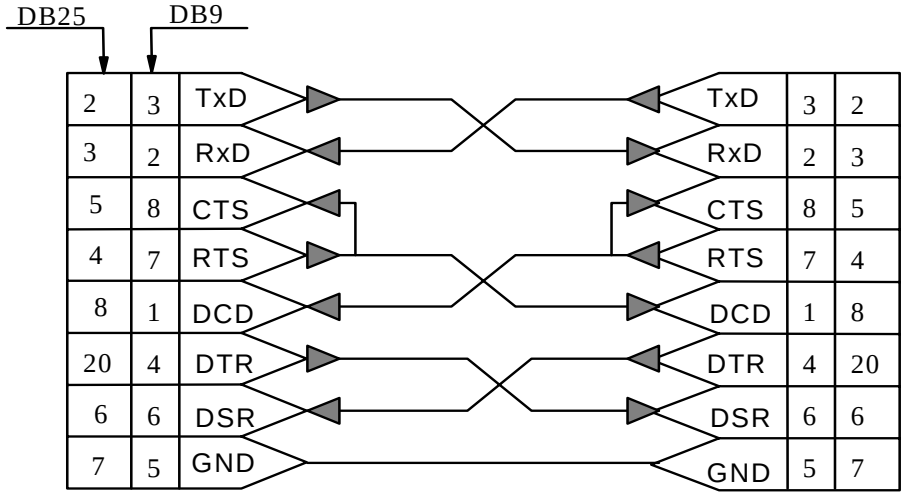


Fig. VIII.15 : câble Null modem

D'autres configurations de câble Null-modem peuvent être proposées. Surtouts si les deux machines utilisent le protocole de communication XON/XOFF. Le câble si dessous fonctionne très bien.

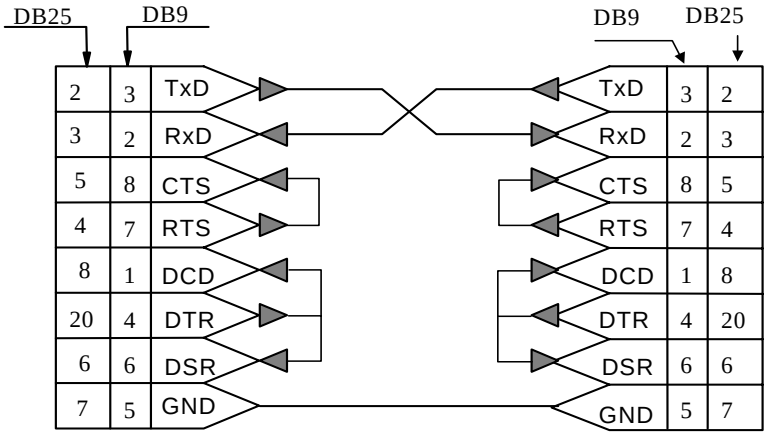


Fig. VIII.16 : câble null-modem 3 fils

VIII.3.1.2 Le moteur de RS*232*-C

L'ACIA qui permet de communiquer avec l'extérieur en standard RS232-C est un UART (Universal Asynchronous Receiver Transmitter).

L'UART ne va pas se contenter de faire la conversion parallèle série, mais va insérer au début de chaque caractère un bit **"0"** supplémentaire appelé **start bit** et à la fin du caractère, il va insérer un bit de parité à la place du $8^{ème}$ bit plus un ou deux bit(s) **"1"** supplémentaire(s) qu'on appelle le(s) **stop bit(s).** Le bit de parité est un bit qui permet de détecter les erreurs à la réception.

A l'émission l'UART génère le bit de parité et à la réception, il contrôle la parité. C'est l'utilisateur qui choisit le type de parité à utiliser, il a le choix entre 4 possibilités :

e → *even* = paire ⇒ Bit de parité = "1" si le nombre de "1" du caractère transmis est pair, il est égal à "0" dans le cas contraire.
o → *odd* = impaire ⇒ Bit de parité = "1" si le nombre de "1" du caractère transmis est impair, il est égal à "0" dans le cas contraire.
m → *mark* ⇒ Bit de parité est forcé à "1".
s → *space* ⇒ Bit de parité est forcé à "0".
n → *none* ⇒ Il n'y a pas de bit de parité, les 8 bits sont utilisés pour la donnée, on peut ainsi transmettre de données codées en ASCII étendu.

Il est évident le contrôle de parité ne permet de détecter qu'une seule erreur, car dans le cas ou il y a deux erreurs, on retombe sur une parité correcte. Quand l'UART détecte une erreur, elle en informe le processeur qui à son tours en informe le logiciel de communication. Souvent les logiciels de communication remplacent le caractère erroné par le caractère "?".

La **vitesse de communication** qui est la cadence avec laquelle les bits sont envoyés en série peut être choisie par l'utilisateur parmi les vitesses normalisées, 300, 600, 1200, 2400, 9600, 19200, etc. Elle est exprimée en **Baud** = bit/seconde.

VIII.3.1.3 Protocole de communication XON/XOFF

L'utilisateur peut choisir entre le protocole de communication *Hardware* RTS/CTS ou un protocole *software* dont le plus utilisé est le protocole XON/XOFF ou DC1/DC3. Avec ce protocole, quand une machine ne peut plus recevoir de données, elle envoie le caractère de contrôle DC3 = XOFF (code ASCII 19) à l'autre machine qui cesse immédiatement d'émettre. Quand la machine est de nouveau prête à recevoir, elle envoie le caractère DC1 = XON (code ASCII 17) à l'autre machine qui reprend l'émission.

VIII.3.1.4 Important

Si on désire réaliser une communication entre deux machines, il est obligatoire que les paramètres de communication soient identiques sur les deux machines.

Exemples :

Vitesse :	9600	4800	19200
Données :	8	7	7
Parité :	n	e	s
stop bit	1	1	2
Protocole	Xon/Xoff	none	RTS/CTS
Duplex	full	full	full

Remarque : Si on choisit de travailler avec 8 bits de données, on est obligé de choisir parité = none.

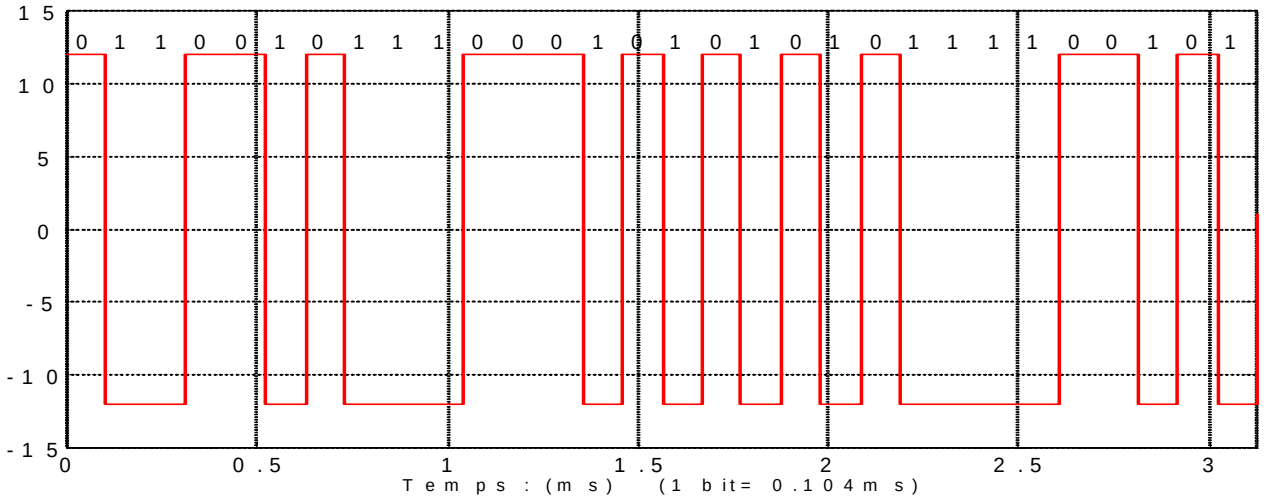


Fig. VIII.17 : exemple de signal RS232 : chaîne STO à 9600 Baud, parité paire, 1 bit de stop

VIII.3.2 Connexion Directe par Câble

Câble série (DB9)

GND(5)	↔	GND(5)
Tx(3)	↔	Rx(2)
Rx(2)	↔	Tx(3)
RTS(7)	↔	CTS(8)
DSR(6)	↔	DTR(4)
CTS(8)	↔	RTS(7)
DTR(4)	↔	DSR(6)

Câble parallèle (DB25)

2	↔	15
3	↔	13
4	↔	12
5	↔	10
6	↔	11
15	↔	2
13	↔	3
12	↔	4
10	↔	5
11	↔	6
25	↔	25

VIII.3.3 Le bus USB (Universal Serial Bus).

Ce standard est provient de l' USBIF (*USB Implementers Forum*) créé en 1995 par Compaq, Digital, IBM, Intel, Microsoft, Nec et Nothern Telecon pour remplacer les ports séries, trop lents et contraignants.

Parmi les point caractéristiques de l'USB, on peut citer :

- Il est Plug and Play, ce qui signifie que lors du branchement d'un équipement, l'OS le détecte automatiquement et charger son pilote sans qu'il soit nécessaire de réinitialiser l'ordinateur. Si le pilote ne peut pas être trouvé, il sera alors demandé à l'utilisateur (CD ou disquette). Si le pilote supporte un chargement à chaud, il peu ainsi être déchargé en cours de session dès que le périphérique est débranché.

- Il permet le hot-pluging c'est à dire la connection à chaud des périphériques comme le clavier, la souris, le joystick, le modem, les hauts parleurs, une caméra,

- Il peut connecter jusqu'à 127 périphériques au total

- Il a un débit global bi-directionnel assez élevé,

  - USB 1.1
    - Low Speed : jusqu'à 1.5 Mb/s brut(donnée+protocole) (blindage câble non obligatoire)
    - Full Speed : Jusqu'à 12 Mb/s brut(donnée+protocole) (câble blindé obligatoire)

  - USB 2.0
    - High Speed : Jusqu'à 480 Mb/s brut(donnée+protocole) (câble blindé obligatoire)

- L'alimentation électrique **:** L'USB prend aussi en charge l'alimentation des périphériques connectés, selon leur consommation. En effet, la norme autorise une consommation maximum de 15 watts par périphérique. Si ce chiffre est largement suffisant pour une paire d'enceinte, une sourie ou un clavier, il n'en va pas forcément de même pour un scanner, un lecteur CD ou une imprimante. C'est pour cette raison que de certains périphériques possèdent leur propre alimentation électrique. Mais, l'USB se charge de les gérer. Vous n'aurez pas besoin de les allumer ou de les éteindre, l'USB activera ces alimentations lors de l'allumage du PC, et les coupera à son extinction.

- Le chaînage USB : La norme USB prévoit à la base de chaîner les périphériques les uns aux autres. Pour que cela soit possible, il est nécessaire que chaque périphérique intègre deux prises, une pour le signal entrant et une seconde pour chaîner le périphérique suivant. Malheureusement, de nombreux périphériques actuellement commercialisés ne comportent qu'une seule prise. (Il est vrai qu'utiliser une souris dotée de deux fils ne doit pas être très pratique) Le second problème est lié à la consommation électrique de chaque périphérique. Il est en effet impossible de connecter 127 périphériques les uns derrière les autres sans ajouter une quelconque alimentation en dehors de celle fournie par la prise sur le PC.
  Les chaînes USB ressemblent beaucoup à une topologie réseau de type bus ou étoile, elles en héritent ainsi des mêmes problèmes. Les solutions à ces problèmes sont forcément les mêmes. Tous ceux qui ont eu l'occasion de travailler sur un réseau au déjà entendu parler des hubs. Il s'agit d'un boîtier dans lequel on insère une prise (le signal entrant), qui est redistribué sur plusieurs prises sortantes. Ainsi, il est possible de connecter plusieurs éléments ne disposant que d'une prise. Les hub peuvent aussi être chaînés entre eux, de manière à former des "grappes". Il existe actuellement deux types de hub, les modèles passifs et les modèles actifs:
  - Le hub passif n'est pas alimenté électriquement, il se contente donc de fournir des prises supplémentaires. S'il résout le problème des périphériques ne disposant que d'une prise, les limitations électriques restent d'actualité.
  - Au contraire, le hub actif est alimenté électriquement. Ainsi, les prises supplémentaires fournies disposent d'un total potentiel électrique, le hub se chargeant de les alimenter au

passage. Avec ce type de hub, il est désormais possible de connecter un grand nombre de périphérique.

La norme USB est prévue pour gérer les défauts de tension. Ainsi, vous serez prévenu quand l'ajout d'un nouveau composant met à mal la tension disponible. Vous ne risquez ainsi pas de provoquer des baisses de tension, qui provoquent toutes sortes de dysfonctionnements désagréables.

- Coût : L'USB est présenté comme un composant bon marché. En effet, il ne nécessite pas de composants matériels coûteux. Sa gestion est implémentée dans le chipset de la machine. D'autres parts, les prises et câbles sont simplifiés à l'extrême, ce qui implique de faible coût de fabrication. Tous les PC intègrent au moins deux prises USB type A en standard, ce qui élimine l'achat de cartes propriétaires, souvent coûteuses.

- La transmission : Le codage de ligne utilisé est le NRZI dans lequel un 0 et représenté par une transition de tension, et un 1 est représenté par l'absence de transition, de sorte qu'une longue séquence de zéros génère un flux d'impulsions régulières.

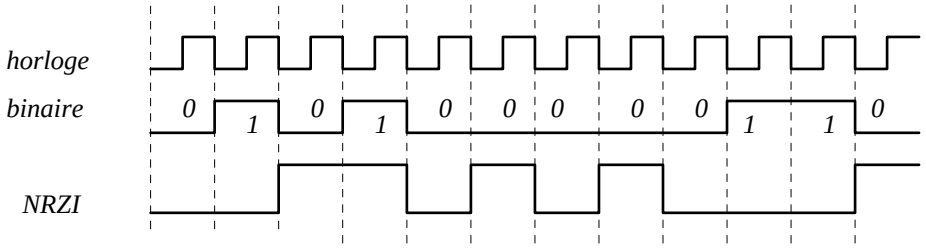


Fig. VIII.18 : code NRZI

Les données échangées entre l'ordinateur et les périphériques sont découpées en trames. Il y a quatre types de trames, Les trames de contrôle, les trames isochrones, les trames "en vrac" et les trames d'interruption.
Les trames de contrôle servent à configurer les périphériques, à leur donner des commandes et les interroger sur leur statut.
Les trames isochrones servent aux périphériques temps réel comme les microphones, les haut-parleurs, et les téléphones. Les paquets de données échangés par ces périphériques doivent arriver dans l'ordre dans lequel il ont été transmises. Sinon les délais engendrés pour attendre tous les paquets et les remettre en ordre peuvent avoir un rôle néfaste su la qualité des communications.
Les trames en vrac servent pour des transferts de masse en direction ou en provenance d'un périphérique sans exigence de temps réel. Ce type de trame est tout à fait adapté au transfert de données informatiques de tout genre.
Finalement, les trames d'interruption sont nécessaires parce que le USB ne supporte pas les interruptions. Par exemple, au lieu que le clavier cause une interruption chaque fois qu'on appuie sur une touche, c'est le système d'exploitation qui le relance à intervalles réguliers pour voir s'il a accumulé des touches dans sa mémoire tampon.

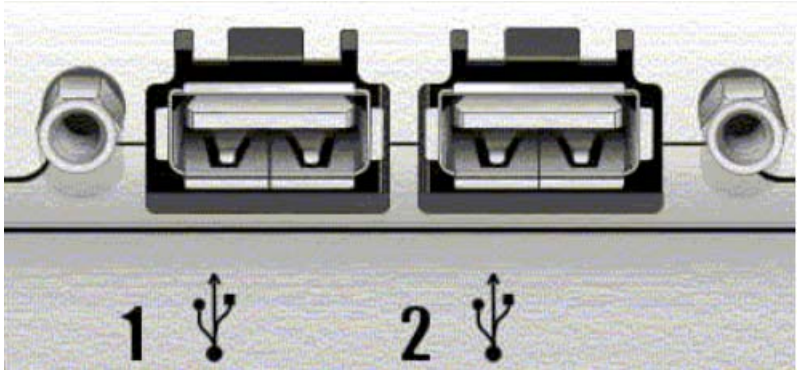
Chaque trame commence par une entête de synchronisation SOF (*Start Of Frame*). Même en absence de données, l'ordinateur et les périphériques échangent des trames ne contenant que le SOF pour maintenir le synchronisme.

Les trames portant des données contiennent un paquet DATA qui permet de transmettre jusqu'à 64 octets d'information. Le paquet DATA consiste en un champ de synchronisation de 8 bits SYN, d'un champ identificateur de 8 bits PID, des données, et d'un CRC (*Cyclic Redundancy Code*) de 16 bits pour détecter les erreurs.

En phase d'échange de données, les trames se terminent par un paquet accusé de réception qui peut être soit : ACK (Ackknowledge) pour indiquer que le paquet de données précédent a été correctement reçu, NAK (Negatife acknowledge) pour indiquer qu'il y a eu erreur de CRC et STALL (attendre, dispositif occupé).

- Le câble

Le câble est formé de 4 fils. Deux servent à l'alimentation en puissance (+5 V), les deux autres servent aux données.



VIII.4 INTERFAÇAGE ANALOGIQUE

Un signal analogique type est le signal vocal récupéré à l'aide d'un microphone. Un signal analogique est un signal continu selon les deux axes : il est existe pour n'importe quel instant et sa valeur peut prendre toute les valeur comprises dans un intervalle [-Vmax, Vmax].

Pour introduire ce signal dans un ordinateur, il faut d'abord le numériser, c'est-à-dire le transformer en une suite de nombres que l'ordinateur que l'ordinateur soit capable de traiter.

L'opération de numérisation se fait en 2 étapes : **L'échantillonnage** et le **codage**.

VIII.4.1 L'échantillonnage

Pendant cette étape, on va prendre des mesures du signal à des intervalles de temps régulier au rythme d'une fréquence d'échantillonnage f_e. La période d'échantillonnage $T_e = \frac{1}{f_e}$ est le temps qui sépare deux échantillons successifs. Si fe = 8000 Hz, cela signifie qu'on prends 8000 échantillons par seconde, deux échantillons successifs sont séparés par *125 µs.*

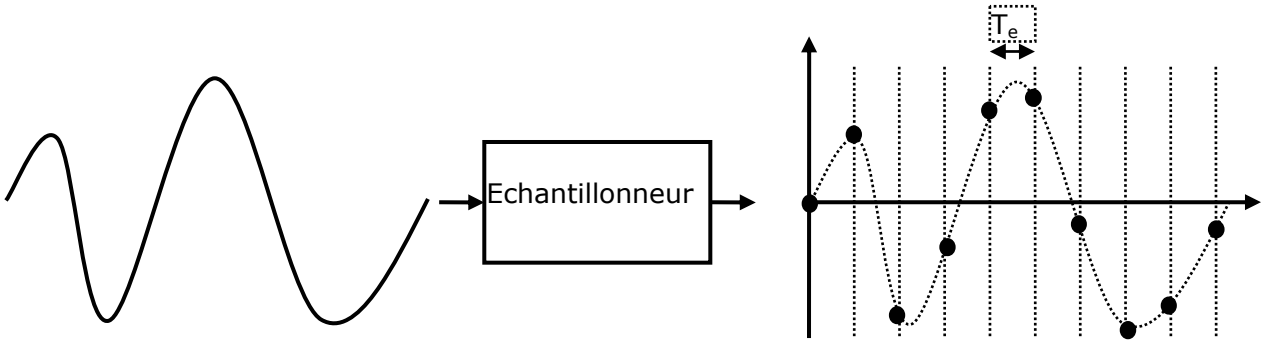


Fig. VIII.19 : Echantillonnage d'un signal

Il parait intuitif que si le signal varie rapidement (fréquence élevée) et si on prend des échantillons trop éloignés, on va avoir perte d'information. D'un autre coté, ça ne sert à rien de prendre trop d'échantillons si le signal peut être représenté avec moins car plus en en prend plus il va falloir de mémoire pour les stocker et de rapidité (puissance) pour les traiter. (*Si on veut suivre l'évolution de la température pendant quelques jours, si on prend une mesure toute les 12h, ce ne sera pas assez "sous-échantillonnage", d'un autre coté ça ne servira strictement à rien de prendre une mesure toute les secondes "sur-échantillonnage")*

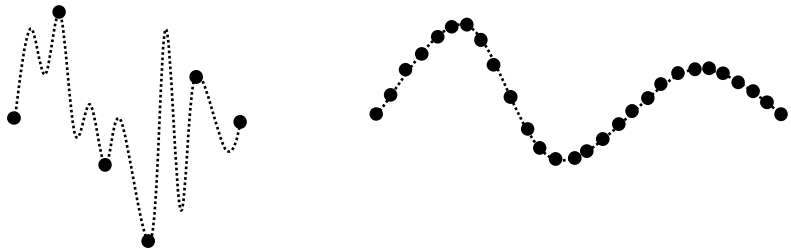


Fig. VIII.20 : choix de la fréquence d'échantillonnage

On voit bien que la fréquence d'échantillonnage doit être choisie en fonction de la fréquence du signal à échantillonner. Shannon a démontré qu'un signal peut parfaitement être reconstitué à partir de ses échantillons à condition d'avoir :

$$f_e \geq 2\, f_{max}$$

VIII.4.2 Codage ou Quantification

Durant cette opération on va affecter une valeur numérique à chaque échantillon. Cette valeur sera représentée sur un nombre fini de bits. Plus le nombre de bits est important, plus la précision est meilleure.

Prenons l'exemple d'un signal évoluant dans l'intervalle [-5V, +5V], si on décide de coder sur $n=3$ bits, on ne peur représenter que $N = 2^3 = 8$ nombres différents : -4,-3, -2, -1, 0, 1, 2, 3. On découpe l'intervalle [-5V, +5V] en 8 intervalles ce qui donne une subdivision de $q = \frac{2V_{max}}{2^n} = 1{,}25V$ qu'on appelle le **pas de quantification**. A toutes les valeurs comprises dans un intervalle on affecte le nombre correspondant (Fig. VIII.21) : à un échantillon de valeur V, on affecte le nombre entier le plus proche de V/q . Par exemple à toute les valeur comprise entre -0,625 et +0,625 on affectera le nombre 0. Le problème est que lors de l'étape inverse qui consiste à reconstituer le signal analogique à partir du signal numérique, au nombre 0 on affectera la valeur 0V, On peut donc commettre une erreur allant jusqu'à 0.625V = q/2.

Pour diminuer l'erreur, il faut augmenter la valeur de n. Mais là encore, ça ne sert à rien de prendre n trop grand, car au bout d'un moment le gain en précision n'est plus perceptible.

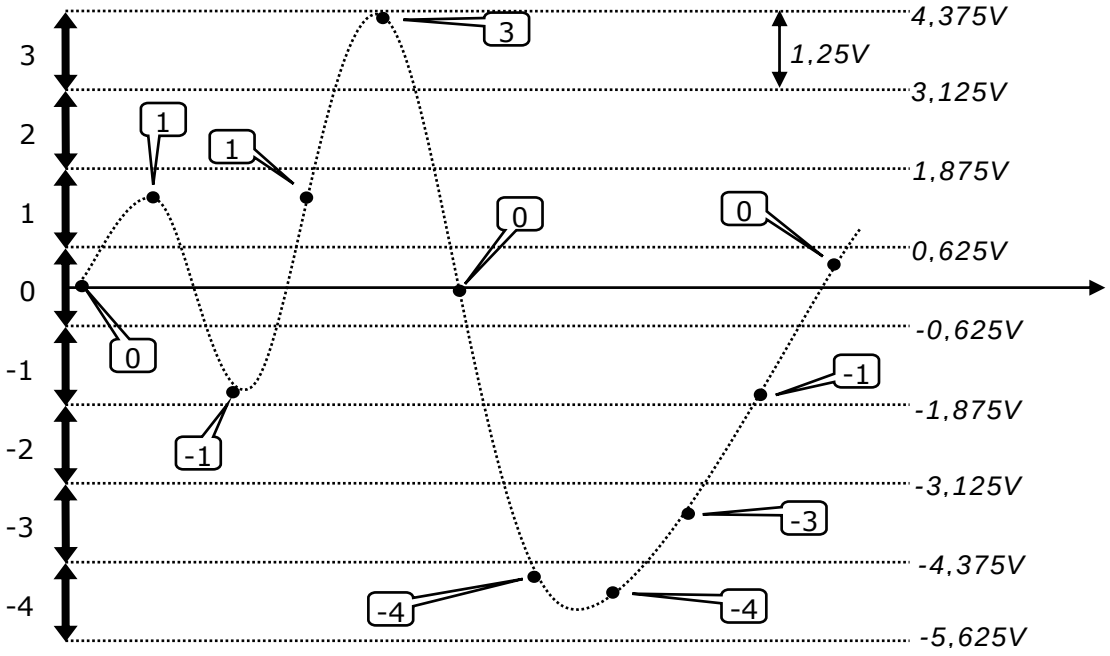


Fig. VIII.21 : Quantification

Dans la pratique, on utilise essentiellement 3 qualités de numérisation :

- **Qualité téléphonique**
  - fe = 8000 Hz
  - n = 8 b/ech
  - → pour un signal d'amplitude $10V_{cc}$, q = 0.039V, E_q = 0.019V
  - → Pour transmettre le signal numérique il faut un débit de 64 kb/s
  - → Un enregistrement de 10s ⇨ fichier de 80 ko

- **Qualité Radio**
  - fe = 22000 Hz
  - n = 16 b/ech
  - → pour un signal d'amplitude $10V_{cc}$, q = 0.15mV, E_q = 75 µV
  - → Pour transmettre le signal numérique il faut un débit de 352 kb/s
  - → Un enregistrement de 10s ⇨ fichier de 440 ko

- **Qualité CD (HI FI)**
  - fe = 44000 Hz
  - n = 16 b/ech
  - → pour un signal d'amplitude $10V_{cc}$, q = 0.15mV, E_q = 75 µV
  - → Pour transmettre le signal numérique il faut un débit de 704 kb/s
  - → Un enregistrement de 10s ⇨ fichier de 880 ko