

الجمهورية الجزائرية الديمقراطية الشعبية

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

1/96

وزارة التربية الوطنية

MINISTRE DE L'EDUCATION NATIONALE

ECOLE NATIONALE POLYTECHNIQUE

المدرسة الوطنية المتعددة التقنيات  
المكتبة — BIBLIOTHEQUE  
Ecole Nationale Polytechnique

DEPARTEMENT ELECTRONIQUE

## PROJET DE FIN D'ETUDES

### SUJET

# ***ETUDE ET MISE EN OEUVRE DE TECHNIQUES DE PROTECTION DE LOGICIELS***

Proposé par:

*Mme Beddek  
Mr Sadoun*

Etudié par:

*Mr N. Bouali  
Mr A. Rehimine*

Dirigé par:

*Mme Beddek  
Mr Sadoun*

PROMOTION

juin 1996

الجمهورية الجزائرية الديمقراطية الشعبية  
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التربية الوطنية  
MINISTRE DE L'EDUCATION NATIONALE

ECOLE NATIONALE POLYTECHNIQUE

المدرسة الوطنية المتعددة التقنيات  
BIBLIOTHEQUE — المكتبة  
Ecole Nationale Polytechnique

DEPARTEMENT ELECTRONIQUE

# PROJET DE FIN D'ETUDES

## SUJET

***ETUDE ET MISE EN OEUVRE DE  
TECHNIQUES DE PROTECTION DE  
LOGICIELS***

Proposé par:

*Mme Beddek  
Mr Sadoun*

Etudié par:

*Mr N.Bouali  
Mr A.Rehimine*

Dirigé par:

*Mme Beddek  
Mr Sadoun*

PROMOTION

juin 1996

## REMERCIEMENT

*A tout ceux qui ont contribué à notre formation de près ou de loin.*

*A nos promoteurs: Mme BEDDEL et Mr SAADOUN pour tout, notamment pour leur coopération directe dans la réalisation de ce travail.*

*Au groupe d'étudiants travaillant au Labo 11, ainsi qu'à tous les étudiants du département d'électronique pour l'ambiance qu'ils ont créée, notamment Ryad et Fayçal.*

*A l'ensemble de personnel de la Bibliothèque et de la Périodique, en particulier Mrs Salah, Karim, Krime, Morsi:*

*Et à tous ceux qui n'ont cessé de nous offrir leur soutien, en particulier: Nassim, Saïd, Farouk, Khelifa; tous de l'E.N.P. sans oublier Nassereddine de l'I.N.T.*

AHMED ET NASSERDDINE

## SOMMAIRE

Résumé

Introduction générale.

Chapitre 1 : Théorie de la protection de l'information -Cryptographie.....4

Résumé.

Introduction.

1. Définition et principe.
  2. Cryptosystèmes modernes.
  3. Applications de la cryptographie.
- Conclusion.

Chapitre 2 : Protection par une ressource logicielle.....11

Résumé.

Introduction.

1. Elements de la programmation système.
  2. Etude des disquettes.
  3. Accès aux Disquettes avec le BIOS.
  4. Formatage Direct.
  5. techniques logicielles de Protection.
  6. Développement d'une protection.
- Conclusion.

Chapitre 3 : Protection par une ressource matérielle ou " Add-On " .....37

Résumé.

Introduction.

1. Intèrfaçage avec un PC.
  2. Etude du port parallèle.
  3. Liaison parallèle entre un PC et un peripherique.
  4. Développement de la Carte.
- Conclusion.

Conclusion Générale.....69

Bibliographie.....70

Annexe

تهدف هذا العمل الى دراسة وإنشاء الطرق الكفيلة بحماية برامج الحاسوب من الاستنساخ والاستعمال غير الشرعي

هذه الطريقة يمكن لمولف البرنامج حماية حقوقه من خلال مراقبة عدد النسخ الاصلية التي يصدرها لبرنامجها.

هناك طريقتين لحماية برامج الحاسوب ، باستعمال النظرية العامة لحماية المعلومة ، تمت دراستها : الحماية باستعمال طرق برمجية أو طرق مادية ، مما سمح من تحقيق الانجازين المتكاملين اللذين تحصلنا عليهما

RESUME

Le but de notre travail consiste en l'étude et la mise en oeuvre des méthodes conduisant à la protection de logiciels contre des utilisations abusives ou non autorisées.

Avec ces méthodes, le développeur du logiciel peut protéger ses droits à travers le contrôle du nombre de copies " Originales " de son logiciel.

Deux aspects de la protection, à travers la cryptographie , ont été abordés : les protections par ressources logicielles ou(et) materielles. Ce qui a conduit à la description des deux réalisations complémentaires que nous avons obtenues.

ABSTRACT

The main of this work is to study and realize some of methods used in program's protection from illegal uses.

Using these methods it will be possible for the developer to protect his rights by controlling the number of copies on his program.

Two aspects of the protection, through the Cryptography , has been approached : the protection by soft resources or(and) hard resources. Which has led to the description of the two complementary realisations which we have obtained.

## INTRODUCTION GENERALE

Le piratage informatique concerne aussi bien la duplication illicite des programmes que la copie de données. Les techniques de protection utilisées sont différentes, les données cryptées peuvent en effet être copiées sans que cela pose de problèmes tant que le pirate ne connaît pas " la clef " de l'algorithme de cryptage.

Du côté protection des programmes, on utilise rarement le cryptage du code source; toujours possible mais lourd à gérer, on utilise plutôt : " Dongle ", " trou laser ", "taux CRC", ...la technologie en ce domaine est évolutive : les éditeurs de logiciels essayent sans cesse de renforcer ces procédures de protection.

La protection des logiciels a véritablement commencé avec le développement de la micro-informatique et la diffusion en masse de logiciels standards. En effet, avec le problème de duplication de programme, le créateur n'a pas les moyens matériels de savoir qui utilise son programme; il fallait donc une protection adaptée au produit, d'où plusieurs générations de protections allant se complexifiant d'avantage.

Toutes les personnes qui ont eu l'occasion de récupérer un fichier effacé par erreur, ont pu vérifier l'existence de fichiers non visibles. Prenons le cas d'un fichier qui a subi la commande "DEL" du DOS. Ce fichier continue d'exister sur le disque avec la totalité de ses données, exception faite d'une petite altération sur la première lettre de son nom. Or, un utilisateur comme PC-Tools permet de visualiser les données du fichier dans différents secteurs du support ( Disquette ou Disque Dur ).

Le même utilitaire permet de voir le contenu du répertoire écrit sur le disque juste après la FAT; dans le répertoire apparaît le nom du fichier " effacé " avec le caractère " ? " au lieu de la première lettre du nom original.

Il suffit alors de remplacer ce caractère pour que le fichier devienne de nouveau visible avec la commande " Dir " du DOS.

Un système très simple de protection de logiciel consiste à " cacher " un fichier nécessaire au bon fonctionnement du programme; toutefois, la banalisation des outils de lecture des supports rend la technique trop facilement contournable. La solution va consister à utiliser des techniques et des astuces plus subtiles, difficiles à localiser, et qui, même si elles sont localisées, demandent beaucoup de travail et / où matériel au pirate.

Notre travail consiste, en l'étude et la mise en oeuvre de techniques de protection de logiciels. Il s'agit de protéger des logiciels contre des utilisations abusives ou non autorisées, et cela par divers aspects de la protection : "SOFT" et(ou) "HARD".

Dans le premier chapitre nous abordons la théorie de la protection de l'information \_Cryptographie, et cela dans le but de voir son intérêt et son utilisation dans la protection de logiciels.

Dans le chapitre 2, nous présentons les techniques de protection par une ressource logicielle, tel que la protection par : formatage de la piste 80 et(ou) la piste formatée hors normes.

Dans le chapitre 3, nous montrons la protection par une ressource matérielle ou "Add\_on", d'où on va assister aux étapes conduisant à la réalisation d'un "dongle" qui va être relié au logiciel qu'on veut protéger.

A la fin, nous terminons notre travail par une conclusion.

## CHAPITRE 1

# THEORIE DE LA PROTECTION DE L'INFORMATION \_ CRYPTOGRAPHIE



## **RESUME**

La Cryptographie est un type de codage, utilisé pour protéger les données, pour assurer la confidentialité de l'information, et d'autre part, garantir son authenticité.

Le but de ce chapitre est de présenter cette succinctement cette discipline pour nous familiariser avec certains termes tels que : clef, Authenticité, ... qui vont être utilisés dans la suite de notre travail.

## **INTRODUCTION**

Les besoins du traitement de l'information mettent en jeu d'énormes masses de données, pour lesquelles le traitement, le stockage, la transmission, doivent être réalisées de manière fiable, sûre et confidentielle. La sûreté et l'inviolabilité de l'information constitueront sans aucun doute l'un des plus grands enjeux des sociétés modernes dans les décennies à venir. Ainsi, des problèmes pratiques de respect de "la propriété d'information" qui ne se posait guère autrefois, qu'aux militaires et diplomates.

La recherche pratique de solutions efficaces à la protection de l'information se développe sur deux grands axes :

i - Les cadres législatifs.

ii - La Cryptographie ; utilisée dès l'antiquité pour protéger des messages à caractère militaire. De nos jours, elle est l'une des méthodes les mieux reconnues pour protéger efficacement l'information traitée dans les réseaux d'ordinateurs complexes. De la simple carte bancaire aux très impressionnants dispositifs de transmission, la cryptographie est partout présente dans les sociétés industrielles.

### **1 - DEFINITION ET PRINCIPE [ 1 ] :**

Cryptographie : C'est un mot grec constitué de deux parties : "Crypte" qui veut dire cacher, et "graphe" qui veut dire écrire. Certains la définient par : science de la protection de l'information.

Le problème traité par la Cryptographie est le suivant : lorsque deux individus souhaitent partager de l'information, ils sont amenés à s'échanger, le plus souvent par un canal de transmission ( téléphone, télex, réseau d'ordinateurs ... ), des messages de longueur quelconque mais finie.

Si l'émetteur de l'information souhaite que celle-ci ne soit accessible en clair qu'à son destinataire, alors il utilisera un procédé de cryptographie. Si par définition, il est capable de mettre en oeuvre une transformation mathématique, agissant sur le message émis, rendant ce dernier inintelligible pour toute tierce partie illégalement branchée sur le canal de transmission. Cela suppose évidemment qu'à l'arrivée, le destinataire du message dispose d'une autre transformation mathématique, éventuellement différente de la première, lui permettra de reconstituer le texte clair à partir du texte brouillé (chiffré), (Fig.1.1)

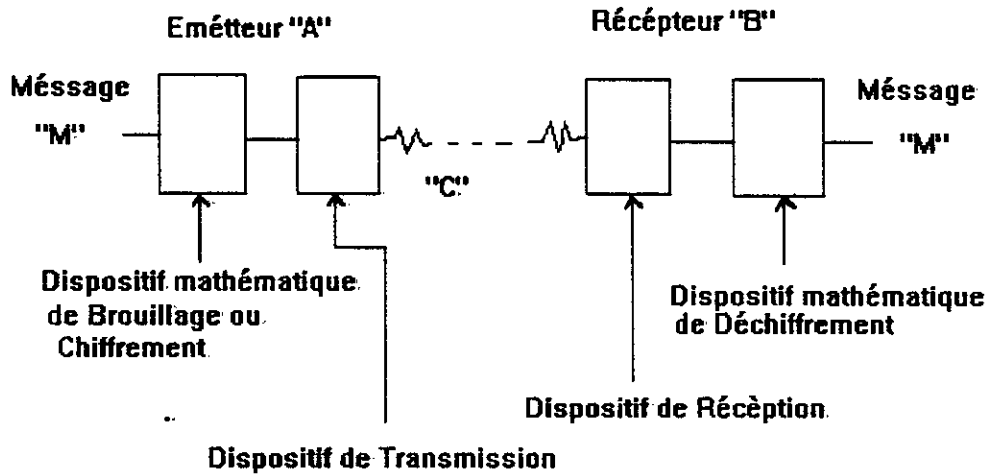


Fig.1.1- Schéma général de la communication chiffrée.

Dans la suite :

- On appelle " message " une suite de  $n$  symboles, pris dans un alphabet  $Z$  de  $q$  symboles. Dans la pratique, le plus souvent,  $Z$  est : soit l'ensemble  $\{0,1\}$ , soit l'ensemble  $Z_m = \{0, 1, \dots, m-1\}$  des entiers modulo  $m$ .

- On appelle " N-gramme ", un message  $M$  de longueur  $N$ , soit une suite  $M = (Z_1, Z_2, \dots, Z_N)$  où chaque  $Z_i$  appartient à  $Z$ .

L'ensemble des bijections de  $Z_m$  dans  $Z_m$ , ou ensemble des permutations d'ordre  $m$ , est noté  $S_m$ . Le Cardinal de cet ensemble vaut exactement  $m!$ . Muni de la composition des applications, notée "  $\circ$  ",  $S_m$  est un groupe fini.

Il faut noter également que l'on peut toujours supposer que l'alphabet de référence est l'un des ensembles  $Z_m$ .

- De manière précise, nous définirons " un Cryptosystème " comme étant un ensemble:

$$U = (M, C, K, \{E_k\}_{k \in K}, \{D_k\}_{k \in K})$$

où  $M$  : représente l'ensemble des messages en clairs.

$C$  : représente l'ensemble des messages chiffrés.

$K$  : représente un espace de paramètres appelés clefs cryptographiques.

$E_k$  : représente pour tout  $k \in K$ , une transformation de chiffrement

$$M \xrightarrow{E_k} C$$

$D_k$  : représente pour tout  $k \in K$ , une transformation de déchiffrement

$$C \xrightarrow{D_k} M$$

telles que, pour tout message  $M$  on ait  $D_k \circ E_k(M) = M$

La notion de "clef" paramétrée ou clef cryptographique est fondamentale : elle sous-entend tous les mécanismes de paramétrisation des algorithmes de Cryptage ainsi que toutes les techniques de protocoles d'échanges cryptographiques

- On appelle " cryptanalyse " l'opération consistant, sur la base du message reçu.

$$C = T_k(M)$$

où  $T_k$  est une transformation mathématique, connue ou inconnue, dépendant du paramètre  $k$ , lui toujours inconnu, à reconstituer les messages originaux  $M$ .

- On appelle " cryptanalyste " l'individu s'adonnant de manière scientifique ( et / ou perverse ) à l'exercice de la cryptanalyse. En pratique, la cryptanalyse revient toujours à la recherche de la clé  $k$  de paramétrage. (Fig.1.2)

Construire un système de cryptographie inviolable passe toujours par la vérification qu'il résiste aux assauts de la cryptanalyse. Ainsi, un bon cryptographe est, par nécessité un bon cryptanalyste ( et réciproquement ! ).

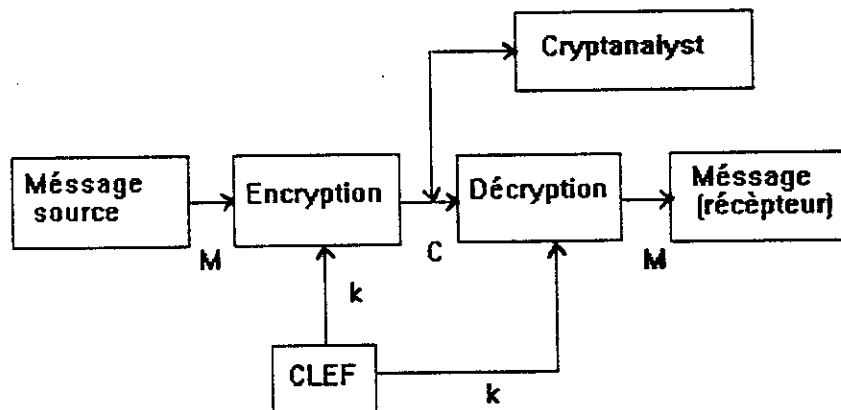


Fig .1.2- Schéma d'un système cryptographique conventionnel.

## 2 - CRYPTOSYSTEMES MODERNES [ 1 ]

Les principaux types de cryptosystèmes utilisés aujourd'hui se répartissent en deux grandes catégories : les cryptosystèmes par flots et les cryptosystèmes par blocs.

- Dans les Cryptosystèmes par flots : l'encryption des messages se fait caractère par caractère, au moyen de substitutions générées aléatoirement selon une loi de répartition équiprobable sur l'alphabet  $Z_m$ . (Fig.1.3)

En d'autres termes, la lettre  $X_n$  du message à transmettre est encryptée en  $Y_n = X_n \oplus K_n$ , où  $K_n$  est une variable à valeurs aléatoires dans l'alphabet de référence  $Z_m$ . La loi de  $K_n$  est telle que toutes les valeurs  $P(K_n = K)$  sont équiprobables ( $0 \leq K < m$ ) et que les  $\{ K_i \}$

$0 \leq i < n$  sont indépendantes, cela quelle que soit la longueur  $n$  du message.

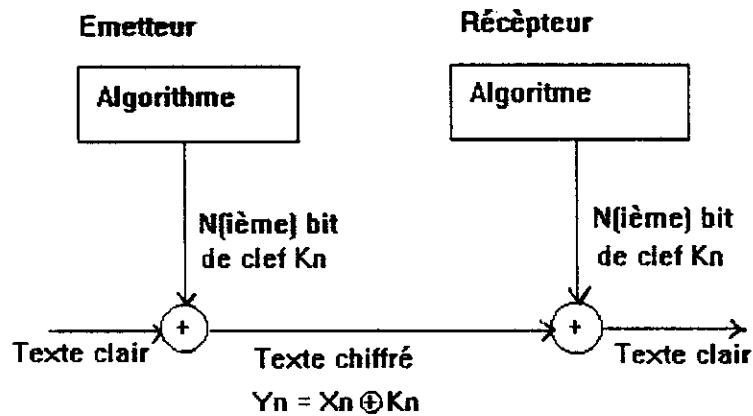


Fig .1.3- Système de chiffrement par Flot.

- Dans les cryptosystèmes par blocs : le texte clair est fractionné en blocs de même longueur à l'aide d'une clef unique  $K$ .

Emetteur et récepteur disposent de deux algorithmes d'encryption / décryption parfaitement réciproques, paramétrés par  $K$ . (Fig.1.4)

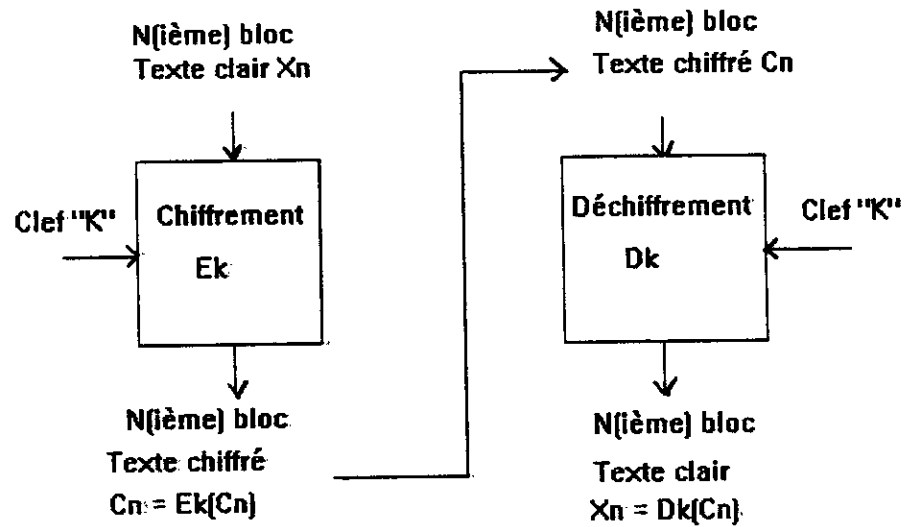


Fig. 1.4- Système de chiffrement par blocs.

### 3 - APPLICATIONS DE LA CRYPTOGRAPHIE [2], [1]

#### 3.1 - AUTHENTIFICATION

L'authentification, en théorie de l'information, est un processus de vérification qui peut revêtir trois principaux aspects :

- i) identifier quelqu'un.
- ii) identifier une information possédée ou émise par quelqu'un.
- iii) identifier une information, ou plus généralement un attribut relatif à un individu.

Dans ce cas un message spécifique  $M_s$  ( par exemple un texte spécifique agrégé par avance ) est ajouté au message à identifier  $M$ ; l'émetteur envoie alors le cryptogramme  $E_k (M, M_s)$ . Si le receveur déchiffre  $E_k (M, M_s)$  reconnaît  $M_s$  alors le message est considéré comme authentique.

### **3.2 - SIGNATURES DIGITALES**

On appelle méthode de signature digitale tout procédé d'authentification de documents générés et gérés par voie purement électronique et / ou informatique. L'utilisation de signatures digitales correspond en général au triple souci : d'amélioration du temps de réponse; d'élimination de documents de type papier ; d'efficacité ( technique et économique ) .

### **CONCLUSION**

Cet exposé n'a donné qu'un petit aperçu de la cryptographie, sans se soucier des recherches théoriques et des développements consacrés aux méthodes et algorithmes de cette discipline.

Les systèmes cryptographiques offrent un moyen efficace de protection de l'information transmise sur des canaux de communication ou stockée dans des mémoires.

Dans ce dernier cas, comme on va voir dans les chapitres suivants, que les systèmes cryptographiques sont très utilisés par les développeurs pour renforcer la protection de logiciel.

## **CHAPITRE 2**

# **PROTECTION PAR UNE RESSOURCE LOGICIELLE**



## **RESUME**

Dans cette section on étudiera la méthode de protection de logiciels par des procédés uniquement de programmation (sans aucun " add on " matériel).

Pour en arriver, on doit passer par la présentation de la programmation système qui nous permet un accès libre aux périphériques du micro-ordinateur, et l'étude des supports d'information pour maîtriser l'organisation des informations sur ces derniers. Et, on donnera un exemple de programme d'accès aux disquettes en utilisant les fonctions BIOS.

## **INTRODUCTION**

Chaque support d'information ( disquette, disque dur, CD-ROM,.... ) est formaté selon des normes spécifiques; prenant l'exemple d'une disquette (1.44 Mo ) formatée par DOS; elle contient 80 pistes , chaque piste contient 18 secteurs, et chaque secteur a une taille de 512 Octets .

En utilisant les fonctions BIOS, on peut donner aux pistes un nombre de secteurs et des tailles de secteurs différentes (c'est à dire : créer de nouvelles normes, non reconnues par DOS).

Si une piste est formatée différemment ( le nombre et(ou) la taille de secteurs ne sont pas standards ). Alors le DOS ne reconnait pas cette piste et donc il ne peut : ni lire, ni écrire dessus. En conséquence il va la déclarer comme une piste defectueuse, ce qui n'est pas le cas en réalité. C'est sur cette idée que se base cette méthode de protection, il est clair que son implémentation nécessite une connaissance préalable de la programmation système et spécialement la manipulation des interruptions BIOS.

## **1 - ELEMENTS DE LA PROGRAMMATION SYSTEME [ 8 ]**

### **1.1 - LE MODELE A TROIS COUCHES**

L'une des tâches fondamentales de la programmation Système consiste à accéder au matériel constituant le P.C. Cet accès ne se fait pas de manière directe nécessairement ; la manière usuelle consiste à utiliser un intermédiaire qui propose des services spécialisés. Cet intermédiaire peut être le BIOS ou DOS qui sont des interfaces logicielles créées pour gérer le matériel.(Fig.2.1)

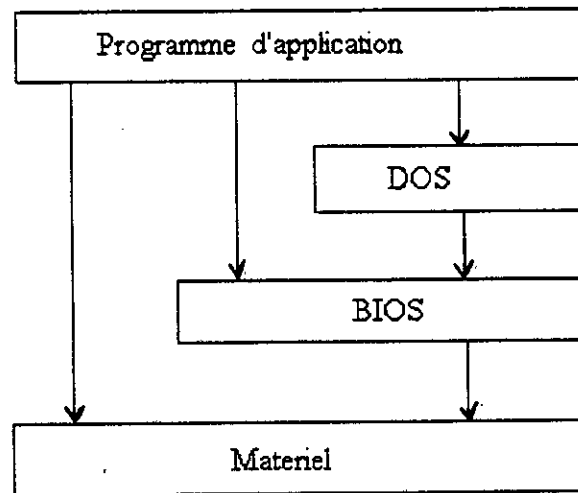


Fig.2.1 - Le mode à trois couches.

LE BIOS ( Basic Input/Output System ) : c'est le logiciel de plus bas niveau, logé en mémoire morte ( ROM ) qui sert d'interface entre le matériel et le DOS.

Un programme d'application pourrait court-circuiter le BIOS en accédant directement au matériel. Mais, dans la plupart des cas il gagne à se servir des fonctions BIOS dont les services sont standardisés et identiques dans tous les P.C.

L'INTERFACE DOS ( Disk Operating System ) : C'est le système d'exploitation le plus courant sur P.C. Il est chargé en mémoire vive ( RAM ) et sert d'interface entre les logiciels d'application et le BIOS.

Le DOS propose également des fonctions d'accès au matériel, elles présentent cependant un autre caractère que les fonctions du BIOS. Car elles considèrent le matériel non plus sous son angle physique mais comme un dispositif logique.

Dans le cas des supports d'information, le phénomène est particulièrement évident : alors que le BIOS raisonne en pistes et en secteurs, le DOS se place au niveau des fichiers et des sous-répertoires ; on peut ainsi s'adresser au DOS en lui demandant d'ouvrir tel fichier et d'en extraire les 1000 caractères premiers sans se préoccuper de l'emplacement du fichier sur le disque, il n'est même pas utile de connaître le type de la mémoire de masse ( disquette, disque, serveur de réseau ). Il suffit de désigner le périphérique par une lettre ( comme A : , B : ou C : ) pour que le DOS sache à quoi il aura à faire.

Il est vrai que l'accès ne sera pas uniquement régi par le DOS : le BIOS sera impliqué dans le mécanisme, il se pourrait dans certaines situations que le DOS accède directement au matériel mais peut importe pour le programmeur.

Quelle est la meilleure voie pour accéder au matériel ; faut-il user de la programmation directe, faire appel aux fonctions BIOS ou celles du DOS ?

### **LE CHOIX DU SERVICE**

Lorsque les fonctions BIOS et DOS se disputent la faveur du programmeur, la décision se fera en fonction de l'application concernée. Il faut rendre compte qu'on n'a pas toujours le choix entre programmation directe du matériel, et les fonctions BIOS et DOS; comme Il existe toute une série de tâches qui ne sont couvertes ni par DOS ni par BIOS.

#### Exemple:

1- Les Fonctions DOS : on les utilise pour travailler avec des fichiers.

- Faire un transfert ou une comparaison de fichiers.
- Créer, copier, renommer,... un fichier.
- ...etc.

2- Les fonctions BIOS : on les utilise pour formater une disquette, pour écrire un caractère sur l'écran,...etc.(voir Fig.2.4)

3- Programmation directe du matériel : elle est utilisée pour :

- Le mode graphique de la carte vidéo.
- La communication avec les interfaces du PC, tel que le port parallèle du PC.

(voir CHAPITRE 3)

- ...etc.

Dans certains cas, la vitesse d'exécution constitue un inconvénient majeur des fonctions du BIOS ou du DOS. Car plus il y a de couches à traverser pour convertir l'intension d'un programme en accès au matériel, plus l'exécution se trouvera ralentie. Le ralentissement est dû à la gestion des différentes couches ; avant de transmettre le contrôle à une autre couche, il faut convertir de nombreux paramètres, charger des informations dans des tables internes,

constituer des mémoires tompons, ... le temps perdu de cette façon ( appelé " Overhead " ) plus il est important, plus il dégrade "l'humeur" du programmeur.

## 1.2 - LES INTERRUPTIONS

Une interruption peut être de deux types : matériel ou logiciel. Les interruptions matérielles sont des signaux qu'adresse le matériel au logiciel pour une demande de requête. Les interruptions logicielles servent de passerelles entre les applications.

Mais, quel que soit l'origine, toutes les interruptions fonctionnent de la même façon ; elles sont repérées par un numéro d'ordre et acceptent en entrées les paramètres qui leurs sont passés dans les registres internes du P.C. Lorsqu'un programme d'application exécute une interruption, il s'arrête de travailler pendant un temps indéterminé, et le niveau inférieur prend le relais.

En effet à chaque fonction d'une interruption correspond un sous-programme qui se trouve à une " couche inférieure ", à celle qui a appelé cette fonction. Ainsi, le sous-programme de la fonction d'affichage d'une chaîne de caractères demandant l'effacement de l'écran est la suite de trois interruptions BIOS qu'exécute le noyau du DOS et à chacune de ces interruptions BIOS correspond un sous-programme tel que celui qui positionne le curseur. Lorsque le matériel a terminé il rend la main et le contrôle revient au BIOS, lorsque le BIOS a terminé , il retourne le contrôle au DOS, qui finit son travail et autorise de nouveau l'utilisateur à continuer : il s'agit donc d'un mécanisme en cascade .

Naturellement, chaque fonction d'une interruption doit procéder à ses propres tests et peut donc échouer, elle se termine alors en renvoyant un code d'erreur dans un registre spécifique, qui peut ( et doit ) être testé par l'appelant.(Fig.2.8)

Il existe effectivement 256 interruptions pour un P.C., c'est pourquoi la table des vecteurs d'interruption possède 256 entrées ou éléments; (la Fig.2.2 représente la table des quelques premiers vecteurs d'interruptions).Les interruptions matérielles et logicielles sont mises sur le même plan, seul le type d'appel les distingue. Dans une interruption logicielle, c'est le programme qui fixe le numéro d'interruption tandis que dans une interruption matérielle c'est le périphérique qui s'en charge. Chaque élément de la table des vecteurs d'interruption occupe deux mots successifs car il s'agit d'un pointeur FAR qui donne le segment et l'offset du gestionnaire associé.

\* FAR : l'adresse est exprimée sous forme de [ Segment : Offset ].

L'adresse où réside le gestionnaire associé à un vecteur se calcule en multipliant le numero de l'interruption par quatre.

N°	ADRESSE	FONCTION
00	000 -003	CPU : Division par zéro
01	004 -007	CPU : Pas à pas
02	008 -00B	CPU : NMI (Défaut dans la RAM )
03	00C -00F	CPU : Point d'arrêt atteint
04	010-013	CPU : Débordement numérique
05	014 -017	Copie d'écran
06	018 -01B	Instruction inconnue
07	01D -01F	Réservé
08	020 -023	IRQ0 : Timer (Appet 18,2 fois/sec. ).
09	024 -027	IRQ1 : clavier
0A	028 -02B	IRQ2 : deuxième 8259
0B	02C -02F	IRQ3 : Interface série 2
0C	030 -033	IRQ4 : Interface série 1
0D	034 -037	IRQ5 : Disque Dur
0E	038 -03B	IRQ6 : Disquette
0F	03C -03F	IRQ7 : Imprimante
10	040 -043	BIOS : Fonction VIDEO
11	044 -047	BIOS : Détermination Configuration
12	048 -04B	BIOS : Détermination taille RAM
13	04C -04F	BIOS : Fonction disquette/disque Dur
14	050 -053	BIOS : Accès à interface Série
15	054 -057	BIOS : Fonction cassette ou étendues
16	058 -05B	BIOS : Interrogation du clavier
17	05C -05F	BIOS : Accès à imprimante parallèle
18	060 -063	Appel du BASIC en ROM.
...	....	....

Fig.2.2- Table des quelques premiers vecteurs d'interruption.

### **1.3 - CONCLUSION**

En étudiant les éléments de la programmation système et pour réaliser une protection par une ressource logicielle, on a utilisé directement les fonctions BIOS; et cela pour créer et gérer de nouvelles normes qui ne sont pas connues par DOS.

## **2 - ETUDE DES DISQUETTES ( SUPPORTS D'INFORMATION ) [ 6 ], [ 8 ]**

Il existe plusieurs types de supports d'informations en micro-informatique :

- Disquette
- Disque dur
- Casette
- CD ROM ...

### **2.1 - PRESENTATION PHYSIQUE DE DISQUETTES**

Une disquette se présente sous forme d'un disque plastique souple sur lequel a été posée une couche de substance magnétique ; possède une épaisseur de l'ordre de 100  $\mu\text{m}$ , alors que la quantité d'enduit magnétique varie entre 1 et 9  $\mu\text{m}$ .

La substance magnétique déposée est à base d'oxyde de fer (  $\text{Fe}_2\text{O}$  ) lié par une résine. Chaque particule d'oxyde forme un aimant élémentaire qui peut être orienté par la tête d'écriture du lecteur.

Pour des raisons d'organisation des données et de faisabilité des échanges d'informations entre le système et la disquette, les disquettes sont divisées en pistes ; sortes de cercles concentriques réparties à intervalles réguliers sur leurs surfaces magnétiques.(Fig.2.3)

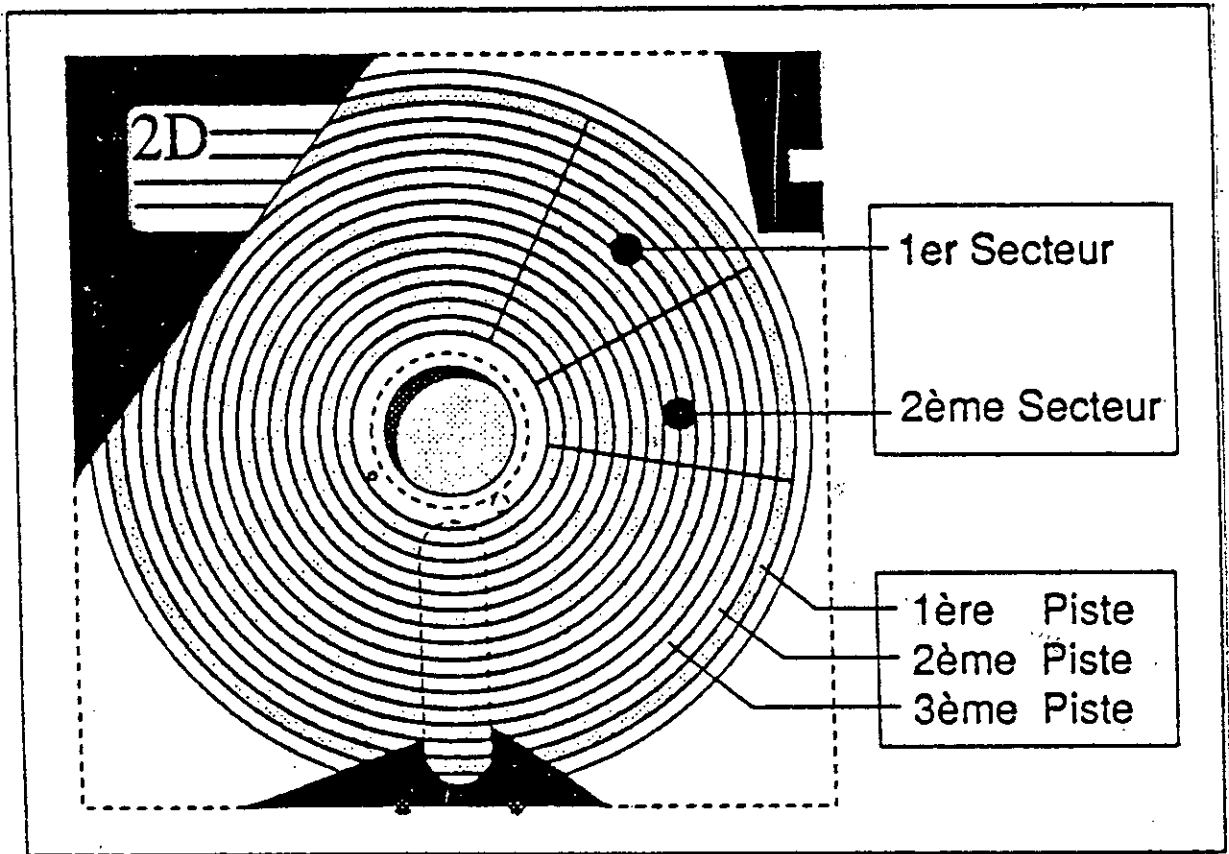


Fig .2.3 - Structure d'une disquette.

Ces pistes sont numérotées de 0 à N-1, N étant le nombre total de pistes et pouvant varier selon le format de la disquette. La piste la plus extérieure porte le numéro " 0 " la suivante le numéro 1 et ainsi de suite jusqu'à la piste située la plus à l'intérieur.

Chaque piste est divisée en nombre constant de secteurs de taille égale, le nombre de ces secteurs varie en fonction du format de la disquette et du lecteur. Les secteurs sont numérotés à partir de 1 jusqu'à N , N représentant le nombre de secteurs par piste.

En fonction du nombre de pistes et de secteurs qu'elle contient, la capacité d'une disquette s'évalue avec la formule suivante :

$$(\text{Nbre de piste}) * (\text{Nbre de secteurs/piste}) * (512 \text{ Octets/secteur}).$$

La valeur ainsi obtenue est évalué en Octets. Elle ne traduit que la capacité d'une face de la disquette et doit donc être multipliée par " 2 ".



## 2.2 - ORGANISATION LOGIQUE DES INFORMATIONS

L'intérêt de connaître l'organisation des supports permet de mieux comprendre la protection des logiciels.

NUMEROTATION DES SECTEURS : la numérotation des secteurs s'effectue dans un ordre séquentiel à partir du secteur 1 : ( la numérotation des secteurs commence à 1 ) ; de la piste 0 : ( la numérotation des pistes commence à 0 ) ; de la face 0 jusqu'au 18<sup>ème</sup> secteur de la piste 79 face 1 ; ceci bien sûr dans le cas d'une disquette 3 pouces (1440 KOctets), mais le principe est le même quel que soit la disquette ou le disque dur ; se sont uniquement les valeurs qui changent:

Afin de gérer la disquette, les premiers secteurs ( dont le nombre dépend du format de la disquette ), sont " réservés " d'office; c'est à dire que l'utilisateur ne pourra pas écrire des données sur ces secteurs ; ils sont réservés dans l'ordre suivant :

- 1 secteur BOOT ( programme de chargement : piste 0, face 0, secteur 1 ).
- secteurs FAT ( table d'allocation des fichiers ).
- secteurs FAT ( 2<sup>ème</sup> copie de la FAT : facultative selon le type de la disquette )
- secteurs répertoire principal

Les secteurs suivants sont libres à l'écriture des fichiers, si la disquette est formatée avec l'option " /S ", on trouvera donc à partir de ces secteurs les fichiers cachés : IO.SYS, MSDOS.SYS, COMMAND.COM.

On note que la deuxième copie de la FAT est faite pour supprimer les erreurs d'entrées/sorties éventuelles à l'écriture sur la disquette ( si détection d'erreur sur la 1<sup>ère</sup> FAT ; possibilité d'aller lire la seconde ).

Si un de ces secteurs " réservés " est endommagé, la disquette est inutilisable ( ou utilisable avec de très grands risques d'erreurs car le système ne sait trouver ses informations qu'à ces endroits là ).

LE BOOT : ce secteur contient des informations relatives au format du disque, ainsi que le programme de chargement en mémoire du système d'exploitation MS-DOS.

ADRESSE	LONGUEUR	CONTENU
00	3	Instruction de saut vers le programme de chargement
03	8	Nom du constructeur et version du syst. D'exploitation
11	2	Taille des secteurs en octets
13	1	Taille des clusters en secteurs
14	2	Nombre de secteurs réservés au BOOT
16	1	Nombre de tables d'allocation ( FAT )
17	2	Nombre d'entrées du répertoire principal
19	2	Taille du disque en secteurs
21	1	Type du disque ( Média descriptor )
22	2	Taille de la FAT
24	2	Nombre de secteurs par piste
26	2	Nombre de têtes
28	2	Nombre de secteurs cachés
30	-	Programme de chargement

Fig.2.4 - Organisation logique des informaions.

Les octets contenus dans les secteurs sont exprimés en hexadécimal. De plus lorsqu'une information est stockée sous forme d'un mot de 16 bits on trouve d'abord l'octet de poids faible suivi de l'octet de poids fort, dans le cas d'un double mot, il y a une inversion au niveau du mot.

#### PRINCIPE DE STOCKAGE D'INFORMATION:

Le système d'exploitation a besoin du " répertoire " et de la " FAT", pour retrouver les informations et le chaînage des fichiers sur la disquette.

Le répertoire contient les caractéristiques du fichier ( nom, extension, taille, date, heure...).

La FAT est un tableau indiquant l'état des clusters de tout le disque. Il faut savoir qu'un cluster est un nombre déterminé de secteurs pour un disque donné.

LE REPERTOIRE : il est situé à un endroit bien précis de la disquette. Il est constitué d'une table dont chaque entrée correspond soit au nom du disque ( si on a donné un nom de volume lors du formatage ), soit à un fichier ou à un sous-répertoire.

Chaque entrée faisant 32 Octets ; la taille du répertoire principal étant fixée à l'avance. Le nombre d'entrées dans le répertoire principal est donc limité. Chaque entrée peut-être découpée en 8 zones.(Fig .2.5)

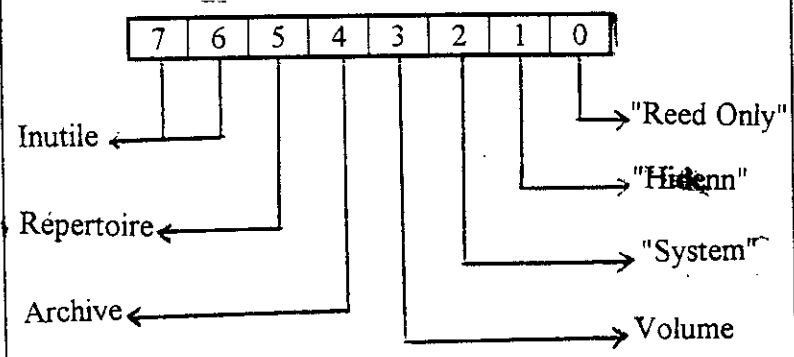
ZONE	LONGUEUR	DEFINITION
Nom	8	Nom du fichier, cadré à gauche Le 1er caractère a une signification particulière : 00 : Entrée non utilisée ( Libre ) E5 : Fichier effacé, entrée disponible. 2E : L'entrée correspond au répertoire lui-même. Si le 2 <sup>ème</sup> octet est également un point, l'entrée correspond au répertoire père. Autre : fichier.
Extension	3	Indique le type ou l'extension du fichier
Attribut	1	Cet octet contient : 
Réservé	10	Réservé à MS-DOS Sans Signification particulière
Heure	2	Heure de création ou de dernière mise à jour
Date	2	Date de création ou de dernière mise à jour
Cluster	2	Numéro du 1er cluster alloué au fichier dans la FAT.
Taille	4	Taille du fichier en octet.

Fig.2.5. Organisation logique d'une entrée d'un répertoire.

La table d'allocation des fichiers ( File allocation table ou FAT ) : C'est une table qui indique l'état de chaque unité d'allocation ou cluster; l'état d'un cluster peut être ( disponible - utilisé - défectueux ). Chaque cluster du disque possède une entrée dans la FAT.

### CARACTERISTIQUES DES DISQUETTES

On différencie entre les disquettes selon leurs formats, les formats standards existants : DD - 5 "1/4 , HD - 5 "1/4 , DD 3 "1/2 , HD 3 "1/2 ... chaque format dispose de certaines caractéristiques normalisées ; le tableau suivant résume les différentes caractéristiques nécessaires à la recherche d'informations sur les disquettes.(Fig.2.6)

Capacité disquette	360K	1.2 M	720 K	1.44 M	2.88 M
Format	5" 1/4	5" 1/4	3" 1/2	3" 1/2	3" 1/2
Nombre de piste	40	80	80	80	80
nombre sect./piste	9	15	9	18	36
nbre total secteur	720	2400	1440	2880	5760
taille secteur	512	512	512	512	512
nbre sect./cluster	2	1	2	1	2
taille cluster	1024	512	1024	512	1024
taille FAT en sect.	2	7	3	9	9
Nbre de FAT	2	2	2	2	2
Expression du cluster dans la FAT en octet	1.5	1.5	1.5	1.5	1.5
Media descriptor	FD	F9	F9	f0	.
Taille répertoire en secteur	7	14	7	14	15
Nbre d'entrée max. Dans le répertoire principal	112	224	112	224	240

Fig.2.6. Caractéristiques des disquettes.

### 3 - ACCES AUX DISQUETTES AVEC LE BIOS [ 4 ], [ 8 ]

#### 3.1 - FONCTIONS DISQUETTE DE L'INTERRUPTION 13H

Le BIOS dispose de toute une série de fonctions concernant l'accès aux disquettes qui peuvent être appelées par l'interruption 13 h (Fig.2.7). Cette interruption joue en plus, le rôle d'interface pour les fonctions disque dur du BIOS.

En règle générale les fonctions disquettes et disques durs de même nature portent un numéro de fonction identique. La différenciation s'effectue dans ce cas par l'indication du lecteur qui doit être changé ( avant le lancement de la fonction ) dans le registre DL .

Concernant le lecteur on choisira entre la valeur 0 ( lecteur A ), ou 1 ( lecteur B ). En revanche le disque dur est représenté par la valeur 80 h ou 81h.

Fonctions Disquettes de l'int 13h du BIOS	
N°	Fonction
00h	Réinitialisation
01h	Lecture de l'état
02h	lecture
03h	Ecriture
04h	vérification
05h	Formatage
08h	Lecture des paramètres Disques
15h	Lecture du type de disque
16h	Détection ouverture lecteur
17h	" Format disquette
18h	" " "

Fig.2.7- Fonctions disquettes de l'interruption 13h du BIOS.

On doit également, charger le N° de la fonction dans le registre AH, avant le lancement de l'interruption. Après l'exécution d'une fonction de l'interruption 13h, un code d'état ( ou d'erreur ) est transmis dans le registre AH .(Fig.2.8)

Une erreur est alors différenciée par une valeur différente de zéro et est accompagnée d'un rapport d'erreur ( Carry-Flag ).

Codes d'état et d'erreurs des fonctions disquettes du BIOS	
Code	Signification
00h	Pas d'erreur
01h	Appel de fonction invalide
02h	Marque d'adresse non trouvé
03h	Tentative d'écriture sur disquette protégée
04h	Secteur non trouvé
06h	La disquette a été changée
08h	Débordement DMA
09h	Dépassement de la limite
10h	Erreur de lecture
20h	Erreur de contrôleur de disquette
40h	Piste non trouvée
80h	Erreur de Time Out le lecteur ne réagit pas

Fig.2.8 - Codes d'état et d'erreurs des fonctions disquettes du BIOS.

### 3.2 - ETUDE DES FONCTIONS DISQUETTES DE L'INTERRUPTION 13H

#### 3.2.1 - REINITIALISATION DU LECTEUR DE DISQUETTE

Si après le lancement d'une fonction disquette, on constate à l'aide du Code d'état ou d'erreur qu'une erreur s'est produite, il est alors préférable de réinitialiser le lecteur de disquettes à l'aide de la fonction 00h . Celle-ci n'attend rien d'autre que le numéro de la

fonction ( AH ) et l'indicateur ( DL ) mais elle communiquera de son côté l'état actuel du lecteur dans le registre AH.

L'indication du N° de lecteur n'a aucune importance car, une réinitialisation s'effectue systématiquement sur l'ensemble des lecteurs de disquettes.

### 3.2.2 - DETECTION DU TYPE DE DISQUE

Un programme ne sait pas toujours d'emblée à quel type de disque il a affaire. Les fonctions 08h et 15h permettent toutefois de communiquer ces informations. La première de ces deux fonctions existe déjà sur le BIOS du PC/XT et sert à différencier les différents formats de lecteurs et de disquettes. Lors de son lancement elle attend uniquement le N° de la fonction dans le registre AH et le N° du lecteur dans le registre DL.

Après l'exécution de cette fonction toute une série d'informations est transmise dans les registres internes du microprocesseur. (Fig.2.9)

Informations transmises par la fonction 08h de l'int 13h.	
Registre	Informations
BL	Type du lecteur 01h = 5"25 360 Ko 02h = 5"25 1.2 Mo 03h = 3"5 720 Ko 04h = 3"5 1.44 Mo
DH	Plus grand N° de faces ( toujours = 1 )
CH	Plus grand N° de pistes
CL	" " " de secteurs
ES:DI	Pointe sur DDPT ( table des paramètres Disque )

Fig.2.9 - Informations transmises par la fonction 08h de l'int 13h.

Notons que le registre BL indique non seulement le format du secteur ( 3 " 1/2 ou 5 " 1/4 ) mais également le format des disquettes ( DD, HD ). Le code des lecteurs permet de déduire automatiquement le nombre de secteurs, de pistes et de têtes.

La table des paramètres disque ( DDPT ) référencée par le pointeur dans les deux registres ES : DI, contient les paramètres nécessaires au BIOS pour la programmation du contrôleur de disquettes. La fonction 15h n'est supportée que par les PC/AT.

### 3.2.3 - LECTURE DES SECTEURS DE DISQUETTE

L'une des fonctions de base que le BIOS est chargé de mettre à disposition consiste à effectuer une lecture des différents secteurs des disquettes. La fonction 02h permet d'effectuer ce travail. Il est nécessaire d'indiquer : les secteurs qui doivent être lus ; en effet, plusieurs secteurs peuvent être lus simultanément à condition qu'ils fassent partie d'une même piste et qu'ils soient contiguës.

Les registres suivants doivent être chargés avant de lancer la fonction.(Fig.2.10)

Registre	Contenu
DL	Numéro de disque
DH	Numéro de tête
CL	Numéro de secteur
CH	Numéro de piste
AL	Nombre de secteurs à lire

Fig.2.10 - Registres d'entrée de la fonction 02h de l'int 13h.

Les données ne peuvent être transférées automatiquement dans une zone de mémoire fixe ; l'adresse d'un tampon doit être indiquée dans le compte de registres ES : DX. Il est recommandé de répéter au moins trois fois chaque tentative de lecture, écriture ou formatage avant de se considérer vaincu, et de supposer une erreur. Il arrive très fréquemment qu'une opération échoue lors de la première tentative, mais réussisse lors d'une seconde voire, une troisième fois. Cela provient du fait que la tête de lecture / écriture n'était pas correctement positionnée lors de la 1<sup>ère</sup> tentative et que le lecteur n'était pas correctement synchronisé avec les variations du signal électrique.



Il n'y a aucun danger qu'une erreur soit inscrite dans les données car le lecteur enregistre pour chaque secteur un " checksum " ( vérification de la somme ) permettant de contrôler la cohérence des informations lues.

### 3.2.4 - ECRITURE DES SECTEURS DE DISQUETTE

La fonction 03h est mise en oeuvre pour l'écriture sur les différents secteurs. Les entrées de registres de cette fonction ressemblent à celle de la fonction 02h.(Fig.2.11)

Registres lors du lancement de la fonction 03h de l'int 13h	
Registre	Contenu
AL	Nombre de secteurs à écrire
DL	Numéro du disque (lecteur)
DH	Numéro de tête (face)
CL	Numéro du secteur
CH	Numéro de la piste
ES:BX	Adresse de la zone mémoire qui contient les secteurs à écrire.

Fig.2.11 - Registres d'entrée de la fonction 03h de l'int 13h.  
La zone mémoire doit être remplie avant le lancement de la fonction.

### 3.2.5 - VERIFIER LES SECTEURS DE LA DISQUETTE

La fonction 04h est utilisée pour vérifier que les données ont été correctement transférées sur la disquette. Ceci correspond à une vérification à l'aide de la valeur CRC. ( Cyclical Redudancy check ) ; c'est une procédure de test très fiable au cours de laquelle les valeurs de chaque Octet à l'intérieur d'un secteur vont être associées à une somme par le biais d'une formule mathématique compliquée.

Vu que la plupart des lecteurs de disquettes sont très fiables et suffisamment précis, cette routine est considérée par la plupart des programmeurs comme superflue.

Les entrées de registres sont semblables à celles des fonctions 02h Et 03h. Avec toutefois une différence : aucune adresse de zone mémoire ne sera communiquée.

### 3.2.6 - LE FORMATAGE DES PISTES DE DISQUETTES

Registres lors du lancement de la fonction 05h	
Registre	Contenu
AL	Nombre de secteurs par piste
DL	Numéro de disque
CH	Numéro de piste
DH	Numéro de tête (face)
ES:BX	Adresse de la zone mémoire qui contient la DDPT.

Fig.2.12 - Registres d'entrées de la fonction 05h de l'int 13h.

Lorsque le format souhaité est paramétré et que la table des paramètres disques est installée, la procédure de formatage proprement dite peut être lancée.

A cet effet on fait appel à la fonction 05h. Qui permet le formatage d'une piste " complète ".

Chaque secteur peut être formaté à raison de 128, 256, 512, ou même 1024 octets par secteur. Il est toutefois impératif de sélectionner le format 512 octets si la disquette doit ensuite être utilisée sous DOS.

### 3.2.7 - La DDPT(Table des paramètres du lecteur de disquettes)

Mise à part les indications concernant le format physique de la disquette, le BIOS a besoin, pour la programmation du contrôleur de disquettes, de toutes une série d'informations supplémentaires se trouvant dans la DDPT.

Une table de ce type existe dans le BIOS en ROM. Pour chaque lecteur et chaque format de disquettes supporté. Il est toutefois possible d'installer une DDPT particulière car le BIOS repère la table des paramètres disques courantes au moyen d'un pointeur FAR fixé dans une zone mémoire affectée à l'interruption 1Eh.

La possibilité qui consiste à installer une DDPT particulière est utilisée par le DOS qui modifie certaines valeurs de cette table. La DDPT elle même se compose de 11 octets comme nous le montre ce tableau.(Fig.2.13)

Structure de la table des paramètres du lecteur	
Valeur	Signification
00h	Vitesse
01h	Heard Load time
02h	Laps de temps
03h	Taille du secteur
04h	Secteurs par piste
05h	Taille du Gap3 en lecture/écriture
.....	.....

Fig.2.13 - Structure de la table des paramètres du lecteur.

Dans le 4<sup>ème</sup> octet on trouve le nombre d'octets par secteur avec lequel on doit travailler pour les opérations de lecture ou écriture. Cette valeur doit bien sûr coïncider avec celle utilisée pour le formatage d'un secteur.

L'octet 4 indique le nombre maximal de secteurs par piste lequel dépend du format de disquette choisi.

#### **4 - FORMATAGE DIRECT [ 8 ]**

Dans cette section on présentera un programme écrit en langage évolué " PASCAL " capable d'effectuer le formatage d'une disquette ( exactement comme la commande FORMAT du DOS ). [ Annexe ]:

Le programme ne se contentera pas d'effectuer le formatage physique de la disquette mais inscrit également les différentes structures de données requises par le DOS ; cela concerne notamment le secteur d'amorçage, le répertoire principal de la disquette, vide dans un premier temps ; ainsi que la FAT.

##### **i)- FORMATAGE PHYSIQUE**

Le formatage proprement dit s'exécute à l'aide de la fonction Physical Format qui a recourt à la procédure Format Track qui correspond à la fonction disquette 05h. Du BIOS afin de formater respectivement une piste. Elle est exécutée pour les pistes de 0 à N chaque piste est tout d'abord formatée sur la face 0 puis ensuite sur la face 1. Il sera bien sûr possible de formater tout d'abord complètement la première face pour ensuite passer à la deuxième ; mais la procédure durerait alors plus longtemps car la tête de lecture / écriture devrait alors balayer deux fois la surface totale de la disquette. Certes cette méthode requiert de basculer incessamment de la tête 0 à la tête 1, mais elle reste toutefois plus rapide que de déplacer la totalité du bras de lecture / écriture de piste en piste.

##### **ii)- FORMATAGE LOGIQUE**

A l'aide de la fonction "logical Format", les différentes structures de données vont être copiées sur la disquette, structures qui permettent au DOS de gérer les fichiers. Le secteur d'amorçage ( BOOT ) est inscrit dans la disquette pour le cas où celle-ci se trouverait dans le lecteur au moment de l'amorçage. Son contenu est défini au début du programme dans la variable Masque Boot.

## **5 - TECHNIQUES LOGICIELLES DE PROTECTION DE LOGICIELS**

### **5.1 - NORME DOS**

Lors de l'exécution d'une opération avec les commandes du DOS ( tel que Copy, DISKCopy, ... ) le système considère les caractéristiques normalisées des disquettes ( taille des secteurs : 512 octets, nombre de pistes, ... ). on dit qu'on fait accès aux disquettes selon la Norme DOS.

Comme indiqué au paragraphe §1; on peut accéder aux disquettes en utilisant les fonctions BIOS et cela n'exige pas d'obéir à ces normes.

Vu que l'utilisateur ne fait accès à celles-ci que par l'intermédiaire du DOS, le développeur peut manipuler certaines variables dans le formatage, l'écriture et la lecture des disquettes pour faire une bonne protection des logiciels.

### **5.2 - TECHNIQUES DE PROTECTIONS**

#### **5.2.1 - ECRITURE SUR LA PISTE 80**

L'astuce est de formater la piste 80 ( ou même 81 ou 82 ) qui est inaccessible par le DOS, puis d'écrire un mot de passe (ou texte) dessus; notons que l'accès à ce mot de passe (ou texte) se fera à l'aide d'un programme qui fait appel aux fonctions de l'int 13h. Du BIOS.

Cette opération doit être faite sur toutes les disquettes qui vont être des " copies originales ".

Avant de copier le logiciel sur ces disquettes, on intercale dans celui-ci une routine de test (voir Fig.2.14);qui consiste à faire une lecture au niveau de la piste 80 et de comparer son contenu avec la signature gravée sur les disquettes " Originales ". Si la comparaison est positive, le programme ( ou logiciel ) protégé est lancé, et l'utilisateur peut y accéder facilement.

Dans le cas contraire, on affiche un message signalant la protection du logiciel, ou une action déterminée( Reformatage de la disquette, Destruction du fichier exécutable,... ).

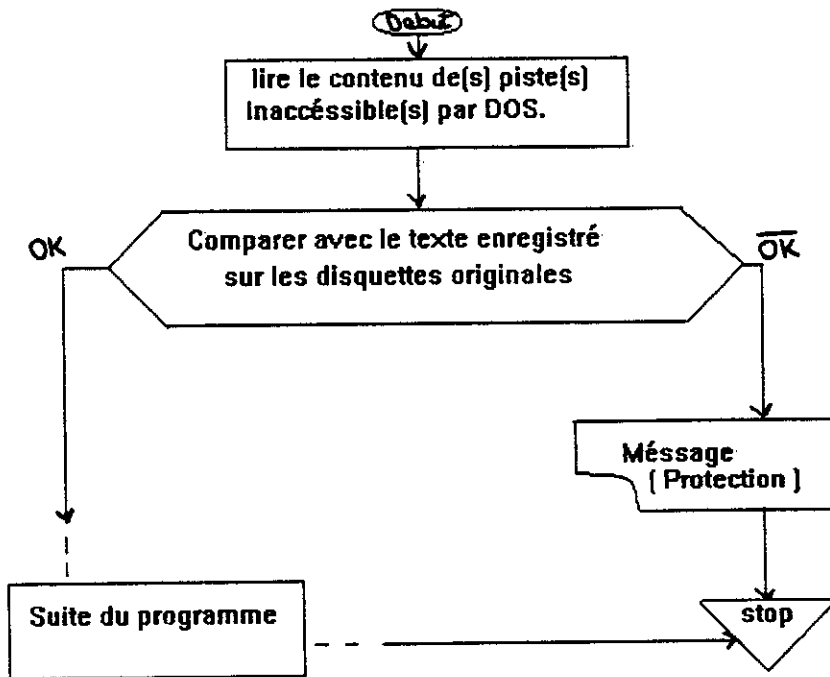


Fig.2.14 - Organigramme de la routine de test.

### 5.2.2 - LA PISTE FORMATEE HORS-NORMES

Comme le DOS ne peut accéder qu'à une piste formatée selon ses normes, on peut utiliser une piste formatée hors-normes pour protéger un programme.

L'astuce est d'écrire sur cette piste formatée hors-normes un mot de passe ( texte ); puis refaire la même procédure pour toute disquette souhaitée être "disquette originale".

Avant de copier le logiciel sur ces disquettes on doit, comme précédemment intercaler une routine de test dans celui-ci.

Idem que pour la technique précédente, le test consiste à lire le contenu de cette piste (qui a un format particulier); et de le comparer avec une signature donnée...

### **5.2.3 - DETOURNEMENT DE LA PROTECTION**

le problème fondamental posé aux gens qui travaillent sur la protection des logiciels est le détournement des fraudeurs.

Pour les deux méthodes exposées précédemment, on peut écrire des programmes qui peuvent détecter le type de protection utilisé, et de reproduire par la suite la disquette originale, et donc la possibilité de diffuser des copies dupliquées.

Pour cette raison ( sécurité ) on travail pour renforcer la protection des logiciels pour la rendre incontournable.

### **5.2.4 - DESTRUCTION PHYSIQUE D'UN SECTEUR**

Il s'agit d'une destruction physique d'un secteur par laser ( troue microscopique dans la disquette ). Ce secteur est déclaré mauvais dans la FAT. Le logiciel, dans sa routine de test, essaye tout simplement d'aller écrire une information sur ce secteur puis aussitôt ira relire ce secteur. Bien sûr, le secteur étant détruit physiquement, la lecture ne correspondra pas à ce qui a été écrit, ce qui veut dire que c'est bien la disquette originale.

Si l'on recopie cette disquette par la commande DISKCOPY; la recopie s'effectue secteur par secteur, donc dans la FAT. Le secteur ( cluster ) en question sera bien déclaré mauvais.

Le logiciel pour tester; allant écrire, trouvera la même valeur qu'à la lecture, ce qui veut dire que malgré la déclaration dans la FAT, le secteur n'était pas physiquement mauvais, et donc qu'il ne s'agit que d'une copie.

## **6 - Developpement d'une protection logicielle**

D' après les techniques logicielles de protection qu'on vu dans le paragraphe précédent; on a élaboré une protection logicielle, et cela en combinanr entre les deux premieres techniques.

De la deuxième technique ( La piste formatée hors normes ), on a fait un formatage hors normes de la piste 80.

Puis on a écrit un mot de passe ( texte, signature,... ), et cela en utilisant la première technique ( Ecriture sur la piste 80 ).



## CONCLUSION

L'implémentation de la méthode avec ses deux versions :

- Formatage hors normes.
- Ecriture dans la piste 80.

nous a permis de voir l'efficacité de la méthode et la simplicité de sa mise en oeuvre lorsqu'on est équipé de la maîtrise de la programmation système.

Parmi les avantages de la protection logicielle qu'on a réalisé :

### 1 - Garder toute la capacité mémoire de la disquette:

Parceque, en faisant un formatage hors normes d'une piste inferieure à la piste 80, on a alors diminuer la cpacité mémoire de la disquette; car le DOS déclarera cette piste comme étant une piste défectueuse.

### 2 - Renforcer la protection:

Notre signature (texte, mot de passe,... ) est :

- Dans la piste 80 ( incéssible par le DOS ).
- Ecrite avec une norme non standard ( inintelligible par le DOS ).

La méthode de protection logicielle qu'on a developpé peut être réalisée pour les autres supports d'information ( Disque dur, CD-ROM,... ).

Mais pour protéger notre technique de protection logicielle, il faut couper court à toute tentative de dépistage de la protection, tel que l'utilisation de l'iltitaire DEBUG du DOS; et

pour cela, il faut réaliser un petit programme qui a pour but d'occuper les interruptions 1 et 3 du BIOS ( voir Fig.2.2 ) et d'empêcher DEBUG de travailler.

## **CHAPITRE 3**

### **PROTECTION PAR UNE RESSOURCE MATERIELLE OU "ADD ON "**

## **RESUME**

Dans ce chapitre, on étudiera un type de protection matériel, qui consiste en l'utilisation d'un circuit à intercaler entre un PC et une imprimante.

Avant de présenter le développement de la carte réalisée, nous présenterons une étude générale sur l'interfaçage du PC, orienté spécialement vers la liaison parallèle, dont on va étudier sa programmation et ses différentes liaisons.

Enfin, nous présenterons le développement de notre carte qui est réalisée à base de micro-contrôleur 68705 P3 [ Annexe ].

## **INTRODUCTION**

Parmi les divers moyens utilisés pour protéger les logiciels contre les copies illicites, le "Dongle" arrive en bonne position. Le secret est généralement bien gardé quant au contenu de ces petites "boîtes noires" que l'on intercale le plus souvent entre l'ordinateur et le câble d'imprimante.

Dans la majorité des cas, un Dongle est un petit boîtier réunissant deux embases DB25 ( un mâle, et un femelle ), et prévu pour venir s'intercaler entre un PC et son imprimante parallèle.

Certains modèles peuvent également être installés sur un port série, une résine d'enrobage est coulée dans le boîtier afin d'interdire l'examen de son schéma interne et donc son éventuelle reproduction.

Bien évidemment, la présence du dongle ne doit modifier en rien le fonctionnement de l'imprimante, et encore moins celui du PC et des logiciels dont il est équipé.

Le principe de la protection est le suivant : dès son lancement, le logiciel protégé interroge le dongle en envoyant un passage codé sur la piste de l'imprimante; s'il est présent, le dongle répond par un autre message codé que le logiciel compare à ce qu'il attend, s'il y a coïncidence l'exécution du programme se poursuit. Si le dongle ne répond pas ou si le message reçu n'est pas conforme ( utilisation d'un dongle différent ), le logiciel réagit selon une procédure défensive ( message d'erreur, debut de formatage du disque dur,... ).

Cette protection n'est pas unique dans ce domaine, d'autres types existent, obéissant au même principe, et utilisant d'autres procédés d'interfaçage. A titre d'exemple : on utilise des cartes s'enfichants dans l'un des slots du PC. Cette carte regroupe quelques circuits intégrés disponibles pour y installer un boîtier représentant une clef personnalisée dont le code de reconnaissance est unique. Avec ce système, la protection s'effectue de manière simple et présente des nets avantages surtout en terme de fiabilité et de transparence pour l'utilisateur.

## **1 - INTERFACAGE AVEC LE PC**

### **1.1 - LE PC COMPATIBLE [ 7 ]**

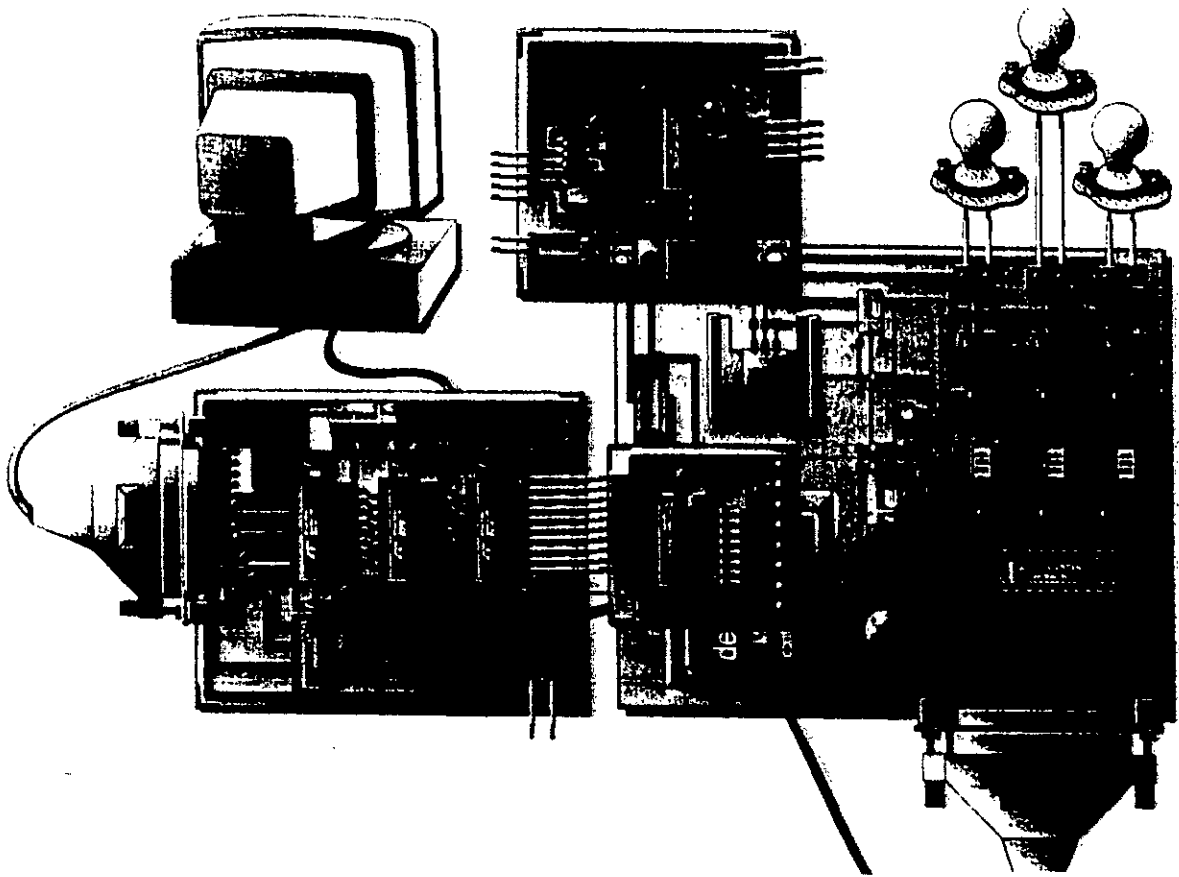
Le PC compatible est devenu le standard en matière d'informatique individuelle. Il est fabriqué à plusieurs centaines de millions d'exemplaires.

Les raisons du succès de cette machine sont multiples :

- Le principe de sa conception de type modulaire avec slots sur lesquels il est possible d'ajouter des cartes supplémentaires ; ainsi de nombreuses sociétés virent le jour, qui se spécialisent dans les cartes vidéo, les cartes de saisie de données, les cartes contrôleurs, ...etc. Cette configuration de type modulaire, permet de faire évoluer la machine au rythme des nouveautés apportées sur ces cartes externes.

- Ces caractéristiques font du PC Compatible une machine universelle accessible par tous, et beaucoup utilisée par les électroniciens, que se soit pour la conception des circuits imprimés, la simulation de fonctionnement des circuits électroniques, ou la commande d'automates et les saisies de données.

Le PC dispose également de ports d'entrées - sorties comme le connecteur imprimante ou la liaison série qui permettent la connexion de circuits autres que ceux pour lesquels ils ont été conçus. Le montage que nous décrirons par la suite utilise le port parallèle pour communiquer avec le PC.



Le choix des différents composants et la conception de la carte d'interface dépend du cahier de charge établi à partir de la fonction désirée par la carte ainsi que du périphérique.

Les éléments principaux constituant l'interfaçage avec le PC sont : (Fig.3.1)

- Le PC lui même.
- Des cartes d'interface.
- Le(s) périphérique(s).

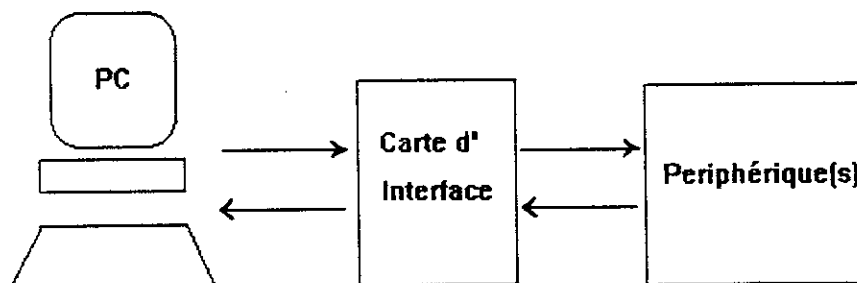


Fig .3.1- Divers constituants d'un interfaçage avec PC.

La carte d'interface comprend en général :

- Un micro-contrôleur ou un micro-processeur : contiendra les sous-programmes de traitement pour le bon fonctionnement du système.
- Un port parallèle et / ou un port série : permettant la communication entre PC et la carte en respectant les micro-séquences du constructeur. (Fig.3.2)

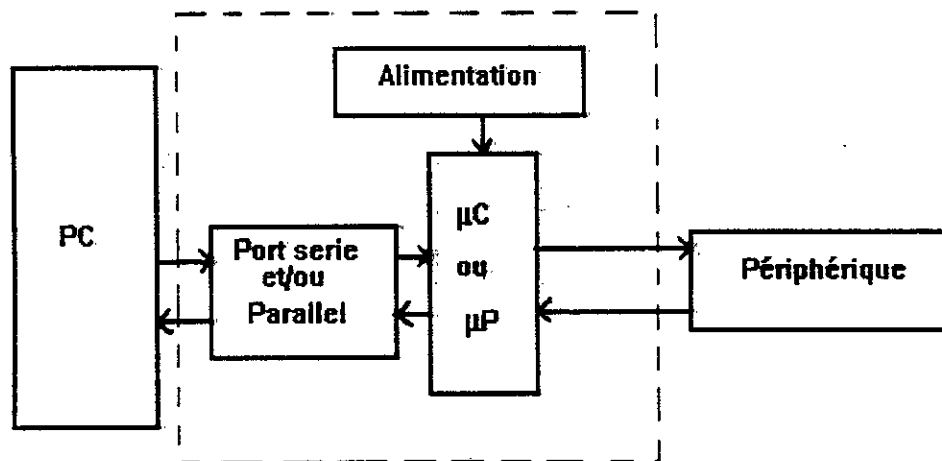


Fig.3.2 - Schéma synoptique de la carte d'interface.

## 1.2 - DEVELOPPEMENT D'UNE CARTE D'INTERFACE

Après l'établissement du schéma électronique de la carte, le développement se passe sur deux axes : (Fig.3.3)

1 - HARD : il faut tester les différents organes de la carte avant la mise au point, en cas d'erreur, il faut revoir le schéma électronique de la carte.

2 - SOFT :

- Dresser l'organigramme selon le protocole de communication.
- Ecrire les programmes en langage machine.
- Programmation de l' EPROM et du microcontrôleur.



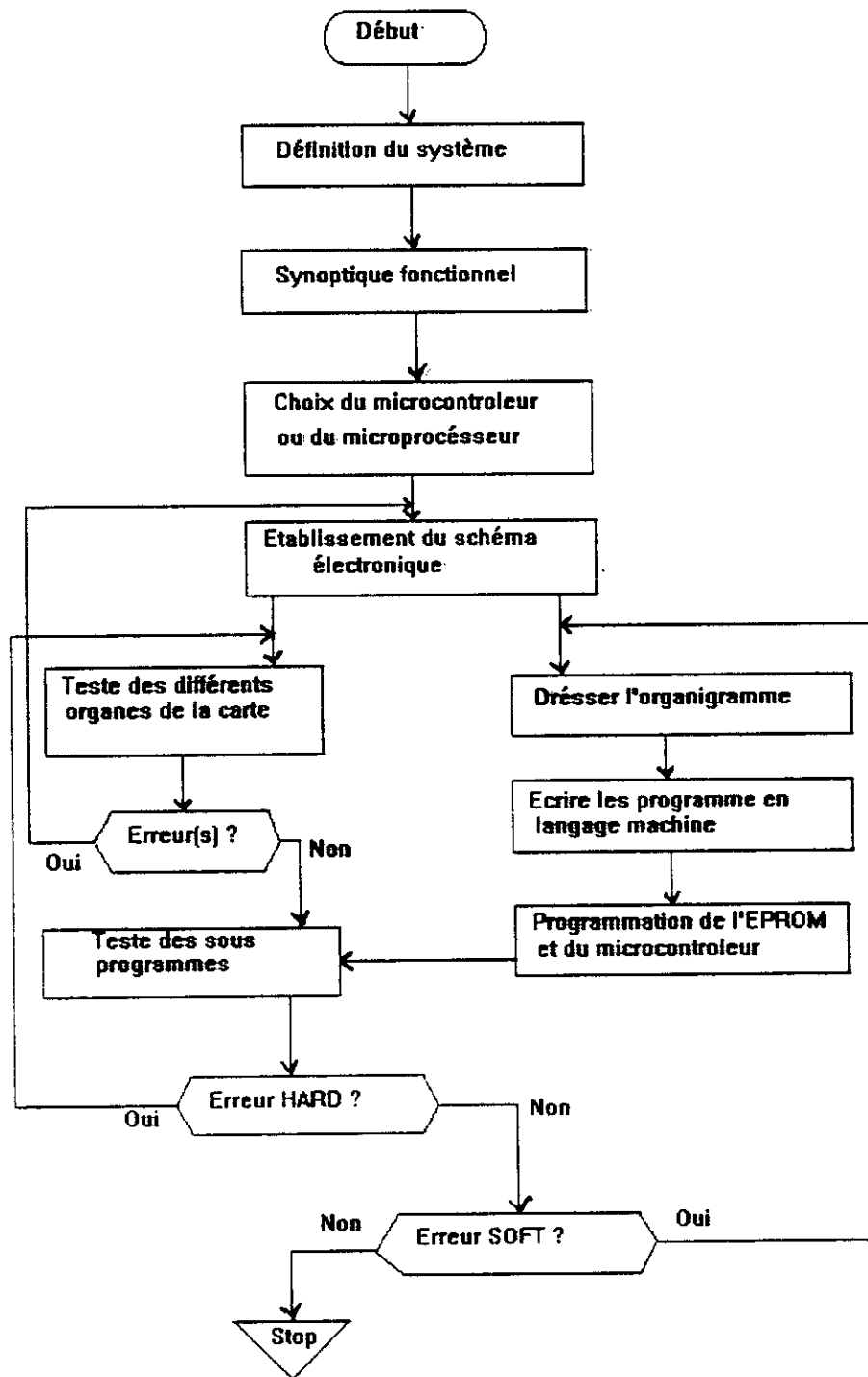


Fig.3.3 - Différentes étapes de développement de la carte d'interface.

## 2 - ETUDE DU PORT PARALLELE [ 8 ], [ 10 ]

Le port parallèle est le moyen le plus facile et le plus sûr pour transmettre les données sur de courtes distances.

Dans cette section, on étudiera les signaux mis en oeuvre dans la réalisation d'une liaison avec ce port, ce qui nous conduira à l'étude de la programmation de ce dernier.

### 2.1 - BROCHAGE ET SIGNAUX

Le port parallèle fait appel à de multiples signaux qui sont divisés en signaux de données, de contrôle, et d'état.

La plupart de ces signaux sont destinés à faire des transferts entre l'ordinateur et l'imprimante, utilisant ainsi le connecteur d' IBM de 25 broches du côté ordinateur et le connecteur centronics 36 broches du côté imprimante.

Ordinateur PIN	Imprimante PIN	Nom	Fonction
1	1	Strobe	Données envoyées
2	2	Do	Ligne de Donnée bit 0
3	3	D1	
:			
9	9	D7	Ligne de Donnée bit 7
10	10	ACK	Donnée reçue
11	11	Busy	Imprimante occupée
12	12	Paper End (P.E.)	Plus de papier sur l'imprimante
13	13	Select	Imprimante ON-LINE
14	14	Autofeed	Changement de ligne
15	32	Error	Erreur de transmission
16	31	INIT	Réinitialisation de l'imprimante
17	36	SélectIN	Mise On-line de l'imprimante
18...25	19...30	GND	Masses

Fig.3.4- Brochage et signaux du port parallèle.

- LES DONNEES

Les huit lignes de données véhiculent l'octet à émettre. Cet octet constitue le code ASCII du caractère. Le niveau de ces signaux est celui de TTL. Ces signaux sont unidirectionnels.

- SIGNAL STROBE

Ce signal actif au niveau bas, indique que les données envoyées par le micro-ordinateur sont valides. La synchronisation du signal STROBE avec les lignes de données est très importante, il faut que l'octet à émettre soit présent sur les lignes de données avant la validation de la ligne STROBE et la destination doit disposer du temps nécessaire pour la lecture. L'impulsion STROBE est donc d'une durée de 1  $\mu$ s.

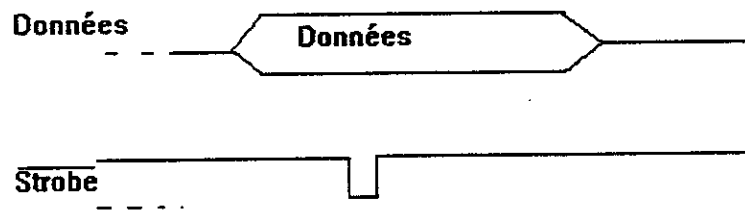


Fig.3.5 - Chronogramme du signal strobe.

- BUSY

C'est un signal envoyé par le destinataire ( imprimante ) signifiant qu'il est occupé à traiter le caractère précédent et qu'il ne peut recevoir d'autres données.

Le signal est actif à l'état bas. Il prend la valeur zéro lorsque l'imprimante est occupée. Sa durée dépend de l'imprimante ( Mémoire TAMPON ).

- SIGNAL ACK ( ACKNOWLEDGE )

C'est un signal signifiant que l'octet envoyé a été bien reçu et que le destinataire est prêt à recevoir de nouveaux caractères.

- SELECT

Ce signal indique que l'imprimante est sélectionnée ; c'est à dire connectée et prête à recevoir les données. S'il est à l'état bas, aucune donnée ne transite.

- PAPER -END

C'est un signal envoyé par l'imprimante vers l'ordinateur indiquant qu'il y a plus de papier, ce qui a pour effet d'interrompre la transmission.

- ERROR

L'imprimante positionne cette ligne à l'état haut à l'apparition d'une anomalie de fonctionnement, ce qui permet de signifier l'interruption de la liaison.

- SELECTIN

C'est un signal qui joue le rôle d'interrupteur il donne l'initiative à l'ordinateur de sélectionner ou pas l'imprimante. S'il est à l'état bas, l'imprimante est sélectionnée.

- AUTOFEED

Ce signal commande le caractère retour-charriot. Lorsqu'il est à l'état bas, on a un saut de ligne automatique; s'il est à l'état haut, un retour ligne ne peut se produire que sous la commande du micro-ordinateur.

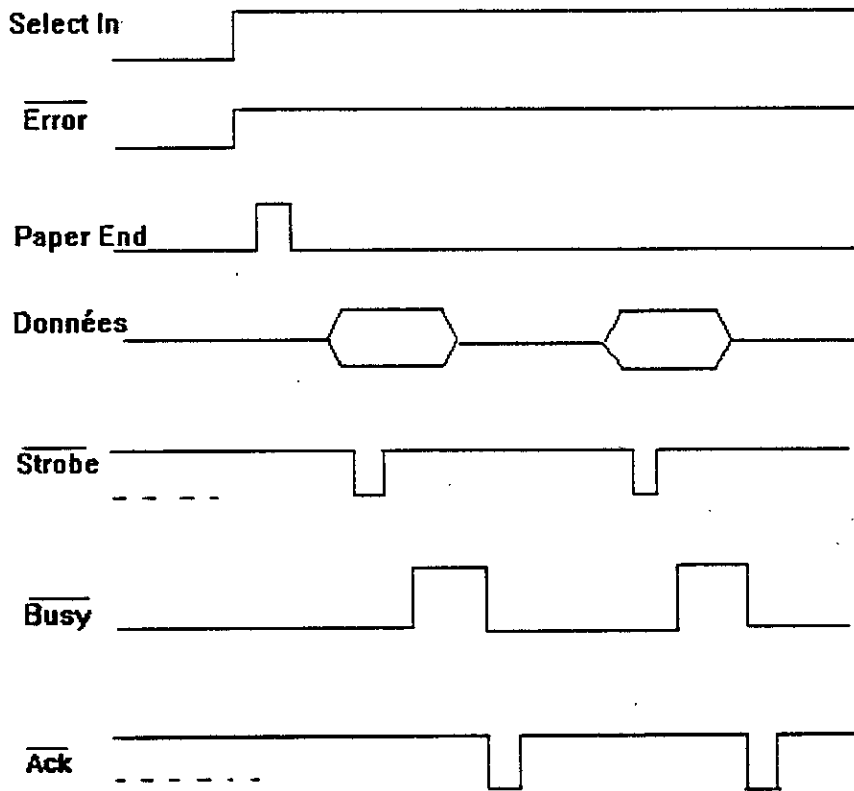


Fig.3.6 - Chronogramme du protocole de la liaison PC-Imprimante.

## 2.2 - PROGRAMMATION DIRECTE DE L'INTERFACE PARALLELE

Tant que le récepteur est capable de suivre l'émetteur, les fonctions du BIOS qui permettent l'émission des caractères sur l'interface parallèle sont parfaites. Mais si un ordinateur communique non pas avec une imprimante mais avec son semblable, les choses deviennent plus délicates. Car les vitesses de transfert exigées sont au-delà des capacités des fonctions du BIOS.

### 2.2.1 - LES PORTS D'ENTREE-SORTIE DES INTERFACES PARALLELES

Il est possible de brancher jusqu'à 3 interfaces parallèles sur un PC. Des espaces d'adressage correspondants sont soigneusement réservés à cet usage.(Fig.3.7)

Port	Interface
3BCh - 3BFh	Interface parallèle sur la carte MDA.
378h - 37Fh	1ère Interface parallèle.
278h - 27Fh	2ème Interface parallèle.

Fig.3.7 - adresses des interfaces paralleles.

Les adresse des ports ne se présentent pas dans l'ordre croissant. Tel est en effet l'ordre dans lequel le BIOS recherche les interfaces au démarrage du système:

### **2.2.2 - LES REGISTRES DE L'INTERFACE**

Indépendamment de leurs adressages, toutes les interfaces parallèles présentent trois registres situés au début de leur zone mémoire : par exemple 378h, 379h, et 37Ah pour la première Interface parallèle.

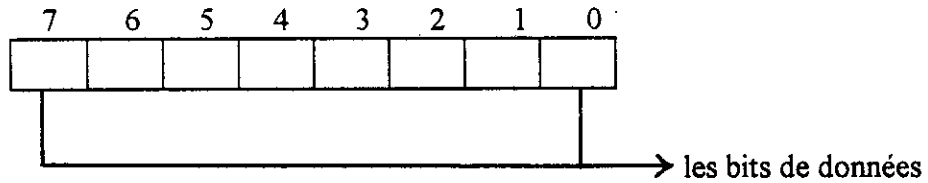
Si on charge la valeur 1 dans l'un des bits, la ligne associé est aussitôt mise sous tension ( ou placée en mode "HIGH" ). Inversement, si le bit est mis à zéro, la ligne est placé en mode "LOW". En principe chaque ligne garde son état jusqu'à ce que le bit associé soit changé par programme. On note que certains de ces lignes ont une logique inversée.

#### **i ) Registre des Données**

Les huit bit du registre de données ne sont pas inversés. Ils représentent les données qui doivent être transférées sur les lignes D0 - D7 . Il faut se rendre compte que ce registre est conçu comme un pur registre de sortie qui n'est pas destiné à réceptionner des données. Il est vrai qu'une imprimante n'est pas destinée à envoyer des données à l'ordinateur; en conséquence un certain nombre de problèmes vont apparaître lorsqu'on cherche à développer un logiciel de communication pour relier le PC avec une interface autre que l'imprimante.

\* LOW : niveau bas.

\* HIGH : niveau haut.



le registre entier est "Write Only"

Adresse d'offset : 0

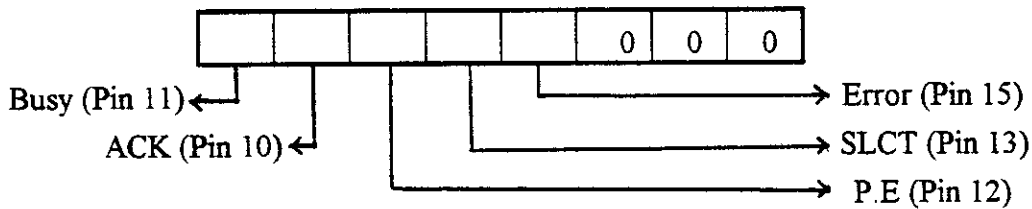
Adresse du port : MDA avec connexion parallèle 3BCh.

1. Connexion parallèle = 378h.

2. Connexion parallèle = 278h.

**ii ) Registre d'état**

Il n'est accessible qu'en lecture et ne peut pas recevoir des données. Il reflète des différentes lignes d'état de l'imprimante.



le registre entier est "read Only".

Adresse d'offset : 1

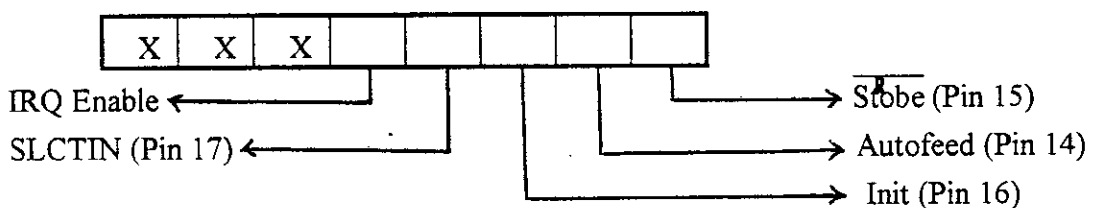
Adresse du port : MDA avec connexion parallèle 3BDh.

1. Connexion parallèle = 379h.

2. Connexion parallèle = 279h.

**iii ) Registre de Commande**

Le troisième registre sert à commander l'imprimante et le matériel. Par ailleurs il joue un rôle important dans la transmission des caractères.



le registre entier est "read -write".

Adresse d'offset : 1

Adresse de port : MDA avec connexion parallèle 3BCh.

1. Connexion parallèle = 37Ah.
2. Connexion parallèle = 27Ah.

### **3 - LIAISONS PARALLELES ENTRE PC ET INTERFACES [ 9 ], [ 10 ]**

#### **3.1 -COMMUNICATION PARALLELE PC-IMPRIMANTE :**

La signification des différentes lignes et de bits associés se conclut normalement dans les coulisses de la communication PC-Imprimante. L'octet à imprimer est d'abord chargé dans le premier registre de l'interface parallèle ( D0...D7 ). S'il est vrai que ces signaux arrivent instantanément à l'imprimante cette dernière a pourtant besoin d'un signal supplémentaire pour les traiter; il ne faut pas oublier que les lignes ( D0...D7) portent en permanence des informations, l'imprimante doit savoir si c'est un caractère à imprimer ou un reste de caractère déjà traité. C'est alors qu'intervient la ligne Strobe; en la mettant à 0 l'ordinateur signale la présence de données à imprimer. Il faut ensuite que ce signal soit très rapidement retiré, sinon l'imprimante risque d'effectuer deux fois la lecture du caractère. Mais il faut aussi un délai minimal de 1 microseconde pour laisser à l'électronique de l'imprimante le temps de lire le caractère.

Or une micro seconde ne peut être suffisante pour permettre à l'imprimante de suivre le rythme de transfert de données; elle signale alors par la ligne Busy qu'elle n'est pas encore prête pour la suite, normalement, ce signal est envoyé immédiatement après réception du signal STROBE pour que l'imprimante ait le temps de lire tranquillement le caractère et de le traiter convenablement.

Ce n'est pas tout, le signal ACK. émis sur la ligne de même nom doit également être mise à zéro par l'imprimante pour avertir l'ordinateur que le caractère a été réceptionné. S'il est vrai que la communication PC - Imprimante ressemble à un monologue, l'imprimante n'est quand même pas complètement muette elle dispose de trois lignes pour donner diverses informations sur son état : ERROR, SLCT, PE.



Ainsi, l'ordinateur dispose de diverses lignes qui lui permettent de commander l'imprimante : AUTOFEED, INIT, SLCTIN.

### 3.2 - LIAISON PARALLELE MODIFIEE POUR LA COMMUNICATION PC-PC

Il a été indiqué que la liaison parallèle ( centronics ) est unidirectionnelle car le registre de données et Write-Only, cependant, le port parallèle possède 5 lignes d'entrées qui véhiculent des informations d'état.

L'idée est donc d'utiliser ces lignes pour transmettre l'information entre deux PC.

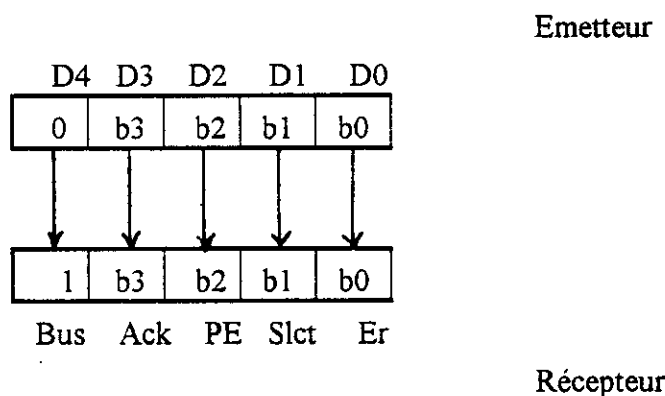
L'octet à transmettre doit être subdivisé en deux quartets qui seront transmis séquentiellement sur les lignes D0...D3.

On utilise donc les quatre lignes ERROR, SLCT, PE, ACK pour recevoir les données. La cinquième ligne Busy associée à la ligne D4 sera utilisée pour le contrôle des flux.

En pratique cette liaison est réalisée par un câble croisé dit " null-modem centronics ". Du point de vue logiciel, la communication parallèle bidirectionnelle est gérée par un programme utilisateur ou par un logiciel de communication spécialisé.

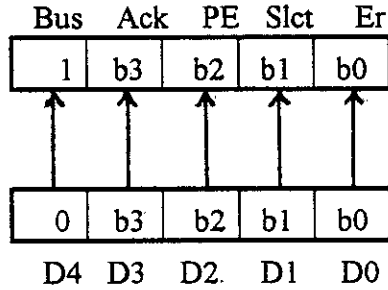
Le protocole de communication conforme au logiciel LAPLINK; respecte cinq étapes :

1°) Emission du premier quartet: l'émetteur commence par transmettre le quartet de poids faible. Il place le quartet sur les lignes D0..D3 et met la ligne D4 à 0 pour que le bit busy du récepteur correspond à 1.



2°) Accusé de réception du 1er quartet : le récepteur place le quartet reçu sur D0..D3 pour les retourner. Il met ensuite le bit D4 à 0 pour que le bit Busy correspond ) 1 chez l'émetteur et le quartet sera validé.

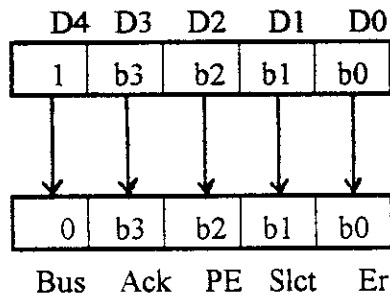
Emetteur



Récepteur

3°) Emission du 2ème quartet : l'émetteur après avoir reçu le 1° quartet transmis, place le quartet du poids fort sur les lignes de données et met D4 à 1, le bit Busy passe à 0 chez le récepteur et les données seront donc validées.

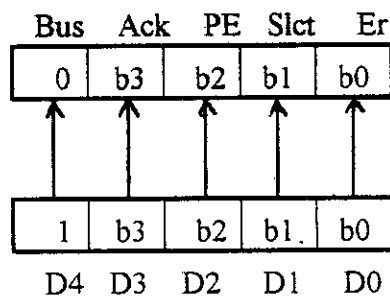
Emetteur



Récepteur

4°) Accusé de réception de l'octet de poids fort : le récepteur retourne le quartet reçu et met le bit D4 à 1 pour que l'émetteur renvoie la valeur 0 dans le bit Busy et valide ainsi les données présentes sur Er, SLC, PE, ACK.

Emetteur



Récepteur

5°) Constitution de l'octet et vérification du transfert : Dès que la communication est terminée, le récepteur constitue un octet à partir des deux quartets. A l'aide du message reçu, l'émetteur vérifie si les données sont transmises correctement.

### **3.3 - LIAISON BIDIRECTIONNELLE ENTRE PC ET MICRO-CONTROLEUR**

On peut adopter le même protocole de communication PC-PC pour réaliser une liaison PC-microcontrôleur; à l'aide de deux procédures l'une pour la réception (Fig.3.8.a), l'autre pour l'émission (Fig.3.8.b)

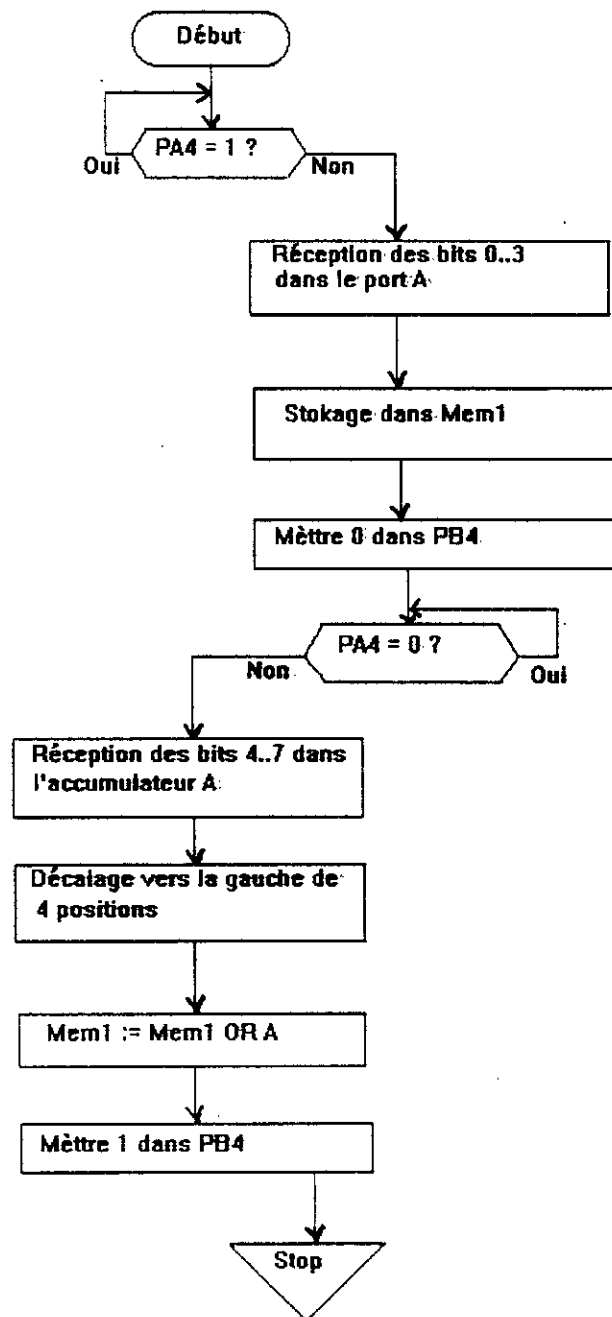


Fig.3.8.a - Procédure de réception d'un octet par le micro-contrôleur.

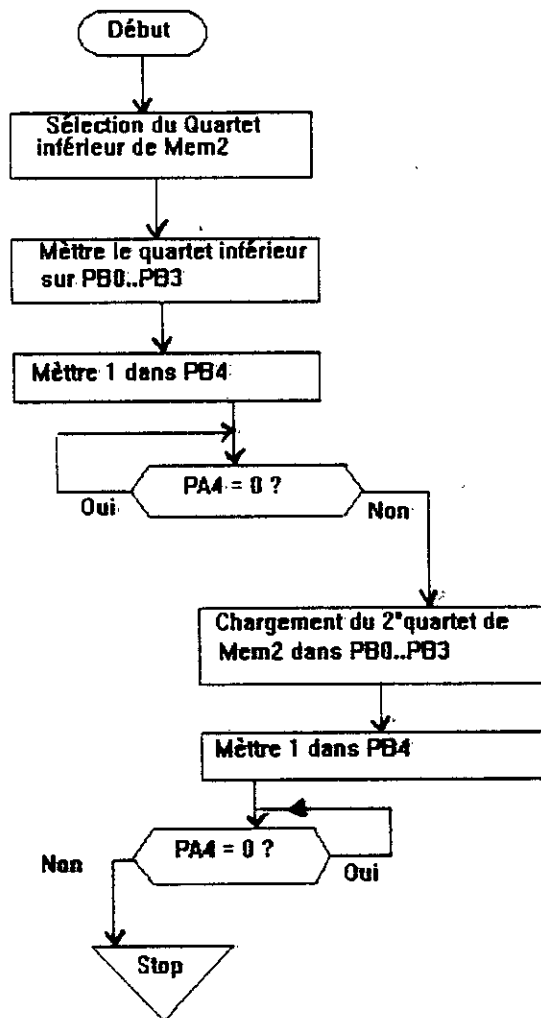


Fig.3.8.b - Procédure d'émission d'un octet par le micro-contrôleur.

#### 4 - DEVELOPPEMENT DE LA CARTE

##### 4.1 - DEFINITION DU SYSTEME

La protection par dongle n'est pas très compliquée. Une routine incluse dans le programme envoie un signal sur le port parallèle, le bouchon ( carte d'interface placée sur le port) reçoit ce signal et émet un signal en retour . Cette réponse est ensuite analysée par la routine du programme. (Fig 3.9)

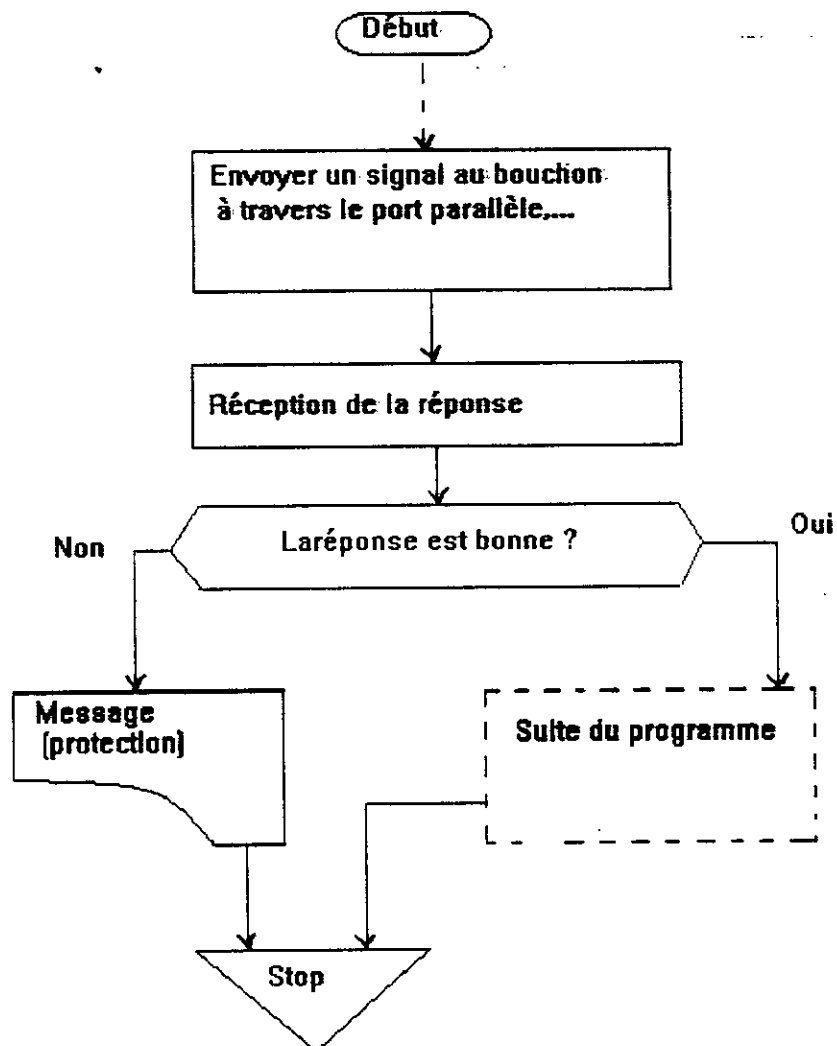


Fig 3.9 - Organigramme de la routine de test du "Dongle".

Ce système de protection qui semble être très fermé risque d'être cassé par la prise d'information au niveau du port parallèle; un analyseur analogique peut, en effet, enregistrer le flux d'informations transitant en entrée et en sortie sur le port.

A partir de là, il est possible d'identifier le "dongle" et concevoir d'autres exemplaires. Mais la mise en oeuvre d'une telle procédure n'est pas à la portée de l'utilisateur non averti, tant au niveau financier qu'au niveau technique. Pour contrer à cette éventualité, on propose des bouchons avec clés aléatoires et variable avec le temps. Il faut noter que le dongle ne doit pas empêcher l'imprimante d'être connectée avec le PC. (Fig 3.10)

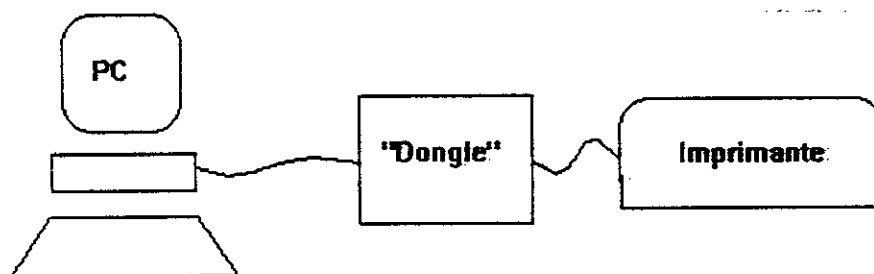


Fig 3.10 - Configuration du système.

#### 4.2 - SYNOPTIQUE FONCTIONNEL

D'après la description et la configuration du système, on exige dans la carte d'interface à vérifier les conditions suivantes :

- i) la carte doit être capable de faire un transfert de données bidirectionnel avec le PC.
- ii) On doit faire un multiplexage entre l'imprimante et le "dongle" pour qu'il soit "transparent" au transfert de données lors de l'impression. (Fig 3.11)



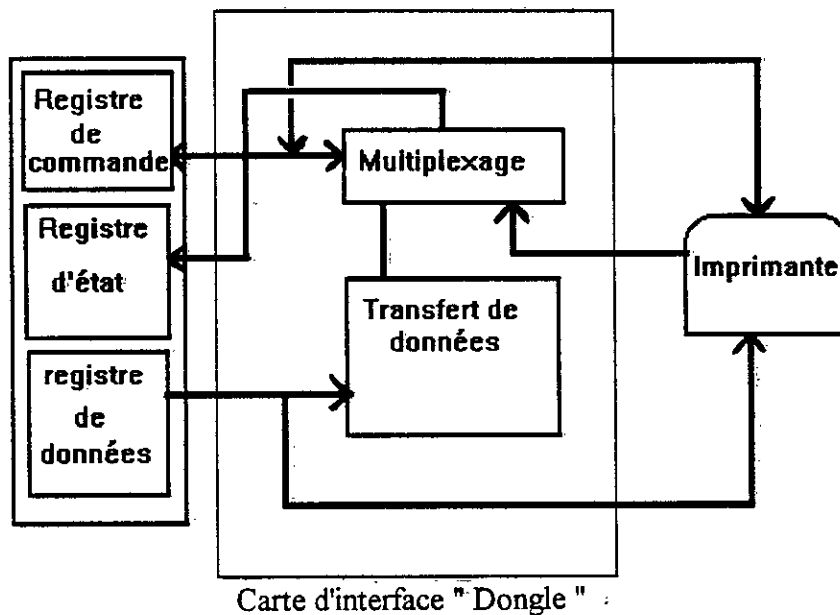


Fig.3.11 - Synoptique fonctionnel du système.

### 4.3 - CHOIX DU MICRO-CONTROLEUR

Pour la réalisation du "dongle" on exige des composants de faibles consommations et moins encombrants. C'est pourquoi pour des dongles professionnels on exploite les techniques de la micro-électronique pour réaliser des circuits spécialisés à cette fonction.

Or, la carte à réaliser ici est un modèle expérimental; de sorte que l'encombrement ne gêne pas, ainsi que la consommation peut être remédiée par une alimentation extérieure.

La limitation par la disponibilité nous oblige à utiliser le micro-contrôleur 68705 P3 de la famille motorola qui sera présenté par la suite [ Annexe ].

### 4.4 - SCHEMA ELECTRONIQUE

La Fig 3.12 nous permet de prévoir les circuits nécessaires dans notre carte à partir des différentes fonctions qui y existent.

#### 4.4.1 - Transfert de données

Le transfert de données est effectuée par le micro-contrôleur qui peut recevoir des données sous formes d'octet, ou d'y émettre selon un protocole de transmission ( Laplink par

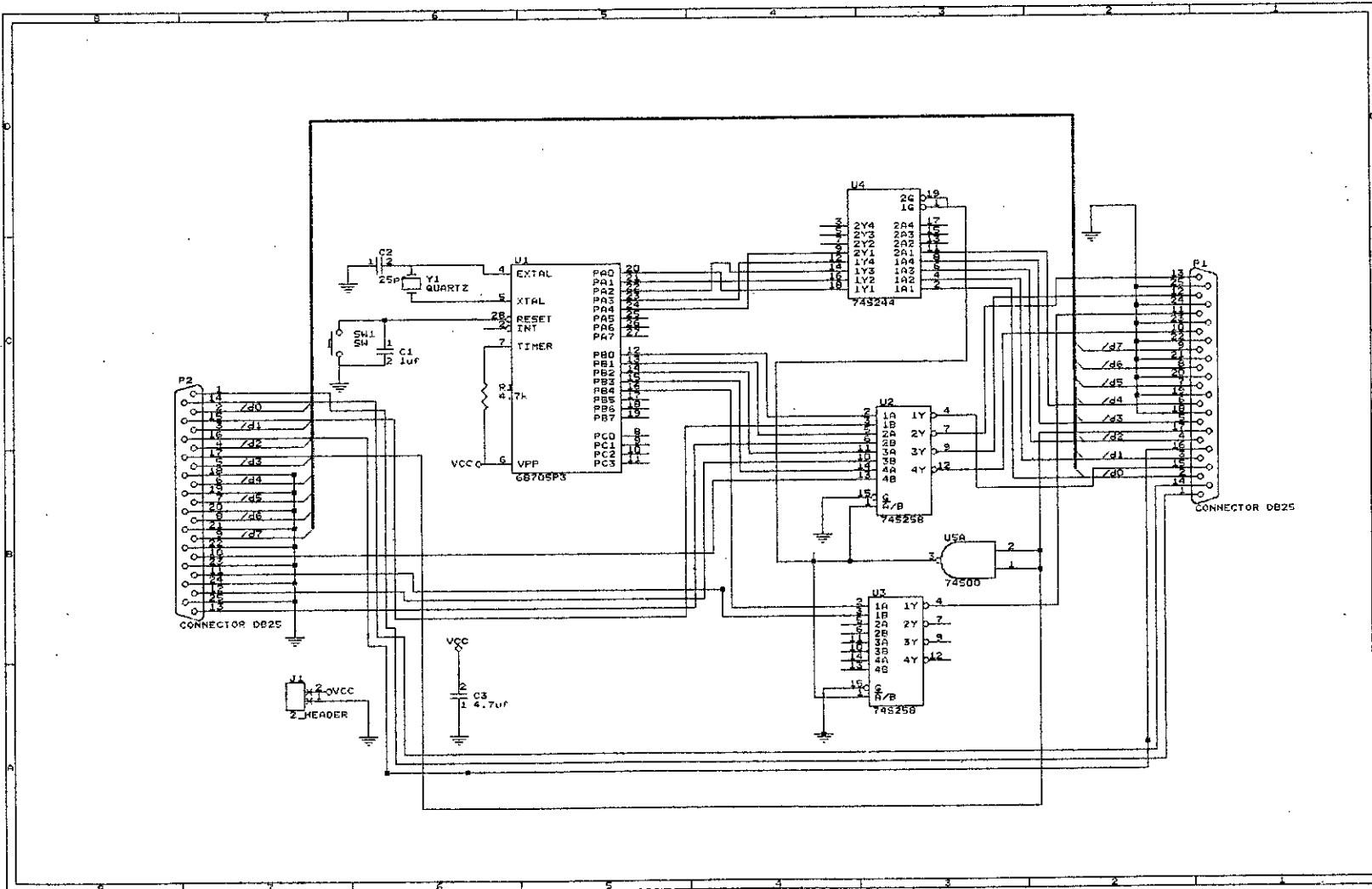
exemple ). Ainsi , le micro-contrôleur est capable de charger un programme dans sa RAM et l'exécuter.

#### **4.4.2 - Multiplexage**

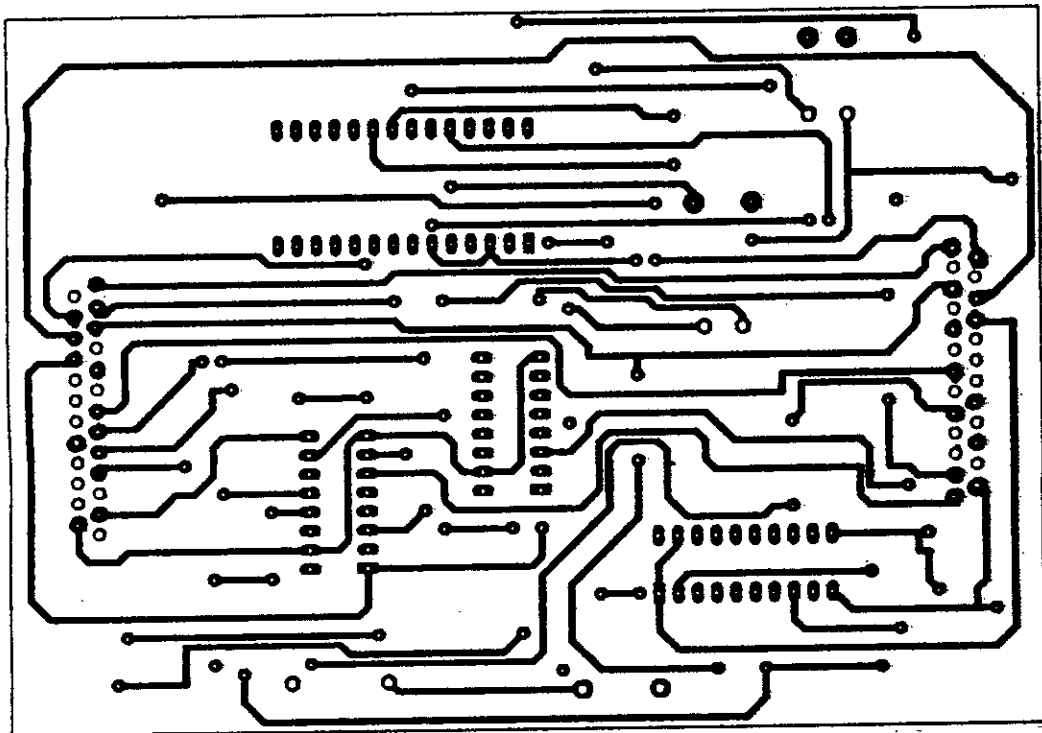
On doit faire un multiplexage entre le bus de sortie du micro-contrôleur et l'état de l'imprimante, ceci peut être fait en utilisant les deux circuits 74LS257.

Le Buffer 74LS244 (Fig.3.13) est également conseillé pour la protection du micro-contrôleur, ainsi que pour l'isolement des deux cas de fonctionnement : en imprimante, et en Dongle.

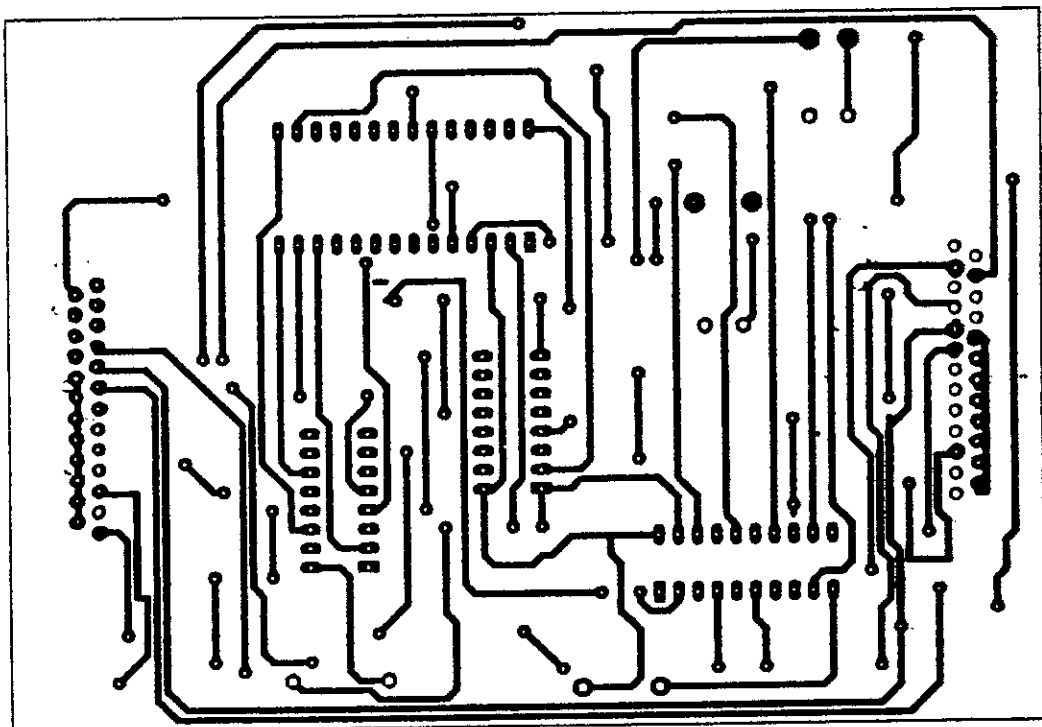
La commande de ces circuits est effectuée à l'aide de ligne Select in ( §2 ) qui est à "1" lorsque l'imprimante est sélectionnée. Il est à noter que l'utilisation de la porte "NOR" est indispensable pour le bon fonctionnement de la carte vu la logique négative de ce pin.



#### 4.5 Réalisation du circuit



Face : 1



Face : 2

Fig 3.13 - Circuit imprimé.

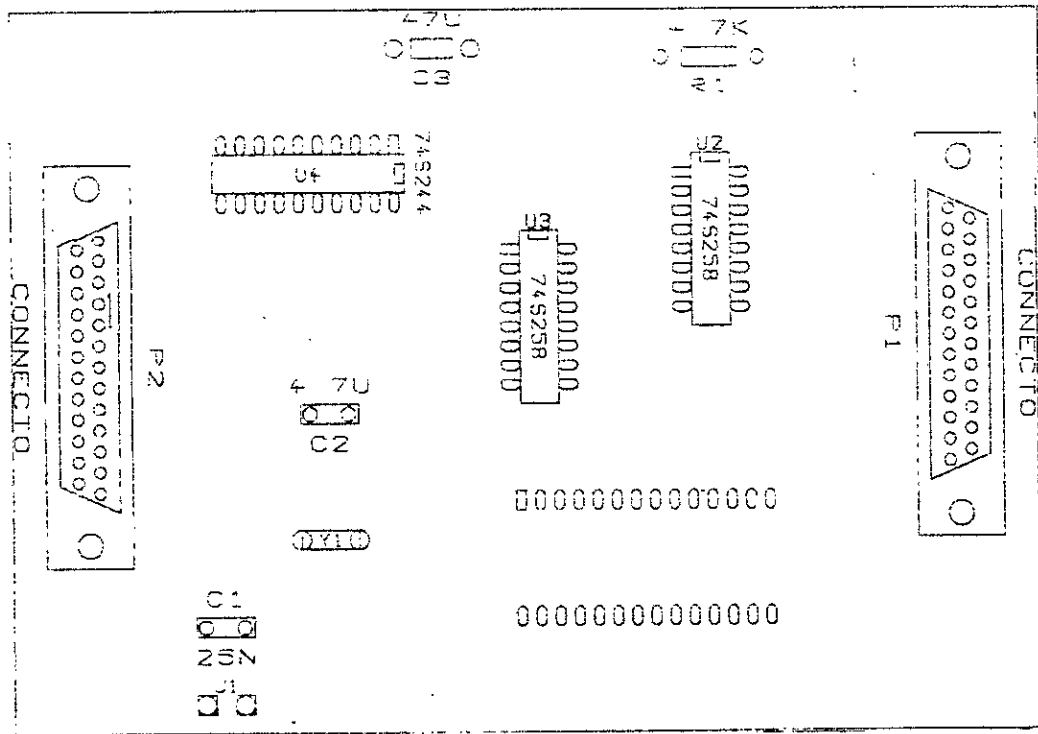


Fig.3.14 - Emplacement des composants.

Liste des composants :

N°	Composant	Nom
1	1uf	C1
2	25pf	C2
3	2_HEADER	J1
4	4.7k	R1
5	4.7uf	C3
6	68705P3	U1
7	74S00	U5A
8	74S244	U4
9	74S258	U2
10	74S258	U3
11	CONNECTOR DB25	P1
12	CONNECTOR DB25	P2
13	QUARTZ	Y1
14	SW	SW1

#### **4.6 - Elaboration du logiciel**

Pour profiter au maximum des capacités du microcontrôleur il est possible de charger dans sa RAM, un programme qui sera exécuté par ce dernier.

Pour renforcer la protection par cette technique, il est possible de graver dans l'EPROM du microcontrôleur un "numéro de série" qui sera vérifié durant l'exécution du programme à charger dans la RAM.

##### **i) Programmation du microcontrôleur**

Le programme à charger dans l'EPROM du microcontrôleur est constitué des procédures suivantes (Fig 3.15.a) :

- Configuration des ports (Port A, Port B, Port C).
- Initialisation de la transmission.
- Réception d'un octet (Parallel byte reception).
- Chargement d'un octet (Parallel byte emission).
- Chargement du programme dans la RAM et son exécution.
- Emission des données cryptées.

##### **ii) Programmation du micro-ordinateur**

Le programme à exécuter par le micro-ordinateur, qui sera considéré comme étant la procédure de test dans la protection du logiciel, est constitué de (Fig 3.15.b) :

- Initialisation du port parallèle et de la carte (  $\overline{\text{Select in}} = 0$  ).
- Initialisation de la transmission.
- Emission des données et du programme.
- Réception des données cryptées.

L'émission et la réception, sont faites en utilisant les procédures "EmetOctet" et "RecOctet".

ParallelByteReception

```

; ---- Quartet infrieur -----
04 02 FD BCLK1 brset 4,PortA,BCLK1

B6 00 SUIK1 lda PortA
A4 0F and #$0F
B7 10 sta Mem1 ; Rception des bits 0..3
17 02 bclr 4,PortB ;envoié d'un '0' (accus de rception)
; ---- Quartet Suprieur -----
05 02 FD BCLK2 brclr 4,PortA,BCLK2

B6 00 SUIK2 lda PortA
48 lsra
48 lsra
48 lsra
48 lsra
BA 10 ora Mem1
B7 10 sta Mem1 ;Mem1 := Mem1 OR A.
16 02 bset 4,PortA ;Envoié d'un '1' (Accus de Rception)
81 rts

```

ParallelByteEmission

```

; ---- Quartet infrieur -----
A6 0F lda #$0F ; slection du quartet inferieur
B4 11 and Mem2 ;
B7 00 sta PortB ; mettre le quartet inf.
17 02 bclr 4,PortB ; Envoié d'un '0' vers /busy du PC
04 02 FD BCLJ1 brset 4,PortA,BCLJ1

; ---- Quartet Suprieur -----
A6 F0 lda #$F0
B4 11 and Mem2 ; Slection du Quartet Suprieur
lsra
lsra
lsra
lsra
B7 00 sta PortB
16 02 bset 4,PortB ;envoié d'un '1' vers /busy du PC

05 02 FD BCLJ2 brclr 4,PortA,BCLJ2
81 rts

```

```
function EmetOctet( Wert : byte ) : boolean;
var Retour : byte;                                { Octet rceptionn }
begin
  (-- Emet le quartet infrieur -----)
  emetocet:=false;
  TO_Count := Timeout*5;                          { Initialise le compteur de Time Out }
  PutB( Wert and $0F );                            { Envoi avec mise 0 de BUSY }
  while ( ( ( GetB and 128 ) = 0 ) and ( TO_Count > 0 ) ) do
  dec(to_count) ;
  if ( TO_Count = 0 ) then exit;                    { Erreur de Time Out ? }
    { Interrompt la transmission }

    { Bits 3-6 en 0-3 }

  (-- Emet le quartet suprieur -----)
  TO_Count := Timeout*5;                          { Initialise le compteur de Time Out }
  PutB( ( Wert shr 4 ) or $10 );                    { Envoie avec mise 1 de BUSY }
  while ( ( ( GetB and 128 ) <> 0 ) and ( TO_Count > 0 ) ) do
  dec(to_count);
  if ( TO_Count = 0 ) then                          { Erreur de Timeout }
    exit; { Interrompt la transmission }

    { Bits 3-6 en 4-7 }
  EmetOctet :=true; { Octet correctement transmis ? }
end;
```

```
function RecOctet : byte;
var LoNib,
  HiNib : byte;                                { Quartets reus }
begin
  (-- Rceptionne et renvoie le quartet infrieur -----)
  TO_Count := Timeout*5;                          { Initialise le compteur de Time Out }
  while ( ( ( GetB and 128 ) = 0 ) and ( TO_Count > 0 ) ) do
  dec(to_count) ;

  if ( TO_Count = 0 ) then                          { Erreur de Time Out ? }
    exit; { Interrompt la transmission }

  LoNib := ( GetB shr 3 ) and $0F;                  { Bits 3-6 en 0-3 }
  PutB( LoNib );                                    { Retour l'expditeur }

  (-- Rceptionne et renvoie le quartet suprieur -----)
  TO_Count := Timeout*5;                          { Initialise le compteur de Timeout }
  while ( ( ( GetB and 128 ) <> 0 ) and ( TO_Count > 0 ) ) do
  dec(to_count) ;
```



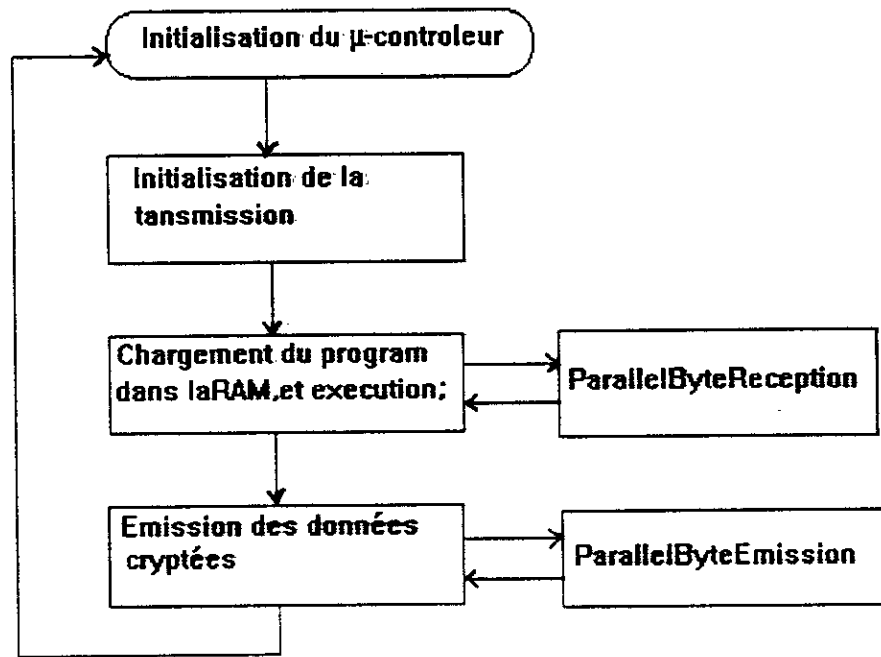


Fig.3.15.a - Organigramme du programme du micro-contrôleur.

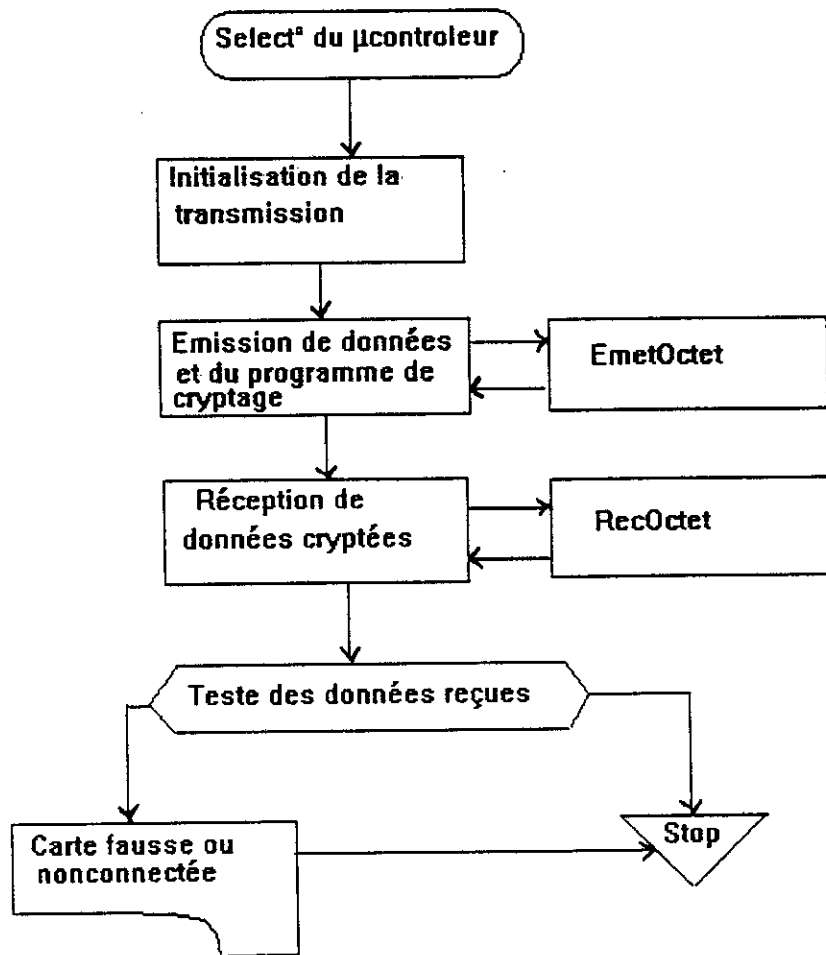


Fig.3.15.b - Organigramme du programme du PC.

## CONCLUSION

Bien que la protection par "dongle" est efficace , elle présente certains inconvénients : l'installation de la "clé" sur le port parallèle limite son utilisation ; surtout si on installe plusieurs dongles simultanément sur le même port. De plus, un dongle augmente le coût de protection.

Toutefois, le "dongle" reste utiliser, surtout, par les développeurs de logiciels de diffusion relativement restreinte; qui trouvent là une protection de mise en oeuvre rapide.

Le modèle réalisé dans ce travail peut être considéré comme un système de communication avec PC, à caractère général. Bien entendu, il peut être utilisé pour protéger plusieurs logiciels à la fois, en effectuant la tâche logicielle nécessaire et tenant compte de l'octet gravé dans l'EPROM (numéro de série).

## CONCLUSION GENERALE

L'étude de la cryptographie nous a permis de voir les méthodes utilisées pour garantir "l'authenticité" et la "confidentialité" de l'information, ainsi que d'introduire le sujet des "Systèmes Cryptographiques à Clefs Publiques".

L'implémentation des méthodes de protection par programmation nous a ouvert des fenêtres sur la programmation système à savoir : l'accès aux interfaces par le BIOS, le détournement des interruptions, ... ainsi que la connaissance de l'organisation des informations sur les supports ( Disquettes et Disques Durs .... ).

Ces méthodes sont très efficaces pour un utilisateur moyen, elle sont donc très utilisées, puisqu'elles ne demandent pas un "add-on" matériel. Mais, la possibilité pour un utilisateur plus expérimenté de les détourner, nous oblige à renforcer la protection.

La méthode de protection matérielle développée ici n'est qu'un modèle expérimental, il a été noté qu'il existe d'autres procédés plus fiables telle que l'utilisation d'une carte à travers les slots de la carte mère.

La réalisation de notre carte nous a conduit à résoudre pas mal de problèmes,

à savoir :

- Etude de l'interfaçage avec le P.C. En particulier à travers le port parallèle.
- Etude de la méthodologie générale de la réalisation d'une carte d'interface.
- Réalisation d'une liaison entre deux P.C., par l'intermédiaire d'un câble " Null-Modem" et le protocole "LAPLINK".
- Réalisation d'une liaison entre le P.C. et micro-contrôleur avec le même protocole ( "LAPLINK" ).

Le sujet abordé reste ouvert, notamment en ce qui concerne :

- L'implémentation directe de la protection sur un exécutable.
- Le renforcement de la protection contre les méthodes de "déprotection", qui ont pour but de court-circuiter les procédures de test, et cela en se basant sur les techniques de "tracing" du côté "SOFT", et par la détection des outils d'analyse logiques du côté "HARD".
- La réalisation d'autres procédés de protection matérielle, à travers la liaison série ou les slots de la carte-mère.

## BIBLIOGRAPHIE

- [ 1 ] Jean - Pierre TUAL, " cryptographie", Techniques de l'ingénieur, traité informatique, 1991.
- [ 2 ] HENK C. A , VAN TILBORG, "An introduction to cryptology", 1988.
- [ 3 ] Jeffroy BEAUQUIER, Béatrice BERNARD, "Systèmes d'exploitation - concepts et algorithmes".
- [ 4 ] Guillaume DE BREBISSON, " Programmation système".
- [ 5 ] F. PIEROT, "Guide PSI du développeur sous MS-DOS", 1989.
- [ 6 ] Lionel FOURNIOUX, "MS-DOS approfondi", Berti Edition, 1993, 255p.
- [ 7 ] Philippe FANGERAS, "Périphériques: Interfaces et technologie", Edition fréquences, 1995.
- [ 8 ] M. TISHER, "La bible P.C.", 1995.
- [ 9 ] Mr BOUROUBI, S. CHARA, "Etude et réalisation d'un système de liaison sans câble entre micro-ordinateur et périphériques (Liaison parallèle et liaison série)", PFE. ENP, 1995.
- [ 10 ] S. MERROUCHE, R. NAILI, "Conception et réalisation d'un lecteur/programmeur de cartes à puce", PFE. USTHB, 1995.
- [ 11 ] N. DJIDI, S. HADJOU DJ, "Commande d'un héliostat par micro-contrôleur", PFE. ENP, 1995.
- [ 12 ] S. LEIBSON, "Manuel des interfaces", McGRAW HILL , 1984.
- [ 13 ] M. MESSUD, " La pratique du microprocesseur; Conception des applications", Berti, 1992, 300p.
- [ 14 ] C. TAVERNIER, " Microcontrôleur 6805 et 68HC05", DUNOD, 1993.

**ANNEXE**

**LE MICRO-CONTROLEUR 68705 P3**

**PROGRAMME DE FORMATAGE**

## LE MICROCONTROLEUR 68705 P3 :

Le micro-contrôleur intègre dans le même boîtier un microprocesseur, une EPROM de 1796 octets utilisateur, une RAM de 112 octets utilisateur, deux ports d'entrées / sorties de huit bits chacun, un port d'entrées / sorties de quatre bits et enfin un Timer , c'est à dire un compteur chronomètre. Le tout dans un boîtier de 28 pattes.

Le 68705 P3 forme un ensemble d'une grande facilité de mise en oeuvre, sans bus de données, ni bus d'adresses puisque seules les broches nécessaires à l'application sont accessibles. L'outil est idéal pour les applications de petites tailles. En supplément, il contient tout ce qu'il faut pour programmer son EPROM, à condition de faire un petit montage pour recopier le contenu d'une mémoire externe contenant le programme.

### 1-BROCHAGE [ 13 ]

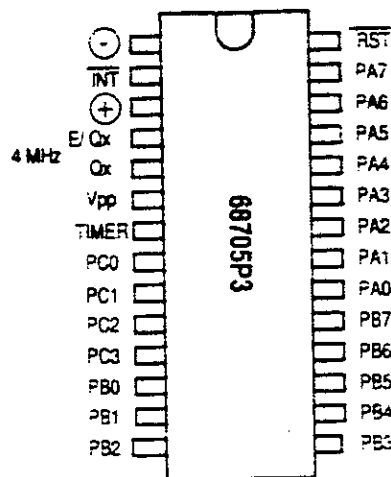


Fig.A.1 - Brochage du MC 68705 P3.

Reset : Quand cette broche est portée au niveau bas puis libérée le micro-contrôleur effectue sa procédure interne d'initialisation, et commence à travailler selon le programme contenu dans son EPROM.

La résistance de " Pull-up " est intégrée dans le boîtier.

La broche RESET doit être reliée à la masse à travers un condensateur de 1  $\mu$ F.

Le vecteur Reset se trouve aux adresses 7FE et 7FF.

A l'initialisation, les 20 broches de l'ensemble des ports sont automatiquement programmées en entrées.

INT : Requête d'interruption matérielle. Portée au niveau bas, elle génère une procédure classique d'interruption masquable. Le vecteur interruption se trouve aux adresses 7FA et 7FB.

Le masque d'interruption est mis à zéro au reset.

Quartz : Les broches  $Q_x$  sont reliées au quartz, 4 Mhz au maximum. La broche E/  $Q_x$  peut se relier à la masse à travers un condensateur de 27 picofarads pour une meilleure stabilité. La broche E peut servir d'horloge pour des composants extérieurs.

Vpp : Cette broche doit être reliée directement au +5V. Elle n'est utilisée que pour la programmation de l'EPROM interne.

PA0...PA7, PB0...PB7, PC0...PC7 :

Ce sont les ports internes, ils se programment et s'utilisent comme le port du 6821, mais plus simplement car les DDR et OR ont chacun leur adresse :

Registre	Adresse
ORA	000
DDRA	004
ORB	001
DDRB	005
ORC	002
DDRC	006



Timer : Cette broche peut s'employer comme entrée pour contrôler le Timer. Elle est utilisée aussi lors de la programmation de l'EPROM.

## 2-LA MEMOIRE [ 14 ]

La mémoire se divise en deux groupes : la RAM, et l'EPROM. Certaines adresses sont réservées à l'utilisateur, d'autres aux ports et au timer; et enfin certaines adresses appartiennent au contrôleur lui même, en particulier la portion 785 à 7F7.

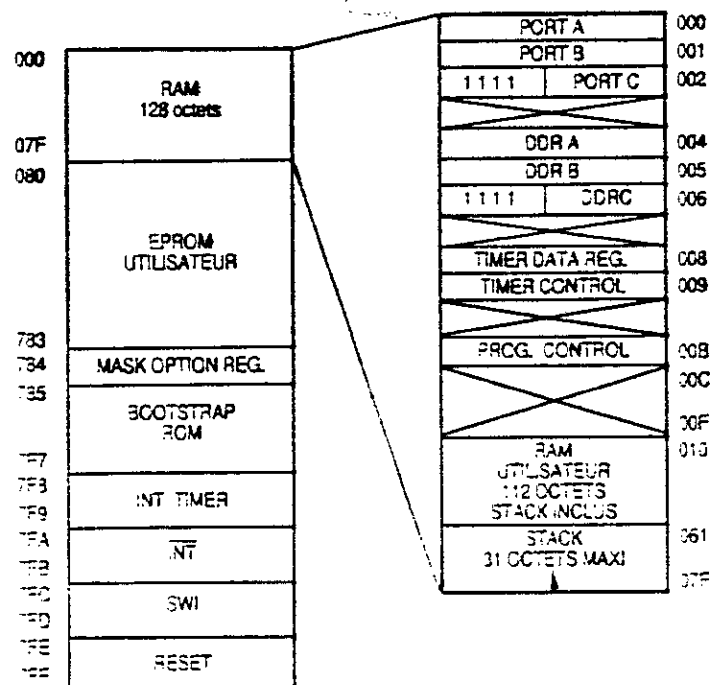


Fig.A.2 - " Map Memory " du MC 68705 P3

MASK option Register ( MOR ) : Registre des Options .

Il se situe à l'adresse hexadécimale 784 dans l'EPROM. Les bits écrits dans ce masque seront pris en compte au Reset.

CK	TOPT	CKS	TIE	X	P2	P1	P
----	------	-----	-----	---	----	----	---

CK : détermine la nature de l'horloge externe.

CK = 0 → Quartz , CK = 1 → Réseau RC.

TOPT : Timer Option, détermine si le Timer est programmable ou non.

TOPT = 0 : Timer programmable.

TOPT = 1 : Timer non programmable.

CKS : Origine de l'horloge du Timer. Cette horloge peut provenir de deux sources; interne qui divise la fréquence du Quartz par 4, ou externe venant de la broche Timer.

TIG : ( Timer input Enable ) : Validation de l'entrée du Timer.

TIE = 0 → Entrée inhibée

TIE = 1 → Entrée validée

P2 P1 P0 : fixe l'échelle de division de l'horloge du Timer pour le comptage interne fixant le temps.

P2	P1	P	DIV
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

### 3-JEU D'INSTRUCTION DU 68705 P3 [ 13 ]

Le jeu d'instruction de ce micro-contrôleur est riche et pratique.

- ADC : Addition avec retenue.
- ADD : Addition.
- AND : ET logique.
- ASL : Décalage arithmétique à gauche.
- ASR : Décalage arithmétique à droite.
- BCC : Branchement si la retenue est nulle .
- BCLn : Mise à zéro d'un bit.
- BCS : Branchement si la retenue est à 1 .
- BEQ : Branchement si égal.
- BHCC : Branchement si la demie retenue est nulle.
- BHCS : Branchement si la demie retenue est égale à 1.
- BHI : Branchement si plus grand.
- BHS : Branchement si plus grand ou égal.
- BIH : Branchement si INT est à 1.
- BIL : Branchement si INT est à 1.
- BIT : Test mémoire BIT à BIT.
- BLO : Branchement si plus petit.
- BLS : Branchement si plus petit ou égal .
- BMC : Branchement si masque d'interruption à 0.
- BMI : Branchement si négatif.
- BMS : Branchement si masque d'interruption à 1.
- BNE : Branchement si différent.
- BPL : Branchement si positif.
- BRA : Branchement inconditionnel.
- BRCLn : Branchement si Bit n à 0.
- BRN : Ne branche jamais.
- BRSETn : Branchement si bit n à 1.
- BSETn : Mise à 1 d'un bit.
- BSR : Branchement à un sous-programme.
- CLC : Mise à 0 du bit c du CCR.

CLI : Mise à 0 du bit I du CCR.  
CLR : Mise à 0.  
CMP : Comparaison accumulateur et mémoire.  
COM : Complément logique.  
CPX : Comparaison index et mémoire.  
DEC : Diminue de 1.  
EOR : " On exclusif " logique.  
INC : Augmente de 1  
JMP : Saut incondtionnel.  
JSR : Saut à 1 sous-programme.  
LDA : Chargement de l'accumulateur.  
LDX : Chargement de l'index.  
LSL : Décalage logique à gauche.  
LSR : Décalage logique à droite.  
NEG : Change le signe.  
NOP : Pas d'opération.  
ORA : " On inclusif " logique avec A.  
ROL : Rotation à gauche.  
ROR : Rotation à droite.  
RSP : Inialisation du pointeur de pile.  
RTI : Retour d(interruption.  
RTS : Retour de sous-programme.  
SBC : Soustraction avec retenue.  
SEC : Mise à 1 du bit " C " du CCR.  
SEI : Mise à 1 du bit " I " du CCR.  
STA : Mise en mémoire de l'accumulateur.  
STX : Mise en mémoire de l'index.  
SUB : Soustraction.  
SWI : Interruption logicielle.  
TAX : Transfert de A dans X.  
TST : Teste si négatif ou nul.  
TXA : Transfert de X dans A.

N°	Lecture		Ecriture		Lecture		Ecriture		Lecture		Ecriture		N°
	Op	Ad	Op	Ad	Op	Ad	Op	Ad	Op	Ad	Op	Ad	
0	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	0
1	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	1
2	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	2
3	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	3
4	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	4
5	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	5
6	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	6
7	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	7
8	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	8
9	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	9
A	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	A
B	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	B
C	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	C
D	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	D
E	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	E
F	BSTO	BSC	BNA	BIL	MTC	DM	MTC	DM	MTC	DM	MTC	DM	F

Abreviations des modes d'adressage

- IM4 : Immediat
- IMU : Immédiat
- DIF : Direct
- EXT : Étendu
- REL : Relatif
- BSC : Mode positionnement de bit
- BTB : Mode test de bit et branchement
- IX : Indirect, offset nul
- IX1 : Indirect avec offset 8 bits
- IX2 : Indirect avec offset 16 bits

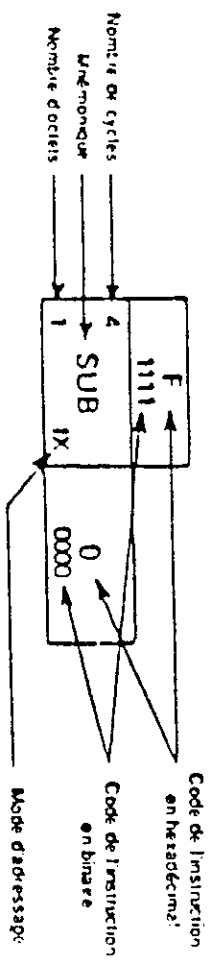


Fig A.3 - Jeu d'instruction du MC 68705 P3

#### **4-PROGRAMMATION DU 68705 P3 [ 11 ]**

En utilisant un montage, Fig 1, fournie par le constructeur, et en suivant son mode d'emploi soigneusement, le programme BOOTSRAPE recopiera fidèlement le contenu d'une EPROM 2716 dans l'EPROM du micro-contrôleur.

##### **LE MODE D'EMPLOI**

- 1 - Alimentation non connectée, S<sub>1</sub> et S<sub>2</sub> fermées.
- 2 - Positionner le micro-contrôleur et l'EPROM à recopier sur leurs supports, sans toucher à S<sub>1</sub> et S<sub>2</sub>.
- 3 - Connecter l'alimentation.
- 4 - Ouvrir S<sub>2</sub> et ensuite S<sub>1</sub> . La programmation s'effectue et dure un peu plus d'une minute. Les LEDS sont éteintes pendant ce temps. Lorsque la programmation est achevée, le LED 1 s'allume et le micro-contrôleur a comparé le contenu de son EPROM à celui de la mémoire extérieure à titre de vérification. Si la vérification donne un résultat positif, la LED 2 s'allume à son tour.
- 5 - Fermer S<sub>2</sub> et ensuite S<sub>1</sub> .
- 6 - Déconnecter les alimentations.
- 7 - Retirer le micro-contrôleur et la mémoire.
- 8 - Fin.

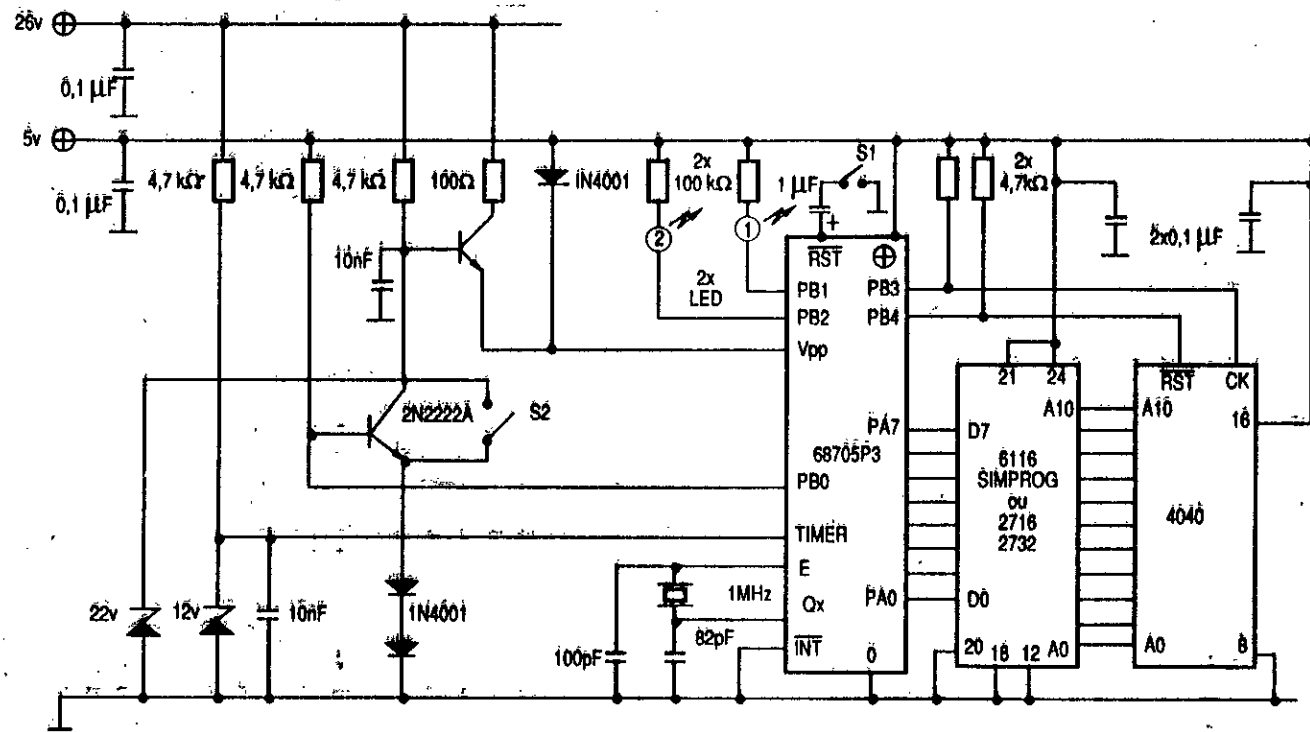


Fig.A.4 - Schéma électronique du programmeur du MC 68705 P3





```

with Regs do
begin
  ah := 5;           { Num,ro de fonction pour appel interruption }
  al := Nombre;     { Nombre de secteurs par piste }
  es := Seg( Champdonn ); { Adresse du Champ de donn,es }
  bx := Ofc( Champdonn ); { dans les registres es:bx }
  dh := Face;      { Num,ro de la face }
  dl := Lecteur;   { Num,ro de lecteur }
  ch := Piste;     { Num,ro de piste }
end;
intr( $13, Regs ); { Appel d'une interruption du BIOS }
if ( Regs.flags and fcarry = 1 ) then { Erreur? }
formattrack := 100
else
Formattrack := Regs.ah; { Lite ,tat erreur }
end;
function WriteTrack( Lecteur, Face, Piste,
Start, Nombre :byte;
Var Buffer ) : byte;
var
  Regs : Registers; { Registre processeur pour appel interruption }
begin
  with Regs do
  begin
    ah := $03;      { Num,ro de fonction pour appel interruption }
    al := Nombre;   { Nombre Secteurs par Piste }
    ch := Piste;    { Num,ro de Piste }
    cl := Start;    { Commencer par le secteur 1 }
    dl := Lecteur;  { Num,ro de lecteur }
    dh := Face;     { Num,ro de la face }
    es := Seg( Buffer ); { Adresse pour tampon }
    bx := Ofc( Buffer );
  end;
  intr( $13, Regs ); { Appel d'une interruption du BIOS }
  if ( Regs.flags and fcarry = 1 ) then { Erreur? }
  writetrack := 100
  else
  WriteTrack := Regs.ah;
  end;
end;
function PhysicalFormat( Drive : byte;
PData : PhysDataType;
Verify : boolean ) : boolean;
var
  Regs : Registers; { Registre processeur pour appel interruption }
  Piste,
  Face,
  Status : byte;    { Valeur de retour des fonctions appellees }
begin
  {-- Formatage de la disquette piste par piste -----}
  for Piste := 0 to PData.Pistes - 1 do { Ecrire toutes les pistes }
  for Face := 0 to PData.Faces - 1 do { Ecrire toutes les faces }
  begin
    Write( $13'Piste: ', Piste : 3, ' Face: ', Face : 2 );
    {-- 5 essais au maximum pour formater une piste -----}
    Status := FormatTrack( Drive, Face, Piste, PData.Secteurs );
    if status = 0 then physicalformat := true
    else physicalformat :=false;
  end;
end;
end;
function LogicalFormat( Drive : byte;
PData : PhysDataType;
LData : LogDataType ) : boolean;
var Status : byte; { Retour de la fonction appellee }
TousSecteurs : word; { Nombre total de secteurs }
i : byte; { Compteur d'it,rations }
AktSector,
AktSide,
AktTrack : byte;
Nombre : integer; { Nombre des secteurs restant ... ,crire }
TamponPiste : PisteBufType; { M,moire pour une piste }

```



```

Begin
  WriteLn( 'formater - 1986 - LABC 11 - E.N.P. Alger' );
  param:= '1440';
  aktDrive:= 0 ;
  if GetFormatParameter( param, 4,PData, LData)
  then
    begin
      ( Format et lecteur sont compatibles )
      GetIntVec( $IE, AncDDPT );      ( Stocker anc DDPT )
      SetIntVec( $IE, PData.DDPT );  ( D, finir nouvelle DDPT )
      ok := PhysicalFormat( AktDrive, PData,true);
      if ok then
        begin
          WriteLn(#13'Ecriture du secteur de boot et des FAT ');
          ok := LogicalFormat( AktDrive, PData, LData );
        end;
        {-- Evaluation du formatage -----}
        if ok then
          WriteLn( #13'Formatage ok. ');
        else
          begin
            WriteLn( #13'Une erreur a interrompu le formatage' );
            SetIntVec( $IE, AncDDPT ); { Restorer anc DDPT }
          end;
        end
      else writeLn('insérer une disquette 1.44K et re,essayer');
    end.

```