Chapitre 10

Le lecteur de disquette

Nous avons vu comment accéder à la mémoire, comment saisir au clavier et comment afficher à l'écran. Il reste à voir comment accéder à la mémoire de masse. Le premier IBM-PC utilisait un lecteur de cassettes numériques, très vite abandonné. IBM est ensuite passé au lecteur de disquette puis au disque dur.

Le lecteur de disquette est abandonné au profit des clés USB mais le disque dur existe encore. Pour des raions de compatibilité, une clé USB est formate comme une disquette ; les fonctionnalités de la routine de service du BIOS concernant le lecteur de disquette peuvent donc être utilisées pour une clé USB, même si le code de la routine proprement dit est revue. Partons donc du modèle de la diquette.

10.1 Description matérielle d'une disquette

Une **disquette** (en anglais *diskette* ou *floppy disk*), voir figure 10.1, est un support de stockage composé d'un disque, en fait une couronne circulaire, fin et flexible (d'où son nom de *disque souple* par opposition au *disque dur*) recouvert d'une couche magnétique, placé dans une enveloppe de plastique pour le protéger de la poussière. Une disquette est lue par une **lecteur de disquette** (en anglais FDD pour *Floppy Disk Drive*). Les premières disquettes (utilisées sur les ordinateurs IBM, pas sur des micro-ordinateurs) avaient un diamètre de 8 pouces (200 mm), puis sont apparues les disquettes de 5,25 pouces (133 mm), avec une enveloppe souple, puis de 3,5 pouces (89 mm), avec une enveloppe rigide.

FIGURE 10.1 – Disquettes de 8, 5.25 et 3.5 pouces

10.1.1 Les principes physiques de l'enregistrement magnétique

Cette section n'est pas indispensable pour la programmation, sauf en ce qui concerne le modèle de répartition des donnés. Le reste n'est donné qu'à titre informatif, par souci de complétude. On peut donc se dispenser de la lire si on préfère.

Les enregistrements magnétiques, que ce soient les magnétophones, les magnétoscopes, les cartes magnétiques, les disquettes ou les disques durs, reposent tous sur les mêmes principes physiques : le ferromagnétisme et l'induction.

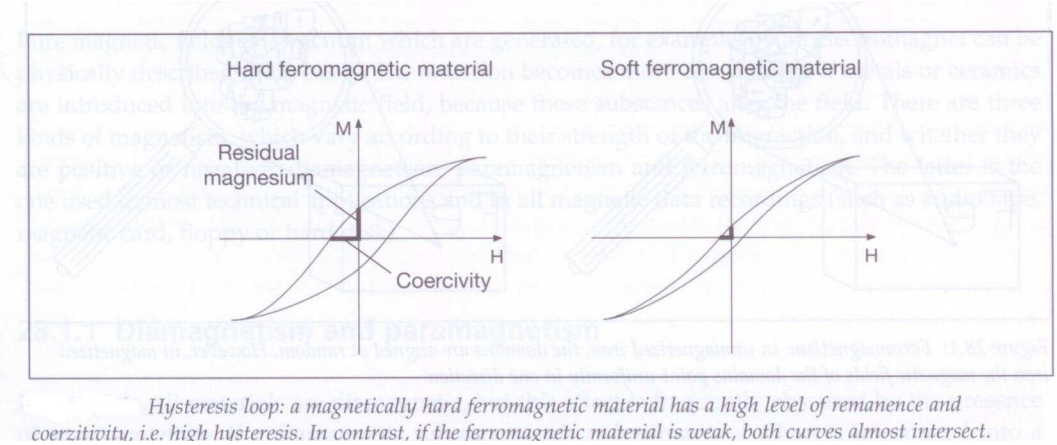
10.1.1.1 Le ferromagnétisme

Le champ magnétique engendré dans le vide par un électro-aimant se décrit facilement en Physique, synthétisé par les équations de Maxwell. La situation devient plus compliquée lorsque des métaux ou des céramiques sont introduits dans le champ magnétique, car ces substances altèrent le champ. Il existe alors trois types de magnétisme : le diamagnétisme, le paramagnétisme et le ferromagnétisme.

— Tous les matériaux son *diamagnétiques* : lorsqu'une substance, telle que l'hydrogène ou l'argent, est introduite dans un champ magnétique, elle affaiblit légèrement celui-ci (de l'ordre de 0,000 0001 % à 0,05 %). Cet affaiblissement ne dépend pas de la température. Pour la plupart des substances, ce phénomène est occulté par le paramagnétisme, plus puissant. On s'aperçoit que le diamagnétisme pur apparaît pour les atomes dont tous les électrons sont appariés, tels que les gaz rares ou les sels de métaux.

— Lorsqu'on introduit une subtance *paramagnétique*, par exemple l'aluminium ou l'oxygène liquide, dans un champ magnétique, elle renforce, cette fois-ci, celui-ci de l'ordre de 0,000 01 % à 0,05 % au lieu de l'affaiblir. On s'aperçoit qu'il s'agit d'atomes dont au moins un électron est non apparié. Cet effet s'accentue aux températures basses.

— Les substances *ferromagnétiques*, comme le fer, peuvent fortement amplifier le champ magnétique, par un facteur pouvant aller jusqu'à un million. Une autre caractèristiques des substances ferromagnétiques est qu'elles conservent un pouvoir magnétique (elles peuvent attirer de la limaille de fer) après avoir été introduites dans un champ magnétique ; on parle de **rémanence** magnétique.



Hysteresis loop: a magnetically hard ferromagnetic material has a high level of remanence and coerzitivity, i.e. high hysteresis. In contrast, if the ferromagnetic material is weak, both curves almost intersect.

FIGURE 10.2 – Le phénomène d'hystérésie

La figure 10.2 montre la relation entre le champ magnétique externe H et la magnétisation M de la substance ferromagnétique. La magnétisation M augmente avec le champ magnétique externe H jusqu'à une valeur de saturation Js. Lorsque le champ magnétique H décroît, la magnétisation M décroît également mais sans suivre la courbe d'augmentation. Lorsque H atteint 0, il reste une magnétisation rémanente. Cette magnétisation rémanente disparaît lorsqu'on applique un champ magnétique de sens contraire. La courbe de la figure 10.2 s'appelle la *boucle hystérésique*.

Les substances diamagnétiques et paramagnétiques ne génèrent pas de champ magnétique permanent ; elles ne conviennent donc pas à l'enregistrement des données à long terme. On se sert donc du ferromagnétisme. Des additifs spéciaux sont utilisés pour créer des substances ferromagnétiques ayant un fort niveau d'hystérésie, c'est-à-dire une rémanence forte et une haute coercivité.

La boucle d'hystérésie montre que, pour enregister des données, il faut utiliser un champ magnétique au moins égal à la coercivité. Par ailleurs, il ne faut pas que le champ soit trop fort pour ne pas magnétiser ou démagnétiser les zones proches, représentant d'autres données. Les substances ferromagnétiques les plus utilisées sont le fer, le cobalt, le nickel et leurs alliages.

10.1.1.2 L'induction

Le champ magnétique utilisé pour enregistrer des données est évidement fourni par un électro-aimant.

Pour lire les données, on utilise le phénomène de l'*induction*, c'est-à-dire le fait qu'un champ magnétique mobile crée du courant électrique.

10.1.2 Disquette

Les phénomènes de ferromagnétisme et d'induction sont utilisés depuis longtemps pour l'enregistrement du son (magnétophone). On utilise alors une bande magnétique se déplaçant devant une tête de lecture-écriture, qui détecte l'information stockée sur la bande ou écrit une nouvelle information sur celle-ci. On peut utiliser une bande magnétique pour conserver de l'information numérique mais le rembobinage de celle-ci a pour conséquence un temps d'accès très long, d'où l'intérêt des disquettes. Au lieu d'un long et fin ruban, une disquette est un disque de plastique recouvert, des deux côtés, d'une substance ferromagnétique, enfermée dans un étui protecteur en plastique.

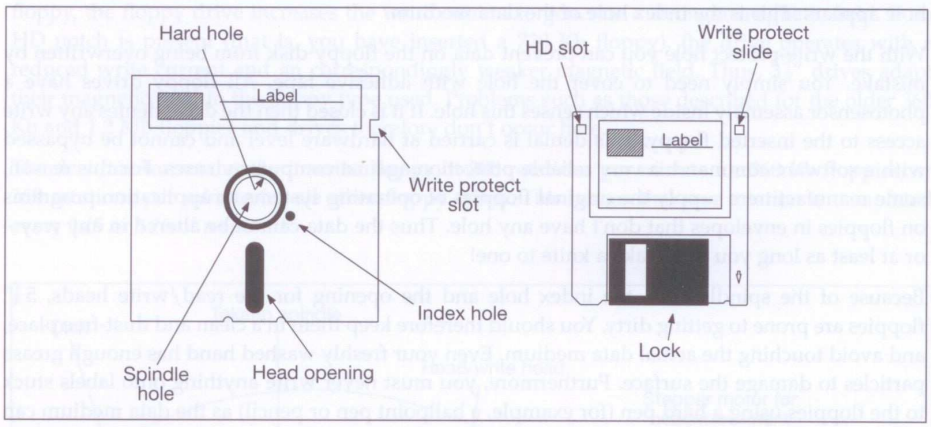


FIGURE 10.3 – Description des disquettes de 5,25 et de 3,5 pouces

La figure 10.3 montre l'essentiel ce qu'il y a à voir sur cet étui protecteur pour une disquette de 5,25 pouces :

— Au centre se trouve un trou (*spindle hole*) dans lequel viendra s'encastrer une broche permettant de faire tourner le disque, ce qui a à la fois pour effet de faire passer chaque zone aimantée devant la tête de lecture-écriture mais, également, par son mouvement, de créer le courant induit de lecture. On remarquera que le bord interne de la couronne circulaire est renforcé pour le protéger des nombreuses fois où il sera embrocher.

— Une ouverture le long d'un rayon (*head opening*) permet à la tête de lecture d'accéder à la couronne circulaire dans laquelle on pourra enregistrer les données. On perd un peu d'espace en haut et en bas de la couronne, entièrement caché dans l'étui protecteur.

— Un petit trou circulaire (*index hole*) dans l'enveloppe, en conjonction avec un petit trou circulaire dans la couronne circulaire, permet plus ou moins, lorsqu'ils se croisent, de définir l'origine des rayons de la couronne circulaire. Comme nous le verrons, il est beaucoup trop gros pour définir celle-ci de façon précise.

— Une encoche (*write protect slot*) permet de spécifier qu'on veut protéger le disque en écriture, lorsqu'elle est recouverte par un bout de ruban adhésif.

10.1.3 Lecteur de disquette

La figure 10.4 montre le principe d'un lecteur de disquette :

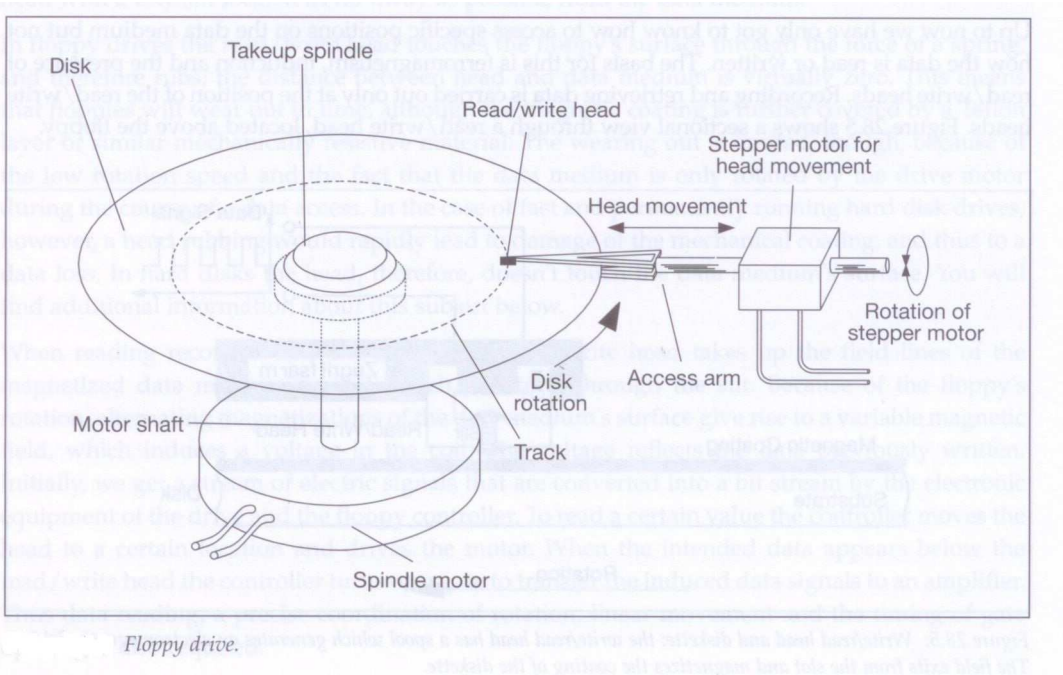


FIGURE 10.4 – Principe d'un lecteur de disquette

— La broche, permettant à la disquette d'être maintenue en place sur l'axe, tourne (*takeup spindle*) en étant reliée par un axe (*motor shaft*) à un moteur qui lui fait effectuer 300 ou 360 rotations par minute (RPM).

— Les deux têtes de lecture-écriture (une par face) se déplacent radialement grâce à un moteur pas à pas de façon à ce qu'elles puissent voir passer plusieurs cercles (appelés pistes) de la couronne circulaire. Elles sont situées à un même bras, une de chaque côté de la disquette, pas exactement au même emplacement pour éviter les interférences (l'écriture d'une face au lieu de l'autre). La même tête est utilisée à la fois pour l'écriture et la lecture alors qu'une tête plus large est utilisée pour effacer avant d'écrire.

— deux cellules photoélectriques permettent de détecter, pour l'une, le passage du trou d'index et, pour l'autre, si la disquette est protégée ou non en écriture.

10.1.4 Modèle de répartition des données sur une disquette

Sur un ruban magnétique, le modèle de répartition des données peut être simple : une fois qu'on a déterminé comment coder un bit, on a une suite linéaire de bits. Pour améliorer le temps d'accès, les données ne sont pas réparties de façon linéaire sur un disque :

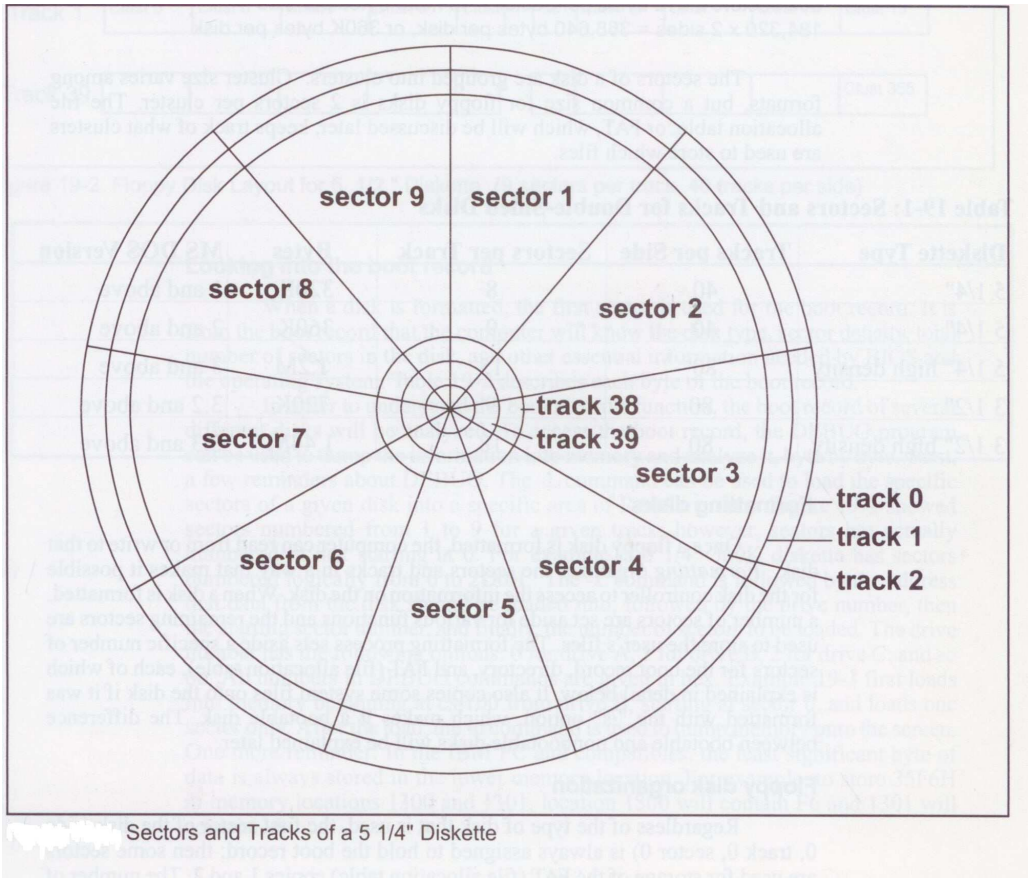


FIGURE 10.5 – Modèle de répartition des données sur une disquette

— Une **disquette** possde deux **faces** (en anglais *side* ou *surface*). Nous avons vu qu'il y a une tête de lecture-écriture par face.

— Chaque face contient un certain nombre de **pistes** (en anglais *track*), à savoir des cercles du même axe que la disquette. Ces pistes sont d'une certaine façon imaginaires : elles ne sont pas matérialisées. Les pistes sont numérotées à partir de 0, correspondant à la piste la plus externe. Une piste est décrite par la rotation de la disquette, une fois la tête de lecture-écriture positionnée par le moteur pas à pas.

— Chaque piste est divisée en un certain nombre de **secteurs** (en anglais *sector*), correspondant à un angle donné. Cet angle est $2\pi/n$, où n est un entier naturel non nul ; une piste a une longueur d'autant plus courte qu'elle est plus proche du centre. Là encore un secteur n'est pas matérialisé. Un secteur contient un nombre fixé d'octets, le plus souvent 512.

Le secteur est la plus petite unité d'information accessible physiquement sur un disque : on ne peut lire (ou écrire) les données que par bloc et non octet par octet. Lorsqu'on veut enregistrer un donnée d'un seul octet, on est obligé d'utiliser un bloc. Pour enregistrer un bloc il faut rechercher le début de celui-ci, ce qui demande un certain temps. Il faut donc trouver un compromis entre le gaspillage de l'espace (maximum lorsqu'un bloc est un secteur) et le temps d'accès (maximum lorsqu'un bloc est constitué d'un seul octet). Ce compromis est le secteur.

— Un **cylindre** (en anglais *cylinder*) est l'ensemble des pistes de même numéro. Un cylindre de disquette contient deux pistes.

10.1.5 Codage d'un bit sur une disquette

La rotation de la disquette permet de récupérer (ou d'écrire) des signaux le long de chaque piste. Il existe deux méthodes pour représenter un bit comme le montre la figure 10.6 :

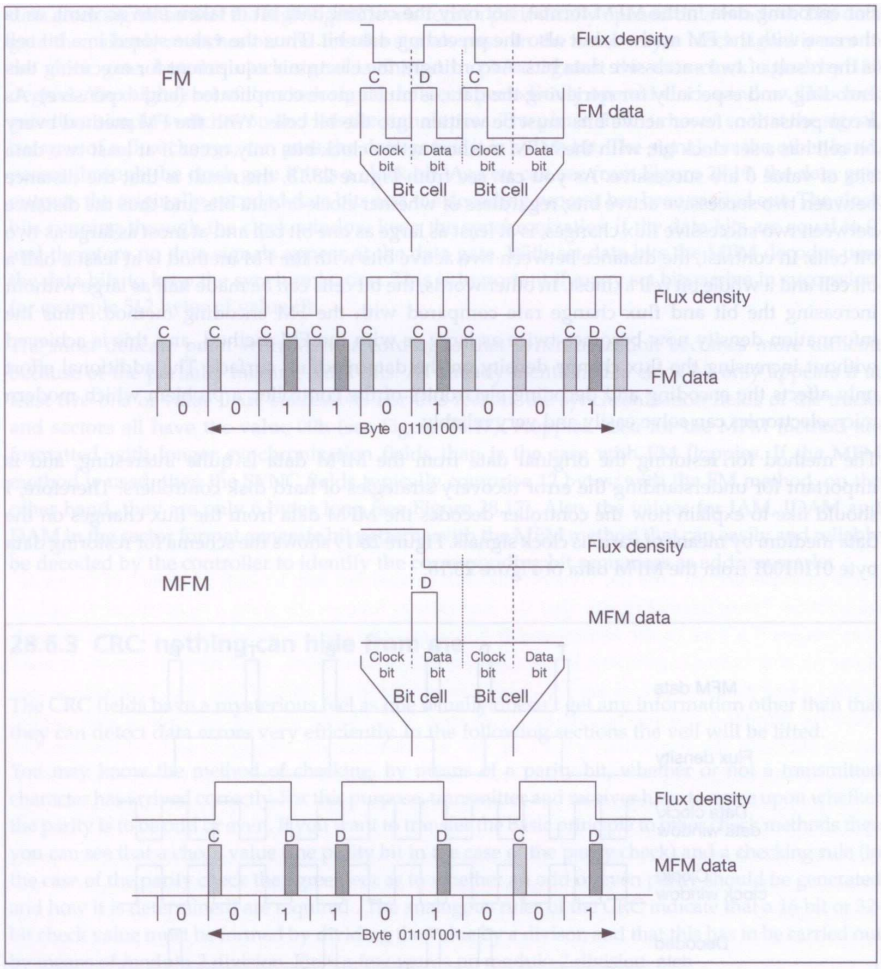


FIGURE 10.6 – Codage d'un bit sur une disquette

Méthode FM.- Dans le format d'enregistrement en modulation de fréquence (**FM** pour *Frequency Modulation*), un bit est stocké dans une **cellule de bit** (*bit cell* en anglais), divisée en deux parties, appelées **bit d'horloge** et **bit de donnée**. Ceci est dû au fait que seul un changement de flux sur le support magnétique est capable d'engendrer un signal : le bit d'horloge est toujours présent.

Pour écrire dans le format FM, on utilise un compteur d'horloge qui envoie une impulsion d'horloge tous les nombres impairs. Pour les nombres pairs, une impulsion est envoyée vers la tête d'écriture si le bit vaut 1 et rien s'il vaut 0.

Lors de la lecture, le circuit de lecture détecte le bit d'horloge et détermine s'il y a un signal actif, représentant la valeur 1 du bit de donnée, entre deux bits d'horloge successifs.

Méthode MFM.- Le format **MFM** (pour *Modified Frequency Modulation*) a été introduit par IBM en 1970 pour son lecteur de disque dur IBM 3330. Dans ce format :

- Le bit d'horloge n'est pas systématiquement présent. Il ne l'est que si les bits d'horloge et de donnée précédent ne le sont pas.
- Le bit de donnée est, comme pour FM, présent si le bit vaut 1.

La distance entre deux bits actifs est donc plus grande que dans le cas de FM. On a besoin de connaître deux cellules de bit successifs pour déterminer la valeur d'un bit. L'équipement électronique pour exécuter le codage et le décodage est beaucoup plus compliqué que dans le cas de FM mais on peut enregistrer une plus grande densité d'information avec la méthode MFM sur le même support. Cette dernière caractéristique fait que la méthode FM a très vite disparu.

10.1.6 Codage des secteurs

Il est facile de repérer une face (grâce à la tête de lecture-écriture correspondante) ou une piste (grâce au positionnement de la tête par le moteur pas à pas). Les bits d'une piste ne sont pas entièrement dédiés aux données. Une partie d'entre eux sert à structurer la piste en secteurs, comme le montre la figure 10.7. Décrivons ce codage dans le cas de MFM.

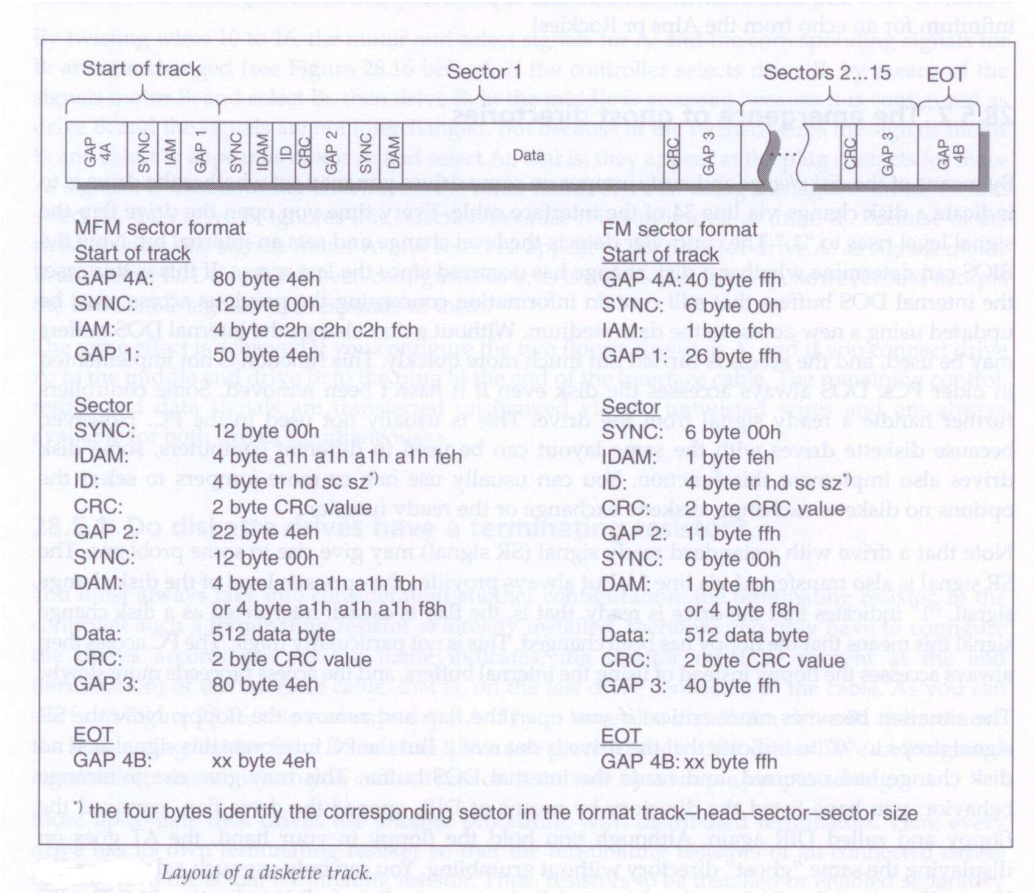


FIGURE 10.7 – Codage d'un secteur sur une disquette

Voyons d'abord comment on repère l'origine d'une piste, à l'aide de 146 octets :

— L'origine d'une la piste commence par le code **GAP 4A**. Il s'agit de 80 octets de valeur 4eh, valeur que l'on retrouve pour les autres GAP. Une disquette spécifie l'origine de la piste au moyen d'un trou d'index, mais sa position n'est pas suffisamment précise pour en déterminer exactement le début. Le motif GAP 4A informe le contrôleur du début de la piste et laisse suffisamment de temps à l'électronique pour être capable de répondre à ce signal.

— Le motif **SYNC**, constitué de 12 octets nuls, permet de synchroniser le décodeur du contrôleur avec la vitesse de rotation de la disquette. Les bits d'horloge de ce motif permettent de synchroniser l'horloge du décodeur, la vitesse de rotation de la disquette étant approximative.

— Le motif **IAM** (pour *Index Address Mark*), constitué de quatre octets de valeurs respectives c2h, c2h, c2h et fch, informe le contrôleur qu'on passe maintenant aux secteurs de la piste.

— Pour laisser le temps de se mettre en train, un motif **GAP1**, constitué de 50 octets de valeur 4eh, suit IAM.

— Les secteurs suivent, codés de la façon décrite ci-dessous.

— La piste se termine par un motif **EOT** (pour *End Of Track*), qui est un **GAP 4B**, de taille pas précisément définie, jouant le rôle de tampon.

Maintenant que nous avons vu comment l'origine de la piste est repérée, voyons comment on code un secteur, occupant 654 octets pour des données utiles de 512 octets :

— Tout secteur commence par un motif de synchronisation SYNC.

— La synchronisation est suivie par le motif **IDAM** (pour *ID Address Mark*), constitué de trois octets a1h et d'un octet feh, indiquant le début du champ d'identification du secteur concerné.

— L'identificateur **ID** est constitué de quatre octets spécifiant la piste, la tête, le secteur et la taille du secteur.

— Le contrôleur du lecteur de disquette calcule, au moment du formatage, à partir des quatre octets IDAM et de l'identificateur ID, un code de vérification CRC (pour *Cyclic Redundancy Check*) de taille deux octets, apparaissant tout de suite après ID. Son rôle est analogue au bit de parité en plus sophistiqué.

— Un motif **GAP 2** de 22 octets suit le CRC. Il laisse le temps au contrôleur, lors de la lecture, de vérifier le CRC.

— On synchronise à nouveau le contrôleur grâce au motif SYNC qui suit.

— Le motif **DAM** (pour *Data Address Mark*), constitué de trois octets a1h et d'un octet fbh ou f8h, indique le début des données.

— Les données occupent les octets suivants, au plus 512, la taille étant précisée dans ID.

— Au moment de l'écriture, le contrôleur calcule un CRC pour les données et en stocke la valeur juste après celles-ci.

— Le secteur se termine par le motif **GAP 3**, constitué de 80 octets de valeur 4Eh, servant de tampon élastique entre chaque secteur.

10.2 Contrôleur de lecteur de disquette

Le **contrôleur de disque** (en anglais *disk drive controller*, plus précisément FDC pour *Floppy Disk Controller* dans le cas d'un contrôleur de lecteurs de disquettes) est l'interface entre le microprocesseur et le disque. Le contrôleur reçoit, par exemple, l'information lui demandant de lire des données sur un secteur donné ; son rôle est alors de placer la tête de lecture-écriture au bon endroit, de lire les données et de les renvoyer au microprocesseur.

L'IBM PC utilise le μPD765 de NEC comme contrôleur. Le PC/AT peut être muni d'un contrôleur Intel 82072A alors que le PS/2 utilise un Intel 82077A. Décrivons le PD765.

10.2.1 Brochage du PD765

Les broches du PD765 sont indiquées sur la figure 10.8.

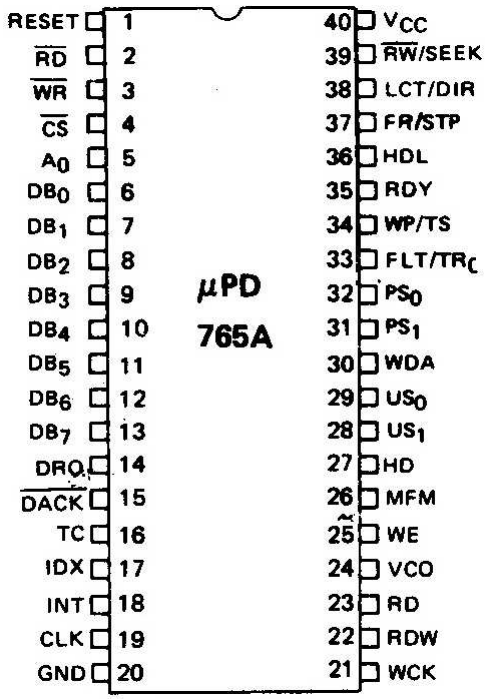


FIGURE 10.8 – Brochage du PD765 de NEC

Le symbole utilisé pour chaque broche et la description lapidaire de sa fonction apparaissent dans le tableau suivant :

No.	Symbole	Fonction
1	RESET	Reset input
2	$\overline{\text{RD}}$	Read control input
3	$\overline{\text{WR}}$	Write control input
4	$\overline{\text{CS}}$	Chip select input
5	A0	Data or status select input
6-13	DB0–DB7	Bidirectional data bus
14	DRQ	DMA request output
15	$\overline{\text{DACK}}$	DMA acknowledge input
16	TC	Terminal count input
17	INDEX	Index output
18	INT	Interrupt request output
19	CLK	Clock input
20	GND	Ground
21	WCLK	Write clock input
22	WINDOW	Read data window input
23	RDATA	Read data input
24	SYNC	VCO Sync output
25	WE	Write enable output
26	MFM	MFM output
27	SIDE	Head select output
28,29	US1,US0	FDD unit select output
30	WDATA	Write data output
31,32	PS1,PS0	Preshift output
33	FLT/TRK0	Fault/Track zero input
34	WRT/2SIDE	Write protect/two side input
35	READY	Ready input
36	HDLD	Head load output
37	FLTR/STEP	Fault reset/step output
38	LCT/DIR	Low current direction output
39	$\overline{\text{RW}}$/SEEK	Read/write/seek output
40	Vcc	DC power (+5V)

Donnons-en une description plus complète :

— RESET

L'entrée RESET place le FDD dans l'état d'attente (*idle state*). Il réinitialise les broches de sortie du FDC à 0, sauf PS0, prenant la valeur 1 et WDATA, indéfinie. Lorsque l'entrée RDY est maintenue à niveau haut durant la réinitialisation, le FDC engendre un signal de 1,024ms sur la broche INT. Pour effacer cette interruption il faut utiliser la commande « Sense Interrupt Status ».

— $\overline{\text{RD}}$ (*ReaD strobe*)

L'entrée RD est utilisé pour permettre le transfert de données depuis le FDC vers le bus des données, lorsqu'il se trouve à niveau bas et que soit $\overline{\text{CS}}$, soit $\overline{\text{DACK}}$ est à niveau haut.

— $\overline{\text{WR}}$ (*WRite strobe*)

L'entrée $\overline{\text{WR}}$ permet le transfert des données vers le FDC depuis les bus des données, lorsqu'il est à niveau bas.

— A0 (*Data/Status Select*)

L'entrée A0 sélectionne l'origine, registre des données (A0 = 1) ou registre de statut (A0 = 0), des données accéssibles par le bus des données.

— $\overline{\text{CS}}$ (*Chip Select*)

Le FDC est accessible lorsque $\overline{\text{CS}}$ est à niveau bas, ce qui rend actif $\overline{\text{RD}}$ et $\overline{\text{WR}}$.

— DB0–DB7 (*Data Bus*)

DB0–DB7 constitue un bus de données bidirectionnel de 8 bits. Il est désactivé lorsque $\overline{\text{CS}}$ est à niveau haut.

— DRQ (*Dma ReQuest*)

Le FDC place la sortie DRQ à niveau haut pour une requête de transfert DMA.

— $\overline{\text{DACK}}$ (*Dma ACKnowledge*)

Lorsque l'entrée $\overline{\text{DACK}}$ est à niveau bas, un cycle DMA est actif : le contrôleur est en train d'effectuer un transfert DMA.

— TC (*Terminal Count*)

L'entrée TC à niveau haut indique la fin d'un transfert DMA.

— INDEX

L'entrée INDEX est à niveau haut au début d'une piste.

— INT (*INTerrupt*)

La sortie INT spécifie une requête d'interruption de la part du FDC. En mode Non-DMA, le signal est émis pour chaque octet. En mode DMA, il est émis à la fin d'une opération.

— CLK (*CLocK*)

CLK est l'entrée du FDC pour un signal carré d'horloge, à niveau TTL et en simple phase à 8 Mhz ou à 4 Mhz.

— WCLK (*Write CLocK*)

L'entrée WCLK initialise le taux d'écriture des données du FDD. Il s'agit de 500 Khz pour les lecteurs FM, 1 Mhz pour les lecteurs MFM pour les opérations à 8 Mhz du FDC.

— WINDOW (*Read Data WINDOW*)

L'entrée WINDOW est engendrée par le PLL (*Phase-Locked Loop*). Elle est utilisée pour les données d'exemple du FDD et pour distinguer les bits de données et d'horloge du FDC.

— RDATA (*Read DATA*)

L'entrée RDATA permet de lire des données depuis le FDD, contenant des bits de données et d'horloge. Pour éviter un blocage, il faut utiliser ensemble RDATA et WINDOW.

— WDATA (*Write DATA*)

WDATA est la sortie horloge et données vers le FDD.

— WE (*Write Enable*)

La sortie WE permet d'écrire des données dans le FDD.

— SYNC (*VCO Sync*)

La sortie SYNC inhibe le VCO du PLL à niveau bas et l'active à niveau haut.

— MFM (*MFM mode*)

La sortie MFM montre le mode d'opération du VCO. Il est à niveau haut pour MFM, à niveau bas pour MF.

— SIDE (*Head Select*)

La tête 1 est choisie lorsque l'entrée SIDE est 1 (niveau haut), la tête 0 lorsque SIDE est 0.

— US0, US1 (*Unit Select 0, 1*)

Les entrées US0 et US1 permettent le choix entre quatre lecteurs de disquette.

— PS0, PS1 (*Preshift 0, 1*)

Les sorties PS1 et PS0 sont les signaux de requête de précompensation en écriture pour le mode MFM :

PS0	PS1	Shift (MFM WDATA)
0	0	Normal
0	1	Late
1	0	Early
1	1	-

— READY

L'entrée READY indique que le FDD est prêt à recevoir des données.

— HDLD (*HeaD LoaD*)

La sortie HDLD spécifie à la tête de lecture-écriture du FDD de s'approcher de la disquette.

— FLT/TRK0 (*FauLT/TRacK 0*)

En mode lecture-écriture, l'entrée FLT détecte les conditions d'échec du FDD. En mode recherche, TRK0 indique que la piste en cours est 0.

— WPRT/2SIDE (*Write PRotecT/Two SIDE*)

En mode lecture-écriture, l'entrée WPRT détecte le statut de protection en écriture. En mode recherche, 2SIDE indique les disquettes à deux faces.

— FLTR/STEP (*FauLT Reset/STEP*)

En mode lecture-écriture, la sortie FLTR réinitialise la bascule d'erreur du FDD. En mode recherche, STEP émet des pulsations d'étape pour déplacer la tête vers un autre cylindre.

— LCT/DIR (*Low CurrenT/DIRection*)

En mode lecture-écriture, la sortie LCT indique que la tête de lecture-écriture est positionnée à un cylindre supérieur à 42. En mode recherche, la sortie DIR détermine la direction dans laquelle la tête se déplacera lorsqu'elle recevra une impulsion d'étape. Si DIR est 0, les recherches sont effectuées en arrière, en avant si DIR est 1.

— $\overline{\text{RW}}$/SEEK (*Read-Write/SEEK*)

La sortie $\overline{\text{RW}}$/SEEK spécifie le mode lecture-écriture à niveau bas, recherche à niveau haut.

— GND (*GRounD*)

Doit être relié à la terre.

— Vcc (+5V)

Doit être relié à une alimentation de +5 V.

10.2.2 Les registres du FDC

Le PD765 comporte trois registres et a donc besoin de trois ports : le *registre de sortie numérique* (DOR pour *Digital Output Register*), le *registre de statut* (*Main Status Register*) et le *registre de données* (*Data Register*).

Registre de sortie numérique DOR.- La structure de ce registre est la suivante :

7	6	5	4	3	2	1	0
MOTD	MOTC	MOTB	MOTA	DMA	$\overline{\text{REST}}$	DR1	DR0

— MOTD, MOTC, MOTB, MOTA : contrôle du moteur pour les lecteurs de disquette D, C, B et A respectivement (1 = démarrage du moteur, 0 = arrêt du moteur).
— DMA : DMA et canal IRQ (1 = activé, 0 = désactivé).
— REST : réinitialisation du contrôleur (1 = contrôleur activé, 0 = exécuter la réinitialisation du contrôleur).
— DR1, DR0 : choix du lecteur de disquette [00 = lecteur 0 (A), 01 = lecteur 1 (B), 10 = lecteur (C), 11 = lecteur 3 (D)].

On ne peut qu'écrire dans ce registre. Si le bit REST vaut 1, le contrôleur accepte et exécute les commandes. S'il est égal à zéro, le contrôleur ignore toutes les commandes et effectue une réinitialisation de tous les registres internes (autres que DOR).

Un lecteur de disquette ne peut être choisi que si son moteur est en marche (on peut le mettre en marche au moment du choix).

Exemple 1.- Pour démarrer le moteur du lecteur de disquette A et le sélectionner pour qu'il soit prêt pour une opération, on utilise les données suivantes :
— MOTA = 1 (démarrage du moteur du lecteur A),
— DMA = 1 (si on veut utiliser le DMA et les interruptions),
— REST = 1 (activation du contrôleur pour que les commandes puissent être exécutées),
— DR1, DR0 = 00 (sélection du lecteur A),
— tous les autres bits à zéro.

Ainsi 16 + 8 + 4 + 0 = 28, ou 01Ch, est envoyé au port adéquat (3f2h sur l'IBM PC) :

```
mov     al,01ch
mov     dx,03f2h
out     dx,al
```

Exemple 2.- Pour réinitialiser le contrôleur, il faut envoyer 0 au port 3f2h :

```
mov     al,0
mov     dx,03f2h
out     dx,al
```

Ceci a pour effet d'arrêter les moteurs de tous les lecteurs de disquette, de ne sélectionner aucun lecteur, de désactiver le DMA et la ligne IRQ et de réinitialiser les registres.

Registre de statut principal MSR.- La structure de ce registre est la suivante :

7	6	5	4	3	2	1	0
MRQ	DIO	NDMA	BUSY	ACTD	ACTC	ACTB	ACTA

— MRQ (pour *Main ReQuest*) : 1 = le registre des données est prêt à recevoir ou envoyer des données ou des commandes, 0 = registre des données non prêt.

— DIO (pour *Data Input/Output*) : 1 = du contrôleur vers le microprocesseur, 0 = du microprocesseur vers le contrôleur.

— NDMA (pour *Non-DMA mode*) : 1 = contrôleur non en mode DMA (dans ce cas, le contrôleur émet une interruption matérielle chaque fois qu'il veut recevoir ou envoyer un octet de données), 0 = contrôleur en mode DMA (qui peut être utilisé pour transférer des données vers ou depuis la mémoire vive).

— BUSY : 1 = une instruction est en train d'être effectuée, 0 = pas d'instruction en cours.

— ACTD, ACTC, ACTB, ACTA : le lecteur correspondant est actif (1) ou non actif (0). Actif veut dire que le lecteur est en train de positionner sa tête de lecture-écriture ou en train de se recalibrer.

Le MSR est seulement lu. Il contient les informations sur le statut du contrôleur.

Exemple 3.- Pour tester si le contrôleur est prêt à recevoir des commandes ou des données, il est nécessaire de tester MRQ et DIO. Ceci implique de lire le port, de masquer les bits et d'effectuer une comparaison :

```
mrqloop:
    mov dx,03f4h
    in  al,dx
    and al,0c0h
    cmp al,080h
    jne mrqloop
```

Ce code bouclera jusqu'à ce que le contrôleur dise qu'il est prêt à recevoir des données.

Registre des données.- C'est un registre de huit bits qui fournit un accès indirect à une pile de registres. Une commande nécessite de un à neuf octets ; le premier octet spécifie au contrôleur le nombre d'octets nécessaires. Le contrôleur envoie les octets de commande aux registres adéquats de sa pile, ce qui permet de se passer d'un regsitre d'index, utilisé par exemple dans les contrôleurs graphiques.

10.2.3 Le jeu de commandes du FDC

Le μPD765 comporte un jeu de 13 commandes, résumées dans le tableau suivant :

Nom de la commande	Fonction
Commandes de transfert de données	
Lire la piste complète	x2h
Écrire un secteur	x5h
Lire un secteur	x6h
Écrire sur un secteur effacé	x9h
Lire un secteur effacé	xch
Formater une piste	xdh
Commandes de contrôle	
Corriger les données du lecteur	x3h
Vérifier le statut du lecteur	x4h
Calibrer le lecteur	x7h
Vérifier le statut de l'interruption	x8h
Lire l'identificateur du secteur	xah
Rechercher/ranger la tête	xfh

Toutes les commandes sont transférées *via* le registre de donnée. Avant d'écrire une commande ou de lire l'octet de statut, il est nécessaire de lire le bit MRQ du registre MSR de façon à s'assurer que le registre de données est prêt à recevoir un octet. On doit également fixer le format du lecteur avant les opérations de lecture, d'écriture et de formatage.

10.2.3.1 Format des commandes de transfert de données

Ce sont les commandes utilisées pour transférer des données entre une disquette et la mémoire vive ou pour formater une piste. Toutes les commandes de transfert des données renvoient leurs résultats en utilisant le même format, résumé dans le tableau suivant :

Octet \ Bit	7	6	5	4	3	2	1	0
0	M	F	S	0	0	1	1	0
1	x	x	x	x	x	HD	DR1	DR0
2	Cylindre							
3	Tête							
4	Numéro de secteur							
5	Taille du secteur							
6	Longueur de la piste/numéro de secteur maximum							
7	Longueur de GAP3							
8	Longueur des données							

— Le bit **M** du premier octet spécifie s'il s'agit d'une opération multi-pistes, c'est-à-dire qu'on lit les deux pistes du cylindre en utilisant les deux têtes à la fois (1) ou d'un seul côté (0).

— Le bit **F** spécifie la méthode de codage : 1 pour MFM et 0 pour FM.

— Le bit **S** (pour *Skip*) spécifie le mode de dépassement : si S = 1, on doit passer les DAM (*Data Address Mark*) effacées et ne pas le faire si S = 0.

— Le bit **HD** (pour *HeaD*) du deuxième octet spécifie le numéro de la tête de lecture-écriture. Il doit désigner la même tête que dans le troisième octet.

— Les bits **DR1**–**DR0** spécifient le lecteur de la façon suivante : 00 = lecteur 0 (A), 01 = lecteur 1 (B), 10 = lecteur 2 (C) et 11 = lecteur 3 (D).

— Les octets 2, 3 et 4 permettent de spécifier le premier secteur à lire (ce qui ne sert à rien pour cette première commande).

— Le cinquième octet spécifie la taille du secteur : 0 = 128 octets, 1 = 256 octets, 2 = 512 octets, 3 = 1 024 octets, ..., 7 = 16 KiO.

— Le sixième octet spécifie le nombre de secteurs par piste ou le numéro maximum de secteur pour lequel l'instruction doit être exécutée.

— Le septième octet spécifie la longueur de GAP3 avec un minimum de 32. La valeur standard est 42.

— Le huitième octet spécifie la longueur des données, en octets, et n'a de sens que s'il s'agit de 0 (sinon on a toujours FFh).

10.2.3.2 Lecture d'une piste (x2h)

Lorsqu'une commande de lecture de piste est émise, les données d'une piste sont complètement lues. La spécification du secteur est ignorée pour cette commande : la lecture commence au premier secteur qui suit IDAM (*Index Address Mark*), en lisant secteur par secteur (sans se préoccuper du numéro de secteur logique spécifié dans ID), jusqu'à ce que la fin de la piste soit atteinte.

La piste est considérée comme un bloc continue de données. Le tampon de lecture doit être assez grand pour contenir l'ensemble des données. Les opérations multi-pistes ne sont donc pas permises pour cette commande.

L'envoi des neuf octets de commande décrits précédemment constitue la *phase de commande*. Elle est suivie d'une *phase de résultats*, constituée de sept octets :

Octet \ Bit	7	6	5	4	3	2	1	0
0	ST0							
1	ST1							
2	ST2							
3	Cylindre							
4	Tête							
5	Numéro de secteur							
6	Taille du secteur							

Les registres de statut ST0–ST2 renvoient des informations qui peuvent nous aider, soit à confirme une exécution correcte de la commande, soit à déterminer la cause d'une erreur :

— Le format de ST0 est :

7	6	5	4	3	2	1	0
IC_1	IC_0	SE	UC	NR	HD	US_1	US_0

  — IC_1–IC_0 constitue le code d'interruption (*Interrupt Code*) :
    — 00 = terminaison normale (la commande s'est terminée sans erreur).
    — 01 = terminaison anormale (le contrôleur a commencé l'exécution de la commande mais n'a pas pu la terminer correctement).
    — 10 = commande non valide (le contrôleur n'a donc pas commencé l'exécution de la commande).
    — 11 = terminaison anormale (le lecteur n'était pas prêt).
  — SE (*Seek End*) indique que le contrôleur a terminé une commande de recherche ou de calibration ou a exécuté correctement une commande de lecture ou d'écriture comprenant une recherche implicite.
  — UC (*Unity Check*) est positionné si le lecteur a échoué ou si une recalibration ne peut pas trouver la piste 0 après 79 impulsions.
  — NR (*drive Not Ready*) est positionné si le lecteur n'est pas prêt.
  — HD (*HeaD*) spécifie la tête de lecture actuellement active.
  — US_1–US_0 (*Unit Select*) spécifie le lecteur actuellement sélectionné.

— Le format de ST1 est :

7	6	5	4	3	2	1	0
EN	xx	DE	TO	xx	NDAT	NW	NID

  — EN (*ENd of cylinder*) est positionné lorsque le compteur de secteur excède le nombre de secteurs de la piste, c'est-à-dire lorsque le contrôleur essaie d'accéder à un secteur se trouvant après le dernier secteur du cylindre.
  — DE (*Data Error*) est positionné lorsque le contrôleur a détecté une erreur dans le champ ID ou dans le champ de données d'un secteur.
  — TO (*Time-Out*) est positionné lorsqu'aucun signal n'a été reçu de la part du contrôleur DMA ou du microprocesseur dans l'intervalle de temps requis.
  — NDAT (*No DATa*) est positionné lorsque le secteur spécifié n'a pas été trouvé par le contrôleur ou lorsque le contrôleur ne peut pas lire ID ou lorsque le contrôleur ne peut pas déterminer correctement la suite de secteurs.
  — NW (*Not Writable*) est positionné lorsque le disque est protégé en écriture alors qu'on essaie de lire.
  — NID (*No ADdress mark*) est positionné lorsque ID n'a pas été trouvé après une révolution complète du disque ou lorsque le contrôleur ne peut pas trouver un motif DAM ou le secteur spécifié.

— Le format de ST2 est :

7	6	5	4	3	2	1	0
xx	DADM	CRCE	WCYL	SEQ	SERR	BCYL	NDAM

  — DADM (*Deleted ADdress Mark*) est positionné lorsqu'un DADM est détecté lors d'une commande de lecture d'un secteur ou qu'un DAM est détecté lors d'une commande de lecture d'un secteur effaçé.
  — CRCE (*CRC Error*) est positionné lorsqu'une erreur de CRC est décelée dans le champ données du secteur.
  — WCYL (*Wrong CYLinder*) est positionné lorsque la piste spécifiée par le contrôleur et celle spécifiée par l'ID sont différentes.
  — SEQ (*Seek EQual*) est positionné lorsque la condition *seek equal* est remplie.
  — SERR (*Seek ERRor*) est positionné lorsque le contrôleur ne trouve pas le secteur correspondant lors d'une recherche sur le cylindre.
  — BCYL (*Bad CYLinder*) indique que l'adresse de piste dans ID diffère de l'adresse de piste spécifiée par le contrôleur.
  — NDAM (*Not DAM*) est positionné lorsque le contrôleur ne peut pas trouver un DAM valide ou effacé.

10.2.3.3 Écriture d'un secteur (x5h)

Le format de cette commande est le même que pour la commande précédente, n'en différant que par le premier demi-octet de poids faible :

Octet \ Bit	7	6	5	4	3	2	1	0
0	M	F	S	0	0	1	0	1
1	x	x	x	x	x	HD	DR1	DR0
2	Cylindre							
3	Tête							
4	Numéro de secteur							
5	Taille du secteur							
6	Longueur de la piste/numéro de secteur maximum							
7	Longueur de GAP3							
8	Longueur des données							

Elle permet de transférer un ou plusieurs secteurs de la mémoire vive vers le contrôleur, à partir duquel ils sont transférés au disque. Lorsque le contrôleur écrit un secteur, il écrit également un DAM valide. Cette commande peut concerner les deux têtes. La phase de résultats est la même que pour la commande précédente.

10.2.3.4 Lecture d'un secteur (x6h)

Le format de cette commande est le même que pour la première commande, n'en différant que par le premier demi-octet de poids faible :

Octet \ Bit	7	6	5	4	3	2	1	0
0	M	F	S	0	0	1	1	0
1	x	x	x	x	x	HD	DR1	DR0
2	Cylindre							
3	Tête							
4	Numéro de secteur							
5	Taille du secteur							
6	Longueur de la piste/numéro de secteur maximum							
7	Longueur de GAP3							
8	Longueur des données							

Elle permet de transférer un ou plusieurs secteurs ayant un DAM valide du disque vers le contrôleur, puis vers la mémoire vive. La phase de résultats est la même que pour la première commande.

10.2.3.5 Écriture d'un secteur effacé (x9h)

Le format de cette commande est le même que pour la première commande, n'en différant que par le premier octet :

Octet \ Bit	7	6	5	4	3	2	1	0
0	M	F	0	0	1	0	0	1
1	x	x	x	x	x	HD	DR1	DR0
2	Cylindre							
3	Tête							
4	Numéro de secteur							
5	Taille du secteur							
6	Longueur de la piste/numéro de secteur maximum							
7	Longueur de GAP3							
8	Longueur des données							

Elle effectue la même chose que la commande d'écriture d'un secteur sauf qu'elle écrit un DDAM au lieu d'un DAM normal. La phase de résultats est la même que pour la première commande.

10.2.3.6 Lecture d'un secteur effacé (xCh)

Le format de cette commande est le même que pour la première commande, n'en différant que par le premier octet :

Octet \ Bit	7	6	5	4	3	2	1	0
0	M	F	0	0	1	1	0	0
1	x	x	x	x	x	HD	DR1	DR0
2	Cylindre							
3	Tête							
4	Numéro de secteur							
5	Taille du secteur							
6	Longueur de la piste/numéro de secteur maximum							
7	Longueur de GAP3							
8	Longueur des données							

Elle effectue la même chose que la commande de lecture d'un secteur sauf qu'elle ne peut lire que les secteurs avec un DDAM. Les secteurs ayant un DAM normal sont ignorés. La phase de résultats est la même que pour la première commande.

10.2.3.7 Formatage d'une piste (xDh)

Le format de cette commande est le suivant :

Octet \ Bit	7	6	5	4	3	2	1	0
0	0	F	0	0	1	1	0	1
1	x	x	x	x	x	HD	DR1	DR0
2	Taille du secteur							
3	Longueur de la piste							
4	Longueur de GAP3							
5	Octet de remplissage							

Cette commande formate une piste. Un tampon de format constitué de quatre octets doit être fourni pour chaque secteur de la piste. Ce tampon contient l'identificateur ID du secteur correspondant. Le tampon doit être suffisamment grand pour contenir les données nécessaires pour tous les secteurs de la piste. Le format du tampon est le suivant :

Octet	Commande
0	Piste
1	Tête
2	Numéro de secteur
3	Taille du secteur

où la taille du secteur utilise les codes décrits lors de la première commande.

Il vaut mieux programmer le contrôleur de DMA pour qu'il lise le tampon de formats depuis un canal DAM (le canal 2 dans le cas du PC). On peut cependant, également, transférer les données de formatage en utilisant un échange de données piloté par interruption. Dans ce cas, le contrôleur émet une interruption matérielle avant le formatage de chaque secteur. La routine de service peut alors transférer les quatre octets de format pour le prochain secteur à formater.

Le formatage débute lorsque le lecteur a fourni un signal sur la broche IDX, indiquant ainsi le début d'une piste. Les secteurs sont formatés l'un après l'autre jusqu'à ce que le lecteur retransmette ce même signal, indiquant cette fois la fin de la piste (notons que le début et la fin de la piste correspondent au même point, matérialisé par le trou d'index de la disquette).

La phase de résultats est la même que pour la première commande.

10.2.3.8 Correction des données du lecteur (x3h)

Le format de cette commande est le suivant :

Octet \ Bit	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	1
1	Step Rate				Head Unload Time			
2	Head Load Time							NDM

Cette commande est utilisées pour transmettre les données de contrôle mécanique au contrôleur pour tous les lecteurs de disquette connectées, conformément aux tableaux ci-dessous :

Step Rate (ms)				
Entry	Data Transfer Rate			
	1M	500k	300k	250k
0h	8.0	16	26.7	32
1h	7.5	15	25.0	30
2h	7.0	14	23.3	28
...	...	...	...	...
Eh	1.0	2	3.3	4
Fh	0.5	1	1.7	2

Head Unload Time (ms)				
Entry	Data Transfer Rate			
	1M	500k	300k	250k
0h	128	256	426	512
1h	8	16	26.7	32
2h	16	32	53.3	64
...	...	...	...	...
Eh	112	224	373	448
Fh	120	240	400	480

Head Load Time (ms)				
Entry	Data Transfer Rate			
	1M	500k	300k	250k
0h	128	256	426	512
1h	1	2	3.3	4
2h	2	4	6.6	8
...	...	...	...	...
Eh	126	252	420	504
Fh	127	254	423	508

Le champ NDM (*Non-DMA Mode*) est égal à 0 pour un transfert des données par DMA et à 1 sinon.

Cette commande n'a pas de phase de résultat.

10.2.3.9 Vérification du statut d'un lecteur (x4h)

Le format de cette commande est le suivant :

Octet \ Bit	7	6	5	4	3	2	1	0
0	0	F	0	0	1	1	0	1
1	x	x	x	x	x	HD	DR1	DR0

Cette commande renvoie des information de statut relatives à l'état des lecteurs de disquettes connectées.

La phase de résultats consiste à lire un octet :

Octet \ Bit	7	6	5	4	3	2	1	0
0	ST3							

Le registre de statut 3 contient des informations sur le lecteur : Le format de ST0 est :

7	6	5	4	3	2	1	0
ESIG	WPDR	RDY	TRK0	DSDR	HDDR	DS1	DS0

— ESIG (*Error SIGnal*) est positionné lorsque le signal d'erreur du lecteur est actif, c'est-à-dire lorsqu'une erreur est survenue.

— WPDR (*Write PRotection*) est positionné lorsque la disquette est protégée en écriture.

— RDY (*ReaDY*) est positionné lorsque le lecteur est prêt.

— TRK0 (*TRacK 0*) est positionné lorsque la tête est au-dessus de la piste 0.

— DSDR (*Double Sided DRive*) est positionné lorsque le lecteur peut lire les deux côtés d'une disquette.

— HDDR (*HeaD DRive*) indique la tête active.

— DS1–DS0 spécifie le lecteur actuellement sélectionné.

10.2.3.10 Calibration d'un lecteur de disquette (x7h)

Le format de cette commande est le suivant :

Octet \ Bit	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	1	1
1	x	x	x	x	x	0	DR1	DR0

Cette commande est utilisée pour positionner la tête de lecture-écriture sur le cylindre 0. Si une erreur de recherche survient au cours de l'accès à un secteur, la tête peut être déplacée à un cylindre absolu pour recalibrer le lecteur.

Cette commande ne renvoie pas de phase de résultat mais une interruption est émise une fois qu'elle a été effectuée.

Lorsque le contrôleur reçoit cette commande, il positionne le signal DIR à 0 et passe au lecteur jusqu'à 79 impulsions de déplacement de la tête de lecture-écriture. Après chacune de ces impulsions, le contrôleur vérifie le signal TRK0. S'il est actif (c'est-à-dire si la tête se trouve au-dessus de la piste 0), le contrôleur positionne le bit SE du registre de statut 0 et termine la commande. Si TRK0 n'est pas actif après 79 impulsions de déplacement, le contrôleur positionne les bits SE et EC du registre de statut 0 et termine la commande.

Pour recalibrer le lecteur, il se peut qu'on doive émettre plusieurs commandes de recalibration, en particulier si le lecteur a plus que 80 pistes. Après avoir effectuer cette commande, on doit toujours vérifier que la tête est correctement positionnée au-dessus de la piste 0 en utilisant la commande de vérification du statut d'interruption. Une recalibration est toujours nécessaire au démarrage, pour initialiser correctement la position de la tête.

10.2.3.11 Vérification du statut d'interruption (x8h)

Le format de cette commande est le suivant :

Octet \ Bit	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0

Cette commande est utilisée pour vérifier les informations de statut sur l'état du contrôleur dans la phase de résultats lorsque le contrôleur a émis une interruption.

Le signal d'interruption est réinitialisé par cette commande, qui détermine la source de l'interruption *via* le registre de statut ST0. Si la commande est émise alors qu'il n'y a pas d'interruption en attente, la valeur 80h est renvoyée dans ST0, correspondant au message de commande non valide.

Une interruption est émise dans les cas suivants :

— au début de la phase des résultats des commandes :
  — lecture d'un secteur ;
  — lecture d'un secteur effacé ;
  — écriture d'un secteur ;
  — écriture sur un secteur effacé ;
  — lecture d'une piste ;
  — formatage d'une piste ;
  — lecture d'un ID de secteur ;
  — vérification.

— après l'exécution des commandes suivantes, qui n'ont pas de phase de résultats :
  — calibration d'un lecteur ;
  — recherche ;
  — recherche relative.

— pour l'échange de données entre la mémoire vive et le contrôleur lorsque l'échange piloté par interruption est actif (autrement dit lorsqu'on n'utilise pas le DMA).

La phase de résultats est constituée de deux octets :

Octet \ Bit	7	6	5	4	3	2	1	0
0	ST0							
1	Cylindre en cours							

10.2.3.12 Lecture de l'ID d'un secteur (xAh)

Le format de cette commande est le suivant :

Octet \ Bit	7	6	5	4	3	2	1	0
0	0	F	0	0	1	0	1	0
1	x	x	x	x	x	HD	DR1	DR0

Cette commande est utilisée pour lire le premier identificateur de secteur ID que le contrôleur est capable de détecter. En utilisant cette commande, il est possible de déterminer la position en cours de la tête de lecture-écriture. Si on n'arrive à lire aucun ID lors d'une révolution complète de la disquette, le contrôleur émet un message d'erreur, que l'on récupère dans ST0–2.

La phase de résultats est constituée des sept octets habituels :

Octet \ Bit	7	6	5	4	3	2	1	0
0	ST0							
1	ST1							
2	ST2							
3	Cylindre							
4	Tête							
5	Numéro de secteur							
6	Taille du secteur							

10.2.3.13 Rechercher/ranger la tête (xFh)

Le format de cette commande est le suivant :

Octet \ Bit	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	1	1
1	x	x	x	x	x	HD	DR1	DR0
2	Cylindre							

La commande de recherche de la tête, quelquefois apelée commande de rangement de la tête, déplace la tête de lecture-écriture sur le cylindre spécifié. Lorsque le contrôleur reçoit cette commande, il compare le numéro de cylindre transmis au numéro de cylindre en cours. Le signal de direction (DIR) est alors positionné et des impulsions du moteur pas à pas sont émises jusqu'à ce que les deux numéros de cylindre coïncident.

Cette commande n'a pas de phase de résultats.

10.2.3.14 Commande non valide

Lorsqu'une commande non valide est émise, le contrôleur positionne le bit 7 de ST0. La phase de résultats est alors :

Octet \ Bit	7	6	5	4	3	2	1	0
0	ST0							

10.3 Cas de l'IBM PC

Sur l'IBM/XT, il peut y avoir deux contrôleurs. Ils utilisent le canal 2 de DMA pour le transfert des données, l'émission d'une requête d'interruption matérielle s'effectue *via* IRQ6 pour être servie par INT 0Eh.

	Adresse primaire	Adresse secondaire	Écriture (W) Lecture (R)
Adresse de base	3F0h	370h	
Digital Output Register DOR	3F2h	372h	W
Main Status Register	3F4h	374h	R
Data Register	3F5h	375h	R/W
Canal DMA	2	2	
IRQ	6	6	
INTR	0Eh	0Eh	

10.4 Commentaire du BIOS : lecteur de disquette

10.4.1 Zone de communication du BIOS concernant le lecteur de disquette

Nous avons déjà vu la description de la zone de communication du BIOS consacrée au lecteur de disquette au chapite 3. Elle débute à l'adresse `0400:3Eh` et présente la structure suivante :

Adresse	Variable	Signification
3E	SEEK_STATUS	Réinitialisation
3F	MOTOR_STATUS	État du moteur
40	MOTOR_COUNT	Durée de mise en arrêt
41	DISKETTE_STATUS	Code de retour
42–48	NEC_STATUS	État du contrôleur NEC

Les quatre premiers octets sauvegardent l'état des lecteurs de disquettes pour chaque opération. Les sept octets de NEC_STATUS contiennent l'état retourné par le contrôleur de disquette après chaque opération, dont nous avons vu qu'il est constitué au plus de sept octets. Un résumé sur un octet est placé par le BIOS dans l'octet DISKETTE_STATUS.

Les octets MOTOR_STATUS et MOTOR_COUNT contrôlent la durée d'arrêt du moteur du lecteur de disquette. Le premier octet signale que l'unité de disquette est active et le second détermine la durée de rotation de la disquette après une opération. MOTOR_COUNT contient une valeur de compteur décrémentée en fonction du compteur d'impulsion. Par défaut ce temps de rotation supplémentaire est d'environ deux secondes.

10.4.2 Choix de la fonction

10.4.2.1 La sous-routine principale

La routine de service des entrées-sorties sur disquette (INT 13h) commence à la ligne 2301 du listing :

```
EC52 363031            2301   F3      DB       '601',13,10          ; DISKETTE ERROR
EC55 0D
EC56 0A
                       2302
                       2303   ;-- INT 13 ------------------------------------------------------------
                       2304   ; DISKETTE I/O                                                        :
                       2305   ;       THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4 DISKETTE DRIVES     :
                       2306   ; INPUT                                                               :
                       2307   ;       (AH)=0 RESET DISKETTE SYSTEM                                  :
                       2308   ;              HARD RESET TO NEC, PREPARE COMMAND, RECAL REQUIRED      :
                       2309   ;              ON ALL DRIVES                                          :
                       2310   ;       (AH)=1 READ THE STATUS OF THE SYSTEM INTO (AL)                :
                       2311   ;              DISKETTE_STATUS FROM LAST OPERATION IS USED             :
                       2312   ;                                                                     :
                       2313   ; REGISTERS FOR READ/WRITE/VERIFY/FORMAT                              :
                       2314   ;       (DL) - DRIVE NUMBER (0-3 ALLOWED, VALUE CHECKED)              :
                       2315   ;       (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)           :
                       2316   ;       (CH) - TRACK NUMBER (0-39; NOT VALUE CHECKED)                 :
                       2317   ;       (CL) - SECTOR NUMBER (1-8, NOT VALUE CHECKED,                 :
                       2318   ;                                 NOT USED FOR FORMAT)                :
                       2319   ;       (AL) - NUMBER OF SECTORS ( MAX = 8, NOT VALUE CHECKED, NOT USED :
                       2320   ;                                           FOR FORMAT)               :
                       2321   ;       (ES:BX) - ADDRESS OF BUFFER ( NOT REQUIRED FOR VERIFY)        :
                       2322   ;                                                                     :
                       2323   ;       (AH)=2 READ THE DESIRED SECTORS INTO MEMORY                   :
                       2324   ;       (AH)=3 WRITE THE DESIRED SECTORS FROM MEMORY                  :
                       2325   ;       (AH)=4 VERIFY THE DESIRED SECTORS                             :
                       2326   ;       (AH)=5 FORMAT THE DESIRED TRACK                               :
                       2327   ;              FOR THE FORMAT OPERATION, THE BUFFER POINTER (ES,BX)    :
                       2328   ;              MUST POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS  :
                       2329   ;              FOR THE TRACK.  EACH FIELD IS COMPOSED OF 4 BYTES,      :
                       2330   ;              (C,H,R,N), WHERE C = TRACK NUMBER, H=HEAD NUMBER,       :
                       2331   ;              R = SECTOR NUMBER, N= NUMBER OF BYTES PER SECTOR        :
                       2332   ;              (00=128, 01=256, 02=512, 03=1024).  THERE MUST BE ONE   :
                       2333   ;              ENTRY FOR EVERY SECTOR OF THE TRACK.  THIS INFORMATION  :
                       2334   ;              IS USED TO FIND THE REQUESTED SECTOR DURING READ/WRITE  :
                       2335   ;              ACCESS.                                                :
                       2336   ;                                                                     :
                       2337   ; DATA VARIABLE -- DISK_POINTER                                       :
                       2338   ;       DOUBLE WORD POINTER TO THE CURRENT SET OF DISKETTE PARAMETERS  :
                       2339   ; OUTPUT                                                              :
                       2340   ;       AH = STATUS OF OPERATION                                      :
                       2341   ;              STATUS BITS ARE DEFINED IN EQUATES FOR                 :
                       2342   ;              DISKETTE_STATUS VARIABLE IN THE DATA SEGMENT OF THIS   :
                       2343   ;              MODULE.                                                :
                       2344   ;       CY = 0  SUCCESSFUL OPERATION (AH=0 ON RETURN)                 :
                       2345   ;       CY = 1  FAILED OPERATION (AH HAS ERROR REASON)                :
                       2346   ;       FOR READ/WRITE/VERIFY                                         :
                       2347   ;              DS,BX,DX,CH,CL PRESERVED                               :
                       2348   ;              AL = NUMBER OF SECTORS ACTUALLY READ                   :
                       2349   ;              ***** AL MAY NOT BE CORRECT IF TIME OUT ERROR OCCURS    :
                       2350   ;       NOTE:  IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE      :
                       2351   ;              APPRPRIATE ACTION IS TO RESET THE DISKETTE, THEN RETRY  :
                       2352   ;              THE OPERATION. ON READ ACCESSES, NO MOTOR START DELAY   :
                       2353   ;              IS TAKEN, SO THAT THREE RETRIES ARE REQUIRED ON READS   :
                       2354   ;              TO ENSURE THAT THE PROBLEM IS NOT DUE TO MOTOR          :
                       2355   ;              START-UP.                                              :
                       2356   ;-----------------------------------------------------------------------
                       2357           ASSUME  CS:CODE,DS:DATA,ES:DATA
EC59                   2358           ORG     0EC59H
EC59                   2359   DISKETTE_IO     PROC    FAR
EC59 FB                2360           STI                             ; INTERRUPTS BACK ON
EC5A 53                2361           PUSH    BX                      ; SAVE ADDRESS
EC5B 51                2362           PUSH    CX
EC5C 1E                2363           PUSH    DS                      ; SAVE ALL REGISTERS DURING OPERATION
EC5D 56                2364           PUSH    SI
```

```
EC5E 57                 2365            PUSH    DI
EC5F 55                 2366            PUSH    BP
EC60 52                 2367            PUSH    DX
EC61 88EC               2368            MOV     BP,SP                   ; SET UP POINTER TO HEAD PARM
EC63 E8F300             2369            CALL    DDS
EC66 E81C00             2370            CALL    J1                      ; CALL THE REST TO ENSURE DS RESTORED
EC69 880400             2371            MOV     BX,4                    ; GET THE MOTOR WAIT PARAMETER
EC6C E8F001             2372            CALL    GET_PARM
EC6F 88264000           2373            MOV     MOTOR_COUNT,AH          ; SET THE TIMER COUNT FOR THE MOTOR
EC73 8A264100           2374            MOV     AH,DISKETTE_STATUS      ; GET STATUS OF OPERATION
EC77 80FC01             2375            CMP     AH,1                    ; SET THE CARRY FLAG TO INDICATE
EC7A F5                 2376            CMC                             ;  SUCCESS OR FAILURE
EC7B 5A                 2377            POP     DX                      ; RESTORE ALL REGISTERS
EC7C 50                 2378            POP     BP
EC7D 5F                 2379            POP     DI
EC7E 5E                 2380            POP     SI
EC5F 1F                 2381            POP     DS
EC80 59                 2382            POP     CX
EC81 5B                 2383            POP     BX                      ; RECOVER ADDRESS
EC82 CA0200             2384            RET     2                       ; THROW AWAY SAVED FLAGS
                        2385    DISKETTE_IO     ENDP
                        2386
EC85                    2387    J1      PROC    NEAR
EC85 8AF0               2388            MOV     DH,AL                   ; SAVE # SECTORS IN DH
EC87 80263F007F         2389            AND     MOTOR_STATUS,07FH       ; INDICATE A READ OPERATION
EC8C 0AE4               2390            OR      AH,AH                   ; AH=0
EC8E 7427               2391            JZ      DISK_RESET
EC90 FECC               2392            DEC     AH                      ; AH=1
EC92 7473               2393            JZ      DISK_STATUS
EC94 C606410000         2394            MOV     DISKETTE_STATUS,0       ; RESET THE STATUS INDICATOR
EC99 80FA04             2395            CMP     DL,4                    ; TEST FOR DRIVE IN 0-3 RANGE
EC9C 7313               2396            JAE     J3                      ; ERROR IF ABOVE
EC9E FECC               2397            DEC     AH                      ; AH=2
ECA0 7469               2398            JZ      DISK_READ
ECA2 FECC               2399            DEC     AH                      ; AH=3
ECA4 7503               2400            JNZ     J2                      ; TEST_DISK_VERF
ECA6 E99500             2401            JMP     DISK_WRITE
ECA9                    2402    J2:                                     ; TEST_DISK_VERF
ECA9 FECC               2403            DEC     AH                      ; AH=4
ECAB 7467               2404            JZ      DISK_VERF
ECAD FECC               2405            DEC     AH                      ; AH=5
ECAF 7467               2406            JZ      DISK_FORMAT
ECB1                    2407    J3:                                     ; BAD_COMMAND
ECB1 C606410001         2408            MOV     DISKETTE_STATUS,BAD_CMD ; ERROR CODE, NO SECTORS TRANSFERRED
ECB6 C3                 2409            RET                             ; UNDEFINED OPERATION
                        2410    J1      ENDP
                        2411
```

Commentaires.- 1^o) L'adresse absolue du code est EC59h (ligne 2358) pourpour que celui-ci corresponde avec le vecteur d'interruption 13h. Le segment de code est celui du BIOS, le segment de données est celui de la zone de communication du BIOS (ligne 2357). On permet les interruptions masquables (ligne 2360) puisque la routine de service en aura besoin. On sauvegarde les principaux registres (lignes 2361 à 2367), que l'on restaurera à la fin (lignes 2377 à 2383). On sauvegarde le contenu de SP dans BP (ligne 2368) et on fait en sorte que DS pointe sur la zone de communication du BIOS (ligne 2369).

- 2^o) On fait appel à une procédure (ligne 2370) pour déterminer le numéro de fonction (contenu dans AH) et aller à la sous-routine adéquate suivant celui-ci.

On sauvegarde le nombre de secteurs (eventuel) dans DH (ligne 2388) et on indique que l'on est en train d'effectuer une opération de lecture sur la disquette (ligne 2389) dans la variable MOTOR_STATUS de la zone de communication du BIOS (définie ligne 140).

— Si AH = 0, il s'agit de la fonction 0 de réinitialisation (lignes 2390 et 2391), on fait donc appel à la sous-routine DISK_RESET, définie ligne 2414 et étudiée ci-dessous.

— Si AH = 1, il s'agit de la fonction 1 de demande de statut (lignes 2392 et 2393), on fait donc appel à la sous-routine DISK_STATUS, définie ligne 2462 et étudiée ci-dessous.

— Pour les fonctions suivantes, on réinitialise l'indicateur de statut (ligne 2394). La variable DISKETTE_STATUS de la zone de communication du BIOS est définie ligne 149. Si le numéro de lecteur de disquette est supérieur à 4 (lignes 2395 et 2396), il s'agit d'une commande non valide ; on le spécifie dans la variable DISKETTE_STATUS (ligne 2408) et on termine la routine de service.

— Si AH = 2, il s'agit de la fonction 2 de lecture (lignes 2397 et 2398), on fait donc appel à la sous-routine DISK_READ, définie ligne 2469 et étudiée ci-dessous.

— Si AH = 3, il s'agit de la fonction 3 d'écriture (lignes 2399 à 2401), on fait donc appel à la sous-routine DISK_WRITE, définie ligne 2505 et étudiée ci-dessous.

— Si AH = 4, il s'agit de la fonction 4 de vérification (lignes 2403 à 2404), on fait donc appel à la sous-routine DISK_VERF, définie ligne 2479 et étudiée ci-dessous.

— Si AH = 5, il s'agit de la fonction 5 de formatage (lignes 2405 à 2406), on fait donc appel à la sous-routine DISK_FORMAT, définie ligne 2486 et étudiée ci-dessous.

— Sinon il s'agit d'une commande non valide. Nous avons vu ci-dessus comment on la traite (lignes 2407 à 2409).

- 3^o) Une fois la sous-routine adéquate exécutée, on récupère les paramètres d'attente du moteur (lignes 2371 et 2372), que l'on place (ligne 2373) dans la variable MOTOR_COUNT de la zone de communication du BIOS (définie ligne 146). Si DISKETTE_STATUS vaut 1, on positionne l'indicateur de retenue pour indiquer une erreur (lignes 2374 à 2376).

10.4.2.2 La sous-routine d'obtention des paramètres

La sous-routine précédente fait appel à la sous-routine GET_PARM de chargement des paramètres du lecteur de disquette. Celle-ci commence à la ligne 2737 du listing :

```
                        2724  ;--------------------------------------------------------------------------
                        2725  ; GET_PARM                                                               :
                        2726  ;       THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE     :
                        2727  ;       BLOCK POINTED AT BY THE DATA VARIABLE DISK_POINTER. A BYTE FROM :
                        2728  ;       THAT TABLE IS THEN MOVED INTO AH. THE INDEX OF THAT BYTE BEING  :
                        2729  ;       THE PARM IN BX                                                 :
                        2730  ; ENTRY --                                                             :
                        2731  ;  BX = INDEX OF BYTE TO BE FETCHED * 2                               :
                        2732  ;       IF THE LOW BIT OF BX IS ON, THE BYTE IS IMMEDIATELY OUTPUT      :
                        2733  ;       TO THE NEC CONTROLLER                                          :
                        2734  ; EXIT --                                                              :
                        2735  ;  AH = THAT BYTE FRM BLOCK                                           :
                        2736  ;--------------------------------------------------------------------------
EE6C                    2737  GET_PARM        PROC    NEAR
EE6C 1E                 2738          PUSH    DS                      ; SAVE SEGMENT
EE6D 2BC0               2739          SUB     AX,AX                   ; ZERO TO AX
EE6F 8ED8               2740          MOV     DS,AX
                        2741          ASSUME  DS:ABS0
EE71 C5367800           2742          LDS     SI,DISK_POINTER         ; POINT TO BLOCK
EE75 D1EB               2743          SHR     BX,1                    ; DIVIDE BX BY 2, AND SET FLAG
                        2744                                          ; FOR EXIT
EE77 8A20               2745          MOV     AH,[SI+BX]              ; GET THE WORD
EE79 1F                 2746          POP     DS                      ; RESTORE SEGMENT
                        2747          ASSUME  DS:DATA
EE7A 72C5               2748          JC      NEC_OUTPUT              ; IF FLAG SET, OUTPUT TO CONTROLLER
EE7C C3                 2749          RET
                        2750  GET_PARM        ENDP
```

Commentaires.- 1^o) Le segment des données est sauvegardé sur la pile (lignes 2738 et 2746) et on se place dans le segment commençant à zéro (lignes 2739 à 2741).

- 2^o) On fait pointer SI sur le bloc des paramètres DISK_POINTER. La variable DISK_POINTER est définie ligne 55 comme se trouvant à l'emplacement mémoire 78h. Comme

nous le verrons, elle est initialisée au démarrage avec le décalage de la variable DISK_BASE (ligne 1427).

DISK_BASE est défini ligne 3065 :

```
                        3057   ;--------------------------------------------------------------------------
                        3058   ; DISK_BASE                                                               :
                        3059   ;       THIS IS THE SET OF PARAMETERS REQUIRED FOR DISKETTE OPERATION.     :
                        3060   ;       THEY ARE POINTED AT BY THE DATA VARIABLE DISK_POINTER. TO          :
                        3061   ;       MODIFY THE PARAMETERS, BUILD ANOTHER BLOCK AND POINT               :
                        3062   ;       DISK_POINTER TO IT.                                               :
                        3063   ;--------------------------------------------------------------------------
EFC7                    3064           ORG     0EFC7H
EFC7                    3065   DISK_BASE       LABEL   BYTE
EFC7 CF                 3066           DB      11001111B       ; SRT=C, HD UNLOAD=0F - 1ST SPECIFY BYTE
EFC8 02                 3067           DB      2               ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
EFC9 25                 3068           DB      MOTOR_WAIT      ; WAIT AFTER OPN TIL MOTOR OFF
EFCA 02                 3069           DB      2               ; 512 BYTES/SECTOR
EFCB 08                 3070           DB      8               ; EOT ( LAST SECTOR ON TRACK)
EFCC 2A                 3071           DB      02AH            ; GAP LENGTH
EFCD FF                 3072           DB      0FFH            ; DTL
EFCE 50                 3073           DB      050H            ; GAP LENGTH FOR FORMAT
EFCF F6                 3074           DB      0F6H            ; FILL BYTE FOR FORMAT
EFD0 19                 3075           DB      25              ; HEAD SETTLE TIME (MILLISECONDS)
EFD1 04                 3076           DB      4               ; MOTOR WAIT TIME (1/8 SECONDS)
                        3077
```

La constante MOTOR_WAIT est définie à propos de la zone de communication du BIOS, ligne 147, égale à 37, ce qui correspond à deux secondes.

- 3^o) On divise par deux (ligne 2743) puisque BX contient le double de l'index. On place le paramètre voulu dans le registre AH (ligne 2745), ce qui était recherché.

- 4^o) On revient au segment des données de la zone de communication du BIOS (lignes 2747 et 2748). Si l'indicateur de retenue est positionné, on envoie au contrôleur (ligne 2748), grâce à la sous-routine NEC_OUTPUT étudiée ci-dessous.

10.4.2.3 Sous-routine d'envoi au contrôleur

La sous-routine NEC_OUTPUT commence ligne 2691 :

```
                        2674  ;--------------------------------------------------------------------------
                        2675  ; NEC_OUTPUT                                                             :
                        2676  ;       THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER AFTER TESTING     :
                        2677  ;       FOR CORRECT DIRECTION AND CONTROLLER READY THIS ROUTINE WILL      :
                        3678  ;       TIME OUT IF THE BYTE IS NOT ACCEPTED WITHIN A REASONABLE          :
                        2679  ;       AMOUNT OF TIME. SETTING THE DISKETTE STATUS ON COMPLETION.        :
                        2680  ; INPUT                                                                   :
                        2681  ;       (AH)     BYTE TO BE OUTPUT                                        :
                        2682  ; OUTPUT                                                                  :
                        2683  ;       CY = 0   SUCCESS                                                  :
                        2684  ;       CY = 1   FAILURE -- DISKETTE STATUS UPDATED                       :
                        2685  ;                IF A FAILURE HAS OCCURED, THE RETURN IS MADE ONE LEVEL    :
                        2686  ;                HIGHER THAN THE CALLER OF NEC_OUTPUT.                    :
                        2687  ;                THIS REMOVES THE REQUIREMENT OF TESTING AFTER EVERY       :
                        2688  ;                CALL OF NEC_OUTPUT.                                      :
                        2689  ;       (AL) DESTROYED                                                    :
                        2690  ;--------------------------------------------------------------------------
EE41                    2691  NEC_OUTPUT      PROC    NEAR
EE41 52                 2692          PUSH    DX                      ; SAVE REGISTERS
EE42 51                 2693          PUSH    CX
EE43 BAF403             2694          MOV     DX,03F4H                ; STATUS PORT
EE46 33C9               2695          XOR     CX,CX                   ; COUNT FOR TIME OUT
EE48                    2696  J23:
EE48 EC                 2697          IN      AL,DX                   ; GET STATUS
EE49 A84D               2698          TEST    AL,040H                 ; TEST DIRECTION BIT
EE4B 740C               2699          JZ      J25                     ; DIRECTION OK
EE4D E2F9               2700          LOOP    J23
EE4F                    2701  J24:                                    ; TIME_ERROR
EE4F 800E410080         2702          OR      DISKETTE_STATUS,TIME_OUT
EE54 59                 2703          POP     CX
EE55 5A                 2704          POP     DX                      ; SET ERROR CODE AND RESTORE REGS
EE56 58                 2705          POP     AX                      ; DISCARD THE RETURN ADDRESS
EE57 F9                 2706          STC                             ; INDICATE ERROR TO CALLER
EE58 C3                 2707          RET
EE59                    2708  J25:
EE59 33C9               2709          XOR     CX,CX                   ; RESET THE COUNT
EE5B                    2710  J26:
EE5B EC                 2711          IN      AL,DX                   ; GET THE STATUS
EE5C A880               2712          TEST    AL,080H                 ; IS IT READY
EE5E 7504               2713          JNZ     J27                     ; YES, GO OUTPUT
EE60 E2F9               2714          LOOP    J26                     ; COUNT DOWN AND TRY AGAIN
EE62 EBEB               2715          JMP     J24                     ; ERROR CONDITION
EE64                    2716  J27:                                    ; OUTPUT
EE64 8AC4               2717          MOV     AL,AH                   ; GET BYTE TO OUTPUT
EE66 B2F5               2718          MOV     DL,0F5H                 ; DATA PORT (3F5)
EE68 EE                 2719          OUT     DX,AL                   ; OUTPUT THE BYTE
EE69 59                 2720          POP     CX                      ; RECOVER REGISTERS
EE6A 5A                 2721          POP     DX
EE6B C3                 2722          RET                             ; CY = 0 FROM TEST INSTRUCTION
                        2723  NEC_OUTPUT      ENDP
```

Commentaires.- 1^o) On sauvegarde les registres utilisés DX et CX (lignes 2692, 2693, 2720 et 2721). Le registre DX prend la valeur du port de statut du contrôleur de disquette (ligne 2694), celle-ci apparaissant comme nombre magique et non comme constante. Le registre de compteur est initialisé à zéro (ligne 2695). On attend que le registre des données du contrôleur soit disponible (lignes 2696 à 2700). S'il ne l'est pas au bout d'un temps raisonnable, on place une erreur de « non réponse dans le temps imparti » dans la variable DISKETTE_STATUS (lignes 2701 et 2702), on positionne l'indicateur de retenue pour signaler une erreur (ligne 2706) et on retourne (ligne 2707) en ayant enlevé l'adresse de retour de la pile (ligne 2705).

- 2^o) On attend que le registre de données soit disponible (lignes 2708 à 2715) et on lui envoie l'octet contenu dans AH (lignes 2716 à 2719). Puis on termine la sous-routine (lignes 2720 à 2722).

10.4.3 Réinitialisation

10.4.3.1 La sous-routine principale

La fonction 0 de réinitialisation de INT 13h commence à la ligne 2414 du listing :

```
                    2412  ;----- RESET THE DISKETTE SYSTEM
                    2413
ECB7                2414  DISK_RESET      PROC    NEAR
ECB7 BAF203         2415          MOV     DX,03F2H                ; ADAPTER CONTROL PORT
ECBA FA             2416          CLI                             ; NO INTERRUPTS
ECBB A03F00         2417          MOV     AL,MOTOR_STATUS         ; WHICH MOTOR IS ON
ECBE B104           2418          MOV     CL,4                    ; SHIFT COUNT
ECC0 D204           2419          SAL     AL,CL                   ; MOVE MOTOR VALUE TO HIGH NYBBLE
ECC2 A820           2420          TEST    AL, 20H                 ; SELECT CORRESPONDING DRIVE
ECC4 750C           2421          JNZ     J5                      ; JUMP IF MOTOR ONE IS ON
ECC6 A840           2422          TEST    AL, 40H
ECC8 7506           2423          JNZ     J4                      ; JUMP IF MOTOR TWO IS ON
ECCA A880           2424          TESTL   AL, 80H
ECCC 7406           2425          JZ      J6                      ; JUMP IF MOTOR ZERO IS ON
ECCE FEC0           2426          INC     AL
ECD0                2427  J4:
ECD0 FEC0           2428          INC     AL
ECD2                2429  J5:
ECD2 FEC0           2430          INC     AL
ECD4                2431  J6:
ECD4 0CD8           2432          OR      AL,8                    ; TURN ON INTERRUPT ENABLE
ECD6 EE             2433          OUT     DX,AL                   ; RESET THE ADAPTER
ECD7 C6063E0000     2434          MOV     SEEK_STATUS,0           ; SET RECAL REQUIRED ON ALL DRIVES
ECDC C606410000     2435          MOV     DISKETTE_STATUS,0       ; SET OK STATUS FOR DISKETTE
ECE1 0C04           2436          OR      AL,4                    ; TURN OFF RESET
ECE3 EE             2437          OUT     DX,AL                   ; TURN OFF THE RESET
ECE4 FB             2438          STI                             ; REENABLE THE INTERRUPTS
ECE5 E82A02         2439          CALL    CHK_STAT_2              ; DO SENSE INTERRUPT STATUS
                    2440                                          ;  FOLLOWING RESET
ECE8 A04200         2441          MOV     AL,NEC_STATUS           ; IGNORE ERROR RETURN AND DO OWN TEST
ECEB 3CC0           2442          CMP     AL,0C0H                 ; TEST FOR DRIVE READY TRANSITION
ECED 7406           2443          JZ      J7                      ; EVERYTHING OK
ECEF 800E410020     2444          OR      DISKETTE_STATUS,BAD_NEC ; SET ERROR CODE
ECF4 C3             2445          RET
                    2446
                    2447  ;----- SEND SPECIFY COMMAND TO NEC
                    2448
ECF5                2449  J7:                                     ; DRIVE_READY
ECF5 B403           2450          MOV     AH,03H                  ; SPECIFY COMMAND
ECF7 E84701         2451          CALL    NEC_OUTPUT              ; OUTPUT THE COMMAND
ECFA B80100         2452          MOV     BX,1                    ; FIRST BYTE PARM IN BLOCK
ECFD E86C01         2453          CALL    GET_PARM                ;  TO THE NEC CONTROLLER
ED00 BB0300         2454          MOV     BX,3                    ; SECOND BYTE PARM IN BLOCK
ED03 E86601         2455          CALL    GET_PARM                ;  TO THE NEC CONTROLLER
ED06                2456  J8:                                     ; RESET_RET
ED06 C3             2457          RET                             ; RETURN TO CALLER
                    2458  DISK_RESET      ENDP
                    2459
```

Commentaires.- 1^o) On place le port de contrôle du contrôleur de lecteur de disquette dans le registre DX (ligne 2415). Remarquez que celui-ci apparaît, une fois encore, comme un nombre magique. On ne permet pas les interruptions masquables (ligne 2416).

- 2^o) On place le numéro de lecteur de disquette actif, sous forme codée, dans le registre AL (ligne 2417). La variable MOTOR_STATUS est définie dans la zone de communication du BIOS (ligne 140). On place ce code dans le demi-octet de poids fort de AL (lignes 2418 et 2419). La faute d'orthographe « NYBBLE » apparaît bien dans le code. Le numéro de lecteur est placé, sous forme non codée, dans le demi-octet de poids faible de AL (lignes 2420 à 2431).

- 3^o) Le bit 7 de MOTOR_STATUS est positionné lorsqu'il s'agit d'une opération d'écriture (qui exige un délai). On le positionne (ligne 2432).

- 4^o) On réinitialise le contrôleur en envoyant l'octet adéquat au port de contrôle (ligne 2433).

- 5^o) On initialise les variables SEEK_STATUS et DISKETTE_STATUS de la zone de communication du BIOS (lignes 2434 et 2435).

- 6^o) On active le contrôleur (lignes 2436 et 2437). On permet à nouveau les interruptions masquables (ligne 2438).

- 7^o) On fait appel à la sous-routine CHK_STAT_2 (définie ligne 2885 et étudiée ci-dessous) pour intercepter l'interruption reçue après un recalibrage (lignes 2439 et 2440), qui positionne l'indicateur CF à 1 en cas d'erreur. Si une erreur se produit lors de l'exécution de cette sous-routine, on spécifie sa nature dans la variable DISKETTE_STATUS et on retourne à l'appelant (lignes 2441 à 2445).

- 8^o) Si le lecteur est prêt, on envoie la commande de réinitialisation au contrôleur de disquette (lignes 2449 à 2451), on récupère les octets 0 et 1 de la phase de résultats (lignes 2452 à 2455), dont on ne fait rien, et on retourne à l'appelant (ligne 2457).

10.4.3.2 La sous-routine d'attente d'une interruption

Nous avons vu, lors de la description des commandes du contrôleur de disquette, que certaines commandes lèvent une interruption pour indiquer que celle-ci a été exécutée. La sous-routine WAIT_IN d'attente d'une telle interruption commence à la ligne 2916 du listing :

```
                    2904  ;--------------------------------------------------------------------------
                    2905  ; WAIT_IN                                                                 :
                    2906  ;       THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR. A TIME OUT      :
                    2907  ;       ROUTINE TAKE PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE   :
                    2908  ;       RETURNED IF THE DRIVE IS NOT READY.                          :
                    2909  ; INPUT                                                                  :
                    2910  ;       NONE                                                         :
                    2911  ; OUTPUT                                                                 :
                    2912  ;       CY = 0 SUCCESS                                               :
                    2913  ;       CY = 1 FAILURE -- DISKETTE_STATUS IS SET ACCORDINGLY          :
                    2914  ;       (AX) DESTROYED                                               :
                    2915  ;--------------------------------------------------------------------------
EF33                2916  WAIT_IN         PROC    NEAR
EF33 FB             2917          STI                             ; TURN ON INTERRUPT, JUST IN CASE
EF34 53             2918          PUSH    BX
EF35 51             2919          PUSH    CX                      ; SAVE REGISTERS
EF36 B302           2920          MOV     BL,2                    ; CLEAR THE COUNTERS
EF38 33C9           2921          XOR     CX,CX                   ; FOR 2 SECOND WAIT
EF3A                2922  J36:
EF3A F6063E0080     2923          TEST    SEEK_STATUS,INT_FLAG    ; TEST FOR INTERRUPT OCCURING
EF3F 750C           2924          JNZ     J37
EF41 E2F7           2925          LOOP    J36                     ; COUNT DOWN WHILE WAITING
EF43 FECB           2926          DEC     BL                      ; SECOND LEVEL COUNTER
EF45 75F3           2927          JNZ     J36
EF47 800E410080     2928          OR      DISKETTE_STATUS,TIME_OUT ; NOTHING HAPPENED
EF4C F9             2929          STC                             ; ERROR RETURN
EF4D                2930  J37:
EF4D 9C             2931          PUSHF                           ; SAVE CURRENT CARRY
EF4E B0263E007F     2932          AND     SEEK_STATUS,NOT INT_FLAG        ; TURN OFF INTERRUPT FLAG
EF53 90             2933          POPF                            ; RECOVER CARRY
EF54 59             2934          POP     CX
EF55 5B             2935          POP     BX                      ; RECOVER REGISTERS
EF56 C3             2936          RET                             ; GOOD RETURN CODE COMES
                    2937                                          ;  FROM TEST INST
                    2938  WAIT_INT        ENDP
```

Commentaires.- 1^o) On permet les interruptions masquables, si cela n'avait pas été fait (ligne 2917), on sauvegarde les contenus des registres BX et CX, puisqu'on va les utiliser (lignes 2918, 2919, 2934 et 2945). On initialise les registres BX et CX pour une attente d'au plus 2 secondes (ligne 2921) et on attend que l'interruption se manifeste durant ce temps (lignes 2922 à 2927).

- 2^o) Si aucune interruption n'est survenue durant ce temps, on positionne l'indicateur CF (ligne 2929). Dans tous les cas on change le bit INT_FLAG de la variable SEEK_STATUS et on retourne à l'appelant (lignes 2931 à 2936).

10.4.3.3 La sous-routine de lecture des résultats d'une commande

La sous-routine de lecture des résultats après une commande effectuée au contrôleur de disquette RESULTS commence à la ligne 2972 du listing :

```
                        2960  ;--------------------------------------------------------------------------
                        2961  ; RESULTS                                                                 :
                        2962  ;       THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER HAS        :
                        2963  ;       TO SAY FOLLOWING AN INTERRUPT.                                    :
                        2964  ; INPUT                                                                   :
                        2965  ;       NONE                                                              :
                        2966  ; OUTPUT                                                                  :
                        2967  ;       CY = 0 SUCCESSFUL TRANSFER                                        :
                        2968  ;       CY = 1 FAILURE -- TIME OUT IN WAITING FOR STATUS                  :
                        2969  ;       NEC_STATUS AREA HAS STATUS BYTE LOADED INTO IT                     :
                        2970  ;       (AH) DESTROYED                                                    :
                        2971  ;--------------------------------------------------------------------------
EF69                    2972  RESULTS PROC    NEAR
EF69 FC                 2973          CLD
EF6A BF4200             2974          MOV     DI,OFFSET NEC_STATUS    ; POINTER TO DATA AREA
EF6D 51                 2975          PUSH    CX                      ; SAVE COUNTER
EF6E 52                 2976          PUSH    DX
EF6F 53                 2977          PUSH    BX
EF70 B307               2978          MOV     BL,7                    ; MAX STATUS BYTES
                        2979
                        2980  ;----- WAIT FOR REQUEST FOR MASTER
                        2981
EF72                    2982  J38:                                    ; INPUT_LOOP
EF72 33C9               2983          XOR     CX,CX                   ; COUNTER
EF74 BAF403             2984          MOV     DX,03F4H                ; STATUS PORT
EF77                    2985  J39:                                    ; WAIT FOR MASTER
EF77 EC                 2986          IN      AL,DX                   ; GET STATUS
EF78 A880               2987          TEST    AL,080H                 ; MASTER READY
EF7A 750C               2988          JNZ     J40A                    ; TEST_DIR
EF7C E2F9               2989          LOOP    J39                     ; WAIT_MASTER
EF7E 800E410080         2990          OR      DISKETTE_STATUS,TIME_OUT
EF83                    2991  J40:                                    ; RESULTS_ERROR
EF83 F9                 2992          STC                             ; SET ERROR RETURN
EF84 5B                 2993          POP     BX
EF85 5A                 2994          POP     DX
EF86 59                 2995          POP     CX
EF87 C3                 2996          RET
                        2997
                        2998  ;----- TEST THE DIRECTION BIT
                        2999
EF88                    3000  J40A:
EF88 EC                 3001          IN      AL,DX                   ; GET STATUS REG AGAIN
EF89 A840               3002          TEST    AL,040H                 ; TEST DIRECTION BIT
EF8B 7507               3003          JNZ     J42                     ; OK TO READ STATUS
EF8D                    3004  J41:                                    ; NEC_FAIL
EF8D 800E410020         3005          OR      DISKETTE_STATUS,BAD_NEC
EF92 EBEF               3006          JMP     J40                     ; RESULTS_ERROR
                        3007
                        3008  ;----- READ IN THE STATUS
                        3009
EF94                    3010  J42:                                    ; INPUT_STAT
EF94 42                 3011          INC     DX                      ; POINT AT DATA PORT
EF95 EC                 3012          IN      AL,DX                   ; GET THE DATA
EF96 8805               3013          MOV     [DI],AL                 ; STORE THE BYTE
EF98 47                 3014          INC     DI                      ; INCREMENT THE POINTER
EF99 B90A00             3015          MOV     CX,10                   ; LOOP TO KILL TIME FOR NEC
EF9C E2FE               3016  J43:    LOOP    J43
EF9E 4A                 3017          DEC     DX                      ; POINT AT STATUS PORT
EF9F EC                 3018          IN      AL,DX                   ; GET STATUS
EFA0 A810               3019          TEST    AL,010H                 ; TEST FOR NEC STILL BUSY
EFA2 7406               3020          JZ      J44                     ; RESULTS DONE
EFA4 FECB               3021          DEC     BL                      ; DECREMENT THE STATUS COUNTER
EFA6 75CA               3022          JNZ     J38                     ; GO BACK FOR MORE
EFA8 EBE3               3023          JMP     J41                     ; CHIP HAS FAILED
                        3024
                        3025  ;----- RESULT OPERATION IS DONE
                        3026
EFAA                    3027  J44:
EFAA 5B                 3028          POP     BX
```

```
EFAB 5A                 3029            POP     DX
EFAC 59                 3030            POP     CX                      ; RECOVER REGISTERS
EFAD C3                 3031            RET                             ; GOOD RETURN CODE FROM TEST INST
                        3032    ;---------------------------------------------------------------
                        3033    ; NUM_TRANS                                                     :
                        3034    ;       THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT       :
                        3035    ;       WERE ACTUALLY TRANSFERRED TO/FROM THE DISKETTE           :
                        3036    ; INPUT                                                         :
                        3037    ;       (CH) = CYLINDER OF OPERATION                            :
                        3038    ;       (CL) = START SECTOR OF OPERATION                        :
                        3039    ; OUTPUT                                                        :
                        3040    ;       (AL) = NUMBER ACTUALLY TRANSFERRED                      :
                        3041    ;       NO OTHER REGISTERS MODIFIED                             :
                        3042    ;---------------------------------------------------------------
EFAE                    3043    NUM_TRANS       PROC    NEAR
EFAE A04500             3044            MOV     AL,NEC_STATUS+3         ; GET CYLINDER ENDED UP ON
EFB1 3AC5               3045            CMP     AL,CH                   ; SAME AS WE STARTED
EFB3 A04700             3046            MOV     AL,NEC_STATUS+5         ; GET ENDING SECTOR
EFB6 7A0A               3047            JZ      J45                     ; IF ON SAME CYL, THEN NO ADJUST
EFB8 BB0800             3048            MOV     BX,8
EFBB E8AEFE             3049            CALL    GET_PARM                ; GET EOT VALUE
EFBE 8AC4               3050            MOV     AL,AH                   ;  INTO AL
EFC0 FEC0               3051            INC     AL                      ; USE EOT+1 FOR CALCULATION
EFC2                    3052    J45:
EFC2 2AC1               3053            SUB     AL,CL                   ; SUBSTRACT START FROM END
EFC4 C3                 3054            RET
                        3055    NUM_TRANS       ENDP
                        3056    RESULTS ENDP
```

Commentaires.- 1^o) On positionne le drapeau de direction à zéro (ligne 2973) et on fait pointer le registre DI sur la zone de communication du BIOS concernant le lecteur de disquette (ligne 2974). On sauvegarde les registres (lignes 2975 à 2977 et 3028 à 3030 ; bien entendu « SAVE COUNTER » est une bévue). On place le nombre maximum d'octets de statut dans BL.

- 2^o) On attend de pouvoir lire les résultats de la commande. Pour cela, on initialise le compteur à zéro (ligne 2983), DX avec le port du registre de statut du contrôleur de lecteur de disquette (ligne 2984), on lit l'octet de statut (ligne 2986), on teste si le registre des données est prêt (ligne 2987) et on recommence jusqu'à ce qu'il le soit (ligne 2989) ou que le temps imparti soit dépassé. Dans ce dernier cas, on l'indique dans la variable adéquate (ligne 2990), on positionne l'indicateur CF (ligne 2992) et on retourne à l'appelant (lignes 2993 à 2996).

- 3^o) Lorsque le registre des données est disponible (ligne 2988), on teste le bit de direction. Pour cela on lit à nouveau le registre de données (ligne 3001), on teste si le bit de direction est présent (ligne 3002). Si ce n'est pas le cas, on l'indique dans la variable adéquate (ligne 3005), on positionne l'indicateur CF et on retourne à l'appelant (ligne 3006).

- 4^o) Si le bit de direction est positionné (ligne 3003), on lit le premier octet de résultat. Pour cela, on fait pointer DX sur le port du registre des données du contrôleur de lecteur de disquette (ligne 3011) et on lit le premier octet des résultats de la commande (ligne 3012), que l'on place dans la zone de communication du BIOS à l'endroit adéquat (ligne 3013).

- 5^o) On incrémente le pointeur (ligne 3014) de façon à se trouver à l'emplacement du deuxième octet de la zone de communication, on effectue une boucle pour laisser le temps au contrôleur de présenter le deuxième octet (lignes 3015 et 3016), on fait pointer DX sur le port de statut du contrôleur de disquette (ligne 3017) et on lit le statut (ligne 3018). Si tous les octets de résultat ont été lus (lignes 3019 et 3020), on restaure les valeurs des registres et on retourne à l'appelant (lignes 3027 à 3031).

- 6^o) Si tous les octets de résultat n'ont pas été lus, on décrémente le compteur du nombre maximum d'octets de résultats restant à lire (ligne 3021) et on recommence pour lire le suivant (ligne 3022). Si on en a lu le nombre maximum mais qu'il en reste encore à lire, il y a un problème avec le contrôleur que l'on indique (ligne 3023).

10.4.3.4 La sous-routine de vérification

La sous-routine CHK_STAT_2 commence à la ligne 2885 du listing :

```
                        2872  ;--------------------------------------------------------------------
                        2873  ; CHK_STAT_2                                                        :
                        2874  ;       THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER A          :
                        2875  ;       RECALIBRATE, SEEK, OR RESET TO THE ADAPTER.                  :
                        2876  ;       THE INTERRUPT IS WAITED FOR, THE INTERRUPT STATUS SENSED,    :
                        2877  ;       AND THE RESULT RETURNED TO CALLER.                          :
                        2878  ; INPUT                                                             :
                        2879  ;       NONE                                                        :
                        2880  ; OUTPUT                                                            :
                        2881  ;       CY = 0 SUCCESS                                              :
                        2882  ;       CY = 1 FAILURE -- ERROR IS IN DISKETTE_STATUS                :
                        2883  ;       (AX) DESTROYED                                              :
                        2884  ;--------------------------------------------------------------------
EF12                    2885  CHK_STAT_2      PROC    NEAR
EF12 E81E00             2886          CALL    WAIT_IN                 ; WAIT FOR THE INTERRUPT
EF15 7214               2887          JC      J34                     ; IF ERROR, RETURN IT
EF17 B408               2888          MOV     AH,08H                  ; SENSE INTERRUPT STATUS COMMAND
EF19 E825FF             2889          CALL    NEC_OUTPUT
EF1C E84A00             2890          CALL    RESULTS                 ; READ IN THE RESULTS
EF1F 7204               2891          JC      J34                     ; CHK2_RETURN
EF21 A04200             2892          MOV     AL,NEC_STATUS           ; GET THE FIRST STATUS BYTE
EF24 2460               2893          AND     AL,060H                 ; ISOLATE THE BITS
EF26 3C60               2894          CMP     AL,060H                 ; TEST FOR CORRECT VALUE
EF28 7402               2895          JZ      J35                     ; IF ERROR, GO MARK IT
EF2A F8                 2896          CLC                             ; GOOD RETURN
EF2B                    2897  J34:
EF2B C3                 2898          RET                             ; RETURN TO CALLER
EF2C                    2899  J35:
EF2C 800E410040         2900          OR      DISKETTE_STATUS,BAD_SEEK
EF31 F9                 2901          STC                             ; ERROR RETURN CODE
EF32 C3                 2902          RET
                        2903  CHK_STAT_2      ENDP
```

Commentaires.- 1^o) On attend une interruption pendant au plus 2 secondes (ligne 2886). Si celle-ci ne survient pas, on retourne à l'appelant (lignes 2887, 2897 et 2898) en le lui l'indiquant en positionnant l'indicateur CF.

- 2^o) Lorsque l'interruption survient, on envoie la commande de statut d'interruption au contrôleur de lecteur de disquette (lignes 2888 et 2889). On essaie de lire les résultats (ligne 2890). Si on n'y parvient pas, on retourne à l'appelant en le spécifiant en positionnant l'indicateur CF (ligne 2891).

- 3^o) Sinon on regarde sur le premier octet de résultat s'il y a eu une erreur (lignes 2892 à 2894). Si c'est le cas, on l'indique dans la variable DISKETTE_STATUS (lignes 2895 et 2896), on positionne l'indicateur CF (ligne 2901) et on retourne à l'appelant (ligne 2902). Sinon on efface l'indicateur CF (ligne 2896) et on retourne à l'appelant (ligne 2898).

10.4.4 Statut

La fonction 1 de demande de statut de INT 13h commence à la ligne 2462 du listing :

```
                        2460  ;----- DISKETTE STATUS ROUTINE
                        2461
ED07                    2462  DISK_STATUS     PROC    NEAR
ED07 A04100             2463          MOV     AL,DISKETTE_STATUS
ED0A C3                 2464          RET
                        2465  DISK_STATUS     ENDP
                        2466
```

Elle consiste tout simplement à placer le contenu de la variable DISKETTE_STATUS de la zone de communication du BIOS dans le registre AL.

10.4.5 Lecture

10.4.5.1 La sous-routine principale

La fonction 2 de lecture de secteurs de INT 13h commence à la ligne 2469 du listing :

```
                    2467  ;----- DISKETTE READ
                    2468
ED0B                2469  DISK_READ       PROC    NEAR
ED0B B046           2470          MOV     AL,046H                 ; READ COMMAND FOR DMA
ED0D                2471  J9:                                     ; DISK_READ_CONT
ED0D E88801         2472          CALL    DMA_SETUP               ; SET UP THE DMA
ED10 B4E6           2473          MOV     AH,0E6H                 ; SET UP RD COMMAND FOR NEC CONTROLLER
ED12 EB36           2474          JMP     SHORT RW_OPN            ; GO TO THE OPERATION
                    2475  DISK_READ       ENDP
                    2476
```

Commentaires.- 1^o) On envoie la commande de DMA adéquate (lignes 2470 à 2472) en faisant appel à la sous-routine DMA_SETUP, définie ligne 2821 et étudiée ci-après.

- 2^o) On envoie la commande de lecture (lignes 2473 et 2474) en faisant appel à la sous-routine RW_OPN, définie ligne 2518 et étudiée ci-après.

10.4.5.2 La sous-routine de préparation du DMA

La sous-routine DMA_SETUP commence à la ligne 2821 du listing :

```
                    2812  ;--------------------------------------------------------------------------
                    2813  ; DMA_SETUP                                                              :
                    2814  ;       THIS ROUTINE SETS UP THE DMA FOR READ/WRITE/VERIFY OPERATIONS.  :
                    2815  ; INPUT                                                                  :
                    2816  ;       (AL) = MODE BYTE FOR THE DMA                                     :
                    2817  ;       (ES:BX) - ADDRESS TO READ/WRITE THE DATA                         :
                    2818  ; OUTPUT                                                                 :
                    2819  ;       (AX) DESTROYED                                                   :
                    2820  ;--------------------------------------------------------------------------
EEC8                2821  DMA_SETUP       PROC    NEAR
EEC8 51             2822          PUSH    CX                      ; SAVE THE REGISTER
EEC9 FA             2823          CLI                             ; NO MORE INTERRUPTS
EECA E60C           2824          OUT     DMA+12,AL               ; SET THE FIRST/LAST F/F
EECC 50             2825          PUSH    AX
EECD 58             2826          POP     AX
EECE E608           2827          OUT     DMA+11,AL               ; OUTPUT THE MODE BYTE
EED0 8CC0           2828          MOV     AX,ES                   ; GET THE ES VALUE
EED2 8104           2829          MOV     CL,4                    ; SHIFT COUNT
EED4 03C3           2830          ROL     AX,CL                   ; ROTATE LEFT
EED6 8AE8           2831          MOV     CH,AL                   ; GET HIGHEST NYBLE OF ES TO CH
EED8 24F0           2832          AND     AL,0F0H                 ; ZERO THE LOW NYBBLE FROM SEGMENT
EEDA 03C3           2833          ADD     AX,BX                   ; TEST FOR CARRY FROM ADDITION
EEDC 7302           2834          JNC     J33
EEDE FEC5           2835          INC     CH                      ; CARRY MEANS HIGH 4 BITS MUST BE INC
EEE0                2836  J33:
EEE0 50             2837          PUSH    AX                      ; SAVE START ADDRESS
EEE1 E604           2838          OUT     DMA+4,AL                ; OUTPUT LOW ADDRESS
EEE3 8AC4           2839          MOV     AL,AH
EEE5 E604           2840          OUT     DMA+4,AL                ; OUTPUT HIGH ADDRESS
EEE7 8AC5           2841          MOV     AL,CH                   ; GET HIGH 4 BITS
EEE9 240F           2842          AND     AL,0FH
EEEB E681           2843          OUT     081H,AL                 ; OUTPUT THE HIGH 4 BITS TO
                    2844                                          ;  THE PAGE REGISTER
                    2845
                    2846  ;----- DETERMINE COUNT
                    2847
EEED 8AE6           2848          MOV     AH,DH                   ; NUMBER OF SECTORS
EEEF 2AC0           2849          SUB     AL,AL                   ;  TIMES 256 INTO AX
EEF1 D1E6           2850          SHR     AX,1                    ; SECTORS * 128 INTO AX
EEF3 50             2851          PUSH    AX
EEF4 880600         2852          MOV     BX,6                    ; GET THE BYTES/SECTOR PARM
EEF7 E872FF         2853          CALL    GET_PARM
EEFA 8ACC           2854          MOV     CL,AH                   ; USE AS SHIFT COUNT (0=128, 1=256 ETC)
```

```
EEFC 58              2855          POP     AX
EEFD D3E0            2856          SHL     AX,CL                ; MULTIPLY BY CORRECT AMOUNT
EEFF 48              2857          DEC     AX                   ; -1 FOR DMA VALUE
EF00 50              2858          PUSH    AX                   ; SAVE COUNT VALUE
EF01 E605            2859          OUT     DMA+5,AL             ; LOW BYTE OF COUNT
EF03 8AC4            2860          MOV     AL,AH
EF05 E605            2861          OUT     DMA+5,AL             ; HIGH BYTE OF COUNT
EF07 FB              2862          STI                          ; INTERRUPTS BACK ON
EF08 59              2863          POP     CX                   ; RECOVER COUNT VALUE
EF09 58              2864          POP     AX                   ; RECOVER ADDRESS VALUE
EF0A 03C1            2865          ADD     AX,CX                ; ADD, TEST FOR 64K OVERFLOW
EF0C 59              2866          POP     CX                   ; RECOVER REGISTER
EF0D B002            2867          MOV     AL,2                 ; MODE FOR 8237
EF0F E60A            2868          OUT     DMA+10,AL            ; INITIALIZE THE DISKETTE CHANNEL
EF11 C3              2869          RET                          ; RETURN TO CALLER,
                     2870                                       ;  CFL SET BY ABOVE IF ERROR
                     2871   DMA_SETUP      ENDP
```

Commentaires.- 1^o) On sauvegarde le registre CX (ligne 2822) et on ne permet pas les interruptions masquables (ligne 2823). On envoie l'octet de mode pour le DMA (ligne 2824), on attend un peu (lignes 2825 et 2826) et on l'envoie à nouveau (ligne 2827).

- 2^o) Puisque l'adresse de début du DMA est constituée d'un déclage et d'une page de registre, on calcule ceux-ci. Pour cela on place l'adresse de segment de lecture-écriture dans le registre AX (ligne 2828), on initialise le compteur de décalage à 4 (ligne 2829) et on effectue une rotation à gauche (ligne 2830). On sauvegarde le demi-octet de poids fort de ES dans le registre CH (ligne 2831) et on initialise à zéro le demi-octet de poids faible (ligne 2832). On lui ajoute le contenu de BX pour obtenir l'adresse (ligne 2833). S'il y a une retenue, on en tient compte dans CH (lignes 2834 et 2835). On sauvegarde l'adresse de début ainsi déterminée (ligne 2837).

- 3^o) On envoie l'octet de poids faible du décalage de l'adresse de début au contrôleur DMA (ligne 2838) puis son octet de poids fort (lignes 2839 et 2840) et enfin le demi-octet au registre de page (lignes 2841 à 2842).

- 4^o) On détermine le nombre d'octets à transférer. Pour cela, on place le nombre de secteurs (passé en paramètre *via* le registre DH) dans AX et on le muliplie par 128 (lignes 2848 à 2850). On en sauvegarde la valeur (ligne 2851). On récupère le nombre d'octets par secteur dans le registre AH (lignes 2852 à 2853) et on le place dans CL (ligne 2854). On récupère le nombre obtenu dans AX (ligne 2855), que l'on multiplie par ce que l'on vient d'obtenir (ligne 2856) pour avoir le nombre d'octets. On décrémente le résultat de 1 pour le DMA (ligne 2857) puisqu'on commence à zéro. On sauvegarde (ligne 2858).

- 5^o) On envoie l'octet de poids faible au contrôleur de DMA (ligne 2859) puis l'octet de poids fort (lignes 2860 et 2861). On permet à nouveau les interruptions masquables (ligne 2862), on récupère le nombre d'octets (ligne 2863), la valeur de l'adresse de début (ligne 2864), on les ajoute (ligne 2865), on récupère la valeur de CX (ligne 2866) et on envoie le mode au contrôleur de DMA (lignes 2867 et 2868).

10.4.5.3 La sous-routine de positionnement sur une piste

La sous-routine SEEK de positionnement sur une piste donnée d'un lecteur de disquette donné commence à la ligne 2764 du listing :

```
                    2751  ;--------------------------------------------------------------------------
                    2752  ; SEEK                                                                    :
                    2753  ;       THIS ROUTINE WILL MOVE THE HEAD ON THE NAMED DRIVE TO THHE      :
                    2754  ;       NAMED TRACK.  IF THE DRIVE HAS NOT BEEN ACCESSED SINCE THE      :
                    2755  ;       DRIVE RESET COMMAND WAS ISSUED, THE DRIVE WILL BE RECALUBRATED. :
                    2756  ; INPUT                                                                   :
                    2757  ;       (DL) = DRIVE TO SEEK ON                                         :
                    2758  ;       (CH) = TRACK TO SEEK TO                                         :
                    2759  ; OUTPUT                                                                  :
                    2760  ;       CY = 0 SUCCESS                                                  :
                    2761  ;       CY = 1 FAILURE -- DISKETTE_STATUS SET ACCORDINGLY               :
                    2762  ;       (AX) DESTROYED                                                  :
                    2763  ;--------------------------------------------------------------------------
EE7D                2764  SEEK    PROC    NEAR
EE7D B001           2765          MOV     AL,1                    ; ESTABLISH MASK FOR RECAL TEST
EE7F 51             2766          PUSH    CX                      ; SAVE INPUT VALUES
EE80 8ACA           2767          MOV     CL,DL                   ; GET DRIVE VALUE INTO CL
EE82 D2C0           2768          ROL     AL,CL                   ; SHIFT IT BY THE DRIVE VALUE
EE84 59             2769          POP     CX                      ; RECOVER TRACK VALUE
EE85 84063E00       2770          TEST    AL,SEEK_STATUS          ; TEST FOR RECAL REQUIRED
EE89 7513           2771          JNZ     J28                     ; NO_RECAL
EE8B 08063E00       2772          OR      SEEK_STATUS,AL          ; TURN ON THE NO RECAL BIT IN FLAG
EE8F B407           2773          MOV     AH,07H                  ; RECALIBRATE COMMAND
EE91 EBADFF         2774          CALL    NEC_OUTPUT
EE94 8AE2           2775          MOV     AH,DL
EE96 E8A8FF         2776          CALL    NEC_OUTPUT              ; OUTPUT THE DRIVE NUMBER
EE99 E87600         2777          CALL    CHK_STAT_2              ; GET THE INTERRUPT AND SENSE INT STATUS
EE9C 7229           2778          JC      J32                     ; SEEK_ERROR
                    2779
                    2780  ;----- DRIVE IS IN SYNCH WITH CONTROLLER, SEEK TO TRACK
                    2781
EE9E                2782  J28:
EE9E B40F           2783          MOV     AH,0FH                  ; SEEK COMMAND TO NEC
EEA0 E89EFF         2784          CALL    NEC_OUTPUT
EEA3 8AE2           2785          MOV     AL,DL                   ; DRIVE NUMBER
EEA5 E899FF         2786          CALL    NEC_OUTPUT
EEA8 8AE5           2787          MOV     AH,CH                   ; TRACK NUMBER
EEAA E894FF         2788          CALL    NEC_OUTPUT
EEAD E86200         2789          CALL    XHK_STAT_2              ; GET ENDING INTERRUPT AND
                    2790                                          ;  SENSE STATUS
                    2791
                    2792  ;----- WAIT FOR HEAD SETTLE
                    2793
EEB0 9C             2794          PUSHF                           ; SAVE STATUS FLAGS
EEB1 BB1200         2795          MOV     BX,16                   ; GET HEAD SETTLE PARAMETER
EEB4 E8B5FF         2796          CALL    GET_PARM
EEB7 51             2797          PUSH    CX                      ; SAVE REGISTER
EEB8                2798  J29:
EEB8 892602         2799          MOV     CX,550                  ; 1 MS LOOP
EEBB 0AE4           2800          OR      AH,AH                   ; TEST FOR TIME EXPIRED
EEBD 7406           2801          JZ      J31
EEBF                2802  J30:
EEBF E2FE           2803          LOOP    J30                     ; DELAY FOR 1 MS
EEC1 FECC           2804          DEC     AH                      ; DECREMENT THE COUNT
EEC3 EBF3           2805          JMP     J29                     ; DO IT SOME MORE
EEC5                2806  J31:
EEC5 59             2807          POP     CX                      ; RECOVER STATE
EEC6 90             2808          POPF
EEC7                2809  J32:                                    ; SEEK_ERRORE
EEC7 C3             2810          RET                             ; RETURN TO CALLER
                    2811  SEEK    ENDP
```

Commentaires.- 1^o) On place un masque dans le registre AL (ligne 2765), on sauvegarde les paramètres entrés (ligne 2766), on place la valeur du disque dans CL (ligne 2767), que l'on place dans AL avec le codage adéquat (ligne 2768) et on restaure la valeur de CX (ligne 2769).

- 2^o) Si un recalibrage est nécessaire (lignes 2770 et 2771), on spécifie que le recalibrage ne sera plus nécessaire (ligne 2772), on envoie la commande de recalibrage au contrôleur de lecteur de disquette (lignes 2773 à 2776) et on attend l'interruption indiquant que cela a bien été effectué (ligne 2777). Si une erreur est survenue lors du recalibrage (ligne 2778), on retourne à l'appelant (l'erreur étant spécifiée par la valeur de l'indicateur CF).

- 3^o) On envoie la commande de positionnement sur la piste désirée du disque désiré au contrôleur de lecteur de disquette (lignes 2783 à 2788) et on attend l'interruption indiquant que cela a bien été effectué (ligne 2789).

- 4^o) On laisse le temps à la tête de lecture de se positionner. Pour cela, on sauvegarde le registre des indicateurs (ligne 2794), on récupère le paramètre de positionnement (lignes 2795 et 2796), on sauvegarde les paramètres entrés (ligne 2797), on attend pendant 1 ms (lignes 2798 à 2805), on restaure les valeurs du registre CX et du registre des indicateurs (lignes 2807 et 2808) et on retourne à l'appelant (ligne 2810).

10.4.5.4 La sous-routine de lecture-écriture

La sous-routine RW_OPN commence à la ligne 2518 du listing :

```
                        2511
                        2512  ;----- ALLOW WRITE ROUTINE TO FALL INTO RW_OPN
                        2513
                        2514  ;--------------------------------------------------------------------------
                        2515  ; RW_OPN                                                                  :
                        2516  ;       THIS ROUTINE PERFORMS THE READ/WRITE/VERIFY OPERATION             :
                        2517  ;--------------------------------------------------------------------------
ED4A                    2518  RW_OPN  PROC    NEAR
ED4A 7308               2519          JNC     J11                             ; TEST FOR DMA ERROR
ED4C C606410009         2520          MOV     DISKETTE_STATUS,DMA_BOUNDARY    ; SET ERROR
ED51 B000               2521          MOV     AL,0                            ; NO SECTORS TRANSFERRED
ED53 C3                 2522          RET                                     ; RETURN TO MAIN ROUTINE
ED54                    2523  J11:                                            ; DO_RW_OPN
ED54 50                 2524          PUSH    AX                              ; SAVE THE COMMAND
                        2525
                        2526  ;----- TURN ON THE MOTOR AND SELECT THE DRIVE
                        2527
ED55 51                 2528          PUSH    CX                              ; SAVE THE T/S PARMS
ED56 8AC4               2529          MOV     CL,DL                           ; GET DRIVE NUMBER AS SHIFT COUNT
ED58 B001               2530          MOV     AL,1                            ; MASK FOR DETERMINING MOTOR BIT
ED5A 02E0               2531          SAL     AL,CL                           ; SHIFT THE MASK BIT
ED5C FA                 2532          CLI                                     ; NO INTERRUPTS WHILE DETERMINING
                        2533                                                  ;  MOTOR STATUS
ED5D C6064000FF         2534          MOV     MOTOR_COUNT,0FFH                ; SET LARGE COUNT DURING OPERATION
ED62 84063F00           2535          TEST    AL,MOTOR_STATUS                 ; TEST THAT MOTOR FOR OPERATION
ED66 7531               2536          JNZ     J14                             ; IF RUNNING, SKIP THE WAIT
ED68 80263F00F0         2537          AND     MOTOR_STATUS,0F0H               ; TURN OFF ALL MOTOR BITS
ED6D 08063F00           2538          OR      MOTOR_STATUS,AL                 ; TURN ON THE CURRENT MOTOR
ED71 FB                 2539          STI                                     ; INTERRUPTS BACK ON
ED72 B010               2540          MOV     AL,10H                          ; MASK BIT
ED74 D2E0               2541          SAL     AL,CL                           ; DEVELOP BIT MASK FOR MOTOR ENABLE
ED76 0AC2               2542          OR      AL,DL                           ; GET DRIVE SELECT BITS IN
ED78 0C0C               2543          OR      AL,0CH                          ; NO RESET, ENABLE DMA/INT
ED7A 52                 2544          PUSH    DX                              ; SAVE REG
ED7B BAF203             2545          MOV     DX,03F2H                        ; CONTROL PORT ADDRESS
ED7E EE                 2546          OUT     DX,AL
ED7F 5A                 2547          POP     DX                              ; RECOVER REGISTERS
                        2548
                        2549  ;----- WAIT FOR MOTOR IF WRITE OPERATION
                        2550
ED80 F6063F0080         2551          TEST    MOTOR_STATUS,80H                ; IS THIS A WRITE
ED85 7412               2552          JZ      J14                             ; NO, CONTINUE WITHOUT WAIT
ED87 BB1400             2553          MOV     BX,20                           ; GET THE MOTOR WAIT
ED8A E8DF00             2554          CALL    GET_PARM                        ;  PARAMETER
ED8D 0AE4               2555          OR      AL,AH                           ; TEST FOR NO WAIT
ED8F                    2556  J12:                                            ; TEST_WAIT_TIME
ED8F 7408               2557          JZ      J14                             ; EXIT WITH TIME EXPIRED
ED91 2BC9               2558          SUB     CX,CX                           ; SET UP 1/8 SECOND LOOP TIME
```

```
ED93                    2559   J13:
ED93 E2FE               2560           LOOP    J13                     ; WAIT FOR THE REQUIRED TIME
ED95 FECC               2561           DEC     AH                      ; DECREMENT TIME VALUE
ED97 EBF6               2562           JMP     J12                     ; ARE WE DONE YET
ED99                    2563   J14:                                    ; MOTOR RUNNING
ED99 FB                 2564           STI                             ; INTERRUPTS BACK ON FOR BYPASS WAIT
ED9A 59                 2565           POP     CX
                        2566
                        2567   ;----- DO THE SEEK OPERATION
                        2568
ED9B E8DF00             2569           CALL    SEEK                    ; MOVE TO CORRECT TRACK
ED9E 58                 2570           POP     AX                      ; RECOVER COMMAND
ED9F 8AFC               2571           MOV     BH,AH                   ; SAVE COMMAND IN BH
EDA1 B600               2572           MOV     DH,0                    ; SET NO SECTORS READ IN CASE OF ERROR
EDA3 724B               2573           JC      J17                     ; IF ERROR, THEN EXIT AFTER MOTOR OFF
EDA5 BEF0ED90           2574           MOV     SI,OFFSET J17           ; DUMMY RETURN ON STACK FOR NEC_OUTPUT
EDA9 56                 2575           PUSH    SI                      ;  SO THAT IT WILL RETURN TO MOTOR OFF
                        2576                                           ;  LOCATION
                        2577
                        2578   ;----- SEND OUT THE PARAMETERS TO THE CONTROLLER
                        2579
EDAA E89400             2580           CALL    NEC_OUTPUT              ; OUTPUT THE OPERATION COMMAND
EDAD 8A6601             2581           MOV     AH,[BP+1]               ; GET THE CURRENT HEAD NUMBER
EDB0 D0E4               2582           SAL     AH,1                    ; MOVE IT TO BIT 2
EDB2 D0E4               2583           SAL     AH,1
EDB4 80E404             2584           AND     AH,4                    ; ISOLATE THAT BIT
EDB7 0AE2               2585           OR      AH,DL                   ; OR IN THE DRIVE NUMBER
EDB9 E88500             2586           CALL    NEC_OUTPUT
                        2587
                        2588   ;----- TEST FOR FORMAT COMMAND
                        2589
EDBC 80FF40             2590           CMP     BH,040H                 ; IS THIS A FORMAT OPERATION
EDBF 7503               2591           JNE     J15                     ; NO. CONTINUE WITH R/W/V
EDC1 E962FF             2592           JMP     J10                     ; IF SO, HANDLE SPECIAL
EDC4                    2593   J15:
EDC4 8AE5               2594           MOV     AH,CH                   ; CYLINDER NUMBER
EDC6 E87800             2595           CALL    NEC_OUTPUT
EDC9 8A6601             2596           MOV     AH,[BP+1]               ; HEAD NUMBER FROM STACK
EDCC E87200             2597           CALL    NEC_OUTPUT
EDCF 8AE1               2598           MOV     AH,CL                   ; SECTOR NUMBER
EDD1 E86D00             2599           CALL    NEC_OUTPUT
EDD4 B80700             2600           MOV     BX,7                    ; BYTES/SECTOR PARM FROM BLOCK
EDD7 E89200             2601           CALL    GET_PARM                ;  TO THE NEC
EDDA B80900             2602           MOV     BX,9                    ; EOT PARM FROM BLOCK
EDDD E88C00             2603           CALL    GET_PARM                ;  TO THE NEC
EDE0 B80B00             2604           MOV     BX,11                   ; GAP LENGTH PARM FROM BLOCK
EDE3 E88600             2605           CALL    GET_PARM                ;  TO THE NEC
EDE6 B80D00             2606           MOV     BX,13                   ; DTL PARM FROM BLOCK
EDE9                    2607   J16:                                    ; RW_OPN_FINISH
EDE9 E88000             2608           CALL    GET_PARM                ;  TO THE NEC
EDEC 5E                 2609           POP     SI                      ; CAN NOW DISCARD THAT DUMMY
                        2610                                           ;  RETURN ADDRESS
                        2611
                        2612   ;----- LET THE OPERATION HAPPEN
                        2613
EDED E84301             2614           CALL    WAIT_IN                 ; WAIT FOR THE INTERRUPT
EDF0                    2615   J17:                                    ; MOTOR_OFF
EDF0 7245               2616           JC      J21                     ; LOOK FOR ERROR
EDF2 E87401             2617           CALL    RESULTS                 ; GET THE NEC STATUS
EDF5 723F               2618           JC      J20                     ; LOOK FOR ERROR
                        2619
                        2620   ;----- CHECK THE RESULTS RETURNED BY THE CONTROLLER
                        2621
EDF7 FC                 2622           CLD                             ; SET THE CORRECT DIRECTION
EDF8 BE4200             2623           MOV     SI,OFFSET NEC_STATUS    ; POINT TO STATUS FIELD
EDFB AC                 2624           LODS    NEC_STATUS              ; GET ST0
EDFC 24C0               2625           AND     AL,0C0H                 ; TEST FOR NORMAL TERMINATION
EDFE 743B               2626           JZ      J22                     ; OPN_OK
EE00 3C40               2627           CMP     AL,040H                 ; TEST FOR ABNORMAL TERMINATION
EE02 7529               2628           JNZ     J16                     ; NOT ABNORMAL, BAD NEC
                        2629
                        2630   ;----- ABNORMAL TERMINATION, FIND OUT WHY
                        2631
EE04 AC                 2632           LODS    NEC_STATUS              ; GET ST1
```

```
EE05 D0E0              2633          SAL     AL,1                    ; TEST FOR EOT FOUND
EE07 B404              2634          MOV     AH,RECORD_NOT_FND
EE09 7224              2635          JC      J19                     ; RW_FAIL
EE0B D0E0              2636          SAL     AL,1
EE0D D0E0              2637          SAL     AL,1                    ; TEST FOR CRC ERROR
EE0F B410              2638          MOV     AH,BAD_CRC
EE11 721C              2639          JC      J19                     ; RW_FAIL
EE13 DDE0              2640          SAL     AL,1                    ; TEST FOR DAM OVERRUN
EE15 B40B              2641          MOV     AH;BAD_DMA
EE17 7216              2642          JC      J19                     ; RW_FAIL
EE19 D0E0              2643          SAL     AL,1
EE1B D0E0              2644          SAL     AL,1                    ; TEST FOR RECORD NOT FOUND
EE1D 8404              2645          MOV     AH,RECORD_NOT_FND
EE1F 720E              2646          JC      J19                     ; RW_FAIL
EE21 D0E0              2647          SAL     AL,1
EE23 B403              2648          MOV     AH,WRITE_PROTECT        ; TEST FOR WRITE_PROTECT
EE25 7208              2649          JC      J19                     ; RW_FAIL
EE27 D0E0              2650          SAL     AL,1                    ; TEST MISSING ADDRESS MARK
EE29 B402              2651          MOV     AH,BAD_ADDR_MARK
EE2B 7202              2652          JC      J19                     ; RW_FAIL
                       2653
                       2654  ;----- NEC MUST HAVE FAILED
                       2655
EE2D                   2656  J18:                                    ; RW-NEC-FAIL
EE2D B420              2657          MOV     AH,BAD_NEC
EE2F                   2658  J19:                                    ; RW-NEC
EE2F 08264100          2659          OR      DISKETTE_STATUS,AH
EE33 E87801            2660          CALL    NUM_TRANS               ; HOW MANY WERE REALLY TRANSFERRED
EE36                   2661  J20:                                    ; RW_ERR
EE36 C3                2662          RET                             ; RETURN TO CALLER
EE37                   2663  J21:                                    ; RW_ERR_RES
EE37 E82F01            2664          CALL    RESULTS                 ; FLUSH THE RESULTS BUFFER
EE3A C3                2665          RET
                       2666
                       2667  ;----- OPERATION WAS SUCCESSFUL
                       2668
EE3B                   2669  J22:                                    ; OPN_OK
EE3B E87001            2670          CALL    NUM_TRANS               ; HOW MANY GOT MOVED
EE3E 32E4              2671          XOR     AH,AH                   ; NO ERRORS
EE40 C3                2672          RET
                       2673  RW_OPN  ENDP
```

Commentaires.- 1^o) Si la sous-routine de préparation du DMA a échouée (ligne 2519), on spécifie ce type d'erreur dans la variable adéquate de la zone de communication du BIOS consacrée au lecteur de disquette (ligne 2520), on spécifie qu'aucun secteur n'a été transféré (ligne 2521) et on retourne à l'appelant (ligne 2522).

- 2^o) Sinon on sauvegarde la commande (ligne 2524) et on démarre le moteur. Pour cela, on sauvegarde les paramètres concernant la piste et le secteur (ligne 2528), on place le numéro de lecteur comme compteur de décalage (ligne 2529), on place un 1 à titre de masque dans le registre AL (ligne 2530), que l'on déplace pour spécifier le lecteur de disquette désiré (ligne 2531), on inhibe les interruptions masquables (lignes 2532 et 2533) et on indique une durée assez longue pour l'opération (ligne 2534). Si le moteur tourne déjà, on passe cette phase de démarrage du moteur (lignes 2535 et 2536). Sinon on indique que tous les moteurs sont à l'arrêt (ligne 2537) et que le moteur choisi est en fonctionnement (ligne 2538), on permet à nouveau les interruptions masquables (ligne 2539), on spécifie le moteur qui doit tourner (lignes 2540 à 2542), qu'il ne faut pas réinitialiser mais qu'il faut utiliser le DMA (ligne 2543), on sauvegarde la valeur du registre DX (ligne 2544) que l'on va utiliser, on transmet l'octet de commande au port de contrôle du contrôleur de lecteur de disquette (lignes 2545 et 2546) et on restaure la valeur du registre DX (ligne 2547).

- 3^o) S'il s'agit d'une opération de lecture, on attend que le moteur ait bien démarré. Pour cela, on teste s'il s'agit d'une opération d'écriture (ligne 2551) et on passe cette phase si ce n'est pas le cas (ligne 2552). Sinon, on récupère dans AH les paramètres de démarrage du moteur (lignes 2553 et 2554) et on passe également cette phase si on n'a pas besoin d'attendre

(lignes 255 à 2557). Sinon on attend durant 1/8 de seconde (lignes 2558 à 2560) et on recommence autant de fois que le spécifie le paramètre (lignes 2561 et 2562).

On permet à nouveau les interruptions masquables (ligne 2564) et on restaure la valeur de CX (ligne 2565).

- 4^o) On se déplace sur la piste demandée. Pour cela on fait appel à la sous-routine SEEK (ligne 2569). On récupère la commande (ligne 2570) et on la place dans le registre BH (ligne 2571). On place 0 secteur lu dans le registre DH (ligne 2572) en cas d'erreur. Si une erreur est survenue, on sort après avoir arrêté le moteur (ligne 2573). Sinon, on place une adresse de retour sur la pile (lignes 2574 et 2575).

- 5^o) On envoie les paramètres au contrôleur (lignes 2578 à 2586).

- 6^o) Dans le cas d'une commande de formatage, on procède un peu autrement. On teste s'il s'agit d'une commande de formatage (ligne 2590) et si c'est le cas, on renvoie vers le traitement de celle-ci (ligne 2592), partie de la sous-routine DISK_FORMAT, qui commence ligne 2486.

- 7^o) S'il ne s'agit pas d'une commande de formatage, on continue en envoyant les octets de commande au contrôleur : numéro de cylindre (lignes 2594 et 2595), numéro de tête de lecture-écriture (lignes 2596 et 2597), numéro de secteur (lignes 2598 et 2599), nombre d'octets par secteur (lignes 2600 et 2601), paramètre de fin de piste (lignes 2602 et 2603), longueur du GAP (lignes 2604 et 2605), paramètre DTL (lignes 2606 à 2608). On peut ne pas tenir compte de l'adresse de retour, on la retire donc de la pile (lignes 2609 et 2610).

- 8^o) On attend que l'opération soit effectuée. Pour cela, on attend l'interruption indiquant qu'il en est ainsi (ligne 2614). Si une erreur est survenue (ligne 2616), on récupère les résultats (ligne 2664), que l'on n'utilise pas, et on retourne à l'appelant (ligne 2665), l'erreur étant indiquée par la valeur de l'indicateur CF. Sinon on récupère les résultats (ligne 2617) et on retourne à l'appelant si une erreur est survenue (lignes 2618, 2661 et 2662).

- 9^o) On vérifie les résultats renvoyés par le contrôleur. Pour cela, on se place dans la direction adéquate (ligne 2622), on récupère le contenu du registre ST0 du contrôleur de disquette (lignes 2623 et 2624). Si l'opération ne s'est pas terminée normalement (lignes 2625 et 2626), si le contrôleur n'était pas prêt, on recommence (lignes 2627 et 2628), sinon on recupère le contenu du registre ST1 du contrôleur de disquette (ligne 2632), on recherche la cause de l'échec que l'on place dans le registre AH (lignes 2633 à 2657), on l'indique dans la variable DISKETTE_STATUS (ligne 2660), on détermine combien de secteurs ont été transferrés (ligne 2660) et on retourne à l'appelant (ligne 2662).

- 10^o) Pour déterminer combien de secteurs ont été transférés, on fait appel à la sous-routine NUM_TRANS, définie curieusement ligne 3043 au milieu de la sous-routine RESULTS.

Les paramètres sont le numéro cylindre, passé *via* le registre CH, et le numéro de secteur de début, passé *via* le registre CL. On renvoie le nombre de secteurs transférés *via* le registre AL.

Pour cela on utilise la variable NEC_STATUS de la zone de communication du BIOS pour déterminer le numéro de cylindre de fin (lignes 3044 et 3045) et le numéro de secteur de fin (ligne 3046). S'il s'agit du même cylindre (ligne 3047), il suffit de soustraire le numéro de secteur de fin de celui de début (lignes 3052 et 3053) et de retourner à l'appelant (ligne 3054). Sinon, on récupère le nombre de secteurs d'une piste (lignes 3048 à 3051) et on effectue également la soustration.

- 11^o) Si tout s'est bien déroulé, on place également dans AL le nombre de secteurs transférés (lignes 2669 et 2670), on met l'indicateur CF à 0 (ligne 2671) pour indiquer qu'il n'y a pas d'erreur et on retourne à l'appelant (ligne 2672).

10.4.6 Vérification

La fonction 4 de vérification de INT 13h commence à la ligne 2479 du listing :

```
                        2477   ;----- DISKETTE VERIFY
                        2478
ED14                    2479   DISK_VERF       PROC    NEAR
ED14 B042               2480           MOV     AL,042H                 ; VERIFY COMMAND FOR DMA
ED16 EBF5               2481           JMP     J9                      ; DO AS IF DISK READ
                        2482   DISK_VERF       ENDP
                        2483
```

Autrement dit on prépare la commande de vérification (ligne 2480) et on fait comme pour la lecture (ligne 2481) en renvoyant à la ligne 2471 de la sous-routine pour la lecture.

10.4.7 Formatage

La fonction 5 de formatage de INT 13h commence à la ligne 2486 du listing :

```
                        2484   ;----- DISKETTE FORMAT
                        2485
ED18                    2486   DISK_FORMAT     PROC    NEAR
ED18 800E3F0080         2487           OR      MOTOR_STATUS,80H        ; INDICATE WRITE OPERATION
ED1D B04A               2488           MOV     AL,04AH                 ; WILL WRITE TO THE DISKETTE
ED1F E8A601             2489           CALL    DMA_SETUP               ; SET UP THE DMA
ED22 B440               2490           MOV     AH,040H                 ; ESTABLISH THE FORMAT COMMAND
ED24 EB24               2491           JMP     SHORT RW_OPN            ; DO THE OPERATION
ED26                    2492   J10:                                    ; CONTINUATION OF RW_OPN FOR FMT
ED26 BB0700             2493           MOV     BX,7                    ; GET THE
ED29 E84001             2494           CALL    GET_PARM                ;  BYTES/SECTOR VALUE TO NEC
ED2C BB0900             2495           MOV     BX,9                    ; GET THE
ED2F E83A01             2496           CALL    GET_PARM                ;  SECTORS/TRACK VALUE TO NEC
ED32 BB0F00             2497           MOV     BX,15                   ; GET THE
ED35 E83401             2498           CALL    GET_PARM                ;  GAP LENGTH VALUE TO NEC
ED38 BB1100             2499           MOV     BX,17                   ; GET THE FILLER BYTE
ED3B E9AB00             2500           JMP     J16                     ;  TO THE CONTROLLER
                        2501   DISK_FORMAT     ENDP
                        2502
```

Commentaires.- 1^o) On indique à la variable MOTOR_STATUS de la zone de communication du BIOS que l'on va effectuer une opération d'écriture (ligne 2487), ainsi que pour le DMA (lignes 2488 et 2489), on choisit la commande de formatage du contrôleur de lecteur de disquette (ligne 2490) et on effectue l'opération en faisant appel à la sous-routine RW_OPN (ligne 2491).

- 2^o) Nous avons vu, lors de l'étude de la sous-routine RW_OPN, que, dans le cas d'un formatage, la ligne 2592 renvoie à J10 (c'est-à-dire à la ligne 2492). On envoie au contrôleur le nombre d'octets par secteur (lignes 2493 et 2494), le nombre de secteurs par pistes (lignes 2495 et 2496), la taille du « gap » (lignes 2497 et 2498), on place l'octet de remplissage dans le registre BX (ligne 2499) et on retourne à la sous-routine RW_OPN (ligne 2500, renvoyant à la ligne 2607).

10.4.8 Écriture

La fonction 3 d'écriture de INT 13h commence à la ligne 2505 du listing :

```
                        2503   ;----- DISKETTE WRITE ROUTINE
                        2504
ED3E                    2505   DISK_WRITE      PROC    NEAR
ED3E 800E3F0080         2506           OR       MOTOR_STATUS,80H       ; INDICATE WRITE OPERATION
ED43 B04A               2507           MOV      AL,04AH                ; DMA WRITE COMMAND
ED45 E88001             2508           CALL     DMA_SETUP
ED48 B4C5               2509           MOV      AH,0C5H                ; NEC COMMAND TO WRITE TO DISKETTE
                        2510   DISK_WRITE      ENDP
                        2511
```

Autrement dit, on indique à la variable MOTOR_STATUS de la zone de communication du BIOS que l'on va effectuer une opération d'écriture (ligne 2506), ainsi que pour le DMA (lignes 2507 et 2508), on choisit la commande d'écriture du contrôleur de lecteur de disquette (ligne 2509) et on effectue l'opération en faisant appel à la sous-routine RW_OPN (non indiqué explicitement mais c'est la sous-routine qui suit celle en cours).

10.5 Historique

Le lecteur de disquette (FDD) est inventé chez IBM par Alan Shugart en 1967. La disquette a alors une taille de 8 pouces (200 mm) de diamètre et une capacité de un MiO. Elle est commercialisée en 1971. Les disquettes et les lecteurs de disquette sont produits et améliorés par IBM, *Memorex*, *Shugart Associates* et *Burroughs Corporation*. Le terme « floppy disk » apparaît sous forme imprimée en 1970 et, bien qu'IBM annonce en 1973 son premier produit en tant que « Type 1 Diskette », l'industrie continue à utiliser les termes « floppy disk » ou « floppy ».

En 1976, *Shugart Associates* introduit le lecteur de disquette 5 pouce 1/4 d'une capacité de 160 KiO. Il est utilisé sur le premier IBM PC en août 1981 avec une capacité de 360 KiO.

10.6 Bibliographie

[MES-02] Messmer, Hans-Peter, **The Indispensable PC Hardware Book**, Addison-Wesley, fourth edition, 2002, xx + 1273 p.