Troisième partie

Le BIOS et le démarrage de l'ordinateur

Chapitre 11

Le BIOS et le démarrage de l'ordinateur

Nous allons illustrer la notion de BIOS par le BIOS du tout premier IBM-PC.

11.1 Démarrage d'un ordinateur et ROM BIOS

11.1.1 Rappel sur le démarrage d'un microprocesseur

Notion.- Lors de la mise sous tension, un microprocesseur initialise ses registres puis entre dans une boucle infinie de chargement/exécution des instructions. En général la seule façon de sortir de cette boucle infinie est de couper l'alimentation électrique.

Cas du 8088.- Le démarrage correspond à une tension de +5 V à la broche 21 (*RESET*) du microprocesseur. Le microprocesseur 8088 initialise alors ses registres de la façon suivante :

```
CS = FFFFh
DS = 0000h
SS = 0000h
ES = 0000h
IP = 0000h
Flags Clear
Queue Empty
```

c'est-à-dire qu'il va chercher sa première instruction à l'adresse `FFFF0h`.

Remarque.- Ceci est valable aussi bien pour ce qu'on appelle le *démarrage à froid*, c'est-à-dire lorsqu'on allume l'ordinateur, que pour le *démarrage à chaud*, c'est-à-dire lorsqu'on lui demande de redémarrer (en général parce qu'on est bloqué) par un raccourci clavier ou grâce au bouton spécifique de l'unité centrale. C'est le BIOS qui fait la différence et non le microprocesseur. Nous verrons plus loin comment.

11.1.2 Nécessité de la ROM BIOS

Une question fondamentale se pose :

Où aller chercher cette instruction ?

À l'adresse `FFFF0h` bien sûr. Mais à quoi correspond cette adresse ? Il ne peut pas s'agir de mémoire vive, puisque celle-ci est mise à zéro lors du démarrage (plus exactement elle ne conserve pas les données). Il ne peut pas s'agir non plus de mémoire de masse (disquette, disque dur, CD-ROM...) puisque le microprocesseur ne sait pas accèder par lui-même à la mémoire de masse. La seule possibilité est donc de fournir à cette adresse (et autour de cette adresse) un premier programme conservé de façon permanente.

On peut évidemment concevoir un ensemble d'interrupteurs permettant de programmer ce tout premier logiciel, mais ceci n'est pas très convivial.

On se sert donc de mémoire permanente ROM, en général quelques circuits intégrés de **mémoire BIOS** au vu de ce qu'on va y placer. Cette ROM peut contenir le système d'exploitation en entier (cas des premiers micro-ordinateurs tel que Apple II [à vérifier]) mais alors celui-ci est figé pour une machine donnée. Cette façon de faire n'est pas très intéressante de nos jours, puisque le système d'exploitation évolue très rapidement.

Les concepteurs de l'IBM-PC ont décidé de placer sur cette ROM un système d'exploitation provisoire très rudimentaire, appelé le **BIOS**, dont le rôle est de charger le système d'exploitation proprement dit, placé lui en mémoire de masse (disquette puis disque dur puis CD-ROM). [Qu'en est-il des tous premiers PC sans lecteur de disquette ?]

11.1.3 Que placer dans la ROM BIOS ?

Puisque l'adresse `FFFF0h` est située presqu'à la fin de la mémoire accessible (il ne reste plus que 16 octets pour y arriver), le programme BIOS commence par un saut inconditionnel.

Exercice 1.- *Voyez-vous la raison du choix de cette adresse proche de la fin?*

[Cela laisse la possibilité du programme BIOS de croître en taille.]

Exercice 2.- *Déterminer l'instruction se trouvant à l'adresse* `FFFF0h` *sur votre ordinateur.*

Première méthode.- Vérifions le contenu en utilisant `debug` :

```
- D FFFF:0
EA 5B E0 00 F0 30 36 2F - 31 35 2F 39 00 FC B9 .[...06/15/98...
```

Les cinq premiers octets correspondent à :

```
jmp F000:E05B
```

Commentaire.- Les huit octets suivants donnent la date de la version du BIOS, ici le 15 juin 1998. Il faut évidemment connaître l'organisation de la mémoire vive décidée par IBM pour interpréter ces huit derniers octets. Nous allons y revenir ci-dessous.

Deuxième méthode.- Utilisons également `debug` mais en désassemblant, ce qui nous évite d'avoir à le faire à la main :

```
- U FFFF:0000
FFFF:0000 EA5BE000F0 JMP F000:E05B
FFFF:0005 30362F31   XOR [312F], DH
```

ce qui nous donne bien le saut inconditionnel.

Remarquons cependant, une fois de plus, le défaut (malheureusement irrémédiable) des désassembleurs : cet utilitaire ne désassemble que ce qui est écrit en langage machine (on ne voit pas ce qu'il pourrait faire d'autre, d'ailleurs) ; il essaie donc également de désassembler la date.

Exercice 3.- Vérifions que la procédure de démarrage commence bien à l'adresse indiquée en utilisant une fois de plus `debug` :

```
- R CS
CS 1652
:F000
- R IP
IP 0100
:E05B
- G
```

Si tout se passe bien, l'ordinateur redémarre.

11.1.4 Contenu de la ROM BIOS

La ROM BIOS est utilisée différemment suivant la version du BIOS. Voici, à titre d'exemple, son utilisation pour le BIOS d'IBM pour les PC/XT :

Emplacement mémoire	Description
`F000:E000 - F000:E6F1`	Power-On Start-Up Tests (POST)
`F000:E6F2 - F000:E728`	Boot strap loader (INT 19h)
`F000:E729 - F000:E82D`	RS-232 I/O (INT 14h)
`F000:E82E - F000:E881`	Keyboard I/O (INT 16h)
`F000:E882 - F000:E986`	Keyboard scan code tables
`F000:E987 - F000:EC58`	Keyboard (INT 9h)
`F000:EC59 - F000:EFD1`	Diskette I/O (INT 13h)
`F000:EFD2 - F000:F044`	Printer I/O (INT 17h)
`F000:F045 - F000:F840`	Video I/O (INT 10h)
`F000:F841 - F000:F84C`	Memory check (INT 12h)
`F000:F84D - F000:F85B`	Equipment check (INT 11h)
`F000:F85C - F000:FA6D`	Cassette I/O (INT 15h) [PC only - not used on XT]
`F000:FA6E - F000:FE6D`	Graphics character table
`F000:FE6E - F000:FEF2`	Time of day (INT 1Ah)
`F000:FEF3 - F000:FF52`	Interrupt vector table
`F000:FF53 - F000:FF53`	Dummy return point for unused interrupts
`F000:FF54 - F000:FFD9`	Print screen (INT 5h)
`F000:FFDA - F000:FFEF`	Not used
`F000:FFF0 - F000:FFF4`	First code executed after power-on
`F000:FFF5 - F000:FFFD`	BIOS release date
`F000:FFFE - F000:FFFF`	Not used

Nous reviendrons plus en détail sur la description.

11.1.5 Listing du BIOS

La fin de la ROM est définie également à la fin du listing, à partir de la ligne 5952 :

```
                          5952
                          5953   ;---------------------------------
                          5954   ;       POWER ON RESET VECTOR    :
                          5955   ;---------------------------------
----                      5956   VECTOR  SEGMENT AT OFFFFH
                          5957
                          5958   ;----- POWER ON RESET
                          5959
0000 EA5BE000F0           5960           JMP      RESET
                          5961
0005 31312F30382F38       5962           DB      '11/08/82'            ; RELEASE MARKER
     32
----                      5963   VECTOR  ENDS
                          5964           END
```

Commentaires.- 1^o) Le segment commençant à l'adresse FFFF0h est appelé VECTOR. La première instruction est celle qui renvoie à la routine RESET, qui est la même que pour notre BIOS de 1998 :

```
JMP F000:E05B
```

La routine RESET est définie à partir de la ligne 301.

- 2^o) L'un des premiers BIOS nous donne la signification des 8 derniers octets de la ROM, à savoir a date de révision de celle-ci.

11.1.6 Version du BIOS

Comme nous l'avons déjà vu, la date de la version du BIOS est située sur les neuf octets `F000:FFF5 - F000:FFFD`. Nous avons également vu comment récupérer celle-ci à l'aide de `debug`.

11.1.6.1 Date du BIOS

Comme nous l'avons vu, les huit premiers octets donnent la date, par exemple dans l'exemple vu ci-dessus « 06/15/98 ».

11.1.6.2 Modèle

Le dernier octet, « FC » dans notre exemple, est le **numéro d'identification** du modèle (uniquement valable pour les produits IBM).

Ce numéro donne le modèle conformément au tableau suivant :

Produit	Date BIOS	Modèle	Sous-modèle	Version
PC	04/24/81	FF	--	--
PC	10/19/81	FF	--	--
PC	10/27/82	FF	--	--
PC XT	11/08/82	FE	--	--
Portable	11/08/82	FE	--	--
PC XT	01/10/86	FB	00	01
PC XT	05/09/86	FB	00	02
PC jr	06/01/83	FD	--	--
AT	01/10/84	FC	--	--
AT	06/10/85	FC	00	01
AT	11/15/85	FC	01	00

PC XT Model 286	04/21/86	FC	02	00
PC Convertible	09/13/85	F9	00	00
Personal System 2 Model 30	09/02/86	FA	00	00
Personal System 2 Model 50	*	FC	04	00
Personal System 2 Model 60	*	FC	05	00
Personal System 2 Model 80	*	F8	00	00
Personal System 2 Model 80	*	F8	01	00

11.1.6.3 Première instruction du BIOS

Nous avons également vu que les cinq octets `F000:FFF0 - F000:FFF4` constituent la première instruction du BIOS, qui est un saut inconditionnel `far`, d'ailleurs toujours le même :

```
JMP F000:E05B
```

11.1.7 Numéro de série et copyright

Introduction.- Nous avons vu que le BIOS commence à l'adresse `FFFF0h`, par un saut inconditionnel. Ce saut se faisait à l'adresse `FE05Bh` pour les tous premiers PC. Nous avons vu aussi que la ROM BIOS commence à l'adresse `FE000h`.

Qu'y a-t-il dans les 90 premiers octets de la ROM BIOS ?

Cet emplacement est réservé pour donner un *numéro de série* de 7 chiffres, suivi d'une notice sur le copyright. Le numéro de série est en hexadécimal et la notice évidemment en ASCII.

Exemple.- Voyons ce qu'il en est pour notre BIOS, en utilisant `debug` pour cela :

```
C:\WINDOWS>debug
-d FE00:0
FE00:0000  41 77 61 72 64 20 53 6F-66 74 77 61 72 65 49 42   Award SoftwareIB
FE00:0010  4D 20 43 4F 4D 50 41 54-49 42 4C 45 20 34 38 36   M COMPATIBLE 486
FE00:0020  20 42 49 4F 53 20 43 4F-50 59 52 49 47 48 54 20    BIOS COPYRIGHT
FE00:0030  41 77 61 72 64 20 53 6F-66 74 77 61 72 65 20 49   Award Software I
FE00:0040  6E 63 2E 6F 66 74 77 61-72 65 20 49 6E 63 2E 20   nc.oftware Inc.
FE00:0050  41 77 61 72 64 20 53 6F-66 74 15 E9 12 14 20 43   Award Soft.... C
FE00:0060  1B 41 77 61 72 64 20 4D-6F 64 75 6C 61 72 20 42   .Award Modular B
FE00:0070  49 4F 53 20 76 34 2E 35-31 50 47 00 DB 32 EC 33   IOS v4.51PG..2.3
-d
FE00:0080  EC 35 EC 38 EC 3C EC 41-EC 47 EC 4E EC 77 61 72   .5.8.<.A.G.N.war
FE00:0090  2C 43 6F 70 79 72 69 67-68 74 20 28 43 29 20 31   ,Copyright (C) 1
FE00:00A0  39 38 34 2D 39 38 2C 20-41 77 61 72 64 20 53 6F   984-98, Award So
FE00:00B0  66 74 77 61 72 65 2C 20-49 6E 63 2E 00 74 77 61   ftware, Inc..twa
FE00:00C0  4E 54 42 43 30 36 33 30-42 00 29 20 45 56 41 4C   NTBC0630B.) EVAL
FE00:00D0  55 41 54 49 4F 4E 20 52-4F 4D 20 2D 20 4E 4F 54   UATION ROM - NOT
FE00:00E0  20 46 4F 52 20 53 41 4C-45 00 00 00 00 00 00 00    FOR SALE.......
FE00:00F0  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
-d
FE00:0100  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 23   ...............#
FE00:0110  20 49 42 4D 20 49 53 20-41 20 54 52 41 44 45 4D    IBM IS A TRADEM
FE00:0120  41 52 4B 20 4F 46 20 49-4E 54 45 52 4E 41 54 49   ARK OF INTERNATI
FE00:0130  4F 4E 41 4C 20 42 55 53-49 4E 45 53 53 20 4D 41   ONAL BUSINESS MA
FE00:0140  43 48 49 4E 45 53 20 43-4F 52 50 2E 55 8B EC 56   CHINES CORP.U..V
FE00:0150  1E C5 76 02 81 3C 0F 05-74 3E 81 3C 0F 04 74 17   ..v..<..t>.<..t.
FE00:0160  80 3C F0 74 0A 80 3C F2-74 05 80 3C F3 75 04 46   .<.t..<.t..<.u.F
FE00:0170  89 76 02 1F 5E 5D CF E9-C2 61 20 00 00 08 00 00   .v..^]...a .....
-
```

Dans notre cas il n'y a pas de numéro de série, ce qu'explique peut-être le texte quelques lignes plus bas : "EVALUATION ROM - NOT FOR SALE", ainsi que les répétitions un peu décousues.

11.2 Les tests au démarrage (POST)

L'instruction se trouvant à l'adresse FFFF0h renvoie à ce qui va être lu lors du démarrage de l'ordinateur, en ce qui commençant par des tests (POST). Commençons par voir les grandes étapes de la procédure de démarrage, puis étudierons plus en détail chacune d'elles ensuite.

11.2.1 Grandes étapes

Le contenu du BIOS dépend évidemment du langage machine (du microprocesseur utilisé) et de l'architecture du micro-ordinateur. Cependant, pour un compatible PC, la **procédure de démarrage** (en anglais *System Startup Procedure*) est la suivante :

— le microprocesseur s'initialise, donc va à l'adresse du programme d'initialisation ; la ROM BIOS a été interfacée de telle façon que cette adresse s'y trouve.

— le BIOS commence par tester le microprocesseur ; il teste ensuite la ROM, à la fois la ROM BIOS et la ROM de l'interpréteur BASIC, par somme de contrôle, pour être (presque) sûr que ces deux composants ne sont pas corrompus ; il teste esuite la RAM de la même façon ; les composants connectés au microprocesseur, en particulier le contrôleur graphique, sont alors initialisés.

— le BIOS initialise ensuite les périphériques.

— un programme, appelé **chargeur de démarrage** (en anglais *bootstrap loader*), charge l'**enregistrement de démarrage** (en anglais *boot record*) du disque qui a été désigné comme **disque de démarrage** (en anglais *startup drive*). Puios on fait exécuter le programme qui vient d'être chargé. Ce programme a pour but, dans le cas de MS-DOS, de charger les fichiers (qui seront cachés) `io.sys` et `msdos.sys` ainsi que le **processeur de commandes** de MS-DOS,appelé `command.com`.

Le code des tests au démarrage (en anglais POST pour *Power-On Start-up Tests*) est situé aux adresses `F000:E000 - F000:E6F1`. Nous venons de voir que le point d'entrée est `F000:E05B`.

11.2.2 Test du microprocesseur

Le début de RESET, correspondant au test du microprocesseur (lignes 301–361, c'est-à-dire des instructions se trouvant aux emplacements mémoire E05B à E0AD), a été étudié au chapitre 2.

11.2.3 Initialisations temporaires

On continue de la façon suivante, à la fois à partir de la ligne E0AE du listing et de l'instruction se trouvant à l'emplacement mémoire E0AE) :

```
                       362  ;---------------------------------------
                       363  ;       ROS CHECKSUM TEST I            :
                       344  ; DESCRIPTION                          :
                       365  ;       A CHECKSUM IS DONE FOR THE 8K  :
                       366  ;       ROS MODULE CONTAINNING POO AND :
                       367  ;       BIOS.                          :
                       368  ;---------------------------------------
E0AE                   369  C10:
                       370                                         ; ZERO IN AL ALREADY
E0AE EAA0              371          OUT      0A0H,AL               ; DISABLE NMI INTERRUPTS
E0B0 E683              372          OUT      83H,AL                ; INITIALIZE DMA PAGE REG
E0B2 BAD803            373          MOV      DX,3D8H
E0B5 EE                374          OUT      DX,AL                 ; DISABLE COLOR VIDEO
E0B6 FEC0              375          INC      AL
E0B8 B089              376          MOV      DL,0B8H
E0BA EE                377          OUT      DX,AL                 ; DISABLE B/W VIDEO,EN HIGH RES
```

Commentaires.- 1^o) On inhibe les interruptions non masquables (ligne 371) en envoyant 0 au port A0h, c'est-à-dire le registre de masque NMI. Le registre AL contient bien 0 puisqu'on commence.

- 2^o) On inititialise le registre de page DMA (ligne 372) en envoyant 0 au port 83h, c'est-à-dire pour le canal 1 de la DMA.

- 3^o) On désactive la carte CGA (lignes 373 et 374), en envoyant 0 au port 3D8h, c'est-à-dire le registre de sélection de mode de la carte CGA, en particulier grâce au bit 3 à zéro pour ne pas avoir d'affichage.

- 4^o) On désactive la carte MDA (lignes 375 à 377), en envoyant 1 au port 3B8h, c'est-à-dire le registre de contrôle de cette carte : le bit 0 doit être à 1 ; le bit 3 à 0 spécifie qu'il n'y a pas d'affichage.

11.2.4 Initialisation du PPI

L'initialisation du PPI, objet des lignes 378 à 386 a été étudiée au chapitre 4.

11.2.5 Vérification de la ROM

On continue par la vérification de la ROM, lignes 387 à 396, déjà étudiée au chapitre 3.

11.2.6 Initialisation du DMA

L'initialisation du DMAC (lignes 397 à 502) a été étudiée au chapitre 9.

11.2.7 Vérification de la RAM

On continue par la vérification de la mémoire vive, commençant à la ligne 502. Le code utilise une sous-routine, que nous allons commencer par étudier.

11.2.7.1 Sous-routine de vérification de la RAM

La sous-routine STGTST_CNT (pour *SToraGe TeST CouNTer*) teste un bloc de mémoire et revient avec l'indicateur ZF à 0 s'il y a une erreur :

```
                1308
                1309  ;-----------------------------------------------------------------
                1310  ; THIS SUBROUTINE PERFORMS A READ/WRITE STORAGE TEST ON A BLOCK :
                1311  ;       OF STORAGE.                                             :
                1312  ; ENTRY REQUIREMENTS:                                           :
                1313  ;       ES = ADDRESS OF STORAGE SEGMENT BEING TESTED            :
                1314  ;       DS = ADDRESS OF STORAGE SEGMENT BEING TESTED            :
                1315  ;       CX = WORD COUNT OF STORAGE BLOCK TO BE TESTED           :
                1316  ; EXIT PARAMETERS:                                              :
                1317  ;       ZERO FLAG = 0 IF STORAGE ERROR (DATA COMPARE OR PARITY  :
                1318  ;       CHECK. AL=0 DENOTES A PARITY CHECK. ELSE AL=XOR'ED      :
                1319  ;       BIT PATTERN OF THE EXPECTED DATA PATTERN VS THE ACTUAL  :
                1320  ;       DATA READ.
                1321  ; AX,BX,CX,DX,DI, AND SI ARE ALL DESTROYED.                     :
                1322  ;-----------------------------------------------------------------
                1323
E66F            1324  STGTST_CNT      PROC   NEAR
E66F FC         1325          CLD                             ; SET DIR FLAG TO INCREMENT
E670 2BFF       1326          SUB     DI,DI                   ; SET DI=OFFSET 0 REL TO ES REG
E672 2BC0       1327          SUB     AX,AX                   ; SETUP FOR 0->FF PATTERN TEST
E674            1328  C2_1:
E674 8805       1329          MOV     [DI],AL                 ; ON FIRST BYTE
E676 8A05       1330          MOV     AL,[DI]
E678 32C4       1331          XOR     AL,AH                   ; O.K.?
E67A 754D       1332          JNZ     C7                      ; GO ERROR IF NOT
E67C FEC4       1333          INC     AH
E67E 8AC4       1334          MOV     AL,AH
E680 75F2       1335          JNZ     C2_1                    ; LOOP TILL WRAP THROUGH FF
E682 8BD9       1336          MOV     BX,CX                   ; SAVE WORD COUNT OF BLOCK TO TEST
E684 D1E3       1337          SHL     BX,1                    ; CONVERT TO A BYTE COUNT
E686 B8AAAA     1338          MOV     AX,0AAAAH               ; GET INITIAL DATA PATTERN TO WRITE
E689 BA55FF     1339          MOV     DX,0FF55H               ; SETUP OTHER DATA PATTERNS TO USE
E68C F3         1340          REP     STOSW                   ; FILL STORAGE LOCATIONS IN BLOCK
E68D AB
E68E E461       1341          IN      AL,PORT_B
E690 0C30       1342          OR      AL,00110000B            ; TOGGLE PARITY CHECK LATCHES
E692 E661       1343          OUT     PORT_B,AL
E694 90         1344          NOP
E695 24DF       1345          AND     AL,1100111B
E697 E661       1346          OUT     PORT_B,AL
E699            1347  C3:
E699 4F         1348          DEC     DI                      ; POINT TO LAST BYTE JUST WRITTEN
E69A FD         1349          STD                             ; SET DIR FLAG TO GO BACKWARDS
E69B            1350  C4:
E69B 8BF7       1351          MOV     SI,DI                   ; INITIALIZE DESTINATION POINTER
E69D 8BCB       1352          MOV     CX,BX                   ; SETUP BYTE COUNT FOR LOOP
E69F            1353  C5:                                     ;       INNER TEST LOOP
E69F AC         1354          LODSB                           ; READ OLD TEST BYTE FROM STORAGE [SI]
E6A0 32C4       1355          XOR     AL,AH                   ; DATA READ AS EXPECTED?
E6A2 7525       1356          JNE     C7                      ; NO - GO TO ERROR ROUTINE
E6A4 8AC2       1357          MOV     AL,DL                   ; GET NEXT DATA PATTERN TO WRITE
E6A6 AA         1358          STOSB                           ; WRITE INTO LOC JUST READ [DI]+
E6A7 E2F6       1359          LOOP    C5                      ; DECREMENT BYTE COUNT AND LOOP  CX
                1360
E6A9 22E4       1361          AND     AH,AH                   ; ENDING ZERO PATTERN WRITTEN TO STG ?
E6AB 7416       1362          JZ      C6X                     ; YES - RETURN TO CALLER WITH AL=0
E6AD 8AE0       1363          MOV     AH,AL                   ; SETUP NEW VALUE FOR COMPARE
E6AF 86F2       1364          XCHG    DH,DL                   ; MOVE NEXT DATA PATTERN TO DL
E6B1 22E4       1365          AND     AH,AH                   ; READING ZERO PATTERN TO DL
E6B3 7504       1366          JNZ     C6                      ; CONTINUE TEST SEQUENCE TILL ZERO DATA
E6B5 8AD4       1367          MOV     DL,AH                   ; ELSE SET ZERO FOR END READ PATTERN
```

```
E6B7 EBE0          1368          JMP     C3                 ; AND MAKE FINAL BACKWARDS PASS
E6B9               1369   C6:
E6B9 FC            1370          CLD                        ; SET DIR FLAG TO GO FORWARD
E6BA 47            1371          INC     DI                 ; SET POINTER TO BEG LOCATION
E6BB 74DE          1372          JZ      C4                 ; READ/WRITE FORWARD IN STG
E6BD 4F            1373          DEC     DI                 ; ADJUST POINTER
E6BE BA0100        1374          MOV     DX,0001H           ; SETUP 01 FOR PARITY BIT AND 00 FOR END
E6C1 EBD6          1375          JMP     C3                 ; READ/WRITE BACKWARD IN STG
E6C3               1376   C6X
E6C3 E462          1377          IN      AL,PORT_C          ; DID A PARITY ERROR OCCUR ?
E6C5 24C0          1378          AND     AL,0C0H            ; ZERO FLAG WILL BE OFF PARITY ERROR
E6C7 B000          1379          MOV     AL,000H            ; AL=0 DATA COMPARE OK
E6C9               1380   C7:
E6C9 FC            1381          CLD                        ; SET DIRECTION FLAG TO INC
E6CA C3            1382          RET
                   1383   STGTST_CNT     ENDP
```

Commentaires.- 1^o) On commence par tester toutes les valeurs possibles sur le premier octet du bloc à vérifier.

Pour cela, on décrit la mémoire dans l'ordre croissant (ligne 1325), le décalage de la destination est zéro (ligne 1326), on place le motif (donc 0 au début, ligne 1327) dans le registre AL, on l'écrit comme premier octet du bloc à vérifier (ligne 1329), on le récupère (ligne 1330) et on le compare au motif envoyé (ligne 1331).

Si les deux octets ne sont pas identiques (ligne 1332), on change l'indicateur de direction (ligne 1381) et on termine la sous-routine (ligne 1333). L'indicateur ZF vaut zéro comme voulu.

Sinon on passe au motif suivant (lignes 1333 et 1334) et on recommence le test tant qu'on n'est pas revenu à zéro pour le motif (ligne 1335 ; rappelons que le successeur de FFh est 0h).

- 2^o) Une fois terminée la vérification du premier octet du bloc de mémoire à vérifier, on place dans le registre BX le nombre d'octets à vérifier (lignes 1336 et 1337), on place dans AX un premier motif (ligne 1338), dans DX un second motif (ligne 1339) et on remplit le bloc avec le premier motif (ligne 1340). On active la vérification de la parité de la RAM (lignes 1341 à 1343) puis on la désactive (lignes 1344 à 1346). On se place au dernier octet que l'on vient d'écrire (ligne 1348), on décrira la mémoire dans l'ordre décroissant (ligne 1349), on initialise le pointeur source avec celui de destination (ligne 1351), on prend le compteur en octets (ligne 1353) et on copie octet par octet au même endroit (ligne 1354).

Si on n'écrit pas ce qui a été lu (ligne 1355), on termine sur une erreur (ligne 1356).

On y place ensuit le deuxième motif (lignes 1357 et 1358) et on boucle pour décrire tout le bloc en sens décroissant (ligne 1359).

- 3^o) Si le dernier motif écrit est zéro (lignes 1361 et 1362), on a un problème. Si une erreur de parité est survenue (lignes 1377 et 1378), on met l'indicateur ZF à zéro (ligne 1379) et on termine la sous-routine (en spécifiant une erreur).

- 4^o) Sinon on passe à un autre motif (lignes 1363 et 1364). Tant qu'il ne s'agit pas de zéro (lignes 1365 et 1366), on recommence (lignes 1367 et 1368).

- 5^o) Une fois terminé ce test, on décrira la mémoire dans l'ordre croissant (ligne 1370), on revient au début de la mémoire (ligne 1371),

11.2.7.2 La vérification

Le code du BIOS d'IBM consacré à la vérification de la RAM est le suivant :

```
                     503  ;----------------------------------------------
                     504  ;       BASE 16K READ/WRITE STORAGE TEST       :
                     505  ; DESCRIPTION                                  :
                     506  ;       WRITE/READ/VERIFY DATA PATTERNS        :
                     507  ;       AA,55,FF,01, AND 00 TO 1ST 32K OF      :
                     508  ;       STORAGE. VERIFY STORAGE ADDRESSABILITY. :
                     509  ;----------------------------------------------
                     510
                     511  ;----- DETERMINE MEMORY SIZE AND FILL MEMORY WITH DATA
                     512
E165 BA1302          513          MOV     DX,0213H                          ; ENABLE I/O EXPANSION BOX
E168 B001            514          MOV     AL,01H
E16A EE              515          OUT     DX,AL
                     516
E16B 8B1E7204        517          MOV     BX,DATA_WORD[OFFSET RESET_FLAG]  ; SAVE 'RESET_FLAG' IN BX
E16F8 B90020         518          MOV     CX,2000H                          ; SET FOR 16K WORDS
E172 81FB3412        519          CMP     BX,1234H                          ; WARM START?
E176 7416            520          JE      CLR_STG
E178 BC18E0          521          MOV     SP,OFFSET CL
E17B E9F104          522          JMP     STGTST_CNT
E17E 7412            523  C24:    JE      HOW_BIG                           ; STORAGE OK, DETERMINE SIZE
E180 8AD8            524          MOV     BL,AL                             ; SAVE FAILING BIT PATTERN
E182 8004            525          MOV     AL,04H                            ; <><><><><><><><><><><>
E184 E660            526  C24A:   OUT     PORT_A,AL                         ; <><><>CHECKPOINT 4<><><>
E186 2BC9            527          SUB     CX,CX                             ; BASE RAM FAILURE - HANG
E188 E2F2            528  C24B:   LOOP    C24B                              ; FLIPPING BETWEEN 04 AND
E18A 8608            529          XCHG    BL,AL                             ; FAILING BIT PATTERN
E18C EBF6            530          JMP     C24A
E18E                 531  CLR_STG:
E18E 2BC0            532          SUB     AX,AX                             ; MAKE AX=0000
E190 F3              533          REP     STOSW                             ; STORE 8K WORDS OF 0000
E191 AB
E192                 534  HOW_BIG:
E192 891E7204        535          MOV     DATA_WORD[OFFSET RESET_FLAG],BX  ; RESTORE RESET FLAG
E196 BA0004          536          MOV     DX,0400H                          ; SET POINTER TO JUST>16KB
E199 881000          537          MOV     BX,16                             ; BASIC COUNT OF 16 K
E19C                 538  FILL_LOOP:
E19C 8EC2            539          MOV     ES,DX                             ; SET SEG. REG.
E19E 2BFF            540          SUB     DI,DI
E1A0 B855AA          541          MOV     AX,0AA55H                         ; TEST PATTERN
E1A3 8BC8            542          MOV     CX,AX                             ; SAVE PATTERN
E1A5 268905          543          MOV     ES:[DI],AX                        ; SEND PATTERN TO MEM.
E1A8 B00F            544          MOV     AL,0FH                            ; PUT SOMETHING IN AL
E1AA 268B05          545          MOV     AX,ES:[DI]                        ; GET PATTERN
E1AD 33C1            546          XOR     AX,CX                             ; COMPARE PATTERNS
E1AF 7511            547          JNZ     HOW_BIG_END                       ; GO END IF NO COMPARE
E1B1 B90020          548          MOV     CX,2000H                          ; SET COUNT FOR 8K WORDS
E1B4 F3              549          REP     STOSW                             ; FILL 8K WORDS
E1B5 AB
E1B6 81C20004        550          ADD     DX,400H                           ; POINT TO NEXT 16KB BLOCK
E1BA 83C310          551          ADD     BX,16                             ; BUMP COUNT BY 16KB
E1BD 80FEA0          552          CMP     DH,0A0H                           ; TOP OF RAM AREA YET? (A0000)
E1C0 75DA            553          JNZ     FILL_LOOP
E1C2                 554  HOW_BIG_END:
E1C2 891E1304        555          MOV     DATA_WORD[OFFSET MEMORY_SIZE],BX ; SAVE MEMORY SIZE
                     556
```

Commentaires.- 1^o) La mémoire qui se trouverait sur une carte d'extension est activée (lignes 513 à 515).

- 2^o) L'indicateur RESET_FLAG (qui se trouve dans la zone de communication du BIOS) est sauvegardé dans le registre BX (ligne 517).

Rappelons que DATA_WORD a été déclaré ligne 60 dans le segment de code ABSO, comme début de la zone de communication du BIOS.

Rappelons également que RESET_FLAG a été déclaré à la ligne 199 et qu'il vaut 1234h pour un démarrage « à chaud ». Dans ce dernier cas, on n'a pas besoin de tester la RAM.

- 3^o) Le registre de compteur est initialisé à 2000h, puisqu'un circuit intégré de mémoire RAM a un capacité de 16 KiO (ligne 518).

- 4^o) Dans le cas d'un démarrage à chaud, on saute une partie des tests (lignes 519 et 520).

- 5^o) Le pointeur de pile SP pointe sur le registre de compteur CX (ligne 521).

- 6^o) On fait appel à la sous-routine STGTST_CNT (ligne 522).

Si le test de la mémoire échoue, on sauvegarde le motif en cause (ligne 524), on envoie 4h au port A du PPI (lignes 525 et 526) pour spécifier une capacité de 48 KiO, on remet le compteur à zéro (ligne 527) et on place le motif défectueux sur le port A (lignes 529 et 530).

- 7^o) Dans le cas d'un démarrage à chaud, on efface la mémoire en plaçant 0 à chaque mot du bloc (lignes 531 à 533).

- 8^o) On réinitialise l'indicateur de démarrage à chaud (ligne 535), on déplace le pointeur au-delà du premier bloc de 16 KiO (ligne 536), on initialise BX pour 16 nouveaux KiO (ligne 537), on initialise le registre ES (ligne 538), on initialise DI à 0 (ligne 539), on initialise AX avec le motif (ligne 540), que l'on sauvegarde (ligne 541), on dépose le motif dans la mémoire (ligne 542), on met autre chose dans le registre AL (ligne 543), on essaie de récupérer le motif depuis la mémoire (ligne 544) et on compare (ligne 545).

Si on ne retrouve pas la même chose, on va à la fin (ligne 547), c'est-à-dire qu'on ne pourra pas utiliser ce bloc mémoire, ni les suivants.

S'il n'y a pas de problème, on remplit ce boc de 16 KiO avec le motif (lignes 548 et 549) et on regarde le bloc de 16 KiO suivant (lignes 550 à 553).

- 9^o) À la fin on a donc obtenu la quantité de mémoire réellement disponible, que l'on place dans la variable adéquate de la zone de communication du BIOS (ligne 555).

11.2.8 Initialisation de la pile temporaire

On poursuit en initialisant une pile temporaire pour le démarrage :

```
                        557   ;----- SETUP STACK SEG AND SP
                        558
E1C6 B83000             559           MOV      AX,STACK             ; GET STACK VALUE
E1C9 8ED0               560           MOV      SS,AX                ; SET THE STACK UP
E1CB BC0001             561           MOV      SP,OFFSET TOS        ; STACK IS READY TO GO
```

Commentaires.- 1^o) Le registre de pile est initialisé (lignes 559 et 560) avec l'adresse du segment défini à la ligne 71 :

```
                        67    ;---------------------------------------------
                        68    ; STACK -- USED DURING INITIALIZATION ONLY     :
                        69    ;---------------------------------------------
                        70
----                    71    STACK   SEGMENT  AT 30H
0000 (128               72            DW       128 DUP(?)
     ????
     )
0100                    73    TOS     LABEL    WORD
----                    74    STACK   ENDS
                        75
```

- 2^o) Le sommet de la pile est initialisé avec l'emplacement de fin du segment (ligne 73 avec TOS pour *Top Of Stack*).

11.2.9 Initialisation du contrôleur des interruptions matérielles

L'initialisation du contrôleur d'interruption 8259 (lignes 562 à 573) a été étudiée au chapitre 8.

11.2.10 Affectation temporaire des vecteurs d'interruption

On continue, à partir de la ligne 574, par l'affectation temporaire des vecteurs d'interruption. On y place une routine générique pour certains d'eux et on se sert d'une table des vecteurs d'interruption.

11.2.10.1 Routine de service générique

Il faut affecter une routine de service à chaque interruption. Une routine de service générique est définie : il s'agit d'une routine de service temporaire en attendant que la routine adéquate soit définie ultérieurement ; il s'agit également de la routine qui restera en place pour celles qui ne seronts pas définies :

```
                   5774
                   5775  ;------------------------------------------------------
                   5776  ; TEMPORARY INTERRUPT ROUTINE                          :
                   5777  ;       1. THIS ROUTINE IS ALSO LEFT IN PLACE AFTER THE :
                   5778  ;       POWER ON DIAGNOSTICS TO SERVICE UNUSED         :
                   5779  ;       INTERRUPT VECTORS. LOCATION 'INTR_FLAG' WILL   :
                   5780  ;       CONTAIN EITHER: 1. LEVEL OF HARDWARE INT. THAT :
                   5781  ;       CAUSED CODE TO BE EXEC.                        :
                   5782  ;       2. 'FF' FOR NON-HARDWARE INTERRUPTS THAT WAS   :
                   5783  ;       EXECUTED ACCIDENTLY.                           :
                   5784  ;------------------------------------------------------
FF23               5785  D11     PROC     NEAR
                   5786          ASSUME   DS:DATA
FF23 1E            5787          PUSH     DS
FF24 52            5788          PUSH     DX
FF25 50            5789          PUSH     AX                ; SAVE REG AX CONTENTS
FF26 E830FB        5790          CALL     DDS
FF29 B00B          5791          MOV      AL,0BH            ; READ IN-SERVICE REG
FF2B E620          5792          OUT      INTA00,AL         ; (FIND OUT WHAT LEVEL BEING
FF2D 90            5793          NOP                        ; SERVICED)
FF2E E420          5794          IN       AL,INTA00         ; GET LEVEL
FF30 8AE0          5795          MOV      AH,AL             ; SAVE IT
FF32 0AC4          5796          OR       AL,AH             ; 00? (NO HARDWARE ISR ACTIVE)
FF34 7504          5797          JNZ      HW_INT
FF36 B4FF          5798          MOV      AH,0FFH
FF38 EB04          5799          JMP      SHORT SET_INTR_FLAG  ; SET FLAG TO FF IF NON-HDWARE
FF3A               5800  HW_INT:
FF3A E421          5801          IN       AL,INTA01         ; GET MASK VALUE
FF3C 0AC4          5802          OR       AL,AH             ; MASK OFF LVL BEING SERVICED
FF3E E621          5803          OUT      INTA01,AL
FF40 B020          5804          MOV      AL,EOI
FF42 E620          5805          OUT      INTA00,AL
FF44               5806  SET_INTR_FLAG:
FF44 88266B00      5807          MOV      INTR_FLAG,AH      ; SET FLAG
FF48 58            5808          POP      AX                ; RESTORE REG AX CONTENTS
FF49 5A            5809          POP      DX
FF4A 1F            5810          POP      DS
FF4B               5811  DUMMY_RETURN:                      ; NEED IRET FOR VECTOR TABLE
FF4B CF            5812          IRET
                   5813  D11     ENDP
                   5814
                   5815  ;----------------------------------------------
                   5816  ; DUMMY RETURN FOR ADDRESS COMPATIBILITY       :
                   5817  ;----------------------------------------------
FF53               5818          ORG      0FF53H
FF53 CF            5819          IRET
                   5820
```

<u>Commentaires</u>.- 1^{o}) Les contenus des registres qui vont être utilisés (DS, DX et AX) sont sauvegardés sur la pile (lignes 5789).

- 2^{o}) Le segment des données utilisé est celui de la zone de communication du BIOS (ligne 5790).

- 3^o) On envoie 8h au premier port du PIC 8259 (lignes 5791 et 5792), c'est-à-dire l'octet de contrôle 0CW3 8h = 0000 1000b, afin d'obtenir le contenu du registre IR lors de la prochaine lecture.

- 4^o) On attend un certain temps (ligne 5793) puis on lit le contenu de ce registre IR (ligne 5794), octet que l'on sauvagarde dans le registre AH (ligne 5795).

- 5^o) Si cette interruption est active (lignes 5796 et 5797), on récupère le masque des interruptions matérielles (ligne 5801), on change le masque (lignes 5802 et 5803) et on permet aux autres interruptions d'être actives (lignes 5804 et 5805).

- 6^o) Si l'interruption n'est pas active, on place FF dans le registre AH (lignes 5798 et 5799).

- 7^o) On termine ces deux cas en plaçant le contenu de AH dans INTR_FLAG, indicateur de la zone de communication du BIOS indiquant qu'une interruption est intervenue (défini, rappelons-le, à la ligne 182).

- 8^o) Comme pour toute routine de service d'une interruption, on restaure les contenus des registres utilisés (lignes 5808 à 5810) et on retourne à la ligne appelante (ligne 5811).

11.2.10.2 Table des vecteurs d'interruption

La table des vecteurs d'interruption est définie à partir de la ligne 5737 :

```
                        5737  ;-----------------------------------------------
                        5738  ; THESE ARE THE VECTORS WHICH ARE MOVED INTO    :
                        5739  ; THE 8086 INTERRUPT AREA DURING POWER ON.      :
                        5740  ; ONLY THE OFFSETS ARE DISPLAYED HERE, CODE     :
                        5741  ; SEGMENT WILL BE ADDED FOR ALL OF THEM, EXCEPT :
                        5742  ; WHERE NOTED.                                  :
                        5743  ;-----------------------------------------------
                        5744          ASSUME  CS:CODE
FEF3                    5745          ORG     0FEF3H
FEF3                    5746  VECTOR_TABLE    LABEL  WORD           ; VECTOR TABLE FOR MOVE TO INTERRUPTS
FEF3 A5FE               5747          DW      OFFSET TIMER_INT      ; INTERRUPT 8
FEF5 87E9               5748          DW      OFFSET KB_INT         ; INTERRUPT 9
FEF7 23FF               5749          DW      OFFSET D11            ; INTERRUPT A
FEF9 23FF               5750          DW      OFFSET D11            ; INTERRUPT B
FEFB 23FF               5751          DW      OFFSET D11            ; INTERRUPT C
FEFD 23FF               5752          DW      OFFSET D11            ; INTERRUPT D
FEFF 57EF               5753          DW      OFFSET DISK_INT       ; INTERRUPT E
FF01 23FF               5754          DW      OFFSET D11            ; INTERRUPT F
FF03 65F0               5755          DW      OFFSET VIDEO_IO       ; INTERRUPT 10H
FF05 4DF8               5756          DW      OFFSET EQUIPMENT      ; INTERRUPT 11H
FF07 41F8               5757          DW      OFFSET MEMORY_SIZE_DET ; INTERRUPT 12H
FF09 59EC               5758          DW      OFFSET DISKETTE_IO    ; INTERRUPT 13H
FF0B 39E7               5759          DW      OFFSET RS232_IO       ; INTERRUPT 13H
FF0D 59F8               5760          DW      CASSETTE_IO           ; INTERRUPT 15H(FORMER CASSETTE IO)
FF0F 2EE8               5761          DW      OFFSET KEYBOARD_IO    ; INTERRUPT 16H
FF11 D2EF               5762          DW      OFFSET PRINTER_IO     ; INTERRUPT 17H
                        5763
FF13 0000               5764          DW      00000H                ; INTERRUPT 18H
                        5765          DW      0F600H                ;  MUST BE INSERTED INTO TABLE LATER
                        5766
FF15 F2E6               5767          DW      OFFSET BOOT_STRAP     ; INTERRUPT 19H
FF17 6EFE               5768          DW      TIME_OF_DAY           ; INTERRUPT 1AH -- TIME OF DAY
FF19 4BFF               5769          DW      DUMMY_RETURN          ; INTERRUPT 1BH -- KEYBOARD BREAK ADDR
FF1B 4BFF               5770          DW      DUMMY_RETURN          ; INTERRUPT 1C -- TIMER BREAK ADDR
FF1D A4F0               5771          DW      VIDEO_TERMS           ; INTERRUPT 1D -- VIDEO PARAMETERS
FF1F C7EF               5772          DW      OFFSET DISK_BASE      ; INTERRUPT 1E -- DISK PARMS
FF21 0000               5773          DW      0                     ; INTERRUPT 1F -- POINTER TO VIDEO EXT
                        5774
```

Les routines TIMER_INT sont définies à partir de la ligne 5696, KB_INT de la ligne 1849 comme nous l'avons déjà vu, DISK_INT de la ligne 2948, VIDEO_IO de la ligne 3353 comme nous l'avons déjà vu, EQUIPMENT de la ligne 5119, MEMORY_SIZE_DET de la ligne 5074, DISKETTE_IO de la ligne 2359, RS232_IO de la ligne 1538, CASSETTE_IO de la ligne 5133, KEYBOARD_IO de la ligne 1727, PRINTER_IO de la ligne 3112, BOOT_STRAP de la ligne 1420, TIME_OF_DAY de la ligne 5650, DUMMY_RETURN de la ligne 5811 comme nous venons de le voir, VIDEO_PARMS de la ligne 3399 et DISK_BASE de la ligne 3065.

11.2.10.3 Affectation des vecteurs d'interruption

L'affectation des vecteurs d'interruption, c'est-à-dire des adresses des routines de service, commence ligne 574 :

```
                        574   ;----- SET UP THE INTERRUPT VECTORS TO TEMP INTERRUPT
                        575
E1DE 1E                 576           PUSH    DS
E1DF B92000             577           MOV     CX,32                    ; FILL ALL 32 INTERRUPTS
E1E2 2BFF               578           SUB     DI,DI                    ; FIRST INTERRUPT LOCATION
E1E4 8EC7               579           MOV     ES,DI                    ; SET ES=0000 ALSO
E1E6 B823FF             580   D3:     MOV     AX,OFFSET D11            ; MOVE ADDR OF INTR PROC TO TBL
E1E9 AB                 581           STOSW
E1EA 8CC8               582           MOV     AX,CS                    ; GET ADDR OF INTR PROC SEG
E1EC AB                 583           STOSW
E1ED E2F7               584           LOOP    D3                       ; VECTBL0
                        585
                        586   ;----- ESTABLISH BIOS SUBROUTINE CALL INTERRUPT VECTORS
                        587
E1EF BF4000             588           MOV     DI,OFFSET VIDEO_INT      ; SETUP ADDR TO INTR AREA
E1F2 0E                 589           PUSH    CS
E1F3 1F                 590           POP     DS                       ; SETUP ADDR OF VECTORS
E1F4 8C08               591           MOV     AX,DS                    ; SET AX=SEGMENT
E1F6 BE03FF90           592           MOV     SI,OFFSET VECTOR_TABLE+16  ; START WITH VIDEO ENTRY
E1FA B91000             593           MOV     CX,16
E1FD A5                 594   D3A:    MOVSW                            ; MOVE VECTOR TABLE TO RAM
E1FE 47                 595           INC     DI                       ; SKIP SEGMENT POINTER
E1FF 47                 596           INC     DI
E200 E2FB               597           LOOP    D3A
```

Commentaires.- 1^o) On sauvegarde le contenu du registre DS (ligne 576). On initialise le registre de compteur CX à 32 (ligne 577) puisqu'on va remplir la table pour 32 interruptions. Le premier vecteur d'interruption doit se placer à l'adresse 0 de la mémoire vive ; on initialise donc les registres DI et ES à 0 (lignes 578 et 579).

- 2^o) On communique l'adresse de la routine de service générique des 32 premières interruptions (lignes 580 à 584).

- 3^o) On change la routine de service des seize dernières interruptions matérielles : on initialise DI avec le décalage de la première d'entre elles, c'est-à-dire celle pour l'interruption graphique 10h (ligne 588) ; on identifie le segment des données avec le segment de code (lignes 589 et 590) ; on donne à AX la valeur du segment des données (ligne 591) ; on initialise SI avec la seizième donnée de la table vue précédemment (ligne 592) ; on initialise le registre de compteur CX à 16 (ligne 593) et on effectue la copie (lignes 594 à 597).

11.2.11 Détermination de la configuration

On poursuit par la détermination de la configuration indiquée par les commutateurs DIP, que l'on reporte dans la variable adéquate de la zone de communication du BIOS :

```
                    598  ;---------------------------------------------
                    599  ;       DETERMINE CONFIGURATION AND MFG. MODE  :
                    600  ;---------------------------------------------
                    601
E202 1F             602          POP      DS
E203 1E             603          PUSH     DS                  ; RECOVER DATA SEG
E204 E462           604          IN       AL,PORT_C           ; GET SWITCH INFO
E206 240F           605          AND      AL,00001111B        ; ISOLATE SWITCHES
E208 8AE0           606          MOV      AH,AL               ; SAVE
E20A B0AD           607          MOV      AL,10101101B        ; ENABLE OTHER BANK OF SWS.
E20C E661           608          OUT      PORT_B,AL
E20E 90             609          NOP
E20F E462           610          IN       AL,PORT_C
E211 B104           611          MOV      CL,4
E213 D2C0           612          ROL      AL,CL               ; ROTATE TO HIGH NIBBLE
E215 24F0           613          AND      AL,11110000B        ; ISOLATE
E217 0AC4           614          OR       AL,AH               ; COMBINE WITH OTHER BANK
E219 2AE4           615          SUB      AH,AH
E21B A31004         616          MOV      DATA_WORD[OFFSET EQUIP_FLAG],AX ; SAVE SWITCH INFO
```

Commentaires.- 1^{o}) On revient au segment des données (ligne 602), dont on maintient la sauvegarde sur la pile (ligne 603).

- 2^{o}) On récupère les informations des quatre premiers commutateurs DIP, reliés au port C du PPI (lignes 604 et 605). On les sauvegarde (ligne 606).

- 3^{o}) On active l'autre série de commutateurs DIP (ligne 607 et 608) en envoyant 10101101b sur le port B : lecture des commutateurs, pas d'horloge clavier, activation du renvoi à NMI lors d'une erreur de parité sur une carte d'extension, pas de vérification de parité de la RAM, mise en marche du moteur du lecteur de cassette, *lecture des quatre bits de SW2 à placer dans les bits 0 à 3 du port C*, fourniture du signal de temporisation au lecteur de cassette et mise en marche du temporisateur 2. On récupère (ligne 610), après un léger délai (ligne 609), ces valeurs.

- 4^{o}) On effectue une rotation (ligne 611) pour qu'elles se retrouvent sur le demi-octet de poids fort. On les combine avec celles de SW1 (ligne 614), on remet AH à 0 (ligne 615) et on sauvegarde la configuration dans la zone de communication du BIOS (ligne 616).

11.2.12 Test du clavier

11.2.12.1 Sous-routine de test du clavier

Un PC dont le clavier ne serait pas attaché ne servirait pas à grand chose. On vérifie donc qu'il y a bien un clavier attaché lors du démarrage. Pour cela on envoie un niveau bas au signal d'horloge durant 20 ms et le clavier doit renvoyer le code de recherche AAh, indiquant par là sa présence.

On utilise uns sous-routine pour effectuer ce test, qui doit donc renvoyer la valeur AAh dans le registre BL. Cette procédure KBD_RESET est définie à partir de la ligne 5454 :

```
                         5454
                         5455   ;----------------------------------------------------------------------
                         5456   ;       THIS PROCEDURE WILL SEND A SOFTWARE RESET TO THE KEYBOARD.     :
                         5457   ;       SCAN CODE 'AA' SHOULD BE RETURNED TO THE CPU.                  :
                         5458   ;----------------------------------------------------------------------
FA2A                     5459   KBD_RESET        PROC  NEAR
                         5460           ASSUME   DS:ABS0
FA2A B008                5461           MOV      AL,08H                        ; SET KBD CLK LINE LOW
FA2C E662                5462           OUT      PORT_B,AL                     ; WRITE 8255 PORT B
FA2E B95629              5463           MOV      CX,10582                      ; HOLD KBD CLK LOW FOR 20 MS
FA31                     5464   G8:
FA31 E2F2                5465           LOOP     G8                            ; LOOP FOR 20 MS
FA33 B0C8                5466           MOV      AL,0C8H                       ; SET CLK, ENABLE LINES HIGH
FA35 E661                5467           OUT      PORT_B,AL
FA37                     5468   SP_TEST:                                       ; ENTRY FOR MANUFACTURING TEST 2
FA37 B048                5469           MOV      AL,48H                        ; SET KBD CLK HIGH, ENABLE LOW
FA39 E661                5470           OUT      PORT_B,AL
FA3B B0FD                5471           MOV      AL,0FDH                       ; ENABLE KEYBOARD INTERRUPTS
FA3D E621                5472           OUT      INTA01,AL                     ; WRITE 8259 IMR
FA3F C6066B040D          5473           MOV      DATA_AREA[OFFSET INTR_FLAG] ;   RESET INTERRUPT INDICATOR
FA44 FB                  5474           STI                                    ; ENABLE INTERRUPTS
FA45 2BC9                5475           SUB      CX,CX                         ; SETUP INTERRUPT TIMEOUT CNT
FA47                     5476   G9:
FA47 F6066B0402          5477           TEST     DATA_AREA[OFFSET INTR_FLAG],02H ; DID A KEYBOARD INTR OCCUR?
FA4C 7502                5478           JNZ      G10                           ; YES - READ SCAN CODE RETURNED
FA4E E2F7                5479           LOOP     G9                            ; NO - LOOP TILL TIMEOUT
FA50                     5480   G10:
FA50 E460                5481           IN       AL,PORT_A                     ; READ KEYBOARD SCAN CODE
FA52 8AD8                5482           MOV      BL,AL                         ; SAVE SCAN CODE JUST READ
FA54 B0C8                5483           MOV      AL,0C8H                       ; CLEAR KEYBOARD
FA56 E661                5484           OUT      PORT_B,AL
FA58 C3                  5485           RET                                    ; RETURN TO CALLER
                         5486   KBD_RESET        ENDP
```

Commentaires.- 1^o) Le segment des données commence à 0h (ligne 5460).

- 2^o) On envoie 8h au port B du PIP (lignes 5461 et 5462) : on a 8h = 0000 1000b donc *on lit la valeur du clavier*, *pas d'horloge clavier* (ainsi que pas d'activation de renvoi à NMI, pas de vérification de la parité de la RAM, mise en marche du moteur du lecteur de cassette, pas de lecture de SW2, fourniture du temporisateur au lecteur de cassette et pas de mise en marche du temporisateur 2, ce qui ne nous intéresse pas vraiment).

- 3^o) L'horloge du clavier est maintenue à niveau bas durant 20 ms : le registre de compteur CX est initialisé à 10 582 (ligne 5463) et on boucle (lignes 5464 et 5465).

- 4^o) On envoie C8h au port B du PIP (lignes 5466 et 5467) : on a C8h = 1100 1000b donc *pas de lecture de la valeur du clavier*, *horloge clavier*. On envoie donc le signal d'horloge au clavier.

- 5^o) On envoie 48h au port B du PIP (lignes 5469 et 5470) : on a 48h = 0100 1000b donc *lecture de la valeur du clavier* et *horloge clavier*.

- 6^o) On active l'interruption matérielle du clavier : on envoie FDh au registre de masquage du PIC (lignes 5471 et 5472) ; on a FDh = 1111 1101 donc seule l'interruption matérielle IRQ1, qui correspond au clavier comme nous l'avons vu, est permise.

On met à jour l'indicateur de la zone de de communication du BIOS (ligne 5473) et on permet les interruptions matérielles masquables (ligne 5474).

Remarquons l'erreur dans le listing. On a :

```
FA3F C6066B040D         5473            MOV     DATA_AREA[OFFSET INTR_FLAG]    ; RESET INTERRUPT INDICATOR
```

au lieu de :

```
FA3F C6066B040D         5473            MOV     DATA_AREA[OFFSET INTR_FLAG],AL ; RESET INTERRUPT INDICATOR
```

Ce qui montre que le listing publié n'est pas une copie conforme de ce qui a été généré. Sinon l'erreur aurait été indiquée.

- 7^o) On attend une requête d'interruption matérielle du clavier (lignes 5475 à 5479).

- 8^o) Lorsqu'une interruption clavier intervient, on lit le code de recherche renvoyé (ligne 5481), que l'on sauvegarde dans le registre BL (ligne 5482), on désactive le clavier (lignes 5483 et 5484) et on revient à l'appelant (ligne 5485).

11.2.12.2 Test du clavier

On poursuit par le test du clavier. À l'origine on appelait la sous-routine précédente et on devait recevoir AAh dans le registre BL. En fait des claviers ont par la suite été fabriqués par des firmes autres qu'IBM. Dans ce cas ils doivent renvoyer 65h au lieu de AAh ; un code peut également être renvoyé dans ce cas :

```
E21E B099            617          MOV      AL,99H
E220 E663            618          OUT      CMD_PORT,AL
E222 E8D518          619          CALL     KBD_RESET                 ; SEE IF MFG. JUMPER IN
E225 80FBAA          620          CMP      BL,0AAH                   ; KEYBOARD PRESENT?
E228 7418            621          JE       E6
E22A 80FB65          622          CMP      BL,065H                   ; LOAD MFG. TEST REQUEST?
E22D 7503            623          JNE      D38
E22F E9EFFD          624          JMP      MFG_BOOT                  ; GO TO BOOTSTRAP IF SO
E232 B038            625   D38:   MOV      AL,38H
E234 E661            626          OUT      PORT_B,AL
E236 90              627          NOP
E237 90              628          NOP
E238 E460            629          IN       AL,PORT_A
E23A 24FF            630          AND      AL,0FFH                   ; WAS DATA LINE GROUNDED
E23C 7504            631          JNZ      E6
E23E FE061204        632                   DATA_AREA[OFFSET MFG_TST] ; SET MANUFACTURING TEST FLAG
                     633
```

Commentaires.- 1^o) On envoie au port de commande du PIP (lignes 617 et 618) l'octet 99h = 10011001b : on est en mode définition, mode 0 (c'est-à-dire port simple) pour les ports A et CU en entrée, mode 0 pour le port B en sortie et le port CL en entrée.

- 2^o) On appelle la procédure KBD_RESET.

- 3^o) Si un clavier IBM (ou compatible) est branché (et fonctionne), 'AA' est renvoyé dans le registre BL. On teste (ligne 620). Si c'est bien le cas (ligne 621), on va en E6, c'est-à-dire qu'on passe à la suite (ligne 644).

- 4^o) Si ce n'est pas le cas, on regarde s'il s'agit d'un clavier autre que celui d'IBM (ligne 622) et on effectue le rappatriement du code propre à ce clavier en appelant MFG_BOOT (ligne 624).

- 5^o) Sinon on envoie 38h au port B du PIP (lignes 623, 625 et 626). On a 38h = 0011 1000b donc on lit la valeur du clavier, pas d'horloge clavier, activation de renvoi à NMI, vérification de la parité de la RAM, mise en marche du moteur du lecteur de cassette, pas de lecture de SW2, pas de signal de temporisation pour le lecteur de cassette et pas de mise en marche du temporisateur 2.

- 6^o) On attend un peu (lignes 627 et 628) et on lit le code de recherche (ligne 629). Si la ligne est à la terre, on poursuit (ligne 631).

- 7^o) La ligne 632 servira dans le cas d'un clavier non IBM.

11.2.12.3 Cas d'un clavier autre que celui d'IBM

Nous venons de voir que dans le cas d'un clavier non IBM, il faut récupérer un code propre au fabricant, faisant l'objet de MFG_BOOT (pour *ManuFacturinG BOOT*) :

```
                   254  ;--------------------------------------------------------------------
                   255  ;       LOAD A BLOCK OF TEST CODE THROUGH THE KEYBOARD PORT           :
                   256  ;       FOR MANUFACTURING TEST.                                       :
                   257  ; THIS ROUTINE WILL LOAD A TEST (MAX LENGTH=FAFFH) THROUGH            :
                   258  ;       THE KEYBOARD PORT. CODE WILL BE LOADED AT LOCATION            :
                   259  ;       0000:0500. AFTER LOADING, CONTROL WILL BE TRANSFERED          :
                   260  ;       TO LOCATION 0000:0500. STACK WILL BE LOCATED JUST BELOW       :
                   261  ;       THE TEST CODE. THIS ROUTINE ASSUMES THAT THE FRIST 2          :
                   262  ;       BYTES TRANSFERED CONTAIN THE COUNT OF BYTES TO BE LOADED      :
                   263  ;       (BYTES 1=COUNT LOW, BYTE 2=COUNT HI.)                         :
                   264  ;--------------------------------------------------------------------
                   265
                   266  ;----- FIRST, GET THE COUNT
                   267
E021               268  MFG_BOOT:
E021 EB131A        269          CALL    SP_TEST             ; GET COUNT LOW
E024 8AFB          270          MOV     BH,BL               ; SAVE IT
E026 E80E1A        271          CALL    SP_TEST             ; GET COUNT HI
E029 8AEB          272          MOV     CH,BL
E02B 8ACF          273          MOV     CL,BH               ; CX NOW HAS COUNT
E02D FC            274          CLD                         ; SET DIR. FLAG TO INCRIMENT
E02E FA            275          CLI
E02F BF0005        276          MOV     DI,0500H            ; SET TARGET OFFSET (DS=0000)
E032 B0FD          277          MOV     AL,0FDH             ; UNMASK K/B INTERRUPT
E034 E621          278          OUT     INTA01,AL
E036 B00A          279          MOV     AL,0AH              ; SEND READ INT. REQUEST REG. CMD
E038 E620          280          OUT     INTA00,AL
E03A BA6100        281          MOV     DX,61H              ; SET UP PORT B ADDRESS
E03D B8CC4C        282          MOV     BX,4CCCH            ; CONTROL BITS FOR PORT B
E040 B402          283          MOV     AH,02H              ; K/B REQUEST PENDING MASK
E042               284  TST:
E042 BAC3          285          MOV     AL,BL
E044 EE            286          OUT     DX,AL               ; TOGGLE K/B CLOCK
E045 8AC7          287          MOV     AL,BH
E047 EE            288          OUT     DX,AL
E048 4A            289          DEC     DX                  ; POINT DX AT ADDR. 60 (KB DATA)
E049               290  TST1:
E049 E420          291          IN      AL,INTA00           ; GET IRR REG
E04B 22C4          292          AND     AL,AH               ; KB REQUEST PENDING?
E04D 74FA          293          JZ      TST1                ; LOOP TILL DATA PRESENT
E04F EC            294          IN      AL,DX               ; GET DATA
E050 AA            295          STOSB                       ; STORE IT
E051 42            296          INC     DX                  ; POINT DX BACK AT PORT B (61)
E052 E2EE          297          LOOP    TST                 ; LOOP TILL ALL BYTES READ
                   298
E054 EA00050000    299          JMP     MFG_TEST_RTN        ; FAR JUMP TO CODE THAT WAS JUST
                   300                                      ; LOADED
```

Commentaires.- 1^o) On appelle la procédur SP_TEST (définie à la ligne 5468 et déjà étudiée) pour lire l'octet de poids faible de la quantité de code à récupérer (ligne 269), que l'on place dans le registre BH (ligne 270).

On appelle une fois de plus la procédur SP_TEST pour lire cette fois-ci l'octet de poids fort de la quantité de code à récupérer (ligne 271), que l'on place dans le registre CH (ligne 272).

On place le contenu de BH dans CL (ligne 273), ce qui fait que maintenant CX contient la quantité de code à récupérer.

- 2^o) On décrira dans l'ordre croisant (ligne 274 ; la faute d'ortographe « incriment » au lieu de « increment » est présente dans le code). On masque toutes les interruptions masquable (ligne 275). Le décalage devra être pris à partir de l'emplacement mémoire 500h (ligne 276), emplacement réservé pour ce code comme nous l'avons déjà vu. On permet l'interruption matérielle du clavier (lignes 277 et 278). On envoie une requête de lecture au registre de commande (lignes 279 et 280). On place dans le registre DX l'adresse du port B du PPI (ligne 281). On place dans le registre BX les bits de contrôle pour le port B (ligne 282). On place dans AH le masque pour l'interruption du clavier (ligne 283). On active et désactive l'horloge du clavier (lignes 285 à 288) et DX contient maintenant l'adresse du port des données du clavier (ligne 289).

- 3^o) On attend une requête de la part du clavier (lignes 290 à 293). On récupère alors la donnée de celui-ci (ligne 294) et on la place en mémoire (ligne 295). On recommence tant qu'il y a des octets à récupérer (lignes 296 et 297).

- 4^o) On exécute le code qui vient d'être récupéré (ligne 299), c'est-à-dire à partir de la cellule 500h de la mémoire vive.

Rappelons que MFG_TEST_RETURN a été défini ligne 62 dans le segment ABSO.

11.2.13 Initialisation de l'affichage

On poursuit par l'initialisation de l'affichage :

```
                        ;-----------------------------------------------------
                        ;       INITIALIZE AND START CRT CONTROLLER (6845)    :
                        ;       TEST VIDEO READ/WRITE STORAGE.                :
                        ; DESCRIPTION                                         :
                        ;       RESET THE VIDEO ENABLE SIGNAL.                :
                        ;       SELECT ALPHANUMERIC MODE, 40 * 25, B & W.     :
                        ;       READ/WRITE DATA PATTERNS TO STG. CHECK STG    :
                        ;       ADDRESSABILITY.                               :
                        ; ERROR = 1 LONG AND 2 SHORTS BEEPS                   :
                        ;-----------------------------------------------------
E242                    E6:
E242 A11004                     MOV     AX,DATA_WORD[OFFSET EQUIP_FLAG] ;    GET SENSE SWITCH INFO
E245 50                         PUSH    AX                              ; SAVE IT
E246 B030                       MOV     AL,30H
E248 A31004                     MOV     DATA_WORD[OFFSET EQUIP_FLAG],AX
E24B 2AE4                       SUB     AH,AH
E24D CD10                       INT     10H                             ; SEND INIT TO B/W CARD
E24F B020                       MOV     AL,20H
E251 A31004                     MOV     DATA_WORD[OFFSET EQUIP_FLAF],AX
E254 2AE4                       SUB     AH,AH                           ; AND INIT COLOR CARD
E256 CD10                       INT     10H
E258 58                         POP     AX                              ; RECOVER REAL SWITCH INFO
E259 A31004                     MOV     DATA_WORD[OFFSET EQUIP_FLAG],AX ;    RESTORE IT
                                                                        ; AND CONTINUE
E25C 2430                       AND     AL,30H                          ; ISOLATE VIDEO SWS
E25E 7504                       JNZ     E7                              ; VIDEO SWS SET TO 0?
E260 BF4000                     MOV     DI,OFFSET VIDEO_INT             ; SET INT 10H TO DUMMY
E263 C7054BFF                   MOV     [DI],OFFSET DUMMY_RETURN        ; RETURN IF NO VIDEO CARD
E267 E9A000                     JMP     E18_1                           ; BYPASS VIDEO TEST
E26A                    E7:                                             ; TEST_VIDEO:
E26A 3C30                       CMP     AL,30H                          ; B/W CARD ATTACHED?
E26C 7408                       JE      E8                              ; YES - SET MODE FOR B/W CARD
E26E FEC4                       INC     AH                              ; SET COLOR MODE FOR COLOR CD
E270 3C20                       CMP     AL,20H                          ; 80x25 MODE SELECTED?
E272 7502                       JNE     E8                              ; NO - SET MODE FOR 40x25
E274 B403                       MOV     AH,3                            ; SET MODE FOR 80x25
E276 86E0               E8:     XCHG    AH,AL                           ; SET_MODE:
E278 50                         PUSH    AX                              ; SAVE VIDEO MODE ON STACK
E279 2AE4                       SUB     AH,AH                           ; INITIALIZE TO ALPHANUMERIC MD
E27B CD10                       INT     10H                             ; CALL VIDEO_IO
E27D 58                         POP     AX                              ; RESTORE VIDEO SENSE SW IN AH
E27E 50                         PUSH    AX                              ; RESAVE VALUE
E27F BB00B0                     MOV     BX,0B000H                       ; BEG VIDEO RAM ADDR B/W CD
E282 BAB803                     MOV     DX,388H                         ; MODE REG FOR B/W
E285 B90008                     MOV     CX,2048                         ; RAM WORD CNT FOR B/W CD
E288 B001                       MOV     AL,1                            ; SET MODE FOR BW CARD
E28A 80FC30                     CMP     AH,30H                          ; B/W VIDEO CARD ATTACHED?
E28D 7409                       JE      E9                              ; YES - GO TEST VIDEO STG
E28F B788                       MOV     BH,0B8H                         ; BEG VIDEO RAM ADDR COLOR CD
E291 BA0803                     MOV     DX,308H                         ; MODE REG FOR COLOR CD
E294 B520                       MOV     CH,20H                          ; RAM WORD CNT FOR COLOR CD
E296 FEC8                       DEC     AL                              ; SET MODE TO 0 FOR COLOR CD
E298                    E9:                                             ; TEST_VIDEO_STG:
E298 EE                         OUT     DX,AL                           ; DISABLE VIDEO FOR COLOR CD
E299 813E72043412               CMP     DATA_WORD[OFFSET RESET_FLAG],1234H ; POD INIT BY KBD RESET?
E29F 8EC3                       MOV     ES,BX                           ; POINT ES TO VIDEO RAM STG
E2A1 7407                       JE      E10                             ; YES - SKIP VIDEO RAM TEST
E2A3 8EDB                       MOV     DS,BX                           ; POINT DS TO VIDEO RAM STG
                                ASSUME  DS:NOTHING,ES:NOTHING
E2A5 E8C703                     CALL    STGTST_CNT                      ; GO TEST VIDEO R/W STG
E2A8 7546                       JNE     E17                             ; R/W STG FAILURE - BEEP SPK
                        ;----------------------------------------------
                        ;       SETUP VIDEO DATA ON SCREEN FOR VIDEO    :
                        ;       LINE TEST.                              :
                        ; DESCRIPTION                                   :
                        ;       ENABLE VIDEO SIGNAL AND SET MODE.       :
                        ;       DISPLAY A HORIZONTAL BAR ON SCREEN.     :
                        ;----------------------------------------------
E2AA                    E10:
E2AA 59                         POP     AX                  ; SET VIDEO SENSE SWS (AH)
```

```
E2AB 50               704        PUSH    AX                    ; SAVE IT
E2AC B400             705        MOV     AH,0                  ; ENABLE VIDEO AND SET MODE
E2AE CD10             706        INT     10H                   ; VIDEO
E2B0 882070           707        MOV     AX,7020H              ; WRT BLANKS IN REVERSE VIDEO
                      708
                      709
                      710
E2B3 EB11             711        JMP     SHORT E10A
E2C3                  712        ORG     0E2C3H
E2C3 E99915           713        JMP     NMI_INT
                      714
E2C6                  715  E10A:
E2C6 2BFF             716        SUB     DI,DI                 ; SETUP STARTING LOC
E2C8 B92800           717        MOV     CX,40                 ; NO. OF BLANKS TO DISPLAY
E2CB F3               718        REP     STOSW                 ; WRITE VIDEO STORAGE
E2CC AB
                      719  ;---------------------------------------
                      720  ;       CRT INTERFACE LINES TEST       :
                      721  ; DESCRIPTION                          :
                      722  ;       SENSE ON/OFF TRANSITION OF THE :
                      723  ;       VIDEO ENABLE AND HORIZONTAL    :
                      724  ;       SYNC LINES.                    :
                      725  ;---------------------------------------
E2CD 58               726        POP     AX                    ; GET VIDEO SENSE SW INFO
E2CE 50               727        PUSH    AX                    ; SAVE IT
E2CF 80FC30           728        CMP     AH,30H                ; B/W CARD ATTACHED?
E2D2 BABA03           729        MOV     DX,03BAH              ; SETUP ADDR OF BW STATUS PORT
E2D5 7403             730        JE      E11                   ; YES - GO TEST LINES
E2D7 BADA03           731        MOV     DX,03DAH              ; COLOR CARD IS ATTACHED
E2DA                  732  E11:                                ; LINE_TST:
E2DA B408             733        MOV     AH,8
E2DC                  734  E12:                                ; OFLOOP_CNT:
E2DC 2BC9             735        SUB     CX,CX
E2DE                  736  E13:
E2DE EC               737        IN      AL,DX                 ; READ CRT STATUS PORT
E2DF 22C4             738        AND     AL,AH                 ; CHECK VIDEO/HORZ LINE
E2E1 7504             739        JNZ     E14                   ; ITS ON - CHECK If IT GOES OFF
E2E3 E2F9             740        LOOP    E13                   ; LOOP TILL ON OR TIMEOUT
E2E5 EB09             741        JMP     SHORT E17             ; GO PRINT ERROR MSG
E2E7                  742  E14:
E2E7 28C9             743        SUB     CX,CX
E2E9                  744  E15:
E2E9 EC               745        IN      AL,DX                 ; READ CRT STATUS PORT
E2EA 22C4             746        AND     AL,AH                 ; CHECK VIDEO/HORZ LINE
E2EC 7411             747        JZ      E16                   ; ITS ON - CHECK NEXT LINE
E2EE E2F9             748        LOOP    E15                   ; LOOP IF OFF TILL IT GOES ON
E2F0                  749  E17:                                ; CRT_ERR:
E2F0 1F               750        POP     DS
E2F1 1E               751        PUSH    DS
E2F2 C606150006       752        MOV     DS:MFG_ERR_FLAG,06H   ; <><><>CRT ERRCHKPT. 06<><><>
E2F7 BA0201           753        MOV     DX,102H
E2FA E8D816           754        CALL    ERR_BEEP              ; GO BEEP SPEAKER
E2FD EB06             755        JMP     SHORT E18
E2FF                  756  E16:                                ; NXT_LINE:
E2FF B103             757        MOV     CL,3                  ; GET NEXT BIT TO CHECK
E301 D2EC             758        SHR     AH,CL
E303 75D7             759        JNZ     E12                   ; GO CHECK HORIZONTAL LINE
E305                  760  E18:                                ; DISPLAY_CURSOR:
E305 58               761        POP     AX                    ; GET VIDEO SENSE SWS (AH)
E306 B400             762        MOV     AH,0                  ; SET MODE AND DISPLAY CURSOR
E308 CD10             763        INT     10H                   ; CALL VIDEO I/O PROCEDURE
```

<u>Commentaires</u>.- 1^o) On commence par essayer d'initialiser la carte graphique présente.

Pour cela, on place le mot de configuration EQUIP_FLAG, se trouvant dans la zone de communication du BIOS, dans le registre AX (ligne 645), on le sauvegarde sur la pile (ligne 646) et on remplace son octet de poids faible par 30h. On fait appel à l'interruption 10h (ligne 650), fonction 0h (ligne 649) avec le paramètre 30h (ligne 648) pour initialiser la carte MDA.

On remplace ensuite son octet de poids faible par 20h. On change de même le mot de configuration (ligne 652). On fait appel à l'interruption 10h (ligne 653), fonction 0h avec le paramètre 20h (ligne 651) pour initialiser la carte CGA.

On récupère le mot de configuration originel (ligne 655) et on le replace dans la zone de communication du BIOS (ligne 656).

- 2^o) Si le mot de configuration spécifie qu'aucune carte graphique n'est présente (lignes 658 et 659), on remplace la routine de service de l'interruption graphique 10h par la routine générique (lignes 660 et 661) et on va à la fin de l'initialisation de l'affichage (ligne 662 qui renvoie à la ligne 764).

- 3^o) On va initialiser la carte graphique présente en mode texte, noir et blanc, 25 lignes de 40 caractères.
Si une carte MDA est présente (lignes 664 et 665), on continue en E8 avec l'affectation du mode. Sinon une carte CGA est présente. Si le mode 40x25 n'est pas actif, on passe au mode 40x25 (lignes 666 à 669).

- 4^o) On affecte le mode.
Pour cela, on échange les contenus des registres AH et AL (ligne 670), on sauvegarde le mode choisi sur la pile (ligne 671) et on fait appel à la fonction 0 (ligne 672) de l'interruption 10h (ligne 673).
On restaure le mode dans AH (674) sans le retirer de la pile (ligne 675).

- 5^o) On prépare les paramètres pour tester la mémoire graphique : début de l'adresse de la mémoire graphique de la carte MDA dans le registre BX (ligne 676), le port de son registre de mode dans le registre DX (ligne 677), la capacité de sa mémoire graphique dans le registre CX (ligne 678), son mode dans le registre AL (ligne 679) et on passe au test de sa mémoire graphique si cette carte est présente (lignes 680 et 681, renvoyant à la ligne 686).
Sinon on place les paramètres pour la carte CGA (lignes 682 à 685).

- 6^o) On teste la mémoire graphique. Pour cela on désactive la carte graphique (ligne 687) et le registre ES pointe sur le début de la mémoire graphique (ligne 689). S'il s'agit d'un démarrage à chaud (ligne 689), on n'effectue pas le test de la mémoire graphique (ligne 690, renvoyant à la ligne 702). Le registre DS pointe également sur le début de la mémoire graphique (ligne 691) et on fait appel à la sous-routine STGTST_CNT (ligne 693), déjà étudiée à propos du test de la RAM. S'il y a un problème, on envoie un bip (ligne 694, renvoyant à la ligne 749).

- 7^o) On se place dans le mode choisi. Pour cela on récupère le mode (ligne 703) sans l'enlever de la pile (ligne 704) et on fait appel à la fonction 0 (ligne 705) de l'interruption graphique (ligne 706).

- 8^o) On affiche une ligne blanche. Pour cela on se prépare à écrire des espaces en video inverse (ligne 707). On saute quelques lignes (ligne 711) (par compatibilité avec un BIOS antérieur ?). On initialise le registre DI à zéro (ligne 716), le registre de compteur à 40 (ligne 717) pour 40 caractères à afficher, et on copie dans la mémoire graphique (ligne 718).

- 9^o) On récupère les informations concernant la configuration sur la pile (ligne 726) tout en les maintenant sur celle-ci (ligne 727). Si une carte MDA est présente (ligne 728), on place dans le registre DX l'adresse de port de statut de celle-ci (ligne 729). Sinon on y place l'adresse de port de statut de la carte CGA (ligne 731).
On place 8 dans le registre AH (ligne 733), on initialise le registre de compteur CX à zéro (ligne 735). On attend un rebroussement horizontal (lignes 736 à 740). Si celui-ci ne survient pas dans un certain laps de temps, on affiche un message d'erreur (ligne 741).
On réinitialise le compteur à zéro (ligne 743). Au rebroussement suivant (lignes 744 à 748), on passe à la ligne suivante (lignes 747 et 756 à 759) tant qu'il y en a.
Une fois toutes les lignes testées, on récupère le mode et on l'affecte (lignes 760 à 763).

- 10^o) On affiche le curseur.

11.2.14 Vérification et exécution des ROM optionnelles

Nous avons jusqu'à maintenant vérifié le seul module de ROM obligatoire. Il peut y avoir d'autres modules de ROM, autrement dit des ROM *optionnelles*. Pour les ROM optionnelles, les deux premiers octets contiennent un code de reconnaissance et le troisième octet la taille du code (en multiple de 512 KiO). Le code AA55h spécifie qu'il faut exécuter la ROM après l'avoir vérifiée ([IBM-83], p. 2–10).

11.2.14.1 Passage en revue des ROM optionnelles

Après l'initialisation de l'affichage, on poursuit par le passage en revue des ROM optionnelles :

```
E30A                 764  E18_1:
E30A BA00C0          765          MOV     DX,0C000H
E30D                 766  E18A:
E30D 8EDA            767          MOV     DS,DX
E30F 2BDB            768          SUB     BX,BX
E311 8807            769          MOV     AX,[BX]                ; GET FIRST 2 LOCATIONS
E313 53              770          PUSH    BX
E314 5B              771          POP     BX                     ; LET BUS SETTLE
E315 3D55AA          772          CMP     AX,0AA55H              ; PRESENT?
E318 7505            773          JNZ     E18B                   ; NO? GO LOOK FOR OTHER MODULES
E31A E83616          774          CALL    ROM_CHECK              ; GO SCAN MODULE
E31D EB04            775          JMP     SHORT E18C
E31F                 776  E18B:
E31F 81C28000        777          ADD     DX,0080H               ; POINT TO NEXT 2K BLOCK
E323                 778  E18C:
E323 81FA00C8        779          CMP     DX,0C800H              ; TOP OF VIDEO ROM AREA YET?
E327 7CE4            780          JL      E18A                   ; GO SCAN FOR ANOTHER MODULE
```

Commentaires.- 1^o) Pour cela on initialise le registre DS avec C000h (lignes 765 et 766), c'est-à-dire le début de l'espace mémoire consacré aux ROM optionnelles éventuelles. On place dans AX le code de cette ROM, c'est-à-dire les deux premiers octets de ce segment (lignes 768 et 769) et on attend un petit peu (lignes 770 et 771).

- 2^o) Si ce code est AA55h (ligne 772), il faut la vérifier puis en exécuter le code ; pour cela on appelle la sous-routine ROM_CHECK (ligne 774 renvoyant à la ligne 5283), que nous étudierons dans la sous-section suivante. Au retour de cette sous-routine, si on n'est pas arrivé à la fin de l'espace mémoire consacré aux ROM optionnelles (ligne 779), on passe au bloc suivant (ligne 780).

- 3^o) Si ce code n'est pas AA55h, on passe au bloc suivant (lignes 776 à 780).

11.2.14.2 Vérification des modules optionnels de ROM

La sous-routine ROM_CHECK est définie à partir de la ligne 5283, le registre BX pointant sur le décalage à partir duquel on veut vérifier :

```
                       5279  ;--------------------------------------------------------------------
                       5280  ; THIS ROUTINE CHECKSUMS OPTIONAL ROM MODULES AND                   :
                       5281  ; IF CHECKSUM IS OK, CALLS INIT/TEST CODE IN MODULE                 :
                       5282  ;--------------------------------------------------------------------
F953                   5283  ROM_CHECK        PROC   NEAR
F953 B84000            5284          MOV      AX,DATA                   ; POINT ES TO DATA AREA
F956 8EC0              5285          MOV      ES,AX
F958 2AE4              5286          SUB      AH,AH                     ; ZERO OUT AH
F95A 8A4702            5287          MOV      AL,[BX+2]                 ; GET LENGTH INDICATOR
F95D B109              5288          MOV      CL,09H                    ; MULTIPLY BY 512
F95F D3E0              5289          SHL      AX,CL
F961 8BC8              5290          MOV      CX,AX                     ; SET COUNT
F963 51                5291          PUSH     CX                        ; SAVE COUNT
F964 89B0400           5292          MOV      CX,4                      ; ADJUST
F967 D3E8              5293          SHR      AX,CL
F969 0300              5294          ADD      DX,AX                     ; SET POINTER TO NEXT MODULE
F96B 59                5295          POP      CX                        ; RETRIVE COUNT
F96C EB86FF            5296          CALL     ROS_CHECKSUM_CNT          ; DO CHECKSUM
F96F 7406              5297          JZ       ROM_CHECK_1
F971 EB5700            5298          CALL     ROM_ERR                   ; POST CHECKSUM ERROR
F974 EB1490            5299          JMP      ROM_CHECK_END             ; AND EXIT
F977                   5300  ROM_CHECK_1:
F977 52                5301          PUSH     DX                        ; SAVE POINTER
F978 26C70667000300    5302          MOV      ES:IO_ROM_INIT,0003H      ; LOAD OFFSET
F97F 268C1E6900        5303          MOV      ES:IO_ROM_SEG,DS          ; LOAD SEGMENT
F984 26FF1E6700        5304          CALL     DWORD PTR ES:IO_ROM_INIT  ; CALL INIT./TEST ROUTINE
F989 5A                5305          POP      DX
F98A                   5306  ROM_CHECK_END:
F98A C3                5307          RET                                ; RETURN TO CALLER
                       5308  ROM_CHECK        ENDP
                       5309
```

Commentaires.- 1^o) On pointe le registre ES sur la zone de communication du BIOS (lignes 5284 et 5285), on initialise le registre AH à zéro (ligne 5286). On récupère la quantité de ce que l'on veut vérifier dans le registre CX (lignes 5282 à 5290). On sauvegarde cette quantité sur la pile (ligne 5291). On ajuste cette quantité (lignes 5292 et 5293) pour pointer sur le module suivant (ligne 5294) et on récupère la quantité voulue (ligne 5295).

- 2^o) On appelle la sous-routine ROS_CHECKSUM_CNT pour effectuer la vérification (ligne 5296, commençant ligne 5238 et étudiée au chapitre 3). On appelle la sous-routine ROM_ERR si la somme de contrôle n'est pas trouvée (lignes 5297 et 5298), qui commence ligne 1387 et que nous étudions ci-après, puis on termine (ligne 5299).

- 3^o) Si tout se passe bien lors de la vérification, on sauve le pointeur (ligne 5301), on place le décalage 3h dans la variable IO_ROM_INIT de la zone de communication du BIOS (ligne 5302, définie ligne 180) et le segment dans la variable IO_ROM_SEG (ligne 5303, définie ligne 181), puis on fait appel au code de cette routine (ligne 5304).

11.2.14.3 La sous-routine d'erreur

La sous-routine ROM_ERR est définie à partir de la ligne 1387 :

```
                     1384  ;-----------------------------------------------------------------
                     1385  ; PRINT ADDRESS AND ERROR MESSAGE FOR ROM CHECKSUM ERRORS        :
                     1386  ;-----------------------------------------------------------------
E6CB                 1387  ROM_ERR PROC     NEAR
E6CB 52              1388          PUSH     DX                     ; SAVE POINTER
E6CC 50              1389          PUSH     AX
E6CD 8CDA            1390          MOV      DX,DS                  ; GET ADDRESS POINTER
E6CF 2688361500      1391          MOV      ES:MFG_ERR_FLAG,DH     ; <><><><><><><><><><><><><><>
                     1392                                          ; <><><>CHECKPOINTS C0->F4<><><>
E6D4 81FA00C8        1393          CMP      DX,0C800H              ; CRT CARD IN ERROR?
E6D8 7C0D            1394          JL       ROM_ERR_BEEP           ; GIVE CRT CARD FAIL BEEP
E6DA E8FD18          1395          CALL     PRT_SEG                ; PRINT SEGEMENT IN ERROR
E6DD BE0AF990        1396          MOV      SI,OFFSET F3A          ; DISPLAY ERROR MSG
E6E1 E8C512          1397          CALL     E_MSG
E6E4                 1398  ROM_ERR_ERR:
E6E4 58              1399          POP      AX
E6E5 5A              1400          POP      DX
E6E6 C3              1401          RET
E6E7                 1402  ROM_ERR_BEEP:
E6E7 BA0201          1403          MOV      DX,0102H               ; BEEP 1 LONG, 2 SHORT
E6EA E8EB12          1404          CALL     ERR_BEEP
E6ED EBF5            1405          JMP      SHORT ROM_ERR_END
                     1406  ROM_ERR ENDP
                     1407
```

Commentaires.- 1^o) On sauvegarde les valeurs de DX (ligne 1388) et de AX (ligne 1389), puisqu'on va utiliser ces registres, et on place la valeur de DS dans DX (ligne 1390). On place la valeur de DH dans la variable MFG_ERR_FLAG de la zone de communication du BIOS (ligne 1391, variable définie à la ligne 86).

- 2^o) Si l'erreur est intervenue lors du test de la carte graphique (lignes 1393 et 1394), on va en 1402 pour émettre des bips, ce que nous étudierons ci-dessous (ERR_BEEP est défini ligne 5399).

- 3^o) Sinon l'erreur est intervenue lors du test de la mémoire vive. On appelle la sous-routine PRT_SEG (ligne 1395 renvoyant à la ligne 5939) pour afficher le numéro du segment fautif (la faute d'orthographe « SEGEMENT » se trouve dans le BIOS).

- 4^o) Le registre SI prend (ligne 1396) la valeur du décalage de l'étiquette F3A (définie à la ligne 5252) et on fait appel à la sous-routine E_MSG (ligne 1397 renvoyant à la ligne 5351). On restaure les valeurs de AX et de DX et on termine la sous-routine (lignes 1399 à 1401).

11.2.14.4 Sous-routine d'impression d'un nombre de deux chiffres hexadécimaux

La sous-routine XPC_BYTE (pour *tranSLate and Print asCii*) est définie à partir de la ligne 5315 :

```
                     5310  ;-----------------------------------------------------------------
                     5311  ; CONVERT AND PRINT ASCII CODE                                   :
                     5312  ;       AL MUST CONTAIN NUMBER TO BE CONVERTED.                  :
                     5313  ;       AX AND BX DESTROYED.                                     :
                     5314  ;-----------------------------------------------------------------
F98B                 5315  XPC_BYTE        PROC     NEAR
F98B 50              5316          PUSH    AX                    ; SAVE FOR LOW NIBBLE DISPLAY
F98C B104            5317          MOV     CL,4                  ; SHIFT COUNT
F98E D2E8            5318          SHR     AL,CL                 ; NIBBLE SWAP
F990 E80300          5319          CALL    XLAT_PR               ; DO THE HIGH NIBBLE DISPLAY
F993 58              5320          POP     AX                    ; RECOVER THE NIBBLE
F994 240F            5321          AND     AL,0FH                ; ISOLATE TO LOW NIBBLE
                     5322                                        ; FALL INTO LOW NIBBLE CONVERSION
F996                 5323  XLAT_PR PROC    NEAR
F996 D490            5324          ADD     AL,090H               ; ADD FIRST CONVERSION FACTOR
F998 27              5325          DAA                           ; ADJUST FOR NUMERIC AND ALPHA RANGE
F999 1440            5326          ADC     AL,040H               ; ADD CONVERSION AND ADJUST LOW NIBBLE
F99B 27              5327          DAA                           ; ADJUST HIGH NIBBLE TO ASCHI RANGE
F99C                 5328  PRT_HEX PROC    NEAR
F99C B40E            5329          MOV     AH,14                 ; DISPLAY CHARACTER IN AL
F99E B700            5330          MOV     BH,0
F9A0 CD10            5331          INT     10H                   ; CALL VIDEO_IO
F9A2 C3              5332          RET
                     5333  PRT_HEX ENDP
                     5334  XLAT_PR ENDP
                     5335  XPC_BYTE        ENDP
```

Commentaires.- 1^o) Lors de l'appel de la sous-routine, le registre AL contient le nombre à afficher. Il s'agit d'un octet, donc de deux chiffres hexadécimaux à afficher. Il faut le transformer en son équivalent ASCII. Pour cela, on sauvegarde ce code sur la pile (ligne 5316), on fait passer ce code dans le registre AH (lignes 5317 et 5318) et on appelle la sous-routine XLAT_PR (ligne 5319) pour traduire le demi-octet de poids fort.

- 2^o) La procédure XLAT_PR ajoute 90h (lignes 5324 et 5325) puis 40h (lignes 5326 et 5327) et on affiche en utilisant l'interruption graphique (lignes 5329 à 5332).

- 3^o) On récupère le contenu de AX (ligne 5320), on isole le demi-octet de poids faible (ligne 5321) et on recommence.

11.2.14.5 Sous-routine d'impression du numéro du segment fautif

La sous-routine PRT_SEG est définie à partir de la ligne 5939 :

```
                        5935  ;-----------------------------------------------------------------
                        5936  ;       PRINT A SEGMENT VALUE TO LOOK LIKE A 20 BIT ADDRESS      :
                        5937  ;       DX MUST CONTAIN SEGMENT VALUE TO BE PRINTED              :
                        5938  ;-----------------------------------------------------------------
FFDA                    5939  PRT_SEG PROC    NEAR
FFDA 8AC6               5940          MOV     AL,DH                   ; GET MSB
FFDC E8ACF9             5941          CALL    XPC_BYTE
FFDF 8AC2               5942          MOV     AL,DL                   ; LSB
FFE1 E8A7F9             5943          CALL    XPC_BYTE
FFE4 B03D               5944          MOV     AL,'0'                  ; PRINT A '0'
FFE6 E8B3F9             5945          CALL    PRT_HEX
FFE9 802D               5946          MOV     AL,' '                  ; SPACE
FFEB E8AEF9             5947          CALL    PRT_HEX
FFEE C3                 5948          RET
                        5949  PRT_SEG ENDP
                        5950
----                    5951  CODE    ENDS
                        5952
```

<u>Commentaires</u>.- 1^o) Lors de l'appel de la sous-routine, le registre DX contient le numéro de segment à afficher, qui occupe donc deux octets et non plus un seul octet comme dans la procédure précédente. On place l'octet de poids fort dans le registre AL (ligne 5940) et on appelle la procédure précédente (ligne 5941) pour l'afficher. On place l'octet de poids faible dans le registre AL (ligne 5942) et on appelle la procédure précédente (ligne 5943) pour l'afficher.

- 2^o) On affiche de même un zéro (lignes 5944 et 5945) puisqu'il faut multiplier le numéro de segment par 16 pour obtenir une adresse puis un espace (lignes 5946 et 5947) pour la séparer de la suivante.

11.2.14.6 Sous-routine d'impression du message d'erreur

La sous-routine P_MSG (pour *Print MeSsaGe*) est définie à partir de la ligne 5373 :

```
                        5372
F9CA                    5373  P_MSG   PROC    NEAR
F9CA                    5374  G12A:
F9CA 2E8404             5375          MOV     AL,CS:[SI]              ; PUT CHAR IN AL
F9CD 46                 5376          INC     SI                      ; POINT TO NEXT CHAR
F9CE 50                 5377          PUSH    AX                      ; SAVE PRINT CHAR
F9CF E8CAFF             5378          CALL    PRT_HEX                 ; CALL VIDEO_IO
F9D2 58                 5379          POP     AX                      ; RECOVER PRINT CHAR
F9D3 3CDA               5380          CMP     AL,10                   ; WAS IT LINE FEED?
F9D5 75F3               5381          JNE     G12A                            ; NO,KEEP PRINTING STRING
F9D7 C3                 5382          RET
                        5383  P_MSG   ENDP
                        5384
```

<u>Commentaires</u>.- 1^o) Lorsqu'on fait appel à cette procédure, le message se trouve dans le segment de code CS au décalage SI et se termine par un passage à la ligne. On place dans le registre AL le premier caractère à afficher (ligne 5373), on le sauvegarde (ligne 5377) et on l'affiche en faisant appel à la procédure PRT_HEX (ligne 5378) étudiée ci-dessus.

- 2^o) On récupère le caractère suivant à imprimer (ligne 5379). S'il s'agit d'un passage à la ligne (ligne 5380), on a terminé (ligne 5382). Sinon, on passe au caractère suivant.

11.2.14.7 Sous-routine d'impression de message d'erreur

La sous-routine E_MSG est définie à partir de la ligne 5351 :

```
                    5343  ;---------------------------------------------------------------
                    5344  ; THIS SUBROUTINE WILL PRINT A MESSAGE ON THE DISPLAY           :
                    5345  ;                                                               :
                    5346  ; ENTRY REQUIREMENTS:                                           :
                    5347  ;       SI = OFFSET(ADDRESS) OF MESSAGE BUFFER                  :
                    5348  ;       CX = MESSAGE BYTE COUNT                                 :
                    5349  ;       MAXIMUM MESSAGE LENGTH IS 36 CHARACTERS                 :
                    5350  ;---------------------------------------------------------------
F9A9                5351  E_MSG   PROC    NEAR
F9A9 8BEE           5352          MOV     BP,SI                   ; SET BP NON-ZERO TO FLAG ERR
F9AB E81C00         5353          CALL    P_MSG                   ; PRINT MESSAGE
F9AE 1E             5354          PUSH    DS
F9AF E8A700         5355          CALL    DDS
F9B2 A01D00         5356          MOV     AL,BYTE PTR EQUIP_FLAG  ; LOOP/HALT ON ERROR
F9B5 2401           5357          AND     AL,01H                  ; SWITCH ON?
F9B7 750F           5358          JNZ     G12                     ; NO - RETURN
F9B9                5359  MFG_HALT:
F9B9 FA             5360          CLI                             ; YES - HALT SYSTEM
F9BA B089           5361          MOV     AL,89H
F9BC E663           5362          OUT     CMD_PORT,AL
F9BE B085           5363          MOV     AL,10000101B            ; DISABLE KB
F9C0 E661           5364          OUT     PORT_B,AL
F9C2 A01500         5365          MOV     AL,MFG_ERR_FLAG         ; RECOVER ERROR INDICATOR
F9C5 E660           5366          OUT     PORT_A,AL               ; SET INTO 8255 REG
F9C7 F4             5367          HLT                             ; HALT SYS
F9C8                5368  G12:
F9C8 1F             5369          POP     DS                      ; WRITE_MSG:
F9C9 C3             5370          RET
                    5371  E_MSG   ENDP
```

Commentaires.- 1^o) On sauvegarde le contenu de SI dans le registre BP (ligne 5352), puisque, comme nous venons de le voir, le contenu de SI sera détruit par l'appel de P_MSG, et on fait appel à la sous-routine précédente pour afficher (ligne 5353).

- 2^o) On sauvegarde l'adresse du segment des données (ligne 5354) et on se place dans le segment des données de la zone de communication du BIOS (ligne 5355). On place le contenu de la variable d'équipement de la zone de communication du BIOS dans le registre AL (ligne 5356) pour regarder si on doit s'arrêter ou recommencer lorsqu'on rencontre une erreur (ligne 5357). Si on peut continuer, on restaure la valeur de DS et on retourne à l'appelant (lignes 5358 et 5368 à 5370).

- 3^o) Sinon on arrête le système. Pour cela, on ne permet plus les interruptions masquables (ligne 5360), on envoie 89h au port de commande du PPI (lignes 5361 et 5362) pour ? ?, on désactive le clavier (lignes 5363 et 5364), on place l'indicateur d'erreur sur le port A du PPI (lignes 5365 et 5366) et on s'arrête (ligne 5367).

11.2.14.8 Émission d'un bip d'erreur

La sous-procédure BEEP est définie ligne 5434 :

```
                        5432  ;--- ROUTINE TO SOUND BEEPER
                        5433
FA08                    5434  BEEP    PROC     NEAR
FA08 B0B6               5435          MOV      AL,10110110B          ; SEL TIM 2,LSB,MSB,BINARY
FA0A E643               5436          OUT      TIMER+3,AL            ; WRITE THE TIMER MODE REG
FA0C B83305             5437          MOV      AX,533H               ; DIVISOR FOR 1000 HZ
FA0F E642               5438          OUT      TIMER+2,AL            ; WRITE TIMER 2 CNT - LSB
FA11 8AC4               5439          MOV      AL,AH
FA13 E642               5440          OUT      TIMER+2,AL            ; WRITE TIMER 2 CNT - MSB
FA15 E461               5441          IN       AL,PORT_B             ; GET CURRENT SETTING OF PORT
FA17 BAE0               5442          MOV      AH,AL                 ; SAVE THAT SETTINGH
FA19 0C03               5443          OR       AL,03                 ; TURN SPEAKER ON
FA1B E661               5444          OUT      PORT_B,AL
FA1D 2BC9               5445          SUB      CX,CX                 ; SET CNT TO WAIT 500 MS
FA1F                    5446  G7:
FA1F E2FE               5447          LOOP     G7                    ; DELAY BEFORE TURNING OFF
FA21 FECB               5448          DEC      BL                    ; DELAY CNT EXPIRED?
FA23 75FA               5449          JNZ      G7                    ; NO - CONTINUE BEEPING SPK
FA25 8AC4               5450          MOV      AL,AH                 ; RECOVER VALUE OF PORT
FA27 E661               5451          OUT      PORT_B,AL
FA29 C3                 5452          RET                            ; RETURN TO CALLER
                        5453  BEEP    ENDP
                        5454
```

Commentaires.- 1^o) On envoie l'octet de contrôle 10110110b au temporisateur 8253 (lignes 5435 et 5436), c'est-à-dire 10b donc temporisateur 2, 11b donc LSB puis MSB, 011b donc signal carré, 0b donc compteur en binaire. On envoie le diviseur 533h (lignes 5437 à 5440).

- 2^o) On sauvegarde dans le registre AH la configuration actuelle du port B du PIP (lignes 5441 à 5442 ; la faute d'ortographe « SETTINGH » est bien dans le listing), on met en marche le haut-parleur (lignes 5443 et 5444) durant un multiple de 500 ms (lignes 5445 à 5447), le multiplicateur étant contenu dans le registre BL (lignes 5448 et 5449), puis on revient à la configuration originelle du port B (lignes 5450 à 5453).

La procédure ERR_BEEP est définie ligne 5399 :

```
                    5385  ;----------------------------------------------
                    5386  ; INITIAL RELIABILITY TEST -- SUBROUTINES      :
                    5387  ;----------------------------------------------
                    5388          ASSUME   CS:CODE,DS:DATA
                    5389  ;----------------------------------------------
                    5390  ;       SUBROUTINES FOR POWER ON DIAGNOSTICS   :
                    5391  ;---------------------------------------------------------------------
                    5392  ;       THIS PROCEDURE WILL ISSUE ONE LONG TONE (3 SECS) AND ONE OR    :
                    5393  ;       MORE SHORT TONES (1 SEC) TO INDICATE A FAILURE ON THE PLANAR   :
                    5394  ;       BOARD, A BAD RAM MODULE, OR A PROBLEM WITH THE CRT.            :
                    5395  ; ENTRY PARAMETERS:                                                    :
                    5396  ;       DH = NUMBER OF LONG TONES TO BEEP                              :
                    5397  ;       DL = NUMBER OF SHORT TONES TO BEEP.                            :
                    5398  ;---------------------------------------------------------------------
F9D8                5399  ERR_BEEP PROC    NEAR
F9D8 9C             5400          PUSHF                       ; SAVE FLAGS
F9D9 FA             5401          CLI                         ; DISABLE SYSTEM INTERRUPTS
F9DA 1E             5402          PUSH     DS                 ; SAVE DS REG CONTENTS
F9DB E67800         5403          CALL     DDS
F9DE 0AF6           5404          OR       DH,DH              ; ANY LONG ONES TO BEEP
F9E0 7414           5405          JZ       G3                 ; NO. DO THE SHORT ONES
F9E2                5406  G1:                                 ; LONG_BEEP:
F9E2 B306           5407          MOV      BL,6               ; COUNTER FOR BEEPS
F9E4 E82100         5408          CALL     BEEP               ; DO THE BEEP
F9E7                5409  G2:
F9E7 E2F2           5410          LOOP     G2                 ; DELAY BETWEEN BEEPS
F9E9 FECE           5411          DEC      DH                 ; ANY MORE TO DO
F9EB 75F5           5412          JNZ      G1                 ; DO IT
F9ED 803E120001     5413          CMP      MFG_TST,1          ; MFG TEST MODE?
F9F2 7502           5414          JNE      G3                 ; YES - CONTINUE BEEPING SPEAKER
F9F4 EBC3           5415          JMP      MFG_HALT           ; STOP BLINKING LED
F9F6                5416  G3:                                 ; SHORT_BEEP:
F9F6 B301           5417          MOV      BL,1               ; COUNTER FOR A SHORT BEEP
F9F8 E80D00         5418          CALL     BEEP               ; DO THE SOUND
F9FB                5419  G4:
F9FB E2FE           5420          LOOP     G4                 ; DELAY BETWEEN BEEPS
F9FD FECA           5421          DEC      DL                 ; DONE WITH SHORTS
F9FF 75F5           5422          JNZ      G3                 ; DO SOME MORE
FA01                5423  G5:
FA01 E2FE           5424          LOOP     G5                 ; LONG DELAY BEFORE RETURN
FA03                5425  G6:
FA03 E2F2           5426          LOOP     G6
FA05 1F             5427          POP      DS                 ; RESTORE ORIG CONTENTS OF DS
FA06 9D             5428          POPF                        ; RESTORE FLAGS TO ORIG SETTINGS
FA07 C3             5429          RET                         ; RETURN TO CALLER
                    5430  ERR_BEEP         ENDP
                    5431
```

Commentaires.- 1^o) On sauvegarde les indicateurs sur la pile (ligne 5400), on désactive les interruptions masquables (ligne 5401), on sauvegarde le contenu de DS (ligne 5402) et on prend la zone de communication du BIOS comme segment des données (ligne 5403).

- 2^o) On émet les DH bips longs (lignes 5404 à 5412). Si on ne se trouve pas dans le mode de test (ligne 5413), on arrête de faire clignoter les LEDs (lignes 5414 et 5415, renvoyant à la ligne 5359 étudiée ci-dessus).

- 3^o) On émet ensuite les DL bips courts (lignes 5416 à 5422), on attend un peu (lignes 5423 et 5426), on restaure le segment des données (ligne 5427), les indicateurs (ligne 5428) et on revient à l'appelant (ligne 5429).

11.2.15 Test du contrôleur d'interruption

Une fois le passage en revue des ROM optionnelles effectué, on continue par le test du contrôleur d'interruption :

```
                        781   ;-----------------------------------------------------
                        782   ;       8259 INTERRUPT CONTROLLER TEST                :
                        783   ; DESCRIPTION                                         :
                        784   ;       READ/WRITE THE INTERRUPT MASK REGISTER (IMR)  :
                        785   ;       WITH ALL ONES AND ZEROES. ENABLE SYSTEM       :
                        786   ;       INTERRUPTS. MASK DEVICE INTERRUPTS OFF. CHECK :
                        787   ;       FOR HOT INTERRUPTS (UNEXPECTED).              :
                        788   ;-----------------------------------------------------
                        789           ASSUME   DS:ABS0
E329 1F                 790   C21:    POP      DS
                        791
                        792   ;--- TEST THE IMR REGISTER
                        793
E32A C6D6150405         794   C21A:   MOV      DATA_AREA[OFFSET MFR_ERR_FLAG],05H
                        795                                               ; <><><><><><><><><><><><>
                        796                                               ; <><><>CHECKPOINT 5<><><>
E32F BD00               797           MOV      AL,0                       ; SET IMR TO ZERO
E331 E621               798           OUT      INTA01,AL
E333 E421               799           IN       AL,INTA01                  ; READ IMR
E335 0AC0               800           OR       AL,AL                      ; IMR = 0?
E337 751B               801           JNZ      D6                         ; GO TO ERR ROUTINE IF NOT 0
E339 B0FF               802           MOV      AL,0FFH                    ; DISABLE DEVICE INTERRUPTS
E33B E621               803           OUT      INTA01,AL                  ; WRITE TO IMR
E33D E421               804           IN       AL,INTA01                  ; READ IMR
E33F 0401               805           ADD      AL,1                       ; ALL IMR BIT ON?
E341 7511               806           JNZ      D6                         ; NO - GO TO ERR ROUTINE
                        807
                        808   ;--- CHECK FOR HOT INTERRUPTS
                        809
                        810   ;--- INTERRUPTS ARE MASKED OFF. CHECK THAT NO INTERRUPTS OCCUR.
                        811
E343 A26BD4             812           MOV      DATA_AREA[OFFSET INTR_FLAG],AL  ; CLEAR INTERRUPT FLAG
E346 FB                 813           STI                                 ; ENABLE EXTERNAL INTERRUPTS
E347 2BC9               814           SUB      CX,CX                      ; WAIT 1 SEC FOR ANY INTRS THAT
E349                    815   D4:
E349 E2F2               816           LOOP     D4                         ; MIGHT OCCUR
E34B                    817   D5:
E34B E2F2               818           LOOP     D5
E34D 803E680400         819           CMP      DATA_AREA[OFFSET INTR_FLAG],00H ; DID ANY INTERRUPTS OCCUR?
E352 7409               820           JZ       D7                         ; NO - GO TO NEXT TEST
E354                    821   D6:
E354 BEFFF890           822           MOV      SI,OFFSET E0               ; DISPLAY 101 ERROR
E358 EB4E16             823           CALL     E_MSG
E35B FA                 824           CLI
E35C F4                 825           HLT                                 ; HALT THE SYSTEM
```

Commentaires.- 1^o) On se replace dans le segment absolu pour le segment des données (ligne 789) et on restaure la valeur de DS (ligne 790). On teste le registre IMR du contrôleur d'interruption 8259. Pour cela, on place 5h dans la variable MFR_ERR_FLAG de la zone de communication du BIOS (ligne 794), on initialise IMR à zéro (lignes 797 et 798) et on le lit (lignes 799 et 800). Si on ne retrouve pas zéro (lignes 801 et 802), on affiche l'erreur 101 et on arrête le système (lignes 821 à 825).

L'étiquette E0 est définie ligne 5250 :

```
                        5247   ;---------------------------------------------
                        5248   ;       MESSAGE AREA FOR POST                  :
                        5249   ;---------------------------------------------
F8FF 313031             5250   E0      DB       '101',13,10         ; SYSTEM BOARD ERROR
F902 0D
F903 0A
F904 20323031           5251   E1      DB       ' 201',13,10        ; MEMORY ERROR
F908 0D
F909 0A
F90A 524F40             5252   F3A     DB       'ROM',13,10         ; ROM CHECKSUM ERROR
F90D 0D
F90E 0A
F90F 31383031           5253   F3C     DB       '1801',13,10        ; EXPANSION IO BOX ERROR
F913 0D
F914 0A
F915 50415249545920     5254   D1      DB       'PARITY CHECK 2',13,10
     434845434B2032
F923 0D
F924 0A
F925 50415249545920     5255   D2      DB       'PARITY CHECK 1',13,10
     434845434B2031
F933 0D
F934 0A
F935 3F3F3F3F3F         5256   D2A     DB       '?????',13,10
F93A 0D
F93B 0A
                        5257
```

Si on retrouve zéro, on désactive les interruptions matérielles (lignes 802 et 803), on lit IMR (ligne 804) dans lequel tous les bits devraient valoir 1, on vérifie et on termine comme dans le premier test si ce n'est pas le cas (lignes 805 et 806).

- 2^o) Les interruptions matérielles sont maintenant masquées, aucune ne devrait donc émettre de requête. On va vérifier qu'il en est bien ainsi. Pour cela, on remet à zéro la variable INTR_FLAG de la zone de communication du BIOS (ligne 812), on permet les interruptions masquables (ligne 813), on attend une seconde (lignes 814 à 816) et on regarde si une requête d'interruption est survenue (ligne 819). On passe au test suivant si ce n'est pas le cas (ligne 820) ou on affiche le message d'erreur et on s'arrête si c'est le cas.

11.2.16 Vérification du temporisateur

La vérification du temporisateur, objet des lignes 826 à 869, a été étudiée au chapitre 8.

11.2.17 Test du clavier

Après la vérification du temporisateur, on passe au test du clavier :

```
                        870  ;------------------------------------------------------
                        871  ;       KEYBOARD TEST                                  :
                        872  ; DESCRIPTION                                          :
                        873  ;       RESET THE KEYBOARD AND CHECK THAT SCAN         :
                        874  ;       CODE AA' IS RETURNED TO THE CPU.               :
                        875  ;       CHECK FOR STUCK KEYS.                          :
                        876  ;------------------------------------------------------
E3A2                    877  TST12:
E3A2 B099               878          MOV     AL,99H                  ; SET 8255 MODE A,C=IN B=OUT
E3A4 E663               879          OUT     CMD_PORT,AL
E3A6 A01004             880          MOV     AL,DATA_AREA[OFFSET EQUIP_FLAG]
E3A9 2401               881          AND     AL,01                   ; TEST CHAMBER?
E3AB 7431               882          JZ      F7                      ; BYPASS IF SO
E3AD 803E1204D1         883          CMP     DATA_AREA[OFFSET MFG_TST],1   ; MANUFACTURING TEST MODE?
E3B2 742A               884          JE      F7                      ; YES - SKIP KEYBOARD TEST
E3B4 E87316             885          CALL    KBD_RESET               ; ISSUE RESET TO KEYBRD
E3B7 E31E               886          JCXZ    F6                      ; PRINT ERR MSG IF NO INTERRUPT
E3B9 B049               887          MOV     AL,49H                  ; ENABLE KEYBOARD
E3BB E661               888          OUT     PORT_B,AL
E3BD 80FBAA             889          CMP     BL,0AAH                 ; SCAN CODE AS EXPECTED?
E3C0 7515               890          JNE     F6                      ; NO - DISPLAY ERROR MSG
                        891
                        892  ;-----  CHECK FOR STUCK KEYS
                        893
E3C2 B0C8               894          MOV     AL,0C8H                 ; CLR KBD, SET CLK LINE HIGH
E3C4 E661               895          OUT     PORT_B,AL
E3C6 B048               896          MOV     AL,48H                  ; ENABLE KBD, CLK IN NEXT BYTE
E3C8 E661               897          OUT     PORT_B,AL
E3CA 2BC9               898          SUB     CX,CX
E3CC                    899  F5:                                     ; KBD_WAIT:
E3CC E2FE               900          LOOP    F5                      ; DELAY FOR A WHILE
E3CE E460               901          IN      AL,KBD_IN               ; CHECK FOR STUCK KEYS
E3D0 3C00               902          CMP     AL,0                    ; SCAN CODE = 0?
E3D2 7404               903          JE      F7                      ; YES - CONTINUE TESTING
E3D4 EBB415             904          CALL    XPC_BYTE                ; CONVERT AND PRINT
E3D7                    905  F6:
E3D7 BE4CEC90           906          MOV     SI,OFFSET F1            ; GET MSG ADDR
E3DB E8CB15             907          CALL    E_MSG                   ; PRINT MSG ON SCREEN
```

Commentaires.- 1^o) On envoie l'octet 99h au port de commande du PIP (lignes 878 et 879). Puisque 99h = 1001 1001b, on a 1b pour mode de définition, 00b pour mode 0 du port A, 1b pour port A entrant, 1b pour port CU entrant, 0b pour mode 0 pour le port B, 0b pour port B sortant et 1b pour port CL entrant.

- 2^o) On teste si la configuration dit mode normal ou non (lignes 880 et 881). Si on n'est pas en mode normal, on passe le test du clavier et on va à la suite (ligne 884), c'est-à-dire l'affectation de la table des vecteurs d'interruptions matérielles.

- 3^o) Sinon on appelle la procédure KBD_RESET (ligne 885), définie ligne 5459 et déjà étudiée antérieurement. On affiche un message d'erreur si on ne reçoit pas une requête d'interruption (lignes 886 et 905 à 907).

L'étiquette F1 est définie ligne 2300 :

```
                        2299
EC4C 20333031           2300 F1      DB      ' 301',13,10           ; KEYBOARD ERROR
```

- 4^o) Sinon on active le clavier (lignes 887 et 888). Si le clavier ne renvoie pas, après l'appel à KBD_RESET, le code de recherche 'AA', on affiche un message d'erreur (lignes 889 et 890).

- 5^o) On teste ensuite s'il y a des touches coincées (*stuck keys*). Pour cela on désactive le clavier [en envoyant l'octet C8h au port B du PPI (lignes 894 et 895) : on a C8h = 1100 1000b, 1b pour lecture du clavier, 1b pour activation de l'horloge du clavier, 0b pour non renvoi à NMI en cas d'erreur sur une carte d'extension, 0b pour non vérification de la parité de la mémoire vive, 1b pour lecture des bits 0 à 3 des commutateurs DIP, les deux derniers 0b pour non fourniture des signaux aux temporisateurs] puis on l'active (lignes 896 et 897). On attend un petit peu (lignes 898 à 900) et on lit le code de recherche (ligne 901).

Si celui-ci est 0, on continue (lignes 902 et 903). Sinon on affiche celui-ci (ligne 904), pour indiquer quelle est la touche coincée et on affiche le message d'erreur.

11.2.18 Affectation de la table des vecteurs d'interruptions matérielles

Après le test du clavier, on passe à l'affectation définitive de la table des vecteurs d'interruptions matérielles :

```
                         908   ;------------------------------------------------------
                         909   ;       SETUP HARDWARE INT. VECTOR TABLE              :
                         910   ;------------------------------------------------------
E3DE                     911   F7:
E3DE 1E                  912           PUSH     DS                             ; SETUP_INT_TABLE
E3DF 2BC0                913           SUB      AX,AX
E3E1 8EC0                914           MOV      ES,AX
E3E3 B90800              915           MOV      CX,08                          ; GET VECTOR CNT
E3E6 0E                  916           PUSH     CS                             ; SETUP DS SEG REG
E3E7 1F                  917           POP      DS
E3E8 BEF3FE90            918           MOV      SI,OFFSET VECTOR_TABLE
E3EC BF2000              919           MOV      DI,OFFSET INT_PTR
E3EF                     920   F7A:
E3EF A5                  921           MOVSW
E3F0 47                  922           INC      DI                              ; SKIP OVER SEGMENT
E3F1 47                  923           INC      DI
E3F2 E2FB                924           LOOP     F7A
E3F4 1F                  925           POP      DS
                         926
                         927   ;-----  SET UP OTHER INTERRUPTS AS NECESSARY
                         928
E3F5 C7060B005FF8        929           MOV      NMI_PTR,OFFSET NMI_INT         ; NMI INTERRUPT
E3FB C706140054FF        930           MOV      INT5_PTR,OFFSET PRINT_SCREEN   ; PRINT SCREEN
E401 C706620000FF        931           MOV      BASIC_PTR+2,0F600H             ; SEGMENT FOR CASSETTE BASIC
                         932
                         933   ;-----  SETUP TIMER 0 TO BLINK LED IF MANUFACTURING TEST MODE
                         934
E407 803E120401          935           CMP      DATA_AREA[OFFSET MFG_TST],01H  ; MFG. TEST MODE?
E40C 750A                936           JNZ      EXP_IO
E40E C70670003CF9        937           MOV      WORD PRT(1CH*4),OFFSET BLINK_INT; SETUP TIMER INTR TO BLINK LED
E414 B0FE                938           MOV      AL,0FEH                         ; ENABLE TIMER INTERRUPT
E416 E621                939           OUT      INTA01,AL
```

Commentaires.- 1^o) On sauvegarde l'adresse du segment des données (ligne 912), on initialise le registre ES à zéro (lignes 913 et 914), on initialise le registre de compteur à 8 (ligne 915), le nombre de vecteurs concernés, on initialise l'adresse du segment des données avec celle du segment de code (lignes 916 et 917), on initialise SI (ligne 918) avec le décalage de la table VECTOR_TABLE (définie ligne 5746 et déjà étudiée), DI (ligne 919) avec le décalage de INT_PTR (défini ligne 47 comme emplacement mémoire 20). On place les huit vecteurs (lignes 920 à 924) puis on revient au segment des données originel (ligne 925).

- 2^o) On place les trois vecteurs pour NMI, l'impression d'écran et le lecteur de cassette à la main (lignes 929 à 931).

- 3^o) On active le temporisateur 0 pour faire clignoter les LEDs si on se trouve en mode de test (lignes 933 à 939).

11.2.19 Test des cartes d'extension

Après l'affectation définitive de la table des vecteurs d'interruptions matérielles, on passe au test des cartes d'extension :

```
                  940  ;---------------------------------------------------------------
                  941  ; EXPANSION I/O BOX TEST                                        :
                  942  ;       CHECK TO SEE IF EXPANSION BOX PRESENT - IF INSTALLED,  :
                  943  ;       TEST DATA AND ADDRESS BUSES TO I/O BOX                 :
                  944  ; ERROR='1801'                                                  :
                  945  ;---------------------------------------------------------------
                  946
                  947  ;----- DETERMINE IF BOX IS PRESENT
                  948
E418              949  EXP_IO:                                  ; (CARD WAS ENABLED EARLIER)
E418 BA1002       950          MOV     DX,0210H                 ; CONTROL PORT ADDRESS
E41B B85555       951          MOV     AX,5555H                 ; SET DATA PATTERN
E41E EE           952          OUT     DX,AL
E41F B001         953          MOV     AL,01H                   ; MAKE AL DIFFERENT
E421 EC           954          IN      AL,DX                    ; RECOVER DATA
E422 3AC4         955          CMP     AL,AH                    ; REPLY?
E424 7544         956          JNE     E19                      ; NO RESPONSE, GO TO NEXT TEST
E426 F700         957          NOT     AX                       ; MAKE DATA=AAAA
E428 EE           958          OUT     DX,AL
E429 B001         959          MOV     AL,01H
E42B EC           960          IN      AL,DX                    ; RECOVER DATA
E42C 3AC4         961          CMP     AL,AH
E42E 753A         962          JNE     E19
                  963
                  964  ;-----  CHECK ADDRESS BUS
                  965
E430              966  EXP2:
E430 B80100       967          MOV     BX,0001H
E433 BA1502       968          MOV     DX,0215H                 ; LOAD HI ADDR. REG ADDRESS
E436 B91000       969          MOV     CX,0016                  ; GO ACROSS 16 BITS
E439              970  EXP3:
E439 2E8807       971          MOV     CS:[BX],AL               ; WRITE ADDRESS F0000+BX
E43C 90           972          NOP
E43D EC           973          IN      AL,DX                    ; READ ADDR. HIGH
E43E 3AC7         974          CMP     AL,BH
E440 7521         975          JNE     EXP_ERR                  ; GO ERROR IF MISCOMPARE
E442 42           976          INC     DX                       ; DX=216H (ADDR. LOW REG)
E443 EC           977          IN      AL,DX
E444 3AC3         978          CMP     AL,BL                    ; COMPARE TO LOW ADDRESS
E446 751B         979          JNE     EXP_ERR
E448 4A           980          DEC     DX                       ; DX BACK TO 215H
E449 D1E3         981          SHL     BX,1
E44B E2EC         982          LOOP    EXP3                     ; LOOP TILL '1' WALKS ACROSS BX
                  983
                  984  ;----- CHECK DATA BUS
                  985
E44D B90800       986          MOV     CX,0008                  ; DO 8 TIMES
E450 B001         987          MOV     AL,01
E452 4A           988          DEC     DX                       ; MAKE DX=214H (DATA BUS REG)
E453              989  EXP4:
E453 8AE0         990          MOV     AH,AL                    ; SAVE DATA BUS VALUE
E455 EE           991          OUT     DX,AL                    ; SEND VALUE TO REG
E456 B001         992          MOV     AL,01H
E458 EC           993          IN      AL,DX                    ; RETRIEVE VALUE FROM REG
E459 3AC4         994          CMP     AL,AH                    ; = TO SAVED VALUE
E45B 7506         995          JNE     SHORT EXP_ERR
E45D D0E0         996          SHL     AL,1                     ; FORM NEW DATA PATTERN
E45F E2F2         997          LOOP    EXP4                     ; LOOP TILL BIT WALKS ACROSS AL
E461 EB07         998          JMP     SHORT E19                ; GO ON TO NEXT TEST
E463              999  EXP_ERR:
E463 BE0FF990    1000          MOV     SI,OFFSET F3C
E467 E83F15      1001          CALL    E_MSG
```

Commentaires.- 1^o) On envoie 55h au premier emplacement de carte d'extension (lignes 950 à 952). On change la valeur de AL (ligne 953) et on essaie de récupérer quelque chose sur la carte (ligne 954). S'il n'y a pas de réponse (ligne 955), on passe au test suivant (ligne 956), c'est-à-dire celui de la mémoire additionnelle.

- 2^o) On vérifie qu'il ne s'agit pas d'une réponse due au hasard. Pour cela on envoie AAh (ligne 957) à l'emplacement de première carte d'extension (ligne 958), on change la valeur de AL (ligne 959) et on essaie de récupérer quelque chose sur la carte (ligne 960). Si la réponse n'est pas ce qui a été envoyé (ligne 961), on passe également au test suivant (ligne 962).

- 3^o) On vérifie ensuite le bus des adresses. Pour cela, on initialise BX à 1 (ligne 967), DX à l'adresse du registre de la partie haute des adresses (ligne 968), le registre de compteur à 16 (ligne 969). On écrit AAh à l'adresse F00000h + [BX] (ligne 970), on attend un peu (ligne 971) et on lit la partie haute de l'adresse (ligne 973). Si on ne retrouve pas ce qu'on y a mis (ligne 974), on affiche l'erreur « 1801 » (lignes 979 et 999 à 1001).

Rappelons que F3C a été défini ligne 5253.

- 4^o) Sinon on teste de même la partie basse des adresses (lignes 976 à 979). Puis on teste de même les 15 autres bits (lignes 980 à 982).

- 5^o) On vérifie ensuite le bus des données. Pour cela, on initialise le registre de compteur à 8 (ligne 986), on initialise AL à 1 (ligne 987), le registre DX à l'adresse du registre du bus des données (ligne 988), on sauvegarde la valeur du bus des données dans le registre AH (ligne 990) pour pouvoir effectuer la comparaison, on envoie la valeur au registre (ligne 991), on change la valeur de AL (ligne 992), on récupère la valeur envoyée (ligne 993) et on compare (ligne 994). On affiche un message d'erreur si les valeurs ne sont pas les mêmes (ligne 995). On recommence pour les sept autres bits (lignes 996 et 997) et on passe au test suivant (ligne 998).

11.2.20 Test de la mémoire vive additionnelle

Après le test des cartes d'extension, on passe à la vérification de la mémoire située au-delà des 16 premiers KiO :

```
                        ;--------------------------------------------------------------
                        ;       ADDITIONAL READ/WRITE STORAGE TEST                    :
                        ; DESCRIPTION                                                 :
                        ;       WRITE/READ DATA PATTERNS TO ANY READ/WRITE            :
                        ;       STORAGE AFTER THE FIRST 32K. STORAGE                  :
                        ;       ADDRESSABILITY IS CHECKED.                            :
                        ;--------------------------------------------------------------
                                ASSUME  DS:DATA
E46A                    E19:
E46A E8EC15                     CALL    DDS
E46D 1E                         PUSH    DS
E46E                    E20:
E46E 813E72003412               CMP     RESET_FLAG,1234H      ; WARM START?
E474 7503                       JNE     E20A                  ; CONTINUE TEST IF NOT
E476 E99F00                     JMP     ROM_SCAN              ; GO TO NEXT ROUTINE IF SO
E479                    E20A:
E479 B81000                     MOV     AX,16                 ; STARTING AMT. OF MEMORY OK
E47C EB28                       JMP     SHORT PRT_SIZ         ; POST MESSAGE
E47E                    E20B:
E47E 8B1E0300                   MOV     BX,MEMORY_SIZE        ; GET MEM. SIZE WORD
E482 83EB10                     SUB     BX,16                 ; 1ST 16K ALREADY DONE
E485 B104                       MOV     CL,04H
E487 D3EB                       SHR     BX,CL                 ; DIVIDE BY 16
E489 8BCB                       MOV     CX,BX                 ; SAVE COUNT OF 16K BLOCKS
E48B 880004                     MOV     BX,0400H              ; SET PTR. TO RAM SEGMENT>16K
E48E                    E21:
E48E 8EDB                       MOV     DS,BX                 ; SET SEG. REG
E490 8EC3                       MOV     ES,BX
E492 81C30004                   ADD     BX,0400H              ; POINT TO NEXT 16K
E496 52                         PUSH    DX
E497 51                         PUSH    CX                    ; SAVE WORK REGS
E498 53                         PUSH    BX
E499 50                         PUSH    AX
E49A B90020                     MOV     CX,2000               ; SET COUNT FOR 8K WORDS
E49D E8CF01                     CALL    STGTST_CNT
E4A0 754C                       JNZ     E21A                  ; GO PRINT ERROR
E4A2 58                         POP     AX                    ; RECOVER TESTED MEM NUMBER
E4A3 051000                     ADD     AX,16
E4A6                    PRT_SIZ:
E4A6 50                         PUSH    AX
E4A7 BB0A00                     MOV     BX,10                 ; SET UP FOR DECIMAL CONVERT
E4AA B90300                     MOV     CX,3                  ; OF 3 NIBBLES
E4AD                    DECIMAL_LOOP:
E4AD 3302                       XOR     DX,DX
E4AF F7F3                       DIV     BX                    ; DIVIDE BY 10
E4B1 80CA30                     OR      DL,30H                ; MAKE INTO ASCII
E4B4 52                         PUSH    DX                    ; SAVE
E4B5 E2F6                       LOOP    DECIMAL_LOOP
E4B7 B90300                     MOV     CX,3
E4BA                    PRT_DEC_LOOP
E4BA 58                         POP     AX                    ; RECOVER A NUMBER
E4BB E80E14                     CALL    PRT_HEX
E4BE E2FA                       LOOP    PRT_DEC_LOOP
E4C0 B90700                     MOV     CX,7
E4C3 BE1AE0                     MOV     SI,OFFSET F3B         ; PRINT ' KB OK'
E4C6                    KB_LOOP:
E4C6 2E8A04                     MOV     AL,CS:[SI]
E4C9 46                         INC     SI
E4CA E8CF14                     CALL    PRT_HEX
E4CD E2F7                       LOOP    KB_LOOP
E4CF 58                         POP     AX                    ; RECOVER WORK REGS
E4D0 3D1000                     CMP     AX,16                 ; FIRST PASS?
E4D3 74A9                       JE      E20B
E4D5 5B                         POP     BX
E4D6 59                         POP     CX
E4D7 5A                         POP     DX
E4D8 E2B4                       LOOP    E21                   ; LOOP TILL ALL MEM. CHECKED
E4DA B00A                       MOV     AL,10
E4DC E88014                     CALL    PRT_HEX               ; LINE FEED
```

```
                        1071
                        1072   ;----- DMA TC0 SHOULD BE ON BY NOW - SEE IF IT IS
                        1073
E4DF E408               1074           IN      AL,DMA+08H
E4E1 2401               1075           AND     AL,00000001B          ; TC0 STATUS BIT ON?
E4E3 7533               1076           JNZ     ROM_SCAN              ; GO ON WITH NEXT TEST IF OK
E4E5 1F                 1077           POP     DS
E4E6 C606150003         1078           MOV     MFG_ERR_FLAG,03H      ; <><><><><><><><><><><><>
E4EB E966FE             1079           JMP     06                    ; POST 101 ERROR MSG AND HALT
                        1080
                        1081   ;----- PRINT FAILING ADDRESS AND XOR'ED PATTERN IF DATA COMPARE ERROR
                        1082
E4EE 8AE8               1083   E21A:   MOV     CH,AL                 ; SAVE FAILING BIT PATTERN
E4F0 B00D               1084           MOV     AL,13                 ; CARRAGE RETURN
E4F2 E8A714             1085           CALL    PRT_HEX
E4F5 B00A               1086           MOV     AL,10
E4F7 E8A214             1087           CALL    PRT_HEX
E4FA 58                 1088           POP     AX                    ; RECOVER AMT. OF GOOD MEM.
E4FB 83C406             1089           ADD     SP,6                  ; BALANCE STACK
E4FE 8CDA               1090           MOV     DX,DS                 ; GET FAILING SEGMENT
E500 1F                 1091           POP     DS
E501 1E                 1092           PUSH    DS
E502 A31300             1093           MOV     MEMORY_SIZE,AX        ; LOAD MEM. SIZE WORD TO SHOW
                        1094                                         ; HOW MUCH MEM. WORKING
E505 88361500           1095           MOV     MFG_ERR_FLAG,0H       ; <><><><><><><><><><><><>
                        1096                                         ; <><>CHECKPOINTS 08->A0<><>
E509 E8CE1A             1097           CALL    PRT_SEG               ; PRINT IT
E50C 8AC5               1098           MOV     AL,CH                 ; GET FAILING BIT PATTERN
E50E E87A14             1099           CALL    XPC_BYTE              ; CONVERT AND PRINT CODE
E511 BE04F990           1100           MOV     SI,OFFSET E1          ; SETUP ADDRESS OF ERROR MSG
E515 E89114             1101           CALL    E_MSG                 ; PRINT ERROR MSG
```

Commentaires.- 1^o) On prend comme segment des données celui de la zone de communication du BIOS (lignes 1009 à 1012). S'il s'agit d'un démarrage à chaud, on passe ce test et on va directement au test suivant (lignes 1014 à 1016). Sinon on débute avec la quantité de mémoire déjà vérifiée que l'on affiche (lignes 1017 et 1018).

- 2^o) Le sous-programme PRT_SIZ consiste à afficher la quantité de mémoire passée en revue, quantité contenue dans le registre AX. On sauvegarde le contenu de AX sur la pile (ligne 1041), la base choisie pour l'affichage étant dix (ligne 1042), la quantité sera convertie en trois demi-octets (ligne 1043). On initialise DX à zéro (ligne 1045). On divise le contenu de AX par dix (ligne 1046), le quotient se retrouvant dans DX. Le chiffre des unités est converti en ASCII (ligne 1047) et on le place sur la pile (ligne 1048). On recommence deux fois pour obtenir le chiffre des dizaines et celui des centaines (ligne 1049). On récupère ensuite ces trois chiffres sur la pile et on les affiche, donc dans l'ordre (lignes 1050 à 1054). On affiche à la suite « KB OK » (lignes 1055 à 1061). On récupère le contenu de AX sur la pile (ligne 1062) et, s'il s'agit de la première passe (c'est-à-dire de la mémoire déjà vérifiée par ailleurs), on revient (lignes 1063 et 1064).

L'étiquette F3B est définie ligne 252 :

```
                        251
E01A 204B42204F4B       252    F3B     DB      ' KB OK',13,10        ; KB FOR MEMORY SIZE
E020 0D
                        253
```

- 3^o) On récupère la capacité de mémoire indiquée par les commutateurs DIP (ligne 1021). On lui soustrait 16 (ligne 1022) puisque les 16 premiers KiO ont déjà été vérifiés par ailleurs. On divise la quantité restante par quatre (lignes 1023 et 1024) pour obtenir le nombre de blocs de 16 KiO qu'il faut vérifier, quantité que l'on place dans le registre de compteur (ligne 1025). On initialise BX à l'adresse se trouvant juste après les 16 premiers KiO (ligne 1026) et on vérifie chaque bloc de 16 KiO. Pour cela, on initialise DS et ES avec le contenu de BX (lignes 1028 et 1029), on fait pointer BX sur le prochain bloc à vérifier (ligne 1030), on sauvegarde le contenu des registres de travail sur la pile (lignes 1031 à 1034), on initialise le compteur CX pour 8 K mots (ligne 1035) et on fait appel à la sous-routine STGTST_CNT (ligne 1036, définie ligne 1324 et déjà étudiée). S'il y a un problème, on l'affiche (ligne 1037 renvoyant aux lignes 1083 à 1101). Si le test a réussi, on récupère la quantité déjà testée (ligne 1038), on lui ajoute 16 (ligne 1039) et on affiche la quantité testée (lignes 1040 à 1061 déjà étudiées), on restaure les valeurs des registres de travail (lignes 1062 à 1067) et on recommence jusqu'à ce que toute la mémoire soit vérifiée (lignes 1068). À la fin, on passe à la ligne dans l'affichage (lignes 1069 et 1070).

- 4^o) On vérifie que le DMA est bien en fonctionnement (lignes 1074 et 1075). S'il en est bien ainsi, on passe au test suivant (ligne 1076). Sinon on affiche le message d'erreur « 101 » et on arrête le système (lignes 1077 à 1079).

11.2.21 Test de la mémoire ROM optionnelle

Après le test de la mémoire vive additionnelle, on passe à celui de la ROM optionnelle :

```
                    1102   ;--------------------------------------------------------------
                    1103   ; CHECK FOR OPTIONAL ROM FROM C8000->F4000 IN 2K BLOCKS        :
                    1104   ;       (A VALID MODULE HAS '55AA' IN THE FIRST 2 LOCATIONS,   :
                    1105   ;       LENGTH INDICATOR (LENGTH/512) IN THE 3D LOCATION AND   :
                    1106   ;       TEST/INIT. CODE STARTING IN THE 4TH LOCATION.)         :
                    1107   ;--------------------------------------------------------------
E518                1108   ROM_SCAN:
E518 BA00C8         1109           MOV     DX,0C80H              ; SET BEGINNING ADDRESS
E51B                1110   ROM_SCAN_1:
E51B 8EDA           1111           MOV     DS,DX
E51D 28D8           1112           SUB     BX,BX                 ; SET BX=0000
E51F 8807           1113           MOV     AX,[BX]               ; GET 1ST WORD FROM MODULE
E521 53             1114           PUSH    BX
E522 5B             1115           POP     BX                    ; BUS SETTLING
E523 3D55AA         1116           CMP     AX,0AA55H             ; = TO ID WORD?
E526 7506           1117           JNZ     NEXT_ROM              ; PROCEED TO NEXT ROM IF NOT
E528 E82814         1118           CALL    ROM_CHECK             ; GO CHECK OUT MODULE
E52B EB0590         1119           JMP     ARE_WE_DONE           ; CHECK FOR END OF ROM SPACE
E52E                1120   NEXT_ROM:
E52E 81C28000       1121           ADD     DX,0080H              ; POINT TO NEXT 2K ADDRESS
E532                1122   ARE_WE_DONE:
E532 81FA00F6       1123           CMP     DX,0F600H             ; AT F6000 YET?
E536 7CE3           1124           JL      ROM_SCAN_1            ; GO CHECK ANOTHER ADD. IF NOT
E538 EB0190         1125           JMP     BASE_ROM_CHK          ; GO CHECK BASIC ROM
                    1126   ;-------------------------------------------------------------------
                    1127   ; A CHECKSUM IS DONE FOR THE 4 ROS MODULES CONTAINING BASIC CODE    :
                    1128   ;-------------------------------------------------------------------
E53B                1129   BASE_ROM_CHK:
E53B B404           1130           MOV     AH,4                  ; NO. OF ROS MODULES TO CHECK
E53D                1131   E4:
E53D 2BDB           1132           SUB     BX,BX                 ; SETUP STARTING ROS ADDR
E53F 8EDA           1133           MOV     DS,DX
                    1134
E541 E8AE13         1135           CALL    ROS_CHECKSUM
E544 7403           1136           JE      E5                    ; CONTINUE IF OK
E546 E88201         1137           CALL    ROM_ERR               ; POST ERROR
E549                1138   E5:
E549 81C20002       1139           ADD     DX,0200H              ; POINT TO NEXT 8K MODULE
E54D FECC           1140           DEC     AH                    ; ANY MORE TO DO?
E54F 75EC           1141           JNZ     E4                    ; YES - CONTINUE
```

Commentaires.- 1^o) On prend comme segment des données celui commençant au début de la mémoire ROM additionnelle (lignes 1109 à 1112), on récupère le premier mot du module (ligne 1113), on attend un peu pour stabiliser le bus (lignes 1114 et 1115). Si ce premier mot n'est pas l'identificateur « AA55 », on passe au module suivant (lignes 1116 et 1117). Sinon on appelle la sous-routine ROM_CHECK (ligne 1118), définie ligne 5283 et déjà étudiée ci-dessus. On vérifie ensuite si on est parvenu à la fin de la ROM optionnelle (lignes 1119 et 1123). Si ce n'est pas le cas, on passe au module suivant de ROM additionnelle (ligne 1124).

- 2^o) Sinon on vérifie les quatre modules contenant l'interpréteur BASIC (ligne 1125), ce qui est effectué lignes 1129 à 1141. Rappelons que la sous-routine ROS_CHECKSUM, qui commence ligne 5236, a été étudiée au chapitre 3.

11.2.22 Test du lecteur de disquette

Après le test de la mémoire ROM optionnelle, on passe au test du lecteur de disquette :

```
                        1142   ;---------------------------------------------------------------
                        1143   ;       DISKETTE ATTACHMENT TEST                                :
                        1144   ; DESCRIPTION                                                   :
                        1145   ;       CHECK IF IPL DISKETTE DRIVE IS ATTACHED TO SYSTEM. IF   :
                        1146   ;       ATTACHED, VERIFY STATUS OF NEC FDC AFTER A RESET. ISSUE :
                        1147   ;       A RECAL AND SEEK CMD TO FDC AND CHECK STATUS. COMPLETE  :
                        1148   ;       SYSTEM INITIALIZATION THEN PASS CONTROL TO THE BOOT     :
                        1149   ;       LOADER PROGRAM.                                         :
                        1150   ;---------------------------------------------------------------
E551                    1151   F9:
E551 1F                 1152           POP     DS
E552 A01000             1153           MOV     AL,BYTE PTR EQUIP_FLAG ; DISKETTE PRESENT?
E555 2401               1154           AND     AL,01H                 ; NO - BYPASS DISKETTE TEST
E557 743E               1155           JZ      F15
E559                    1156   F10:
E559 E421               1157           IN      AL,INTA01
E55B 24BF               1158           AND     AL,0BFH                ; ENABLE DISKETTE INTERRUPTS
E55D E621               1159           OUT     INTA01,AL
E55F B400               1160           MOV     AH,0                   ; RESET NEC FDC
E561 8AD4               1161           MOV     DL,AH                  ; SET FOR DRIVE 0
E563 CD13               1162           INT     13H                    ; VERIFY STATUS AFTER RESET
E565 F6C4FF             1163           TEST    AH,0FFH                ; STATUS OK?
E568 7520               1164           JNZ     F13                    ; NO - FDC FAILED
                        1165
                        1166   ;----- TURN DRIVE 0 MOTOR ON
                        1167
E56A BAF203             1168           MOV     DX,03F2H               ; GET ADDR OF FDC CARD
E56D B01C               1169           MOV     AL,1CH                 ; TURN MOTOR ON, EN DMA/INT
E56F EE                 1170           OUT     DX,AL                  ; WRITE FDC CONTROL REG
E570 2BC9               1171           SUB     CX,CX
E572                    1172   F11:                                   ; MOTOR_WAIT:
E572 E2FE               1173           LOOP    F11                    ; WAIT FOR 1 SECOND
E574                    1174   F12:                                   ; MOTOR_WAIT1:
E574 E2FE               1175           LOOP    F12
E576 33D2               1176           XOR     DX,DX                  ; SELECT DRIVE 0
E578 B501               1177           MOV     CH,1                   ; SELECT TRACK 1
E57A 88163E00           1178           MOV     SEEK_STATUS,DL
E57E E8FC08             1179           CALL    SEEK                   ; RECALIBRATE DISKETTE
E581 7207               1180           JC      F13                    ; GO TO ERR SUBROUTINE IF ERR
E583 B522               1181           MOV     CH,34                  ; SELECT TRACK 34
E585 E8F508             1182           CALL    SEEK                   ; SEEK TO TRACK 34
E588 7307               1183           JNC     F14                    ; OK, TURN MOTOR OFF
E58A                    1184   F13:                                   ; DSK_ERR:
E58A BE52EC90           1185           MOV     SI,OFFSET F3           ; GET ADDR OF MSG
E58E E81814             1186           CALL    E_MSG                  ; GO PRINT ERROR MSG
                        1187
                        1188   ;----- TURN DRIVE 0 MOTOR OFF
                        1189
E591                    1190   F14:                                   ; DR0_OFF:
E591 B00C               1191           MOV     AL,0CH                 ; TURN DRIVE 0 MOTOR OFF
E593 BAF203             1192           MOV     DX,03F2H               ; FDC CTL ADDRESS
E596 EE                 1193           OUT     DX,AL
                        1194
```

Commentaires.- 1^o) On se replace dans le segment des données originel (ligne 1152) et on regarde si les commutateurs DIP spécifient qu'il existe un lecteur de disquette (lignes 1153 et 1154). Si ce n'est pas le cas, on passe à la suite du POST (ligne 1155).

- 2^o) Sinon on active les interruptions liées aux lecteurs de disquettes (lignes 1157 à 1159), on initialise le contrôleur des lecteurs de disquette pour le premier lecteur, c'est-à-dire celui de numéro 0 (lignes 1160 à 1162) et on vérifie son statut (ligne 1163). Si on n'a pas réussi, on affiche un message d'erreur (lignes 1164 et 1184 à 1186).

L'étiquette F3 est définie ligne 2301 :

```
EC52 363031            2301   F3      DB       '601',13,10        ; DISKETTE ERROR
EC55 0D
EC56 0A
                       2302
```

- 3^o) On met en marche le moteur du premier lecteur de disquette. Pour cela, on place l'adresse du port du contrôleur de lecteurs de disquette dans DX (ligne 1169), on spécifie la mise en marche du moteur et l'activation du DMA et l'interruption (lignes 1169 et 1170). On attend une seconde que le moteur se mette en marche (lignes 1172 à 1175). On sélectionne le lecteur 0 (ligne 1176) et la première piste (ligne 1177) ; on l'indique à la variable de la zone de communication du BIOS (ligne 1178 ; SEEK_STATUS est défini ligne 135) et on essaie de recalibrer la disquette (ligne 1179). On affiche un message d'erreur si on n'y parvient pas (ligne 1180).

- 4^o) Si on y est parvenu, on vérifie que cela n'était pas dû au hasard en esayant également avec la piste 34 (lignes 1181 et 1182). On affiche un message d'erreur si on n'y parvient pas. Sinon on arrête le moteur et on passe à la suite du POST (lignes 1183 et 1190 à 1193).

11.2.23 Initialisation du tampon du clavier

Après le test du lecteur de disquette, on initialise le tampon du clavier :

```
                       1195   ;----- SETUP PRINTER AND RS232 BASE ADDRESSES IF DEVICE ATTACHED
                       1196
E597                   1197   F15:
E597 C6066B0000        1198          MOV      INTR_FLAG,00H         ; SET STRAY INTERRUPT FLAG = 00
E59C BE1E00            1199          MOV      SI,OFFSET KB_BUFFER   ; SETUP KEYBOARD PARAMETERS
E59F 89361A00          1200          MOV      BUFFER_HEAD,SI
E5A3 89361C00          1201          MOV      BUFFER_TAIL,SI
E5A7 89368000          1202          MOV      BUFFER_START,SI
E5AB 83C620            1203          ADD      SI,32                 ; DEFAULT BUFFER OF 32 BYTES
E5AE 89368200          1204          MOV      BUFFER_END,SI
```

c'est-à-dire qu'on initialise les indicateurs d'interruption à zéro (ligne 1198), les index de début (lignes 1199 et 1200) et de fin du tampon du clavier à la valeur du début du clavier (ligne 2101), le début au début de la zone de mémoire correspondante (ligne 1202) et la fin du tampon 32 octets plus loin (lignes 1203 et 1204).

11.2.24 Activation de l'imprimante

Après l'initialisation du tampon du clavier, on active l'imprimante si elle est présente :

```
E5B2 BF7800                MOV   DX,OFFSET PRINT_TIM_OUT ; SET DEFAULT PRINTER TIMEOUT
E5B5 1E                    PUSH  DS
E5B6 07                    POP   ES
E5B7 881414                MOV   AX,1414H                ; DEFAULT=20
E5BA AB                    STOSW
E5BB AB                    STOSW
E5BC B80101                MOV   AX,0101H                ;RS232 DEFAULT=01
E5BF AB                    STOSW
E5C0 AB                    STOSW
E5C1 E421                  IN    AL,INTA01
E5C3 24FC                  AND   AL,0FCH                 ; ENABLE TIMER AND KB INTS
E5C5 E621                  OUT   INTA01,AL
E5C7 83FD00                CMP   BP,0000H                ; CHECK FOR BP= NON-ZERO
                                                         ; (ERROR HAPPENED)
E5CA 7419                  JE    F15A_0                  ; CONTINUE IF NO ERROR
E5CC BA0200                MOV   DX,2                    ; 2 SHORT BEEPS (ERROR)
E5CF E80614                CALL  ERR_BEEP
E5D2 BE09E890              MOV   SI,OFFSET F30           ; LOAD ERROR MSG
E5D6 E8F113                CALL  P_MSG
E5D9                 ERR_WAIT:
E5D9 B400                  MOV   AH,00
E5DB CD16                  INT   16H                     ; WAIT FOR 'F1' KEY
E5DD 80FC38                CMP   AH,38H
E5E0 75F7                  JNE   ERR_WAIT
E5E2 EB0E90                JMP   F15A                    ; BYPASS ERROR
E5E5                 F15A_0
E5E5 803E120001            CMP   MFG_TST,1               ; MFG MODE
E5EA 7406                  JE    F15A                    ; BYPASS BEEP
E5EC BA0100                MOV   DX,1                    ; 1 SHORT BEEP (NO ERRORS)
E5EF E8E613                CALL  ERR_BEEP
E5F2 A01000          F15A: MOV   AL,BYTE PTR EQUIP_FLAG  ; GET SWITCHES
E5F5 2401                  AND   AL,00000001B            ; 'LOOP POST' SWITCH ON
E5F7 7403                  JNZ   F15B                    ; CONTINUE WITH BRING-UP
E5F9 E95FFA                JMP   START
E5FC 2AE4            F15B: SUB   AH,AH
E5FE A04900                MOV   AL,CRT_MODE
E601 CD10                  INT   10H                     ; CLEAR SCREEN
E603                 F15C:
E603 BDA3F990              MOV   BP,OFFSET F4            ; PRT_SRC_TBL
E607 8E0000                MOV   SI,0
E60A                 F16:                                ; PRT_BASE:
E60A 2E885600              MOV   DX,CS:[BP]              ; GET PRINTER BASE ADDR
E60E B0AA                  MOV   AL,0AAH                 ; WRITE DATA TO PORT A
E610 EE                    OUT   DX,AL
E611 1E                    PUSH  DS                      ; BUS SETTLEING
E612 EC                    IN    AL,DX                   ; READ PORT A
E613 1F                    POP   DS
E614 3CAA                  CMP   AL,0AAH                 ; DATA PATTERN SAME
E616 7505                  JNE   F17                     ; NO - CHECK NEXT PRT CD
E618 895408                MOV   PRINTER_BASE[SI],DX     ; YES - STORE PRT BASE ADDR
E61B 46                    INC   SI                      ; INCREMENT TO NEXT WORD
E61C 46                    INC   SI
E61D                 F17:
E61D 45                    INC   BP                      ; POINT TO NEXT BASE ADDR
E61E 45                    INC   BP
E61F 81FDA9F9              CMP   BP,OFFSET F4E           ; ALL POSSIBLE ADDRS CHECKED?
E623 75E5                  JNE   F16                     ; PRT_BASE
E625 B80000                MOV   BX,0                    ; POINTER TO RS232 TABLE
E628 BAFA03                MOV   DX,3FAH                 ; CHECK IF RS232 CD 1 ATTCH?
E62B EC                    IN    AL,DX                   ; READ INTR IO REG
E62C A8F8                  TEST  AL,0F8H
E62E 7506                  JNZ   F18
E630 C707F803              MOV   RS232_BASE[BX],3F8H     ; SETUP RS232 CD #1 ADDR
E634 43                    INC   BX
E635 43                    INC   BX
E636                 F18:
E636 BAFA02                MOV   DX,2FAH                 ; CHECK IF RS232 CD 2 ATTCH
E639 EC                    IN    AL,DX                   ; READ INTERRUPT IO REG
E63A A8F8                  TEST  AL,0F8H
E63C 7506                  JNZ   F19                     ; BASE_END
```

```
E63E C707F802        1275        MOV     RS232_BASE[BX],2F8H     ; SETUP RS232 CD # 2
E642 43              1276        INC     BX
E643 43              1277        INC     BX
                     1278
                     1279  ;----- SET UP EQUIP FLAG TO INDICATE NUMBER OF PRINTERS AND RS232 CARDS
                     1280
E644                 1281  F19:                                ; BASE_END:
E644 88C6            1282        MOV     AX,SI                   ; SI HAS 2*NUMBER OF RS232
E646 B103            1283        MOV     CL,3                    ; SHIFT COUNT
E648 D2C8            1284        ROR     AL,CL                   ; ROTATE RIGHT 3 POSITIONS
E64A 0AC3            1285        OR      AL,BL                   ; OR IN THE PRINTER COUNT
E64C A21100          1286        MOV     BYTE PTR EQUIP_FLAG+1,AL        ; STORE AS SECOND BYTE
E64F BA0102          1287        MOV     DX,201H
E652 EC              1288        IN      AL,DX
E653 90              1289        NOP
E654 90              1290        NOP
E655 90              1291        NOP
E656 A80F            1292        TEST    AL,0FH
E658 7505            1293        JNZ     F20                     ; NO_GAME_CARD
E65A 8001110010      1294        OR      BYTE PTR EQUIP_FLAG+1,16
E65F                 1295  F20:
                     1296
                     1297  ;----- ENABLE NMI INTERRUPTS
                     1298
E65F E461            1299        IN      AL,PORT_B               ; RESET CHECK ENABLES
E661 0C30            1300        OR      AL,30H
E663 E661            1301        OUT     PORT_B,AL
E665 24CF            1302        AND     AL,0CFH
E667 E661            1303        OUT     PORT_B,AL
E669 8080            1304        MOV     AL,80H                  ; ENABLE NMI INTERRUPTS
E66B E6A0            1305        OUT     0A0H,AL
E66D                 1306  F21:                                ; LOAD_BOOT_STRAP:
E66D CD19            1307        INT     19H                     ; GO TO THE BOOT LOADER
                     1308
```

Commentaires.- 1^o) On récupère (ligne 1205) le décalage de la variable PRINT_TIM_OUT de la zone de communication du BIOS (définie ligne 208), on attend un peu (lignes 1206 et 1207) et on place dans cette variable deux fois la valeur 20 (lignes 1208 à 1210).

- 2^o) On place de même (lignes 1211 à 1213) la valeur par défaut 1 dans la variable RS232_TIM_OUT (définie ligne 209).

- 3^o) On active les interruptions du temporisateur et du clavier (lignes 1214 à 1216).

- 4^o) Si la valeur de BP est non nulle, ce n'est pas normal donc on émet deux bips courts (lignes 1220 et 1221) et on affiche un message d'erreur (lignes 1222 et 1223). On attend alors qu'on appuie sur la touche « F1 » (lignes 1224 à 1228) pour pouvoir continuer (ligne 1229).

L'étiquette F30 est définie ligne 1704 :

```
                     1703
E809 4552524F522E20  1704  F30   DB      'ERROR. (RESUME = F1 KEY)',13,10      ; ERROR PROMPT
     28524553554D45
     203D2022463122
     204B455929
E823 0D
E824 0A
                     1705
```

- 5^o) On regarde les indicateurs d'initialisation (ligne 1231 ; la variable MFG_TST de la zone de communication du BIOS est définie ligne 84). Si on se trouve dans le mode MFG, on passe le bip (ligne 1232). Sinon on émet un bip court pour indiquer qu'il n'y a pas d'erreur (lignes 1233 et 1234).

- 6^o) On regarde si on se trouve dans le mode « LOOP POST » (lignes 1235 et 1236). Si ce n'est pas le cas on continue les tests (ligne 1237), sinon on revient à partir du test du microprocesseur (ligne 1238 ; l'étiquette START est définie ligne 311).

- 7^o) On efface l'écran (lignes 1239 à 1241).

- 8^o) On détermine les ports parallèles. Pour cela, on initialise le registre BP avec le début de la table source de l'imprimante et SI à zéro (lignes 1243 et 1244). L'étiquette F4 est définie ligne 5337 :

```
                    5336
F9A3                5337   F4     LABEL    WORD          ; PRINTER SOURCE TABLE
F9A3 BC03           5338          DW       3BCH
F9A5 7803           5339          DW       378H
F9A7 7802           5340          DW       278H
F9A9                5341   F4E    LABEL    WORD
                    5342
```

On place l'adresse de base de l'imprimante dans le registre DX (ligne 1246) que l'on écrit sur le port A du PPI (lignes 1247 et 1248), on attend un peu (ligne 1249), on le récupère (ligne 1250), on attend un peu (ligne 1251) et on vérifie qu'on obtient bien la même chose (ligne 1252). Si ce n'est pas le cas, on passe à l'adresse suivante possible pour l'imprimante (ligne 1253). Si c'est le cas, on enregistre l'adresse (ligne 1254 ; la variable PRINTER_BASE de la zone de communication du BIOS est définie ligne 82) et on incrémente deux fois SI (lignes 1255 et 1256) pour le prochain enregistrement.

On pointe sur l'adresse suivante (lignes 1258 et 1259). Si on n'a pas déjà passé en revue toutes les adresses possibles (ligne 1260), on la teste (ligne 1261).

- 8^o) On détermine les ports série. Pour cela, on intialise BX avec le début de la table RS232 (ligne 1262), on regarde si une carte est présente sur le premier port possible (lignes 1263 à 1265). Si c'est le cas, on en enregistre l'adresse (ligne 1267 ; la variable RS232_BASE de la zone de communication du BIOS est définie ligne 81).

On recommence pour le deuxième port possible (lignes 1268 à 1277).

- 9^o) On spécifie le nombre de ports parallèles et de ports série dans la variable EQUIP_FLAG de la zone de communication du BIOS (définie ligne 83). Pour cela, puisque SI contient à ce moment deux fois le nombres de cartes série, on place son contenu dans le registre AX (ligne 1282), on effectue une rotation de trois positions à droite (lignes 1283 et 1284), on combine avec le nombre de cartes parllèles (ligne 1285) et on place le tout dans le second octet de la variable (ligne 1286).

- 10^o) On regarde s'il existe un port de jeu. Pour cela, on teste le port 201h (lignes 1287 à 1292). S'il est présent, on l'indique dans la variable EQUIP_FLAG (ligne 1294).

- 11^o) On active les interruptions non masquables. Pour cela, on récupère le masque des interruptions (ligne 1299), on y place les bits adéquats et on replace le masque modifié (lignes 1300 à 1303). On active (lignes 1304 et 1305).

- 12^o) On appelle l'interruption 19h de chargement du système d'exploitation (ligne 1307), que nous étudierons dans la section suivante. On ne reviendra jamais à l'appelant, il est donc inutile de faire suivre cette instruction d'une autre (par exemple l'arrêt du système).

11.3 L'interruption 19h : chargement du système d'exploitation

Principe.- Le BIOS permet, comme son nom l'indique, les entrées-sorties de base, en particulier la lecture au clavier, l'affichage à l'écran ainsi que la lecture et l'écriture sur disquette. Ceci est suffisant pour charger le système d'exploitation. Celui-ci se trouve, en plusieurs fichiers, sur une disquette. On peut donc le charger en utilisant les interruptions du BIOS.

En ce qui concerne le BIOS du PC/XT, lors de l'appel de l'interruption de chargement, le BIOS lit le premier secteur de la disquette du premier lecteur, en place le contenu à l'adresse `07C0:000` et effectue un saut à cette dernière adresse. Tout revient donc à exécuter ce qu'il y a sur les 510 octets suivants de ce premier secteur.

Ce code de moins de 510 octets n'est pas le système d'exploitation lui-même (il en occupe beaucoup plus), mais le **chargeur du système d'exploitation** (*Boot strap loader* en anglais).

Le code.- Le code BIOS de la routine de service de l'interruption 19h commence ligne 1420 :

```
                        1408  ;---INT 19-------------------------------------------------------
                        1409  ; BOOT STRAP LOADER                                             :
                        1410  ;       TRACK 0, SECTOR 1 IS READ INTO THE                      :
                        1411  ;       BOOT LOCATION (SEGMENT 0, OFFSET 7C00)                  :
                        1412  ;       AND CONTROL IS TRANSFERRED THERE.                       :
                        1413  ;                                                               :
                        1414  ;       IF THERE IS A HARDWARE ERROR CONTROL IS                 :
                        1415  ;       TRANSFERRED TO THE ROM BASIC ENTRY POINT.               :
                        1416  ;----------------------------------------------------------------
                        1417          ASSUME   CS:CODE,DS:ABS0
E6F2                    1418          ORG      0E6F2H
                        1419
E6F2                    1420  BOOT_STRAP       PROC    NEAR
E6F2 FB                 1421          STI                             ; ENABLE INTERRUPTS
E6F3 2BC0               1422          SUB      AX,AX                  ; ESTABLICH ADDRESSING
E6F5 8ED8               1423          MOV      DS,AX
                        1424
                        1425  ;----- RESET THE DISK PARAMETER TABLE VECTOR
                        1426
E6F7 C7067800C7EF       1427          MOV      WORD PTR DISK_POINTER, OFFSET DISK_BASE
E6FD 8C0E7A00           1428          MOV      WORD PTR DISK_POINTER+2,CS
                        1429
                        1430  ;----- LOAD SYSTEM FROM DISKETTE -- CX HAS RETRY COUNT
                        1431
E701 B90400             1432          MOV      CX,4                   ; SET RETRY COUNT
E704                    1433  H1:                                     ; IPL_SYSTEM
E704 51                 1434          PUSH     CX                     ; SAVE RETRY COUNT
E705 B400               1435          MOV      AH,0                   ; RESET THE DISKETTE SYSTEM
E707 CD13               1436          INT      13H                    ; DISKETTE_IO
E709 720F               1437          JC       H2                     ; IF ERROR, TRY AGAIN
E70B B80102             1438          MOV      AX,201H                ; READ IN THE SINGLE SECTOR
E70E 28D2               1439          SUB      DX,DX                  ; TO THE BOOT LOCATION
E710 8EC2               1440          MOV      ES,DX
E712 B8007C             1441          MOV      BX,OFFSET BOOT_LOCN
                        1442                                          ; DRIVE 0, HEAD 0
E715 B00100             1443          MOV      CX,1                   ; SECTOR 1, TRACK 0
E718 CD13               1444          INT      13H                    ; DISKETTE_IO
E71A                    1445  H2:
E71A 59                 1446          POP      CX                     ; RECOVER RETRY COUNT
E71B 7304               1447          JNC      H4                     ; CF SET BY UNSUCCESSFUL READ
E71D E2E5               1448          LOOP     H1                     ; DO IT FOR RETRY TIMES
                        1449
                        1450  ;----- UNABLE TO IPL FROM DISKETTE
                        1451
E71F                    1452  H3:
E71F CD18               1453          INT      18H                    ; GO TO RESIDENT BASIC
                        1454
                        1455  ;----- IPL WAS SUCCESSFUL
                        1456
E721                    1457  H4:
```

```
E721 EA007C0000         1458            JMP      BOOT_LOCN
                        1459   BOOT_STRAP        ENDP
                        1460
```

Commentaires.- 1^o) Le segment de code est celui du BIOS et le segment des données commence à zéro (ligne 1417). On place le code à partir de l'emplacement mémoire E6F2h (ligne 1418) pour qu'il corresponde au vecteur de cette interruption. On active les interruptions masquables (ligne 1421) et on initialise le segment DS à zéro (lignes 1422 et 1423).

- 2^o) La variable DISK_POINTER (définie ligne 55 comme se trouvant à l'emplacement mémoire 78h) est initialisée avec le décalage de la variable DISK_BASE (ligne 1427) et la variable suivante avec le contenu du segment de code (ligne 1428).

- 3^o) On charge le chargeur de système d'exploitation à partir des disquettes, en commençant par la première. Pour cela, on initialise le registre de compteur à quatre (ligne 1432), nombre maximum d'essais, on le sauvegarde sur la pile (ligne 1434), on initialise AH à zéro (ligne 1435) pour commencer par le premier lecteur de disquette et on fait appel à l'interruption des lecteurs de disquette (ligne 1436). On recommence (au plus trois fois) tant qu'il y a une erreur (ligne 1437) : on récupère le contenu de CX (ligne 1446) ; si on a réussi à lire on va à la fin (ligne 1447), sinon on recommence (ligne 1448).

Sinon on lit le premier secteur. Pour cela on initialise AX à 201H (ligne 1438), ES à zéro (lignes 1439 et 1440), spécifiant le segment à partir duquel on va copier, BX au décalage de l'endroit à partir duquel on va copier (ligne 1441), CX à un (ligne 1443), pour lecteur 0, tête 0, secteur 1 et piste 0, et on fait appel à l'interruption (ligne 1444).

On exécute ensuite le code chargé (ligne 1458).

- 4^o) Si on n'arrive pas à lire le premier secteur, on exécute l'interpréteur BASIC grâce à l'interruption 18h (ligne 1453). On ne reviendra jamais de cette routine de service, donc aucune instruction ne suit.

Rappelons qu'à la ligne 5765, le vecteur de cette interruption a été initialisé à F600h, c'est-à-dire au début du module ROM contenant l'interpréteur BASIC.

Disquette système.- Une disquette qui comprend un système d'exploitation s'appelle une **disquette système**. Sinon c'est un **disque non-système**.

Dans le cas d'une disquette système nous verrons ce qui se passe ensuite lorsque nous étudierons les systèmes d'exploitation. Dans le cas d'une disquette non-système, le **secteur de boot** est quand même chargé et exécuté, comme nous venons de le voir. On aurait pu faire la différence en utilisant un nombre magique, par exemple, mais ce n'est pas ce qui a été choisi par les concepteurs du PC.

Une conséquence de ce choix est que toute disquette, qu'elle soit système ou non, doit posséder un programme sur le **secteur de boot**, sinon on risque un crash du système. Un utilisateur qui essaie, par inadvertance, de démarrer avec une disquette non-système dans le lecteur, rçoit un message du type :

```
Non-System disk or disk error
Replace and press any key when ready
```

Ce message n'est pas dû au BIOS mais par une partie du secteur de démarrage placé sur chaque disquette formatée par le programme `FORMAT` du DOS.