tremplin micro

Déjà des routines pour le GS !

Le 65C816 décortiqué

Créez vos caractères graphiques !

Ecrivez en HGR !

Initiation au Basic

Votre Apple connaît la musique !

N° 12 - Bimestriel - Deuxième année
3 Janvier - 4 Mars 1987
254 FB - 11 FS - 33 F

AU SOMMAIRE

Avec la collaboration de : ARGOS, Marcel COTTINI, Michel DEVAUX, François GALLET, Madeleine HODÉ, Roland JOST, Yvan KOENIG, Jean PERROT, J.-P. PICHON, Clément RENARD. ■

tremplin micro

12

BIMESTRIEL

Le numéro : 33 F
Abonnement d'un an : 190 F (6 numéros)

Tous nos prix sont indiqués TTC.

EDITIONS JIBENA

Direction-Rédaction :
Editions JIBENA
Guy-HACHETTE

La Petite Motte — Senillé
86100 CHÂTELLERAULT.

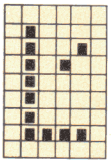
Téléphone :
49-93-66-66

PUBLICITÉ :
Raymond JULLIEN
(1) 45.75.41.81

Commission paritaire :
Les revues qui choisissent d'être réellement au service du Lecteur, en ne l'obligeant pas à glaner, dans plusieurs magazines, les renseignements concernant sa machine, ne bénéficient pas du numéro de Commission Paritaire, et pas davantage des tarifs postaux réduits.

TREMPLIN MICRO — Bimestriel — C'est une publication des Editions JIBENA, 4, rue de la Cour-des-Noues, 75020 PARIS — S.A. au capital de 3 600 000 F — Imprimé par CITÉ-PRESS/PARIS — Dépôt légal à la date de parution — Inscription à la Commission Paritaire des Publications et Agences de Presse : en cours — Directeur de la Publication : Guy-Clément COGNÉ — Diffusion N.M.P.P.

Où est donc passé le GS ?



APPLE *IIGS* existe. *Tremplin Micro* l'a même essayé ; mieux : nous avons testé, sur cette superbe édition de la formule II, tous les programmes du présent numéro. Parfaitement : la compatibilité a été vérifiée, en mode émulation bien sûr, et je vous assure que ça marche. Y compris l'HORLOGE de François Gallet, mais en tenant compte des restrictions indiquées par l'auteur... et en sachant que le *GS* n'a vraiment pas besoin de cette horloge d'appoint !

Quant à l'ORGUE de Roland Jost, il est superbe, mais en utilisant le tableau de bord du *GS* pour le mettre en vitesse lente... comme le *IIe* qu'il émule si bien, mais parfois avec trop de brio. Mais oui : il est réellement très agréable d'observer la rapidité de l'affichage et l'accélération dont bénéficient certaines routines de notre vieil Applesoft.

Je vous mets l'eau à la bouche, pas vrai ? Croyez bien que je le regrette. Si cet aveu peut vous consoler, sachez que je ne suis pas mieux loti que vous : j'ai définitivement restitué son *GS* au service de Presse d'Apple... et j'attends maintenant le mien. Je comprends la déception de certains lecteurs de *Tremplin Micro :* commander un ordinateur personnel est une chose ; pouvoir s'en servir quand bon vous semble en est une autre. Hélas ! dans le monde entier, les constructeurs de centaines de matériels font les mêmes annonces, puis ne tiennent pas leurs promesses. On ne compte plus les clients qui, en désespoir de cause, ont annulé une commande ici... pour en attendre une autre là !

Il faut reconnaître qu'Apple n'avait rien promis du tout. On savait depuis le premier jour que l'Apple *IIGS* ne serait vraiment disponible (c'est-à-dire livrable dans les 48 heures) qu'au début de 1987. Convient-il de regretter que l'annonce en ait été faite un peu trop tôt ? Hum ! qu'auraient dit les inconditionnels de la marque si on avait continué à leur cacher la sortie prochaine d'une machine que tout le monde attendait ?

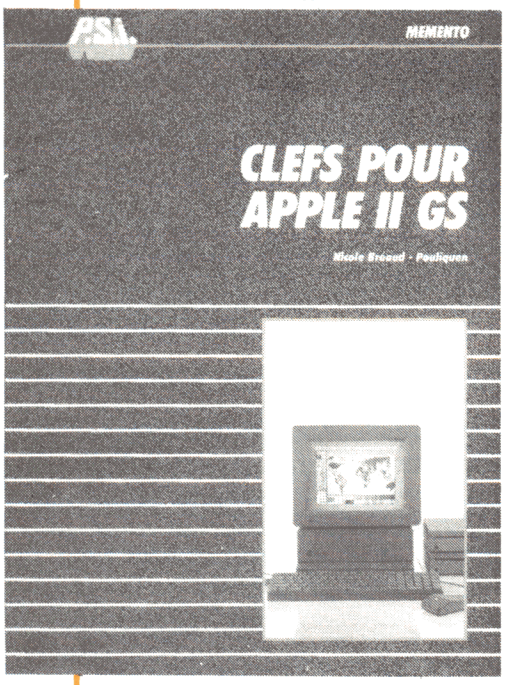
C'est pourtant ce qui se passe avec les imprimantes (Apple s'entend). Je vous assure que nous ne savons rien. Officiellement. Comme certains de nos lecteurs, nous avons appris, dans des magazines édités outre-Atlantique, que plusieurs imprimantes Apple — dont une laser "démocratique" — étaient sur le point d'être commercialisées. C'est récent... et c'est tout... du moins au moment où je rédige ces lignes, c'est-à-dire au début du mois de décembre.

En attendant d'en savoir davantage sur ces merveilles, souhaitons-nous une bonne... une excellente année 1987. Merci de votre attention et rendez-vous dans deux mois !

GUY-HACHETTE.

APPLE *IIGS* :

c'est parti !



N'enviez pas Nicole Bréaud-Pouliquen, auteur du premier MÉMENTO consacré à l'APPLE *IIGS* (au PSI) : exploiter une documentation, aussi complète soit-elle, et en extraire les adresses indispensables au programmeur amateur n'est pas de tout repos... surtout quand il n'existe aucun autre ouvrage permettant d'utiles comparaisons. Je vous invite à découvrir ces *CLEFS POUR APPLE IIGS*. Cette première édition comporte certes des lacunes, mais l'étude du *GS* n'en est qu'à ses débuts ! Que l'on ne s'y trompe pas : si cet ouvrage s'adresse d'abord et surtout à des programmeurs pratiquant l'assembleur (ou le langage C), il fournira d'indispensables informations à tous les autres.

On examinera avec curiosité le système CPW, avec son moniteur, son éditeur et son macro-assembleur, mais on s'attardera sur le répertoire détaillé des outils du bureau électronique (QuickDraw, WindowManager, MenuManager, etc.).

Nicole Bréaud-Pouliquen a même poussé la gentillesse jusqu'à montrer dans un programme d'application, comment ajouter un accessoire de bureau à une application. Ce n'est pas d'une simplicité évidente, mais je sais que bon nombre d'Applemaniaques se réjouissent déjà à la pensée de pouvoir prochainement dompter la bête ! **Guy-Hachette**

DÉPLACEMENT DE L'ÉCRAN.TEXTE

GS exclusivement

```
10  PRINT CHR$ (21): HOME
15  FOR I = 768 TO 793: READ R: POKE I,R: NEXT : POKE
    2039,32
20  FOR I = 24 TO 1 STEP - 1: HTAB I: VTAB I: PRINT I;: NEXT
30  PRINT : CALL 768
40  POKE 778,96: POKE 781,04
50  GOSUB 90: HOME : GOSUB 90: CALL 768
60  X = X + 1: IF X < 9 THEN 50
70  POKE 778,04: POKE 781,96
80  VTAB 20: PRINT "(M)ENU  ";: GET R$: PRINT R$: IF R$ =
    "M" THEN PRINT CHR$ (4)"RUN MENU.GS  "
85  HOME : END
90  VTAB 22: INVERSE : PRINT "TOUCHE SVP ";: NORMAL :
    GET R$: PRINT : RETURN
95  DATA 8,24,251,194,48,169,255,3,162,0,4,160,0,96,169,
    255,3,84,0,0,56,251,226,48,40,96
```

LA LIGNE 95

```
*300.319
300 :  08 18 FB C2 30 A9 FF 03
308 :  A2 00 04 A0 00 60 A9 FF
310 :  03 54 00 00 38 FB E2 30
318 :  28 60
```

LES COULEURS DE L'ÉCRAN.TEXTE

GS exclusivement

```
10  PRINT CHR$ (4)"PR£3": PRINT : HTAB 15: PRINT "  LES
    COULEURS DE L'ECRAN TEXTE DE VOTRE APPLE IIGS  "
20  A = PEEK (49186): B = PEEK (49204): REM Couleur
    actuelle (A = Fond / B = Cadre)
25  FOR I = 236 TO 251: POKE 49186,I: GOSUB 90: NEXT
30  CALL - 198
35  FOR J = 60 TO 75: POKE 49204,J: GOSUB 90: NEXT
40  POKE 49186,A: POKE 49204,B
50  VTAB 22: PRINT "(M)ENU ";: GET R$: IF R$ = "M" THEN
    PRINT CHR$ (4)"RUN MENU.GS"
80  END
90  POKE - 16368,0: WAIT - 16384,128,127: POKE -16368,0:
    RETURN
```

■

UTILITAIRE | DOS | ProDOS

Fonctionne aussi sur l'APPLE IIGS

Roland JOST

ECRITURE EN MODE

Haute Résolution étendue

En dehors de logiciels fermés tels que PURPLESOFT, ARLEQUIN ou EXTASIE, difficiles à modifier, il n'existe encore que peu d'utilitaires permettant de gérer la double haute résolution de l'Apple *IIe* 128 K ou de son cousin l'Apple *IIc*. Dans le n°10 de *Tremplin Micro*, Thierry Gauthier a présenté un programme d'inversion d'écran DHGR. Je vous propose maintenant un utilitaire permettant l'écriture en page DHGR.

ECRIT.DHGR est appelé par la fonction AMPERSAND et permet, à partir d'un programme Applesoft, d'afficher du texte en page haute résolution étendue. Les possibilités sont les suivantes :

- Alphabet français accentué, alphabet grec, indices et exposants (intéressant pour les programmes de chimie, physique ou maths). Tous les caractères peuvent être écrits en gras, en double hauteur, en inverse. Enfin l'écriture peut se faire horizontalement, vers le haut, vers le bas ou en oblique.

Routine LM et table de caractères

Tremplin Micro ne publie pas le programme source, mais celui-ci figure sur la disquette n°12. Pour taper *ECRIT.DHGR*, procédez donc de la manière habituelle : CALL -151, puis les différentes lignes de codes. Sauvez-le par :
BSAVE ECRIT.DHGR,A$4000,L465

Attaquez-vous ensuite à la table de caractères (ce n'est pas du gâteau, mais elle vous rendra de nombreux services !) :
4200 : 60 10 10 08 08 04 04 03...
et sauvez-la par :
BSAVE NEWSET2,A$4200,L2048

SIGNATURE (voir bulletin de commande) vous permettra un contrôle facile de la saisie de ces routines... et, d'une manière générale, de toutes celles à paraître dans *T.M.*

Comment utiliser ECRIT.DHGR ?

Un BLOAD ECRIT.DHGR vous permettra de l'installer en mémoire. Votre programme BASIC devra faire pointer la fonction Ampersand en $4000 (POKE 1013,76: POKE 1014,0: POKE 1015,64) et afficher la page DHGR.

Pour afficher une variable alphanumérique, il suffira de faire :

& PRINT "chaîne" AT X,Y ou
& PRINT A$ AT X,Y avec la position horizontale X (entre 0 et 79) et la position verticale Y (entre 0 et 183)

Pour afficher une variable numérique, il faudra la transformer en chaîne de caractères par la fonction STR$. Cela n'est pas un handicap et vous permettra d'ajouter un affichage formaté (variables chaînes).

Ecriture en double hauteur : faire SCALE = 2: Pour revenir en mode normal : SCALE = 0.

Ecriture dans différentes directions : la direction de l'affichage peut être modifiée par l'instruction ROT. *(suite page 4)*

ROT = 0 : affichage horizontal
ROT = 7 : en oblique à 45° vers le bas
ROT = 8 : verticalement, de haut en bas
ROT = 15 : en oblique vers le haut
ROT = 16 : verticalement, de bas en haut.

Les autres possibilités s'obtiennent en insérant des caractères de contrôle dans la chaîne :

CTRL-B : Le prochain caractère sera décalé vers le bas.
CTRL-E : Le prochain caractère sera mis en exposant.
CTRL-F : Le prochain caractère sera en grec minuscule.
CTRL-G : Le caractère qui suit sera en grec majuscule.
CTRL-I : Le caractère qui suit sera mis en indice.
CTRL-O : Le caractère qui suit sera décalé vers le haut.
CTRL-P : Tous les caractères qui suivent seront imprimés en gras jusqu'à la fin de la chaîne ou la rencontre du CTRL-N.
CTRL-Z : Tous les caractères qui suivent seront imprimés en INVERSE jusqu'à la fin de la chaîne ou la rencontre d'un CTRL-N.

CTRL-N : Affichage normal (ni gras ni inverse).

Le source Assembleur est bien documenté et peut se passer d'explications supplémentaires.

Le programme Applesoft DEMO.ECRIT.DHGR permet de se rendre compte des possibilités d'ECRIT.DHGR et de la façon de l'utiliser.

ATTENTION !

Les chaînes de caractères renferment des caractères de contrôle invisibles au listage et qui se traduisent à l'impression par un comportement inquiétant de l'Imprimante. Un truc pour le lister est de capturer le programme en fichier TEXTE, de le saisir dans un traitement de texte et d'éliminer ces caractères avant impression. Une autre solution est apportée par le programme **SIGNATURE** d'Yvan KOENIG.

DEMO.ECRIT.DHGR

```
10 REM TEST ECRIT.DHGR                                                    003A
11 :
20 D$ =  CHR$ (4): PRINT D$"BLOAD ECRIT.DHGR": PRINT D$"BLOAD NEWSET2,    0311
   A$4200"
22 POKE 1013,76: POKE 1014,0: POKE 1015,64: REM Revectorise l'ampersan    2C7C
   d                                                                      F9CB
30 ROT= 0: SCALE= 0                                                       003A
34 :
35 REM On positionne les 'switches' de passage en double haute résolut
   ion graphique                                                          003A
36 :                                                                      0C1E
40 POKE 49246,0: REM AN3 (Annunciator) off                                3219
42 POKE 49232,0: REM sélectionne le mode graphique (GR on)                0920
44 POKE 49239,0: REM selectionne la haute résolution (HIRES on)           151E
46 POKE 49165,0: REM sélectionne le mode 80 colonnes (80 COL on)          261B
48 POKE 49234,0: REM sélectionne le mode pleine page
50 POKE 49153,0: REM permet de stocker directement dans la page graphi    DF1B
   que auxiliaire (80STORE on)                                            091E
52 POKE 49156,0: REM Déconnecte RAMWRT                                    FC1C
54 POKE 49154,0: REM Déconnecte RAMRD                                     1A1D
56 POKE 49236,0: REM commute la mémoire principale                        003A
59 :
60 REM On efface la page double haute résolution                          003A
61 :                                                                      2391
65 HGR : REM Efface la page graphique principale                          381E
66 POKE 49237,0: REM sélectionne la page graphique auxiliaire             4F8D
67 CALL 62450: REM efface la page HGR                                     1A1D
68 POKE 49236,0: REM retour à la page HGR principale                      261B
69 POKE 49234,0: REM pleine page graphique

                                                                          003A
90 :
91 REM DEMONSTRATION
```

```
100 SCALE= 1: INVERSE : &  PRINT "ECRIT-DHGR" AT 15,1: NORMAL : REM Ce
    titre sera écrit en hauteur double et en noir sur fond blanc          5AB1
105 GOSUB 500                                                             1B45
110 A$ = "permet d'écrire des textes en mode Double Haute Résolution.":
    SCALE= 0: &  PRINT A$ AT 5,20                                         564F
115 GOSUB 500                                                             1B45
120 A$ = "EN MODE ctZINVERSEctN ou NORMAL": &  PRINT A$ AT 10,30: REM I
    l y a un CTRL-Z devant INVERSE et un CTRL-N après.                    3C1F
125 GOSUB 500                                                             1B45
127 A$ = "ctP en caractères GRAS :"ctGABCDEFGHIJKL..."                    8817
128 &  PRINT A$ AT 20,50: GOSUB 500                                       CA05
130 A$ = "VERTICALEMENT": ROT= 8: &  PRINT A$ AT 70,20: ROT= 16: &  PRI
    NT A$ AT 72,130                                                       E347
135 GOSUB 500                                                             1B45
140 A$ = "EN OBLIQUE ": ROT= 1: &  PRINT A$ AT 10,90: ROT= 9: &  PRINT
    A$ AT 10,80: ROT= 0                                                   84F2
145 GOSUB 500                                                             1B45
147 A$ = "en DOUBLE HAUTEUR": SCALE= 2: &  PRINT A$ AT 40,85: SCALE= 0    D3F7
150 A$ = "Formules mathématiques :""Y = axctE2 + bxctE3 + cxctE4": &  P
    RINT A$ AT 5,130: REM Il y a un CTRL-E avant les caractères à mettr
    e en exposant                                                         5292
155 GOSUB 500                                                             1B45
160 A$ = "Equations chimiques :"CHctI3CHctI2OH + CHctI3COOH --> CHctI3C
    OOCctI2HctI5 + HctI2O": REM il y a un CTRL-I devant les caractères
    à mettre en indice                                                    2AB4
165 &  PRINT A$ AT 5,150                                                  138A
166 GOSUB 500                                                             1B45
170 A$ = "Alphabet grec :"y = ctGSxctIiyctIi - ctFC(x,y)": &  PRINT A$
    AT 5,170: REM Il y a un CTRL-G devant S , un CTRL-I devant chaque i
     et un CTRL-F devant C                                                C7A4
190 &  PRINT "à vous d'utiliser toutes ces possibilités..." AT 15,180.    3CED
200 GET R$                                                                DB34
201 :                                                                     003A
202 REM on déconnecte la double haute résolution
203 :                                                                     003A
205 POKE 49247,0: REM ANONCIATEUR off                                     F91F
211 POKE 49164,0: REM CLR80VID : 80 COL off                               F61D
212 POKE 49233,0: REM TXTSET : MODE TEXTE                                 CD1A
213 POKE 49238,0: REM LORES :LO-RES off                                   2B1F
214 POKE 49152,0: REM CLR80COL : 80 STORE off                             B11A
215 POKE 49236,0: REM LOWSCR:PAGE 2 off                                   1A1D
220 END                                                                   0180
500 REM BOUCLE DE TEMPORISATION
510 FOR T = 1 TO 1000: NEXT : RETURN                                      0FFF
```

Si vous manquez de temps
(ou de courage !) pour recopier
les polices de caractères de notre
ami Roland JOST, offrez-vous
la disquette numéro 12 de
TREMPLIN MICRO !

ECRIT DHGR

```
4000: C9 BA F0 01 60 A2 08 86 1C A6 E7 F0 04 A2 10 86   81D9
4010: 1C A6 32 E0 3F D0 06 A2 FF 86 EF D0 06 A2 00 86   7BFD
4020: EF 86 FA 20 B1 00 C9 22 D0 06 20 81 DE 38 B0 07   AE6F
4030: 20 E3 DF 85 A0 84 A1 A0 00 B1 A0 85 FC C8 B1 A0   AFB7
4040: 85 FD C8 B1 A0 85 FE 20 B7 00 C9 C5 F0 01 60 20   4FF4
4050: F5 E6 86 EB 20 F5 E6 86 ED A0 00 84 A0 84 06 84   F08C
4060: 1D A0 42 84 CF A4 A0 B1 FD C9 1F 10 06 20 50 41   ECF3
4070: 4C CF 40 18 65 06 85 CE A6 EB 86 08 18 66 08 A9   C97F
4080: 00 69 00 F0 05 8D 54 C0 D0 03 8D 55 C0 18 A5 ED   E81E
4090: 65 1C 85 EE 20 D8 40 A5 F9 C9 08 D0 09 A5 ED 18   771E
40A0: 65 1C 85 ED D0 29 C9 10 D0 0A 38 A5 ED E5 1C 85   E5EF
40B0: ED 18 90 1B C9 08 30 0E 38 A5 ED E5 F9 18 69 08   2EF0
40C0: 85 ED E6 EB 90 09 A5 ED 18 65 F9 85 ED E6 EB E6   5C0D
40D0: A0 A4 A0 C4 FC D0 8E 60 20 1D 41 A9 00 85 09 A6   44BD
40E0: ED 20 3B 41 85 1A 84 1B E8 A4 09 B1 CE A4 FA C0   E339
40F0: 01 D0 05 85 FB 0A 05 FB 45 EF E6 09 A4 08 91 1A   80DA
4100: 85 A1 A5 E7 F0 0E 20 3B 41 85 1A 84 1B A5 A1 A4   D774
4110: 08 91 1A E8 E4 EE D0 C9 A9 42 85 CF 60 A5 CE A2   01BA
4120: 07 18 65 CE 90 02 E6 CF CA D0 F6 18 65 1D 90 02   DC55
4130: E6 CF 85 CE A9 00 85 06 85 1D 60 8A 4A 4A 4A A8   D84E
4140: B9 A1 41 48 8A 29 07 0A 0A 18 79 B9 41 A8 68 60   A0AC
4150: C9 02 D0 03 E6 ED 60 C9 05 D0 09 A9 80 85 06 A9   98D5
4160: 02 85 1D 60 C9 06 D0 05 A9 40 85 06 60 C9 07 D0   291C
4170: 05 A9 C0 85 06 60 C9 0F D0 03 C6 ED 60 C9 10 D0   27C0
4180: 05 A2 01 86 FA 60 C9 09 D0 05 A9 80 85 06 60 C9   480C
4190: 0E D0 07 A9 00 85 EF 85 FA 60 C9 1A A9 FF 85 EF   82E0
41A0: 60 00 80 00 80 00 80 00 80 28 A8 28 A8 28 A8 28   A1F8
41B0: A8 50 D0 50 D0 50 D0 50 D0 20 20 21 21 22 22 23   EC11
41C0: 23 20 20 21 21 22 22 23 23 20 20 21 21 22 22 23   FD18
41D0: 23                                                4623
```

Sauvez cette routine par un :
BSAVE ECRIT.DHGR, A$4000,L465

Le programme source (bien commenté par l'auteur) figure sur la disquette n°12 de **Tremplin Micro**.

ECRIT-DHGR

permet d'écrire des textes en mode Double Haute Résolution.
EN MODE INVERSE ou NORMAL

en caractères GRAS : ABCDEFGHIJKL...

EN OBLIQUE
EN OBLIQUE

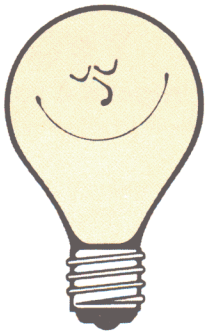
en DOUBLE HAUTEUR

Alphabet grec : y = Σxiyi - ∀(x,y)
À vous de jouer maintenant ...

Formules mathématiques : $Y = ax^2 + bx^3 + cx^4$

VERTICALEMENT INTERLIGNE

Yvan KOENIG



répond à vos questions...

• *M. D. a essayé d'utiliser PRO.TYPC sans lire l'article d'accompagnement. Ce n'est pas grave, mais c'est frustrant pour l'auteur à qui l'on pose alors des questions auxquelles il a déjà répondu.*

Oui, PRO.TYPC est relogeable. C'est d'ailleurs l'intérêt principal de ce programme qui montre comment mettre en place une routine entre ProDOS et ses buffers.

Par contre, MINIE que M. D. tente d'accoupler à PRO.TYPC (quel sera le fruit de cette union ?) ne dispose pas d'un relogeur, mais d'un transporteur. Claude AUBRY a préféré libérer de la place pour son programme - à une adresse donnée - et le copier à cette adresse sans modification. Je vous garantis que ce n'est pas par ignorance ! En fait, afin de permettre de modifier quelques paramètres depuis le BASIC, il était quasiment indispensable que l'implantation de MINIE fût fixe. Cela condamne simplement à mettre en place MINIE, puis PRO.TYPC, mais attention ! ne faites pas 36 BRUN PRO.TYPC consécutifs sinon le message IL N'Y A PLUS DE PLACE vous attend au coin... de l'écran !

Le petit programme PRO.FP (page 60 de T.M. n°11) permet de supprimer toutes les manipulations subies par ProDOS (déplacement des buffers, interruptions, reset...). Il permet d'éviter des déplacements de buffers abusifs. Attention ! PRO.FP supprime également MINIE. C'est en gros l'équivalent du FP sous DOS 3.3, mais SANS effacement du BASIC ce qui permet de l'exécuter DANS un programme.

• *M. D. demande aussi comment récupérer, sous ProDOS, des informations sauvées dans un fichier TXT et contenant des virgules.*

Il y a la bonne vieille méthode GET, à savoir :

1000 ONERR GOTO 1050 *(suite page 8)*

NEWSET2

Roland JOST

```
4200: 60 10 10 08 08 04 04 03 08 8C 14 92 3E A1 63 00  5117
4210: 1F 22 22 1E 22 22 1F 00 3F 22 02 02 02 02 83 00  09D0
4220: 08 8C 14 92 22 A1 7F 00 3F 22 02 0E 02 22 3F 00  0F50
4230: 1C 08 3E 6B 3E 08 1C 00 3F 22 02 02 02 02 83 00  BA1B
4240: B3 22 22 3E 22 22 B3 00 8C 08 08 08 08 08 8C 00  306C
4250: 0C 12 21 3F 21 12 0C 00 B3 12 0A 06 0A 12 B3 00  8461
4260: 08 8C 14 92 22 A1 63 00 B3 36 2A 2A 22 22 B3 00  FC94
4270: B3 22 26 2A 32 22 B3 00 1C 22 A1 A1 A1 22 1C 00  E18B
4280: BF 22 22 22 22 22 B3 00 3F 21 00 1E 00 21 3F 00  57FA
4290: 1F 22 22 1E 02 02 83 00 3F 22 04 08 04 22 3F 00  93DA
42A0: BF 08 08 08 08 08 8C 00 36 AD 08 08 08 08 8C 00  2A02
42B0: 63 A1 22 92 14 8C 08 00 1C 22 A1 A1 22 92 B3 00  5447
42C0: B3 22 14 08 14 22 B3 00 1C 08 6B 2A 1C 08 8C 00  F043
42D0: BF 22 10 08 04 22 BF 00 3E 06 06 06 06 06 3E 00  CC78
42E0: 00 02 04 08 10 20 00 00 3E 30 30 30 30 30 3E 00  E4AA
42F0: 08 1C 2A 08 08 08 08 08 00 00 00 00 00 00 00 00  F076
4300: 00 00 00 00 00 00 00 00 08 08 08 08 08 00 08 00  2D30
4310: 14 14 14 00 00 00 00 00 00 08 08 08 08 08 08 00  D26C
4320: 08 3C 0A 1C 28 1E 08 00 06 26 10 08 04 32 30 00  6C62
4330: 04 0A 0A 04 2A 12 2C 00 08 08 08 00 00 00 00 00  759C
4340: 08 04 02 02 02 04 08 00 08 10 20 20 20 10 08 00  0CAE
4350: 08 2A 1C 08 1C 2A 08 00 00 08 08 3E 08 08 00 00  D202
4360: 00 00 00 00 08 08 04 00 00 00 00 3E 00 00 00 00  6952
4370: 00 00 00 00 00 00 08 00 00 20 10 08 04 02 00 00  4546
4380: 1C 22 32 2A 26 22 1C 00 08 0C 08 08 08 08 1C 00  904E
4390: 1C 22 20 18 04 02 3E 00 3E 20 10 18 20 22 1C 00  F09E
43A0: 10 18 14 12 3E 10 10 00 3E 02 1E 20 20 22 1C 00  7E88
43B0: 38 04 02 1E 22 22 1C 00 3E 20 10 08 04 04 04 00  B43E
43C0: 1C 22 22 1C 22 22 1C 00 1C 22 22 3C 20 10 0E 00  DBB6
43D0: 00 00 08 00 08 00 00 00 00 00 08 00 08 08 04 00  962C
43E0: 10 08 04 02 04 08 10 00 00 00 3E 00 3E 00 00 00  F3B6
43F0: 04 08 10 20 10 08 04 00 1C 22 10 08 08 00 08 00  42BE
4400: 10 20 1C 20 3C 22 3C 00 08 14 22 22 3E 22 22 00  06E8
4410: 1E 22 22 1E 22 22 1E 00 1C 22 02 02 02 22 1C 00  CF64
4420: 1E 22 22 22 22 22 1E 00 3E 02 02 1E 02 02 3E 00  5488
4430: 3E 02 02 1E 02 02 02 00 3C 02 02 02 32 22 3C 00  E738
4440: 22 22 22 3E 22 22 22 00 1C 08 08 08 08 08 1C 00  066A
4450: 20 20 20 20 20 22 1C 00 22 12 0A 06 0A 12 22 00  0960
4460: 02 02 02 02 02 02 3E 00 22 36 2A 2A 22 22 22 00  F35C
4470: 22 22 26 2A 32 22 22 00 1C 22 22 22 22 22 1C 00  FCEC
4480: 1E 22 22 1E 02 02 02 00 1C 22 22 22 2A 12 2C 00  F070
4490: 1E 22 22 1E 0A 12 22 00 1C 22 02 1C 20 22 1C 00  0678
44A0: 3E 08 08 08 08 08 08 00 22 22 22 22 22 22 1C 00  2156
44B0: 22 22 22 22 22 14 08 00 22 22 22 2A 2A 36 22 00  7BD8
44C0: 22 22 14 08 14 22 22 00 22 22 14 08 08 08 08 00  E73A
44D0: 3E 20 10 08 04 02 3E 00 08 14 08 00 00 00 00 00  7EDE
44E0: 00 3C 02 02 02 3C 10 08 20 10 1C 22 3E 02 3C 00  1E80
44F0: 08 14 2A 08 08 08 08 08 00 00 00 00 00 00 00 7F  F2ED
4500: 00 02 04 08 10 20 00 00 00 00 1C 20 3C 22 3C 00  B414
```

NEWSET2 (suite)

```
4600: 00 04 08 12 24 08 10 00 00 00 00 2E 11 19 B7 00  BA69
4610: 00 B8 A4 3C A2 26 9D 01 00 00 A6 28 98 10 88 00  36FC
4620: 10 84 08 8E 12 12 0C 00 00 00 9C 82 1E 02 8E 00  A82E
4630: 90 10 3C AA 2A 9E 84 04 00 00 A6 28 98 10 88 00  30D4
4640: 00 00 36 A4 24 92 10 88 00 00 88 08 84 04 0C 00  124C
4650: 30 28 24 9E 12 0A 06 00 00 00 24 8A 06 89 31 00  36AA
4660: 06 84 08 8C 14 92 22 00 00 00 A4 24 92 2A 81 01  56EC
4670: 00 00 A3 24 14 0C 04 00 00 00 8C 92 22 92 8C 00  AB49
4680: 00 00 BF 15 14 14 92 00 08 38 84 9C 04 82 1C 10  B9A0
4690: 00 00 18 24 92 0E 81 01 00 00 00 BE 89 09 87 00  E135
46A0: 00 00 BF 09 84 04 82 00 00 00 A3 24 92 8A 86 00  603B
46B0: 00 00 A3 24 14 0C 04 00 00 00 00 23 29 2D 9B 00  1EFF
46C0: 00 00 91 14 8C 8C 14 A2 00 10 89 2A 95 8F 82 02  91DE
46D0: 04 9C 04 82 82 04 84 06 38 0C 0C 06 0C 0C 38 00  A8DC
46E0: 08 08 08 08 08 08 08 08 0E 18 18 30 18 18 0E 00  EAEC
46F0: 00 00 2C 1A 00 3E 00 00 7F 7F 7F 7F 7F 7F 7F 7F  2F7C
4700: 00 00 00 00 00 00 00 00 00 00 00 08 08 08 00 08  3620
4710: 00 00 14 14 00 00 00 00 00 00 10 3C 08 1E 04 00  E19E
4720: 00 00 08 1C 08 1C 08 00 00 00 14 10 08 04 14 00  6694
4730: 00 00 1C 14 1C 00 00 00 00 00 08 08 00 00 00 00  245C
4740: 00 00 08 04 04 04 08 00 00 00 08 10 10 10 08 00  875C
4750: 00 00 00 14 08 14 00 00 00 00 00 08 1C 08 00 00  545C
4760: 00 00 00 00 10 10 08 00 00 00 00 00 1C 00 00 00  2444
4770: 00 00 00 00 00 08 00 00 00 00 10 10 08 04 04 00  4E38
4780: 00 00 1C 14 14 14 1C 00 00 00 0C 08 08 08 1C 00  C0B4
4790: 00 00 1C 10 1C 04 1C 00 00 00 1C 10 18 10 1C 00  81D8
47A0: 00 00 04 14 1C 10 10 00 00 00 1C 04 1C 10 1C 00  0FBC
47B0: 00 00 1C 04 1C 14 1C 00 00 00 1C 10 08 08 08 00  6CB0
47C0: 00 00 1C 14 1C 14 1C 00 00 00 1C 14 1C 10 1C 00  89F4
47D0: 00 00 00 08 00 08 00 00 00 00 00 00 08 00 08 04  D024
47E0: 00 00 10 08 04 08 10 00 00 00 00 1C 00 1C 00 00  B46C
47F0: 00 00 04 08 10 08 04 00 00 00 1C 10 08 00 08 00  0C64
4800: 00 00 00 36 49 49 36 00 00 00 1C 14 1C 14 14 00  F672
4810: 00 00 0C 14 0C 14 0C 00 00 00 1C 04 04 04 1C 00  3690
4820: 00 00 0C 14 14 14 0C 00 00 00 1C 04 0C 04 1C 00  1BA0
4510: 02 02 1E 22 22 22 1E 00 00 00 3C 02 02 02 3C 00  0924
4520: 20 20 3C 22 22 22 3C 00 00 00 1C 22 3E 02 3C 00  F0D8
4530: 18 24 04 1E 04 04 04 00 00 00 1C 22 22 3C 20 1C  E242
4540: 02 02 1E 22 22 22 22 00 08 00 0C 08 08 08 1C 00  7EF2
4550: 10 00 18 10 10 10 12 0C 02 02 22 12 0E 12 22 00  A5F0
4560: 0C 08 08 08 08 08 1C 00 00 00 36 2A 2A 2A 22 00  7E26
4570: 00 00 1E 22 22 22 22 00 00 00 1C 22 22 22 1C 00  8A44
4580: 00 00 1E 22 22 1E 02 02 00 00 3C 22 22 3C 20 20  4980
4590: 00 00 3A 06 02 02 02 00 00 00 3C 02 1C 20 1E 00  09DE
45A0: 04 04 1E 04 04 24 18 00 00 00 22 22 22 32 2C 00  2D2E
45B0: 00 00 22 22 22 14 08 00 00 00 22 22 2A 2A 36 00  5A50
45C0: 00 00 22 14 08 14 22 00 00 00 22 22 22 3C 20 1C  8052
45D0: 00 00 3E 10 08 04 3E 00 20 10 1C 22 3E 02 3C 00  B782
45E0: 04 08 22 22 22 32 2C 00 10 20 1C 22 3E 02 3C 00  F0BA
45F0: 2C 1A 00 00 00 00 00 00 03 01 00 00 00 00 00 00  3C4A
```

Yvan KOENIG
répond à vos questions... (suite)

1010 PRINT CHR$ (4)"OPEN fichier" : PRINT CHR$(4)"READ fichier"

1020 GET X$: traitez X$ comme bon vous semble

1040 GOTO 1020

1050 PRINT CHR$(4)"CLOSE"

On peut également prévoir, AVANT écriture, de remplacer les virgules par un caractère inemployé (le tréma par exemple) et faire l'opération inverse en lecture.

Si l'on dispose d'une zone mémoire disponible capable de contenir le fichier à lire, on peut utiliser PRO.TYPY, version modifiée de PRO.TYPC pour y déposer nos précieuses virgules afin de les exploiter à loisir par la suite.

BLOAD PRO.TYPC
CALL - 151

```
610D : C8 B1 48 D9 C3 61 D0 25 (bug déjà mentionné)
6152 : 20 A0 61
615D : 20 A9 61
617E : A0 00
6180 : 98 20 AD 61 C8 D0 FA B9
6188 : 00 02 EA EA EA EA EA EA
6190 : EA EA 20 AD 61 C8 CC DB
6198 : BE D0 EC EE AF 61 D0 BB
61A0 : A9 60 8D AF 61 AD D0 BE
61A8 : 60 8C D9 BE 98 99 00 60
61B0 : 60
61C7 : 59
```

BSAVE PRO.TYPY,A$6037,L401

TYPY fichier déposera vos octets en $6000, l'octet haut de l'adresse de début du buffer se trouve en $61A1, les NOPs correspondent à un besoin précis de M. D. qui souhaite remplacer les "return" par des "linefeed" ce qui se fait avec la modification suivante :

```
618A : 09 80 C9 8D D0 02 A9 8A
```

Nous transmettrons volontiers vos lettres à Yvan KOENIG, mais n'oubliez pas de joindre une enveloppe timbrée pour la réponse et un timbre pour la réexpédition de votre courrier.

*Nous vous rappelons que, si le courrier de **TM** est gratuit, nous ne pouvons répondre qu'aux demandes respectant la règle du jeu énoncée plus haut.*

Tremplin Micro

• *Je possède un APPLE IIe et une imprimante PANASONIC KX-P1091. Tout fonctionne bien avec une carte interface dans le slot 1, mais il est impossible de faire démarrer l'imprimante avec APPLEWORKS. J'ai modifié la configuration, mais toujours rien...*

Je ne connais pas particulièrement votre imprimante, mais il me semble qu'il s'agit en fait de l'une des multiples "compatibles EPSON" à liaison parallèle. Vous ne dites rien de l'interface utilisée alors que c'est là que se trouve presque toujours le problème avec APPLEWORKS. Avec la version 1.2 de ce logiciel et les interfaces parallèles "type EPSON", il n'y a pas de solution. Il faut essayer de remplacer l'interface par un modèle accepté par AW (GRAPPLER par exemple). Je vous suggère une formule de dépannage : utilisez l'option *Ajoutez Imprimante, Impression sur disque.*

C'est gourmand en espace disque, mais vous pouvez, le cas échéant, couper votre texte en deux. Ensuite la version TYPE de la routine PRO.TYPC de *Tremplin Micro* n°9 vous permettra d'envoyer votre texte formaté vers votre imprimante.

• *Un petit problème à vous soumettre : certains logiciels reconnaissent automatiquement la présence de deux lecteurs de disquettes, et le programme ne se "plante" pas sur un "I/O ERROR" si un seul lecteur est présent. S'agit-il dans tous les cas d'un essai de lecture du D2 suivi d'un ONERR (etc.)... RESUME... ou alors existe-t-il une adresse (du côté de $60A...) qui permet, par un PEEK judicieux, de savoir s'il y a un ou deux lecteurs dans le port 6 ? Une courte routine serait la bienvenue pour les programmeurs en herbe que nous sommes, désireux d'améliorer le confort et la convivialité de nos programmes.*

En BASIC votre problème n'est pas difficile à régler.

```
xxx  D$=CHR$(4)
10Ø  NBD=2 : REM 2 drives par défaut
11Ø  ONERR GOTO 13Ø
12Ø  PRINT D$ "VERIFY XYZ,D2": GOTO 16Ø:
     REM Au cas où XYZ existerait !
13Ø  POKE 216,Ø: ER=PEEK(222): IF ER=6 THEN
     15Ø: REM Le fichier n'existe pas mais il y a
     un disque en place
14Ø  IF ER<>8 THEN PRINT "Erreur N°"ER"
     ligne "PEEK(218) + 256 * PEEK(219) : END
15Ø  NBD=1 : REM Un seul disque en place
16Ø  REM Suite du programme
```

Attention ! un disque DOS 3.3 en lecteur 2 serait traité comme une absence de disque sous ProDOS... et vice-versa. **Y.K.**

NEWSET2 (suite et fin)

```
4830: 00 00 1C 04 0C 04 04 00 00 00 1C 04 04 1C 1C 00  A890
4840: 00 00 14 14 1C 14 14 00 00 00 1C 08 08 08 1C 00  96BC
4850: 00 00 10 10 10 14 1C 00 00 00 14 0C 04 0C 14 00  2DA4
4860: 00 00 04 04 04 04 1C 00 00 00 22 36 2A 22 22 00  2DF2
4870: 00 00 12 16 1E 1A 12 00 00 00 1C 14 14 14 1C 00  84E6
4880: 00 00 1C 14 1C 04 04 00 00 00 1C 14 14 1C 1C 20  35F0
4890: 00 00 1C 14 0C 14 14 00 00 00 1C 04 1C 10 1C 00  81CC
48A0: 00 00 1C 08 08 08 08 00 00 00 14 14 14 14 1C 00  69A8
48B0: 00 00 14 14 14 14 08 00 00 00 22 22 2A 36 22 00  601E
48C0: 00 00 14 14 08 14 14 00 00 00 14 14 08 08 08 00  4E98
48D0: 00 00 1C 10 08 04 1C 00 00 00 1C 04 04 04 1C 00  7298
48E0: 00 00 04 04 08 10 10 00 00 00 1C 10 10 10 1C 00  6C98
48F0: 08 08 08 08 08 2A 14 08 00 00 00 00 00 00 00 7F  B3ED
4900: 20 20 20 20 20 20 20 00 00 00 00 1C 10 1C 1C 00  7E44
4910: 00 00 04 1C 14 14 1C 00 00 00 00 1C 04 04 1C 00  B[illegible]4
4920: 00 00 10 1C 14 14 1C 00 00 00 00 1C 1C 04 1C 00  1BC8
4930: 00 00 18 08 0C 08 08 00 00 00 00 1C 14 1C 10 1C  51B4
4940: 00 00 04 1C 14 14 14 00 00 00 08 00 08 08 08 00  EA7C
4950: 00 00 08 08 08 08 04 00 00 00 04 14 0C 0C 14 00  5468
4960: 00 00 0C 08 08 08 1C 00 00 00 00 3E 2A 2A 2A 00  72FC
4970: 00 00 00 1C 14 14 14 00 00 00 00 1C 14 14 1C 00  E7B8
4980: 00 00 00 1C 14 14 1C 04 00 00 00 1C 14 14 1C 10  BBD4
4990: 00 00 00 1C 04 04 04 00 00 00 00 18 04 10 0C 00  BA60
49A0: 00 00 08 18 08 08 10 00 00 00 00 14 14 14 1C 00  0F98
49B0: 00 00 00 14 14 14 08 00 00 00 00 2A 2A 2A 14 00  C6D6
49C0: 00 00 00 14 08 08 14 00 00 00 00 14 14 1C 10 18  4EA4
49D0: 00 00 00 1C 18 0C 1C 00 00 00 00 00 00 00 00 00  515C
49E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  0000
49F0: 00 00 00 00 00 00 00 00 00 3E 3E 3E 3E 3E 3E 7F  FBF3
```

BSAVE NEWSET2,A$4200,L2048

AMATEURS DE CARRÉS MAGIQUES... à vos claviers !

Vos preuves ne sont plus à faire, tant sur le plan des mots croisés, que sur celui de l'informatique. Je crois que vous êtes aussi un fervent de toutes les récréations intellectuelles. J'ai trouvé dans un vieux magazine informatique, un programme sur les carrés magiques. Mais je n'arrive pas à en faire quelque chose de valable sur mon APPLE. Les chiffres défilent et ne formeront jamais un carré avec une telle programmation. Ni doué en matière de carrés magiques et très moyen en programmation (mais fidèle lecteur de T.M. je me suis demandé si vous n'auriez pas concocté dans une revue (on ne peut tout lire) un petit programme de ce type qui correspondrait à ma recherche... Peut-être ce petit divertissement amuserait-il les lecteurs de T.M. (façon de réunir les civilisations qui cultivèrent, à travers le symbolisme, ces carrés (sans ordinateur) et nous "modernes" qui croyons toujours avoir tout inventé). Je continue à chercher une solution, mais si vous pouviez vous pencher sur cela et m'épauler, j'en serais ravi... **P.C.**

AVIS AUX AMATEURS ! Les meilleurs programmes seront publiés, mais je donnerais la préférence à celui qui se révèlera capable de réaliser des carrés magiques PAIRS et IMPAIRS. ***Guy-HACHETTE***

Votre bibliothèque INFORMATIQUE

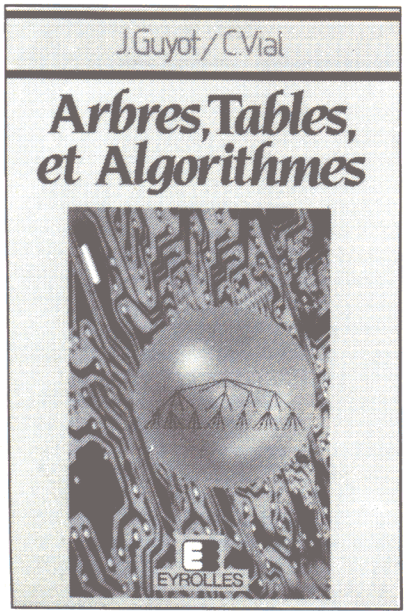
par Clément **RENARD**

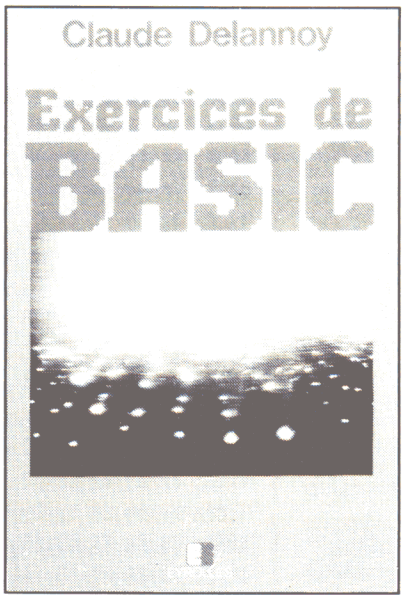
• *ARBRES, TABLES ET ALGORITHMES (Eyrolles)*

Les auteurs sont français et ingénieurs : Jacques Guyot est docteur en analyse numérique et Christian Vial docteur en informatique. Sans doute en déduisez-vous qu'ils sont on ne peut plus qualifiés pour traiter le sujet de cet ouvrage, et vous avez raison. Il est vrai que les auteurs sont également enseignants-chercheurs... d'où un livre parfaitement structuré, constituant un cours magistral sur des points souvent négligés par les programmeurs en herbe :

— Résolution informatique d'un problème (analyse et modélisation) ;
— Langage algorithmique (schéma de résolution) ;
— Structures des données (piles, files, arbres, graphes) ;
— Diverses méthodes de tri ;
— Gestion de tables, etc.

L'explication des diverses méthodes de tri, notamment, est particulièrement claire. Des schémas la rendent même attrayante. En conclusion : sujet(s) aride(s), mais dont l'étude sérieuse devrait permettre aux lecteurs de mieux maîtriser la programmation.





• *EXERCICES DE BASIC (Eyrolles)*

Nous avons déjà eu l'occasion de parler des ouvrages de Claude Delannoy, dont on connaît notamment *Apprendre à programmer en Basic* et *Initiation à la programmation* (même éditeur).

Comme le rappelle fort justement l'avant-propos de ce nouveau recueil, la maîtrise de la programmation ne s'obtient que par la pratique. Je suis tenté d'ajouter : et par l'étude d'exercices similaires à ceux du livre de Claude Delannoy.

Certes, le Basic utilisé n'est pas l'Applesoft, mais les lecteurs de *Tremplin Micro* se retrouveront néanmoins en terrain connu, même s'ils n'ont pas la possibilité d'utiliser le "IF... THEN... ELSE" que connaissent actuellement la plupart des Basic. Claude Delannoy leur montre en effet comment remplacer cette instruction par deux lignes.

J'ai beaucoup aimé la présentation de ces *Exercices de Basic*. Pour chaque exemple, l'auteur donne d'abord l'énoncé du problème à résoudre (pardon : à programmer), les données éventuelles, les résultats, un tableau obtenu à la suite d'un run, le programme lui-même, un autre tableau regroupant la liste des variables, puis une discussion du programme avec des conseils pour modifier ou remplacer telle ou telle instruction. A conseiller aux débutants.

• *DÉBUTER EN C (Eyrolles)*

Nous devons à Christian Le Bras cette traduction française du livre du Californien F. Richard Moore. L'auteur nous signale que son ouvrage ne devait constituer, à l'origine, que l'un des chapitres d'un manuel de musique par ordinateur... mais n'en déduisez pas qu'il ne s'adresse qu'à des lecteurs musiciens. Par contre, pour réellement en tirer profit, il est bon d'être motivé pour la programmation et surtout d'avoir accès à un ordinateur équipé du système d'exploitation Unix.

Débuter en C ne concerne sous doute qu'un petit nombre de lecteurs de *Tremplin Micro*... encore que l'on parle sérieusement d'un langage C sur nouvel Apple *IIGS*.

UTILITAIRE | DOS 3.3 Exclusivement

Fonctionne aussi sur l'APPLE IIGS

Marcel COTTINI

BIT MAP

PROGRAMME DEMANDÉ PAR DE NOMBREUX LECTEURS, TRÈS ATTACHÉS À LEUR VIEUX DOS 3.3 !

Visualiser l'occupation d'une disquette DOS 3.3 ne sera plus un problème pour le lecteur de *T.M.* Le programme BIT MAP, écrit en langage Basic AppleSoft, occupe 15 secteurs sur la disquette. Il permet la visualisation de l'occupation des secteurs de chaque piste au niveau de la disquette, ainsi que du nombre de secteurs libres, donc disponibles à l'utilisateur. Ces renseignements sont fournis par l'intermédiaire d'un graphique simulant la surface de la disquette à considérer. Les secteurs occupés seront matérialisés par le symbole X, tandis que les secteurs libres seront laissés vierges. Au bas du graphe sera inscrit le nombre total de secteurs libres. Certains modules composant le corps du programme pourront être intégrés dans vos réalisations futures comme sous-programme appelé par un GOSUB par exemple.

L'utilisation même du programme est très simple. Après avoir lancé son exécution, l'utilisateur pourra opter soit pour la lecture des instructions, soit poursuivre le déroulement des opérations. Dans le premier cas, un certain nombre de renseignements seront fournis à l'utilisateur, afin de lui donner tous les moyens d'une utilisation rationnelle. Dans le second cas, il vous sera offert d'être renseigné sur l'occupation de la disquette venant d'être lue, ou sur une éventuelle disquette de votre choix.

A propos du programme

Le programme implante à partir de l'adresse $0300 un petit sous-programme pour la recherche de l'adresse de la liste des paramètres pour la gestion des commandes, ainsi que l'adresse d'appel à RWTS. A partir de l'adresse $0308 (776) débute la table IOB interne au programme.

300 :	LDA £$03	Charge l'accumulateur avec le byte de poids fort de la table IOB.
302 :	LDY £$08	Charge le registre Y avec le byte de poids faible de la table IOB. Pointeur de la table IOB = $0308
307 :	RTS	Retour au sous-programme appelant, en l'occurrence le programme Basic.

Au retour du sous-programme machine (appel par CALL 768) il sera nécessaire de mettre à zéro le contenu du registre d'état (statut byte). Cette opération est effectuée à la ligne 240 (POKE 72,0).

A chaque consultation d'une nouvelle disquette, la VTOC (Volume Table Of Contents) sera lue, puis une conversion numérique réalisée aux lignes 710 à 730.

Les lignes 760 à 800 implantent le sous-programme machine à partir de l'adresse $0300 par la routine de S.H. LAM qui mérite quelques explications. La technique retenue consiste à placer dans une chaîne de caractères des commandes à destination du moniteur. Il sera alors possible d'installer en mémoire un court programme en langage machine avec une relative facilité. L'installation en mémoire s'effectue en trois étapes.

La chaîne est d'abord scindée en caractères qui sont placés dans le tampon du clavier (IN - $0200) les uns à la suite des autres. Cette opération occupe le tampon clavier à partir de l'adresse $0200 (512 en décimal). Dans un deuxième temps, le CALL - 144 appelle un sous-programme résident qui met en œuvre les routines examinant le tampon clavier et vérifiant l'exécution des commandes inscrites dans la chaîne de caractères.

En dernier lieu, il faut revenir au langage Basic. Pour ce faire, il suffit de se brancher à l'adresse $D9C6, ce qui correspond à un retour au mode RUN, suite de l'exécution du programme Basic AppleSoft. Le POKE 72,0 à la ligne 800, met le registre d'état à zéro (statut byte -- adresse $48).

```
  1 REM ************************
  2 REM *  SECTEURS LIBRES     *
  3 REM *   TRACE LA BIT MAP   *
  4 REM * COPYRIGHT AVRIL 85   *
  5 REM * MARCEL COTTINI (H-H) *
  6 REM ************************
  7 REM ctJctJctJ
 10 GOSUB 760: REM MET EN PLACE SOUS-PROG
    RAMME MACHINE                           1B4D
 20 TEXT : HOME                             1C5A
 30 VTAB 5: HTAB 10: INVERSE : PRINT " ¯¯
    ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯"                     8B98
 40 VTAB 6: HTAB 10: PRINT " ¯DISK FREE S
    PACE ¯¯"                                F0DA
 50 VTAB 7: HTAB 10: PRINT " ¯¯¯¯¯¯¯¯¯¯¯¯
    ¯¯¯¯¯¯¯¯"                               B1C2
 60 VTAB 8: HTAB 10: PRINT " AFFICHE LA B
    IT MAP "                                F813
 70 VTAB 9: HTAB 10: PRINT " ¯¯¯¯¯¯¯¯¯¯¯¯
    ¯¯¯¯¯¯¯¯"; NORMAL                       619B
 80 IF PR = 1 THEN  GOTO 150                4E55
 90 VTAB 11: HTAB 1: PRINT "VOULEZ VOUS L
    ES INSTRUCTIONS (O/N) ";                9600
100 GET A$: PRINT A$: IF A$ = "O" THEN  G
    OSUB 820: GOTO 120                      DCB1
110 IF A$ <  > "N" GOTO 90                  2D58
120 VTAB 13: PRINT "ESPACE LIBRE DU DISK
    QUI VIENT"                              3136
130 VTAB 14: PRINT "D'ETRE LU PAR LE 'DOS
    ' ? (O/N) ";                            E2BA
140 GET A$: IF A$ = "O" THEN VTC = 46011:
    GOTO 310: REM VTC = ADRESSE MéMOIRE B
    IT MAP                                  3BC8
150 VTC = 36864: REM ADRESSE $9000          02C8
160 VTAB 16: HTAB 5: PRINT " NUMERO SLOT:
     ";: GET A$: PRINT A$:S =  VAL (A$):
    IF S < 1 OR S > 7 THEN  GOTO 160        E34A
170 VTAB 18: HTAB 5: PRINT "NUMERO DRIVE:
     ";: GET A$: PRINT A$:D =  VAL (A$):
    IF D < 1 OR D > 2 THEN  GOTO 170        C933
180 IOB = 776: REM TABLE IOB                7C4E
190 POKE IOB + 1,S * 16: REM NUMERO DU SL
    OT                                      713C
200 POKE IOB + 2,D: REM NUMERO DU DRIVE     D5FD
210 POKE IOB + 4,17: REM PISTE A LIRE
                                            1823
220 POKE IOB + 5,0: REM SECTEUR à LIRE      3CEC
230 CALL 768: REM APPEL A RWTS              8331
240 POKE 72,0: REM MET A 0 REGISTRE D'ETA
    T PS                                    A47E
250 ER =  PEEK (IOB + 13): REM CODE ERREU
    R                                       CDA0
260 IF ER = 0 THEN VTC = 36864: GOTO 310    3649
270 VTAB 20: HTAB 5: PRINT "UNE ERREUR A
    ETE RENCONTREE"                         8B92
280 VTAB 21: HTAB 5: PRINT "PENDANT LA LE
    CTURE DU DISK."                         5A5F
290 VTAB 22: HTAB 5: PRINT "LE CODE ERREU
    R EST :¯";ER                            645C
300 END                                     0180
310 REM ctJctJctJECRITURE VTOCctJctJctJ
320 HOME                                    2F97
330 GOSUB 610: REM CHARGE LA TABLE          1747
340 GOSUB 700: REM CHARGE LA TABLE NUMERI
    QUE                                     FF47
350 GOSUB 500: REM CODES INVERSES           1B45
360 FOR I = 0 TO 34                         82F2
370 X = VTC + 56 + I * 4: REM ADRESSE BIT
     MAP PISTE I                            5557
380 X0 =  INT ( PEEK (X) / 16):X1 =  PEEK
    (X) - X0 * 16:X2 =  INT ( PEEK (X + 1
    ) / 16):X3 =  PEEK (X + 1) - X2 * 16    E5E4
390 PS$ = TT$(X0) + TT$(X1) + TT$(X2) + T
    T$(X3)                                  1289
400 FOR II = 15 TO 0 STEP  - 1              C2FB
410 VTAB 20 - II: HTAB 4 + I: PRINT  MID$
    (PS$,II + 1,1)                          787E
420 NEXT II                                 4714
430 FS = FS + NN(X0) + NN(X1) + NN(X2) +
    NN(X3)                                  B3FC
440 NEXT I                                  9DCB
450 VTAB 21: HTAB 4: INVERSE : PRINT " ¯-
    --> ¯"FS" SECTEURS ¯LIBRES": NORMAL     ECA1
460 VTAB 24: HTAB 25: INVERSE : PRINT "CO
    NTINUER (O/N)";: NORMAL                 6A55
470 GET A$: IF A$ = "O" THEN  CLEAR :PR =
    1: GOTO 20                              8777
480 IF A$ <  > "N" THEN 460                 EAA2
490 HOME : END                              8E51
500 VTAB 1: PRINT " ¯¯¯¯¯¯¯¯¯PISTES"        5503
510 NM$ = "SECTEURS": FOR X = 1 TO 8: VTA
    B 6 + X: HTAB 1: PRINT  MID$ (NM$,X,1
    ): NEXT X                               1D64
520 CA$ = " 0000000000111111111122222222
    233333 "                                80B9
530 VTAB 3: HTAB 3: INVERSE : PRINT CA$     0E4C
540 CA$ = " 01234567890123456789012345678
    901234 "                                931D
550 VTAB 4: HTAB 3: PRINT CA$               C075
560 VTAB 21: HTAB 3: PRINT " ¯¯¯¯¯¯¯¯¯¯¯¯
    ¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯¯"          15C0
570 VTAB 5: FOR I = 0 TO 9: HTAB 3:: PRIN
    T CHR$ (48 + I): NEXT I                 25C0
580 VTAB 15: FOR I = 0 TO 5: HTAB 3: PRIN
    T CHR$ (65 + I): NEXT I                 6CB2
590 VTAB 5: FOR I = 0 TO 16: HTAB 39: PRI
```

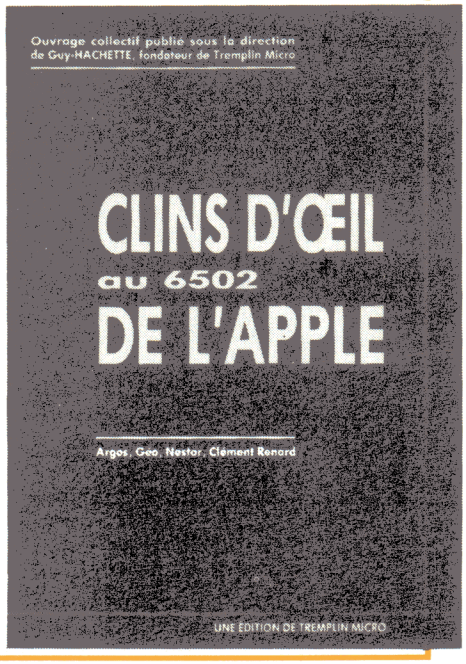
```
    NT " ": NEXT I: NORMAL                                  BA73
600 RETURN                                                  63B1
610 REM ctJctJCHARGE LA TABLEctJctJ
620 DIM TT$(15)                                             D609
630 FOR I = 0 TO 15: READ TT$(I): NEXT I                    BA1D
640 RETURN                                                  63B1
650 REM ctJctJDONNEES DE LA TABLEctJctJ
660 DATA "XXXX","XXX ","XX X","XX "                         74D7
670 DATA "X XX","X X ","X  X","X  "                         48D7
680 DATA " XXX"," XX "," X X"," X  "                        49F7
690 DATA "  XX","  X "," ""X","  """                        FAF7
700 REM ctJctJCONVERSION NUMERIQUEctJctJ
710 DIM NN(15)                                              5BD9
720 FOR I = 0 TO 15: READ NN(I): NEXT I                     BCED
730 RETURN                                                  63B1
740 REM ctJctJDONNES NUMERIQUES ctJctJ
750 DATA 0,1,1,2,1,2,2,3,1,2,2,3,2,3,3,4                    7037
760 HEX$ = "300:A9 03 A0 08 20 D9 03 60 0
    1 60 01 00 00 00 19 03 00 98 FF FF 01
     00 00 60 01 00 01 EF D8"                               C76E
770 HEX$ = HEX$ + " ND9C6G": REM RETOUR '
    RUN' PROCHAINE LIGNE                                    5499
780 FOR I = 1 TO  LEN (HEX$): REM LONGUEU
    R DE LA CHAINE                                          D1C9
790 POKE 511 + I, ASC ( MID$ (HEX$,I,1))
    + 128: REM STOCKE LA CHAINE DANS BUFF
    ER CLAVIER                                              C73D
800 NEXT : POKE 72,0: CALL  - 144: REM AP
    PEL SOUS PROGRAMME BUFFER                               7C62
810 RETURN                                                  63B1
820 REM ctJctJctJINSTRUCTIONSctJctJctJ
830 VTAB 11: HTAB 1                                         0905
840 PRINT "CE PROGRAMME LIT LA TABLE D'OC
    CUPATION"                                               480E
850 PRINT "(VTOC) DU DISK ET LA RESTITUE
    A L'ECRAN"                                              5AE7
860 PRINT "EN SECTEURS OCCUPES (MARQUES P
    AR DES X)"                                              0357
870 PRINT                                                   75BA
880 PRINT "LE PROGRAMME AFFICHE LES PISTE
    S (DE 0"                                                F07C
890 PRINT "A 34) ET LES SECTEURS DES PIST
    ES (DE F"                                               4B65
900 PRINT "A 0).LES SECTEURS OCCUPES SONT
    ,MATERIA-"                                              A14A
910 PRINT "LISES PAR DES 'X',LES SECTEURS
     LIBRES,"                                               73EE
920 PRINT "PAR DES VIDES."                                  77A6
930 PRINT                                                   75BA
940 PRINT "POUR UNE DISQUETTE DIFFERENTE
    DE CELLE- ";                                            8CD6
950 PRINT "CI, METTEZ-LA D'ABORD EN PLACE
    , PUIS"                                                 2C21
960 PRINT "REPONDEZ 'N' A LA PROCHAINE QU
    ESTION."                                                E9AE
970 GOSUB 990                                               1952
980 RETURN                                                  63B1
990 VTAB 24: HTAB 1: INVERSE : PRINT " (C
    )ONTINUE OU (ESC) POUR QUITTER..... "
    ;: NORMAL                                               961D
1000 GET A$: IF  ASC (A$) = 27 THEN  HOME
    : NORMAL : END                                          95CB
1010 VTAB 11: HTAB 1: CALL  - 958                           BC3A
1020 RETURN                                                 63B1
```

Une édition à ne pas manquer !

N'attendez plus et plongez-vous dès maintenant dans l'étude du langage machine. Ce n'est pas aussi difficile que vous le pensez et l'on obtient rapidement, à partir de routines simples, mais efficaces, des résultats surprenants.
La rapidité du langage machine — celui que votre Apple comprend le mieux — est proprement stupéfiante.

CLINS D'ŒIL au 6502 DE L'APPLE

regroupe les principales routines publiées dans les premiers numéros de *TREMPLIN MICRO*, mais aussi plusieurs programmes inédits. Une disquette est fournie avec le livre (utiliser le bulletin de commande, à la fin de la revue).



L'INSTRUCTION BIT

UTILE ET MAL CONNUE

Supposons une routine commune à 4 parties de votre programme. Chacune de ces 4 parties a une entrée particulière, qui positionne une variable VAR. La routine se compose :

- de fonctions exécutées dans tous les cas ;
- de fonctions qui peuvent être modifiées ou sautées pour certaines valeurs de VAR.

Il s'agit là d'un schéma courant, que l'on trouve dans presque toutes les analyses. Il illustre deux utilisations simples et commodes de l'instruction BIT :

1. Routine à plusieurs entrées. Si VAR prend, selon l'entrée, les valeurs $00, $40, $80 et $C0 vous pouvez écrire :

```
ENTREE1   LDA  £$00
          JMP  COMMUN
ENTREE2   LDA  £$40
          JMP  COMMUN
ENTREE3   LDA  £$80
          JMP  COMMUN
ENTREE4   LDA  £$C0
          STA  VAR
```

Vous gagnerez de la place en écrivant :

```
VAR       EQU  $810
ENTREE1   LDA  £$00
          DFB  $2C
ENTREE2   LDA  £$40
          DFB  $2C
ENTREE3   LDA  £$80
          DFB  $2C
ENTREE4   LDA  £$C0
          STA  VAR
```

Une fois assemblé, ce code vous donnera (avec vos entrées respectivement en $820, $823, $826 et $829) :

```
820 :  A9  00       LDA  £$00
822 :  2C  A9  40    BIT  $40A9
825 :  2C  A9  80    BIT  $80A9
828 :  2C  A9  C0    BIT  $C0A9
82B :  8D  10  08    STA  $0810

820 :  A9  00  2C  A9  40  2C  A9  80
828 :  2C  A9  C0  8D  10  08
```

Après chaque entrée, sauf la dernière, une ou plusieurs instructions BIT s'interposent. Elles peuvent modifier la retenue ou le bit V du registre d'état. Mais comme vous ne testez immédiatement ni l'une ni l'autre, le déroulement de votre programme n'en sera pas perturbé.

2. Tester une variable sans modifier les valeurs de Ac, de X ou de Y.

Nous avons donné à VAR des valeurs qui préparent le test par l'instruction BIT. En effet, avec ces valeurs, si vous écrivez BIT VAR, BMI vous branche sur les deux valeurs négatives $80 et $C0 (entrées 3 et 4). Entre elles, BVC vous branche sur la valeur $80 (bit 6 clear, la valeur $80 s'écrit sur son octet avec le bit 7 à 1 et les autres à 0). Parallèlement, BPL vous branche sur les deux valeurs positives $00 et $40, entre lesquelles BVC vous départage.

Le test complet de VAR, pour choisir par exemple entre 4 sorties, s'écrira :

```
BIT  VAR          .......
BMI  NEGATIF
BVC  SORTIE1      NEGATIF  BVC  SORTIE3
JMP  SORTIE2               JMP  SORTIE4
                  .......
```

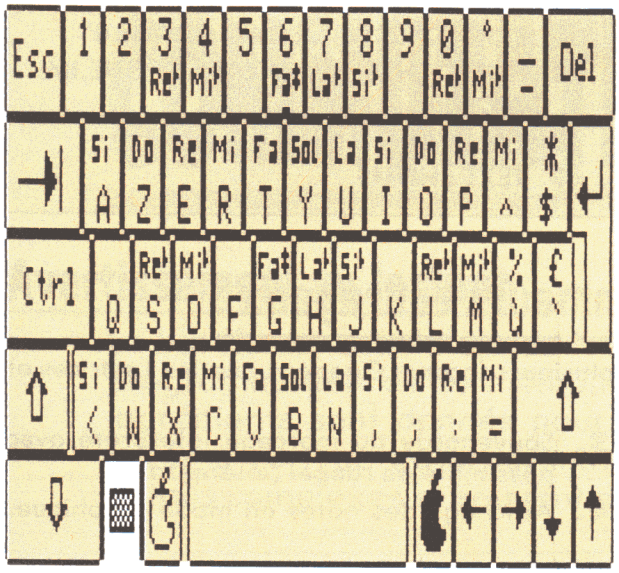
Recommandation importante : après le test d'une variable par BIT, faites vos branchements uniquement avec BMI, BPL, BVC et BVS. Si vous avez des "bugs" pensez à vérifier que vous avez bien tenu compte de cette recommandation.

Madeleine HODÉ, auteur de GRIBOUILLE.

MUSIQUE | DOS | ProDOS

Fonctionne aussi sur l'APPLE IIGS

Roland JOST



ORGUE sur Apple

Dans le numéro 6 de *Tremplin Micro*, un lecteur demandait un programme transformant le clavier de l'Apple en orgue. En voici un pour tous les lecteurs qui n'ont pas un ENSONIQ dans leur micro. La disposition des touches rappelle celle d'un piano (cf. figure).

Le haut-parleur de l'Apple, s'il est mis sous tension à intervalles judicieux, permet de générer des notes sur plusieurs octaves :

NOTE	FRÉQUENCE					
	Octave 1		Octave 2		Octave 3	
	déc.	hexa.	déc.	hexa.	déc.	hexa.
Sol	128	$80	64	$40	32	$20
Fa£/Solb	136	$88	68	$44	34	$22
Fa	145	$91	72	$48	36	$24
Mi	152	$98	76	$4C	38	$26
Re£/Mib	162	$A2	81	$51	40	$28
Re	172	$AC	86	$56	43	$2B
Do£/Reb	182	$B6	91	$5B	45	$2D
Do	192	$C0	96	$60	48	$30
Si	204	$CC	102	$66	51	$33
La£/Sib	216	$D8	108	$6C	54	$36
La	230	$E6	115	$73	57	$39
Sol£/Lab	242	$F2	121	$79	60	$3C
Sol	255	$FF	128	$80	64	$40

Ces valeurs se retrouvent dans la table des fréquences.

PROGRAMME :

```
300 : 20 53 D5 AA E0 0D D0 01
308 : 60 A0 50 84 07 BC 30 03
310 : F0 EE 84 06 2C 30 C0 88
318 : D0 04 C6 07 F0 E2 CA D0
320 : F6 A6 06 4C 14 03 00 00
```

TABLE DES FRÉQUENCES :

```
328 : 00 00 00 00 00 00 00 00
330 : 00 00 00 00 00 00 00 00
338 : 00 00 00 00 00 00 00 00
340 : 00 00 00 00 00 00 00 00
348 : 00 00 00 00 00 00 00 00
350 : 00 36 5B 00 00 00 00 51
358 : 00 28 19 00 66 00 72 66
360 : 2D 00 00 5B 51 00 44 3C
368 : 36 00 56 60 CC 4C 00 00
370 : 2D 66 80 98 A2 56 00 88
378 : 79 33 6C 00 5B 51 73 30
380 : 2B 00 4C B6 48 39 91 C0
388 : AC 40 60 28 00 44 26 00
390 : 00 00 00 00 00 00 00 00
398 : 00 00 00 00 00 00 00 00
3A0 : 00 1B 00 1C 1D 1E 00 00
3A8 : 00 1F 00 00 20 3C 00 22
```

Sauvegarde par **BSAVE ORGUE.0,A$300,L$B0**

Comment utiliser ORGUE ?

Contentez-vous de taper le petit programme ci-contre. Il se chargera d'appeler la routine en langage machine (ORGUE.Ø) et l'image du clavier... si vous avez eu le bon goût de la composer. Vous pouvez évidemment utiliser ORGUE sans le dessin du clavier, mais c'est moins facile. *ORGUE.IM* figure sur la disquette *Tremplin Micro* (33 secteurs). Notez que la frappe de RETURN déconnecte immédiatement le clavier musical.

ORGUE.DEMO

```
10 TEXT : HOME : PRINT CHR$ (21): HGR
20 POKE 49234,0: REM  PAS DE TEXTE
30 PRINT CHR$ (4)"BLOAD ORGUE.Ø"
40 PRINT CHR$ (4)"BLOAD ORGUE.IM,
   A$2000"
50 CALL 768
60 TEXT : HOME
```

Extensions possibles pour programmeurs courageux :

Ce programme pourrait être modifié ou amélioré de plusieurs façons. Gageons que les Lectrices et les Lecteurs de *TM* vont s'y essayer !

1. Changement d'octave par pression d'une touche POM ou TAB ;
2. Possibilité de changer de tempo ;
3. Sauvegarde du morceau interprété avec possibilité de rappel (intéressant) ;
4. Affichage des notes en mode graphique.

Assemblage par ProCODE

```
                        0  * Source du programme de génération de sons:
                        1
                        2  *
                        3  * ORGUE
                        4  *
                        5  DUREE     EQU  $07
                        6  FREQU.    EQU  $06
                        7  *
                        8  INCHR     EQU  $D553       ; Entrée d'un caractère
                        9  SPKR      EQU  $C030       ; Active le haut-parleur
                       10  *
                       11             ORG  $300
                       12  *
0300: 20 53 D5          13  ORGUE     JSR  INCHR       ; saisie d'une touche
0303: AA                14             TAX              ; au clavier.
0304: E0 0D             15             CPX  £$0D        ; est-ce un <RETURN>?
0306: D0 01             16             BNE  01          ; non , on continue.
0308: 60                17             RTS              ; oui , retour au BASIC.
0309: A0 50             18  01         LDY  £$50        ; durée d'une note
030B: 84 07             19             STY  DUREE
030D: BC 30 03          20             LDY  $330,X      ; recherche dans table
0310: F0 EE             21             BEQ  ORGUE       ; si 0 , attente touche.
0312: 84 06             22             STY  FREQU.      ;
0314: 2C 30 C0          23  02         BIT  SPKR        ; active le haut-parleur
0317: 88                24  03         DEY              ; et on boucle .....
0318: D0 04             25  04         BNE  ORGUE2
031A: C6 07             26             DEC  DUREE
031C: F0 E2             27             BEQ  ORGUE
031E: CA                28  ORGUE2    DEX
031F: D0 F6             29             BNE  03
0321: A6 06             30             LDX  FREQU.
0323: 4C 14 03          31             JMP  02          ; boucle obligée.
```

Marcel COTTINI

COMMUTATEURS LOGIQUES

Avertissement au lecteur

Le but de cet article est de lever le voile sur l'aspect souvent flou de la programmation de la mémoire vive, principale ou auxiliaire. Tous les ingrédients sont donnés pour que le programmeur averti puisse avoir accès à ce type de programmation sans pour autant rentrer dans des détails complexes et souvent mals acceptés par le lecteur.

Pourquoi une mémoire auxiliaire ?

L'Apple *II* dans sa version de base intègre sur sa carte mère des "chips" ayant une capacité de mémoire vive de 64 Ko, disponibles à l'utilisateur. Le microprocesseur de la première génération, un 65Ø2, ne permet d'adresser que 65536 (2^{16}) adresses de la mémoire, les 64 Ko se trouvant sur la carte mère. Très vite, le concepteur s'est aperçu que sa "machine" manquait de place pour implanter des programmes devenant de plus en plus performants. L'évolution dans le domaine informatique s'effectue suivant une courbe ascensionnelle très accentuée, par le fait du boum technologique mis au service du grand public. C'est ainsi que l'on trouve des cartes interfaces d'extension mémoire de toutes les tailles (64 Ko, 128 Ko, 512 Ko, etc.).

Une question vient alors à l'esprit : si le microprocesseur 65Ø2 (ou le 65CØ2 de l'Apple *IIc*) ne peut adresser que 64 Ko, à quoi servent les 64 Ko de la mémoire auxiliaire ? Une deuxième question, plus pertinente que la précédente pose le problème du procédé devant être mis en place pour atteindre cette mémoire auxiliaire.

Le mécanisme est en fait très simple. Il suffit d'imaginer que les deux mémoires sont alignées l'une en face de l'autre et séparées par une ligne fictive et imaginaire, comme deux rangées de maisons séparées par une rue. Pour passer d'une rangée de maisons à celle d'en face, il suffit de franchir la rue qui les sépare. Pour permettre au processeur de dialoguer soit avec la mémoire principale, soit avec la mémoire auxiliaire, il suffit d'actionner ce qu'on appelle un **commutateur logique**. Pour ceux qui pratiquent l'Apple *II* depuis un certain temps, ces commutateurs sont souvent appelés des "switches".

L'Apple *II* est pourvu, de par sa conception d'origine, de toute une série de commutateurs, dont les différents aspects seront abordés dans cet article. Ces commutateurs sont implantés dans la ROM (mémoire morte) et placés en parallèle sur la RAM (mémoire vive).

Définition

Un commutateur logique permet de sélectionner ou commuter un endroit déterminé de l'espace mémoire (RAM — mémoire vive — ou ROM — mémoire morte).

C'est ainsi que votre Apple *IIe* avec une carte 8Ø colonnes étendue en place ou le *IIc*, possèdent 128 Ko de mémoire vive (RAM). La RAM se répartit en 64 Ko de mémoire principale et 64 Ko de mémoire auxiliaire. Chaque zone de cet espace peut être programmée par l'utilisateur averti soit en lecture, soit en écriture, ou encore en lecture/écriture, par le simple fait d'activer des commutateurs logiques. Ces différents modes sont sélectionnés en écrivant ou en lisant à certaines adresses qui servent de point d'entrée aux commutateurs logiques.

La plupart de ces commutateurs ont trois points d'entrées :

1. un pour les connecter ;
2. un autre pour les débrancher ;
3. un troisième pour lire l'état du commutateur.

Ces commutateurs permettent de "switcher" (commuter) la mémoire principale ou la mémoire auxiliaire à partir d'une interface qui pourra être soit un

(suite page 18).

programme Basic, soit un programme assembleur. Le fait d'écrire (par POKE en Basic, STA en binaire) ou de lire (par PEEK en Basic, LDA en binaire) dans certaines adresses mémoire, renseigne le système sur ce qu'il doit afficher à l'écran et où il doit lire ou écrire les données qui lui sont nécessaires : mémoire principale et/ou mémoire auxiliaire.
L'utilisation de ces commutateurs logiques est cependant réservée aux programmeurs avertis. Certains logiciels du commerce ont été développés et conçus pour ce genre de manipulation. Une mauvaise utilisation ou un manque de pratique peut rendre l'exécution d'un programme aléatoire et même lui faire perdre des données.

Commutation des différents modes

Il existe deux types de commutateurs :
1. les commutateurs d'affichage ;
2. les commutateurs affectant la mémoire.

Ces différents modes de commutation sont activés en lisant ou en écrivant à certaines adresses de la mémoire servant de point d'entrée aux commutateurs logiques. En temps normal, les commutateurs sont gérés par les instructions et commandes à partir d'un programme implanté en mémoire centrale ou à partir de la ROM résidente (AppleSoft ou Moniteur). En effet, certaines routines internes de la mémoire morte par exemple, utilisent ces commutateurs pour leurs propres besoins. Les sous-programmes MLI de PRODOS, logés dans les 16 Ko de la mémoire haute, utilisent ce mode de commutation pendant l'exécution d'un programme, pour écrire ou lire des données soit dans la mémoire principale, soit dans la mémoire auxiliaire.

Tableau des zones mémoire

Mode de visualisation	Page	Adresse basse	Adresse haute
Texte 40 colonnes Graphique basse résolution	1 2	$0400 ou 1024 $0800 ou 2048	$07FF ou 2047 $08FF ou 3071
Texte 80 colonnes	1	$0400 ou 1024	$07FF ou 2047
Graphique haute résolution - 8 Ko	1 2	$2000 ou 8192 $4000 ou 16384	$3FFF ou 16383 $5FFF ou 24575
Graphique haute résolution - 16 Ko	1	$2000 ou 8192	$3FFF ou 16383

Tableau des commutateurs d'affichage

Nom	Signification	Adresse		Com.	Lire
		Hex	Décimal		Ecrire
TEXT	Ecran texte	$C051	- 16303	on	
	Ecran graphique	$C050	- 16304	off	
	Lecture de l'état de TEXT	$C01A	- 16358		lire
MIXED	Texte au bas du graphique	$C053	- 16301	on	
	Graphique pleine page	$C052	- 16302	off	
	Lecture de MIXED	$C01B	- 16357		lire
PAGE2	Affichage de la page 2	$C055	- 16299	on	
	Affichage de la page 1	$C054	- 16300	off	
	Lecture de PAGE2	$C01C	- 16356		lire
HIRES	Graphique haute résolution	$C057	- 16297	on	
	Graphique basse résolution	$C056	- 16298	off	
	Lecture de HIRES	$C01D	- 16255		lire
80COL	Texte 80 colonnes	$C00D	- 16371	on	écrire
	Texte 40 colonnes	$C00C	- 16372	off	écrire
	Lecture de 80COL	$C01F	- 16353		lire
80STORE	Accès page 1 auxiliaire	$C001	- 16383	on	écrire
	Retour mémoire principale	$C000	- 16384	off	écrire
	Lire l'état de 80STORE	$C018	- 16360		lire

Tableau des commutateurs mémoires

Nom	Signification	Adresse		Com.	Lire
		Hex	Décimal		Ecrire
RAMRD	Lecture mémoire auxiliaire	$C003	- 16381	on	écrire
	Lecture mémoire principale	$C002	- 16382	off	écrire
	Lire le commutateur RAMRD	$C013	- 16365		lire
RAMWRT	Ecriture en mémoire AUX	$C005	- 16379	on	écrire
	Ecriture en mémoire MAIN	$C004	- 16380	off	écrire
	Lire le commutateur RAMWRT	$C014	- 16354		lire
ALTZP	Commute pile, page Ø AUX	$C009	- 16373	on	écrire
	Commute pile, page Ø MAIN	$C008	- 16374	off	écrire
	Lire le commutateur ALTZP	$C016	- 16352		lire
80STORE	Accès page 1 AUX	$C001	- 16383	on	écrire
	Pour utiliser RAMRD/RAMWRT	$C000	- 16384	off	écrire
	Lire le commutateur 80STORE	$C018	- 16360		lire
PAGE2	Accès mémoire AUX	$C055	- 16299	on	
	Accès mémoire MAIN	$C054	- 16300	off	
	Lire le commutateur PAGE2	$C01C	- 16356		lire
HIRES	Accès page 1 graphique AUX	$C057	- 16297	on	
	Pour utiliser RAMRD/RAMWRT	$C056	- 16298	off	
	Lire le commutateur HIRES	$C01D	- 16355		lire

AUX = Mémoire auxiliaire MAIN = Mémoire principale.

Remarques

MIXED et HIRES n'ont d'effet sur la visualisation que lorsque TEXT est "off". La fonction PAGE2 est différente si 80STORE est "on". Lorsque 80STORE est connecté, PAGE2 n'a plus d'effet sur la visualisation, il permet au processeur d'adresser, soit la page 1 de la mémoire principale (MAIN), soit la page 1 de la mémoire auxiliaire (AUX).

Pour actionner PAGE2 et 80STORE, il est nécessaire de passer par un programme, car l'Apple *II* utilise ces commutateurs logiques lorsque celui-ci est en mode 80 colonnes.

La page 1 texte de la mémoire principale (MAIN), a sa copie en mémoire auxiliaire (AUX) aux mêmes adresses. Pour stocker des informations en page 1 AUX, il faut connecter 80STORE en écrivant à l'adresse $C001 (-16383), puis actionner PAGE2 en écrivant ou en lisant à l'adresse $C055 (- 16299). Les premiers caractères qui apparaissent en colonnes impaires sont en mémoire principale, les suivants, colonnes paires, sont en mémoire auxiliaire aux mêmes adresses.

Les commutateurs d'accès aux pages d'écran ont priorité sur les commutateurs RAMRD et RAMWRT.

La ROM AppleSoft est en parallèle avec la Bank switched memory. Son activation est indépendante de la position des commutateurs (voir le tableau des commutateurs mémoires).

En mode graphique haute résolution 16 Ko (560 points horizontaux), on visualise 40 paires de 14 points par ligne de graphique, les 7 premiers de chaque paire étant en mémoire auxiliaire, les 7 suivants en mémoire principale. Ainsi les points Ø à 6, 14 à 20... 545 à 552 d'une même ligne sont implantés en page 1 AUX, tandis que les points 7 à 13, 21 à 28... 553 à 560 d'une même ligne sont implantés en page 1 MAIN.

Pour placer des informations directement dans la page 1 haute résolution AUX, branchez 80STORE et HIRES, puis PAGE2.

La haute résolution 8 Ko utilise la visualisation normale, à savoir la mémoire principale (HGR — adresses $2000 à $3FFF ou HGR2 — adresses $4000 à $5FFF).

Le texte 80 colonnes et la haute résolution 16 Ko utilisent à la fois la mémoire principale et la mémoire auxiliaire. Ce type de graphique désigne les cinq modes de visualisation avec l'utilisation d'une carte couleur du type chat mauve ève. En mode d'affichage 80 colonnes, les caractères en colonnes impaires sont inscrits en mémoire principale et ceux des colonnes paires en mémoire auxiliaire.

Commutation de la Bankswitch sous ProDOS

- 16 Ko du haut de la mémoire (RAM)

Le fichier PRODOS, une fois la disquette système bootée, se loge dans la carte langage d'un Apple *II*, ou en haut de la mémoire vive d'un Apple *IIe* ou *IIc*. Il occupe alors les 16 Ko du haut de la mémoire vive. Comme les adresses $D000-$FFFF ne comptabilisent que 12 Ko, on pourrait se

(suite page 20).

demander où se logent les 4 Ko manquants. C'est la bank 2 qui est utilisée conjointement avec la bank 1 par simple commutation alternée des deux banks. L'action des commutateurs logiques permet, entre autres, d'effectuer soit une lecture, soit une écriture, soit encore une lecture et écriture de l'espace mémoire du haut de la RAM.

Situation typique de la mémoire principale (64 Ko)

ROM : $D000-$FFFF *
Bank 1 : $D000-$DFFF
Bank 2 : $D000-$DFFF
* (AppleSoft + Moniteur)

Situation typique de la mémoire auxiliaire (64 Ko)

RAM : $D000-$FFFF *
Bank 1 : $D000-$DFFF *
Bank 2 : $D000-$DFFF *
* (Disponible à l'utilisateur)

Activation des commutateurs

$D000-$DFFF Bank 1 avec 4 Ko
$D000-$DFFF Bank 2 avec 4 Ko
$E000-$FFFF Bank 1 et 2 avec 8 Ko

Modes de commutation de la Memory Bank

Commutateurs	Bank 1	Bank 2	ROM
2 × LDA $C08B	R/W		
2 × LDA $C083		R/W	
2 × LDA $C089	W		R
2 × LDA $C081		W	R

Légende : *R/W = Lecture/écriture R = Lecture W = Ecriture*

Adresses de contrôle :

$C080 (- 16256) Sélectionne la RAM en lecture et commute la Bank 2. Protège la RAM contre l'écriture.

$C081 (- 16255) Sélectionne la ROM en lecture et commute la Bank 2. Autorise l'écriture sur la RAM si la commutation est effectuée deux fois (**Ex.** : LDA $C081, LDA $C081).

$C082 (- 16254) Sélectionne la ROM en lecture et commute la Bank 2. Protège la RAM contre l'écriture.

$C083 (- 16253) Sélectionne la RAM en lecture et commute la Bank 2. Autorise l'écriture sur la RAM si la commutation est effectuée deux fois.

$C088 (- 16248) Sélectionne la RAM en lecture et commute la Bank 1. Protège la RAM contre l'écriture.

$C089 (- 16247) Sélectionne la ROM en lecture et commute la Bank 1. Autorise l'écriture sur la RAM si la commutation est effectuée deux fois.

$C08A (- 16246) Sélectionne la ROM en lecture et commute la Bank 1. Protège contre l'écriture sur la RAM.

$C08B (- 16245) Sélectionne la RAM en lecture et commute la Bank 1. Autorise l'écriture sur la RAM si la commutation est effectuée deux fois.

TRUCS et ASTUCES

PROGRAMME : La Boîte aux idées (Tremplin Micro n°8)

QUESTION : Comment imprimer les lignes dans lesquelles se trouve le mot cherché ?

RÉPONSE : En ajoutant ces 3 lignes au programme. Pour obtenir l'impression, taper **CTRL-I** au lieu de l'option **2**.

```
156 IM = Ø: IF R$ = CHR$ (9) THEN R$ = "2":IM = 1
541 IF IM THEN PRINT D$"PR£1": PRINT : PRINT CHR$ (9)"96N":
    PRINT CHR$ (27)"E"; CHR$ (27)"LØ1Ø": PRINT
546 IF IM THEN PRINT : PRINT D$"PR£Ø": PRINT D$"PR£3": PRINT :
    GOTO 125
```

Attention ! Supprimer deux lignes de REM (19Ø et 2ØØ) pour que la longueur du programme ne dépasse pas 55ØØ.

Marcel COTTINI

LE 65C816

Nous publions, dans ce numéro, la suite et la fin de notre étude sur le 65C816, microprocesseur qui équipe l'APPLE *IIGS*.

3 LES REGISTRES DU MICROPROCESSEUR 65C816

Différentes fonctions du 65C816

Généralités

Le 65C816 (ou 65C802) est un microprocesseur aux possibilités très variées. C'est ainsi qu'il permet la gestion de registres 16 bits tout en ayant la capacité d'adresser 16 Mo de mémoire vive (compteur ordinal sur 24 bits). Une configuration matérielle intégrant ce type de processeur permet une programmation plus rapide, tout en gardant l'accès aux programmes existants pour le 6502. La sélection entre l'un ou l'autre mode (émulation ou native) s'effectue par l'adressage du bit E du registre d'état. Cette opération est réalisée par l'instruction XCE, échange du bit de retenue (C) et du bit d'émulation (E). C'est en somme un excellent produit pour la gamme des 16 bits grand public. Le bus de données est de 8 bits de large, tandis que les registres internes sont de 16 bits.

Description des fonctions du 65C816

Instructions relatives aux registres et décodage. Les instructions machine, suivant le mode d'adressage requis, sont traitées en codes machine, avant d'être véhiculées vers l'entrée du microprocesseur. Plusieurs instructions sont reliées entre elles et envoyées sur le bus de données par l'intermédiaire du registre concerné. Cette instruction est alors décodée suivant une référence temps (horloge du processeur) et une référence de procédure (circuits d'interruption du processeur).

Unité de contrôle temps (Timing Control Unit — TCU)

Le circuit horloge maintient en ligne chaque instruction et lui attribue un certain temps évalué en cycles machine. C'est ainsi que l'instruction RTS (retour de sous-programme) nécessite un temps de 6 cycles pour arriver à terme. Pour chaque instruction, le facteur temps (TCU) est nul au départ, le code machine est recherché puis exécuté suivant un certain nombre de cycles temps. Un ou plusieurs cycles machine sont requis pour chaque instruction machine. Chaque transfert de données est lié d'une part au décodage de l'instruction, puis à son exécution via un registre suivant une notion de temps (cycle machine).

Unité arithmétique et logique (Arithmetic and Logic Unit — ALU)

Toutes les opérations arithmétiques et logiques sont traitées par l'intermédiaire d'un module interne du microprocesseur, d'une taille de 16 bits (ALU — 16 bits). Lors d'opérations additionnelles de données, le module ALU calcule l'adresse effective pour les adressages des modes relatifs et indexés. Le résultat de cette opération sur ce type de données est rangé soit à une adresse mémoire spécifiée au préalable, soit dans un registre interne du processeur. La retenue, Carry Flag, sera positionnée suivant le résultat de l'opération effectuée par le module ALU. Elle pourra être soit négative ou positive, soit déborder. *(suite page 22)*

Registres internes du 65C816

Registres internes (Internal Registers)

Ils se différencient du 6502 par leur nombre, leur taille et leur organisation. On peut scinder la structure des registres en trois parties, chacune ayant une taille de 8 bits de large. La taille équivalente de chaque registre sera celle déterminée par le mode actuel (native : E = 0, ou émulation : E = 1).
Le 65C816 comporte un registre supplémentaire par rapport au 6502 (Direct Register). Il est du format 16 bits. Deux registres annexes, Program Bank Register et Data Bank Register, autorisent respectivement une extension du registre compteur ordinal (PC) et des registres d'index X et Y. Program Bank Register, associé au compteur ordinal (PC) permet alors de traiter une adresse mémoire sur 24 bits ($000000 à $FFFFFF). Data Bank Register, associé aux registres 16 bits X ou Y, permet de pointer une zone mémoire sur 24 bits ($000000 à $FFFFFF).

Registres de programmation du 65C816 (ou 65C802)

8 bits	8 bits		8 bits
Data Bank Reg. DBR			
	Registre X High XH	(X)	Registre X Low XL
	Registre Y High YH	(Y)	Registre Y Low YL
00	Pointeur de pile SH	(S)	Pointeur de pile SL
	Accumulateur High AH (B)	(C)	Accumulateur Low AL (A)
Program Bank Reg. PBR	Compteur ordinal PCH	(PC)	Compteur ordinal PCL
00	Registre Direct DH	(D)	Registre Direct DL
		(P)	Registre d'état

Détail du registre d'état P du microprocesseur :

Status Register (P)

		1	B				E
N	V	M	X	D	I	Z	C

1 Toujours à 1 si E = 1.

B Break : Remise à 0 de la pile après interruption si E = 1.

E Mode émulation : 0 = Native ou 1 = Emulation.

N	Négative Bit du signe	= 1 si résultat est négatif.
V	Overflow Débordement	= 1 si le résultat est celui attendu.
M	Mémory/Accumulator Select Sélection de la mémoire ou de A	= 1 si commuté en mode 8 bits. = 0 si commuté en mode 16 bits.
X	Index Reg. Select Sélection du registe d'index	= 1 si commuté en 8 bits. = 0 si commuté en 16 bits.
D	Décimal Mode Mode décimal	= 1 si commuté en mode décimal.
I	IRQ Disable Inhibition des interruptions	= 1 si interruption inhibée.
Z	Zéro Résultat nul	= 1 si le résultat est nul.
C	Carry Retenue	= 1 si le résultat est celui attendu.

Accumulateur (Accumulator — C (A + B))

L'accumulateur est un registre à usage général utilisé pour effectuer des opérations arithmétiques ou logiques : charger (LDA), ou sauvegarder (STA) une donnée ou comme opérante (ROL, ROR, etc.).
Mode native (E = 0). Lorsque le bit M du registre d'état, sélection de la mémoire (Memory Select) est nul (M = 0), l'accumulateur travaille sur des données de 16 bits de large.
Mode émulation (E = 1). Lorsque le bit M du registre d'état a comme valeur un (M = 1), l'accumulateur travaille sur des données de 8 bits de large.

Dans ce dernier cas, la partie poids fort de l'accumulateur (Acc. High — B) sera sauvegardée momentanément en liaison avec une instruction interne du processeur (SWAP Accumulator — instruction SWA).

En mode 8 bits (émulation — A) l'accumulateur est dénommé registre A. Il est alors compatible avec le mode du 6502. En mode 16 bits (native — C), c'est sous registre C qu'on le retrouve, avec comme octet de poids faible (A — Acc. Low) et comme octet de poids fort (B — Acc. High).

Registre page de données (Data Bank Register — DBR)

C'est un registre annexe d'une taille de 8 bits, utilisé en association soit avec le registre d'index X (16 bits), soit avec le registre d'index Y (16 bits), pour

adresser par multiplexage une page mémoire sur 24 bits. Dans ce cas, DBR contient l'octet de poids fort (8 bits) et les registres X ou Y les 16 bits les plus à droite (YH + YL pour le registre Y et XH + XL pour le registre X).
L'adressage des 16 Mo de la mémoire vive (RAM) disponible se fait d'une manière similaire au microprocesseur 8086, c'est-à-dire par fragmentation de la RAM en zones de données appelées "Bank" ou pages sur le 65C816. Chaque page (Bank) est formée d'un bloc de 64 Ko dont l'adresse partielle, codée sur 8 bits, se trouve dans le registre 8 bits Data Bank. Cette adresse partielle représente le poids fort de l'adresse complète sur 24 bits.
En mode native (E = 0), le registre de page de données (DB) de 8 bits de large, mémorise l'adresse par défaut de la page (bank address) pour le transfert de la zone mémoire. L'adressage sur 24 bits se compose alors des 16 bits de l'adresse effective (index X ou Y) et des 8 bits du registre page de données (DBR). Le contenu du registre DB est multiplexé avec la valeur de la donnée se trouvant à une adresse mémoire et a donc la possibilité de gérer simultanément une donnée et l'adresse de ladite donnée. Cette prouesse de la technique est réalisée durant la première phase du cycle de transfert de la donnée vers la mémoire centrale. Le registre page de données (Data Bank Register — DBR) est initialisé à la valeur zéro pendant la phase du Reset.

Registre adressage direct (Direct Register — D)

Le registre adressage direct 16 bits est pourvu d'une structure lui permettant d'adresser toutes les instructions du processeur, en mode adressage direct. L'adressage effectif de la page zéro est réalisé en ajoutant le deuxième octet de l'instruction au registre direct.
Le registre direct est adressable soit en mode 8 bits (émulation — DL), soit en mode 16 bits (native — D). C'est un faux 24 bits dont les 8 bits les plus à gauche sont toujours positionnés à zéro. Les 16 bits les plus à droite forment deux groupes. Les 8 bits les plus à gauche font partie de l'octet de poids fort (DH), les 8 bits les plus à droite représentant l'octet de poids faible (DL). Ce registre autorise un accès plus rapide à la mémoire. C'est en somme une extension de la page zéro du 6502. La page zéro peut être placée où l'on veut avec les microprocesseurs 65C816 et 65C802. Le registre direct (D) est initialisé à la valeur zéro pendant la phase du Reset.

Registres d'index X et Y (Index — X and Y)

Le microprocesseur 65C816 intègre deux registres d'index (X et Y) dont le rôle est double : une utilisation générale et une utilisation comme index d'une adresse effective de la mémoire par exemple. Dans ce dernier cas plusieurs modes d'adressages indexés sont offerts au programmeur. Les registres d'index X et Y sont utilisés lors de l'exécution d'une instruction relative à un adressage indexé. Le microprocesseur recherche alors le code de l'instruction et l'adresse de base, puis modifie l'adresse initiale en lui additionnant la valeur de l'index du registre concerné (X ou Y).
L'adressage indirect indexé comporte deux modes : l'adressage préindexé et l'adressage postindexé. Ils sont sélectionnés dans le mode native avec le bit E du registre d'état égal à zéro. Pour effectuer des opérations sur des adresses indexées sur 16 bits de large, il est nécessaire de positionner le bit X du registre d'état (Index Register Select) à la valeur zéro (X = 0). Dans le cas d'un calcul sur une adresse partielle de 8 bits de large, le bit X du registre d'état devra avoir comme valeur un (X = 1).
Le registre d'index X (X — Index Register X) est adressable soit en mode 8 bits (émulation — XL), soit en mode 16 bits (native — X). En mode 16 bits, il se compose d'un octet de poids fort (XH) et d'un octet de poids faible (XL). Le registre d'index Y (Y — Index Register Y) est adressable soit en mode 8 bits (émulation — YL), soit en mode 16 bits (native). Il se compose d'un octet de poids fort (YH) et d'un octet de poids faible (YL).

Registre d'état du processeur (Processor Status — P)

Le registre d'état du microprocesseur intègre 8 bits destinés à renseigner le système (et le programmeur) sur diverses situations (Status) lors des opérations en cours. Ces différents bits sont des drapeaux (Flags) de situation et de modes (status flag and mode flag).
La retenue (C), le bit du signe (N), le bit de débordement (V) et le bit du résultat nul (Z) sont des drapeaux (flags) de situation servant à effectuer un report sur le module de l'unité arithmétique et logique (ALU) du processeur. Les drapeaux de situation effectuent un test de branchement pour la poursuite éventuelle de l'instruction en cours d'exécution (si la condition posée est remplie). Le bit du mode décimal (D), le bit d'inhibition de l'interruption (I), le bit de sélection de la mémoire/accumulateur (M) et le bit de sélection du registre d'index (X) sont des drapeaux autorisant une sélection d'un mode déterminé. Ces bits flags sont utilisés pour annoncer au microprocesseur un changement d'opération dans l'exécution d'un programme en cours.
La sélection du mode émulation (E) et du Break (B) est accessible en positionnant certains bits du registre d'état du processeur. Ce sont en somme des bits fictifs dont l'état est conditionné par la valeur d'un ou de plusieurs bits réels du registre d'état.

(suite page 24)

Le processeur gère et modifie ces bits internes (E et B). Le mode émulation/native est sélectionné en permutant les bits de la retenue (C) et de l'émulation (E). Cette opération est effectuée grâce à une instruction machine (XCE) propre au microprocesseur (voir à cet effet la table de comparaison des différents mode (native : E = Ø ou émulation : E = 1)). Les bits M et X sont toujours positionnés à la valeur un (M/X = 1) en mode émulation (8 bits). Lorsqu'une interruption survient pendant le mode émulation, le bit B (Break flag) est sauvegardé sur la pile comme valeur du bit 4 (X) du registre d'état du microprocesseur.

Registre page programme (Program Bank Register — PBR)

Le registre page programme (Program Bank) est de 8 bits de large. Il contient l'adresse partielle (High Byte) lors de la recherche de chaque instruction au niveau d'un programme. Le registre PB mémorise l'adresse de la ligne où l'instruction du programme en cours d'exécution a été recherchée.

L'adresse de 24 bits de large consiste en une association de l'adresse 16 bits du compteur ordinal (instruction en cours d'exécution) avec l'adresse de poids fort sur 8 bits contenue dans le registre page programme (PBR). Le compteur ordinal (PC) contient les 16 bits les plus à droite de l'adresse de l'instruction en cours d'exécution. Le contenu du registre PB est multiplexé avec les données du programme et a donc la possibilité de gérer simultanément une instruction et l'adresse de la ligne où l'instruction en cours d'exécution a été recherchée. Cette prouesse de la technique est réalisée durant la première phase du cycle de lecture de la mémoire. C'est le compteur ordinal (PC) qui gère les instructions pas-à-pas (cycles machine) pendant la lecture des données du programme implanté en mémoire centrale. Le registre page de données (PB) est initialisé à la valeur zéro pendant la phase du Reset.

Comme pour le registre page de données (DB), l'adressage des 16 Mo de la mémoire vive (RAM) disponible se fait par fragmentation. C'est ainsi qu'un programme implanté en mémoire centrale, est vu par le processeur comme des pages (Bank). Ce sont des blocs de 64 Ko dont l'adresse partielle (8 bits les plus à gauche) est codée sur 8 bits. Cette adresse partielle représente le poids fort de l'adresse complète. Le registre PB est complémentaire au registre PC. Associé avec le registre PC comme octet de poids fort d'une adresse sur 24 bits, ils forment ensemble le pointeur de la ligne de l'instruction d'un programme en cours d'éxécution.

Compteur ordinal (Program Counter — PC)

Le registre du compteur ordinal gère une adresse sur 16 bits de large au fur et à mesure de l'exécution des instructions du microprocesseur. Il renseigne ce dernier, séquence par séquence, de l'adresse actuelle durant toute la phase d'exécution d'une instruction machine. Le registre est incrémenté au fur et à mesure lors de chaque cycle machine (Timing) de l'instruction en cours ou de l'opérante recherchée au niveau de la mémoire (programme).

Le compteur ordinal (PC) est adressable soit en mode 8 bits (émulation — PCL), soit en mode 16 bits (native — PC). Il se compose d'un octet de poids fort (PCH) et d'un octet de poids faible (PCL). Associé au registre PB, il permet de pointer une adresse sur 24 bits (adresses $000000 à $FFFFFF) à l'intérieur d'un programme. Dans ce cas, l'association des deux registres forme une adresse sur trois octets.

Pointeur de pile (Stack Pointer — S)

En mode native (E = Ø), le pointeur de pile est un registre de 16 bits de large et pointe la prochaine adresse disponible sauvegardée dans l'espace mémoire de la pile. A chaque fois qu'on empile, S est décrémenté et, à tout moment, S pointe vers le prochain emplacement libre de la pile.

L'opération empiler (Push) permet de placer un élément au sommet de la pile. L'opération dépiler (Pull) consiste en un transfert du contenu du sommet de la pile vers le registre déclaré.

Le pointeur de pile a deux fonctions essentielles : d'une part gérer des pointeurs pour mémoriser des adresses de retour d'un sous-programme (RTS ou GOSUB) ; d'autre part pointer vers une adresse après une interruption. Le pointeur de pile pointe simplement vers une adresse de retour de sous-programme ou suivant le niveau d'interruption rencontré. Le registre pointeur de pile (S) est adressable soit en mode 8 bits (émulation — SL), soit en mode 16 bits (native — S). C'est un faux 24 bits dont les 8 bits les plus à gauche sont toujours à zéro. Les 16 bits les plus à droite forment deux groupes de 8 bits. Les 8 bits les plus à gauche font partie de l'octet de poids fort (SH), les 8 bits les plus à droite représentent l'octet de poids faible (SL).

Registre pointeur de pile en mode native (E = 0)

Bits

S															
SH								SL							
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

S = Stack Pointer
SH = Stack High Byte
SL = Stack Low Byte

Les bits marqués par des x sont disponibles comme bits du pointeur de pile.

En mode émulation (E = 1), l'octet de poids fort du pointeur de pile (SH) est toujours égal à 1, ce qui le met au même niveau que le processeur 6502 : 8 bits disponibles (bits Ø-7) avec le bit 8 toujours à 1.

Registre pointeur de pile en mode émulation (E = 1)

S = Stack Pointer SH = Stack High Byte SL = Stack Low Byte

Les bits marqués par des x sont disponibles comme bits du pointeur de pile, le bit 8 étant alors à 1 tandis que les bits F à 9 sont nuls.

Bits

S															
SH								SL							
F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	1	x	x	x	x	x	x	x	x

4 CODES OPÉRATIONS DU MICROPROCESSEUR 65C816

Symboles et différents modes d'adressage

£	adressage immédiat
(a)	adressage absolu indirect
(a,x)	adressage absolu indexé indirect
(d)	adressage direct indirect
(d),y	adressage direct indirect indexé avec Y
(d,x)	adressage direct indexé indirect
(r,s),y	adressage indirect indexé relativement à la pile
a	adressage absolu
a,x	adressage absolu indexé avec X
a,y	adressage absolu indexé avec Y
A	opération effectuée sur l'accumulateur
al	adressage absolu long
al,x	adressage absolu indexé long
d	adressage direct
d,x	adressage direct indexé avec X
d,y	adressage direct indexé avec Y
i	adressage implicite
r	adressage relatif au compteur ordinal
r,s	adressage relatif à la pile
rl	adressage long relatif au compteur ordinal
s	adressage de la pile — empilement/dépilement
xya	adressage "block bank" source vers "bank" de destination
<d>	adressage long direct indirect
<d>,y	adressage long direct indirect indexé avec Y

ØØ	BRK	implicite : i
Ø1	ORA	direct indexé avec le registre X : d,x
Ø2	COP	implicite : i
Ø3	ORA	relatif à la pile : r,s
Ø4	TSB	direct : d ou page zéro : ØØ
Ø5	ORA	direct : d ou page zéro : ØØ
Ø6	ASL	direct : d ou page zéro : ØØ
Ø7	ORA	direct indirect long : <d>
Ø8	PHP	la pile (empilement/dépilement) : s
Ø9	ORA	adressage immédiat : £
ØA	ASL	accumulateur : A
ØB	PHD	la pile (empilement/dépilement) : s

0C	TSB	absolu : a
0D	ORA	absolu : a
0E	ASL	absolu : a
10	BPL	relatif au compteur ordinal : r
11	ORA	direct indirect indexé : (d),y
12	ORA	direct indirect : (d)
13	ORA	indirect indexé relativement à la pile : (r,s),y
14	TRB	direct : d
15	ORA	direct indexé avec X : d,x
16	ASL	direct indexé avec X : d,x
17	ORA	direct indirect long indexé :<d>,y
18	CLC	implicite : I
19	ORA	absolu indexé avec Y : a,y
1A	INC	accumulateur : A
1B	TCS	implicite : i
1C	TRB	absolu : a
1D	ORA	absolu indexé avec le registre X : a,x
1E	ASL	absolu indexé avec le registre X : a,x
1F	ORA	absolu long indexé avec X : al,x
20	JSR	absolu : a
21	AND	direct indexé indirect : (d,x)
22	JSL	absolu long : al
23	AND	relatif à la pile : r,s
24	BIT	direct : d, ou page zéro : 00
25	AND	direct : d, ou page zéro : 00
26	ROL	direct : d, ou page zéro : 00
27	AND	direct indirect long :<d>
28	PLP	la pile : s
29	AND	immédiat : £
2A	ROL	accumulateur : A
2B	PLD	la pile : s
2C	BIT	absolu : a
2D	AND	absolu : a
2E	ROL	absolu : a
2F	AND	absolu long : al
30	BMI	relatif au compteur ordinal : r
31	AND	direct indirect indexé : (d),y
32	AND	direct indirect : (d)
33	AND	indirect indexé relativement à la pile : (r,s),y
34	BIT	direct indexé avec le registre X : d,x
35	AND	direct indexé avec le registre X : d,x
36	ROL	direct indexé avec le registre X : d,x
37	AND	direct indirect long indexé :<d>,y
38	SEC	implicite : i
39	AND	absolu indexé avec le registre Y : a,y
3A	DEA	accumulateur : A
3B	TSC	implicite : i
3C	BIT	absolu indexé avec le registre X : a,x
3D	AND	absolu indexé avec le registre X : a,x
3E	ROL	absolu indexé avec le registre X : a,x
3F	AND	absolu long indexé avec le registre X : al,x
40	RTI	la pile : s
41	EOR	direct indexé indirect : (d,x)
42	WDC	implicite : i (utilisation future avec le 65C832)
43	EOR	relatif à la pile : r,s
44	MVP	block bank source vers banck de desintation (Move) : xya
45	EOR	direct : d, ou page zéro : 00
46	LSR	direct : d, ou page zéro : 00
47	EOR	direct indirect long :<d>
48	PHA	la pile : s
49	EOR	immédiat : £
4A	LSR	accumulateur : A
4B	PHK	la pile : s
4C	JMP	absolu : a
4D	EOR	absolu : a
4E	LSR	absolu : a
4F	EOR	absolu long : al
50	BVC	relatif au compteur ordinal : r
51	EOR	direct indirect indexé Y : (d),y
52	EOR	direct indirect long :<d>
53	EOR	indirect indexé relativement à la pile : (r,s),y
54	MVN	block bank source vers bank destination (Move) : xya
55	EOR	direct indexé avec X : d,x
56	LSR	direct indexé avec X : d,x
57	EOR	direct indirect long indexé :<d>,y
58	CLI	implicite : i
59	EOR	absolu indexé Y : a,y
5A	PHY	la pile : s
5B	TCD	implicite : i
5C	JMP	absolu long : al
5D	EOR	absolu indexé avec X : a,x
5E	LSR	absolu indexé avec X : a,x
5F	EOR	absolu long indexé avec X : al,x
60	RTS	la pile : s
61	ADC	direct indexé indirect : (d,x)
62	PER	la pile : s
63	ADC	relatif à la pile : r,s
64	STZ	direct : d, ou page zéro : 00
65	ADC	direct : d, ou page zéro : 00
66	ROR	direct : d, ou page zéro : 00
67	ADC	direct : indirect long :<d>
68	PLA	la pile : s
69	ADC	immédiat : £
6A	ROR	accumulateur : A
6B	RTL	la pile : s
6C	JMP	absolu indirect : (a)
6D	ADC	absolu : a
6E	ROR	absolu : a
6F	ADC	absolu long : al
70	BVS	relatif au compteur ordinal : r
71	ADC	direct indirect indexé : (d),y
72	ADC	direct : d
73	ADC	indirect indexé relativement à la pile : (r,s),y
74	STZ	direct indexé avec X : d,x
75	ADC	direct indexé avec X : d,x
76	ROR	direct indexé avec X : d,x
77	ADC	direct indirect long indexé :<d>,y
78	SEI	implicite : i

79	ADC	absolu indexé avec Y : a,y
7A	PLY	la pile : s
7B	TDC	implicite : i
7C	JMP	absolu indexé indirect : (a,x)
7D	ADC	absolu indexé avec X : a,x
7E	ROR	absolu indexé avec X : a,x
7F	ADC	absolu long indexé avec X : al,x
8Ø	BRA	relatif au compteur ordinal : r
81	STA	direct indexé indirect : (d,x)
82	BRL	relatif au compteur ordinal, long : rl
83	STA	relatif à la pile : r,s
84	STY	direct : d, ou page zéro : ØØ
85	STA	direct : d, ou page zéro : ØØ
86	STX	direct : d, ou page zéro : ØØ
87	STA	direct indirect long :<d>
88	DEY	implicite : i
89	BIT	immédiat : £
8A	TXA	implicite : i
8B	PHB	la pile : s
8C	STY	absolu : a
8D	STA	absolu : a
8E	STX	absolu : a
8F	STA	absolu long : al
9Ø	BCC	relatif au compteur ordinal : r
91	STA	direct indirect indexé avec Y : (d),y
92	STA	direct indirect : (d)
93	STA	indirect indexé relativement à la pile : (r,s),y
94	STY	direct indexé avec X : d,x
95	STA	direct indexé avec X : d,x
96	STX	direct indexé avec Y : d,y
97	STA	direct indirect indexé avec Y : (d),y
98	TYA	implicite : i
99	STA	absolu indexé avec Y : a,y
9A	TXS	implicite : i
9B	TXY	implicite : i
9C	STZ	absolu : a
9D	STA	absolu indexé avec X : a,x
9E	STZ	absolu indexé avec X : a,x
9F	STA	absolu long indexé avec X : al,x
AØ	LDY	immédiat : £
A1	LDA	direct indexé indriect : (d,x)
A2	LDX	immédiat : £
A3	LDA	relatif à la pile : r,s
A4	LDY	direct : d, ou page zéro : ØØ
A5	LDA	direct : d, ou page zéro : ØØ
A6	LDX	direct : d, ou page zéro : ØØ
A7	LDA	direct indirect long :<d>
A8	TAY	implicite : i
A9	LDA	immédiat : £
AA	TAX	implicite : i
AB	PLB	la pile : s
AC	LDY	absolu : a
AD	LDA	absolu : a
AE	LDX	absolu : a
AF	LDA	absolu long : al
BØ	BCS	relatif au compteur ordinal : r
B1	LDA	direct indirect indexé avec Y : (d),y
B2	LDA	direct indirect : (d)
B3	LDA	indirect indexé relativement à la pile : (r,s),y
B4	LDY	direct indexé avec X : d,x
B5	LDA	direct indexé avec X : d,x
B6	LDX	direct indexé avec Y : d,y
B7	LDA	direct indirect long indexé avec Y :<d>,y
B8	CLV	implicite : i
B9	LDA	absolu indexé avec Y : a,y
BA	TSX	implicite : i
BB	TYX	implicite : i
BC	LDY	absolu indexé avec X : a,x
BD	LDA	absolu indexé avec X : a,x
BE	LDX	absolu indexé avec Y : a,y
BF	LDA	absolu long indexé avec X : al,x
CØ	CPY	immédiat : £
C1	CMP	direct indexé indirect : (d,x)
C2	REP	immédiat : £
C3	CMP	relatif à la pile : r,s
C4	CPY	direct : d, ou page zéro : ØØ
C5	CMP	direct : d, ou page zéro : ØØ
C6	DEC	direct : d, ou page zéro : ØØ
C7	CMP	direct indirect long :<d>
C8	INY	implicite : i
C9	CMP	immédiat : £
CA	DEX	implicite : i
CB	WAI	implicite : i
CC	CPY	absolu : a
CD	CMP	absolu : a
CE	DEC	absolu : a
CF	CMP	absolu long : al
DØ	BNE	relatif au compteur ordinal : r
D1	CMP	direct indirect indexé avec Y : (d),y
D2	CMP	direct indirect : (d)
D3	CMP	indirect indexé relativement à la pile : (r,s),y
D4	PEI	la pile : s
D5	CMP	direct indexé avec X : d,x
D6	DEC	direct indexé avec X : d,x
D7	CMP	direct indirect long indexé avec Y :<d>,y
D8	CLD	implicite : i
D9	CMP	absolu indexé avec Y : a,y
DA	PHX	la pile : s
DB	STP	implicite : i
DC	JML	absolu indirect : (a)
DD	CMP	absolu indexé avec X : a,x
DE	DEC	absolu indexé avec X : a,x
DF	CMP	absolu long indexé avec X : al,x
EØ	CPX	immédiat : £
E1	SEC	direct indexé indirect : (d,x)
E2	SEP	immédiat : £
E3	SBC	relatif à la pile : r,s
E4	CPX	direct : d, ou page zéro : ØØ
E5	SEC	direct : d, ou page zéro : ØØ
E6	INC	direct : d, ou page zéro : ØØ

E7	SBC	direct indirect long : <d>
E8	INX	implicite : i
E9	SEC	immédiat : £
EA	NOP	implicite : i
EB	XBA	implicite : i
EC	CPX	absolu : a
ED	SEC	absolu : a
EE	INC	absolu : a
EF	SBC	absolu long : al
FØ	BEQ	relatif au compteur ordinal : r
F1	SEC	direct indirect indexé avec Y : (d),y
F2	SBC	direct indirect : (d)
F3	SBC	indirect indexé relativement à la pile : (r,s),y
F4	PEA	la pile : s
F5	SEC	direct indexé avec X : (d),x
F6	INC	direct indexé avec X : (d),x
F7	SBC	direct indirect long indexé avec Y : <d>,y
F8	SED	implicite : i
F9	SEC	absolu indexé avec Y : a,y
FA	PLX	la pile : s
FB	XCE	implicite : i
FC	JSR	absolu indexé indirect : (a,x)
FD	SBC	absolu indexé avec X : a,x
FE	INC	absolu indexé avec X : a,x
FF	SBC	absolu long indexé avec X :

5 ADRESSAGE SOMMAIRE DU 65C816

	Modes d'adressage	6502		65C816	
		Cycles	Bytes	Cycles	Bytes
1.	Immédiat	2	2	2 (3)	2 (3)
2.	Absolu	4 (5)	3	4 (3 + 5)	3
3.	Absolu long	—	—	5 (3)	4
4.	Direct	3 (5)	2	3 (3 + 4 + 5)	2
5.	Accumulateur	2	1	2	1
6.	Implicite	2	1	2	1
7.	Direct indirect indexé	5 (1)	2	5 (1 + 3 + 4)	2
8.	Direct indirect indexé long	—	—	6 (3 + 4)	2
9.	Direct indexé indirect	6	2	6 (3 + 4)	2
10.	Direct indexé avec X	4 (5)	2	4 (3 + 4 + 5)	2
11.	Direct indexé avec Y	4	2	4 (3 + 4)	2
12.	Absolu indexé avec X	4 (1 + 5)	3	4 (1 + 3 + 5)	3
13.	Absolu long indexé	—	—	5 (3)	4
14.	Absolu indexé avec Y	4 (1)	3	4 (1 + 3)	3
15.	Relatif	2 (1 + 2)	2	2 (2)	2
16.	Relatif long	—	—	3 (2)	3
17.	Absolu indirect (JUMP)	5	3	5	3
18.	Direct indirect	—	—	5 (3 + 4)	2
19.	Direct indirect long	—	—	6 (3 + 4)	2
20.	Absolu indexé indirect	—	—	6	3
21.	Pile	3-7	1-3	3-8	1-4
22.	Relatif à la pile	—	—	4 (3)	2
23.	Indirect indexé relativement à la pile	—	—	7 (3)	2
24.	Déplacement d'une page de données (X,Y,C) — source/destination/longueur	—	—	7	3

Légende des indices du tableau :

1 Ajouter 1 cycle si une limite de page mémoire est dépassée (bytes ou expression désignant une adresse).
2 Ajouter 1 cycle si le branchement est réalisé.
3 Ajouter 1 cycle et 1 byte pour l'adressage immédiat. Règle valable pour des opérations réalisées sur 16 bits avec M = Ø et X = Ø (M et X = bits du registre d'état).
4 Ajouter 1 cycle si le résultat est différent de zéro : Registre direct, octet de faible poids (DL) différent de zéro.
5 Ajouter 2 cycles si le bit M = 1 et 3 cycles si le bit M = Ø (M = bit du registre d'état). Règle valable pour les routines utilisant les fonctions Read/Modify/Write. ■

UTILITAIRE | ProDOS | SOURIS

François GALLET

HORLOGE

Fonctionne aussi sur l'APPLE IIGS

Voici un utilitaire dont une première approche avait été réalisée par Claude AUBRY dans le numéro 5 de *Tremplin Micro* et qui, j'en suis certain, rendra un service inestimable aux heureux possesseurs de l'Apple *IIc* ou de l'Apple *IIe* avec 65CØ2 et carte souris.

Que fait donc ce programme ?

Il vous permettra d'avoir en permanence la date et l'heure à l'écran (en 8Ø colonnes), ainsi que la possibilité d'horodater vos œuvres. De plus, vous pourrez à tout moment, modifier ces paramètres, en utilisant simplement l'Ampersand.

Enfin, si vous travaillez relativement tard et qu'un jour nouveau vous surprenne, vous n'aurez pas à vous inquiéter du changement de date, car celui-ci se fera automatiquement, même s'il faut pour cela changer d'année. Ainsi, vos fichiers seront correctement horodatés.

Comment fonctionne-t-il ?

Ce programme, utilisable sous ProDOS, prend environ 1,4 K de mémoire et se loge à partir de l'adresse $9ØØØ, juste sous les buffers CAT.E/S débutant en $96ØØ. HIMEM a été repositionné à $9ØØØ, ce qui place les buffers d'ouverture de fichiers sous **HORLOGE** (en protégeant ainsi la routine et en vous permettant de disposer d'environ 6Ø octets pour y apporter d'éventuelles modifications).

L'horloge proprement dite débute à l'adresse $939Ø par un SEI, qui inhibe toute nouvelle interruption de type IRQ, ceci de manière à pouvoir initaliser le système. On teste bien sûr la présence de la carte SOURIS, et on place l'adresse de saut effective en $3FE-$3FF, adresse qui sera celle du comptage et du traitement, débutant à $941D.

Pour vous permettre d'apprécier cet utilitaire, voici un petit exercice :

- Tapez **RUN HORLOGE.BAS** puis (Return).
- Tapez **&** puis (Return)

Un cadre doit apparaître au milieu de l'écran. Indiquez la date puis répondez par :

- (**N**)on, en cas d'erreur.
- (**O**)ui, pour continuer.

Dans le cas d'une réponse affirmative, la date s'inscrit en haut à gauche, dans le cadre prévu à cet effet. Puis, la mention **Mise à l'heure de l'horloge** apparaît. Réagissez de la même façon que précédemment, mais en indiquant l'heure actuelle, puis répondez à la question :

Etes-vous d'accord ?

Si vous tapez (**O**)ui, l'heure se met à jour, le cadre disparaît et vous retrouvez l'écran tel qu'il était avant cet exercice, à la différence que vous disposez de la date courante et d'une horloge.

Si vous tapez (**N**)on, le cadre reste, mais vous retournez à la mention concernant la date. Le fait de taper sur (FLÈCHE-BASSE) vous permet de revenir à la partie **Mise à l'heure de l'horloge**. Dans tous les cas, vous pouvez abandonner en tapant sur la touche **Esc**.

Deuxième exercice :

- Faites **CATALOG**
- Mettez la date et l'heure à jour, par exemple :

  JJ/MM//AA ——» 31/12/85
  HH:MM SS ——» 18:3Ø ØØ

(Suite page 30)

Le cadre a occulté une partie de l'écran, mais vous retrouverez ce dernier dès que vous aurez terminé la mise à jour.

- Tapez ce programme Basic :
  10 REM *** NUL ***
- Faites **SAVE NUL**, puis **CATALOG**.
  Que constatez-vous ?

Votre programme est maintenant horodaté. Vous remarquez une date identique dans les deux rubriques **CREATION** et **MODIFICATION**. Ceci est dû au fait que votre programme vient d'être créé.

Dernier exercice pour les couche-tard.

- Faites **&**
- Modifiez seulement l'heure en mettant 23:59 50

Dès que l'heure atteint 24:00 00, elle se remet à zéro (RAZ), et — ô joie ! — la date passe du 31/12/85 au 01/01/86. Essayez de refaire **SAVE NUL** puis **CATALOG**. Que constatez-vous ?

Voilà ! vous savez tout. Rien ne vous empêchera, pour vos propres besoins, d'apporter une touche personnelle à ce programme.

J'allais oublier : il existe quelques restrictions à l'emploi de cet utilitaire :

- Un **Reset** le perturbera définitivement. Si cela arrive, faites **PR£3**, sauvegardez votre programme s'il y en a un, tapez **RUN HORLOGE.BAS**, puis rechargez le fruit de vos cogitations.
- Si vous voulez effacer l'écran, faites **HOME** ou la séquence **Esc Shift-à**, et non pas **PR£3**.
- Si vous déplacez HIMEM, il faut qu'il soit inférieur à l'adresse $9000, car c'est là que se loge **HORLOGE.**
- **HORLOGE** n'a pas de calendrier perpétuel et donc, ne gère pas le 29 février.
- Si vous exécutez un programme en page texte, assurez-vous de placer des tabulations de ligne supérieures à 2.
- N'essayez pas d'exécuter un programme utilisant les pages graphiques, mais rien ne vous empêche de l'écrire. Si vous voulez l'horodater, faites la séquence suivante :
  - — Ne chargez pas **HORLOGE**.
  - — Ecrivez et testez votre programme, puis sauvegardez-le.
  - — Chargez **HORLOGE** par **RUN HORLOGE.BAS** et mettez à jour la date et l'heure.
  - — Rechargez votre programme.
  - — Faites **DELETE programme** (permet d'effacer le SANS DATE dans la rubrique CREATION).
  - — Sauvegardez à nouveau votre programme.

Après ces quelques remarques, je n'ai plus qu'une chose à vous souhaiter : bon travail !

P.S. : Ce programme est protégé contre toute entrée non conforme : erreurs de date du type 35/03/86 ou 24/24/86 ou encore 33/00/86 voire 31/11/86, ou une heure du type 25:00 00 ou 12:60 00 ou 12:00 60 etc. ...

HORLOGE.BAS

```
10  REM *** HORLOGE.BAS ***
20  D$ = CHR$ (4) : REM CTRL-D
30  POKE 1014,132: POKE 1015,144: REM $3FE-3FF ——» Saut en $9084
40  PRINT D$ ; "BLOAD HORLOGE.C"
50  CALL 37776
60  HOME : PRINT TAB (2) ; "Date " ; TAB (9) ; "Aucune" ; TAB (61) ; "Horloge
70  PRINT CHR$ (27) : INVERSE : FOR C=0 TO 79 : PRINT CHR$ (76) ; : NEXT
80  VTAB 1 : HTAB 1 : PRINT CHR$ (95) ; : HTAB 18 : PRINT CHR$ (90) ; : HTAB 59 : PRINT
    CHR$ (90) ; : HTAB 80 : PRINT CHR$ (90) : NORMAL : PRINT CHR$ (24)
90  POKE 34,2 : VTAB 2 : HTAB 1 : NEW
```

C-MONITOR

Le programme source se trouve sur la disquette de *Tremplin Micro* n°12.

```
9000: CD E9 F3 E5 A0 C0 A0 EC A7 E8 E5 F5 F2 E5 A0 E4  013E
9010: E5 A0 EC A7 E8 EF F2 EC EF E7 E5 00 C8 C8 BA CD  BE9F
9020: CD A0 D3 D3 00 A0 A0 CD E9 F3 E5 A0 C0 A0 EA EF  E1BA
9030: F5 F2 A0 E4 E5 A0 EC E1 A0 E4 E1 F4 E5 A0 A0 A0  AFDB
9040: 00 CA CA AF CD CD AF C1 C1 00 C5 F4 E5 F3 AD F6  6D42
9050: EF F5 F3 A0 E4 A7 E1 E3 E3 EF F2 E4 A0 A8 CF AF  F834
9060: CE A9 A0 00 B0 B0 A0 B0 B0 A0 B0 B0 00 B0 B0 A0  7077
9070: B0 B0 A0 B0 B0 00 B0 00 1F 1C 1F 1E 1F 1E 1F 1F  5603
9080: 1E 1F 1E 1F 48 DA 5A 38 AD 54 C0 20 6E 93 A9 40  DDF9
9090: 85 43 20 11 C3 AD 55 C0 20 6E 93 A9 44 85 43 20  A474
90A0: 11 C3 AD 54 C0 20 C9 91 A9 09 20 24 FC A9 1B 85  F24A
90B0: 24 64 32 A9 25 A0 90 20 3A DB A9 FF 85 32 A9 0C  9A01
90C0: 20 24 FC A9 24 85 24 A9 41 A0 90 20 3A DB 20 6D  D892
90D0: 92 20 C5 92 A0 03 20 4E 93 AC 76 90 C0 01 30 19  0569
90E0: C0 0D 10 15 B9 77 90 48 A0 00 20 4E 93 AD 76 90  3E4E
90F0: F0 07 68 38 ED 76 90 10 08 78 20 3A FF 58 4C A8  98BF
9100: 90 20 D5 92 20 32 93 A9 00 20 24 FC A9 08 85 24  C83F
9110: A9 6D A0 90 20 3A DB 20 03 93 20 C9 91 A9 09 20  FB7D
9120: 24 FC A9 1B 85 24 64 32 A9 00 A0 90 20 3A DB A9  BCDA
9130: FF 85 32 A9 0C 20 24 FC A9 24 85 24 A9 1C A0 90  8316
9140: 20 3A DB 20 6D 92 20 C5 92 A0 00 20 4E 93 AD 76  998F
9150: 90 C9 18 10 18 A0 03 20 4E 93 AD 76 90 C9 3C 10  9805
9160: 0C A0 06 20 4E 93 AD 76 90 C9 3C 30 08 78 20 3A  2375
9170: FF 58 4C 1D 91 20 D5 92 A0 07 B9 64 90 09 B0 99  DF7E
9180: 94 95 88 10 F5 A0 02 A9 BA 99 94 95 A0 05 A9 A0  DF6B
9190: 99 94 95 18 AD 54 C0 20 81 93 A9 40 85 3D A9 43  4E66
91A0: 85 3F 20 11 C3 18 AD 55 C0 20 81 93 A9 44 85 3D  9475
91B0: A9 47 85 3F 20 11 C3 A9 17 85 25 20 24 FC A9 01  B5FC
91C0: 85 24 20 F0 FD 7A FA 68 60 A9 1B 20 F0 FD A9 4C  3FB8
91D0: 85 19 A9 08 85 25 20 2B 92 A9 10 85 25 20 2B 92  B316
91E0: A9 5A 85 19 A9 19 85 1D 20 37 92 A9 5F 85 19 A9  7A3D
91F0: 37 85 1D 20 37 92 A5 20 48 A5 21 48 A5 22 48 A5  6E91
9200: 23 48 A6 1C 86 20 A5 1B 38 E5 1C 85 21 A4 1F C8  D3FD
9210: 84 22 A4 1E 84 23 20 58 FC 68 85 23 68 85 22 68  C30A
9220: 85 21 68 85 20 A9 18 20 F0 FD 60 A9 1A 85 1C A9  F6EE
9230: 37 85 1B 20 43 92 60 A9 08 85 1F A9 10 85 1E 20  65FD
9240: 57 92 60 A6 1C A5 25 20 22 FC 86 24 A5 19 20 F0  B78B
9250: FD E8 E4 1B 30 F4 60 A4 1F 84 25 20 22 FC A5 1D  9AD4
9260: 85 24 A5 19 20 F0 FD C8 C4 1E 30 ED 60 A0 00 A9  D1E4
9270: 0C 85 25 20 24 FC A2 24 86 24 DA 5A 20 0C FD C9  9E8C
9280: 8A D0 07 7A FA 68 68 4C 1A 91 C9 8B D0 07 7A FA  1A3B
9290: 68 68 4C A5 90 C9 9B D0 07 7A FA 68 68 4C 93 91  A240
92A0: C9 B0 30 D8 C9 BA 10 D4 7A 99 64 90 20 F0 FD C8  92C4
92B0: FA E8 C0 02 F0 0A C0 05 F0 06 C0 08 F0 06 80 B8  B24F
92C0: C8 E8 80 B4 60 A0 00 B9 64 90 49 B0 99 64 90 C8  11DF
92D0: C0 08 D0 F3 60 A9 0E 85 25 20 24 FC A0 1B 84 24  FCEF
92E0: A9 4A A0 90 20 3A DB A9 0E 85 25 20 24 FC A2 35  C4D0
92F0: 86 24 20 0C FD C9 CF F0 09 C9 CE D0 F5 68 68 4C  48DC
9300: A5 90 60 A0 06 20 4E 93 AD 76 90 0A 8D 91 BF 20  11F6
9310: 4E 93 AD 76 90 0A 0A 0A 0A 0A 8D 90 BF AD 91 BF  F49F
9320: 69 00 8D 91 BF 20 4E 93 AD 76 90 0D 90 BF 8D 90  E873
9330: BF 60 A0 00 B9 64 90 09 B0 99 6D 90 C8 C0 08 D0  521B
9340: F3 A0 02 A9 AF 99 6D 90 A0 05 99 6D 90 60 A9 00  5AC7
```

(SUITE ET FIN PAGE 32)

HORLOGE • C

(SUITE ET FIN DE LA PAGE 31)

```
9350: 8D 76 90 A2 0A 18 AD 76 90 79 64 90 8D 76 90 CA  27D4
9360: D0 F4 C8 79 64 90 88 88 88 88 8D 76 90 60 A9 28  3CDD
9370: 85 42 85 3C A9 04 85 3D A9 D0 85 3E A9 07 85 3F  BCA7
9380: 60 A9 28 85 42 85 3C A9 04 85 43 A9 D0 85 3E 60  34CA
9390: 78 20 C5 93 A9 1D A2 94 8D FE 03 8E FF 03 A0 19  6BC3
93A0: 20 AC 93 A0 12 A9 09 20 AC 93 58 60 48 B1 06 AE  F287
93B0: C0 93 AC C1 93 8D C3 93 8E C4 93 68 20 C2 93 60  4B58
93C0: 00 00 4C 00 00 A2 07 A9 00 85 06 A9 C8 85 07 C6  29EC
93D0: 07 CA 30 20 A0 0C B1 06 C9 20 D0 F3 A0 FB B1 06  8082
93E0: C9 D6 D0 EB A5 07 8D C4 93 8D C0 93 0A 0A 0A 0A  84F2
93F0: 8D C1 93 60 20 58 FC 20 3A FF A9 0C 85 25 20 24  B2B1
9400: FC A9 14 85 24 A9 9C A0 95 20 3A DB A9 0C 85 25  8A70
9410: 20 24 FC A9 3A 85 24 20 0C FD 4C 00 C6 78 48 DA  A9A1
9420: 5A A0 13 20 AC 93 EE 85 95 AD 85 95 C9 3B D0 48  6C57
9430: A9 00 8D 85 95 A2 07 FE 94 95 BD 94 95 C9 BA 90  EA19
9440: 19 A9 B0 9D 94 95 CA FE 94 95 BD 94 95 C9 B6 D0  B45E
9450: 09 A9 B0 9D 94 95 CA CA 10 DD AD 94 95 C9 B2 D0  01CA
9460: 17 AD 95 95 C9 B4 D0 10 A0 07 B9 8B 95 99 94 95  9E8D
9470: 88 C0 00 10 F5 20 DF 94 A2 03 A0 07 AD 54 C0 20  3A0D
9480: B8 94 88 AD 55 C0 20 B8 94 88 CA 10 EF AD 54 C0  E314
9490: A0 00 B9 94 95 49 B0 99 86 95 C8 C0 05 D0 F3 A0  4E1F
94A0: 03 20 BF 94 AD 93 95 8D 92 BF 20 BF 94 AD 93 95  C071
94B0: 8D 93 BF 7A FA 68 58 40 B9 94 95 9D 23 04 60 A9  6F02
94C0: 00 8D 93 95 A2 0A 18 AD 93 95 79 86 95 8D 93 95  B297
94D0: CA D0 F4 C8 79 86 95 88 88 88 88 8D 93 95 60 A0  A5BF
94E0: 07 B9 6D 90 49 B0 99 64 90 88 10 F5 A0 03 20 4E  04E1
94F0: 93 AC 76 90 B9 77 90 48 A0 00 20 4E 93 EE 76 90  7EE2
9500: 68 38 ED 76 90 30 03 4C 44 95 A0 03 20 4E 93 EE  3F7D
9510: 76 90 AD 76 90 C9 0D 10 03 4C 3A 95 A0 06 20 4E  C1D1
9520: 93 EE 76 90 AD 76 90 C9 64 30 05 A9 00 8D 76 90  B2D8
9530: A0 07 20 6C 95 A9 01 8D 76 90 A0 04 20 6C 95 A9  7273
9540: 01 8D 76 90 A0 01 20 6C 95 20 32 93 A9 00 20 24  F428
9550: FC A9 08 85 24 A9 6D A0 90 20 3A DB 20 03 93 A9  F630
9560: 17 20 24 FC A9 01 85 24 20 F0 FD 60 A2 00 AD 76  8CDC
9570: 90 38 E9 0A 30 03 E8 10 F8 18 69 0A 99 64 90 88  7D7E
9580: 8A 99 64 90 60 00 B0 B0 A0 B0 B0 B0 B0 BA B0 B0  9A51
9590: A0 B0 B0 B0 B0 B0 BA B0 B0 A0 B0 B0 D0 E1 F3 A0  A26E
95A0: E4 E5 A0 E3 E1 F2 F4 E5 A0 D3 EF F5 F2 E9 F3 A0  65BD
95B0: AD AD BE A0 D2 E5 E4 FB ED E1 F2 F2 E1 E7 E5 A0  954D
95C0: A1 00    C9A1
```

Pour terminer : **BSAVE HORLOGE.C,A$9000,L$5C2**

UTILITAIRE | DOS | ProDOS

Guy HACHETTE

Fonctionne aussi sur l'APPLE IIGS

CRÉEZ vos caractères graphiques

Si vous avez déjà essayé de concevoir vos propres caractères graphiques, vous avez pu constater qu'il s'agit d'une œuvre de longue haleine — certes captivante, mais non moins fastidieuse. C'est pour vous que j'ai mis au point cette énième version d'un utilitaire de création de caractères. Son utilisation est simple, mais avant d'en arriver là, je vous propose de revenir sur quelques notions préliminaires indispensables.

La représentation d'un caractère

Ø	Ø	Ø	1	1	1	1	Ø	$1E
Ø	Ø	1	Ø	Ø	Ø	1	Ø	$22
Ø	Ø	1	Ø	Ø	Ø	1	Ø	$22
Ø	Ø	Ø	1	1	1	1	Ø	$1E
Ø	Ø	1	Ø	Ø	Ø	1	Ø	$22
Ø	Ø	1	Ø	Ø	Ø	1	Ø	$22
Ø	Ø	Ø	1	1	1	1	Ø	$1E
Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	$ØØ

Chargez un set (ou police) classique de caractères ASCII à l'adresse $4ØØØ (on peut le trouver sur les disquettes TOOL KIT, mais aussi sur bon nombre de disquettes françaises et américaines... et vous en avez un exemple plus loin). Tapez **CALL - 151**, puis **4110.4117** et l'indispensable **RETURN**. Votre écran vous renverra probablement la ligne suivante :

4110 - 1E 22 22 1E 22 22 1E ØØ

Ce sont les huit octets capables d'afficher un **B** en mode graphique. En les représentant sous leur forme binaire, vous obtenez le dessin matérialisé ci-contre. Il s'agit bien d'un **B**... mais regardé dans une glace. Contrariant non ? D'où la raison d'être d'un utilitaire qui vous autorisera à tracer normalement chacun de vos caractères !

Le programme travaille pour vous !

UNE MATRICE 7 x 8 Vous allez vous contenter d'une grille de 7 cases de large (on négligera le 8e bit, réservé à la couleur, et qui sera automatiquement mis à Ø) sur 8 de haut (8 octets). A l'écran, le tracé sera matérialisé par des O majuscules et les blancs par des accents circonflexes. Compatibilité oblige : je n'ai fait que quelques clins d'œil au langage machine et je n'ai pas utilisé les caractères souris disponibles sur les dernières machines, du moins dans cette version (un programme plus élaboré figurera dans *ROUTINES LM pour 65CØ2 et 65Ø2*, ouvrage à paraître au début de janvier).

O	O	O	O	O		
O	O			O	O	
O	O			O	O	
O	O	O	O	O		
O	O			O	O	
O	O			O	O	
O	O	O	O	O		

CARACTÈRES POUR LA COULEUR Si vous désirez créer une police capable de donner de bons résultats sur un écran couleur, contentez-vous de doubler tous les points horizontaux isolés, comme vous l'indique clairement la partie en couleur du **B** matérialisé ci-contre. De cette manière, vos caractères seront forcément blancs (deux points positifs placés côte à côte produisent du blanc, quelle que soit leur couleur). Dans cet exemple, les codes du **B** deviennent : **1F 33 33 1F 33 33 1F ØØ**

(Suite page 34)

CRÉEZ VOS CARACTÈRES GRAPHIQUES (suite)

ATTENTION ! (80 colonnes)
Ce programme nécessite la présence d'une carte 80 colonnes classique. La touche ESCAPE ne permettra pas de s'évader (voir la fin du tableau ci-contre) si l'on dispose d'un Apple ancien modèle.

LOMEM Nous n'avons pas prévu de LOMEM, mais il peut être nécessaire (utilisation de longue durée) de fixer LOMEM au-dessus de la routine LM d'affichage.
(LOMEM : 17400, par exemple).

LONGUEUR Une seule longueur pour ce type de police de caractères : 768 ($300). Si vous désirez créer des polices spéciales, ne respectant pas l'ordre établi (1 = espace, 2 = point d'exclamation, etc.), il vous suffira de multiplier le nombre de caractères créés par 8 pour connaître la longueur réelle de votre set.

MODE AUTOMATIQUE
Attention ! si vous chargez une police existante, vous passez en mode automatique. Autrement dit, dès que vous avez envoyé un caractère (par RETURN et Oui... la première lettre suffit), le suivant s'affiche. Comment en obtenir un autre ? En tapant CTRL-A... puis le code ASCII, tout simplement. Un seul moyen pour sortir du mode auto : ESCAPE et FIN DE PROGRAMME. Une astuce pour redémarrer sans effacer la police en mémoire : GOTO 1000.

Et maintenant, place à la création d'un jeu original de caractères, propre à faire votre bonheur et — qui sait ? — celui d'autres Lectrices et Lecteurs de Tremplin Micro !

RAPPEL DES COMMANDES

La grille est matérialisée par les (accents circonflexes). On s'y déplace en utilisant les flèches de l'Apple.

- **TAB** ou **BARRE d'espacement** permet de matérialiser un point et **DEL**ete de le supprimer.
- **V** crée une colonne de points à partir du curseur (sauf sur la dernière ligne, peu utilisée) et petit **v** supprime les points d'une colonne.
- De la même manière, **H** crée une rangée de points à partir du curseur et **h** la supprime.
- **CTRL-E** efface la grille.
- **CTRL-G** affiche la fonte actuelle en mode graphique (en HGR).
- **CTRL-W** déplace tous les points d'une case vers la gauche.
- **CTRL-A** pour appeler un caractère dont on doit fournir le code ASCII (police normale).
- **CTRL-R** pour remplacer un caractère par celui que l'on vient de tracer (il faudra ensuite taper un RETURN).
- **CTRL-P** pour charger une police existante.
- **RETURN** pour envoyer le caractère tracé.
- **ESCAPE** pour terminer à n'importe quel moment. Si l'Apple n'accepte pas ESCAPE en 80 colonnes, taper **CTRL-A** puis 127 (dernier caractère). Envoyer par RETURN et l'on entre dans la phase finale.
- Pour ne pas mémoriser la police, fournir "0" comme titre !

COMMENT AFFICHER SIMPLEMENT LE CARACTÈRE D'UNE POLICE EN HGR ?

Chargez la police à l'adresse $4000, puis essayez cette courte démonstration (pour arrêter : **CTRL-C**) :

```
10  HGR : HCOLOR = 3: HOME
15  FOR I = 768 TO 785: READ R: POKE I,R: NEXT : DATA 32,
    245,230,138,160,0,10,162,6,10,148,249,54,249,202,16,
    248,96
20  VTAB 22: INPUT "X,Y,CARACTÈRE ";X,Y,C$
25  ADRESSE = 16384 + ( ASC (C$) - 32) * 8
30  FOR F = 0 TO 7: CALL 768, PEEK (AD + F)
35  FOR L = 0 TO 6: IF PEEK (249 + L) = 1 THEN HPLOT X + L,
    Y + F
40  NEXT : NEXT : GOTO 20
```

■

POLICE.CRÉER 1

```
100 TEXT : PRINT  CHR$ (4)"PR£3": PRINT : HOME                    EDEA
105 ONERR  GOTO 160                                               A6E7
110 PRINT  CHR$ (4)"BLOAD POLICE.LM": PRINT  CHR$
    (4)"BLOAD POLICE.HGR"                                         3560
115 CALL 819: REM $4000 COS 42FF à zéro                           422E
120 PRINT  TAB( 20)"ctOCREATION D'UNE POLICE GRAPH
    IQUE DE CARACTERESctN"                                        7E76
125 POKE 6,0: POKE 7,64: REM $4000, adresse de la
    fonte à créer                                                 A60B
130 GOSUB 330                                                     F046
135 GOSUB 305                                                     0148
140 :                                                             003A
145 REM *** ctOCREATION DE LA POLICEctN ****
150 :                                                             003A
155 C = C + 1                                                     C74F
160 POKE 1403,0: VTAB 3: CALL  - 868: PRINT "ctOCA
    RACTEREctN";" NUMERO "C" - ASC "C + 31": " CHR
    $ (C + 31)                                                    A734
165 V = 10:H = 1:R$ = ""                                          42CE
170 IF H > 13 THEN H = 13: CALL  - 198                            0799
175 IF H < 1 THEN H = 1: CALL  - 198                              2B35
180 IF V < 10 THEN V = 10: CALL  - 198                            10B1
185 IF V > 17 THEN V = 17: CALL  - 198                            CEBD
190 VTAB V: HTAB H: PRINT R$: VTAB V: HTAB H: GET
    R$: PRINT                                                     CB26
195 R =  ASC (R$):R$ = ""                                         3593
200 IF R = 1 THEN 410                                             4959
205 IF R = 16 THEN 490                                            2097
210 IF R = 7 THEN A =  PEEK (6):B =  PEEK (7): VTA
    B 22: CALL  - 868: CALL 17152: PRINT : CALL  -
    198: GET R$: PRINT : POKE 6,A: POKE 7,B: POKE
    49233,0: GOTO 165                                             39BF
215 IF R = 86 THEN  FOR I = V TO 16: HTAB H: VTAB
    I: PRINT "O": NEXT : GOTO 190                                 F618
220 IF R = 72 THEN  HTAB H: VTAB V: FOR I = H TO 1
    3 STEP 2: PRINT "O ";: NEXT : PRINT : GOTO 190                8857
225 IF R = 118 THEN  FOR I = V TO 16: HTAB H: VTAB
    I: PRINT "^": NEXT : GOTO 190                                 0453
230 IF R = 104 THEN  HTAB H: VTAB V: FOR I = H TO
    13 STEP 2: PRINT "^ ";: NEXT : PRINT : GOTO 19
    0                                                             DC92
235 IF R = 5 THEN  GOSUB 310: GOTO 165                            598D
240 IF R = 21 THEN H = H + 2: GOTO 170                            0CCD
245 IF R = 8 THEN H = H - 2: GOTO 175                             9DA8
250 IF R = 10 THEN V = V + 1: GOTO 185                            F4EC
255 IF R = 11 THEN V = V - 1: GOTO 180                            7BE9
260 IF R = 9 OR R = 32 THEN R$ = "O": GOTO 190                    B479
265 IF R = 127 THEN R$ = "^": GOTO 190                            BA94
270 IF R = 13 OR R = 27 THEN 365                                  BDEE
275 IF R = 18 THEN 530                                            6F94
```

(Suite page 36)

100 Présence de la carte 80 colonnes obligatoire.

101 Prévoir LOMEM : 17400.

105 Toute erreur renvoie automatiquement à la routine de création de police (sans modifier la valeur de C, numéro du caractère en cours).

115 Une courte routine en LM efface la mémoire de $4000 à 42FF. Ainsi, les caractères non créés seront à zéro.

125 Aux adresses 6 et 7, on poke (c'est aussi celle de tout set de caractères chargé) l'adresse de la police à créer.

130 Vers l'affichage d'un résumé (presque complet) des commandes.

135 Affichage de la grille.

210 Avant d'aller à la routine d'affichage (page HGR), on sauvegarde le contenu des adresses 6-7, malmenées par cette routine (récupérée dans un ancien numéro de *TM* et légèrement raccourcie).

236 Avec les anciens Apple, ajoutez une ligne ainsi conçue :
IF R = 94 OR R = 79 THEN H = H + 2: GOTO 170

260 Deux possibilités pour tracer un point : TAB et BARRE d'espacement.

265 Une seule solution pour effacer un point : la touche DELETE.

Les ct suivis d'une lettre indiquent qu'il faut taper un CTRL + le caractère indiqué. En 80 colonnes, CTRL-O provoque l'affichage, en mode inverse et CTRL-N l'interrompt.

POLICE.CRÉER (suite)

```
280 IF R = 23 THEN  CALL 913: GOTO 190                    87A0
285 GOTO 190                                              3B45
290 :                                                     003A
295 REM **** ctODESSIN DE LA GRILLEctN  ****
300 :                                                     003A
305 POKE 32,0: POKE 33,80                                 3C67
310 POKE 1403,0: VTAB 10: FOR I = 1 TO 8: FOR J =
    1 TO 7: PRINT "^ ";: NEXT : PRINT : NEXT : RET
    URN                                                   0D5E
315 :                                                     003A
320 REM ***** ctOCOMMANDES DIVERSESctN *****
325 :                                                     003A
330 RESTORE :V = 9: FOR I = 1 TO 10: IF I > 5 THEN
    V = 10                                                77ED
335 READ A$: POKE 1403,40: VTAB I + V: PRINT A$           57D3
340 NEXT                                                  0582
345 RETURN                                                63B1
350 :                                                     003A
355 REM **** ctOUN CARACTERE TERMINEctN ****
360 :                                                     003A
365 VTAB 22: CALL  - 868: PRINT "Bien d'accord pou
    r mémoriser ce caractère (oui/non)ctG ";: GET
    R$: PRINT : IF R$ = "o" OR R$ = "O" THEN 375          0FC7
370 GOTO 165                                              4347
375 CALL 768                                              8331
380 IF R = 27 OR C = 96 THEN 445                          E6E9
385 IF P AND (C + 1) < 97 THEN R = C + 1: GOTO 425        ED3A
390 GOTO 135                                              2144
395 :                                                     003A
400 REM *** ctORAPPELER UN CARACTEREctN ****
405 :                                                     003A
410 VTAB 22: CALL  - 868: INPUT "Quel est le carac
    tère à rappeler (code ASCII) ctG";R$: IF R$ =
    "" THEN 190                                           96AA
415 R =  VAL (R$) - 31                                    CEFB
420 IF R < 1 OR R > 127 THEN 410                          6DE3
425 CALL 841,R:C = R: GOTO 160                            75C2
430 :                                                     003A
435 REM *****ctO TITRE DE LA POLICEctN *****
440 :                                                     003A
445 VTAB 22: CALL  - 868: INPUT "ctOTITRE DE VOTRE
     POLICE ? ctN";T$                                     7449
450 IF T$ = "" THEN 445                                   439A
455 IF T$ = "0" THEN 465                                  AFCC
460 PRINT  CHR$ (4)"BSAVE"T$",A$4000,L768"                9309
465 VTAB 22: CALL  - 868: PRINT "$GOTO 1000 POUR R
    ETROUVER TOUTES LES DONNEES": VTAB 21                 57B1
470 END                                                   0180
475 :                                                     003A
480 REM **** ctOCHARGEMENT DE POLICEctN ****
485 :                                                     003A
```

280 Un CTRL-W envoie à une routine qui déplace l'image du caractère d'un espace vers la gauche. On n'a rien prévu dans les autres sens.

330 Le tableau des commandes pourrait être plus complet, mais rien ne vous interdit de lui adjoindre celles qui ont volontairement été omises, pour ne pas charger inutilement la page.

365 Il faut obligatoirement répondre par **O** ou **o** pour que les codes du caractère en cours soient calculés et mémorisés. Toute autre réponse renvoie au caractère. Sinon, la grille redevient vierge ou, dans le cas de l'exploitation d'une police existante (P = 1), le caractère suivant est affiché !

410 Quand on rappelle un caractère par CTRL-A, le numérotage repart à partir de ce caractère. Ce n'est pas grave dans le cas où une police a été chargée (P = 1 = mode auto), mais si on travaille en mode normal de création revenir au dernier caractère créé par un autre CTRL-A. Rassurez-vous : on s'y fait très vite et il est toujours possible de contrôler le travail en cours par un CTRL-W.

465 GOTO 1000 vous permettra de vous tirer d'un mauvais pas (DISK FULL... par exemple). Notez à ce propos que vous pouvez faire un **BSAVE FONTE, A$4000, $L300** de votre fonte en mode direct (elle est bien là !).

490 Il est facile d'obtenir le catalogue de disquette en tapant le ?

Sous PRODOS, si vous changez de disquette, veillez à ce qu'elle ait le même préfixe... ou modifiez le programme Basic en ajoutant par exemple :

,D1 ou ,S6,D1

aux commandes DOS ou PRODOS.

530 Le numéro d'ordre est calculé à partir du code ASCII du caractère demandé, mais c'est enfantin, le premier caractère étant l'espace (32), le numéro d'ordre est toujours égal à :
Code ASCII - 31

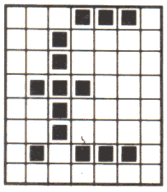
```
490 VTAB 22: CALL  - 868: INPUT "NOM DE LA POLICE
    (ctOCATALOGUEctN = ? ctG) ";F$                    2A24
495 POKE 34,5: HOME : POKE 34,0: GOSUB 330: GOSUB
    305                                               A80A
500 IF F$ = "" THEN 190                               9689
505 IF F$ = "?" THEN  POKE 34,5: HOME : PRINT  CHR
    $ (4)"CATALOG": PRINT : PRINT : GOTO 490          4689
510 PRINT  CHR$ (4)"BLOAD"F$",A$4000":R = 1:P = 1:
    GOTO 425                                          6267
515 :                                                 003A
520 REM *** ctOUN CARAC POUR UN AUTREctN ***
525 :                                                 003A
530 VTAB 22: CALL  - 868: INPUT "Code ASCII du car
    actère à remplacer ctG";R$: IF R$ = "" THEN 19
    0                                                 B73E
535 C =  VAL (R$) - 31: IF C < 1 OR C > 96 THEN 53
    0                                                 30C3
540 R = 16376 + C * 8: POKE 6,R -  INT (R / 256) *
    256: POKE 7, INT (R / 256)                        051F
545 CALL  - 198: GOTO 160                             E973
550 :                                                 003A
555 DATA "DEPLACEMENTS: Flèches de l'Apple","MATER
    IALISER: Tab ou barre d'espacement"," "SUPPRI
    MER: Delete"," "TERMINE: Escape"," DEL GRI
    LLE: Ctrl-E","LIGNE PLEINE: H ou V"," LIGNE V
    IDE: h ou v"                                      AF6C
560 DATA "CAR A REVOIR: Ctrl-A","POLICE EXIST: Ctr
    l-P","REMPLACE CAR: Ctrl-R"                       9B94
1000 POKE 51,0: HOME : GOTO 120                       5EC4
```

Fonte américaine* Police française

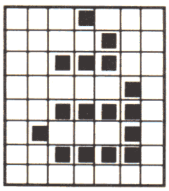
Si vous récupérez une police américaine, amusez-vous à la transformer pour l'utiliser en français... Il faut remplacer les caractères 35, 64, 91, 92, 93, 96, 123, 124, 125 et 126 par d'authentiques caractères français. Pour vous aider, voici un exemple de tracé pour une police de type ASCII :

** Attention ! ne confondez pas la police classique (768 octets : 8 octets par caractère) avec les fontes utilisables par DRAW !*

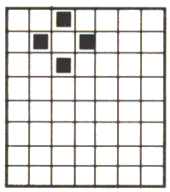
35/$23



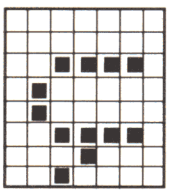
64/$40



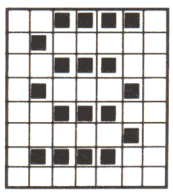
91/$5B



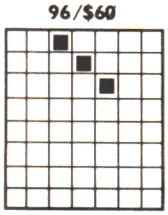
92/$5C



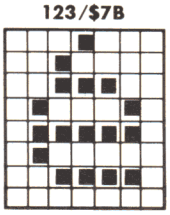
93/$5D



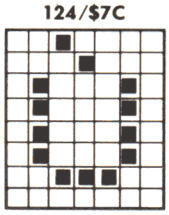
96/$60



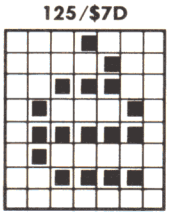
123/$7B



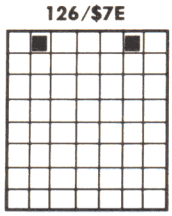
124/$7C



125/$7D



126/$7E



POLICE.LM 2

(BSAVE POLICE.LM,A$300,L$B5)

```
0300-  A2 00      LDX  £$00
0302-  8A         TXA
0303-  69 09      ADC  £$09
0305-  20 C1 FB   JSR  $FBC1
0308-  A0 07      LDY  £$07
030A-  2C 55 C0   BIT  $C055
030D-  B1 28      LDA  ($28),Y
030F-  2C 54 C0   BIT  $C054
0312-  4A         LSR
0313-  36 18      ROL  $18,X
0315-  88         DEY
0316-  10 F2      BPL  $030A
0318-  E8         INX
0319-  E0 08      CPX  £$08
031B-  90 E5      BCC  $0302
031D-  A0 07      LDY  £$07
031F-  B9 18 00   LDA  $0018,Y
0322-  91 06      STA  ($06),Y
0324-  88         DEY
0325-  10 F8      BPL  $031F
0327-  A5 06      LDA  $06
0329-  18         CLC
032A-  69 08      ADC  £$08
032C-  85 06      STA  $06
032E-  90 02      BCC  $0332
0330-  E6 07      INC  $07
0332-  60         RTS
```

X = Ø passé dans A.

+ 9 pour VTAB.
BASCALC fournit l'adresse de la ligne.
Y = 7 (nombre de positions horiz.).
TXT page 2.
Lecture de la mémoire $28-29 + Y.
TXT page 1.
Décalage à droite (bit sortant dans retenue).
La retenue rentre à droite dans $18-19 + X.
Y = Y - 1.
Si Y est plus grand ou égal à zéro : boucle.
Sinon X = X + 1.
X est-il égal à 8 ?
S'il est inférieur, on continue le travail.
Y = Ø7.

Transfert des octets du caractère créé dans la police (à l'adresse $6-7 + Y + 7 à Ø).

On incrémente l'adresse-police caractère de 8 pour le caractère suivant.

Retour au Basic.

NETTOYAGE $4000-42FF

```
0333-  A9 40      LDA  £$40
0335-  85 07      STA  $07
0337-  A2 03      LDX  £$03
0339-  A0 00      LDY  £$00
033B-  84 06      STY  $06
033D-  98         TYA
033E-  91 06      STA  ($06),Y
0340-  88         DEY
0341-  D0 FB      BNE  $033E
0343-  E6 07      INC  $07
0345-  CA         DEX
0346-  D0 F6      BNE  $033E
0348-  60         RTS
```

$4Ø dans l'accumulateur pour adresse-police ($4ØØØ).

X = Ø3.

Y = ØØ est stocké dans la partie basse de l'adresse-police.

Y dans A.
Ø dans $6-7 + Y ;
Y = Y - 1 ;
on continue à mettre les mémoires à Ø.

Partie haute de l'adresse-police incrémentée et X = X - 1 pour un nouveau tour.

La partie $4ØØØ-42FF est à zéro.

RAPPEL D'UN CARACTÈRE

```
0349-  20 F5 E6   JSR  $E6F5
034C-  A9 40      LDA  £$40
034E-  85 07      STA  $07
0350-  A9 00      LDA  £$00
0352-  CA         DEX
0353-  F0 09      BEQ  $035E
0355-  18         CLC
0356-  69 08      ADC  £$08
```

GETBYTC saute un caractère : GETBYT évalue l'expression.

Adresse de la police ($4ØØØ).

A = Ø et X (fourni par GETBYT à partir du Basic) = X -1.

Si X = Ø, saut.

Annulation de la retenue pour l'addition.

```
0358-  90 F8      BCC  $0352
035A-  E6 07      INC  $07
035C-  D0 F4      BNE  $0352
035E-  85 06      STA  $06
0360-  86 18      STX  $18
0362-  8A         TXA
0363-  18         CLC
0364-  69 09      ADC  #$09
0366-  20 C1 FB   JSR  $FBC1
0369-  A4 18      LDY  $18
036B-  B1 06      LDA  ($06),Y
036D-  A0 07      LDY  #$07
036F-  0A         ASL
0370-  0A         ASL
0371-  48         PHA
0372-  90 04      BCC  $0378
0374-  A9 CF      LDA  #$CF
0376-  D0 02      BNE  $037A
0378-  A9 DE      LDA  #$DE
037A-  2C 55 C0   BIT  $C055
037D-  88         DEY
037E-  91 28      STA  ($28),Y
0380-  2C 54 C0   BIT  $C054
0383-  68         PLA
0384-  C8         INY
0385-  88         DEY
0386-  D0 E8      BNE  $0370
0388-  E6 18      INC  $18
038A-  A5 18      LDA  $18
038C-  C9 08      CMP  #$08
038E-  90 D3      BCC  $0363
0390-  60         RTS
```

Pas de retenue : on saute. Sinon on incrémente la partie haute de l'adresse.

Ne pas oublier la partie basse modifiée.
X dans $18
et X dans A.

Annulation de la retenue pour addition (VTAB vrai) et BASCALC fournit l'adresse.

Récupération de HTAB.
Lecture de la mémoire $6-7 + Y.
Y = 7.

Un décalage bit inutilisé.
Un décalage à gauche.
Résultat empilé.
S'il n'y a pas de retenue, saut.

Autrement : O dans A et petit saut.

Accent circonflexe dans A.
TXT page 2.
Y = Y - 1
et on place le caractère à l'adresse $28-29 + Y.
TXT page 1.
Récupération de A sur la pile.

Si Y n'est pas à zéro, on va vers un nouveau décalage.

Mémoire $18 incrémentée.
Si on est inférieur à 8, le processus recommence.

Retour au Basic.

DÉCALAGE DU CARACTÈRE VERS LA GAUCHE

```
0391-  A2 10      LDX  #$10
0393-  8A         TXA
0394-  20 C1 FB   JSR  $FBC1
0397-  A0 00      LDY  #$00
0399-  2C 55 C0   BIT  $C055
039C-  C8         INY
039D-  B1 28      LDA  ($28),Y
039F-  88         DEY
03A0-  91 28      STA  ($28),Y
03A2-  C8         INY
03A3-  C0 07      CPY  #$07
03A5-  D0 F5      BNE  $039C
03A7-  88         DEY
03A8-  A9 DE      LDA  #$DE
03AA-  91 28      STA  ($28),Y
03AC-  CA         DEX
03AD-  E0 09      CPX  #$09
03AF-  10 E2      BPL  $0393
03B1-  2C 54 C0   BIT  $C054
03B4-  60         RTS
```

Au départ, on est sur la dernière ligne de la grille.

Y = Ø.
TXT page 2.

Y = Y + 1 pour lecture.

Y = Y - 1 pour écriture (on écrase l'octet).

Y retrouve sa valeur et on continue aussi longtemps que Y n'est pas égal à Ø7.

Y = Y - 1 pour mettre l'accent circonflexe dans le dernier octet.

X = X - 1.
On renouvelle l'opération jusqu'à X = Ø inclus.

Retour TXT page 1.
BASIC.

(Page 40 : POLICE.HGR)

POLICE.HGR 3

(BSAVE POLICE.HGR,A$4300,L$6E)

OBJET : Affichage d'une police graphique classique à partir de l'adresse $2000.

```
4300-  20 E2 F3   JSR  $F3E2
4303-  85 06      STA  $06
4305-  85 08      STA  $08
4307-  A9 03      LDA  £$03
4309-  85 19      STA  $19
430B-  A9 40      LDA  £$40
430D-  85 07      STA  $07
430F-  6A         ROR
4310-  85 09      STA  $09
4312-  A9 28      LDA  £$28
4314-  85 18      STA  $18
4316-  A0 00      LDY  £$00
4318-  A5 08      LDA  $08
431A-  85 0A      STA  $0A
431C-  A5 09      LDA  $09
431E-  85 0B      STA  $0B
4320-  B1 06      LDA  ($06),Y
4322-  48         PHA
4323-  C0 00      CPY  £$00
4325-  F0 10      BEQ  $4337
4327-  84 0C      STY  $0C
4329-  A6 0C      LDX  $0C
432B-  A5 0B      LDA  $0B
432D-  18         CLC
432E-  69 04      ADC  £$04
4330-  CA         DEX
4331-  D0 FA      BNE  $432D
4333-  85 0B      STA  $0B
4335-  A4 0C      LDY  $0C
4337-  68         PLA
4338-  81 0A      STA  ($0A,X)
433A-  C8         INY
433B-  C0 08      CPY  £$08
433D-  90 D9      BCC  $4318
433F-  A5 06      LDA  $06
4341-  18         CLC
4342-  69 08      ADC  £$08
4344-  85 06      STA  $06
4346-  90 02      BCC  $434A
4348-  E6 07      INC  $07
434A-  E6 08      INC  $08
434C-  D0 02      BNE  $4350
434E-  E6 09      INC  $09
4350-  C6 18      DEC  $18
4352-  D0 C2      BNE  $4316
4354-  C6 19      DEC  $19
4356-  F0 15      BEQ  $436D
4358-  A5 08      LDA  $08
435A-  18         CLC
435B-  69 58      ADC  £$58
```

HGR initialise et efface la page.
Comme A est à Ø, on en profite pour mettre les parties basses de $6-7 et $8-9 à Ø.

$3 dans $19 (nombre de lignes).

$4Ø dans partie haute $6-7 ($4ØØØ) : adresse-police.

Où $4Ø devient $2Ø
pour partie haute $8-9 ($2ØØØ) : adr.HGR.

$28 (4Ø) dans $18 : nombre de caractères par ligne.

Registre Y initialisé à Ø.

Récupération de l'adresse de mémo dans $A-B.

On lit un octet du caractère ($6-7 + Y) et on l'empile.

Si Y est égal à zéro, joli petit saut...

Sinon, Y mémorisé en $C...
ou X le récupère.

Adresse affichage = adr. + 4...
autant de fois qu'il y a de X.

X = X - 1 et suite si l'on n'est pas à zéro.

Nouvelle valeur de $B.
Récupération de la valeur de Y.
On retrouve A sur la pile
et il reste à le poquer en $A-B + X.

Y = Y + 1. Si Y est inférieur à 8, il faut poursuivre les opérations.

Adresse lecture incrémentée de 8,
mais adresse affichage incrémentée de 1
($6-7 + Ø8, mais $8-9 + 1).

Nombre de caractères par ligne décrémenté, on continue aussi longtemps qu'il n'est pas à Ø.
Quand il atteint zéro, nombre de lignes décrémenté.
Si celui-ci est à zéro, saut car on a fini.

Pour une nouvelle ligne, adr. = adr. + $58.

```
435D-   85 08       STA   $08
435F-   90 02       BCC   $4363
4361-   E6 09       INC   $09
4363-   A5 19       LDA   $19
4365-   C9 01       CMP   £$01
4367-   D0 A9       BNE   $4312
4369-   A9 10       LDA   £$10
436B-   D0 A7       BNE   $4314
436D-   60          RTS
```

Où en est-on du nombre de lignes ?
S'il n'est pas égal à 1 (dernière à traiter), on renvoie à 1 ligne de 40 caractères !
Sinon il ne reste que 16 caractères pour la dernière (40 + 40 + 16 = 96).
Terminé !

ASCII.F 3

(BSAVE ASCII.F,A$4000,L$2FF)

```
4000: 00 00 00 00 00 00 00 00 ( ) 08 08 08 08 08 00 08 00 (!) 2D30
4010: 14 14 14 00 00 00 00 00 (") 18 24 04 1E 04 04 3A 00 (£) 72DC
4020: 08 3C 0A 1C 28 1E 08 00 ($) 06 26 10 08 04 32 30 00 (%) 6C62
4030: 04 0A 0A 04 2A 12 2C 00 (&) 08 08 08 00 00 00 00 00 (') 759C
4040: 08 04 02 02 02 04 08 00 (() 08 10 20 20 20 10 08 00 ()) 0CAE
4050: 08 2A 1C 08 1C 2A 08 00 (*) 00 08 08 3E 08 08 00 00 (+) D202
4060: 00 00 00 00 08 08 04 00 (,) 00 00 00 3E 00 00 00 00 (-) 6952
4070: 00 00 00 00 00 00 08 00 (.) 00 20 10 08 04 02 00 00 (/) 4546
4080: 1C 22 32 2A 26 22 1C 00 (0) 08 0C 08 08 08 08 1C 00 (1) 904E
4090: 1C 22 20 18 04 02 3E 00 (2) 3E 20 10 18 20 22 1C 00 (3) F09E
40A0: 10 18 14 12 3E 10 10 00 (4) 3E 02 1E 20 20 22 1C 00 (5) 7E88
40B0: 38 04 02 1E 22 22 1C 00 (6) 3E 20 10 08 04 04 04 00 (7) B43E
40C0: 1C 22 22 1C 22 22 1C 00 (8) 1C 22 22 3C 20 10 0E 00 (9) DBB6
40D0: 00 00 08 00 08 00 00 00 (:) 00 00 08 00 08 08 04 00 (;) 962C
40E0: 10 08 04 02 04 08 10 00 (<) 00 00 3E 00 3E 00 00 00 (=) F3B6
40F0: 04 08 10 20 10 08 04 00 (>) 1C 22 10 08 08 00 08 00 (?) 42BE
4100: 08 10 1C 20 3C 22 3C 00 (à) 08 14 22 22 3E 22 22 00 (A) 0FD0
4110: 1E 22 22 1E 22 22 1E 00 (B) 1C 22 02 02 02 22 1C 00 (C) CF64
4120: 1E 22 22 22 22 22 1E 00 (D) 3E 02 02 1E 02 02 3E 00 (E) 5488
4130: 3E 02 02 1E 02 02 02 00 (F) 3C 02 02 02 32 22 3C 00 (G) E738
4140: 22 22 22 3E 22 22 22 00 (H) 1C 08 08 08 08 08 1C 00 (I) 066A
4150: 20 20 20 20 20 22 1C 00 (J) 22 12 0A 06 0A 12 22 00 (K) 0960
4160: 02 02 02 02 02 02 3E 00 (L) 22 36 2A 2A 22 22 22 00 (M) F35C
4170: 22 22 26 2A 32 22 22 00 (N) 1C 22 22 22 22 22 1C 00 (O) FCEC
4180: 1E 22 22 1E 02 02 02 00 (P) 1C 22 22 22 2A 12 2C 00 (Q) F070
4190: 1E 22 22 1E 0A 12 22 00 (R) 1C 22 02 1C 20 22 1C 00 (S) 0678
41A0: 3E 08 08 08 08 08 08 00 (T) 22 22 22 22 22 22 1C 00 (U) 2156
41B0: 22 22 22 22 22 14 08 00 (V) 22 22 22 2A 2A 36 22 00 (W) 7BD8
41C0: 22 22 14 08 14 22 22 00 (X) 22 22 14 08 08 08 08 00 (Y) E730
41D0: 3E 20 10 08 04 02 3E 00 (Z) 0E 0A 0E 00 00 00 00 00 (°) A2E0
41E0: 00 00 3C 02 02 3C 10 08 (ç) 1C 22 1C 22 1C 20 1E 00 (§) 906A
41F0: 00 00 08 14 22 00 00 00 (^) 00 00 00 00 00 00 00 7F (_) EFBD
4200: 04 08 10 00 00 00 00 00 (`) 00 00 1C 20 3C 22 3C 00 (a) F0F2
4210: 02 02 1E 22 22 22 1E 00 (b) 00 00 3C 02 02 02 3C 00 (c) 0924
4220: 20 20 3C 22 22 22 3C 00 (d) 00 00 1C 22 3E 02 3C 00 (e) F0D8
4230: 18 24 04 1E 04 04 04 00 (f) 00 00 1C 22 22 3C 20 1E (g) DE44
4240: 02 02 1E 22 22 22 22 00 (h) 08 00 0C 08 08 08 1C 00 (i) 7EF2
4250: 10 00 18 10 10 10 12 0C (j) 02 02 22 12 0E 12 22 00 (k) A5F0
4260: 0C 08 08 08 08 08 1C 00 (l) 00 00 36 2A 2A 2A 22 00 (m) 7E26
4270: 00 00 1E 22 22 22 22 00 (n) 00 00 1C 22 22 22 1C 00 (o) 8A44
4280: 00 00 1E 22 22 1E 02 02 (p) 00 00 3C 22 22 3C 20 20 (q) 4980
```

(suite au verso)

ASCII.F 3 (suite et fin)

```
4290: 00 00 3A 06 02 02 02 00 (r) 00 00 3C 02 1C 20 1E 00 (s) 09DE
42A0: 04 04 1E 04 04 24 18 00 (t) 00 00 22 22 22 32 2C 00 (u) 2D2E
42B0: 00 00 22 22 22 14 08 00 (v) 00 00 22 22 2A 2A 36 00 (w) 5A50
42C0: 00 00 22 14 08 14 22 00 (x) 00 00 22 22 22 3C 20 1E (y) 7C54
42D0: 00 00 3E 10 08 04 3E 00 (z) 10 08 1C 22 3E 02 3C 00 (é) 576A
42E0: 04 08 22 22 22 32 2C 00 (ù) 08 10 1C 22 3E 02 3C 00 (è) 78A2
42F0: 00 14 00 00 00 00 00 00 (¨) 00 2A 14 2A 14 2A 00 00 ( ) 24BA
```

Attention ! ne pas taper les mentions en couleur. Elles vous indiquent à quel caractère correspond chaque groupe de 8 codes. En bout de ligne, c'est le code-contrôle vérifiable avec "SIGNATURE".

- **Pour ceux qui n'ont pas d'assembleur et possèdent notre disquette SIGNATURE**

POLICE.HGR (BSAVE POLICE.HGR,A$4300,L$6F)

```
4300: 20 E2 F3 85 06 85 08 A9 03 85 19 A9 40 85 07 6A  C936
4310: 85 09 A9 28 85 18 A0 00 A5 08 85 0A A5 09 85 0B  4016
4320: B1 06 48 C0 00 F0 10 84 0C A6 0C A5 0B 18 69 04  2F36
4330: CA D0 FA 85 0B A4 0C 68 81 0A C8 C0 08 90 D9 A5  4865
4340: 06 18 69 08 85 06 90 02 E6 07 E6 08 D0 02 E6 09  6948
4350: C6 18 D0 C2 C6 19 F0 15 A5 08 18 69 58 85 08 90  B1F7
4360: 02 E6 09 A5 19 C9 01 D0 A9 A9 10 D0 A7 60        B182
```

POLICE.LM (BSAVE POLICE.LM,A$300,L$B5)

```
0300: A2 00 8A 69 09 20 C1 FB A0 07 2C 55 C0 B1 28 2C  7A67
0310: 54 C0 4A 36 18 88 10 F2 E8 E0 08 90 E5 A0 07 B9  7BDB
0320: 18 00 91 06 88 10 F8 A5 06 18 69 08 85 06 90 02  B990
0330: E6 07 60 A9 40 85 07 A2 03 A0 00 84 06 98 91 06  81C0
0340: 88 D0 FB E6 07 CA D0 F6 60 20 F5 E6 A9 40 85 07  B3A0
0350: A9 00 CA F0 09 18 69 08 90 F8 E6 07 D0 F4 85 06  97B9
0360: 86 18 8A 18 69 09 20 C1 FB A4 18 B1 06 A0 07 0A  3FB2
0370: 0A 48 90 04 A9 CF D0 02 A9 DE 2C 55 C0 88 91 28  5639
0380: 2C 54 C0 68 C8 88 D0 E8 E6 18 A5 18 C9 08 90 D3  1A9F
0390: 60 A2 10 8A 20 C1 FB A0 00 2C 55 C0 C8 B1 28 88  E682
03A0: 91 28 C8 C0 07 D0 F5 88 A9 DE 91 28 CA E0 09 10  9E98
03B0: E2 2C 54 C0 60                                   A382
```

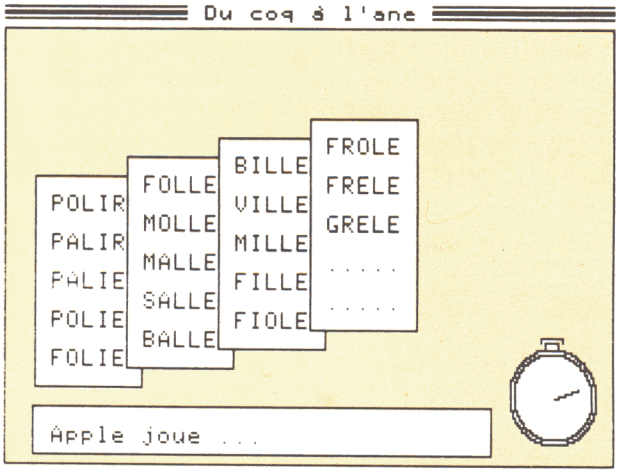
JEU | DOS | ProDOS

Fonctionne aussi sur l'APPLE IIGS

Michel DEVAUX

DU COQ À L'ÂNE

MICHEL DEVAUX récidive. Cette fois, il nous propose un jeu relativement simple, s'adressant aussi bien aux enfants qu'à leurs parents. Son seul défaut : il est long... long à taper et exige la présence, sur la disquette, d'un dictionnaire de mots de 5 lettres. Aussi, exceptionnellement, les habitués des disquettes *TREMPLIN MICRO* recevront ce jeu sur une disquette supplémentaire spéciale (sans augmentation de prix, qu'ils se rassurent !). Ils y trouveront aussi la fonte de caractères parue dans le numéro 5 de *TM...* et réutilisée dans le programme d'aujourd'hui. Commençons par le dictionnaire. Notez qu'il n'est pas indispensable, au départ, qu'il contienne un grand nombre de mots. Par contre, il est indispensable qu'il figure sur la disquette avec sa longueur prévue (L$3418).



CRÉADICO

```
10 REM ENTREE DES MOTS
12 REM
14 HOME : PRINT  CHR$ (21)                    7226
16 PRINT : PRINT " "Ce programm
   e permet la création du fi"                6B31
18 PRINT : PRINT " "chier DICTI
   ONNAIRE . Il vous autorise"                3A32
20 PRINT : PRINT " "à écrire pu
   is à mémoriser sur "disque"                B471
22 PRINT : PRINT " "des mots de
    cinq lettres utiles "pour"                6724
24 PRINT : PRINT " "le jeu du C
   oq à l'Ane ". """""""""""""""             A858
26 PRINT : PRINT " "Lorsque vou
   s "estimerez "avoir "entré"                C277
28 PRINT : PRINT " "suffisammen
   t de mots,entrez une ligne"                DF6B
30 PRINT : PRINT " "vide ... ""
   """""""""""""""""""""""""""               7FA4
32 PRINT : PRINT " "Appuyez sur
    une touche pour commencer";               2EC4
34 GET R$                                     DB34
36 HOME : POKE 37911,0                        22EB
38 LET AD = 24576:MOT = 1                     7032
40 INPUT M$                                   FCF5
42 IF M$ = "" THEN 66                         6B62
44 IF  LEN (M$) <  > 5 THEN 62                8E53
46 FOR I = 1 TO 5                             14C1
48 LET R$(I) =  MID$ (M$,I,1)                 EA08
50 IF R$(I) < "A" OR R$(I) > "Z
   " THEN 62                                  1B8A
52 NEXT I                                     9DCB
54 FOR I = 1 TO 5                             14C1
56 POKE AD + I - 1, ASC (R$(I))               82BC
58 NEXT I                                     9DCB
60 LET AD = AD + 5: GOTO 40                   E3CA
62 PRINT "Mot incorrect ..."                  8361
64 GOTO 40                                    7A0F
66 FOR I = 1 TO 5                             14C1
68 POKE AD + I - 1,0                          09A5
70 NEXT I                                     9DCB
72 PRINT  CHR$ (4)"BSAVE DICTIO
   NNAIRE,A$6000,L$3418"                      3037
```

COQ.BAS

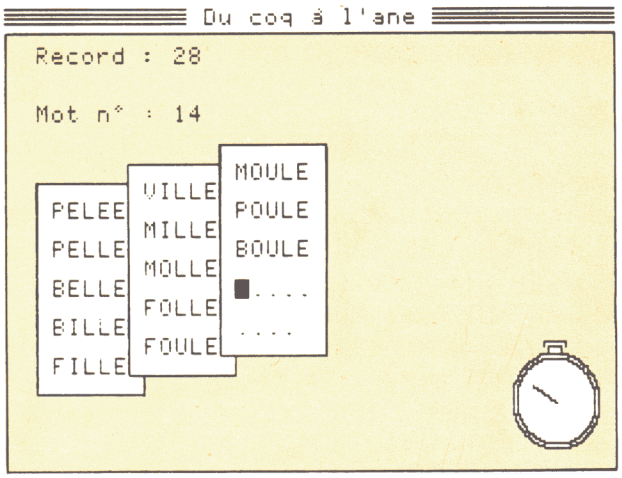
```
10 PRINT  CHR$ (4)"BLOAD FONTE.LM"                    9F2F
12 PRINT  CHR$ (4)"BLOAD COQ.LM"                      2796
14 PRINT  CHR$ (4)"BLOAD DICTIONNAIRE"                EB65
16 REM ctJctJ GOSUBctJ                                
18 GOSUB 94                                           5E1D
20 GOSUB 38                                           821B
22 GOSUB 190                                          094A
24 IF JEU = 1 THEN  GOSUB 308                         DDA1
26 IF JEU = 2 THEN  GOSUB 220                         9D9B
28 GOSUB 362                                          0F4B
30 IF R$ = "O" THEN 22                                EDAE
32 GOSUB 424                                          024A
34 TEXT : HOME                                        1C5A
36 END                                                0180
38 REM ctJctJDESSIN DE L'ECRANctJ                     
40 HGR                                                2391
42 TEXT : HOME                                        1C5A
44 GOSUB 114                                          2146
46 HCOLOR= 7                                          49C9
48 FOR I = 0 TO 7 STEP 2                              D0BB
50 HPLOT 0,I TO 83,I                                  6ED9
52 HPLOT 196,I TO 279,I                               6780
54 NEXT                                               0582
56 HPLOT 0,10 TO 0,191 TO 279,191 TO 279,10 TO 0,10   67A0
58 HPLOT 252,182                                      70F3
60 FOR I = 0 TO 6.28 STEP .05                         5BB3
62 HPLOT  TO 252 + 20 *  SIN (I),163 + 20 *  COS (I)  558C
64 NEXT I                                             9DCB
66 HPLOT 248,143 TO 248,140 TO 254,140 TO 254,143     9F52
68 HPLOT 246,140 TO 246,138 TO 256,138 TO 256,140     9A5A
70 HPLOT 252,180                                      6CF1
72 FOR I = 0 TO 6.28 STEP .05                         5BB3
74 HPLOT  TO 252 + 18 *  SIN (I),163 + 18 *  COS (I)  269A
76 NEXT I                                             9DCB
78 CALL  - 198: VTAB 23: HTAB 1: CALL  - 958          546E
80 VTAB 23: HTAB 1: PRINT "Tapez le numéro de votre choix : "  F90F
82 VTAB 23: HTAB 37: GET R$: IF R$ < "1" OR R$ > "2" THEN 82  9E09
84 HOME :JEU =  VAL (R$)                              AE31
86 POKE  - 16304,0: POKE  - 16300,0                   0AEE
88 POKE  - 16297,0: POKE  - 16302,0                   39FB
90 VTAB 1: HTAB 14: PRINT "Du coq à l'ane"            94A3
92 CALL R1                                            EC0F
94 REM ctJctJINITIALISATION ctJ                       
96 PRINT  CHR$ (21)                                   A455
98 LET R1 = 16384: REM FONTE.LM                       C103
100 LET R2 = 20556: REM EFFACEMENT D'UNE PARTIE D'ECRAN  D300
102 LET R3 = 20624: REM ROUTINE JOUEUR                C2FD
104 LET R4 = 21414: REM CE MOT FAIT-IL PARTI DU DICTIONNAIRE ?  17FC
106 LET R5 = 21513: REM APPLE JOUE                    DAFD
108 LET R6 = 20352: REM COMBIEN DE MOTS DANS LE DICTIONNAIRE ?  B1FE
110 LET REC =  PEEK (37911)                           868C
112 RETURN                                            63B1
114 REM ctJctJPRESENTATION ctJ                        
116 PRINT " DU COQ A L'ANE "                          9BA2
118 PRINT "_____"                                     A5D6
120 PRINT "Passez du Coq à l'Ane :""C'est à dire";    5686
122 PRINT "d'un mot à un autre en "ne modifiant qu'";  C549
124 PRINT "une lettre et en "respectant l' ordre du";  A69E
126 PRINT "mot précédent . """""""""""""""""""""";    08BD
128 PRINT "Ecrivez les mots de votre choix en ayant";  43FC
130 PRINT "toutefois conscience qu'Apple les mémori";  FB77
132 PRINT "se et qu'il "pourra les réutiliser à "un";  89CA
134 PRINT "autre moment du jeu ( évitez donc d ' in";  2882
136 PRINT "venter des mots ... )"                      CF24
138 PRINT "Deux possibilités de jeu vous sont offer";  2B6F
140 PRINT "tes ":""""""""""""""""""""""""""""""""""""  15A4
142 PRINT "1. Vous jouez seul(e)
```

```
     et essayez d ' éta";          4A79
144 PRINT "blir un record de dur
    ée ".  """""""""""""""""      336B
146 PRINT "2. Vous jouez contre
    l'ordinateur .  """""         AEE1
148 VTAB 21: PRINT "_____________
    ____________________________
    "                             9815
150 PRINT "  """""""""""""""""""""
    """""""un moment svp";        A945
152 RETURN                        63B1
154 REM ctJctJCADREctJ
156 IF MOT = 01 THEN CH = 03:CV
    = 10: GOTO 170                13D1
158 IF MOT = 06 THEN CH = 09:CV
    = 09: GOTO 170                1FE4
160 IF MOT = 11 THEN CH = 15:CV
    = 08: GOTO 170                28DC
162 IF MOT = 16 THEN CH = 21:CV
    = 07: GOTO 170                BDDD
164 IF MOT = 21 THEN CH = 27:CV
    = 06: GOTO 170                D8DE
166 IF MOT = 26 THEN CH = 33:CV
    = 05: GOTO 170                6CDF
168 RETURN                        63B1
170 POKE 6,CH: POKE 7,CV: POKE 8
    ,CH + 6: POKE 9,CV + 10       678F
172 CALL R2:X = CH - 1:Y = CV -
    1: HCOLOR= 7                  54F0
174 HPLOT X * 7,Y * 8 TO (X + 7)
    * 7,Y * 8                     2764
176 HPLOT  TO (X + 7) * 7,(Y + 1
    1) * 8 - 1 TO X * 7,(Y + 11)
    * 8 - 1                       1C0F
178 HPLOT  TO X * 7,Y * 8         4634
180 FOR I = 1 TO 5                14C1
182 VTAB CV - 1 + I * 2: HTAB CH
    + 1: PRINT "....."            80B4
184 NEXT I                        9DCB
186 CALL R1                       EC0F
188 RETURN                        63B1
190 REM ctJctJTIRAGE DU PREMIER
    MOTctJ
192 LET MOT = 1:NM = 1: GOSUB 15
    4                             23F5
194 CALL R6                       E214
196 LET X =  PEEK (20364) * 256
    + PEEK (20363) - 24576        F835
198 LET X = X / 5                 092A
200 LET R =  INT ( RND (1) * X)   A16F
202 LET R = R * 5                 351D
204 HCOLOR= 0: HPLOT 21,86 TO 56
    ,86                           8452
206 FOR I = 1 TO 5                14C1
208 POKE 21759 + I, PEEK (24575
    + R + I)                      4763
210 VTAB 11: HTAB 3 + I: PRINT
    CHR$ ( PEEK (24575 + R + I))  F5A9
212 NEXT I                        9DCB
214 LET MOT = 2:NM = 2            2273
216 CALL R1                       EC0F
218 RETURN                        63B1
220 REM ctJctJJOUER CONTRE APPLE
    ctJ
222 HCOLOR= 7                     49C9
224 HPLOT 13,167 TO 223,167 TO 2
    23,187 TO 13,187 TO 13,167    4DE7
226 LET J = 1: GOSUB 154          9779
228 VTAB 23: HTAB 4: PRINT "A vo
    us de jouer ...": CALL R1     FC32
230 LET I =  INT ((MOT - 1) / 5)  3B22
232 LET I = MOT - I * 5           22C4
234 LET MV = CV - 1 + I * 2       2DBD
236 LET MH = CH                   459A
238 POKE 10,MH: POKE 7,MV         16D4
240 LET AD = 21759 + (NM - 1) *
    5                             CAB4
242 POKE 20820, INT (AD / 256)    A8F2
244 POKE 20819,AD -  INT (AD / 2
    56) * 256                     46AF
246 CALL R3                       E811
248 :                             003A
250 IF  PEEK (8) <  > 255 THEN
    RETURN                        D5C9
252 :                             003A
254 CALL R4                       E612
256 :                             003A
258 LET NM = NM + 1               02A9
260 LET MOT = MOT + 1             F653
262 IF MOT = 31 THEN MOT = 1      5386
264 :                             003A
266 LET J = 2: GOSUB 154          3B7A
268 POKE 6,4: POKE 7,23: POKE 8,
    31: POKE 9,23: CALL R2        34CC
270 VTAB 23: HTAB 4: PRINT "Appl
    e joue ...": CALL R1          14FB
272 LET I =  INT ((MOT - 1) / 5)  3B22
274 LET I = MOT - I * 5           22C4
276 LET MV = CV - 1 + I * 2       2DBD
278 LET MH = CH                   459A
280 POKE 10,MH: POKE 7,MV         16D4
282 LET AD = 21759 + (NM - 1) *
    5                             CAB4
284 POKE 21615, INT (AD / 256)    54F5
286 POKE 21614,AD -  INT (AD / 2
    56) * 256                     1FA9
288 CALL R5                       E413
```

(SUITE PAGE 46)

COQ.BAS (suite)

```
290 :                                          003A
292 IF  PEEK (8) <  > 255 THEN
    RETURN                                     D5C9
294 :                                          003A
296 LET NM = NM + 1                            02A9
298 LET MOT = MOT + 1                          F653
300 IF MOT = 31 THEN MOT = 1                   5386
302 :                                          003A
304 POKE 6,4: POKE 7,23: POKE 8,
    31: POKE 9,23: CALL R2                     34CC
306 GOTO 226                                   2B45
308 REM ctJctJJOUER SEULctJ
310 LET J = 0                                  6EF4
312 VTAB 3: HTAB 3: PRINT "Recor
    d :";REC                                   4C5E
314 VTAB 6: HTAB 3: PRINT "Mot n
    ° :";NM: CALL R1                           1465
316 GOSUB 154:I =  INT ((MOT - 1
    ) / 5)                                     59FC
318 LET I = MOT - I * 5                        22C4
320 LET MV = CV - 1 + I * 2                    2DBD
322 LET MH = CH                                459A
324 POKE 10,MH: POKE 7,MV                      16D4
326 LET AD = 21759 + (NM - 1) *
    5                                          CAB4
328 POKE 20820, INT (AD / 256)                 A8F2
330 POKE 20819,AD -  INT (AD / 2
    56) * 256                                  46AF
332 CALL R3                                    E811
334 :                                          003A
336 IF  PEEK (8) = 255 THEN 344                24E3
338 LET NM = NM - 1                            08AA
340 IF NM > REC THEN REC = NM                  3AFA
342 RETURN                                     63B1
344 :                                          003A
346 CALL R4                                    E612
348 :                                          003A
350 LET NM = NM + 1                            02A9
352 LET MOT = MOT + 1                          F653
354 IF MOT = 31 THEN MOT = 1                   5386
356 :                                          003A
358 POKE 6,3: POKE 7,6: POKE 8,1
    5: POKE 9,6: CALL R2                       416F
360 GOTO 314                                   2F43
362 REM ctJctJPERDU !ctJ
364 POKE 6,4: POKE 7,23: POKE 8,
    31: POKE 9,23: CALL R2                     34CC
366 VTAB 23: HTAB 4                            AB0B
368 IF J = 1 THEN  PRINT "Vous a
    vez perdu ..."                             D4A7
370 IF J = 2 THEN  PRINT "Vous a
```




```
    vez gagné ..."                             33C0
372 IF JEU = 1 THEN  PRINT "C'es
    t terminé ..."                             6ABE
374 CALL R1                                    EC0F
376 FOR I = 1 TO 1000: NEXT I                  AE52
378 POKE 6,4: POKE 7,4: POKE 8,2
    6: POKE 9,12: CALL R2                      2D9D
380 HCOLOR= 7                                  49C9
382 HPLOT 21,24 TO 181,24 TO 181
    ,95 TO 21,95 TO 21,24                      ACDE
384 VTAB 6: HTAB 5: PRINT "Voule
    z-vous"                                    8374
386 VTAB 8: HTAB 5: PRINT "recom
    mencer une autre"                          2BF0
388 VTAB 10: HTAB 5: PRINT "part
    ie ? """"O/N"                              1DF0
390 CALL R1                                    EC0F
392 GET R$: IF R$ <  > "O" AND R
    $ <  > "N" THEN 392                        E39B
394 IF R$ = "N" THEN  RETURN                   5AFA
396 POKE 6,6: POKE 7,5: POKE 8,2
    8: POKE 9,13: CALL R2                      92A3
398 HPLOT 35,32 TO 195,32 TO 195
    ,103 TO 35,103 TO 35,32                    0240
400 VTAB 9: HTAB 7: PRINT "Recom
    mencer"                                    C54A
402 VTAB 11: HTAB 7: PRINT "le m
    ème jeu """O/N"                            F5A0
404 CALL R1                                    EC0F
406 GET N$: IF N$ <  > "O" AND N
    $ <  > "N" THEN 406                        6E8B
408 IF N$ = "O" THEN 414                       C2DF
410 IF JEU = 1 THEN JEU = 2: GOT
    O 414                                      37BA
412 IF JEU = 2 THEN JEU = 1                    703C
414 POKE 6,2: POKE 7,3: POKE 8,3
```

```
    2: POKE 9,24: CALL R2          149A
416 POKE 6,32: POKE 7,3: POKE 8,
    40: POKE 9,17: CALL R2         BFCE
418 POKE 6,4: POKE 7,23: POKE 8,
    31: POKE 9,23: CALL R2         34CC
420 HPLOT 0,191 TO 279,191 TO 27
    9,10                           66A4
422 RETURN                         63B1
424 REM ctJctJSAUVER LE DICTIONN
    AIRE ?ctJ
426 POKE 6,6: POKE 7,5: POKE 8,2
    8: POKE 9,13: CALL R2          92A3
428 HPLOT 35,32 TO 195,32 TO 195
    ,103 TO 35,103 TO 35,32        0240
430 VTAB 7: HTAB 7: PRINT "Désir
    ez-vous sauver"                BED4
432 VTAB 9: HTAB 7: PRINT "le Di
    ctionnaire "et"                BD1D
434 VTAB 11: HTAB 7: PRINT "le R
    ecord ? """O/N"                0E7E
436 CALL R1                        EC0F
438 GET R$: IF R$ <  > "O" AND R
    $ <  > "N" THEN 438            F89C
440 IF R$ = "N" THEN 446           5AE7
442 POKE 37911,REC                 44C4
444 PRINT : PRINT  CHR$ (4)"BSAV
    E DICTIONNAIRE,A$6000,L$3418
    "                              352B
446 RETURN                         63B1
```

Fin du programme Basic ■

COQ.LM

Vous vérifierez facilement vos copies de programmes si vous utilisez la disquette **SIGNATURE** mise au point par Yvan KŒNIG, pour les lecteurs de *Tremplin Micro*.

(Bulletin de commande à la fin de la revue).

```
4F80: A9 00 8D 8B 4F A9 60 8D 8C 4F AD 00 60 D0 01 60   50BF
4F90: 18 AD 8B 4F 69 05 8D 8B 4F AD 8C 4F 69 00 8D 8C   1A7E
4FA0: 4F 4C 8A 4F 00 00 00 00 00 00 00 00 00 00 00 00   3074
4FB0: A9 FF 8D CC 4F A9 54 8D CD 4F AD 40 54 8D C9 4F   A4DC
4FC0: AD 41 54 8D CA 4F A2 01 BD 04 60 DD 09 55 D0 06   93BD
4FD0: E8 E0 06 D0 F3 60 18 AD CC 4F 69 05 8D CC 4F AD   9194
4FE0: CD 4F 69 00 8D CD 4F AD 6E 54 CD CC 4F D0 D7 AD   23D9
4FF0: 6F 54 CD CD 4F D0 CF A9 FF 85 08 60 EA EA EA 00   A59E
5000: A9 00 85 00 A5 06 38 E9 01 85 01 85 06 A5 07 38   21F0
5010: E9 01 85 02 A2 00 18 26 02 E8 E0 03 D0 F8 A2 00   8A88
5020: 18 A5 01 65 06 85 01 A5 00 69 00 85 00 E8 E0 06   8410
5030: D0 EE 60 A5 09 38 E9 01 85 03 A2 00 18 26 03 E8   6641
5040: E0 03 D0 F8 A5 03 18 69 07 85 03 60 20 00 50 20   2F53
5050: 33 50 A5 02 A6 01 A4 00 20 11 F4 A4 E5 A9 00 91   855D
5060: 26 C8 C4 08 D0 F9 A6 02 E6 02 E4 03 D0 E4 60 20   912E
5070: 00 50 A2 00 86 03 A5 02 A6 01 A4 00 20 11 F4 A6   C438
5080: 03 BD D0 52 A4 E5 91 26 E6 02 E8 E0 08 D0 E5 60   8AEF
5090: A9 01 85 08 A9 00 85 09 85 0C A2 07 20 EC F6 20   3ECA
50A0: 05 51 2C 30 C0 A9 D0 8D 82 50 18 A5 08 65 0A 85   A403
50B0: 06 20 6F 50 AD 00 C0 38 E9 80 85 0B C9 0D D0 03   3C2C
50C0: 4C 84 51 C9 7F D0 03 20 58 51 C9 08 D0 03 20 58   3821
50D0: 51 38 E9 41 30 0A A5 0B 3C E9 5B 10 03 20 2C 51   FCC9
50E0: A9 00 8D 10 C0 A9 19 20 A8 FC A6 0C E8 86 0C D0   F588
50F0: B4 A2 00 20 EC F6 20 05 51 A5 09 18 69 03 85 09   FA8E
5100: C9 5D D0 96 60 A2 FC A0 00 A9 A3 20 57 F4 A6 09   CF90
5110: BD E8 52 85 04 E8 BD E8 52 85 05 E8 BD E8 52 A8   9670
5120: E8 A5 04 A6 05 20 3A F5 20 CB F5 60 A5 08 C9 06   6647
5130: D0 01 60 A5 0B 38 E9 41 A2 00 18 2A E8 E0 03 D0   FAC2
5140: F9 8D 82 50 18 A5 08 65 0A 85 06 20 6F 50 A6 08   7FA4
5150: A5 0B 9D 04 55 E6 08 60 A5 08 C9 01 D0 01 60 A5   F041
5160: 08 C9 06 D0 0A C6 08 A9 E0 8D 82 50 4C 74 51 A9   7321
5170: D8 8D 82 50 A5 08 65 0A 85 06 20 6F 50 C6 08 D0   115B
5180: EE E6 08 60 A5 08 C9 06 F0 03 4C E0 50 EA EA EA   C2E5
5190: A9 00 85 F9 AD 53 51 8D BA 51 8D 55 53 8D B3 53   43D8
51A0: 38 E9 05 8D BD 51 AD 54 51 8D BB 51 8D 56 53 8D   0C6F
51B0: B4 53 E9 00 8D BE 51 A2 01 BD 04 55 DD FF 54 F0   F565
```

→

```
51C0: 02 E6 F9 E8 E0 06 D0 F1 A5 F9 C9 01 F0 09 20 58  1749
51D0: 51 20 3A FF 4C E0 50 4C 48 53 00 00 00 00 00 00  F60D
51E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  0000
51F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  0000
5200: 08 14 22 22 3E 22 22 00 1E 22 22 1E 22 22 1E 00  ABC4
5210: 1C 22 02 02 02 22 1C 00 1E 22 22 22 22 22 1E 00  E468
5220: 3E 02 02 1E 02 02 3E 00 3E 02 02 0E 02 02 02 00  AEF8
5230: 3C 02 02 02 32 22 3C 00 22 22 22 3E 22 22 22 00  F0DC
5240: 1C 08 08 08 08 08 1C 00 20 20 20 20 20 22 1C 00  753E
5250: 22 12 0A 06 0A 12 22 00 02 02 02 02 02 02 3E 00  90CC
5260: 22 36 2A 2A 22 22 22 00 22 22 26 2A 32 22 22 00  F01C
5270: 1C 22 22 22 22 22 1C 00 1E 22 22 1E 02 02 02 00  DE68
5280: 1C 22 22 22 2A 12 2C 00 1E 22 22 1E 0A 12 22 00  F3A8
5290: 1C 22 02 1C 20 22 1C 00 3E 08 08 08 08 08 08 00  3F28
52A0: 22 22 22 22 22 22 1C 00 22 22 22 22 22 14 08 00  DBAE
52B0: 22 22 22 2A 2A 36 22 00 22 22 14 08 14 22 22 00  39CA
52C0: 22 22 14 08 08 08 08 00 3E 20 10 08 04 02 3E 00  7B32
52D0: 7F 7F 7F 7F 7F 7F 7F 7F 00 00 00 00 00 00 08 00  A500
52E0: 00 00 00 00 00 00 00 00 FB 00 97 FE 00 97 00 01  2B28
52F0: 98 03 01 99 04 01 9A 06 01 9D 07 01 9F 07 01 A1  9CC8
5300: 07 01 A4 07 01 A6 06 01 A9 04 01 AB 03 01 AC 00  576A
5310: 01 AD FE 00 AE FB 00 AE F9 00 AE F7 00 AD F4 00  BA42
5320: AC F3 00 AA F1 00 A8 F0 00 A6 F0 00 A4 F0 00 A1  D09D
5330: F0 00 9F F1 00 9C F3 00 9A F5 00 99 F7 00 97 F9  A1BE
5340: 00 97 FB 00 97 00 00 00 A9 FF 8D 58 53 A9 54 8D  C993
5350: 59 53 A2 01 BD 04 55 DD 04 55 D0 08 E8 E0 06 D0  FB11
5360: F3 4C CE 51 AD 58 53 18 69 05 8D 58 53 AD 59 53  64CD
5370: 69 00 8D 59 53 AD 55 53 CD 58 53 D0 D5 AD 56 53  266A
5380: CD 59 53 D0 CD A2 00 20 EC F6 20 05 51 A9 E0 8D  B246
5390: 82 50 18 A5 08 65 0A 85 06 20 6F 50 A9 00 8D 10  C1B6
53A0: C0 A9 FF 85 08 60 A9 FF 8D B6 53 A9 5F 8D B7 53  0F32
53B0: A2 01 BD 04 55 DD FF 5F D0 06 E8 E0 06 D0 F3 60  64BB
53C0: AD B6 53 18 69 05 8D B6 53 8D E6 53 8D FC 53 8D  7901
53D0: 06 54 AD B7 53 69 00 8D B7 53 8D E7 53 8D FD 53  33B5
53E0: 8D 07 54 A2 01 BD 0E 60 D0 C6 AD B3 53 8D F9 53  1AD8
53F0: AD B4 53 8D FA 53 A2 01 BD 04 55 9D 0E 60 E8 E0  4E1A
5400: 06 D0 F5 A9 00 9D 0E 60 60 A9 01 85 08 A9 00 85  5044
5410: 09 A9 FF 8D 40 54 A9 5F 8D 41 54 AD 53 51 8D 43  EE1D
5420: 54 AD 54 51 8D 44 54 A2 07 20 EC F6 20 05 51 2C  5E18
5430: 30 C0 A9 FF 20 A8 FC A2 00 86 0C A2 01 A0 00 BD  8B90
5440: 11 94 DD 04 55 D0 01 C8 E8 E0 06 D0 F2 C0 04 D0  5C98
5450: 46 20 B0 4F A5 08 C9 FF D0 3D AD 40 54 8D 6B 54  4574
5460: AD 41 54 8D 6C 54 A2 01 86 08 BD 04 60 9D 09 55  CDDC
5470: 38 E9 41 A2 00 18 2A E8 E0 03 D0 F9 8D 82 50 18  E451
5480: A5 08 65 0A 85 06 20 6F 50 A6 08 E8 E0 06 D0 D8  47AA
5490: A9 FF 85 08 4C CD 54 18 AD 40 54 69 05 8D 40 54  B48A
54A0: AD 41 54 69 00 8D 41 54 A9 40 20 A8 FC A5 0C 18  0843
54B0: 69 03 85 0C 90 85 A2 00 20 EC F6 20 05 51 18 A5  A0E9
54C0: 09 69 03 85 09 C9 5D F0 03 4C 27 54 60 A2 00 20  AF05
54D0: EC F6 20 05 51 60 00 00 00 00 00 00 00 00 00 00  12B8
54E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  0000
54F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  0000
5500: 42 41 4C 4C 45 56 4F 47 55 45 42 41 47 55 45 FF  2D49
5510: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  FFF0
5520: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  FFF0
5530: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  FFF0
5540: 0C FF FF FF FF A0 A0 FF FF FF FF FF FF FF FF FF  243F
5550: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  FFF0
5560: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  FFF0
5570: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  FFF0
```

COQ.LM

(suite et fin)

BSAVE COQ.LM, A$4FØØ,L$68Ø

FONTE.LM

Reprendre les codes parus dans le numéro 5 de *T.M.* (pages 34 et 35).

Modifiez ainsi les adresses : **4004 : A9 00 85 25**

Les variables numériques dans la mémoire de l'Apple

Dans votre programme en Basic, les représentations des variables numériques sont fort claires : elles correspondent à celles auxquelles nous ont habitués les mathématiques élémentaires :

P = 0.75 ou
PI = 3.1416

Dans la mémoire de notre Apple, c'est une autre histoire ! A chaque variable numérique correspond (dans les exemples présents) une série de 7 octets. Les numéros 1 et 2 sont réservés au nom de la variable (c'est-à-dire à ses deux premières lettres, codées en ASCII). Le troisième concerne l'Exposant et les quatre suivants la Mantisse. Il peut être intéressant , notamment dans une routine en langage machine, de représenter un nombre connu sous cette forme. On en verra un exemple plus loin. Notre courte démo vous montre exactement ce que deviennent les variables **P** et **PI** de la ligne 20 après un RUN.

VARMEM

```
10 PRINT  CHR$ (4)"PR£3": PRINT : HOME      E227
15 PRINT "ctOLES VARIABLES NUMERIQUES DAN
   S LA MEMOIRE DE VOTRE APPLE IIctNctN"    6DB2
20 P = 0.75:PI = 3.1416                     3BBA
25 PRINT : INVERSE : LIST 20: NORMAL        F6C1
30 PRINT "§CALL-151...return": PRINT        3D99
35 PRINT "*69.6C ¨¨¨¨¨return": PRINT        74D2
40 PRINT "*69:F0 09 FE 09...réponse de l'
   Apple (début et fin des variables)"      E514
45 PRINT "*9F0.9FD ¨¨¨return": PRINT        621C
50 PRINT "Réponse (très claire) de votre
   ctOGSctN (celle du IIe sera sur 2 lign
   es):": PRINT                             2084
55 PRINT : PRINT "0/09F0:50 00 82 40 00 0
   0 00 50 49 82 49 0F F9 73-P..à...PI.I.
   ys"                                      1552
60 PRINT " ¨¨¨¨¨¨ctOPctN ¨¨¨¨Ex M1 M2 M3
   M4 ctOPctN ¨ctOIctN ¨Ex M1 M2 M3 M4"     1694
```

VOICI CE QUE VOUS OBTIENDREZ SUR VOTRE ÉCRAN :

```
§RUN

LES VARIABLES NUMERIQUES DANS LA MEMOIRE DE VOTRE APPLE II

20 P = 0.75:PI = 3.1416

§CALL-151...return

*69.6C       return

*69:F0 09 FE 09...réponse de l'Apple (début et fin des variables)
*9F0.9FD     return

Réponse (très claire) de votre GS (celle du IIe sera sur 2 lignes):

0/09F0:50 00 82 40 00 00 00 50 49 82 49 0F F9 73-P..à...PI.I.ys
       P     Ex M1 M2 M3 M4 P  I  Ex M1 M2 M3 M4
```

EXPOSANT

Les variables numériques dans la mémoire de l'Apple

Comment multiplier ou diviser une variable numérique par 2... sans passer par la multiplication et la division traditionnelles ?

```
 10 TEXT : PRINT  CHR$ (21): HOME                                  27E9
 20 A = 1000                                                       2BD2
 30 PRINT "A = ";: INVERSE : PRINT A: NORM
    AL : PRINT                                                     4CCF
 40 AD = ( PEEK (105) +  PEEK (106) * 256)
    + 2                                                           6D62
 50 FOR I = 1 TO 20: POKE AD,( PEEK (AD))
    + 1: PRINT A: NEXT                                            A685
 60 POKE 32,13: POKE 33,27: HOME : LIST 20
    ,50: PRINT : PRINT                                            043E
 70 INVERSE : PRINT "MULTIPLICATION OBTENU
    E";: NORMAL : PRINT " en ¨augmentant l
    'EXPOSANT de 1 à chaque tour de la bou
    cle."                                                         2265
 80 GOSUB 200                                                      0942
 90 GOSUB 210: POKE 34,1: HOME : PRINT : P
    RINT                                                          07B3
100 A = 1000                                                       2BD2
110 FOR I = 1 TO 16: POKE AD,( PEEK (AD))
    - 1: PRINT A: NEXT                                            548B
120 POKE 32,13: POKE 33,27: HOME : LIST 10
    0,110: PRINT : PRINT                                          D39A
130 PRINT : INVERSE : PRINT " ET MAINTENAN
    T DIVISION";: NORMAL : PRINT " en dimi
    nuant ¨l'EXPOSANT de 1 à chaque tour d
    e la boucle."                                                 22FB
140 GOSUB 200: GOSUB 210                                           1CBF
150 VTAB 22: PRINT " ¨¨¨¨¨¨<M>ENU DE DISQU
    ETTE ctG";: GET R$: PRINT R$: POKE 34,
    0: IF R$ = "M" THEN  PRINT  CHR$ (4)"R
    UN MENU,D1"                                                   AF15
160 END                                                            0180
200 CALL  - 198: VTAB 19: PRINT " ¨¨¨¨¨¨";:
    INVERSE : PRINT "UNE TOUCHE SVP";: GET
    R$: PRINT R$: NORMAL : RETURN                                 593E
210 POKE 32,0: POKE 33,40: RETURN                                  104E
```

Juste pour s'amuser un peu, et avec les limites que comporte ce jeu !

La ligne 40 nous fournit l'adresse du A que vient d'afficher la ligne 30. On ajoute 2 à cette adresse pour obtenir celle de l'Exposant de A.

La ligne 60 se contente d'incrémenter la valeur de l'Exposant ce qui a pour effet immédiat de multiplier A par 2.

La seconde partie de la démo fait exactement le contraire. On décrémente la valeur de l'Exposant ce qui a pour effet de diviser A par 2.

Amusant, non ?

DÉMO.LM

```
 10 DATA 32,88,252,169,27,160,3,32,249,234
    ,32,46,237,32,98,252,206,27,3,173,27,3
    ,201,122,176,233,96,140,122,0,0,0                             9181
 20 FOR I = 768 TO 799: READ R: POKE I,R:
    NEXT                                                          C332
 30 CALL 768                                                       8331
 40 VTAB 23: PRINT "<M>ENU DE DISQUETTE ct
    G";: GET R$: VTAB 22: PRINT : IF R$ =
    "M" THEN  PRINT  CHR$ (4)"-/TMC/MENU"                         D203
```

Cette démonstration accomplit la même tâche que la seconde partie de la précédente : elle divise A par deux.

Après un **RUN DEMOLM**, tapez un **CALL - 151** suivi d'un indispensable **RETURN**, puis :

```
300 :  20 58 FC   JSR  $FC58
303 :  A9 1B      LDA  £$1B
305 :  A0 03      LDY  £$03
307 :  20 F9 EA   JSR  $EAF9
30A :  20 2E ED   JSR  $ED2E
30D :  20 62 FC   JSR  $FC62
310 :  CE 1B 03   DEC  $031B
313 :  AD 1B 03   LDA  $031B
316 :  C9 7A      CMP  £$7A
318 :  B0 E9      BCS  $0303
31A :  60         RTS
31B :  79 7A 00
31E :  00
31F :  00
```

Le HOME traditionnel efface votre écran.

Adresse de la variable dans l'Accumulateur (partie basse) et dans le registre Y (partie haute).

MOVFM place la variable dans FAC.

PRNTFAC affiche la valeur courante de FAC.

CR envoie un gentil retour chariot.

La valeur de l'Exposant est décrémentée.

Il est placé dans A et s'il est plus grand ou égal à $7A, on continue.

Retour au Basic.

Ici, le nom de la variable est inutile et on se contente d'utiliser l'Exposant ($31B) et la Mantisse : 7A 00 00 00. Vous noterez que l'Exposant est ici à $79 parce que ce listage a été effectué après le fonctionnement de la routine. Au départ, l'Exposant était égal à $8C.

PETITE EXPÉRIENCE :

Tapez **30A : 60**, puis **300G**... et **9D.A2** (sans oublier un **RETURN** à la suite de chaque opération). Votre Apple vous affichera les deux lignes suivantes (il s'agit de la partie de FAC qui nous intéresse) :

```
009D :  79 FA 00
00A0 :  00 00 7A
```

où, comme vous pouvez le constater, l'Exposant se trouve bien en $9D... tandis que la Mantisse est quelque peu malmenée.

En fait, $A2 contient effectivement M1 tandis que $9E contient M1 + $80, mais ce n'est vrai que si M1 est une valeur positive. Quand M1 ▷= $80, $9E = M1.

SUITE DE L'EXPÉRIENCE :

Tapez **30A : 20**, puis **310 : 60**, **300G**... et **9D.A2**. Vous constatez que FAC ne correspond plus à rien. Ses valeurs ont été détruites. **Moralité :** ne pas essayer de décrémenter l'Exposant dans FAC... pour gagner du temps par exemple !

FAC VERS PAGE ZÉRO :

Comment copier FAC en page zéro, à une autre adresse ? Simplement en passant par la routine MOVML (**$EB23**), comme ceci :

```
300 :  A2 F9      LDX  £$F9
302 :  20 23 EB   JSR  $EB23
305 :  60         RTS
```

Si FAC contient **9D : 8C 8F 00 00 00 0F**, après exécution, on aura :
F9 : 8C 0F 00 00 00

FAC VERS UNE AUTRE ADRESSE :

Même facilité, mais l'adresse de destination doit être placée dans X (partie basse) et Y (partie haute) :

```
300 :  A2 08      LDX  £$08
302 :  A0 03      LDY  £$03
304 :  20 2B EB   JSR  $EB2B
307 :  60
```

De plus, on n'utilise plus MOVML, mais MOVMF (**$EB2B**). Après un **300G**, on aura (avec les valeurs actuelles de FAC) :
308 : 8C 0F 00 00 00

BASIC VERS FAC... ET RÉCUPÉRATION

```
300 :  20 BE DE   JSR  $DEBE
303 :  20 7B DD   JSR  $DD7B
306 :  A2 0E      LDX  £$0E
308 :  A0 03      LDY  £$03
30A :  20 2B EB   JSR  $EB2B
30D :  60         RTS
30E :  00 00 00 00 00
```

Dernier essai : tapez la courte routine LM ci-contre, puis repassez en BASIC. Pianotez alors un **A = 4000**, puis **CALL 768,A**. Après le **CALL -151**, vous constaterez que **30E.312** contiennent 8C 7A 00 00 00 C.Q.F.D. !

IMPORTANT : Lors de ces transferts FAC ne change pas.

Caractères souris

de votre Apple *IIc* ou *IIe**

Fonctionne aussi sur l'APPLE IIGS

Simple démonstration montrant comment afficher les caractères souris de l'Apple à partir d'une routine en langage machine.

En DOS 3.3, remplacer 325 : 4C 17 9A par 60 EA EA ou 4C EA 03.

* Cette routine ne fonctionne qu'avec les Apple *IIe* transformés mais aussi sur le GS en émulation.

```
10 DATA 32,0,195,169,145,32,240,253,162,6
   4,160,12,138,32,40,3,32,240,253,232,13
   8,132,36,32,40,3,32,240,253,32,98,252,
   232,224,96,144,231,76,23,154,9,128,32,
   240,253,169,189,32,240,253,169,160,32,
   240,253,138,96                                4DD2
20 FOR I = 768 TO 824: READ R: POKE I,R:
   NEXT                                          1E27
30 CALL 768                                      8331
40 VTAB 23: PRINT "<M>ENU DE DISQUETTE<c><t>G
    ";: GET R$: PRINT : IF R$ = "M" THEN
   PRINT  CHR$ (4)"RUN MENU"                     F369
```

LA LIGNE 10 (DATA)

```
0300-   20 00 C3    JSR   $C300     PR£3... si vous préférez.
0303-   A9 91       LDA   £$91      PRINT CHR$ (14)... Q
0305-   20 F0 FD    JSR   $FDF0     (40 colonnes, caractères élargis).
0308-   A2 40       LDX   £$40      Caractère de départ.
030A-   A0 0C       LDY   £$0C      Pour HTAB (deuxième colonne).
030C-   8A          TXA             X dans A vers la routine caractères normaux.
030D-   20 28 03    JSR   $0328
0310-   20 F0 FD    JSR   $FDF0     Affichage caractères souris.
0313-   E8          INX             X = X + 1 pour suivant, et dans A.
0314-   8A          TXA
0315-   84 24       STY   $24       CH = Y (HTAB, 2e colonne).
0317-   20 28 03    JSR   $0328     Même opération que plus haut.
031A-   20 F0 FD    JSR   $FDF0
031D-   20 62 FC    JSR   $FC62     CR : retour chariot.
0320-   E8          INX             X = X + 1 pour le suivant...
0321-   E0 60       CPX   £$60      mais seulement si X est plus petit que $60.
0323-   90 E7       BCC   $030C
0325-   4C 17 9A    JMP   $9A17     Retour au Basic.
```

CARACTÈRES NORMAUX

```
0328-   09 80       ORA   £$80      On force le bit 7 à 1 et on affiche le caractère
032A-   20 F0 FD    JSR   $FDF0     ASCII.
032D-   A9 BD       LDA   £$BD      Puis le signe égal...
032F-   20 F0 FD    JSR   $FDF0
0332-   A9 A0       LDA   £$A0      ...et un espace.
0334-   20 F0 FD    JSR   $FDF0
0337-   8A          TXA             X dans A, puis retour.
0338-   60          RTS
```

Yvan KOENIG

Protection sélective dans /RAM

Plusieurs lecteurs de *Tremplin Micro* m'ont demandé comment protéger une partie de la mémoire auxiliaire, lors de l'utilisation de /RAM (disque virtuel en AUX, sous ProDOS, pour ceux qui l'ignorent encore). APPLE dans la note technique PRODOS n°8 répondait qu'il suffisait de sauver un fichier bidon de 8K pour protéger la zone $2000-$3FFF et un second pour protéger la zone $4000-$5FFF (un unique fichier de 16K protègerait $2000-$5FFF). Rien n'est prévu pour des cas plus délicats et il faut se souvenir que cette formule occupe inutilement des pages mémoires puisque, chaque fichier BIN créé utilise 16 blocs pour les "données", plus 1 bloc "TSL". En désassemblant PRODOS, j'ai trouvé la routine de gestion du RAM disk et je dois reconnaître que je n'ai pas tout compris. Heureusement, Sandy MOSSBERG est venu à la rescousse dans NIBBLE d'Octobre 1986 (où il présente et commente soigneusement les routines liées au RAM disk).

Page	.0	.2	.4	.6	.8	.A	.C	.E
0.	*	*	*	*	*	*	*	*
1.	09	1A	2B	3C	4D	5D	5E	5F
2.	08	0A	0B	0C	0D	0E	0F	10
3.	11	12	13	14	15	16	17	18
4.	19	1B	1C	1D	1E	1F	20	21
5.	22	23	24	25	26	27	28	29
6.	2A	2C	2D	2E	2F	30	31	32
7.	33	34	35	36	37	38	39	3A
8.	3B	3D	3E	3F	40	41	42	43
9.	44	45	46	47	48	49	4A	4B
A.	4C	4E	4F	50	51	52	53	54
B.	55	56	57	58	59	5A	5B	5C
C.	*	*	*	*	*	*	*	*
D.	60	61	62	63	64	65	66	67
D.	68	69	6A	6B	6C	6D	6E	6F
E.	70	71	72	73	74	75	76	77
F.	78	79	7A	7B	7C	7D	7E	*

bank 1
bank 2

Pour ce qui nous intéresse ici, je ne retiendrai que la table de correspondance Numéros de blocs-Pages mémoire que je reproduis ci-contre en inversant le mode d'entrée (on connaît la page, on cherche le numéro de bloc correspondant).

En fait, un bloc est constitué de deux pages. La page de rang "pair" dont le numéro sert d'entrée à la table ci-contre, et la page de rang "impair" qui la suit.

EXEMPLE : Bloc $51 = pages $A8 + $A9

La table d'occupation du RAM disk est située en mémoire AUX de $3C2 à $3D1. Il suffit de manipuler cette table pour protéger ce que l'on souhaite. Je rappelle que dans cette table, chaque octet stocke la situation de 8 blocs, le bit correspondant à un bloc étant à 0 si le bloc est "utilisé", à 1 dans le cas contraire.

Par exemple, pour protéger $2000-$3FFF la liste des blocs "utilisés" est la suivante :

PAGE	20	22	24	26	28	2A	2C	2E	30	32	34	36	38	3A	3C	3E
BLOC	08	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18

Ce qui en terme de table d'occupation se traduit par : $3C3 = %01000000 $3C4 = %00000000 $3C5 = %01111111

(suite page 54)

Si on souhaite protéger $2000-$5FFF il faut faire :

$3C3 = %01000000 | $3C6 = %00000000
$3C4 = %00000000 | $3C7 = %00111111
$3C5 = %00100000

Si on souhaite protéger $6000-$7FFF (pseudo page 3 seule) il faut faire :

$3C7 = %11010000 | $3C8 = %00000000
$3C9 = %00011111

Récapitulation			
	1 seule	1 + 2	1 + 2 + 3
3C3	01000000	01000000	01000000
3C4	00000000	00000000	00000000
3C5	01111111	00100000	00100000
3C6		00000000	00000000
3C7		00111111	00010000
3C8			00000000
3C9			00011111
ce qui vous laisserait			
103 blocs		87 blocs	71 blocs

Si vous avez vraiment besoin de beaucoup de place il est possible de protéger tout ce qui se trouve en dessous de $C000 en marquant les blocs 00-5F occupés, ce qui se fait en mettant 00 de $3C3 à $3CD. Il vous restera 31 blocs en RAM disk.

Le programme assembleur PROTECT.RAM vous permet de réaliser les quatre configurations envisagées.

Il vous montre en prime une des solutions utilisables pour récupérer des informations dans la mémoire auxiliaire. Il fallait disposer de cette possibilité pour examiner l'état de la table d'occupation du volume, ce que fait la routine EXAMINE qui de plus, affiche les 16 octets lus. La routine FREEALL permet de vider complètement le RAM disk, toutes les zones "protégées" sont libérées et tous les fichiers éventuels sont "effacés". Le programme basic PROTECT.DEMO vous montre de manière succincte comment utiliser PROTECT.RAM.

Nota : Un PATCH pour ProDOS a été publié dans le numéro de novembre de NIBBLE, mais il n'est pas compatible avec le PROTECT.RAM d'Yvan KŒNIG. En effet, ce Patch déplace la table d'occupation du /RAM disk.

PROTECT.RAM

```
0300: 4C 61 03 4C 50 03 4C 3F 03 4C 30 03 4C 77 03 8D   3EAF
0310: 05 C0 A9 00 8D 25 0E A2 2B 9D 00 0E E8 D0 FA A9   A901
0320: FF A2 0E 9D C3 03 CA 10 FA A9 FE 8D D1 03 D0 43   3901
0330: 8D 05 C0 A9 00 A2 0A 9D C3 03 CA 10 FA 30 34 8D   6BCF
0340: 05 C0 A9 1F 8D C9 03 A9 00 8D C8 03 A9 10 D0 05   FA75
0350: 8D 05 C0 A9 3F 8D C7 03 A9 00 8D C6 03 A9 20 D0   2129
0360: 05 8D 05 C0 A9 7F 8D C5 03 A9 00 8D C4 03 A9 40   CDBA
0370: 8D C3 03 8D 04 C0 60 A2 09 B5 00 48 BD B4 03 95   7BB5
0380: 00 CA 10 F5 A9 C2 85 04 A9 03 85 05 A2 10 A0 00   5A4B
0390: 20 00 00 99 BE 03 20 DA FD A9 A0 20 ED FD C8 E6   F672
03A0: 04 D0 02 E6 05 CA D0 E8 20 8E FD 68 95 00 E8 E0   D8B3
03B0: 0A 90 F8 60 8D 03 C0 AD 00 10 8D 02 C0 60 00 00   30AE
03C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

BSAVE PROTECT.RAM, A$300,L206

PROTECT.RAM.DÉMO

```
10 D$ =  CHR$ (4): PRINT D$"PR£
   3": PRINT                          BD30
13 PRINT D$"BLOAD PROTECT.RAM"        1417
14 GOTO 20                            460D
15 PRINT D$"PR£1"                     3C5C
20 ROUT = 768                         F5BF
30 GOSUB 500                          1B45
40 CALL ROUT: REM "Protège HGR1       95D6
50 GOSUB 500                          1B45
60 CALL ROUT + 3: REM "Protège
```

```
    HGR1+HGR2                          17D1
 70 GOSUB 500                          1B45
 80 CALL ROUT + 6: REM "Protège
    HGR1+HGR2+HGR3                     1DD4
 90 GOSUB 500                          1B45
100 CALL ROUT + 9: REM "Protège
    presque tout                       23D7
110 GOSUB 500                          1B45
120 CALL ROUT + 15: REM "Libère
    RAMdisk                            5F04
130 GOSUB 500                          1B45
140 PRINT D$"BSAVE/RAM/BIDON1,A$
    2000,L$2000"                       DA63
150 GOSUB 500                          1B45
160 PRINT D$"BSAVE/RAM/BIDON2,A$
    2000,L$2000"                       8D64
170 GOSUB 500                          1B45
180 PRINT D$"DELETE /RAM/BIDON2"       7015
190 PRINT D$"BSAVE /RAM/BIDON1,A
    $2000,L$4000"                      5985
200 GOSUB 500                          1B45
210 CALL ROUT + 15: REM "Libère
    le RAMdisk                         5F04
220 GOSUB 500                          1B45
230 PRINT D$"PR£0"                     3A5B
240 END                                0180
500 PRINT D$"CAT /RAM"                 966D
510 CALL ROUT + 12: REM "Récupèr
    e table occupation volume          5501
530 GET X$: PRINT                      F92E
540 RETURN                             63B1
```

LE PROGRAMME SOURCE FIGURE SUR LA DISQUETTE TREMPLIN MICRO N°12.

BRIC-À-BRAC

• SAISIE FICHIER BINAIRE (TM n°11, page 24)

Lignes 240 et 250 : Permuter "1" et "2".

(D. P.)

• ENLÈVE.DOS (TM N°11, page 45)

Deux lignes erronées :

2760 LDX $2B	Récupère le numéro du slot actif lors du boot.
2762 LDA $C088,X	Arrête le lecteur avec lequel on vient de booter.

(Yvan KOENIG)

• CALENDRIER PERPÉTUEL

Suite au rectificatif du programme *Calendrier Perpétuel* (TM n°5) paru dans TM n°8, page 55, j'affirme que celui-ci est erroné. En effet, si la logique est bonne, (il suffit d'éliminer les années centenaires non bissextiles, c'est-à-dire non divisibles par 400), nous trouvons les années 1600, 1700, 1800, 1900, 2000 avec un mois de février correct, mais nous nous apercevons que les années qui devraient être bissextiles (par exemple 1980, 1984) ne le sont pas.

Exemple : 1984 : Année bissextile

INT (U/4) = U/4 → 1

INT (U/400) < > U/400 → 1

Résultat B = 0 donc année non bissextile ! La solution est simple. Il suffit d'adopter la modification de la ligne 85 (voir programme ci-après).

```
10  TEXT : NORMAL : PRINT  CHR$ (
    21); CHR$ (12): HOME
20  PRINT  CHR$ (4)"PR£1": PRINT
     CHR$ (27)"c"; CHR$ (9)"132N
    "; CHR$ (27)"Q"
50 U = 1599
51  FOR K = 1 TO 52
52  FOR L = 1 TO 8
84 FE = 28:U = U + 1
85 B = ( INT (U / 4) = (U / 4)) +
    ( INT (U / 100) <  > (U / 10
    0)) - ( INT (U / 400) <  > (
    U / 400)):FE = FE + B
90  POKE 36,L * 17 - 17: PRINT U"
          "FE;
95  NEXT L: PRINT " ": NEXT K
```

(C.A.)

• MORSE

Suite à la lettre de M. Christian G. de LIANCOURT (voir TM n°7, page 60), il existe en effet des programmes d'émission-réception Morse (CW) et Radiotélétype (RTTY) pour Apple *IIe*. Le plus connu d'entre eux est le "Super-Ratt" qui, sans interface, et à partir du connecteur pour manette de jeux, permet l'émission et la réception de ce type de signaux selon des standards multiples (codes Baudot, ASCII, vitesses variables). Ce programme permet également la mise en place d'une boîte aux lettres électronique (Mail Box) en RTTY et de nombreuses autres applications (stockage de textes, impression etc.)... L'Emission-réception RTTY et CW impose toutefois l'utilisation d'un codeur décodeur compatible TTL ou via un optocoupleur. De nombreuses descriptions de ces montages ont été faites dans la Presse radio amateur. Enfin, il faut posséder, pour l'émission ou la réception, une licence pour l'exploitation d'une station radio électrique d'amateur.

(E. M. 54420 PULNOY)

• CONSEILS AUX UTILISATEURS D'UN APPLE *IIc*

Pour rendre le programme "CONFIG" (TM n°9, pages 53-54) compatible avec tous les Apple *IIc*, il faut le modifier de la manière suivante :

```
BLOAD CONFIG,A$8000
CALL - 151
805C : 11 C3
BE00G
BSAVE CONFIG,A$8000,L$100
```

Luc MOREAU

• DISQUETTE KONICA

Les floppy disks Konica sont capables de conserver toutes vos données sur plus de 30 millions de passages. C'est du moins ce que garantit le fabricant pour des disquettes certifiées 100% sans erreur.

Tremplin Micro teste actuellement les 5 pouces 1/4 et les 3 pouces 1/2. Konica France se fera certainement un plaisir d'adresser sa documentation aux lecteurs de la revue. Voici ses coordonnées :
46 à 52 rue Arago
92800 PUTEAUX.

• DISQUETTE TDK

Une autre grande marque de disquettes est disponible sur le marché français : TDK. Nous en soumettons aussi quelques exemplaires à des tests d'utilisation très poussés.
TDK : Compagnie Electro Son,
41-43, rue de Villeneuve,
Silic 197, 94563 RUNGIS cedex.

• BIZARRE

Dans un programme Basic, pour entrer une commande DOS, il suffit de faire PRINT CHR$ (4);"COMMANDE DOS", mais quand on est en 80 colonnes, la précédente commande ne marche pas. J'aimerai savoir quelle commande ou quel POKE il faut faire pour que l'Apple accepte une commande DOS en 80 col.

(L. P. 42155 POUILLY-LES-NONAINS)

R Bizarre ! Avez-vous placé un PRINT à la suite du PRINT CHR$ (4)"PR£3" qui vous a permis de passer en 80 colonnes ? Après un GET, n'oubliez pas davantage de placer un PRINT.

• TASC et NUMÉRO DE VOLUME

1. Au secours ! comment obtenir de bons résultats, sous ProDOS, avec un programme compilé avec TASC et transféré par CONVERT ?

2. Comment changer le numéro de volume d'une disquette DOS 3.3 ?

(Anonyme).

R **1.** J'ai déjà eu l'occasion de dire qu'il est inutile de perdre son temps à transférer des programmes APPLESOFT compilés avec TASC ou un autre compilateur. Le BASIC.SYSTEM utilise la mémoire d'une façon différente de celle adoptée par le DOS 3.3, ce qui pose des problèmes. En outre, les commandes Système sont manipulées autrement, ce qui explique l'apparition, à l'écran, d'ordres non exécutés. Il n'y a pas de remède et je pense qu'il n'y en aura pas. La com-

R pilation perd une grande partie de son intérêt avec le *IIGS*.

2. Il n'est pas possible de modifier le numéro de volume d'une disquette DOS 3.3. Celui-ci est en effet inscrit sur toutes les pistes lors du formatage de la disquette. Il vous faudra initialiser une autre disquette en précisant le numVol choisi et y transférer les fichiers avec un utilitaire de votre choix (FID, COPY trucmuche...).

(Yvan KOENIG)

• OKI MICROLINE 80

Je possède un APPLE *IIe* (65C02) + interface parallèle APPLE + imprimante OKI microline 80. Pour la sortie de texte, pas de problème. Par contre, je ne parviens pas à imprimer les caractères semi-graphiques (codés de 128 à 256). Peut-être est-ce un problème de bit 7 à forcer à 1, mais je ne connais pas la marche à suivre ? De plus j'ai un utilitaire "OKI-GRAPH" qui, d'après la notice, permet à cette imprimante la recopie des pages graphiques. Hélas ! lui non plus ne fonctionne pas. ***(J.-P. C. 42210 UNIAS)***

R Pourquoi avoir acheté une carte APPLE pour utiliser une OKI microline 80 ? Il existe sur le marché des interfaces qui permettent d'exploiter à fond cette remarquable machine. La plus connue (et la plus imitée... pour ne pas dire copiée) est la carte GRAPPLER qui dispose d'une commande Ctrl I H pour forcer la reconnaissance du bit 8. Je rappelle que l'APPLESOFT force le bit 8 de tout ce qu'il envoie à 1 et que les interfaces s'empressent généralement de le mettre à 0.

MICROSHOP
6, rue de Châteaudun
75009 PARIS (1) 48.78.80.63
diffuse une "imitation" GRAPPLER à 595 F TTC (vérifier que le câble est compris), et vous pouvez sans doute trouver moins cher en consultant les publicités de l'OI (désolé : il n'y a pas encore de pub dans *Tremplin Micro*). ***(Yvan KOENIG)***

ATTENTION ! ***Nous ne répondons qu'aux lettres accompagnées d'une enveloppe timbrée. De plus, il ne nous est pas possible de recopier des listages pour vérifier le fonctionnement de la routine correspondante. La disquette est indispensable.***

AVEC LE CERCLE,

nous tournons vraiment en rond !

Voici la routine que vous propose monsieur PICHON :

Fonctionne aussi sur l'**APPLE *IIGS***

```
 20 HOME                                                    2F97
 40 ONERR  GOTO 400                                         7AE4
 50 XC = 140:YC = 90                                        570F
 70 TEXT                                                    1389
 80 INPUT " "RAYON ";R: IF R > 90 OR
    R < 2 THEN 70                                           96FD
 90 R2 = R ^ 2                                              7DA4
105 B =  INT (R * .75)                                      82EC
120 DIM A(B)                                                9F5A
130 HGR : HCOLOR= 3                                         F190
140 FOR I = 0 TO B                                          EFCD
145 A(I) = (R2 - I ^ 2) ^ .5                                87BF
146 NEXT : PRINT : VTAB 22                                  DFB6
148 INPUT " "CERCLE VIDE OU PLEIN (V/
    P) ";W$: IF W$ = "P" THEN 1000                          BDC1
150 FOR I = 1 TO B                                          F9CE
160 HPLOT XC + I,YC + A(I)                                  16AA
165 HPLOT XC + A(I),YC + I                                  61AA
170 HPLOT XC - I,YC + A(I): HPLOT XC
    - A(I),YC + I                                           5C90
180 HPLOT XC + I,YC - A(I): HPLOT XC
    - I,YC + A(I)                                           6C90
185 HPLOT XC - I,YC - A(I): HPLOT XC
    - A(I),YC - I                                           B092
186 HPLOT XC + A(I),YC - I                                  67AB
200 NEXT                                                    0582
300 END                                                     0180
400 FOR I = 768 TO 777: READ W: POKE
    I,W: NEXT                                               9A38
410 DATA 104,168,104,166,223,154,72,1
    52,72,96                                                E47F
420 TEXT : HOME                                             1C5A
500 PRINT " ERREUR "; PEEK (218) +  P
    EEK (219) * 256                                         931A
510 PRINT " "A(I)= ";A(I)                                   1AAC
520 END                                                     0180
1000 HPLOT XC - R,YC TO XC + R,YC TO X
    C,YC - R TO XC,YC + R                                   83CC
1005 ONERR  GOTO 400                                        7AE4
1010 FOR I = 1 TO B: HPLOT  TO XC + I,
    YC + A(I) TO XC + I,YC - A(I)                           994C
1020 HPLOT  TO XC - I,YC - A(I) TO XC
    - I,YC + A(I)                                           4E46
1030 HPLOT  TO XC - A(I),YC - I TO XC
    + A(I),YC - I                                           4746
1040 HPLOT  TO XC - A(I),YC + I TO XC
    + A(I),YC + I                                           4544
1200 NEXT                                                   0582
```

La micro et les livres

• APPLE II ProDOS, GUIDE DU PROGRAMMEUR

(EDITIONS DU SYBEX)
Parution fin décembre 1986
***Auteur :** Marcel COTTINI*

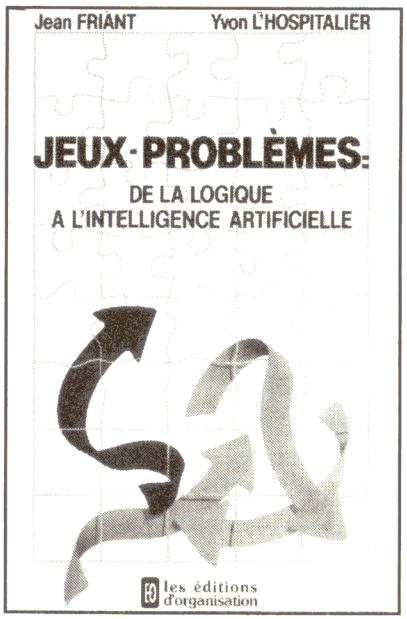
C'est la suite logique d'un précédent ouvrage, ainsi que le fruit d'une série d'articles et de réflexions de l'auteur. Depuis la sortie du système d'exploitation ProDOS, bon nombre de versions ont fait leur apparition. L'auteur, dans cet ouvrage, traite et réactualise ProDOS pour permettre au programmeur averti, comme au novice, d'effectuer une mise à niveau. Les versions 1.Ø.1, 1.Ø.2 et 1.1.1 sont traitées, la version 2.Ø n'ayant que très peu changé.

Vous apprendrez dans un premier temps à vous familiariser avec ProDOS, car l'accent est mis sur l'organisation d'une disquette ProDOS. Tout vous sera dévoilé au sujet du boot d'une disquette, ainsi que sur les moyens employés pour implanter les fichiers PRODOS et BASIC.SYSTEM. Les routines principales (MLI, REBOOT, ROM Contrôleur, Relocator, Loader) seront commentées sous forme de modules regroupant les sous-programmes et commandes diverses. Votre curiosité sera piquée par les listings détaillés et commentés des principales routines employées par ProDOS : ROM de la carte contrôleur, Loader du système ProDOS, PRODOS Relocator et REBOOT MLI. Cet ouvrage, outil de travail indispensable à tout programmeur et hobbyiste digne de ce nom, prédestine à une bonne maîtrise du système ProDOS et favorise une éventuelle transition entre l'Apple *IIe* et le *IIGS*, dernier-né d'Apple Computer.

• JEUX-PROBLÈMES : de la logique à l'intelligence artificielle*

Si vous vous intéressez à l'intelligence artificielle, je vous promets de bons, d'excellents moments avec le livre de Jean FRIANT et d'Yvon L'HOSPITALIER. Les auteurs désirent initier leurs lecteurs à une sérieuse formation dans le domaine de la déduction logique. Pour y parvenir, ils utilisent des jeux et énigmes accompagnés de leurs solutions. Petit à petit, on apprend à formaliser, schématiser, modéliser, raisonner et généraliser (je reprends à dessein les grandes lignes de l'ouvrage). Peut-on parler d'informatique de la cinquième génération ? Peut-être. Toujours est-il que, sans être un fort en maths, l'auteur de ces lignes a pris un réel plaisir à découvrir des modèles de déduction logique dont il tirera sans doute le plus grand profit. Un livre à s'offrir et à étudier plutôt qu'à lire.

* **Les Editions d'organisation** 5, rue Rousselet — Paris 7e.



• APPLE II ProDOS, GUIDE DU PROGRAMMEUR APPLESOFT

(EDITIONS TREMPLIN MICRO)
Parution fin décembre 1986
***Auteur :** Marcel COTTINI*

Comme le Hard, le Soft subit les mêmes règles de contrainte et d'évolution technologique imposées par l'homme, seul maître après Dieu de la "machine". Parmi le Soft se trouvent des programmes très particuliers appelés programmes système ou Disk Operating System (DOS). Ce sont des sous-programmes et routines qui forment un ensemble cohérent capable de gérer une configuration micro-informatique donnée. Le DOS 3.3 fut en son temps un des pionniers des programmes système. Plus près de nous, le système ProDOS, Professional Disk Operating System, a été élaboré pour trouver son application dans le milieu professionnel, avec l'utilisation d'un disque dur du type Profile. Très vite, Apple Computer a adapté la version professionnelle à la série des Apple *II*, avec tout d'abord un système ProDOS qui tournait sur une configuration de 48 Ko. Par la suite, la version d'origine a été modifiée et limité aux modèles Apple *II* ayant une RAM de 64 Ko minimum. ProDOS, système d'exploitation de remplacement du DOS 3.3, sollicite de plus en plus le hobbyiste, qu'il soit simple utilisateur de soft ou programmeur averti. C'est le lien entre le cœur de la "machine" (le microprocesseur) et son environnement (les périphériques).

APPLE II ProDOS, GUIDE DU PROGRAMMEUR APPLESOFT, est un ouvrage destiné à donner au lecteur tous les moyens élémentaires pour une meilleure compréhension de certains mécanismes internes du système. Son objectif premier est de familiariser le programmeur pratiquant l'AppleSoft sous ProDOS, en traitant plus particulièrement le Basic System. Le programmeur en langage machine ne sera pas oublié, bon nombre d'éléments se trouvant en bonne place dans ce livre. Par ailleurs, l'Apple *IIGS* tournant sous un système ProDOS revu et adapté au microprocesseur 16 bits, il est vivement conseillé de maîtriser totalement ProDOS/8 bits avant de se lancer dans toute nouvelle aventure.

ProDOS/8 bits est un système spécifique au microprocesseur 8 bits (65Ø2 et 65CØ2). ProDOS/16 bits est un système destiné à l'Apple *IIGS* équipé d'un microprocesseur 16 bits (65C816). Les deux versions de ProDOS diffèrent dans bien des domaines tout en gardant une certaine compatibilité au moment de la commutation en mode émulation. Ce livre est donc parfaitement adapté à tous ceux qui désirent rester fidèles à la gamme des Apple *II*, tout en préparant le lecteur à un passage en douceur, d'un type de "machine" à un autre.

(Bulletin de commande page 67).

Yvan KŒNIG répond à nos lecteurs

• COPIE EN /RAM ET ZÉRO BARRÉ

Questions : *Comment copier en /RAM certains fichiers d'une disquette ProDOS ? Comment obtenir le zéro barré avec une imprimante IMAGEWRITER ?* ***(P.P. 69006 LYON)***

R Pour copier des fichiers de tous types dans /RAM il existe au moins deux programmes performants sur le marché. De ce fait je ne suis guère tenté de refaire ce qui est déjà disponible. Il y a suffisamment à faire par ailleurs ! NIBBLE a publié ProDOS COPY dans son numéro de décembre 1985. Ce programme (plus quelques autres) est disponible sur le disque "ProDOS Library 2" au prix de $29.95 + $2.50 pour le port. Personnellement je préfère PROCMD de Glen BREDON distribué par Call A.P.P.L.E. Ce produit regroupe une trentaine de commandes ajoutables à ProDOS. Lors de son introduction sur le marché ce produit était vendu aux adhérents de Call A.P.P.L.E. $35 plus $9.52 pour le port avion. J'ignore quel est son prix actuellement.

Adresses :

NIBBLE
45 Winthrop St., Concord, MA 01742, USA
Call A.P.P.L.E.
2890 S.W. 43 rd St., Renton, WA 98055, USA

Pour les zéros barrés, je ne sais quel manuel vous avez. Il y a une erreur dans le manuel IMW+ *IIc*. Je reproduis ci-dessous un petit programme qui vous donne deux modes d'accès.
La formule "€ALL" n'est pas indispensable ici mais elle est utile pour d'autres séquences de commande qui ne peuvent être envoyées directement depuis le Basic, soit qu'elles ne fonctionnent pas dans ce cas, soit qu'elles génèrent des effets parasites sur les sauts de ligne par exemple.

```
10 D$ = CHR$(4) : E$ = CHR$(27) : PRINT D$"BLOAD BARREUR" : PRINT D$"PR£1"
20 PRINT E$"D" CHR$(0) CHR$ (1);"0000000": REM zéros barrés
30 PRINT "OOOOOOO";:REM ce sont des lettres O
40 PRINT E$"Z" CHR$(0) CHR$(1);"0000000": REM zéros classiques
50 PRINT
60 CALL 768 : PRINT "00000 OOOOO ";: REM 5 chiffres et 5 lettres
70 CALL 768+3 :PRINT "00000"
80 PRINT D$"PR£0"
```

BARREUR COUT = $FDED

```
300 :  A2  44      LDX  £'D'
302 :  2C          HEX  2C      Pour sauter instruction suivante.
```

R

```
303 :  A2  5A      LDX  $'Z'
305 :  A9  1B      LDA  £$1B    Escape.
307 :  20  ED FD   JSR  COUT
30A :  8A          TXA          Récupère le caractère spécifique.
30B :  20  ED FD   JSR  COUT
30E :  A9  00      LDA  £0
310 :  20  ED FD   JSR  COUT
313 :  A9  01      LDA  £1
315 :  4C  ED FD   JMP  COUT
```

BSAVE BARREUR,A$300,L$18

Question : *J'ai suivi à la lettre vos indications, mais la routine que vous m'avez fait parvenir n'a donné aucun résultat. L'imprimante n'a sorti aucun caractère : ni alphabétique, ni semi-graphique. Je pense donc que le mieux est de me procurer la carte GRAPPLER, comme vous me le conseillez, sachant que j'avais acheté l'ensemble OKI microline 80 et interface parallèle d'occasion à une personne qui n'avait ni notice ni explications à me donner sur le fonctionnement. Pourriez-vous donner aux lecteurs de TREMPLIN MICRO qui n'ont pas les moyens de s'offrir l'IMAGEWRITER II, des conseils sur le choix d'une imprimante graphique meilleur marché fonctionnant sans trop de problèmes sur APPLE ?* ***J.-P. C. (42210 UNIAS)***

R Le choix d'une imprimante est fort large. Les IMAGEWRITER elles-mêmes posent parfois des problèmes car elles recèlent encore quelques bugs. En fait les ennuis sont généralement liés aux interfaces. GRAPPLER existe en version parallèle mais aussi, depuis un an en version série (je ne l'ai pas essayée). Sachez que, quel que soit votre choix, vous vous trouverez forcément face à des petits problèmes irritants à résoudre. La carte SSC, interface série officielle APPLE se comporte, sur plusieurs points, autrement que la sortie série du *IIc*, ce qui n'est pas de tout repos. La carte ALPHABITS de STREET ELECTRONICS est nettement plus proche des caractéristiques de la sortie du *IIc*, mais comme il n'y a pas d'importateur c'est le cirque en cas de pépin. Il peut aussi y avoir des difficultés de relation logiciel-imprimante.
APPLEWRITER se révèle incapable de traiter correctement les séquences permettant de basculer du Français à l'Allemand et d'en revenir de même qu'il n'est pas possible de charger des caractères personnalisés depuis le programme. Je suis sur le point de terminer un ensemble de patchs qui règlera tous ces problèmes mais çà ne devrait pas être commercialisé avant 1987. ■

Et encore...

• DMP UTILITIES

M. L.P. (42155 POUILLY-LES-NONAINS) informe dans la rubrique "Du coq à l'âne" de *T.M.* n°11 de la possibilité d'utiliser "DMP UTILITIES". Je suis en plein accord avec lui, d'autant qu'il est très facile de se procurer ce logiciel auprès de :

RÉSEAU PLANÉTAIRE
LES NOUVEAUX LOGICIELS
BP 3
43260 ST JULIEN CHAPTEUIL
Tél. : 71.08.73.49

En espérant que cette information intéressera de nombreux fervents, comme moi, de votre revue. ***G.R. (77240 CESSON)***

• SAISIE LM ASSISTÉE

Dans le programme «Saisie LM assistée» de *T.M.* n°3, à la ligne n°1040, il est préférable de mettre «1040 CALL — 198: POKE — 16368,0... etc.» plutôt que POKE — 16398,0. ***(X. à PERPIGNAN)***

• LA BOÎTE AUX IDÉES

Je vous écris à propos du programme BOÎTE.BAS paru dans *T.M.* n°8. Quand on exécute son programme, on remarque rapidement que la recherche ne s'effectue pas sur la dernière chaîne de caractères entrée. Cela semble dû aux valeurs poquées en 8 et 9 (ligne 535). En effet, la valeur finale est toujours inférieure d'une unité (Quand N sera égal à 256 on poquera en 8 la valeur - 1, ce qui provoquera vraisemblablement un ILLEGAL QUANTITY ERROR).
Le remède à ce défaut est très simple : remplacer tous les (N -1) par N. La ligne 535 devient :

```
535 L = LEN (M$) : POKE 252,L :
    POKE 8, N - INT(N/256) * 256
    : POKE 9, INT(N/256) : M = 0
```

M. H. (78260 ACHÈRES)

• RUBANS SCRIBE

Pour ceux qui cherchent des rubans SCRIBE :

PRÉCISION Data Products
P.O. Box 8367
Grand Rapids, MI 49518 U.S.A.

$2.95 pièce, par 6 minimum. Ecrire pour port et emballage.

• L'ORDINATEUR POUR MIEUX VOIR

Ce nouveau système informatique de traitement d'images mesure images et échantillons avec une puissance analytique et une précision inégalées à ce jour pour un coût relativement aussi peu élevé. D'emploi très souple, ce C-VAS 3000 se sert d'une technologie mise au point à l'Unité d'analyse d'images Wolfson de l'université de Manchester (Grande-Bretagne) qui en fait un puissant instrument d'inspection visuelle pour une vaste gamme d'applications industrielles et de recherche.

Notre photo montre une caméra de télévision de 625 lignes dirigée sur un circuit imprimé afin d'en donner de plus nombreux détails sur un écran de visualisation en couleur. On peut relier la caméra à une macro-visionneuse pour l'examem d'images photographiques venant de nombreuses sources. La caméra peut également être reliée à un microscope pour l'analyse directe de toute matière biologique ou autre.

Le système de base se compose d'un terminal et d'un clavier, d'une machine de traitement d'images spécialisée et d'un écran de visualisation couleurs de haute définition avec «souris» de graphiques. Sont offerts en option :

— trois terminaux supplémentaires ;

— une imprimante ultra-rapide et différents systèmes de liaison avec diverses sources vidéo et numériques.

La souplesse d'emploi vient de l'adoption d'un ensemble de base d'«outils» de logiciels qui simplifient grandement le travail de programmation.

VISUAL MACHINES LIMITED,
Enterprise House, Manchester Science Park, Lloyd Street North, Manchester M15 4EN, Angleterre.
Tél. : 061 226 8666.

Lorsque vous écrivez à Tremplin Micro, n'oubliez pas de joindre une enveloppe timbrée pour la réponse.

JEU VISUEL

INITIATION

Mémo-visu

GRAPHISME : HGR.

LANGAGE : Basic.

MATÉRIEL : Apple avec moniteur couleur. Si l'on dispose d'un Apple *IIe* avec carte EVE, on obtiendra un fond couleur en mode texte (ligne 100). Avec la carte Féline (ou l'Apple *IIc*), l'instruction sera sans effet.

OBJECTIF : Développer la mémoire visuelle en associant celle-ci à une rapide perception des couleurs.

DÉROULEMENT :

Le programme affiche 8 carrés dans un ordre aléatoire et également colorés de façon aléatoire.
Il s'agit, pour le joueur, d'indiquer dans quel ordre les carrés sont apparus sur l'écran.

• HCOLOR •

0 NOIR1	4 NOIR2
1 VERT	5 ORANGE
2 VIOLET	6 BLEU
3 BLANC1	7 BLANC2

LA PAGE HGR DE L'APPLE

HGR

C'est l'abréviation de HIGH (haut)... et de GRAPHIC (graphique). Quand on tape **HGR**, en mode direct, on passe immédiatement en mode graphique haute résolution et l'on dispose d'une page de 280 points de large sur 160 points de haut (0 à 279 et 0 à 159). Une fenêtre texte de 4 lignes subsiste sous l'écran graphique. En mémoire, la page HGR (page 1) occupe l'emplacement $2000-3FFF (de 8192 à 16383) .

EXTENSION

Un **POKE - 16302,0** (ou **49234,0**) permet d'étendre HGR à tout l'écran en supprimant la fenêtre texte. On dispose alors de 192 points en hauteur, au lieu de 160.
POKE -16301,0 (**49235,0**) fait instantanément réapparaître la fenêtre texte sans détruire la page graphique.

HCOLOR

8 couleurs sont disponibles (de 0 à 7 ; voir la palette ci-contre). Pourquoi 2 noirs et 2 blancs ? Simplement parce que l'épaisseur du trait noir ou blanc est moins importante que celle des autres couleurs.

Dans les colonnes paires, on a le violet et le bleu.

Dans les colonnes impaires, le vert et l'orange.

Il est indispensable d'initialiser la couleur avant le premier HGR. Si on ne se soumet pas à cette petite formalité, on risque d'être en noir... et d'avoir un écran vide.

(Suite page 62)

```
100 PRINT  CHR$ (21): CLEAR : POKE  - 1619
    9,16 * 14 + 9: HOME : REM Carte Chat M
    auve, carac orange sur fond bleu turqu
    oise                                          E7A6
105 GOSUB 235                                     154A
110 A = 4:B = 65                                  FFFC
115 POKE 230,32: HCOLOR= 3: HPLOT 0,0: CAL
    L - 3082                                      9D93
120 POKE 49236,0: POKE 49234;0: POKE 49239
    ,0: POKE 49232,0                              D21F
125 FOR J = 1 TO 8:R =  PEEK (767 + J)            9E0A
130 HC = 1 +  INT ( RND (1) * 6): IF HC =
    3 THEN 130                                    87A2
135 IF R < 5 THEN H = (R - 1) * 70:V = 8:
    GOTO 145                                      D4C6
140 V = 84:H = (R - 5) * 70                       D1B6
145 HC(R) = HC                                    8D89
150 HCOLOR= HC: GOSUB 225: GOSUB 345              A926
155 NEXT J                                        A3CC
160 GOSUB 265                                     E74D
165 POKE 49235,0: HOME :J = 0:PT = 0              C74F
170 J = J + 1                                     935D
175 VTAB 22: PRINT "QUEL A ETE LE CARRE NU
    MERO ";: INVERSE : PRINT J;: NORMAL :
    PRINT " ";: CALL  - 198: GET R$: PRINT
    R$                                            0C27
180 IF  PEEK (J + 767) =  ASC (R$) - 64 TH
    EN VTAB 24: CALL  - 868:PT = PT + 1: P
    RINT "EXACT: NOTE = "PT;: GOTO 190            CCAA
185 VTAB 24: CALL  - 868: PRINT "INEXACT -
     BONNE REPONSE: " CHR$ ( PEEK (J + 767
    ) + 64);                                      6284
190 VTAB 21: PRINT : IF J < 8 THEN 170            718F
195 GOSUB 250                                     E747
200 VTAB 22: CALL  - 958: PRINT "VOTRE SCO
    RE EST ";: INVERSE : PRINT PT: NORMAL         064D
205 VTAB 24: PRINT "(1) UNE AUTRE PARTIE -
     (0) TERMINE ->ctG ";: GOSUB 250              53E0
210 IF R = 49 THEN 100                            5191
215 IF R <  > 48 THEN 205                         5566
220 TEXT : POKE  - 16200,16 * 0 + 15: HOME
    : END : REM Carte Chat Mauve:écran noi
    r,caractères blancs sur fond noir             2C84
225 FOR I = 0 TO B: HPLOT A + H,V + I TO B
    + H,V + I: NEXT                               BAE0
230 RETURN                                        63B1
235 FOR I = 768 TO 775: POKE I,I - 767: NE
    XT                                            827D
240 FOR I = 0 TO 7:R =  INT ( RND (1) * 7)
    :T =  PEEK (768 + I): POKE 768 + I, PE
    EK (768 + R): POKE 768 + R,T: NEXT            2E9C
245 RETURN                                        63B1
```

100

Passage en mode 40 colonnes par PRINT CHR$ (21).

115

POKE 230,32 permet d'écrire dans HGR (pour HGR2 : POKE 230,64). Cette ligne efface l'écran avec la couleur 3. Cela permet d'avoir un écran propre quand on arrive à HGR.

120

Passage en mode HGR sans effacement d'écran et sans la fenêtre de texte.

130

Choix aléatoire de la couleur. On exclut le noir 0 et le blanc 3, mais il reste le noir 4.

165

POKE 49235,0 permet de retrouver les 4 lignes de texte.

IMPORTANT

Ne tapez pas les chiffres HEXA en bout de ligne : ils sont destinés à vérifier la copie de votre programme en utilisant notre disquette SIGNATURE.

235

Numéros de 1 à 7 stockés de 768 à 775.

240

Mélange aléatoire de ces numéros.

250

LIGNE D'ATTENTE. Souvent utilisée dans *TREMPLIN MICRO*. On trouve dans R la valeur ASC de la touche pressée.

ÉCRITURE HGR

Cette partie du programme vous montre comment écrire facilement (!) — et en Basic — dans une page graphique haute résolution.
La longueur maximum de la variable A$ est 255 caractères. Si vous la respectez, vous pouvez dessiner n'importe quoi. Attention ! en couleur, doublez l'épaisseur des traits verticaux, comme ici.

```
250 CALL  - 198: POKE  - 16368,0: WAIT  -
    16384,128,127: POKE  - 16368,0: VTAB 2
    1: PRINT :R =  PEEK ( - 16384): RETURN   D6FE
255 :                                        003A
260 REM ABCDEFGH
265 XO =  - 15:Y = 14                        7D6E
270 FOR I = 1 TO 8:X = XO + 70 * I: IF I >
    4 THEN Y = 90:X = XO + 70 * (I - 4)      E231
275 HCOLOR= 3: IF HC(I) = 6 OR HC(I) = 5 T
    HEN HCOLOR= 7                            D25C
280 ON I GOSUB 360,420,480,540,600,660,720
    ,780                                     C1AF
285 NEXT                                     0582
290 :                                        003A
295 RETURN                                   63B1
300 :                                        003A
305 :                                        003A
310 REM ECRITURE DU TEXTE<ctJ>
315 FOR H = 0 TO 7                           E0C1
320 FOR L = 1 TO 7                           43C6
325 IF  MID$ (A$,L + H * 7,1) = "*" THEN
    HPLOT X + L,Y + H                        A5C9
330 NEXT : NEXT                              143E
335 :                                        003A
340 REM ESPACE<ctJ>
345 FOR BR = 1 TO 10:BZZZ =  PEEK (49200):
    NEXT : RETURN                            836B
350 :                                        003A

355 REM A<ctJ>
360 A$ = ""                                  EE79
365 A$ = A$ + "  "**  "  "                   BB9A
370 A$ = A$ + "  **  **  "                   42AE
375 A$ = A$ + "**  "  "**"                   6CAE
380 A$ = A$ + "**  "  "**"                   6CAE
385 A$ = A$ + "********"                     59CC
390 A$ = A$ + "**  "  "**"                   6CAE
395 A$ = A$ + "**  "  "**"                   6CAE
400 :                                        003A
405 GOSUB 310: RETURN                        972F
410 :                                        003A
415 REM B<ctJ>
420 A$ = ""                                  EE79
425 A$ = A$ + "******  "                     FCC2
430 A$ = A$ + "**  "  "**"                   6CAE
435 A$ = A$ + "**  "  "**"                   6CAE
440 A$ = A$ + "******  "                     FCC2
445 A$ = A$ + "**  "  "**"                   6CAE
450 A$ = A$ + "**  "  "**"                   6CAE
455 A$ = A$ + "******  "                     FCC2
460 :                                        003A
465 GOSUB 310: RETURN                        972F
470 :                                        003A

475 REM C<ctJ>
480 A$ = ""                                  EE79
485 A$ = A$ + "  *****  "                    76B8
490 A$ = A$ + "**  "  "**"                   6CAE
495 A$ = A$ + "**  "  "  "                   FC9A
500 A$ = A$ + "**  "  "  "                   FC9A
505 A$ = A$ + "**  "  "  "                   FC9A
510 A$ = A$ + "**  "  "**"                   6CAE
515 A$ = A$ + "  *****  "                    76B8
520 :                                        003A
525 GOSUB 310: RETURN                        972F
530 :                                        003A
535 REM D<ctJ>
540 A$ = ""                                  EE79
545 A$ = A$ + "******  "                     FCC2
550 A$ = A$ + "**  "  "**"                   6CAE
555 A$ = A$ + "**  "  "**"                   6CAE
560 A$ = A$ + "**  "  "**"                   6CAE
565 A$ = A$ + "**  "  "**"                   6CAE
570 A$ = A$ + "**  "  "**"                   6CAE
575 A$ = A$ + "******  "                     FCC2
580 :                                        003A
585 GOSUB 310: RETURN                        972F
590 :                                        003A
```

(Suite page 64)

MEMO-VISU

SUITE

```
595 REM E                                
600 A$ = ""                          EE79
605 A$ = A$ + "********"             59CC
610 A$ = A$ + "**      **"           FC9A
615 A$ = A$ + "**      **"           FC9A
620 A$ = A$ + "******  **"           6BB8
625 A$ = A$ + "**      **"           FC9A
630 A$ = A$ + "**      **"           FC9A
635 A$ = A$ + "********"             59CC
640 :                                003A
645 GOSUB 310: RETURN                972F
650 :                                003A
655 REM F
660 A$ = ""                          EE79
665 A$ = A$ + "********"             59CC
670 A$ = A$ + "**      **"           FC9A
675 A$ = A$ + "**      **"           FC9A
680 A$ = A$ + "******  **"           6BB8
685 A$ = A$ + "**      **"           FC9A
690 A$ = A$ + "**      **"           FC9A
695 A$ = A$ + "**      **"           FC9A
700 :                                003A
705 GOSUB 310: RETURN                972F
```

```
710 :                                003A
715 REM G
720 A$ = ""                          EE79
725 A$ = A$ + "  ******  "           76B8
730 A$ = A$ + "**    **"             6CAE
735 A$ = A$ + "**      **"           FC9A
740 A$ = A$ + "**  ******"           ADB8
745 A$ = A$ + "**    **"             6CAE
750 A$ = A$ + "**    **"             6CAE
755 A$ = A$ + "  ******  "           76B8
760 :                                003A
765 GOSUB 310: RETURN                972F
770 :                                003A
775 REM H
780 A$ = ""                          EE79
785 A$ = A$ + "**    **"             6CAE
790 A$ = A$ + "**    **"             6CAE
795 A$ = A$ + "**    **"             6CAE
800 A$ = A$ + "********"             59CC
805 A$ = A$ + "**    **"             6CAE
810 A$ = A$ + "**    **"             6CAE
815 A$ = A$ + "**    **"             6CAE
820 :                                003A
825 GOSUB 310: RETURN                972F
```

CE PROGRAMME FONCTIONNE SUR APPLE *IIe* et APPLE *IIc*

INITIATION

TROIS petits airs

LE ROI DAGOBERT

La routine **SON** se cache dans les lignes 20 et 25.

Après avoir demandé à votre Apple d'interpréter l'air (arrangé par Jean Perrot) du **Roi Dagobert**, tapez un **BSAVE SON, A$300,L23** et vous disposerez de la réplique, en programme BIN, des lignes 20 et 25...

Quand **SON** est en mémoire, il est inutile de le réinstaller pour jouer un autre air... comme vous le prouve **Frère Jacques**...

```
10 PRINT  CHR$ (4);"PR£3": PRINT                          7091
15 HOME : VTAB 10: HTAB 30: PRINT "LE BON
    ROI DAGOBERT"                                         0541
20 POKE 770,173: POKE 771,48: POKE 772,19
   2: POKE 773,136: POKE 774,208: POKE 77
   5,5: POKE 776,206: POKE 777,1: POKE 77
   8,3: POKE 779,240: POKE 780,9: POKE 78
   1,202                                                  953D
25 POKE 782,208: POKE 783,245: POKE 784,1
   74: POKE 785,0: POKE 786,3: POKE 787,7
   6: POKE 788,2: POKE 789,3: POKE 790,96                 A716
30 READ I,J: IF I = 0 THEN 45                             48A3
35 POKE 768,I: POKE 769,J: CALL 770                       5646
40 GOTO 30                                                400E
45 VTAB 22: PRINT "ENCORE UNE FOIS ? ";:
   GET R$: PRINT : IF R$ = "O" OR R$ = "o
   " THEN  VTAB 22: CALL  - 868: RESTORE
   : GOTO 30                                              5905
50 HOME : END                                             8E51
55 DATA 102,128,102,255,115,128,115,255,1
   28,128,128,255,1,16,115,255,1,16,102,1
   28,96,128,102,128,115,128,128,128,115,
   128,128,255,1,16,128,128,128,128,115,1
   28                                                     6536
60 DATA 102,255,102,128,102,128,96,128,86
   ,128,115,255,115,128,115,128,128,128,1
   15,128,102,255,102,128,102,128,96,128,
   86,128,115,255,115,128,115,255                         AE5A
65 DATA 1, 50,102,128,102,255,115,128,115
   ,255,128,128,128,255,1,32,128,255,102,
   128,96,128,102,128,115,128,128,128,115
   ,128,128,255,0,0                                       1998
```

```
10 HOME : VTAB 10: POKE 1403,33: PRINT "F
   RERE JACQUES"                                          69F9
15 READ I,J: IF I = 0 THEN 30                             309D
20 POKE 768,I: POKE 769,J: CALL 770                       5646
25 GOTO 15                                                4611
```

FRÈRE JACQUES

(suite au verso)

TROIS PETITS AIRS

Ne manquez pas de lire, page 14, l'excellent article de Roland JOST (ORGUE SUR APPLE).

```
30 VTAB 22: PRINT "ENCORE UNE FOIS ? ";:
   GET R$: PRINT : IF R$ = "O" OR R$ = "o
   " THEN  VTAB 22: CALL  - 868: RESTORE
   : GOTO 15                                  8308
35 HOME                                       2F97
40 DATA 144,128,128,128,115,128,144,128,1
   44,128,128,128,115,128,144,128,115,128
   ,102,128,96,255,115,128,107,128,96,255
   ,96,64,86,38,96,68,102,68                  2092
45 DATA 115,128,144,128,96,68,86,38,96,38
   ,102,68,115,128,144,128,128,128,192,12
   8,144,255,128,128,192,128,144,255,0,0
                                              56C0
```

AU CLAIR DE LA LUNE

Ici, Jean Perrot appelle la routine **SON** par un BLOAD, mais si vous utilisez le programme à la suite des deux autres, la ligne 20 devient superflue.

```
10 PRINT  CHR$ (4);"PR£3": PRINT              7091
15 VTAB 10: HTAB 30: PRINT "AU CLAIR DE L
   A LUNE"                                    C339
20 PRINT  CHR$ (4);"BLOAD SON"                1E17
25 READ I,J: IF I = 0 THEN 40                 429E
30 POKE 768,I: POKE 769,J: CALL 770           5646
35 GOTO 25                                    5012
40 VTAB 22: PRINT "UNE AUTRE FOIS ? ";: G
   ET R$: PRINT : IF R$ = "O" OR R$ = "o"
   THEN  VTAB 22: CALL  - 868: RESTORE :
   GOTO 25                                    E0CE
45 HOME                                       2F97
50 DATA 96,128,96,128,96,128,86,128,76,25
   5,86,255,96,128,76,128,86,128,86,128,9
   6,255,1,128,96,128,96,128,96,128,86,12
   8,76,255,86,255,96,128,76,128,86,128,8
   6,128,96,255,1,128                         9601
55 DATA 86,128,86,128,86,128,86,128,115,2
   55,115,255,86,128,96,128,102,128,115,1
   28,128,255,1,128,96,128,96,128,96,128,
   86,128,76,255,86,255,96,128,76,128,86,
   128,86,128,96,255,0,0                      3660
```

Bientôt, avec l'Apple *IIGS* et son processeur ENSONIQ, les mélomanes pourront s'en donner à cœur joie, mais bon nombre de compositeurs utilisent déjà le clavier de leur Apple pour créer de petits airs.

TREMPLIN MICRO aimerait en regrouper une centaine sur une disquette d'amis distribuée à prix coûtant (20 F + frais d'envoi).

A vous de jouer, amie Lectrice, ami Lecteur ! *

G.-H.

** Programmes sur disquette exclusivement (retournée après utilisation).*

N°12

Bulletin de commande et d'abonnement

A retourner à : **TREMPLIN MICRO — Senillé — 86100 CHATELLERAULT**

Nom ..... Prénom .....

Adresse complète .....

Code postal |_|_|_|_|_| Ville .....

LES DISQUETTES FONCTIONNENT SOUS DOS ET ProDOS (à condition de posséder une version Apple de ce SYSTEME D'EXPLOITATION)

QUANTITÉ	DÉSIGNATION	PRIX UNITAIRE	TOTAL
	Abonnement à 6 numéros (un an) : FRANCE à partir du numéro : _____	190 F	
	Abonnement à 6 disquettes (un an) : FRANCE à partir du numéro : _____	600 F	
	MINIE LE BUREAU DE MINIE LES SOURCES DU BUREAU — Logiciels de Claude AUBRY	180 F 150 F 50 F	
	DISQUETTE UTILITY-DISK (Marcel COTTINI)	170 F	
	LIVRE : APPLE // PRODOS, GUIDE DU PROGRAMMEUR APPLESOFT	120 F	
	LIVRE : LE 65Ø2 PAS à PAS	120 F	
	LIVRE + DISQUETTE : CLINS D'ŒIL AU 65Ø2	160 F	
	LIVRE + DISQUETTE : ROUTINES LM POUR 65CØ2 ET 65Ø2	160 F	
	RELIURE-ÉCRIN	40 F	
	DISQUETTE-INDEX (N°1 à 6)	30 F	
	DISQUETTE-INDEX (N°7 à 12)	30 F	
	SIGNATURE (CHECK-LIST)	30 F	
	DISQUETTE DE TREMPLIN MICRO : * *(IIe et IIc)* N°1 ☐ N°4 ☐ N°7 ☐ N°10 ☐ N°2 ☐ N°5 ☐ N°8 ☐ N°11 ☐ N°3 ☐ N°6 ☐ N°9 ☐ N°12 ☐	105 F	
	NUMÉRO DE TREMPLIN MICRO : * N°1 ☐ N°2 ☐ N°3 ☐ N°4 ☐ N°5 ☐ N°6 ☐ N°7 ☐ N°8 ☐ N°9 ☐ N°10 ☐ N°11 ☐ N°12 ☐	 30 F 33 F	

* *Cochez la (ou les) case(s) de votre choix.*

Participation aux frais d'envoi (gratuit pour les abonnés) + 10 F

Envoi en paquet-poste recommandé à partir de 400 F

Numéro d'abonné ou de client : _____ **Total à payer** _____ **F**

Merci de libeller votre règlement à l'ordre de TREMPLIN MICRO / Editions JIBENA.

Mode de règlement choisi : ☐ Chèque ☐ Mandat-lettre ☐ Carte Bleue

VISA

N° de votre Carte Bleue |_|_|_|_| |_|_|_|_| |_|_|_|_| |_|_|_|_|

Date d'expiration |_|_|_|_| **Signature** (obligatoire)

Montant à débiter _____ **F**

vous intéresse... Ceci vous intéresse... Ceci

• *JOUER AVEC LES MATHS** *par Henri Camous*

Professeur de mathématiques appliquées, Henri CAMOUS réussit à nous donner, dans cet ouvrage de 168 pages, le résumé d'une longue réflexion créatrice.

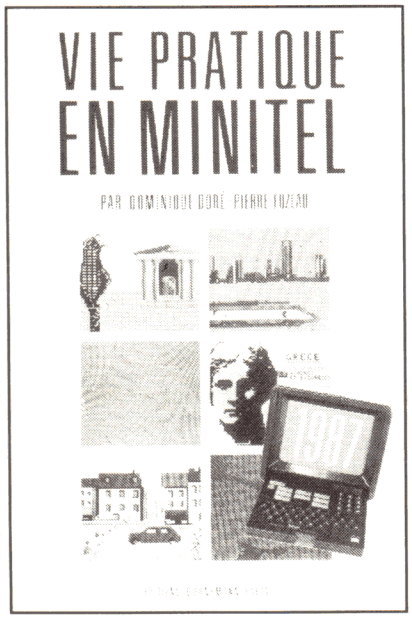
Si vous aimez les maths, vous jouerez volontiers avec elles, calculette en main... Il vous arrivera aussi de vous retrouver en terrain connu quand l'auteur évoquera le micro-ordinateur. L'objectif avoué de ce livre est, tout en nous distrayant, d'entretenir ou de développer notre aptitude générale à la résolution de problèmes. Il y parvient et il est certain que bon nombre de ses lecteurs aimeront ensuite aller un peu plus loin !

*** Editions de l'Organisation**
5, rue Rousselet — 75007 PARIS — Tél. (1) 45.67.18.40

• *VIE PRATIQUE EN MINITEL** *par D. Doré et P. Fuzeau*

La première partie de ce livre met en scène une famille : les Leminit utilisent le minitel dans leur vie de tous les jours. Ce guide pratique comprend des fiches classées dans l'ordre alphabétique (depuis animaux, annuaires, banques... jusqu'à tourisme et transports). Ces fiches aideront le lecteur à se retrouver parmi les 2000 services du minitel puisque les auteurs en mentionnent 10%, sélectionnés pour leur présentation agréable, leur rapidité, leur efficacité.

Par le biais du *CLUB VIE PRATIQUE EN MINITEL*, les lecteurs minitélistes sont invités à dialoguer avec les auteurs, à exprimer leurs impressions, et les services qu'ils apprécient pourront être retenus par les auteurs pour l'édition de 1988.



*** Editions Bornemann**
15, rue de Tournon 75006 PARIS — Tél. (1) 43.26.05.88

• *AU RÉSEAU PLANÉTAIRE**

Un avantage non négligeable pour les clients du *Réseau Planétaire* : des prix très compétitifs. Voici quelques nouveautés de fin d'année :

— **THETIS** (générateur de jeux pour Apple *IIe-IIc*). Comme le précise la notice (en français) qui l'accompagne, THETIS a pour vocation d'automatiser la création de jeux d'aventures. A partir d'un scénario et d'une suite d'images définies par l'utilisateur, il génère la disquette de jeu et celle-ci pourra ensuite tourner indépendamment du programme de création (178 F TTC).

— **EAMON** : Les disquettes EAMON sont très connues aux USA. Il s'agit d'une série comprenant une centaine de jeux d'aventures (textes en anglais). Des utilitaires permettent en outre de créer vos propres personnages... et leurs aventures. Le *RÉSEAU PLANÉTAIRE* a eu la bonne idée de traduire le *MASTER EAMON* en français (disquette et mode d'emploi). La version française est débuggée et améliorée... mais son prix reste très bas (100 F TTC... le master + LE HLM MAUDIT). Il fallait le faire !

*** Le Réseau Planétaire**
BP3 — 43260 St-Julien-Chapteuil — Tél. 71.08.73.49

• *VIFI NATHAN**

Les enseignants connaissent de réputation le département *VIFI NATHAN*. Nombreux sont les programmes éducatifs tournant sur Apple *IIe* et *IIc*... et par exemple :

— **L'HÔTEL DES MARIONNETTES** (jeu de logique). S'adresse à des enfants de 8 à 13 ans. Entre les Moppets grands ou petits, gros ou minces, habillés de rouge ou de bleu et de type Bibbitou Gribbit... on se perd un peu, mais ce n'est vrai que si l'on n'est pas capable de raisonner logiquement !

— **LA BOÎTE À PUCES** : C'est aussi un jeu de logique (à partir de 9 ans), mais qui permet de découvrir ce que sont les circuits électroniques, notamment en utilisant les notions d'horloge, de bascule, de déclencheur à retardement... Un petit logiciel sympathique capable d'intéresser les enfants... avec l'aide du papa ou du maître.

— **CONJUGUER** : Quand le *Robert* et *VIFI* s'associent, cela donne un intéressant outil pédagogique d'apprentissage de la conjugaison. Ce logiciel n'est pas récent, mais il est certain qu'il peut permettre à des enfants de découvrir, en s'amusant devant l'écran d'un Apple, les 47 formes de chaque verbe, présentées en un tableau immédiatement accessible à l'écran.

*** VIFI INTERNATIONAL**
21, Boulevard Poissonnière — 75002 PARIS.

6502 - 65C02 - 68000

BASIC - ASSEMBLEUR - PASCAL

DOS - PRODOS - CP/M

][+, //e, //e+, //c, MACINTOSH

POUR PROGRAMMER VOTRE APPLE :

Extrait du sommaire de la revue Pom's 28

- un programme de transmission de fichiers d'un Apple à l'autre par le Minitel : TEXT, Basic, binaire, fichier système, AppleWorks... Le protocole adopté (sollicitation / acquiècement) donne aux ordinateurs une transmission 'intelligente' : par exemple, le récepteur saura redemander une donnée mal comprise ;
- l'équivalent sur Macintosh pour permettre des transferts Mac/Mac, Mac/Apple // et réciproquement ;
- un programme 'PicoExpert' de conception didactique pour approcher l'intelligence artificielle et les pointeurs en Pascal ;
- deux commandes externes ProDOS donnant l'équivalent (amélioré) du MONCIO disponible sous DOS. Les commandes adressées à ProDOS par votre programme Basic (ou par le FILER par exemple) peuvent alors être suivies, contrôlées et analysées ;
- des éléments sur le codage des messages de ProDOS ;
- ...

Pom's, chez votre marchand de journaux

Bon de commande à retourner à :
Éditions MEV - 64, rue des Chantiers
78000 Versailles

Je désire recevoir :

- Revue Pom's 28 ..... à 45,00 F .....
- Disquette Pom's 28..... à 60,00 F .....

Total : .....

Nom, adresse :

_____

_____

_____

Chasseur d'images

Chaque mois, le meilleur de la technique et de la pratique photo !

Chez votre marchand de journaux !