tremplin micro

APPLE

APPLE II GS
Adressage dynamique

La souris et l'assem[illegible]

M 1631 - 13 - 33,00 F
3791631033006 00130

N° 13 - Bimestriel - Troisième année
5 Mars - 4 Mai 1987
254 FB - 11 FS - 33 F

tremplin micro 13

Avec la collaboration de :

Argos, Nicole Bréaud-Pouliquen, Robert Cazenave, Maurice Chavelli, Marcel Cottini, François Gallet, Madeleine Hodé, Yvan Kœnig, Nestor, Pierre Privat et Clément Renard.

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.

SOMMAIRE

À BÂTONS ROMPUS

Que de déceptions chez les premiers possesseurs d'un Apple *IIGS* ! Je ne compte plus les S.O.S. téléphoniques et les lettres plus ou moins virulentes, soulignant que l'article paru dans le numéro 11 de *Tremplin Micro* n'a pas été étranger à la décision d'un achat que déjà l'on regrette...

Heureusement, du moins pour l'avenir de la machine, les réactions positives se révèlent nettement plus nombreuses que les premières, mais émanent généralement d'utilisateurs considérés comme particulièrement compétents : programmeurs, membres de clubs de micro-informatique, applemaniaques inconditionnels.

Reste que le mal existe. APPLE aurait certainement pu le prévenir. Le guérir se révélera plus difficile et laissera en tout cas des séquelles.

Heureux membres du Club Apple

Premier problème : un certain nombre de personnes, généralement mal informées, ont payé leur Apple *IIGS* au prix fort alors que tous les membres du *Club Apple* (adhésion gratuite) pouvaient bénéficier, pendant la plus grande partie du mois de décembre, d'une remise de trente pour cent (non seulement sur le *GS*, mais sur la plupart des produits de la marque, Apple *IIc*, excepté). Une condition : être membre du Club au 1er décembre 1986. Une autre (exigée par des revendeurs pointilleux) : avoir passé la commande pendant la période effectivement concernée par la promotion... et non avant. Tant pis pour les imbéciles qui, accordant toute leur confiance à la marque avaient commandé leur *GS* immédiatement après l'annonce de sa commercialisation ! Personnellement, j'ai conseillé à tous mes correspondants de ne point céder aux pressions de leur concessionnaire et d'exiger LEUR remise de trente pour cent quelle que soit la date d'enregistrement de la commande. Il y a eu des cris et des grincements de dents et je connais actuellement des mécontents jurant, mais un peu tard, qu'on ne les prendra jamais plus à ce petit jeu stupide.

Un Apple *IIGS*, mais pour quoi faire ?

Je vais être dur, mais franc, comme toujours : si vous avez acheté ou commandé un *GS* sans savoir pourquoi, ne venez pas vous plaindre de la pauvreté de sa documentation ou de l'absence de logiciels utilisant toutes les ressources du 65C816. Bien entendu, si vous avez décidé cet achat simplement pour le plaisir de disposer, chez vous, du dernier produit à l'enseigne de la POMME, libre à vous. Je préciserai même que je comprends parfaitement cette démarche. Mais alors, prenez patience et attendez de connaître votre nouveau jouet avant de lui découvrir tous les défauts possibles. Des qualités, il en possède votre Apple *IIGS*, mais elles se dissimulent presque toutes derrière une compatibilité qui pourrait bien être considérée, plus tard, comme pénalisante.

Les logiciels viendront en leur temps, mais les auteurs ne se bousculent pas au portillon. Je vous promets que *Tremplin Micro* leur ouvrira toutes grandes ses colonnes pour parler de leurs réalisations. Il est certain que l'Apple *IIGS* permettra aux programmeurs d'aller plus loin, mais seulement dans la mesure où ils accepteront de passer par des langages plus évolués que le Basic Applesoft : le C, par exemple, et l'Assembleur. Les OUTILS mis à leur disposition par le *GS* autorisent des logiciels dignes de ceux de Mac.

Littérature... pauvre

Mais pour programmer une machine aussi complexe que l'Apple *IIGS*, il convient de posséder de bons manuels de références, avec si possible de nombreux exemples. Apple propose bien son *INSIDE APPLE IIGS* (plus de 3000 pages de documentation et 8 disquettes)*, mais ne comptez pas sur ce monument pour vous fournir des EXEMPLES. Or, je le sais, c'est ce qu'attendent tous nos correspondants. Je rappelle à ce propos que deux ouvrages doivent paraître au PSI *(Assembleur de l'Apple IIGS* et *LA BOÎTE À OUTILS DE L'APPLE IIGS)*... mais j'ignore à quelle date. J'admire d'ailleurs les auteurs qui se sont attelés à une telle besogne : ce n'est pas gagné d'avance !

Vœux pieux

Alors, le *GS ?* Personnellement, je souhaite qu'il ne connaisse pas le sort de l'Apple *III* (le meilleur, d'après ceux qui en sont encore les heureux utilisateurs), mais plutôt celui de l'Apple *II*, son illustre ancêtre. J'aurais aimé que Microsoft (ce sont des gens qui ignorent totalement *Tremplin MIcro*... sans doute trop petit pour les intéresser) et Apple se mettent d'accord pour nous concocter, sur le *GS*, un vrai Basic, ressemblant par exemple à celui du Mac. Le langage C me séduit, mais le temps de compilation est long... trop long. J'aimerais aussi avoir ProDOS 16 en ROM (je ne suis pas opposé à sa présence sur une carte) car sa durée de chargement est également interminable. Quant au reste — et notamment les programmes —, rassurez-vous : ils arrivent, mais accordez-nous le temps de comprendre la machine et celui de l'alimenter en instructions comprises par le 65C816 !

G-H.

* VIF — 5 rue de Bassano — 75116 PARIS (2000 F TTC).

UTILITAIRE | DOS 3.3 | ProDOS

Fonctionne aussi sur l'APPLE IIGS

Maurice CHAVELLI

TRANSITION

DE HGR à HGR2

CERTAINS jeux utilisent, dans leur présentation graphique, un très élégant scrolling des pages graphiques. Je me suis demandé s'il était facile à réaliser. Dans cet article, je vous propose deux versions :

- La première est très économe en espace mémoire, mais n'est pas très rapide.
- La deuxième se révèle beaucoup plus gourmande, mais elle est deux fois plus rapide que la précédente.

Bien sûr, il n'est pas interdit de concevoir une formule encore plus performante ! Mais je vous promets de longues réflexions sur l'adressage des pages graphiques de votre Apple chéri. **SCROLL.DÉMO** vous présente les deux versions avec des écrans graphiques sommaires. Libre à vous de remplacer ceux-ci par vos chefs-d'œuvre. Il est facile de modifier les programmes pour obtenir un défilement continu ou d'utiliser N pages. A vos claviers !

SCROLL.DEMO

```
 1 FOR C = 896 TO 916: READ A%: POKE C,A%: NEXT    4D47
 2 POKE 232,128: POKE 233,3                        1E01
 3 DATA 1,0,4,0,54,54,54,110,36,36,36,108,54,54,
   54,110,36,36,36,36,0                            78FA
 4 SPEED= 100                                      193A
 5 TEXT : HOME : POKE  - 16197,0                   8E7A
 6 VTAB 8: HTAB 3: PRINT "Ce programme est une d
   émonstration": PRINT : PRINT : HTAB 7: PRINT
   "des routines de transition": PRINT : PRINT :
   HTAB 9: PRINT "de pages graphiques HGR."        FFB2
 7 PRINT : PRINT : PRINT : PRINT "Nous allons co
   mmencer par créer 2 écrans"                     F6EF
 8 FOR C = 1 TO 500: NEXT                          82D7
 9 HGR2 : HCOLOR= 3: ROT= 0: SCALE= 1              0495
10 FOR X = 0 TO 272 STEP 16: FOR Y = 0 TO 184 ST
   EP 14                                           6501
11 DRAW 1 AT X,Y: NEXT Y,X                         3400
12 FOR X = 8 TO 264 STEP 16: FOR Y = 7 TO 177 ST
   EP 14                                           1713
13 DRAW 1 AT X,Y: NEXT Y,X                         3400
14 HGR : POKE  - 16302,0                           E1A5
15 FOR X = 1 TO 273 STEP 16: FOR Y = 0 TO 184 ST
   EP 14                                           8A03
16 DRAW 1 AT X,Y: NEXT Y,X                         3400
```

FORME

Les lignes 1 à 3 établissent une table de forme(s) sommaire (seulement un carré !) à partir de l'adresse 896.

Si vous avez la chance de posséder un écran couleur, vous obtiendrez des écrans de couleur différente par le décalage horizontal (1 point) de la forme.

Nous vous donnons, plus loin, un moyen de tracer un peu plus rapidement un écran damier, mais ce n'était pas le propos de Maurice Chavelli.

TRANSITION

VOS PROGRAMMES

Les débutants pourront passer directement à la page 7.

SCROLL.DEMO (suite)

```
17 FOR X = 9 TO 265 STEP 16: FOR Y = 7 TO 177 ST
   EP 14                                          7215
18 DRAW 1 AT X,Y: NEXT Y,X                        3400
19 FOR C = 1 TO 100: NEXT : HOME                  66A4
20 TEXT : VTAB 15: HTAB 8: PRINT "Voici la versi
   on n° 1...": FOR C = 1 TO 1000: NEXT           4859
21 PRINT  CHR$ (4)"BLOADSCROLL1.C"                A53D
22 POKE 49232,0: POKE 49239,0: POKE 49234,0       1AC8
23 CALL 768: HOME                                 AB02
24 TEXT : VTAB 15: HTAB 8: PRINT "Voici la versi
   on n° 2..."                                    4D1D
25 POKE 230,32                                    89DF
26 FOR X = 1 TO 273 STEP 16: FOR Y = 0 TO 184 ST
   EP 14                                          8A03
27 DRAW 1 AT X,Y: NEXT Y,X                        3400
28 FOR X = 9 TO 265 STEP 16: FOR Y = 7 TO 177 ST
   EP 14                                          7215
29 DRAW 1 AT X,Y: NEXT Y,X                        3400
30 PRINT  CHR$ (4)"BLOADSCROLL2.C"                C03E
31 POKE 49232,0: POKE 49239,0: POKE 49234,0       1AC8
32 CALL 768                                       8331
33 FOR C = 1 TO 1000: NEXT : HOME : TEXT : SPEED
   = 255                                          2616
```

SCROLL1.S

PROGRAMME SOURCE

Assemblage :

ProCODE

```
                    1  ******************
                    2  * Transition V1 *
                    3  ******************
                    4
                    5  * Cette version effectue la transition en 26s
                    6
                    7  PTR      =    $8          ; pointeur de ligne
                    8  HBASL    =    $26
                    9  HBASH    =    $27
                   10  HPAG     =    $E6
                   11  HPOSN    =    $F417
                   12
                   13           ORG  $300
                   14
                   15  * Transfert partie de la routine HPOSN
                   16  * pour améliorer la vitesse d'exécution
                   17
0300: A2 60        18           LDX  £$60        ; installe un RTS à la fin
0302: 8E 78 03     19           STX  $378
0305: A2 20        20           LDX  £32         ; c'est parti pour 33 octets
0307: BD 17 F4     21  BCL      LDA  HPOSN,X
030A: 9D 57 03     22           STA  HPOS,X
030D: CA           23           DEX
030E: 10 F7        24           BPL  BCL
```

```
                        25
                        26  * Début du programme
                        27
0310: A2 00             28          LDX  £0          ; première ligne
0312: A9 20             29  BCLX    LDA  £$20        ; sélectionne la page 1
0314: 85 E6             30          STA  HPAG
0316: 8D 2B 03          31          STA  MEM+2       ; adresse de la ligne 0
0319: A9 00             32          LDA  £0
031B: 8D 2A 03          33          STA  MEM+1
031E: A9 01             34          LDA  £1          ; première ligne
0320: 85 08             35          STA  PTR         ; initialise le pointeur
0322: 20 57 03          36  DEBUT   JSR  HPOS        ; calcul adresse de la ligne
0325: A0 27             37          LDY  £39         ; c'est parti pour 40 octets
0327: B1 26             38  BCLY1   LDA  (HBASL),Y   ; transfert d'un octet
0329: 99 00 00          39  MEM     STA  $0,Y
032C: 88                40          DEY              ; un de moins
032D: 10 F8             41          BPL  BCLY1       ; le dernier?
032F: A5 26             42          LDA  HBASL       ; sauvegarde des coordonnées
0331: 8D 2A 03          43          STA  MEM+1
0334: A5 27             44          LDA  HBASH
0336: 8D 2B 03          45          STA  MEM+2
0339: E6 08             46          INC  PTR         ; une ligne de plus
033B: A5 08             47          LDA  PTR
033D: C9 C0             48          CMP  £192        ; la dernière?
033F: D0 E1             49          BNE  DEBUT       ; non
0341: 06 E6             50          ASL  HPAG        ; sélectionne la page 2
0343: 8A                51          TXA
0344: 20 57 03          52          JSR  HPOS        ; transfert de la ligne X...
0347: A0 27             53          LDY  £39
0349: B1 26             54  BCLY2   LDA  (HBASL),Y
034B: 99 D0 3F          55          STA  $3FD0,Y
034E: 88                56          DEY
034F: 10 F8             57          BPL  BCLY2
0351: E8                58          INX              ; une ligne de plus
0352: E0 C0             59          CPX  £192        ; la dernière?
0354: D0 BC             60          BNE  BCLX        ; non
0356: 60                61          RTS              ; oui => retour BASIC
                        62  HPOS    DFB
```

SCROLL2.S

PROGRAMME SOURCE

Assemblage :

ProCODE

```
 1  ******************
 2  * Transition V2 *
 3  ******************
 4
 5  * Cette version effectue la transition en 13s
 6
 7  PTR      =    $8          ; pointeur de ligne
 8  HBASL    =    $26
 9  HBASH    =    $27
10  HPAG     =    $E6
11  HPOSN    =    $F417
12
```

(suite page 6)

SCROLL2.S (suite)

```
                        13          ORG  $300
                        14
                        15  * Création sous-programme en $6000
                        16
0300: A9 20             17          LDA  £$20       ; sélectionne la page 1
0302: 85 E6             18          STA  HPAG
0304: A9 00             19          LDA  £0         ; pointe en $6000
0306: 8D 73 03          20          STA  IND1+1
0309: A9 60             21          LDA  £$60
030B: 8D 74 03          22          STA  IND1+2
030E: A9 A2             23          LDA  £$A2       ; installe un LDX £$27
0310: 20 72 03          24          JSR  IND1
0313: A9 27             25          LDA  £$27
0315: 20 72 03          26          JSR  IND1
0318: A9 01             27          LDA  £1         ; ligne 1
031A: 85 08             28          STA  PTR
031C: A9 BD             29  BCL     LDA  £$BD       ; installe un LDA
031E: 20 72 03          30          JSR  IND1
0321: 20 66 03          31          JSR  IND
0324: C6 08             32          DEC  PTR
0326: A9 9D             33          LDA  £$9D       ; installe un STA
0328: 20 72 03          34          JSR  IND1
032B: 20 66 03          35          JSR  IND
032E: E6 08             36          INC  PTR
0330: E6 08             37          INC  PTR
0332: A5 08             38          LDA  PTR
0334: C9 C0             39          CMP  £192       ; la dernière?
0336: D0 E4             40          BNE  BCL        ; non
0338: A2 00             41          LDX  £0
033A: BD 7E 03          42  BCLX    LDA  FIN,X      ; installe la fin
033D: 20 72 03          43          JSR  IND1
0340: E8                44          INX
0341: E0 07             45          CPX  £7
0343: D0 F5             46          BNE  BCLX
                        47
                        48  * Début du programme
                        49
0345: 06 E6             50          ASL  HPAG       ; page 2
0347: A9 00             51          LDA  £0
0349: 85 08             52          STA  PTR        ; ligne 0
034B: 20 00 60          53  DEPART  JSR  $6000      ; monte d'un cran
034E: A5 08             54          LDA  PTR
0350: 20 17 F4          55          JSR  HPOSN      ; transfert ligne 191
0353: A0 27             56          LDY  £39
0355: B1 26             57  BCLY    LDA  (HBASL),Y
0357: 99 D0 3F          58          STA  $3FD0,Y
035A: 88                59          DEY
035B: 10 F8             60          BPL  BCLY
035D: E6 08             61          INC  PTR        ; une ligne de plus
035F: A5 08             62          LDA  PTR
0361: C9 C0             63          CMP  £192       ; la dernière?
0363: D0 E6             64          BNE  DEPART     ; non
0365: 60                65          RTS
                        66
```

```
                    67  * Routine pour la création
                    68
0366: A5 08         69  IND       LDA   PTR
0368: 20 17 F4      70            JSR   HPOSN       ; calcul ligne A
036B: A5 26         71            LDA   HBASL       ; sauvegarde...
036D: 20 72 03      72            JSR   IND1
0370: A5 27         73            LDA   HBASH
0372: 8D 00 60      74  IND1      STA   $6000
0375: EE 73 03      75            INC   IND1+1
0378: D0 03         76            BNE   IND2
037A: EE 74 03      77            INC   IND1+2
037D: 60            78  IND2      RTS
                    79
037E: CA            80  FIN       HEX   CA          ; DEX
037F: 30            81            HEX   30          ; BMI
0380: 03            82            HEX   03
0381: 4C            83            HEX   4C          ; JMP $6002
0382: 02            84            HEX   02
0383: 60            85            HEX   60
0384: 60            86            HEX   60          ; RTS
```

SCROLL1.C

Ecriture sur disquette : **BSAVE SCROLL1.C,A$300,L$57**

```
0300: A2 60 8E 78 03 A2 20 BD 17 F4 9D 57 03 CA 10 F7   875D
0310: A2 00 A9 20 85 E6 8D 2B 03 A9 00 8D 2A 03 A9 01   BE9E
0320: 85 08 20 57 03 A0 27 B1 26 99 00 00 88 10 F8 A5   E873
0330: 26 8D 2A 03 A5 27 8D 2B 03 E6 08 A5 08 C9 C0 D0   895B
0340: E1 06 E6 8A 20 57 03 A0 27 B1 26 99 D0 3F 88 10   A7AF
0350: F8 E8 E0 C0 D0 BC 60                              8C6C
```

SCROLL2.C

BSAVE SCROLL2.C,A$300,L$85

```
0300: A9 20 85 E6 A9 00 8D 73 03 A9 60 8D 74 03 A9 A2   9F38
0310: 20 72 03 A9 27 20 72 03 A9 01 85 08 A9 BD 20 72   3A29
0320: 03 20 66 03 C6 08 A9 9D 20 72 03 20 66 03 E6 08   D3AC
0330: E6 08 A5 08 C9 C0 D0 E4 A2 00 BD 7E 03 20 72 03   FC4D
0340: E8 E0 07 D0 F5 06 E6 A9 00 85 08 20 00 60 A5 08   FBE3
0350: 20 17 F4 A0 27 B1 26 99 D0 3F 88 10 F8 E6 08 A5   1D94
0360: 08 C9 C0 D0 E6 60 A5 08 20 17 F4 A5 26 20 72 03   E4DF
0370: A5 27 8D 00 60 EE 73 03 D0 03 EE 74 03 60 CA 30   97AF
0380: 03 4C 02 60 60                                    A911
```

Si le langage machine et l'assembleur ne vous inspirent pas, contentez-vous de recopier ces routines à la suite d'un CALL-151.

Les caractères HEXA indiqués en bout de ligne sont destinés au contrôle de la saisie (contrôle facile avec votre disquette SIGNATURE).

DAMIER

```
0300: A2 00 86 09 86 18 A9 00 85 06 A9 00 85 07 A9 11   8FF2
0310: 85 08 A5 18 4A 90 06 A9 08 85 06 C6 08 A6 06 A4   6484
0320: 07 A5 09 20 11 F4 A0 03 A2 4B 20 5D F6 A9 10 18   79AE
0330: 65 06 85 06 90 02 E6 07 C6 08 10 E1 A5 09 69 08   4D53
0340: 85 09 E6 18 A5 18 C9 18 90 BC 60 36 36 36 6E 24   0B0A
0350: 24 24 6C 36 36 36 6E 24 24 24 24 00               B154
```

Voir page 8

BSAVE DAMIER, A$300,L$5C

DAMIER HGR

LA FORME ? un simple carré. Le programme : aussi clair que possible (le problème pourrait être traité autrement), pour être compris par des utilisateurs peu habitués au langage machine.

L'adresse horizontale du point initial de chaque carré est sur deux octets (elle est stockée en $6-7). L'adresse verticale (un seul octet) est mémorisée en $9. On utilise par ailleurs deux pointeurs : le nombre de carrés par ligne, décrémenté jusqu'à Ø inclus (il est stocké en $8) et le nombre de lignes (dans la case mémoire $18). Telle qu'elle se présente, la routine fonctionne aussi bien avec le 65Ø2 qu'avec le 65CØ2... ou le 65C816 de l'Apple IIGS. Vous remarquerez qu'elle n'utilise pas la pile et ne cherche pas non plus à tirer parti des registres.

La couleur du damier est modifiée par la ligne Basic 11, en décalant toutes les adresses horizontales d'un point (POKE 775,Ø et POKE 792,8 ou bien POKE 775,1 et POKE 792,9).

Vous trouverez page 10 le listage de DAMIER1 : même longueur, mais l'affichage des carrés se fait par colonne au lieu de s'effectuer par ligne.

XDRAW (ou DRAW) : En Basic, pour utiliser XDRAW ou DRAW, il est nécessaire d'indiquer le numéro de la forme dans la table, et aussi de l'adresse de cette table à l'APPLE (pointeurs $E8-E9). En langage machine, il suffit de placer dans X la partie basse de l'adresse du premier octet de la forme et dans Y sa partie haute. Les pointeurs $E8 et E9 ne sont pas utilisés. Par contre, avant de tracer une forme, il est indispensable de positionner le curseur ($F411).

Pour obtenir un damier plus compact (comme celui tracé par le programme de Maurice CHAVELLI, page 3), il suffit de majorer le nombre de lignes : ($346 : C9 1B) et de n'ajouter que 7 pts par ligne : ($33E : 69 Ø7).

DAM.DEMO

```
 1 PRINT  CHR$ (21): HGR : HGR2 : TEXT : HOME      727E
 3 PRINT  CHR$ (4)"BLOAD DAMIER"                   D49E
 5 HCOLOR= 3: ROT= 0: SCALE= 1                     D7CB
 7 POKE 49232,0: POKE 49234,0: POKE 49237,0: POK
   E 49239,0                                       B720
 9 POKE 230,64: CALL 768: REM TRACE HGR2           664F
11 POKE 775, ABS ( PEEK (775) - 1): POKE 792,8 +
   PEEK (775)                                      6C14
13 POKE 49232,0: POKE 49239,0: POKE 49236,0: POK
   E 230,32                                        31E3
15 CALL 768: REM TRACE HGR                         8331
17 IF  PEEK (49152) < 128 THEN 7                   C24C
19 POKE 49168,0: PRINT : TEXT                      D8D8
21 VTAB 23: PRINT "(M)ENU DE DISQUETTEctG ";: GE
   T R$: PRINT : HOME                              A4DB
23 IF R$ = "M" THEN  PRINT  CHR$ (4)"RUN MENU,D1
   "                                               259D
```

```
                                0  *  DAMIER (ASSEMBLAGE PROCODE)
                                1  *
                                2  Hor      EQU  $6        ; Position horizontale de la forme.
                                3  Pthor    EQU  $8        ; Nombre de formes par ligne.
                                4  Ver      EQU  $9        ; Position verticale de la forme.
                                5  Ptver    EQU  $18       ; Nombre de rangées.
                                6  Hposn    EQU  $F411     ; Positionne le curseur.
                                7  XDraw    EQU  $F65D     ; Trace la forme (X,Y).
                                8  *
                                9          ORG  $300
                               10  *
300 :  A2 00                   11          LDX  £$00      ; Initialisation.
302 :  86 09                   12          STX  Ver
304 :  86 18                   13          STX  Ptver
306 :  A9 00                   14  BCL3    LDA  £$00
308 :  85 06                   15          STA  Hor
30A :  A9 00                   16          LDA  £$00
30C :  85 07                   17          STA  Hor+1
30E :  A9 11                   18          LDA  £$11      ; Nombre de formes horizontales.
310 :  85 08                   19          STA  Pthor
312 :  A5 18                   20          LDA  $18       ; Nombre de lignes ?
314 :  4A                      21          LSR            ; Pair ou impair ?
315 :  90 06                   22          BCC  BCL1      ; Pair : on saute.
317 :  A9 08                   23          LDA  £$08      ; Sinon 8 de plus horizontalement
319 :  85 06                   24          STA  Hor       ; que l'on stocke
31B :  C6 08                   25          DEC  Pthor     ; et une forme de moins.
31D :  A6 06                   26  BCL1    LDX  Hor       ; Partie basse dans X.
31F :  A4 07                   27          LDY  Hor+1     ; Partie haute dans Y.
321 :  A5 09                   28          LDA  Ver       ; Position verticale dans A.
323 :  20 11 F4                29          JSR  Hposn     ; Et curseur en place.
326 :  A0 03                   30          LDY  £$03      ; Adresse du premier octet
328 :  A2 4B                   31          LDX  £$4B      ; de la forme dans Y,X.
32A :  20 5D F6                32          JSR  XDraw     ; XDraw fait le reste.
32D :  A9 10                   33          LDA  £$10      ; Plus 16 horizontalement.
32F :  18                      34          CLC            ; Annulation de retenue.
330 :  65 06                   35          ADC  Hor
332 :  85 06                   36          STA  Hor
334 :  90 02                   37          BCC  BCL2      ; Sans retenue : saut.
336 :  E6 07                   38          INC  Hor+1     ; Partie haute éventuelle.
338 :  C6 08                   39  BCL2    DEC  Pthor     ; Une forme en moins.
33A :  10 E1                   40          BPL  BCL1      ; Suite jusqu'à 0 inclus.
33C :  A5 09                   41          LDA  Ver       ; Position verticale majorée
33E :  69 08                   42          ADC  £$08      ; de 8 (damier normal).
340 :  85 09                   43          STA  Ver
342 :  E6 18                   44          INC  Ptver     ; Ne pas oublier le nombre
344 :  A5 18                   45          LDA  Ptver     ; de lignes.
346 :  C9 18                   46          CMP  £$18      ; Est-il égal à 24 ?
348 :  90 BC                   47          BCC  BCL3      ; Inférieur : encore une.
34A :  60                      48          RTS            ; Sinon c'est terminé.
                               49
                               50  *       TABLE DE FORME(S) : UN CARRÉ
                               51
34B :  36 36 36 6E 24 24 24 6C 36
354 :  36 36 6E 24 24 24 24 00
                               52          HEX  3636366E2424246C3636366E2424242400
```

VOIR LES CODES DANS LE BAS DE LA PAGE 7.

(voir DAMIER1 page suivante)

```
                                0  *  DAMIER1 (ASSEMBLAGE PROCODE)
                                1  *
                                2  Hor      EQU  $6       ; Position horizontale de la forme.
                                3  Pthor    EQU  $8       ; Nombre de formes par ligne.
                                4  Ver      EQU  $9       ; Position verticale de la forme.
                                5  Ptver    EQU  $18      ; Nombre de rangées.
                                6  Hposn    EQU  $F411    ; Positionne le curseur.
                                7  XDraw    EQU  $F65D    ; Trace la forme (X,Y).
                                8  *
                                9          ORG  $300
                               10  *
300 : A9 00                    11          LDA  £$00     ; Initialisation.
302 : 85 06                    12          STA  Hor
304 : A9 00                    13          LDA  £$00
306 : 85 07                    14          STA  Hor+1
308 : A9 22                    15          LDA  £$22     ; Nombre de formes horizontales.
30A : 85 08                    16          STA  Pthor
30C : A2 00                    17  BCL2    LDX  £$00     ; Initialisation verticale.
30E : 86 09                    18          STX  Ver
310 : A5 08                    19          LDA  Pthor    ; Rangée ?
312 : 4A                       20          LSR           ; Paire ou impaire ?
313 : 90 05                    21          BCC  BCL1     ; Paire : on saute.
315 : A9 08                    22          LDA  £$08     ; Sinon 8 de plus verticalement
317 : 85 09                    23          STA  Ver      ; que l'on stocke
319 : E8                       24          INX           ; et plus une rangée.
31A : 86 18                    25  BCL1    STX  Ptver    ; Nombre de lignes stocké.
31C : A6 06                    26          LDX  Hor      ; Partie basse dans X.
31E : A4 07                    27          LDY  Hor+1    ; Partie haute dans Y.
320 : A5 09                    28          LDA  Ver      ; Position verticale dans A.
322 : 20 11 F4                 29          JSR  Hposn    ; Et curseur en place.
325 : A0 03                    30          LDY  £$03     ; Adresse du premier octet
327 : A2 4B                    31          LDX  £$4B     ; de la forme dans Y,X.
329 : 20 5D F6                 32          JSR  XDraw    ; XDraw fait le reste.
32C : A5 09                    33          LDA  Ver      ; Position verticale majorée
32E : 18                       34          CLC
32F : 69 10                    35          ADC  £$10     ; de 16 (damier normal).
331 : 85 09                    36          STA  Ver
333 : A6 18                    37          LDX  Ptver    ; Nombre de lignes dans X.
335 : E8                       38          INX
336 : E8                       39          INX           ; On écrit 1 ligne sur 2.
337 : E0 18                    40          CPX  £$18     ; Est-il égal à 24 ?
339 : 90 DF                    41          BCC  BCL1     ; Inférieur : encore une.
33B : A5 06                    42          LDA  Hor      ; On avance horizontalement.
33D : 18                       43          CLC           ; Annulation de la retenue.
33E : 69 08                    44          ADC  £$08     ; Plus une forme.
340 : 85 06                    45          STA  Hor      ; Ecriture partie basse.
342 : 90 02                    46          BCC  S        ; Pas de retenue : saut.
344 : E6 07                    47          INC  Hor+1    ; Sinon incrém. partie haute.
346 : C6 08                    48  S       DEC  Pthor    ; Une forme en moins
348 : 10 C2                    49          BPL  BCL2     ; Si plus grand ou égal 0...
34A : 60                       50          RTS           ; Sinon c'est terminé.
                               51
                               52  *       TABLE DE FORME(S) : UN CARRÉ
                               53
34B : 36 36 36 6E 24 24 24 6C 36
354 : 36 36 6E 24 24 24 24 00
                               54          HEX  3636366E2424246C3636366E2424242400
```

Pour sauver ce programme :
BSAVE DAMIER1,A$300,L92

Pour utiliser DAMIER1, modifiez la ligne 3 de **DAM.DEMO**, mais surtout sa ligne 11, qui devient : **POKE 769, ABS(PEEK(769) — 1)**

DAMIER RAPIDE

Essayez aussi cette routine. Vous constaterez qu'elle n'utilise pas XDRAW... mais l'affichage (simulé) d'un caractère graphique : le carré. Celui-ci est représenté par une série de $7F. Dans la démo, on remplace successivement la valeur $7F par des valeurs allant de $Ø à... $7F, d'où la production de divers écrans. (On peut interrompre le programme en pressant ESCAPE).

```
10  PRINT CHR$ (21): HGR2 : HCOLOR = 3: PRINT CHR$
    (4)"BLOAD DAMRAP"
15  CALL 768: GOSUB 45
20  FOR I = Ø TO 127: POKE 800,I: CALL 768: GOSUB 45:
    NEXT
25  HOME : TEXT
30  VTAB 22: PRINT "(M)ENU DE DISQUETTE ";: GOSUB 45
35  IF R$ = "M" THEN PRINT CHR$ (4)"RUN MENU,D1"
40  HOME : END
45  CALL - 198: GET R$: PRINT:IF R$ <▷ CHR$ (27) THEN
    RETURN
50  POP : GOTO 25
```

DAMRAP (ASSEMBLAGE PROCODE)

```
                    0  *
                    1  *
                    2  VT       EQU  $18      ; Pointeur ligne.
                    3  WNDWDTH  EQU  $21      ; Largeur fenêtre.
                    4  WNDBTM   EQU  $23      ; Bas de fenêtre.
                    5  BASL     EQU  $28      ; Adresse basse colonne 1.
                    6  BAS2L    EQU  $2A      ; Pointeur ligne de destination.
                    7  HPAG     EQU  $E6      ; Drapeau page HGR ($2Ø/4Ø).
                    8  VTAB     EQU  $FC22    ; On utilisera VTAB+2.
                    9  *
                   10           ORG  $3ØØ     ;
                   11  *
3ØØ : A2 ØØ        12           LDX  £$ØØ     ; Registre X = Ø.
3Ø2 : 8A           13  BCL4     TXA           ; X passé dans A.
3Ø3 : 85 18        14           STA  VT       ; Pointeur ligne.
3Ø5 : 2Ø 24 FC     15           JSR  VTAB+2   ; Positionnement.
3Ø8 : AØ ØØ        16           LDY  £$ØØ     ; Registre Y à zéro.
3ØA : A5 28        17  BCL3     LDA  BASL     ; Transfert de la partie
3ØC : 85 2A        18           STA  BAS2L    ; basse, puis de la partie
3ØE : A5 29        19           LDA  BASL+1   ; haute, mais après,
31Ø : 29 Ø3        2Ø           AND  £$Ø3     ; AND ØØØØØØ11 et ORA $E6
312 : Ø5 E6        21           ORA  HPAG     ; ($7 devient $43...
314 : 85 2B        22           STA  BAS2L+1  ; $4 devient $4Ø, etc.).
316 : A5 18        23           LDA  VT       ; Pointeur ligne dans A.
318 : 4A           24           LSR           ; Bit Ø dans la retenue.
319 : 9Ø Ø2        25           BCC  BCL1     ; Si C = Ø (pas de retenue),
31B : E6 2A        26           INC  BAS2L    ; on n'incrémente pas.
31D : A2 ØØ        27  BCL1     LDX  £$ØØ     ; X = 1 : compteur.
31F : A9 7F        28  BCL2     LDA  £$7F     ; Carré plein = 8 fois $7F.
321 : 91 2A        29           STA  (BAS2L),Y ; Ecriture HGR ou HGR2.
323 : 18           3Ø           CLC           ; Annulation de la retenue.
324 : A5 2B        31           LDA  BAS2L+1  ; Partie haute dans A
326 : 69 Ø4        32           ADC  £$Ø4     ; majorée pour le point
328 : 85 2B        33           STA  BAS2L+1  ; suivant.
32A : E8           34           INX           ; Ne pas oublier pointeur.
32B : EØ Ø8        35           CPX  £$Ø8     ; Est-on à 8 ?
32D : 9Ø FØ        36           BCC  BCL2     ; Non : encore un point.
32F : C8           37           INY           ; On trace un carré sur
33Ø : C8           38           INY           ; deux donc Y + 2.
331 : C4 21        39           CPY  WNDWDTH  ; A-t-on terminé la ligne ?
333 : 9Ø D5        4Ø           BCC  BCL3     ; Non : passons à l'autre.
335 : A6 18        41           LDX  VT       ; VT dans X.
337 : E8           42           INX           ; VT = VT + 1.
338 : E4 23        43           CPX  WNDBTM   ; Est-ce la dernière ligne ?
33A : 9Ø C6        44           BCC  BCL4     ; Non : encore une autre.
33C : 6Ø           45           RTS           ; Oui : terminé !
```

BSAVE DAMRAP,A$3ØØ,L$3D

(Suite page 12)

DAMIER.VC

Essayez enfin cette dernière routine, plus compacte (26 octets), mais proche de la précédente. Constatez, grâce à la boucle de la ligne 30 (on peut en sortir par ESCAPE) l'importance de la valeur initiale de X ($304). Peut-être est-il possible de faire encore plus court : rien ne vous interdit d'essayer !

Remarque : Pour que cette routine fonctionne, il faut que l'adresse $E5 soit à 0... ce qui n'est pas le cas après DAM.DEMO... d'où le POKE 229,0 (qui donne le premier carré blanc). Un POKE 229.1 permet de démarrer avec un carré noir.

```
10  TEXT : HOME : PRINT CHR$ (21) : POKE 229,0
20  FOR I = 768 TO 793: READ R: POKE I,R: NEXT
30  FOR I = 31 TO 1 STEP - 1: POKE 772,I: GOSUB 70:
    CALL 768: NEXT
40  GOSUB 70: TEXT
50  VTAB 22: PRINT "(M)ENU DE DISQUETTE ";: GET R$:
    PRINT : IF R$ = "M" OR R$ = "m" THEN PRINT CHR$
    (4)"RUN MENU.D1"
60  HOME : END
70  CALL - 198: POKE 49168,0: WAIT 49152,128,127: POKE
    49168,0: IF (PEEK (49152)) ◁ ▷ 27 THEN RETURN
80  TEXT : POP : GOTO 50
90  DATA 32,216,243,162,31,169,127,192,120,208,1,200,145,
    229,200,200,192,249,208,243,230,230,202,16,242,96
```

DAMIER.VC

```
300 :  20  D8  F3    JSR  $F3D8     HGR2 (à la sortie : Y = 0).
303 :  A2  1F        LDX  £$1F      Compteur pour partie haute de l'adresse ($E6).
305 :  A9  7F        LDA  £$7F      Pour obtenir un carré plein.
307 :  C0  78        CPY  £$78  ]   Si Y est différent de 0, pas d'incrémentation.
309 :  D0  01        BNE  $030C ]
30B :  C8            INY            Y = Y + 1.
30C :  91  E5        STA  ($E5),Y   Une ligne du carré.
30E :  C8            INY        ]   Ne pas oublier qu'il s'agit d'un damier !
30F :  C8            INY        ]
310 :  C0  F9        CPY  £$F9  ]   Si Y est différent de $F9 on continue.
312 :  D0  F3        BNE  $0307 ]
314 :  E6  E6        INC  $E6   ]   Sinon partie haute de l'adresse incrémentée,
316 :  CA            DEX        ]   et compteur décrémenté en conséquence.
317 :  10  F2        BPL  $030B     On poursuit (de $40 à $5F).
319 :  60            RTS            Retour.
```

BSAVE DAMIER.VC,A$300,L$1A

UTILITAIRE | DOS 3.3 | ProDOS

Fonctionne aussi sur l'APPLE IIGS

Clément RENARD

VISUALISER

une police graphique

Version 2

Seconde version d'une routine permettant de visualiser, sur l'un des écrans graphiques de l'Apple, les 96 caractères d'une police classique (768 octets). Nous donnons ici le source de CAR.VIS2. Il ne diffère de CAR.VIS1 (dont on trouvera les codes plus loin) que par la présence de la ligne **32** : — EOR $7F.

Pour obtenir un affichage normal, il faut : EOR $ØØ... car EOR $7F affiche les caractères en mode inverse. Les utilisateurs d'un moniteur couleur savent que les résultats ne sont malheureusement pas extraordinaires en mode normal et inacceptables en mode inverse. Par contre, avec une police spéciale (voir *Tremplin Micro* n°12), on obtient une visualisation correcte en mode normal.

CAR.DÉMO

```
1Ø TEXT : PRINT CHR$ (21): HGR2 : HCOLOR = 3: PRINT CHR$
   (4)"BLOAD CAR.VIS1": PRINT CHR$ (4)"BLOAD FG,A$6ØØØ"
15 CALL 768: GOSUB 4Ø
2Ø HOME : TEXT
25 VTAB 22: PRINT "(M)ENU DE DISQUETTE ";: GOSUB 4Ø
3Ø IF R$ = "M" THEN PRINT CHR$ (4)"RUN MENU,D1"
35 HOME : END
4Ø CALL - 198: GET R$: PRINT : RETURN
```

CAR.VISU2 (ASSEMBLAGE PAGE PROCODE)

```
                    Ø  *
                    1  *
                    2  Carac    EQU  $Ø6     ; Nombre de caractères.
                    3  Lgn      EQU  $Ø7     ; Pointeur ligne.
                    4  Wndwdth  EQU  $21     ; Largeur fenêtre ($28).
                    5  Bas1     EQU  $28     ; Adresse basse colonne 1.
                    6  Bas21    EQU  $2A     ; Point. ligne destination.
                    7  Hpag     EQU  $E6     ; Drapeau HGR = $2Ø, HGR2 = $4Ø.
                    8  Adrf     EQU  $5FF8   ; Adresse fonte moins 8
                    9  Vtab     EQU  $FC22   ; On utilise Vtab+2.
                   1Ø  *
                   11           ORG  $3ØØ
                   12  *
3ØØ : A2 ØØ        13           LDX  £$ØØ    ; Index X à zéro.
3Ø2 : 86 Ø6        14           STX  Carac   ; Et zéro pour Carac.
3Ø4 : 8A           15  BCL6     TXA          ; X passé dans A,
3Ø5 : 85 Ø7        16           STA  Lgn     ; puis dans pointeur ligne.
3Ø7 : 2Ø 24 FC     17           JSR  Vtab+2  ; Curseur.
3ØA : AØ ØØ        18           LDY  £$ØØ    ; Registre Y à zéro S.V.P.
```

(suite page 14)

```
30C : A5 28        19  BCL5  LDA  Basl     ; Adresse partie basse passée
30E : 85 2A        20        STA  Bas2l    ; dans Bas2l, puis partie
310 : A5 29        21        LDA  Basl+1   ; haute dans Bas2l, mais après
312 : 29 03        22        AND  £$03     ; AND 00000011 et ORA $20 ou 40
314 : 05 E6        23        ORA  HPAG     ; ($7 = $43 — $4 = $40 — $5 = $41...)
316 : 85 2B        24        STA  Bas2l+1  ; pour adresse HGR ou HGR2.
318 : A2 08        25        LDX  £$08     ; Registre X = 8 (compteur).
31A : EE 26 03     26  BCL2  INC  BCL3+1   ; Adresse basse incrémentée.
31D : D0 03        27        BNE  BCL1     ; Si le résultat n'est pas 0,
31F : EE 27 03     28        INC  BCL3+2   ; sinon on incrémente partie haute
322 : CA           29  BCL1  DEX           ; X = X — 1.
323 : D0 F5        30        BNE  BCL2     ; Si X différent de 0, encore.
325 : BD F8 5F     31  BCL3  LDA  $5FF8,X  ; Au départ, X = 0.
328 : 49 7F        32        EOR  £$7F     ; $7F = INVERSE — 00 = NORMAL.
32A : 91 2A        33        STA  ($2A),Y  ; Ecriture.
32C : 18           34        CLC           ; Annulation de la retenue.
32D : A5 2B        35        LDA  Bas2l+1  ; Partie haute majorée de $4,
32F : 69 04        36        ADC  £$04     ; pour écriture du point
331 : 85 2B        37        STA  Bas2l+1  ; suivant du caractère.
333 : E8           38        INX           ; X = X + 1.
334 : E0 08        39        CPX  £$08     ; Si X plus petit que 8,
336 : 90 ED        40        BCC  BCL3     ; on n'a pas terminé.
338 : E6 06        41        INC  Carac    ; A-t-on passé tous les
33A : A5 06        42        LDA  Carac    ; caractères de la police ?
33C : C9 60        43        CMP  £$60     ; Il y en a $60 (96).
33E : B0 0C        44        BCS  BCL4     ; Si on a terminé : vers fin.
340 : C8           45        INY           ; Sinon on saute un espace
341 : C8           46        INY           ; pour plus de clarté.
342 : C4 21        47        CPY  Wndwdth  ; Ne pas dépasser la limite.
344 : 90 C6        48        BCC  BCL5     ; C'est le cas ? suite BCL5.
346 : A6 07        49        LDX  Lgn      ; Sinon on récupère le nombre
348 : E8           50        INX           ; de lignes et on le majore de
349 : E8           51        INX           ; 2 pour plus de lisibilité.
34A : D0 B8        52        BNE  BCL6     ; Suite.
34C : A9 F8        53  BCL4  LDA  £$F8     ; Remise en état de l'adresse
34E : 8D 26 03     54        STA  BCL3+1   ; pour une autre fonte
351 : A9 5F        55        LDA  £$5F     ; (inutile si la routine est
353 : 8D 27 03     56        STA  BCL3+2   ; rechargée à chaque fois).
356 : 60           57        RTS           ; Retour au Basic.
```

BSAVE CAR.VIS2,A$300,L$57

CAR.VIS1

BSAVE CAR.VIS1,A$300,L$57

```
300 : A2 00 86 06 8A 85 07 20 24 FC A0 00 A5 28 85 2A
310 : A5 29 29 03 05 E6 85 2B A2 08 EE 26 03 D0 03 EE
320 : 27 03 CA D0 F5 BD F8 5F 91 2A 18 A5 2B 69 04 85
330 : 2B E8 E0 08 90 EF E6 06 A5 06 C9 60 B0 0E C8 C8
340 : C4 21 90 C8 A6 07 E8 E8 E0 09 90 B8 A9 F8 8D 26
350 : 03 A9 5F 8D 27 03 60
```

■

IMPRIMER UN GRAPHIQUE

POUR mener à bien l'impression d'un graphique il faut, entre autres choses, avoir compris le rôle de la carte interface, car c'est souvent elle qui fait tout échouer.

Limitons-nous aux cartes Apple (carte parallèle, ou Super-Serial). Supposons que votre carte soit sur le slot 1. En assembleur, la méthode classique pour transmettre des données à l'imprimante consiste à :

1. Placer l'adresse du slot, (c'est-à-dire $C100) dans les octets $36 et $37 en page zéro : $36 devra recevoir £$00 et $37 devra recevoir £$C1.
2. Envoyer à l'imprimante un flux d'octets par la routine COUT (=$FDED).

Par exemple :

```
LDA  £$C1
JRS  $FDED   provoque l'écriture de
             A sur votre papier.
```

Ce flux d'octets transite par la carte, qui a un droit de regard sur les valeurs qu'elle transmet à l'imprimante. Droit légitime et bénéfique si vous imprimez votre roman. Droit abusif et catastrophique si vous imprimez (ou tentez d'imprimer) un graphique.

En effet, si le dessin du graphique comporte, en un point quelconque, un assemblage de points correspondant aux valeurs $0D ou $8D (retour chariot) ou aux valeurs $09 ou $89 (Control-I), la carte va immédiatement se sentir concernée.

Après retour chariot, elle ajoute "Line feed" ($0A). Cet octet supplémentaire décale tout et fausse la signification du flux d'octets que reçoit l'imprimante.

Quant au Ctrl-I, la carte ne le transmet jamais à l'imprimante. Elle le garde dans ses profondeurs, et elle attend la suite, qu'elle interprète comme un ordre à elle donné. Les conséquences sont imprévisibles. Généralement, vous assisterez à une danse frénétique de l'imprimante, qui se mettra à éjecter des mètres de papier, en écrivant ici ou là quelques caractères bizarres, uniquement pour gâcher vos feuilles.

Remède : "Shunter" la carte (elle ne l'a pas volé), en plaçant directement vos octets sur $C090 (carte parallèle) ou sur $C098 (carte Super-Serial). En outre vous devrez, après l'envoi de chaque octet, faire une petite boucle d'attente, pour remplacer les routines qui, dans la carte, vérifient que l'imprimante est libre et peut recevoir l'octet suivant. Donc, pour transmettre à l'imprimante les octets du graphique (et uniquement ceux-là), lorsque Ac contient la valeur à transmettre, remplacez $FDED par :

```
STA  $C090   (si carte parallèle en slot 1)
STA  $C098   (si carte Super-Serial en slot 1)
LDA  £$2D
JSR  $FCA8   (boucle d'attente fonction de Ac. Tester
             la bonne valeur à prendre sur l'accumulateur,
             variable selon la rapidité de l'imprimante entre
             £$22 et £$2F).
```

Madeleine HODÉ, auteur de **Gribouille.**

SUR APPLE IIc, essayez cette petite routine :

```
10 PRINT  CHR$ (4)"PR£1": PRINT : PRIN
   T CHR$ (27)"n": PRINT  CHR$ (9)"096
   N": FOR I = 768 TO 785: READ C: POK
   E I,C: NEXT
20 READ C: POKE 773,C: IF C = 0 THEN 6
   0
30 PRINT  CHR$ (27)"G0032";
40 CALL 768:X = X + 1: IF X < 18 THEN
   30
50 PRINT :X = 0: GOTO 20
60 PRINT : PRINT  CHR$ (4)"PR£0"
70 DATA 162,32,160,16,169,0,153,136,19
   2,169,45,32,168,252,202,208,241,96:
   REM Avec une carte parallèle, rempl
   acer 136 par 128
80 DATA 13,141,9,137,0
```

Sous-programme LM (ligne 70)

```
300 :  A2  20       LDX  £$20      Slot * 16
302 :  A0  10       LDY  £$10
304 :  A9  00       LDA  £$00
306 :  99  88  C0   STA  $C088,Y   Carte parallèle : $80
309 :  A9  2D       LDA  £$2D
30B :  20  A8  FC   JSR  $FCA8
30E :  CA           DEX
30F :  D0  F1       BNE  $0302
311 :  60           RTS                ■
```

Vous obtiendrez ceci :

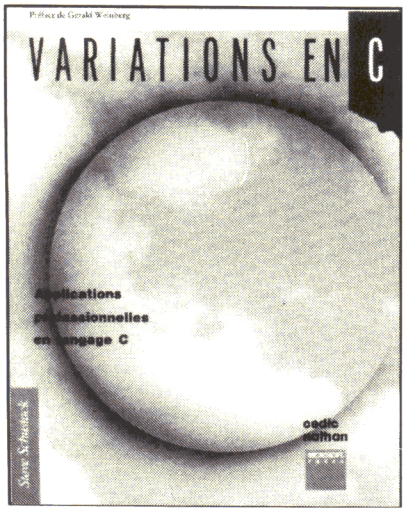
Votre bibliothèque INFORMATIQUE

par NESTOR

• VARIATIONS EN C (Steve Schustack)

Avantage non négligeable, les lecteurs de *VARIATIONS EN C* peuvent acquérir la disquette des exemples étudiés dans le livre. Ceux-ci présentent un autre intérêt : ils sont presque tous extraits d'un même programme (un logiciel de gestion), écrit sur un IBM PC (c'est là que le bât blesse un applemaniaque convaincu !) avec deux lecteurs de disquettes, 512 K de RAM, la version 2.00 de MS-DOS et la version 3.00 du compilateur C de Microsoft (que je ne connais absolument pas).

Je constate que nous disposons, sur n'importe quel Apple *IIGS* normalement constitué, des mêmes étourdissantes possibilités... du moins pour ce qui est de la RAM et des lecteurs. Pour le reste, j'espère que l'on ne va pas nous obliger à utiliser MS-DOS pour tâter du C : ce serait trop bête. Foi de Nestor, je me refuserai toujours à passer par des voies aussi tortueuses pour satisfaire ma passade pour un langage que j'ai pourtant diablement envie de m'offrir. Et je vous promets que ces *VARIATIONS EN C* sont rudement tentantes : présentation, abondance des matières et clarté des textes... C'est vrai que c'est un produit MICROSOFT PRESS : une référence à laquelle s'ajoute celle de Cedic Nathan, tout un programme !



CEDIC NATHAN *(Microsoft Press), 6 Boulevard Jourdan, 75014 PARIS*
Tél. (1) 45.65.06.06

• LA PRATIQUE D'APPLEWORKS (Michel Mossetti)

Si vous figurez au nombre des heureux élus qui ont pu s'offrir un APPLE *IIGS* pour leur petit Noël (30% de remise aux membres non largués du Club Apple), vous avez pu bénéficier d'un cadeau nullement négligeable : la version 1.4 du logiciel intégré d'Apple, AppleWorks.

Pour les néophytes, je rappelle que ce logiciel comprend non seulement un traitement de texte, mais un module de gestion de fichiers et de calcul électronique (lisez "tableur"). Pour utiliser au mieux ces outils, vous aurez tout intérêt à vous munir d'une bonne documentation, et notamment celle que vous propose Michel Mossetti. Ne souriez pas : même s'il est facile de se servir d'AppleWorks, c'est encore plus commode quand on a sous les yeux les 200 pages d'un bouquin bourré d'exemples (ils sont tous archivés dans un index des opérations). Comment établir une lettre de relance (élémentaire, mon cher Nestor), un état budgétaire ou un poulet (les jeunes ne savent pas à quoi cela ressemble !)... à une petite amie ? (Presque) tout y est !

McGRAW-HILL*, 28 Rue Beaunier, 75014 PARIS*

• LEXIQUE APPLEWORKS (Michel Mossetti)

J'ai omis de préciser que Michel Mossetti exerce la profession d'informaticien depuis plus de quinze ans (il est aussi ingénieur CNAM et enseignant au Centre National de la Recherche Scientifique). Dans ce second ouvrage, l'auteur a adopté le classement alphabétique et il est certain que celui-ci présente bon nombre d'avantages pour l'utilisateur moyen. Il supporte la version 1.3 d'AppleWorks, mais on n'y parle évidemment pas de la version 1.4, celle tournant sur l'Apple *IIGS*. Ne disposant pas, à l'heure où je rédige ces lignes, de cette nouvelle version, je n'ai pas été en mesure de la vérifier. Une chose est certaine : les exemples resteront valables, de même que l'utilisation des diverses commandes. Une remarque au passage : pour une raison que j'ignore, je n'avais jamais eu l'occasion de compulser des ouvrages de McGraw-Hill et j'ai été très agréablement surpris par leur présentation claire et sérieuse (le format, notamment, est fort bien choisi). A quand les outils du *GS* chez cet éditeur ?

McGRAW-HILL*, 28 Rue Beaunier, 75014 PARIS*

• SYSTÈMES D'EXPLOITATION DES MICRO-ORDINATEURS (M. Courvoisier, R. Valette)

Cet ouvrage présente sous forme concise les principes fondamentaux des systèmes d'exploitation utilisés dans les micro-calculateurs, puis il les illustre au moyen de quelques systèmes très répandus. Dans la première partie, la présentation des concepts part du schéma classique en couches, en insistant sur la gestion des tâches, problème de base pour l'utilisateur des micro-calculateurs dans le domaine de la commande des procédés industriels. La deuxième partie décrit quelques systèmes d'exploitation. Le choix se porte sur CP/M et MS-DOS pour les systèmes de type "ordinateur personnel" basés sur des processeurs 8 ou 16 bits, iRMX 86 pour les systèmes destinés aux applications temps réel, et UNIX qui est un système d'exploitation largement répandu dans le domaine universitaire et qui tend à devenir le standard des stations de travail basées sur des processeurs 32 bits.

DUNOD*, 17 Rue Rémy-Dumoncel, 75661 PARIS (95 F ttc)*

UTILISATION d'un fichier direct

ÉCRIRE et LIRE
REMPLACER
RECHERCHER

FIC.DIRECT

```
100 TEXT : NORMAL :D$ =  CHR$ (4): PRINT D
    $"PR£3": HOME : PRINT : PRINT D$"BLOAD
     RECHFIC1"                                  0CE2
105 F$ = "FIC":L = 042: REM Longueur maxim
    um de la donnée + 2 (guillemets au déb
    ut et return à la fin)                      573C
110 ONERR  GOTO 580                             78ED
115 VTAB 1: INVERSE : PRINT "FICHIER DIREC
    T DE LONGUEUR "L - 2: NORMAL                B230
120 :                                           003A
125 GOSUB 450: REM VERS NOMBRE DE DONNEES       1B49
130 GOTO 520                                    0942
135 :                                           003A
140 REM ***************************
145 REM *         SAISIE          *
150 REM ***************************
155 :                                           003A
160 VTAB 4: PRINT "";: CALL  - 657              807A
165 A$ = "": FOR X = 512 TO 509 + L:A =  P
    EEK (X): IF A <  > 162 AND A > 159 AND
    A <  > 141 THEN A$ = A$ +  CHR$ (A - 1
    28): NEXT                                   8E1D
170 VTAB 4: CALL  - 958: PRINT A$: IF A$ =
    "" THEN  POP : GOTO 520                     74A5
175 VTAB 22: CALL  - 868: PRINT "<C>ORRECT
     <R>EFAIRE ";: GET R$: PRINT : IF R$ =
    "R" THEN  GOSUB 190: GOTO 160               77A1
180 IF R$ <  > "C" THEN 175                     9CAB
185 RETURN                                      63B1
190 VTAB 5: POKE 1403,L - 3: PRINT "/": RE
    TURN                                        9E58
195 :                                           003A
200 REM ***************************
205 REM *         ECRIRE          *
210 REM ***************************
215 :                                           003A
220 PRINT : VTAB 4: CALL  - 958: FOR I = 1
    TO L - 2: PRINT "_";: NEXT : PRINT          B48E
225 GOSUB 160                                   1F47
230 N = N + 1: PRINT                            0B59
```

LONGUEUR D'UN ENREGISTREMENT

Elle est contenue dans la variable **L** (ligne 105). Elle est égale au nombre de caractères plus deux (le return final et les guillemets du début, qui permettent d'utiliser la ponctuation).

TITRE DU FICHIER

Il est représenté par la variable **F$** (ligne 105).

ONERR GOTO

Ne doit normalement servir que pour créer le fichier. Par la suite, il est souhaitable de supprimer cette ligne... ou de revoir le traitement très sommaire des possibles erreurs. La longueur est testée et les caractères interdits (guillemets et divers contrôles) filtrés par la ligne 165. On peut prévoir un **DISK FULL** (code ER = 9), etc.

```
235 GOSUB 470                                          E74B
240 PRINT D$                                           BF22
245 VTAB 1: HTAB 32: CALL  - 868: PRINT N:
    GOTO 220                                           63F8
250 :                                                  003A
255 REM ************************
260 REM *       REMPLACER      *
265 REM ************************
270 :                                                  003A
275 VTAB 22: CALL  - 868: INPUT "REMPLACER
     LE NUMERO ";R$:R =  VAL (R$): IF R >
    N THEN 275                                         4E10
280 IF R = 0 THEN  GOSUB 455: VTAB 20: PRI
    NT N" enregistrements": GOTO 275                   72F7
285 PRINT D$"READ"F$",R"R: INPUT A$: PRINT
    D$                                                 6A7F
290 VTAB 4: CALL  - 958: PRINT A$                      A864
295 GOSUB 190: GOSUB 160                               2CCB
300 GOSUB 475: PRINT D$: GOTO 520                      6828
305 :                                                  003A
310 REM ************************
315 REM *        CHERCHER      *
320 REM ************************
325 :                                                  003A
330 V = 3: VTAB 22: CALL  - 868: INPUT "CH
    AINE RECHERCHEE:ctG ";CH$: IF CH$ = ""
    THEN 520                                           CFB6
335 LC =  LEN (CH$)                                    8642
340 FOR I = 1 TO N                                     E1DA
345 PRINT D$"READ"F$",R"I: INPUT A$: PRINT
    D$                                                 B776
350 CALL 768,CH$,A$: IF  NOT  PEEK (24) TH
    EN GOSUB 360                                       1FF0
355 NEXT I: GOTO 330                                   7146
360 V = V + 1: VTAB V: CALL  - 958: PRINT
    I;: HTAB 4: PRINT A$                               57B1
365 IF V = 20 THEN  CALL  - 198: VTAB 22:
    PRINT "PRESSEZ UNE TOUCHE ";: GET R$:
    PRINT :V = 3                                       81D4
370 RETURN                                             63B1
375 :                                                  003A
380 REM ************************
385 REM *         LIRE         *
390 REM ************************
395 :                                                  003A
400 VTAB 22: CALL  - 868: INPUT "LIRE LE N
    UMERO ";R$:R =  VAL (R$): IF R > N THE
    N 400                                              C097
405 IF R$ = "" THEN 520                                3392
410 IF R = 0 THEN  GOSUB 455: VTAB 20: PRI
    NT N" enregistrements": GOTO 400                   36ED
415 PRINT D$"READ"F$",R"R: INPUT A$: PRINT
    D$                                                 6A7F
420 VTAB 4: CALL  - 958: PRINT A$: GOTO 40
```

PRINT D$

D$** contient CHR$(4), commande comprise par le DOS et par ProDOS. Quand un fichier est ouvert, après une commande **WRITE ou **READ**, ne pas oublier de reprendre la main pour un PRINT D$, notamment pour un INPUT... sous peine de se retrouver dans la nature !

PONCTUATION

On a vu plus haut qu'il est possible de l'utiliser en passant par **CALL-657** ($FD6F). Ne pas oublier que cette routine ne filtre pas naturellement les mauvais caractères (contrôles divers et ESCAPE).

CARACTÈRE INTERDIT

Ne pas utiliser les guillemets.

RECHERCHE

Elle est effectuée par une routine en langage machine (voir plus loin). Celle-ci vous autorise à utiliser indifféremment les minuscules ou les majuscules. Attention ! les caractères accentués conservent par contre leur valeur. Sur la disquette, **RECHFIC0** permet de différencier les minuscules et les majuscules.
Pour rechercher un signe de ponctuation (normalement refusé lors de sa saisie), tapez par exemple : "," ou ":"...

COMPATIBILITÉ

Ce programme fonctionne sous **DOS** et **ProDOS**, mais vous constaterez qu'il se révèle beau-

coup plus performant avec **ProDOS**. La recherche, notamment, est plus rapide, le tampon autorisant la lecture d'un nombre supérieur d'octets par séquence.

IMPORTANT

N'oubliez pas que le fichier direct est gourmand en mémoire-disquette. Si vous fixez la longueur d'un enregistrement à 200, par exemple, le mot "Toto" exigera le même nombre d'octets qu'une phrase de 200 caractères. Par contre, on accède facilement et instantanément à n'importe quelle donnée existante.

NOMBRE

Fonction de la disquette utilisée... Il est stocké dans l'enregistrement 0 et contenu dans la variable N. La variable R est utilisée pour le numéro d'un enregistrement à lire ou à remplacer.

```
    0                                                    03DD
425 :                                                    003A
430 REM ************************
435 REM *     SUR LE FICHIER     *
440 REM ************************
445 :                                                    003A
450 PRINT D$"OPEN"F$",L"L                                BF0A
455 PRINT D$"READ"F$",R0"                                CADE
460 INPUT N: PRINT D$: VTAB 1: HTAB 36: PR
    INT N: RETURN                                        7CA1
465 PRINT D$"WRITE"F$",R0": PRINT N: RETUR
    N                                                    EC7A
470 PRINT D$"WRITE"F$",R"N: GOTO 480                     D3EC
475 PRINT D$"WRITE"F$",R"R                               3E6F
480 PRINT  CHR$ (34) + A$: RETURN                        7271
485 PRINT D$"CLOSE"F$                                    7D46
490 : HOME : END                                         8E8B
495 :                                                    003A
500 REM ************************
505 REM *     LIGNE DE CHOIX     *
510 REM ************************
515 :                                                    003A
520 VTAB 22: PRINT "<L>IRE "<E>CRIRE "<R>E
    MPLACER "<C>HERCHER "<F>INctG ";: GET
    R$: PRINT : IF R$ >  CHR$ (82) THEN R$
    =  CHR$ ( ASC (R$) - 32)                             2FF1
525 IF R$ = "L" THEN 400                                 4BDB
530 IF R$ = "E" THEN 220                                 7ED4
535 IF R$ = "R" THEN 275                                 81EB
540 IF R$ = "C" THEN 330                                 3CD4
545 IF R$ = "F" THEN  GOSUB 465: GOTO 485                3F16
550 GOTO 520                                             0942
555 :                                                    003A
560 REM ************************
565 REM *        ERREUR          *
570 REM ************************
575 :                                                    003A
580 ER =  PEEK (222): POKE 216,0: IF ER <
    > 5 THEN 590                                         ABCD
585 GOSUB 465: PRINT D$: GOTO 520                        3C27
590 VTAB 22: PRINT D$: PRINT "ERREUR IMPRE
    VUE ": GOSUB 465: GOTO 485                           B72B
```

PAGE SUIVANTE: EXPLICATIONS DE <RECHFIC1>

CODES DE RECHFIC1 (ligne 100)

```
0300: 20 BE DE 20 E3 DF A0 02 B1 83 99 18 00 88 10 F8    E4B5
0310: 20 BE DE 20 E3 DF A0 02 B1 83 99 06 00 88 10 F8    88A3
0320: A4 18 B1 19 99 5E 03 88 10 F8 A4 06 A6 18 CA 10    5C52
0330: 04 E8 86 18 60 88 10 01 60 BD 5E 03 D1 07 F0 EE    E9B7
0340: 49 20 C9 41 90 10 C9 5B 90 08 C9 61 90 08 C9 7B    2ED5
0350: B0 04 D1 07 F0 D8 E8 E4 18 F0 D1 C8 D0 F8          C689
```

BSAVE RECHFIC1,A$300,L$5E

RECHERCHE D'UNE VARIABLE DANS UNE AUTRE

(routine **RECHFIC** de **FIC.DIRECT**)

```
300 : 20 BE DE    JSR  $DEBE
303 : 20 E3 DF    JSR  $DFE3
306 : A0 02       LDY  £$02
308 : B1 83       LDA  ($83),Y
30A : 99 18 00    STA  $0018,Y
30D : 88          DEY
30E : 10 F8       BPL  $0308
310 : 20 BE DE    JSR  $DEBE
313 : 20 E3 DF    JSR  $DFE3
316 : A0 02       LDY  £$02
318 : B1 83       LDA  ($83),Y
31A : 99 06 00    STA  $0006,Y
31D : 88          DEY
31E : 10 F8       BPL  $0318
320 : A4 18       LDY  $18
322 : B1 19       LDA  ($19),Y
324 : 99 5E 03    STA  $035E,Y
327 : 88          DEY
328 : 10 F8       BPL  $0322
```

CHKCOM teste la présence de la virgule et PTRGET recherche la variable par son nom.

On sait lire la longueur de la variable à l'adresse trouvée en $83-84. La même adresse +1 et +2 fournit l'adresse réelle de la chaîne.

Mêmes opérations pour la seconde variable. La première représentait la chaîne recherchée, celle-ci la chaîne dans laquelle on recherche. Le choix des adresses $18-1B et $6-8 est arbitraire.

Longueur de la chaîne à rechercher dans Y.

Ecriture des Y octets de cette chaîne à partir de $35E. On notera que L est en vérité égal à L + 1 (0 à L), mais c'est sans importance.

RECHERCHE

```
32A : A4 06       LDY  $06
32C : A6 18       LDX  $18
32E : CA          DEX
32F : 10 04       BPL  $0335
331 : E8          INX
332 : 86 18       STX  $18
334 : 60          RTS
335 : 88          DEY
336 : 10 01       BPL  $0339
338 : 60          RTS
339 : BD 5E 03    LDA  $035E,X
33C : D1 07       CMP  ($07),Y
33E : F0 EE       BEQ  $032E
340 : 49 20       EOR  £$20
342 : C9 41       CMP  £$41
344 : 90 10       BCC  $0356
346 : C9 5B       CMP  £$5B
348 : 90 08       BCC  $0352
34A : C9 61       CMP  £$61
34C : 90 08       BCC  $0356
34E : C9 7B       CMP  £$7B
350 : B0 04       BCS  $0356
352 : D1 07       CMP  ($07),Y
354 : F0 D8       BEQ  $032E
356 : E8          INX
357 : E4 18       CPX  $18
359 : F0 D1       BEQ  $032C
35B : C8          INY
35C : D0 F8       BNE  $0356
```

Longueur de la grande chaîne dans Y.
Longueur de la petite dans X.

X = X - 1. Jusqu'à X = 0, on saute.

Ça, c'est pour renseigner le Basic.
Si PEEK(24) = 0, on a trouvé A dans B !

Y = Y - 1. Jusqu'à Y = 0, on continue.

Sinon c'est fini et, $18 contient toujours la longueur de la chaîne recherchée... et PEEK (24) n'est pas égal à 0.

COMPARAISON 1

Si c'est oui, on passe au caractère suivant.
Où A devient a et Z... z, on vice-versa.

Si plus petit que "A", pas de comparaison.

Si plus petit que $5B (plus grand que "Z"), saut vers COMPARAISON2.

Si plus petit que "a", pas de comparaison.

Si plus grand que "z",
pas de comparaison non plus.

COMPARAISON2

Si égalité, vers caractère suivant.

On réajuste la valeur de Y pour que la recherche reprenne à Y + 1.

■

Marcel **COTTINI**

SYSTÈME D'EXPLOITATION ProDOS

Le dernier-né des systèmes d'exploitation, en l'occurrence ProDOS, suscite chez bon nombre de lecteurs pratiquant l'Apple *II*, certaines réticenses à l'emploi. Cet article a la tâche de familiariser l'utilisateur débutant ou averti, avec les programmes systèmes et lui donner tous les ingrédients pour une meilleure utilisation de ProDOS.

ProDOS, Professional Disk Operating System, est un ensemble de micro-programmes qui, associés les uns aux autres, forment un programme système, capable de gérer les divers périphériques reliés à un environnement Apple *II*. C'est une suite d'instructions faisant partie des programmes systèmes (Operating System), permettant entre autres, la gestion d'un lecteur de disquettes à l'aide de commandes évoluées. Lors des différentes manipulations, l'opérateur n'a pas besoin de s'occuper du déplacement de la tête de lecture, ni de se soucier de la gestion et de l'écriture des divers fichiers sur la disquette. Ce rôle est attribué au système d'exploitation.

Comme tout programme (soft), ProDOS occupe une place en mémoire centrale, suivant ses possibilités intrinsèques. En règle générale, tout concepteur de logiciels effectue un compromis entre les possibilités offertes par le matériel (hard) et le logiciel (soft). Pour le fichier PRODOS, la mémoire se trouve amputée de 12 Ko, l'Apple *IIe* ou *IIc* répartit d'une façon astucieuse cette occupation. Les 12 Ko en haut de la mémoire sont attribués au fichier PRODOS par commutation alternée du système et de la ROM. Cet espace est normalement occupé par la ROM AppleSoft et par le Moniteur. En réalité, le fichier PRODOS occupe 16 Ko, dont 4 Ko sont obtenus par la commutation de la Bank Switched Memory — bank 1 et 2.

ProDOS, développé par la société Apple Computer, est le dernier-né d'une gamme de programmes systèmes, destiné à fonctionner sur la série des Apple *II*. Il permet d'accroître considérablement la sauvegarde de la mémoire de masse, par l'utilisation d'un disque dur. Ce système a fait son apparition officielle début 1984 avec la version 1.Ø ; une nouvelle version revue et corrigée, circule depuis début 1985. Une autre accompagne les logiciels les plus récents. La dénomination ProDOS recouvre en fait l'association d'un système d'exploitation et d'un programme interface.

STRUCTURE DU SYSTÈME D'EXPLOITATION ProDOS

Le système d'exploitation ProDOS se compose :

— d'un fichier PRODOS qui se loge dans la carte langage, et regroupe tous les sous-programmes en langage machine — MLI —. Il se trouve sauvegardé sous la forme d'un programme du type SYS, avec comme nom de fichier : PRODOS ou KERNEL.

— d'un fichier également du type SYS, sous la forme XXX.SYSTEM, et qui se loge à l'ancien emplacement du DOS 3.3.

Chaque disquette de démarrage standard ProDOS comporte au moins deux fichiers communs écrits en langage machine : PRODOS et XXX.SYSTEM.

Remarque : XXX.SYSTEM représente un format dont les XXX désignent un nom de fichier pouvant être tout programme répondant à cette convention, par exemple un traitement de texte en langage machine. En général c'est BASIC. SYSTEM qui est utilisé pour la plupart des disquettes conventionnelles.

Fichier PRODOS

Ecrit en langage machine, il renferme tous les sous-programmes et routines MLI (Machine Language Interface). C'est le cœur du système (noyau — KERNEL), permettant de gérer tous les fichiers autres que ceux du type AppleSoft. Il est l'équivalent du File Manager et de RWTS du DOS 3.3 ; il occupe l'ensemble de la carte langage (à l'exception de la zone $D000-$D0FF de la bank 2), ainsi que l'espace $BF00-$BFFF servant à commuter les différentes parties hautes de la mémoire, et à recevoir des appels de programmes ou de routines externes.

PRODOS fut écrit au départ pour fonctionner sur un Apple *II* de 48 Ko de mémoire, *(suite page 22)*

en se plaçant sous l'adresse $C000, et sur un *IIe* de 64 Ko, en se plaçant sur la carte langage. Cette dernière version a été retenue pour la commercialisation. Elle se trouve sur la disquette sous la forme d'un fichier du type SYS, qui ne peut être que "booté". PRODOS se loge sur les 16 Ko de la carte langage d'un Apple *II+* ou en haut de la mémoire sur un *IIe* et un *IIc*, et fixe HIMEM à $BF00. Lorsque PRODOS est chargé en mémoire, les commandes exécutées sont des commandes PRODOS. Lorsque celui-ci reçoit une commande qu'il ne peut pas interpréter, il la transmet à l'AppleSoft qui teste sa validité avant de l'utiliser. Si, pour une raison quelconque, l'ordre n'aboutit pas, un message d'erreur sera renvoyé par le système Basic.

Fichier BASIC.SYSTEM

Ecrit en langage machine, il est l'interface entre un programme AppleSoft et le fichier PRODOS. Il permet de passer la main au Basic pour exécuter des commandes de l'interpréteur AppleSoft. Il est logé aux adresses $9600-$BEFF. Comme PRODOS, BASIC.SYSTEM est un fichier du type SYS, uniquement "bootable". BASIC.SYSTEM est sollicité en mémoire, à chaque fois qu'il s'agit de traiter des programmes du type AppleSoft. Lorsque celui-ci est chargé en mémoire centrale, HIMEM est fixé à l'adresse équivalente du DOS 3.3 — $9600 —

ALLOCATION DES BLOCS SUR UNE DISQUETTE ProDOS

Après le formatage d'une disquette ProDOS par le programme utilitaire FILER, celle-ci présente un certain nombre de blocs occupés et de blocs libres. Le tableau qui suit reflète l'image type d'une disquette après formatage sous ProDOS.

Blocs alloués par ProDOS	
Blocs 0-1	Loader
Blocs 2-5	Volume-Directory
Bloc 6	Bit-Map
Blocs 7-279	Volume-Subdirectory
	Fichiers systèmes
	Utilisateur

Le bloc $02 est généralement désigné sous l'appellation Key-Block et les blocs $03 à $05 sont appelés Non Key-Block.

Capacité physique d'une disquette	
Pistes	35 (0-34)
Secteurs par piste	16 (0-15)
Secteurs par disquette	560
Bytes par secteur	256
Bytes par piste	4 096
Bytes par disquette	143 360

Capacité logique d'une disquette	
280 blocs par disquette	de 0 à 279 ($000-$117)
8 blocs par piste	de 0 à 7 ($00-$07)
1 bloc	2 secteurs
512 bytes par bloc	de 0 à 511 ($000-$1FF)

Un bloc se divise en deux parties : partie gauche et partie droite. Les 256 premiers bytes d'un bloc sont ceux de gauche, les 256 bytes suivants, ceux de droite.

Capacité d'une disquette après formatage :
Blocs disponibles : 273 blocs
Bytes disponibles : 139 776 bytes.

A ce total, il faudra cependant retrancher le nombre de blocs nécessaires pour la sauvegarde des fichiers PRODOS et BASIC.SYSTEM, indispensables pour démarrer le système.

PRODOS version 1.0.2 : 31 blocs
BASIC.SYSTEM version 1.0.2 : 21 blocs
Total = 52 blocs ou 26 624 bytes
Bytes réellement disponibles : 139 776 - 26 624 = 113 152 bytes.

NOTION DE VOLUME BIT-MAP ET SYSTEM BIT-MAP

Deux Bit-Map sont disponibles pour le système ProDOS :

— System Bit-Map — SBM — de la page globale de PRODOS, aux adresses $BF59-$BF6F
— Volume Bit-Map — VBM — sur la disquette, bloc 6.

System Bit-Map — RAM 48 Ko

Pour le traitement des fichiers texte, ProDOS gère automatiquement les pages allouées aux tampons mémoire par la commande OPEN. Le programmeur peut, avant l'implantation d'un programme en mémoire centrale, solliciter par exemple une routine qui pourra fournir la première page disponible.

Implantation d'un programme en mémoire :

— Il sera au préalable chargé au bas de la zone mémoire non encore utilisée par le système.

— A ce programme sera associé un sous-programme Move, qui aura une double fonction : modifier la System Bit-Map et le pointeur de HIMEM, puis placer le programme à son adresse définitive.

ProDOS suit les mêmes règles de chargement, lors du boot d'une disquette.

Volume Bit-Map — Implanté sur la disquette

C'est la table d'allocation des blocs d'une disquette — bloc 6. Une disquette 35 pistes, utilise les 35 premiers bytes du Volume Bit-Map pour gérer l'occupation de ses blocs. Chaque type de la table marque 8 blocs, ce qui fait un total de :
35 * 8 = 280 blocs.

La table occupe 512 bytes (1 bloc) et peut gérer à la limite : 8 * 512 = 4 096 blocs, mais seuls 280 blocs sont utilisés par les versions actuelles de ProDOS. Chaque bit représente un bloc : si le bit est à 1, le bloc est libre, si par contre il est à 0, le bloc est occupé.

Table d'allocation des blocs au niveau de la disquette :
Bloc 06 d'une disquette ProDOS

Byte $00	=	Piste $00	=	Blocs $000-$007
Byte $01	=	Piste $01	=	Blocs $008-$00F
Byte $02	=	Piste $02	=	Blocs $010-$017
Byte $22	=	Piste $22	=	Blocs $110-$117

La valeur du byte est donnée suivant la position relative dans la table d'occupation, au niveau de la disquette ProDOS.

Représentation d'un byte :

Byte $00
Blocs $000-$007 0 1 2 3 4 5 6 7
0 0 0 0 0 0 0 1 = 01

Le bloc $007 est libre et les blocs $000-$006 occupés.

Byte $01
Blocs $008-$00F 8 9 A B C D E F
1 1 1 1 1 1 1 1 = FF

Les blocs $008-$00F sont tous libres.

Dump de la Bit-Map d'une disquette ProDOS

Le premier byte de la Bit-Map affiche 01 et désigne un bloc libre : c'est le bloc 7 de la disquette. Les trente-quatre bytes suivants, sont tous positionnés à la valeur FF (255 en décimal), les blocs 8 à 279 sont inutilisés (8 * 34 = 272 blocs). Le total des blocs libres est : 272 + 1 = 273 blocs.

Dump de la Bit-Map d'une disquette ProDOS
Block 00 06

$00	01	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
$10	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
$20	FF	FF	FF	00	00	00	00	00	00	00	00	00	00	00	00	00
$30	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$40	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$80	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
$F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

ORGANISATION INTERNE DE ProDOS
TABLES DES MATIÈRES

— Volume-Directory
— Volume-Subdirectory

ORGANISATION DU FICHIER PRODOS

— Version 1.0.2

Par comparaison au DOS 3.3, le fichier PRODOS n'occupe pas de place bien précise sur la surface de la disquette. Pour cette raison, il sera fait référence à une position relative (0 dans notre cas) par rapport à une valeur arbitraire donnée au bloc de départ. L'adresse de départ, $2000 dans notre listing, est celle où le fichier PRODOS est implanté par le Loader au moment du boot de la disquette et juste avant l'appel de la routine Move.

ADRESSES ET VECTEURS DU MLI

• Routine Move — Relocator.

Sous-programme pour transférer PRODOS dans la carte langage.
Blocs 04 : adresses $2000-$29FF.

• RAM Disk-Driver
Bloc 5 : adresses $2A00-$2BFF. *(suite page 24)*

— RAM Disk-Driver (64 Ko auxiliaires $0200-$03FF).
Bloc 6 : *partie gauche, adresses $2C00-$2C7E.*
— RAM Disk-Driver ($FF00-$FF7E Bank 1).
Bloc 6 : partie gauche, adresses $2C7F-$2CFF.
— Bytes nuls.

• Le MLI — Machine Language Interface.

Bloc 6 : partie droite, adresses $2D00-$2DFF.
— Début du MLI ($D000-$D0FF Bank 1).
Blocs 7-21 : adresses $2E00-$4BFF.
— MLI ($D100-$EEFF Bank 1).
Bloc 22 : partie gauche ; adresses $4C00-$4CFF.
— Fin du MLI ($EF00-$EFFF Bank 1).
Bloc 22 : partie droite ; adresses $4D00-$4D05.
— MLI ($F000-$F005 Bank 1).
Bloc 22 : partie droite ; adresses $4D06-$4DFF.
— Bytes nuls.

• La Page Globale de PRODOS

Bloc 23 :partie gauche ; adresses $4E00-$4EFF.
— Page Globale pour appel de PRODOS dans la carte langage adresses $BE00-$BEFF.
Bloc 23 : partie droite ; adresses $4F00-$4FFF.
— Page Globale du BASIC.SYSTEM ($BF00-$BFFF).

• Les Interruptions

Bloc 24 : partie gauche ; adresses $5000-$5069.
— Routine Horloge ($F142-$F1AB).
Bloc 24 : partie gauche ; adresses $506A-$507C.
— Date ($F1AC-$F20E).
Bloc 24 : partie gauche ; adresses $507D-$509A.
— Bytes nuls.
Bloc 24 : partie gauche ; adresses $509B-$50EB.
— Interrupt ($FF9B-$FFEB).
Bloc 24 : partie gauche ; adresses $50EC-$50F9.
— Bytes nuls.
Bloc 24 : partie gauche ; adresses $50FA-$50FF.
— NMI, IRQ et Reset ($FFFA-$FFFF).
Bloc 24 : partie droite ; adresses $5100-$51FF.
— Bytes nuls.

• Le Disk-Driver

Bloc 25 : partie gauche ; adresses $5200-$5395.
— Disk-Driver ($F800-$F995).
Bloc 25 : partie gauche ; adresses $5396-$53FF.
— Données du Disk-Driver ($F996-$F9FF).
Bloc 25 : partie droite ; adresses $5400-$54FF.
— Table des nibbles ($FA00-$FAFF).
Bloc 26 : partie gauche ; adresses $5500-$5572.
— Bytes nuls.
Bloc 26 : partie droite ; adresses $5573-$5584.
— Données Disk-Driver ($FB74-$FB84).
Bloc 26 : partie droite ; adresses $5585-$55FF.
— Disk-Driver ($FB85-$FBFF).
Bloc 27 : adresses $5600-$57FF.
— Disk-Driver ($FC00-$FDFF).
Bloc 28 : partie gauche ; adresses $5800-58BD.
— Disk-Driver ($FE00-$FEBD).
Bloc 28 : partie gauche ; adresses $58BE-$58FF.
— Bytes nuls ($FEBE-$FEFF).

• La routine REBOOT

Bloc 28 : partie droite ; adresses $5900-$59FF.
— Début de Reboot ($D100-$D1FF Bank 2).
Bloc 29 : adresses $5A00-$5BFF.
— Routine Reboot ($D200-$D3FF Bank 2).

TABLE DES MATIÈRES PRINCIPALE

Le Volume-Directory — Entrée du nom d'un Volume.

Ce type de Directory est implanté aux blocs 0002-0005 de la disquette. Par comparaison au Subdirectory, le Volume-Directory ne permet pas sa duplication. Le Volume-Directory occupe 4 blocs sur une disquette standard au format 5 pouces 1/4, pouvant contenir un total de 51 noms de fichiers.

Les quatre premiers bytes de chaque bloc du Directory sont réservés pour les pointeurs arrière et avant — Backward et Forward. Le pointeur avant — Forward Pointer — désigne à l'intérieur du Directory, le numéro du bloc qui précède son propre bloc.

Le pointeur arrière — Backward Pointer — désigne à l'intérieur du Directory, le numéro du bloc qui suit son propre bloc.

Au niveau du Key-Block, bloc 0002, le pointeur arrière affublé de la valeur 0000 signifie que c'est le premier bloc du Directory.

Au niveau du bloc 0005, Non Key-Block, le pointeur avant affublé de la valeur 0000 signifie que c'est le dernier bloc du Directory.

VOLUME-DIRECTORY

Key Block Header Volume Name — Block 00 02

Le premier nom qui figure dans la table des matières principale (Volume -Directory) est celui du Volume sauvegardé lors du formatage de la disquette par le programme FILER. Le Volume-Directory permet la sauvegarde d'un certain nombre de paramètres particuliers.

Position	Contenu
$00-$01	Pointeur arrière — LByte, HByte : 00 00 ; 0000 signifie que c'est le premier bloc.
$02-$03	Pointeur avant — LByte, HByte : 03 00 ; $0003 pointe le bloc 3 du Directory.
$04	Storage Type/Name Length. 4 bits de poids fort qui caractérisent la sauvegarde d'un fichier. 4 bits de poids faible qui désignent la longueur du nom du Volume. **Exemple :** $F6 ou F = Fichier Volume et 6 = 6 caractères.

$05-$13	Nom du fichier — USERS.DISK par ex.
$14-$1B	Bytes réservés pour une application future.
$1C-$1F	Date et heure de la sauvegarde d'un nom du Volume-Directory.
$20-$21	Versions de PRODOS.
$22	Access Byte. Byte d'accès au fichier. Exemple : $C3 = Lecture/écriture permise.
$23	Nombre de bytes alloués à chaque entrée du Volume-Directory : — 39 bytes pour chaque entrée Volume.
$24	Nombre d'entrées possibles par bloc. — 13 entrées par bloc.
$25-$26	Total des noms de fichiers actuellement sauvegardés dans la table des matières ou Volume-Directory : — maximum 51 fichiers.
$27-$28	Pointeur de la Bit-Map — 06 00 = 0006.
$29-$2A	Capacité totale de la disquette — Blocs : — 18 01 = 0118 = 280 blocs.

VOLUME-DIRECTORY

Position relative des entrées Configuration d'une entrée

La suite de la table des matières, après la sauvegarde des paramètres du Volume-Directory, permet la gestion de 51 noms de fichiers. Le bloc 0002 ou Key Block peut contenir 12 noms de fichiers (13 moins celui du nom du Volume) et les 3 blocs suivants, 13 noms chacun, ce qui fait un total de : (13 * 3) + 12 = 51 noms de fichiers.

Structure type d'un bloc d'entrées	
Pointeurs avant et arrière	4 bytes
13 entrées de noms de fichiers	507 bytes
Total	511 bytes

Le byte à la position 512 est inutilisé au niveau des 4 blocs du Volume-Directory et contient la valeur 00.

Chaque bloc d'une table des matières se divise en un certain nombre d'entrées de noms de fichiers, dont chacune peut être désignée par un pointeur relatif : pointeur de début $00 ; pointeur de fin $26 ; total £$27, soit 39 bytes par entrée.

Points d'entrées des noms de fichiers dans le Directory-Block		
N° d'ordre des entrées	Positions	
	Hex.	Déc.
Entrée 1	$0004	004
Entrée 2	$002B	043
Entrée 3	$0052	082
Entrée 4	$0079	121
Entrée 5	$00A0	160
Entrée 6	$00CF	199
Entrée 7	$00EE	238
Entrée 8	$115	277
Entrée 9	$13C	316
Entrée 10	$163	355
Entrée 11	$18A	394
Entrée 12	$1B1	433
Entrée 13	$1D8	472

VOLUME-DIRECTORY

Non Key Blocks Deuxième bloc du Directory — Block 00 03

Position	Contenu
$00-$01	Pointeur arrière — LByte, HByte. Pointe le bloc précédent : 02 00 ; 0002 = bloc 02.
$02-$03	Pointeur avant — LByte, HByte. Pointe le bloc suivant : 04 00 ; $0004 = bloc 04.
$04-$2A	Première entrée d'un nom de fichier.
$2B-$51	Deuxième entrée d'un nom de fichier. . . . .
$1D8-$1FE	Treizième entrée d'un nom de fichier.
$1FF	Inutilisé.

Troisième bloc du Directory — Block 00 04

Position	Contenu
$00-$01	Pointeur arrière — LByte, HByte. Pointe le bloc précédent : 03 00 ; 0003 = bloc 03.

(suite page 26)

$02-$03	Pointeur avant — LByte, HByte. Pointe le bloc suivant : 05 00 ; 0005 = bloc 05.
$04-2A	Première entrée d'un nom de fichier.
$2B-$51	Deuxième entrée d'un nom de fichier. . . . .
$1D8-$1FE	Treizième entrée d'un nom de fichier.
$1FF	Inutilisé.

Quatrième bloc du Directory — Block 00 05

Position	Contenu
$00-$01	Pointeur arrière — LByte, HByte. Pointe le bloc précédent : 04 00 ; 0004 = bloc 04.
$02-$03	Pointeur avant — LByte, HByte. Pointe le bloc suivant. Ce pointeur est nul : dernier bloc.
$04-$2A	Première entrée d'un nom de fichier.
$2B-$51	Deuxième entrée d'un nom de fichier. . . . .
$1D8-$1FE	Treizième entrée d'un nom de fichier.
$1FF	Inutilisé.

VOLUME-DIRECTORY

Entrée d'un nom de fichier

La position relative des paramètres de chaque entrée du nom d'un fichier dans la table des matières est donnée par rapport à chaque entrée possible. Chaque pointeur débute avec la valeur $00 et termine avec la valeur $26 (£$27 ou 39 bytes alloués à chaque fichier). Chaque bloc peut contenir 13 noms de fichiers accompagnés de ses caractéristiques, le byte 512 étant toujours inutilisé.

Une table des matières principale peut contenir un total de 52 noms de fichiers, le premier nom étant réservé à celui du nom du Volume de la disquette.

Entrée d'un nom de fichier Positions relatives $00-$26

Position	Contenu
$00	Storage Type/Name Length. 4 bits de poids fort ayant pour valeur la caractéristique de sauvegarde du fichier. 4 bits de poids faible qui désignent la longueur du nom du fichier.
$01-$0F	Nom du fichier programme — 15 caractères maximum.
$10	File Type. Type du fichier — $0B = BIN.
$11-$12	Pointe le premier bloc du fichier. 12 00 = 0012 ; pointe le bloc $0012.
$13-$14	Blocks Used. Nombre total de blocs occupés par le fichier : 45 00 = 0045 ; blocs occupés = $0045.
$15-$17	EOF Pointer. Longueur d'un fichier texte. Fin d'un fichier binaire : 55 00 00 = $000055 bytes de longueur.
$18-$1B	Date & Time. Date et heure de création du fichier. Format : 00 00 00 00. — 2 bytes pour la date et 2 bytes pour l'heure, avec une carte horloge.
$1C-$1D	Versions de PRODOS.
$1E	Access Byte. Paramètre d'accès au fichier.
$1F-$20	Aux. Info. Information auxiliaire : adresse d'implantation du programme par exemple.
$21-$24	Date & Time. Date et heure de la modification. Format : 00 00 00 00. — 2 bytes pour la date et 2 bytes pour l'heure, avec une carte horloge.
$25-$26	Pointe le Key-Block d'un Directory dans lequel se trouve sauvegardé le nom du fichier.

TABLE DES MATIÈRES AUXILIAIRE

Volume-Subdirectory

Cette table des matières particulière est souvent désignée sous les termes *Subdirectory, sous-catalogue* ou encore *table des matières auxiliaire.* Ce type de fichier, désignant cette table des matières, ne peut être créé que par la commande CREATE. Le nom du fichier Subdirectory se trouve sauvegardé dans le Volume-Directory, il pointe le Volume-Subdirectory. Ce dernier, appelé très sou-

vent Key-Block Subdirectory, occupe au départ un bloc. Il est possible de sauvegarder dans ce nouveau Volume, des fichiers programmes ou Subdirectory. Le fichier ainsi créé sera du type DIR et aura comme Storage byte la valeur DC, tandis que dans le Volume-Subdirectory, le nom du Volume aura comme valeur EC. Les quartets D et E représentent Storage Type, tandis que le quartet C équivaut à Name Length.

Entrée d'un nom de Volume du Subdirectory Header-Entry — Key-Block

Les 4 premiers bytes sont réservés aux pointeurs arrière et avant, suivant le même principe que la représentation du Volume-Directory. Deux bytes pour le pointeur arrière — Backward Pointer — et 2 bytes pour le pointeur avant — Forward Pointer. A la suite, se trouvent les 39 bytes de la première entrée de la table des matières auxiliaire — Volume-Subdirectory.

En opposition au Volume-Directory, le Volume-Subdirectory n'a qu'un nombre limité de blocs. Lorsque le Subdirectory est créé par CREATE, BASIC.SYSTEM + commande MLI, le système lui attribue un seul bloc. S'il arrive par la suite que plus de 13 noms de fichiers doivent y être sauvegardés, un deuxième bloc sera chaîné au premier (nom du Volume-Subdirectory compris). Etant donné la structure de chaînage d'un Subdirectory, les blocs ne peuvent en aucun cas se suivre dans le sens ascendant.

Position	Contenu
$00-$01	Pointeur arrière — 00 00 = $0000.
$02-$03	Pointeur avant — 00 00 = $0000.
$04	Storage Type/Name Length. 4 bits de poids fort qui caractérisent le type de sauvegarde. 4 bits de poids faible qui désignent la longueur du nom du Volume-Subdirectory.
$05-$13	Nom du Volume-Subdirectory — SUBDIRECTORY par exemple. Maximum 15 caractères.
$14-$1B	Le nom de "uHUSTON !" est sauvegardé à cet endroit, mais il n'a aucune signification précise.
$1C-$1F	Create Date & Time. Date et heure de création du Volume-Subdirectory.
$20-$21	Versions de PRODOS.
$22	Access Byte. Paramètre d'accès au fichier — $C3.
$23	Longueur d'une entrée dans la table des matières — 39 ou £$27 bytes.
$24	Nombre d'entrées possibles par bloc — 512/39 = 13 ou 0D.
$25-$26	Nombre de fichiers actuellement sauvegardés dans le Volume-Subdirectory — 01 00 = £$0001 fichier.
$27-$28	Parent Pointer. Numéro du bloc de la table des matières où est sauvegardé le nom du fichier qui pointe ce Volume-Subdirectory. 02 00 = $0002 = bloc 2, si le nom du fichier se trouve sauvegardé dans le bloc 2.
$29	Parent Entry. Numéro de l'entrée dans le bloc parent.
$2A	Longueur d'une entrée dans la table des matières du parent, c'est-à-dire le Directory dans lequel se trouve le nom du fichier Subdirectory.

LES BLOCS INDEX

Sous le système d'exploitation DOS 3.3, le bloc index (Index Block) est référencé sous le terme de *Track Sector List.*

— 1 bloc index = 2 secteurs = 512 bytes.

— Une disquette ProDOS est composée de blocs de données d'une taille de 512 bytes chacun.

BLOCS RÉPERTOIRES : Index Blocks.

Il existe deux types de blocs index :

— ***Master Block.*** C'est un bloc répertoire principal, qui peut pointer 128 blocs répertoires auxiliaires (Index Blocks) dont chacun peut à nouveau pointer 256 blocs de données. Ce type de bloc pointe toujours un bloc index, mais jamais un bloc de données d'un programme. Un fichier avec un Storage Type de valeur 3 contient un Master Block qui pointe un ou plusieurs blocs index. Cette limitation de blocs index est due au maximum de 24 bits alloués au pointeur EOF (longueur d'un fichier). La grandeur d'un tel pointeur est : 128 * 256 * 512 = 16777226 = 16 mégabytes, où 128 représente le maximum de blocs index, 256 le maximum de blocs de données et 512 le nombre de bytes par bloc. Un Master Block ne se différencie pas visuellement d'un bloc index, sa structure interne reste la même.

— ***Index Block.*** C'est un bloc répertoire auxiliaire encore appelé *bloc index ;* il se compose d'un bloc de 512 bytes et pointe un maximum de 256 blocs de données. Ce type de bloc pointe toujours un bloc de données. Un fichier avec un Storage Type de valeur 2 contient un seul bloc index, tandis qu'avec un Storage Type 3, la capacité est de 128 blocs index.

LES TAMPONS ALLOUÉS À PRODOS

FCB, VCB et VBM

ProDOS sauvegarde en mémoire différentes données auxquelles les fichiers et les périphériques pourront se référer par la suite. Ce sont autant de renseignements placés dans des tables bien structurées, où chaque élément d'une donnée a une signification précise.

Structure et adresses des tables — Memory Layout pour un Apple *II* — 64 Ko

$F000-$F0FF	$100 (256) bytes pour la sauvegarde des variables des sous-programmes MLI.
$F100-$F1FF	$100 (256) bytes pour la sauvegarde du préfixe (PREFIX) et du chemin actuel (Pathname).
$F200-$F2FF	$100 (256) bytes pour la sauvegarde du Volume Control Block (VCB) : 8 entrées maximum, à raison de $20 (32) bytes par entrée.
$F300-$F3FF	$100 (256) bytes pour la sauvegarde du File Control Block (FCB) : 8 entrées maximum, à raison de $20 (32) bytes par entrée.
$F400-$F5FF	$200 (512) bytes pour la sauvegarde de la Volume Bit-Map (VBM) actuelle.
$F600-$F7FF	$200 (512) bytes pour la sauvegarde des Volume-Directory et Volume-Subdirectory actuels.

File Control Block — FCB

C'est un espace mémoire réservé pour les entrées des fichiers de données, ou de Directory ; il se trouve aux adresses $F300-$F3FF. Cette structure se trouve uniquement en mémoire vive (RAM) et n'a rien de commun avec les entrées d'un fichier de données ou de Directory d'une disquette.

Lorsqu'un fichier est ouvert par la commande OPEN, qu'il s'agisse d'un fichier de données, de Directory ou de Subdirectory, le système sauvegarde certaines caractéristiques de ce fichier à un emplacement mémoire appelé *File Control Block — FCB.* Divers renseignements sont alors disponibles pour ProDOS : le numéro du bloc d'entrée du Volume-Directory, le numéro du premier bloc de données, le pointeur à l'intérieur d'un fichier (Mark), le pointeur de la fin d'un fichier (EOF), le byte de référence du périphérique en ligne (Unit Number), etc.

Un fichier de données ProDOS est toujours créé par la commande MLI CREATE. Le FCB a une capacité maximale de 8 entrées, ce qui correspond à une zone mémoire ayant une possibilité de sauvegarde pour les caractéristiques de 8 fichiers.

Un maximum de 8 fichiers peuvent être ouverts simultanément sous ProDOS, d'où le nombre maximum de 8 entrées autorisées. Chaque entrée ampute la mémoire de 32 bytes, ce qui totalise 256 bytes pour l'espace mémoire alloué à FCB. Si l'on a une suite de plusieurs entrées dans la table FCB, elles s'échelonnent d'une manière graduelle et progressive, c'est-à-dire que la première entrée débute à l'adresse $F300, la deuxième entrée à l'adresse $F320, etc.

Entrée d'un File Control Block

Pos.	Adres.	Signification
$00	F300	File Reference Number. Un byte : utilisé comme référence interne du MLI, indique le numéro d'ordre dans les entrées de FCB.
$01	F301	Unit Number — DSSS0000. Un byte : renseigne le MLI sur le périphérique en ligne.
$02	F302	Key Block Number. Deux bytes qui indiquent le numéro du bloc — Key Block — du Directory où se trouve sauvegardé le nom d'un fichier nommé DATA1 par exemple.
$04	F304	Directory Block. Deux bytes qui indiquent le numéro du bloc où se trouve réellement sauvegardé l'entrée du fichier au nom de DATA1 cité comme exemple.
$06	F306	Entry Number. Un byte : DATA2 est la deuxième entrée à l'intérieur de ce même bloc — DATA2 étant un nom de fichier cité comme exemple.
$07	F307	Storage Type. Un byte : il caractérise le genre du fichier DATA ; ce n'est en fait qu'une partie de Storage Type, et plus précisément 4 bits qui sont utilisés. Le byte pouvant prendre les valeurs suivantes : $01, $02, $03, $0D, $0E ou $0F.
$08	F308	Statut byte. Un byte : il représente l'image binaire de plusieurs données (Index bloc modifié, bloc de données modifié, fichier modifié depuis que la commande OPEN a été déclarée, etc.).
$09	F309	Access byte. Un byte : il caractérise la façon dont un fichier sera abordé par la

		suite : écriture autorisée, lecture seule, etc.
$0A	F30A	Newline Character. Un **byte** : il est déterminé par la fonction **NEWLINE**, et désigne le dernier caractère à lire dans un fichier.
$0B	F30B	Buffer Index. Un byte : numéro d'ordre du tampon dont le pointeur se trouve sauvegardé dans la page globale de PRODOS.
$0C	F30C	Data Pointer. Deux bytes (LByte, HByte) : pointent le premier bloc des données du fichier DATA.
$0E	F30E	Index Block Number. Deux bytes (LByte, HByte) : cette paire de bytes est nulle si le fichier DATA ne possède pas de répertoire des blocs, fichier de moins de 512 bytes de longueur, sinon les deux bytes indiquent le nombre de blocs index.
$10	F310	First Block. Deux bytes : c'est le numéro du premier bloc qui se trouve en mémoire.
$12	F312	Pointeur MARK. Trois bytes (LByte, MByte, HByte) : pointent la position relative dans un fichier.
$15	F315	Pointeur EOF. Trois bytes (LByte, MByte, HByte) : désignent la longueur d'un fichier texte, qui correspond à la position absolue dans un fichier.
$18	F318	Blocks Used. Deux bytes (LByte, HByte) : désignent le nombre de blocs occupés par le fichier.
$1A	F31A	Un byte inutilisé.
$1B	F31B	Level. Un byte : utilisé par les commandes CLOSE et FLUSH : c'est la copie de l'adresse $BF94 de la page globale de PRODOS.
$1C	F31C	Flag Byte — Drapeau. Un byte : prend la valeur £$00, pour un fichier fermé et £$80 pour un fichier ouvert ou modifié.
$1D	F31D	Deux bytes inutilisés.
$1F	F31F	Newline Enable Mask. Un byte de masque, utilisé par NEWLINE ($C9).

Volume Control Block — VCB

Volume Control Block désigne les entrées des périphériques (drives) en ligne et se trouve aux adresses $F200-$F2FF. Comme FCB, cette structure existe uniquement dans la mémoire de l'Apple et n'a rien de commun avec les entrées des fichiers de données ou Directory de la disquette.

8 entrées au maximum sont disponibles, et chacune occupe 32 bytes, ce qui permet au système de mémoriser 8 lecteurs de disquettes. L'occupation de l'espace mémoire alloué au VCB est dégressive, c'est-à-dire que la première entrée se fera à l'adresse $F2E0, la seconde à l'adresse $F2C0, etc.

La raison d'être de VCB est due essentiellement à la notion de Volume des disquettes mises en place dans les drives en ligne. C'est ainsi que le nom du Volume d'une disquette, placé en en-tête d'un Volume-Directory, sera copié aux bytes relatifs $01-$0F (1-15) de l'entrée VCB, tandis que le numéro de bloc du Key Block sera copié en $16 (22).

Entrée d'un Volume Control Block — VCB

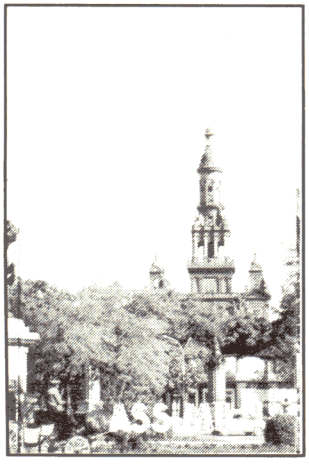
Pos.	Adres.	Signification
$00	F2E0	Name Length. Un byte : désigne la longueur du nom d'un fichier.
$01	F2E1	Name. Nom du Volume-Directory, avec un maximum de 15 caractères.
$10	F2F0	Unit Number — DSSS0000. Un byte qui renseigne sur le périphérique en ligne — Drive et slot.
$11	F2F1	Flag Byte — Drapeau. Un byte qui informe que le fichier est ouvert ou fermé : dans le premier cas le byte prendra comme valeur £$80 et dans le second cas £$00.
$12	F2F2	Total Blocks. Deux bytes qui désignent le nombre total des blocs alloués à la disquette ; c'est la capacité maximale de sauvegarde — $0118 = 280 blocs pour un disque 35 pistes standard.
$14	F2F4	Free Blocks. Deux bytes qui renseignent sur le nombre de blocs disponibles de la disquette.
$16	F2F6	Blocks Number Key-Blocks. Deux bytes qui renseignent sur la position du Key Block — Volume-Directory — ($0002 normalement).
$18	F2F8	Deux bytes inutilisés.
$1A	F2FA	VBM Block. Deux bytes (LByte, HByte) qui pointent le premier bloc de la Bit-Map — normalement bloc $0006.

(suite page 30)

$1C	F2FC	VBM Extent Number. Un byte qui renseigne sur le numéro — valeur relative du bloc de la Bit-Map : 00 = premier bloc de la VBM. VBM Extent est une configuration engendrée par un Volume qui nécessite plus d'un bloc pour sa Bit-Map.
$1D	F2FD	Un byte inutilisé.
$1E	F2FE	File Open. Un byte qui désigne le nombre de fichiers ouverts à cet instant.
$1F	F2FF	Un byte inutilisé.

BIBLIOGRAPHIE :

L'auteur signant ses articles sous la plume de Marcel COTTINI a écrit un certain nombre de livres traitant les sujets les plus divers sur l'informatique grand public, en particulier *LA PRATIQUE DE L'APPLE II*. Parmi ses ouvrages *SYSTEME PRODOS DE L'APPLE II* des éditions du PSI est principalement destiné à l'étude du système d'exploitation ProDOS. Il s'adresse à tous ceux qui désirent sortir des sentiers battus de la programmation rétrograde. Un livre à lire et à relire.



Le contrôle des connaissances sur Apple

ASSIMIL récidive, mais cette fois avec l'espagnol

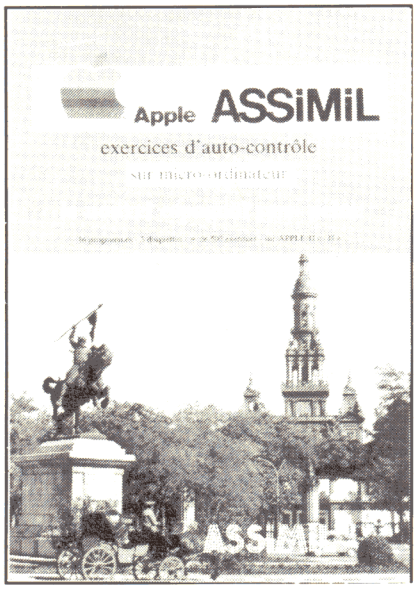
Qui ne connaît pas les fameuses méthodes ASSIMIL... et notamment *Le nouvel anglais sans peine ?* Qui n'a jamais eu entre les mains l'un des populaires petits bouquins du fameux éditeur international ? ASSIMIL est devenu synonyme — et pas seulement en France — d'*auto-apprentissage* des langues. Il faut avouer que les auteurs sont passés maîtres dans l'art — ô combien difficile — d'expliquer la prononciation des mots de chaque langue (cela va de l'anglais à l'hébreu, mais en passant par le chinois, l'arabe, le brésilien, le serbo-croate, l'espéranto... et j'en oublie, tant la liste est longue). Naturellement, les cassettes facilitent bien les choses, mais j'ai connu un jeune professeur qui se familiarisa avec le russe sans autre guide que le manuel.

ET L'APPLE ?

J'y viens : il vous aide tout simplement à contrôler vos connaissances grâce à 16 programmes, enregistrés sur 3 disquettes et vous proposant plus de 500 exercices. Aujourd'hui, ce contrôle concerne l'espagnol, mais la même méthode existe en anglais (nous en avions d'ailleurs parlé il y a de cela quelques mois).

Le but des auteurs est à la fois simple et ambitieux : vous aider à assimiler 87 points de grammaire, orthographe ou vocabulaire de l'espagnol courant, d'abord en vous présentant la règle correspondante, ensuite en vous proposant de résoudre les exercices d'application.

Le logiciel, réalisé par d'authentiques pédagogues, prévoit bien entendu les principaux cas d'erreur ce qui le rend d'autant plus sympathique.



Renseignements : 11, rue des Pyramides — 75001 PARIS — Tél. : (1) 42.60.40.66.

Pierre PRIVAT

BRICO

Fonctionne aussi sur l'APPLE IIGS

AMBITIEUX, ÉNORME, mais pratique pour un tas de calculs !

UNE rapide lecture des lignes 185 à 215 du listage ci-après vous donnera un avant-goût des nombreuses possibilités du *BRICO* (pensez "BRICOLAGE"), programme de Pierre PRIVAT (Castres, 81). N'hésitez pas à poursuivre votre examen sommaire, en vous attardant sur les lignes des divers menus de ce programme. Je vous l'accorde : on peut lui reprocher de pécher par excès de longueur. Vous reconnaîtrez pourtant qu'il était difficile de concevoir un programme à la fois simple et court, mais néanmoins, capable de présenter tous les avantages de celui-ci. Et puis, qui vous empêche de supprimer une certaine redondance d'instructions en utilisant les astuces habituelles ?

Notez au passage, que la disquette *TREMPLIN MICRO* vous évitera l'ankylose du poignet qui — je n'en doute pas — vous guette déjà !

```
100 : REM ************ BRICO, par Pierre PR
    IVAT, Oct 1986 ********                  
105 :                                        003A
110 TEXT : PRINT : PRINT  CHR$ (4);"BLOAD I
    MP.LM"                                   5A8B
115 PRINT : CLEAR : HOME : LOMEM: 28755      5E6B
120 M$ = "X"                                 99DD
125 ONERR  GOTO 2510                         0C18
130 D$ =  CHR$ (4):BIP$ =  CHR$ (7)          271C
135 PRINT D$;"PR£3": PRINT                   2D8D
140 M1$ = "(ESC) --> "Retour au menu généra
    l"                                       F079
145 M2$ = "L'unité utilisée est le: "        2BB6
150 M3$ = "ATTENTION: Utilisez le point à l
    a place de la virgule !"                 D603
155 PI = 3.1415926524                        C8D1
160 :                                        003A
165 REM ***  MENU GENERAL ***
170 :                                        003A
175 TA = 15:ST = 0                           C9AC
180 HOME : HTAB 27: PRINT "BRICO, "AIDE AU
    BRICOLAGE": GOSUB 2690                   420F
185 VTAB 5: HTAB 14: PRINT "1. --> "Calcule
    r un périmètre"                          2AE4
190 PRINT : HTAB 14: PRINT "2. --> "Calcule
    r une surface"                           DEFB
195 PRINT : HTAB 14: PRINT "3. --> "Calcule
    r un volume"                             1886
200 PRINT : HTAB 14: PRINT "4. --> "Calcule
    r des dosages"                           1BEE
205 PRINT : HTAB 14: PRINT "5. --> "Calcule
    r la quantité de revètement (sols/murs/
    plafond)"                                6696
210 PRINT : HTAB 14: PRINT "6. --> "Convert
    ir des unités"                           1369
215 PRINT : HTAB 14: PRINT "7. --> "Calcule
    r des consommations"                     99D5
220 PRINT : HTAB 14: PRINT "8. --> "Termine
    r"                                       D1EB
225 GOSUB 2435: IF REP < 1 OR REP > 8 THEN
    PRINT BIP$: GOTO 225                     6E05
230 ON REP GOTO 250,505,815,1150,1405,1985,
    2210,2730                                3642
235 :                                        003A
240 REM ***  PERIMETRES  ***
245 :                                        003A
250 T$ = "PERIMETRE": GOSUB 1920             0AEF
255 VTAB 2: HTAB 30: PRINT "CALCULER UN ";T
    $                                        B920
260 GOSUB 2690                               9F81
265 VTAB 5: HTAB 29: PRINT "1. --> "Carré"   6944
270 PRINT : HTAB 29: PRINT "2. --> "Rectang
    le"                                      061A
275 PRINT : HTAB 29: PRINT "3. --> "Paralle
    logramme"                                C511
280 PRINT : HTAB 29: PRINT "4. --> "Triangl
    e"                                       53DD
285 PRINT : HTAB 29: PRINT "5. --> "Trapèze
    "                                        E1DB
290 PRINT : HTAB 29: PRINT "6. --> "Cercle"  0C37
```

```
295 PRINT : HTAB 26: PRINT M1$                                  9088
300 GOSUB 2435: IF REP < 1 OR REP > 6 THEN
    PRINT BIP$: GOTO 300                                        3CFD
305 ON REP GOTO 325,355,355,385,430,475                         0ECD
310 :                                                           003A
315 REM ***  PER. CARRE  ***
320 :                                                           003A
325 HOME : HTAB 30: PRINT T$;" D'UN CARRE":
    GOSUB 2710                                                  2924
330 VTAB 10: HTAB 15: INPUT "Longueur d'un
    coté ? ";C: IF C < = 0 THEN  PRINT BIP
    $: GOTO 330                                                 8B4B
335 P = C * 4: GOSUB 2530: GOSUB 2470: GOTO
    325                                                         054B
340 :                                                           003A
345 REM ***  PER. RECTANGLE  ***
350 :                                                           003A
355 HOME : HTAB 16: PRINT T$;" D'UN RECTANG
    LE OU D'UN PARALLELOGRAMME": GOSUB 2710                     59B7
360 VTAB 8: HTAB 15: INPUT "Longueur ? ";LO
    : IF LO < = 0 THEN  PRINT BIP$: GOTO 3
    60                                                          4826
365 VTAB 10: HTAB 15: INPUT "Largeur ? ";LA
    : IF LA < = 0 THEN  PRINT BIP$: GOTO 3
    65                                                          6CD9
370 P = (LO + LA) * 2: GOSUB 2530: GOSUB 24
    70: GOTO 355                                                C64A
375 :                                                           003A
380 REM ***  PER. TRIANGLE  ***
385 :                                                           003A
390 HOME : HTAB 28: PRINT T$;" D'UN TRIANGL
    E": GOSUB 2710                                              6B14
395 VTAB 9: HTAB 15: INPUT "Longueur du 1er
     coté ? ";C1: IF C1 < = 0 THEN  PRINT
    BIP$: GOTO 395                                              C303
400 VTAB 11: HTAB 15: INPUT "Longueur du 2m
    e coté ? ";C2: IF C2 < = 0 THEN  PRINT
    BIP$: GOTO 400                                              971D
405 VTAB 13: HTAB 15: INPUT "Longueur du 3m
    e coté ? ";C3: IF C3 < = 0 THEN  PRINT
    BIP$: GOTO 405                                              8127
410 P = C1 + C2 + C3: GOSUB 2530: GOSUB 247
    0: GOTO 390                                                 0AFB
415 :                                                           003A
420 REM ***  PER. TRAPEZE  ***
425 :                                                           003A
430 HOME : HTAB 29: PRINT T$;" D'UN TRAPEZE
    ": GOSUB 2710                                               22DA
435 VTAB 8: HTAB 15: INPUT "Longueur 1er co
    té ? ";C1: IF C1 < = 0 THEN  PRINT BIP
    $: GOTO 435                                                 F544
440 VTAB 9: HTAB 15: INPUT "Longueur 2me co
    té ? ";C2: IF C2 < = 0 THEN  PRINT BIP
    $: GOTO 440                                                 D73F
445 VTAB 10: HTAB 15: INPUT "Longueur 3me c
    oté ? ";C3: IF C3 < = 0 THEN  PRINT BI
    P$: GOTO 445                                                E06F
450 VTAB 11: HTAB 15: INPUT "Longueur 4me c
    oté ? ";C4: IF C4 < = 0 THEN  PRINT BI
    P$: GOTO 450                                                016F
455 P = C1 + C2 + C3 + C4: GOSUB 2530: GOSU
    B 2470: GOTO 430                                            DE35
460 :                                                           003A
465 REM ***  PER. CERCLE  ***
470 :                                                           003A
475 HOME :T$ = "CIRCONFERENCE": HTAB 27: PR
    INT T$;" D'UN CERCLE": GOSUB 2710                           C9E7
480 VTAB 11: HTAB 15: INPUT "Diamètre du ce
    rcle ? ";D: IF D < = 0 THEN  PRINT BIP
    $: GOTO 480                                                 8E3E
485 P = D * PI: GOSUB 2530: GOSUB 2470: GOT
    O 475                                                       EBB7
490 :                                                           003A
495 REM ***  SURFACES  ***
500 :                                                           003A
505 T$ = "SURFACE": GOSUB 1920                                  514B
510 HTAB 30: PRINT "CALCULER UNE ";T$: GOSU
    B 2690                                                      9E12
515 VTAB 5: HTAB 29: PRINT "1. --> "Carré"                      6944
520 PRINT : HTAB 29: PRINT "2. --> "Rectang
    le"                                                         061A
525 PRINT : HTAB 29: PRINT "3. --> "Parallé
    logramme"                                                   C511
530 PRINT : HTAB 29: PRINT "4. --> "Triangl
    e"                                                          53DD
535 PRINT : HTAB 29: PRINT "5. --> "Trapèze
    "                                                           E1DB
540 PRINT : HTAB 29: PRINT "6. --> "Cercle"                     0C37
545 PRINT : HTAB 29: PRINT "7. --> "Sphère"                     CA89
550 PRINT : HTAB 26: PRINT M1$                                  9088
555 GOSUB 2435: IF REP < 1 OR REP > 7 THEN
    PRINT BIP$: GOTO 555                                        C70A
560 ON REP GOTO 580,610,645,680,715,755,775                     1CA0
565 :                                                           003A
570 REM ***   SURF. CARRE  ***
575 :                                                           003A
580 HOME : HTAB 29: PRINT T$;" D'UN CARRE":
    GOSUB 2710                                                  F02C
585 VTAB 10: HTAB 15: INPUT "Longueur d'un
    coté ? ";C: IF C < = 0 THEN  PRINT BIP
    $: GOTO 585                                                 A357
590 S = C * C: GOSUB 2570: GOSUB 2470: GOTO
    580                                                         7D64
595 :                                                           003A
600 REM ***  SURF. RECTANGLE  ***
605 :                                                           003A
610 HOME : HTAB 29: PRINT T$;" D'UN RECTANG
    LE": GOSUB 2710                                             F054
615 VTAB 8: HTAB 15: INPUT "Longueur ? ";LO
    : IF LO < = 0 THEN  PRINT BIP$: GOTO 6
```

```
    15                                                          7229
620 VTAB 10: HTAB 15: INPUT "Largeur ? ";LA
    : IF LA < = 0 THEN  PRINT BIP$: GOTO 6
    20                                                          65D3
625 S = LO * LA: GOSUB 2570: GOSUB 2470: GO
    TO 610                                                      1000
630 :                                                           003A
635 REM ***  SURF. PARALLELOGRAMME  ***
640 :                                                           003A
645 HOME : HTAB 26: PRINT T$;" D'UN PARALLE
    LOGRAMME": GOSUB 2710                                       2A11
650 VTAB 8: HTAB 15: INPUT "Longueur ? ";LO
    : IF LO < = 0 THEN  PRINT BIP$: GOTO 6
    50                                                          4028
655 VTAB 10: HTAB 15: INPUT "Hauteur ? ";HT
    : IF HT < = 0 THEN  PRINT BIP$: GOTO 6
    55                                                          8B05
660 S = LO * HT: GOSUB 2570: GOSUB 2470: GO
    TO 645                                                      BB17
665 :                                                           003A
670 REM ***  SURF. TRIANGLE  ***
675 :                                                           003A
680 HOME : HTAB 29: PRINT T$;" D'UN TRIANGL
    E": GOSUB 2710                                              EA15
685 VTAB 8: HTAB 15: INPUT "Base ? ";BS: IF
    BS < = 0 THEN  PRINT BIP$: GOTO 685                         D0CE
690 VTAB 10: HTAB 15: INPUT "Hauteur ? ";HT
    : IF HT < = 0 THEN  PRINT BIP$: GOTO 6
    90                                                          9D04
695 S = (BS * HT) / 2: GOSUB 2570: GOSUB 24
    70: GOTO 680                                                CD5E
700 :                                                           003A
705 REM ***  SURF. TRAPEZE  ***
710 :                                                           003A
715 HOME : HTAB 29: PRINT T$;" D'UN TRAPEZE
    ": GOSUB 2710                                               22DA
720 VTAB 8: HTAB 15: INPUT "Grande base ? "
    ;GB: IF GB < = 0 THEN  PRINT BIP$: GOT
    O 720                                                       F37D
725 VTAB 9: HTAB 15: INPUT "Petite base ? "
    ;PB: IF PB < = 0 THEN  PRINT BIP$: GOT
    O 725                                                       B9AF
730 VTAB 10: HTAB 15: INPUT "Hauteur ? ";HT
    : IF HT < = 0 THEN  PRINT BIP$: GOTO 7
    30                                                          8BFF
735 S = ((GB + PB) * HT) / 2: GOSUB 2570: G
    OSUB 2470: GOTO 715                                         EFFC
740 :                                                           003A
745 REM ***  SURF. CERCLE  ***
750 :                                                           003A
755 HOME : HTAB 29: PRINT T$;" D'UN CERCLE"
    : GOSUB 2710                                                A76D
760 VTAB 8: HTAB 15: INPUT "Diamètre ? ";D:
    IF D < = 0 THEN  PRINT BIP$: GOTO 760                       8B8E
765 S = PI * ((D / 2) ^ 2): GOSUB 2570: GOS
    UB 2470: GOTO 755                                           B75C
770 :                                                           003A
775 REM ***  SURF. SPHERE  ***
780 :                                                           003A
785 HOME : HTAB 28: PRINT T$;" D'UNE SPHERE
    ": GOSUB 2710                                               A6CA
790 VTAB 8: HTAB 15: INPUT "Diamètre ? ";D:
    IF D < = 0 THEN  PRINT BIP$: GOTO 790                       9591
795 S = (PI * ((D / 2) ^ 2)) * 4: GOSUB 257
    0: GOSUB 2470: GOTO 785                                     70AE
800 :                                                           003A
805 REM ***  VOLUMES  ***
810 :                                                           003A
815 T$ = "VOLUME": GOSUB 1920                                   741A
820 HTAB 29: PRINT "CALCULER UN ";T$: GOSUB
    2690                                                        57D5
825 VTAB 5: HTAB 20: PRINT "1. --> "Cube"                       42B7
830 PRINT : HTAB 20: PRINT "2. --> "Parallé
    lépipède rectangle"                                         061E
835 PRINT : HTAB 20: PRINT "3. --> "Prisme
    droit à base parallélogramme"                               B635
840 PRINT : HTAB 20: PRINT "4. --> "Prisme
    droit à base triangulaire"                                  DB32
845 PRINT : HTAB 20: PRINT "5. --> "Prisme
    droit à base trapézoidale"                                  1466
850 PRINT : HTAB 20: PRINT "6. --> "Cylindr
    e                                                           76B8
855 PRINT : HTAB 20: PRINT "7. --> "Sphère"                     9480
860 PRINT : HTAB 17: PRINT M1$                                  9688
865 GOSUB 2435: IF REP < 1 OR REP > 7 THEN
    PRINT BIP$: GOTO 865                                        EF0E
870 ON REP GOTO 890,920,960,1000,1040,1085,
    1120                                                        DC41
875 :                                                           003A
880 REM ***  VOL. CUBE  ***
885 :                                                           003A
890 HOME : HTAB 28: PRINT T$;" D'UN CUBE":
    GOSUB 2710                                                  0ADD
895 VTAB 10: HTAB 15: INPUT "Longueur d'un
    coté ? ";C: IF C < = 0 THEN  PRINT BIP
    $: GOTO 895                                                 9F5B
900 V = C ^ 3: GOSUB 2605: GOSUB 2470: GOTO
    890                                                         E45C
905 :                                                           003A
910 REM ***  VOL. PARALLELEPIPEDE RECTANGLE
      ***
915 :                                                           003A
920 HOME : HTAB 21: PRINT T$;" D'UN PARALLE
    LEPIPEDE RECTANGLE": GOSUB 2710                             D9B5
925 VTAB 8: HTAB 15: INPUT "Longueur ? ";LO
    : IF LO < = 0 THEN  PRINT BIP$: GOTO 9
    25                                                          362D
930 VTAB 9: HTAB 15: INPUT "Largeur ? ";LA:
    IF LO < = 0 THEN  PRINT BIP$: GOTO 930                      9FBD
935 VTAB 10: HTAB 15: INPUT "Hauteur ? ";HT
```

```
    : IF HT < = 0 THEN  PRINT BIP$: GOTO 9
    35                                          9506
940 V = (LO * LA) * HT: GOSUB 2605: GOSUB 2
    470: GOTO 920                               3FBD
945 :                                           003A
950 REM ***  VOL. PRISME DROIT BASE PARALLE
    LOG.  ***
955 :                                           003A
960 HOME : HTAB 16: PRINT T$;" D'UN PRISME
    DROIT A BASE PARALLELOGRAMME": GOSUB 27
    10                                          523E
965 VTAB 8: HTAB 15: INPUT "Longueur parall
    èlogramme ? ";LO: IF LO < = 0 THEN  PR
    INT BIP$: GOTO 965                          50DE
970 VTAB 9: HTAB 15: INPUT "Hauteur parallè
    logramme ? ";HP: IF HP < = 0 THEN  PRI
    NT BIP$: GOTO 970                           1E82
975 VTAB 10: HTAB 15: INPUT "Hauteur du pri
    sme ? ";HH: IF HH < = 0 THEN  PRINT BI
    P$: GOTO 975                                D09B
980 V = (LO * HP) * HH: GOSUB 2605: GOSUB 2
    470: GOTO 960                               A8C0
985 :                                           003A
990 REM ***  VOL. PRISME DROIT BASE TRIANG.
     ***
995 :                                           003A
1000 HOME : HTAB 18: PRINT T$;" D'UN PRISME
    DROIT A BASE TRIANGULAIRE": GOSUB 2710      7172
1005 VTAB 8: HTAB 15: INPUT "Base triangle ?
     ";BA: IF BA < = 0 THEN  PRINT BIP$: G
    OTO 1005                                    BA43
1010 VTAB 9: HTAB 15: INPUT "Hauteur triangl
    e ? ";HT: IF HT < = 0 THEN  PRINT BIP$
    : GOTO 1010                                 0475
1015 VTAB 10: HTAB 15: INPUT "Hauteur du pri
    sme ? ";HH: IF HH < = 0 THEN  PRINT BI
    P$: GOTO 1015                               AABD
1020 V = ((BA * HT) / 2) * HH: GOSUB 2605: G
    OSUB 2470: GOTO 1000                        8E1C
1025 :                                          003A
1030 REM ***  VOL. PRISME DROIT BASE TRAPEZ.
     ***
1035 :                                          003A
1040 HOME : HTAB 18: PRINT T$;" D'UN PRISME
    DROIT A BASE TRAPEZOIDALE": GOSUB 2710      876F
1045 VTAB 8: HTAB 15: INPUT "Grande base tra
    pèze ? ";GB: IF GB < = 0 THEN  PRINT B
    IP$: GOTO 1045                              3921
1050 VTAB 9: HTAB 15: INPUT "Petite base tra
    pèze ? ";PB: IF PB < = 0 THEN  PRINT B
    IP$: GOTO 1050                              CD4A
1055 VTAB 10: HTAB 15: INPUT "Hauteur trapèz
    e ? ";HT: IF HT < = 0 THEN  PRINT BIP$
    : GOTO 1055                                 E0A3
1060 VTAB 11: HTAB 15: INPUT "Hauteur du pri
    sme ? ";HH: IF HH < = 0 THEN  PRINT BI
    P$: GOTO 1060                               64BE
1065 V = (((GB + PB) * HT) / 2) * HH: GOSUB
    2605: GOSUB 2470: GOTO 1040                 5ED1
1070 :                                          003A
1075 REM ***  VOL. CYLINDRE  ***
1080 :                                          003A
1085 HOME : HTAB 30: PRINT T$;" D'UN CYLINDR
    E": GOSUB 2710                              5F11
1090 VTAB 8: HTAB 15: INPUT "Diamètre cylind
    re ? ";D: IF D < = 0 THEN  PRINT BIP$:
    GOTO 1090                                   ED35
1095 VTAB 10: HTAB 15: INPUT "Hauteur cylind
    re ? ";HC: IF HC < = 0 THEN  PRINT BIP
    $: GOTO 1095                                CD8C
1100 V = (((D / 2) ^ 2) * PI) * HH: GOSUB 26
    05: GOSUB 2470: GOTO 1085                   7836
1105 :                                          003A
1110 REM ***  VOL. SPHERE  ***
1115 :                                          003A
1120 HOME : HTAB 30: PRINT T$;" D'UNE SPHERE
    ": GOSUB 2710                               36C3
1125 VTAB 10: HTAB 15: INPUT "Diamètre sphèr
    e ? ";D: IF D < = 0 THEN  PRINT BIP$:
    GOTO 1125                                   4002
1130 V = ((D ^ 3) * PI) / 6: GOSUB 2605: GOS
    UB 2470: GOTO 1120                          8286
1135 :                                          003A
1140 REM ***  DILUTIONS/DOSAGES  ***
1145 :                                          003A
1150 HOME : PRINT : HTAB 23: PRINT "CALCULER
     DES DOSAGES ET DILUTIONS": GOSUB 2690      45B4
1155 TA = 10                                    FEC6
1160 VTAB 7: HTAB 20: PRINT "1. --> ¯1 + X"     DB8E
1165 PRINT : PRINT : HTAB 20: PRINT "2. -->
    ¯X %"                                       C50D
1170 PRINT : PRINT : HTAB 20: PRINT "3. -->
    ¯X produit / Y produit"                     391F
1175 PRINT : PRINT : HTAB 17: PRINT M1$         DE7C
1180 GOSUB 2435: IF REP < 1 OR REP > 3 THEN
    PRINT BIP$: GOTO 1180                       FF31
1185 ON REP GOTO 1205,1270,1335                 44FC
1190 :                                          003A
1195 REM ***  DOSAGE 1 + X  ***
1200 :                                          003A
1205 HOME : HTAB 28: PRINT "DILUTIONS DE FOR
    ME 1 + X"                                   771A
1210 HTAB 15: PRINT "1 partie de concentré p
    our X parties de diluant.": GOSUB 2690      20C3
1215 VTAB 5: HTAB 10: INPUT "Unité utilisée
    ? ";U$                                      FAB2
1220 VTAB 7: HTAB 10: PRINT M3$                 D1A2
1225 VTAB 10: HTAB 10: INPUT "Valeur de X ?
    ";X: IF X < = 0 THEN  PRINT BIP$: GOTO
    1225                                        4574
```

```
1230 VTAB 12: HTAB 10: INPUT "Quantité total
     e à obtenir ? ";Q: IF Q < = 0 THEN  PR
     INT BIP$: GOTO 1230                    7291
1235 Z = X + 1:QC = Q / Z:QC =  INT (QC * 10
     0) / 100:QD = Q - QC                   F5E9
1240 VTAB 15: HTAB 10: PRINT "Il vous faudra
      donc:"                                C7C4
1245 PRINT : HTAB 13: PRINT "- ";QC;" ";U$;"
      de concentré, et"                     3399
1250 HTAB 13: PRINT "- ";QD;" ";U$;" de dilu
     ant.": GOSUB 2470: GOTO 1150           1D8C
1255 :                                      003A
1260 REM ***  DOSAGE X %  ***
1265 :                                      003A
1270 HOME : HTAB 30: PRINT "DOSAGES DE FORME
      X %"                                  0807
1275 HTAB 16: PRINT "Diluer X % d'un produit
      dans une quantité donnée": GOSUB 2690 2305
1280 VTAB 5: HTAB 10: INPUT "Unité utilisée
     ? ";U$                                 FAB2
1285 PRINT : HTAB 10: PRINT M3$             A083
1290 VTAB 10: HTAB 10: INPUT "Valeur de X ?
     ";X: IF X < = 0 THEN  PRINT BIP$: GOTO
     1290                                   3976
1295 VTAB 12: HTAB 10: INPUT "Quantité à laq
     uelle on ajoute le produit ? ";Q: IF Q
     < = 0 THEN  PRINT BIP$: GOTO 1295      7192
1300 D = (Q / 100) * X:D =  INT (D * 1000) /
     1000                                   9701
1305 VTAB 15: HTAB 10: PRINT "Il vous faudra
      donc ajouter:"                        D2FE
1310 PRINT : HTAB 13: PRINT "- ";D;" ";U$;"
     de produit, et"                        FC99
1315 PRINT : HTAB 10: PRINT "Vous aurez au t
     otal: ";Q + D;" ";U$;" de solution.": G
     OSUB 2470: GOTO 1150                   100C
1320 :                                      003A
1325 REM ***  DOSAGE X / Y  ***
1330 :                                      003A
1335 HOME : HTAB 20: PRINT "DOSAGES DE FORME
      X UNITES POUR Y UNITES"               0590
1340 HTAB 10: PRINT "Ajouter X unités d'un p
     roduit par Y unités d'un autre produit"
     : GOSUB 2690                           61DF
1345 VTAB 5: HTAB 10: INPUT "Unité utilisée
     ? ";U$                                 FAB2
1350 PRINT : HTAB 10: PRINT M3$             A083
1355 VTAB 9: HTAB 10: INPUT "Valeur de X ? "
     ;X: IF X < = 0 THEN  PRINT BIP$: GOTO
     1355                                   E450
1360 VTAB 11: HTAB 10: INPUT "Valeur de Y ?
     ";Y: IF Y < = 0 THEN  PRINT BIP$: GOTO
     1360                                   0C78
1365 VTAB 13: HTAB 10: INPUT "Quantité total
     e de produit à obtenir ? ";Q: IF Q < =
     0 THEN  PRINT BIP$: GOTO 1365          A08B
1370 QX = (Q / (X + Y)) * X:QX =  INT (QX *
     100) / 100:QY = Q - QX                 AE80
1375 PRINT : HTAB 10: PRINT "Il vous faudra
     donc: "                                9896
1380 PRINT : HTAB 13: PRINT "- ";QX;" ";U$;"
      de produit X, et"                     CC76
1385 HTAB 13: PRINT "- ";QY;" ";U$;" de prod
     uit Y.": GOSUB 2470: GOTO 1150         2D30
1390 :                                      003A
1395 REM ***  ROULEAUX REVETEMENT  ***
1400 :                                      003A
1405 HOME : PRINT : PRINT : HTAB 23: PRINT "
     CALCUL DE LA QUANTITE DE REVETEMENT": G
     OSUB 2690                              3F04
1410 TA = 1                                 9F96
1415 VTAB 9: HTAB 20: PRINT "1. --> Murs (P
     apiers peints)"                        DF5B
1420 PRINT : PRINT : HTAB 20: PRINT "2. -->
      Plafond ou sol (rouleaux)"            2A2C
1425 PRINT : PRINT : HTAB 20: PRINT "3. -->
      Sols et murs (carrelages)"            7E1C
1430 PRINT : PRINT : HTAB 17: PRINT M1$     DE7C
1435 GOSUB 2435: IF REP < 1 OR REP > 3 THEN
     PRINT BIP$: GOTO 1435                  0334
1440 ON REP GOTO 1445,1645,1735             560C
1445 T$ = "MURS (Papier peint)"             81A5
1450 HOME : HTAB 29: PRINT T$: HTAB 12: PRIN
     T M3$: GOSUB 2690                      D0C4
1455 VTAB 5: INPUT "Hauteur des murs (en m)
     ? ";H: IF H < = 0 THEN  PRINT BIP$: GO
     TO 1455                                77FE
1460 HM = H * 100                           7208
1465 VTAB 7: INPUT "Périmètre de la pièce (e
     n m) ? ";P: IF P < = 0 THEN  PRINT BIP
     $: GOTO 1465                           99BF
1470 VTAB 9: INPUT "Total des largeurs des o
     uvertures (en m) ? ";O: IF O < 0 OR O >
     = P THEN  PRINT BIP$: GOTO 1470        8D0D
1475 PP = (P - O) * 100                     F084
1480 VTAB 12: INPUT "Longueur d'un rouleau (
     en m) ? ";L: IF L < = 0 THEN  PRINT BI
     P$: GOTO 1480                          618D
1485 LO = L * 100                           6712
1490 VTAB 15: INPUT "Largeur d'un rouleau (e
     n cm) ? ";LA: IF LA < = 0 THEN  PRINT
     BIP$: GOTO 1490                        87F7
1495 VTAB 18: INPUT "Distance entre motifs i
     dentiques (raccord) (en cm) ? ";MT: IF
     MT < 0 THEN  PRINT BIP$: GOTO 1495     6768
1500 LE = HM + 5 + MT: REM *** Longueur d'un
      lé ***                                575C
1505 NR =  INT (LO / LE): REM *** Nbre de lé
     s/rouleau ***                          658B
1510 CH =  INT (LO - (NR * LE)): REM *** Chu
```

```
     te/rouleau ***                                          1F2F
1515 VTAB 21: HTAB 29: PRINT "Est-ce correct
     (O/N) ? ";: GET R$                                      60B0
1520 IF R$ = "N" OR R$ = "n" THEN 1445                       94BD
1525 IF R$ = "O" OR R$ = "o" THEN 1535                       F0BF
1530 PRINT BIP$: GOTO 1515                                   0D6A
1535 IF LO = LE - 5 THEN  HOME : PRINT : VTA
     B 10: HTAB 24: PRINT "ATTENTION: Travai
     l de précision !":NR = 1:CH = 0:LE = LO
     : FOR I = 1 TO 20: PRINT BIP$;: NEXT :
     GOTO 1545                                               1B4C
1540 IF NR =  < 0 THEN  PRINT BIP$: HOME : V
     TAB 10: HTAB 20: PRINT "TRAVAIL IMPOSSI
     BLE: Papier trop court ! ": GOTO 1625                   CE7E
1545 HOME : HTAB 29: PRINT T$: GOSUB 2690                    9BF9
1550 PRINT : PRINT "Par rouleau, vous aurez
     donc:"                                                  C7D0
1555 PRINT : HTAB 13: PRINT "- ";NR;" lé";:
     IF NR > 1 THEN  PRINT "s, et"                           5C70
1560 IF LO = HM OR CH = 0 THEN  PRINT : HTAB
     13: PRINT "- aucune chute": GOTO 1570                   9149
1565 PRINT : HTAB 13: PRINT "- ";CH;" cm de
     chute"                                                  9CAA
1570 NL =  INT (PP / LA): IF (PP / LA) > NL
     THEN NL = NL + 1: REM *** Nbre de lés n
     écessaire ***                                           13E0
1575 RO =  INT (NL / NR):RE = (NL / NR) - RO
     : REM *** Nbre rlx nécessaire et reste
     ***                                                     51FB
1580 FL = 0: REM *** Mise à 0 du flag de res
     te ***                                                  C392
1585 IF RE THEN RO = RO + 1:FL = 1: REM ***
     Flag si reste non nul ***                               F1E0
1590 RL = (RO * NR) - NL:RP = ((RL * LE) + C
     H) / 100: REM *** Reste papier (Lés+Chu
     tes) ***                                                4023
1595 PRINT : PRINT "Il vous faudra donc ";N
     L;" lés, soit:"                                         0C9E
1600 PRINT : HTAB 13: PRINT "- ";RO;" roulea
     u";: IF RO > 1 THEN  PRINT "x"                          B7E8
1605 IF CH = 0 AND FL = 0 THEN  PRINT : PRIN
     T "Et vous n'aurez pas de reste": GOTO
     1625                                                    9B72
1610 PRINT : PRINT : PRINT "Et il vous reste
     ra:"                                                    4B11
1615 IF CH > 0 AND (RO - FL) > 0 THEN  PRINT
     : HTAB 13: PRINT "- ";RO - FL;" chute(s
     ) de ";CH;" cm": PRINT                                  BA2C
1620 IF FL THEN  HTAB 13: PRINT "- une longu
     eur de ";RP;" m"                                        D38D
1625 GOSUB 2470: GOTO 1405                                   A12C
1630 :                                                       003A
1635 REM *** PLAFOND/SOL  ***
1640 :                                                       003A
1645 HOME : HTAB 32: PRINT "PLAFONDS et SOLS
     ": GOSUB 2690                                           0930
1650 VTAB 8: HTAB 20: PRINT "1. --> Pose en
     longueur": VTAB 12: HTAB 20: PRINT "2.
     --> Pose en largeur"                                    CCF3
1655 GOSUB 2435: IF REP < 1 OR REP > 2 THEN
     PRINT BIP$: GOTO 1655                                   6537
1660 ON REP GOTO 1665,1670                                   7112
1665 L$ = " Pose en longueur": GOTO 1675                     65D7
1670 L$ = " Pose en largeur"                                 20C0
1675 T$ = "PLAFONDS ET SOLS"                                 11FD
1680 HOME : HTAB 22: PRINT T$;L$: GOSUB 2690                 BA9D
1685 VTAB 5: INPUT "Longueur de pièce (en m)
     ? ";LO: IF LO <  = 0 THEN  PRINT BIP$:
     GOTO 1685                                               DA00
1690 LO = LO * 100                                           5061
1695 VTAB 7: INPUT "Largeur de la pièce (en
     m) ? ";LA: IF LA <  = 0 THEN  PRINT BIP
     $: GOTO 1695                                            CB35
1700 LA = LA * 100                                           BD45
1705 IF REP = 1 THEN HM = LO:PP = LA                         7790
1710 IF REP = 2 THEN HM = LA:PP = LO                         6391
1715 O = 0: GOTO 1480                                        4501
1720 :                                                       003A
1725 REM ***  CARRELAGE  ***
1730 :                                                       003A
1735 HOME : HTAB 28: PRINT "CARRELAGES SOLS
     ET MURS": HTAB 12: PRINT M3$: GOSUB 269
     0                                                       F6E9
1740 VTAB 5: PRINT "CARREAUX:"                               ECA4
1745 VTAB 6: INPUT "Longueur (cm): ";LC: IF
     LC <  = 0 THEN 1745                                     5C4C
1750 VTAB 7: INPUT "Largeur (cm): ";WC: IF W
     C <  = 0 THEN 1750                                      5C00
1755 VTAB 8: INPUT "Largeur des joints (mm):
      ";J:J = J / 10: IF J < 0 THEN 1755                     B35D
1760 SC = (LC + J) * (WC + J):SC =  INT (SC
     * 100) / 100                                            6B30
1765 VTAB 10: PRINT "Est-ce correct (O/N) ?
     ";: GET R$                                              7073
1770 IF R$ = "N" OR R$ = "n" THEN 1735                       69BF
1775 IF R$ = "O" OR R$ = "o" THEN 1785                       E6C6
1780 PRINT BIP$: GOTO 1765                                   0771
1785 PRINT : VTAB 13: HTAB 1: CALL  - 958: V
     TAB 12: HTAB 1: PRINT "SURFACE A CARREL
     ER:"                                                    57BC
1790 VTAB 14: INPUT "Longueur (m): ";LO: IF
     LO <  = 0 THEN 1790                                     0350
1795 VTAB 15: INPUT "Largeur ou hauteur (m):
      ";LA: IF LA <  = 0 THEN 1795                           52DD
1800 VTAB 20: HTAB 1: CALL  - 868: VTAB 20:
     HTAB 1: PRINT "Longueur/largeur correct
     es ? (O/N) ";: GET R$                                   8992
1805 IF R$ = "N" OR R$ = "n" THEN 1785                       5FC4
1810 IF R$ = "O" OR R$ = "o" THEN 1820                       FEBC
1815 PRINT BIP$: GOTO 1800                                   3967
```

```
1820 S = LO * LA:S =  INT (S * 100) / 100:ST
     = ST + S                                   6B13
1825 VTAB 12: HTAB 21: CALL  - 868: PRINT ST
     ;" m2"                                     FC26
1830 VTAB 20: HTAB 1: CALL  - 868: VTAB 20:
     HTAB 1: PRINT "Voulez-vous ajouter une
     surface (O/N) ? ";: GET R$                 3AE0
1835 IF R$ = "N" OR R$ = "n" THEN 1845          39C1
1840 IF R$ = "O" OR R$ = "o" THEN 1785          E6C6
1845 PRINT : VTAB 20: HTAB 1: CALL  - 868: V
     TAB 20: HTAB 1: PRINT "Tout est-il corr
     ect ? (O/N) ";: GET R$                     3C56
1850 IF R$ = "N" OR R$ = "n" THEN ST = 0: GO
     TO 1735                                    C04B
1855 IF R$ = "O" OR R$ = "o" THEN 1865          CCC5
1860 PRINT BIP$: GOTO 1845                      6770
1865 ST = ST * 10000:NC =  INT (ST / SC): IF
     (ST / SC) ) NC THEN NC = NC + 1            42EF
1870 HOME : HTAB 28: PRINT "CARRELAGES SOLS
     ET MURS": GOSUB 2690                       DC1E
1875 VTAB 4: PRINT "Vous devez carreler ";ST
     / 10000;" m2"                              AD45
1880 PRINT : PRINT : PRINT "Il vous faudra a
     u moins ";NC;" carreau";: IF NC ) 1 THE
     N PRINT "x"                                6372
1885 PRINT : PRINT : PRINT "de dimensions: "
     ;LC;" x ";WC;" cm"                         E9C7
1890 PRINT : PRINT : PRINT "avec des joints
     de: ";J * 10;" mm."                        CE12
1895 PRINT : PRINT : PRINT "Mais je vous con
     seille d'acheter quelques carreaux de p
     lus pour pallier les": PRINT "éventuels
      accidents de découpe !"                   C93D
1900 GOSUB 2470:ST = 0: GOTO 1735               C613
1905 :                                          003A
1910 REM ***  SAISIE UNITES  ***
1915 :                                          003A
1920 DIM U$(4):U$(1) = "mm":U$(2) = "cm":U$(
     3) = "m":U$(4) = "km"                      50BD
1925 HOME : HTAB 32: PRINT "CHOIX DE L'UNITE
     ": GOSUB 2690                              FFFB
1930 VTAB 5: HTAB 23: PRINT "Quelle unité vo
     ulez-vous utiliser ?"                      5456
1935 PRINT : PRINT : HTAB 31: PRINT "1 "--)
     "millimètre"                               DB8F
1940 PRINT : HTAB 31: PRINT "2 "--) "centimè
     tre"                                       3398
1945 PRINT : HTAB 31: PRINT "3 "--) "mètre"     D426
1950 PRINT : HTAB 31: PRINT "4 "--) "kilomèt
     re"                                        0F56
1955 GOSUB 2435                                 8D7E
1960 :U =  VAL (R$): IF U ( 1 OR U ) 4 THEN
     PRINT BIP$: GOTO 1955                      5FA5
1965 U$ = U$(U): HOME : RETURN                  0624
1970 :                                          003A
1975 REM ***  CONVERSION UNITES  ***
1980 :                                          003A
1985 DIM U$(2,8)                                C2E6
1990 U$(1,1) = "mm":U$(1,2) = "cm":U$(1,3) =
     "m ":U$(1,4) = "km":U$(1,5) = "l ":U$(1
     ,6) = "g ":U$(1,7) = "kg":U$(1,8) = ""C
     "                                          B644
1995 U$(2,1) = "in":U$(2,2) = "ft":U$(2,3) =
     "yd":U$(2,4) = "M ":U$(2,5) = "pt":U$(2
     ,6) = "oz":U$(2,7) = "lb":U$(2,8) = ""F
     "                                          E7D1
2000 HOME : HTAB 31: PRINT "CONVERSION D'UNI
     TES": GOSUB 2690                           0F27
2005 VTAB 5: HTAB 20: PRINT "1. "Système mét
     rique --) Système impérial"                39E7
2010 VTAB 10: HTAB 20: PRINT "2. "Système im
     périal --) Système métrique"               4D14
2015 VTAB 15: HTAB 18: PRINT M1$                B8D7
2020 GOSUB 2435: IF REP ( 1 OR REP ) 2 THEN
     PRINT BIP$: GOTO 2020                      AA2A
2025 ON REP GOTO 2030,2050                      30FE
2030 U1 = 1:U2 = 2: HOME : HTAB 24: PRINT "C
     ONVERSION Métrique --) Impérial": GOSUB
     2690                                       E86D
2035 VTAB 4: FOR I = 1 TO 8: HTAB 31: PRINT
     I;". :""";U$(1,I);" "--) "";U$(2,I): PR
     INT : NEXT                                 5C2A
2040 GOSUB 2435: IF REP ( 1 OR REP ) 8 THEN
     PRINT BIP$: GOTO 2040                      F032
2045 GOTO 2070                                  5B74
2050 U1 = 2:U2 = 1: HOME : HTAB 24: PRINT "C
     ONVERSION Impérial --) Métrique": GOSUB
     2690                                       B86D
2055 VTAB 4: FOR I = 1 TO 8: HTAB 31: PRINT
     I;". :""";U$(2,I);" "--) "";U$(1,I): PR
     INT : NEXT                                 C52A
2060 GOSUB 2435: IF REP ( 1 OR REP ) 8 THEN
     PRINT BIP$: GOTO 2060                      FC34
2065 GOTO 2070                                  5B74
2070 HOME : HTAB 29: PRINT "CONVERSION DE ";
     U$(U1,REP);" EN ";U$(U2,REP): GOSUB 269
     0                                          7223
2075 TA = 1: VTAB 6: PRINT M3$                  6D40
2080 VTAB 8: INPUT "Valeur à convertir ? ";V
     1: IF REP (  ) 8 AND V1 (  = 0 THEN  PR
     INT BIP$: GOTO 2080                        AA1F
2085 IF U1 = 1 THEN  ON REP GOSUB 2115,2120,
     2125,2130,2135,2140,2145,2150: GOTO 210
     0                                          A163
2090 VTAB 12: PRINT V1;" ";U$(U1,REP);" ""=
     """;V2;" ";U$(U2,REP)                      663A
2095 IF U1 = 2 THEN  ON REP GOSUB 2155,2160,
     2165,2170,2175,2180,2185,2190: GOTO 210
     0                                          AB84
2100 V2 =  INT (V2 * 1000) / 1000               A71B
```

```
2105 VTAB 12: PRINT V1;" ";U$(U1,REP);" "=
     "";: IF REP < > 8 AND V2 < 1 THEN PR
     INT "0";                                  EF9E
2110 PRINT V2;" ";U$(U2,REP): GOSUB 2470: GO
     TO 2000                                   1FDE
2115 V2 = V1 * 0.03937: RETURN : REM *** Cal
     culs conversion ***                       4EF8
2120 V2 = V1 * 0.0328: RETURN                  5DBF
2125 V2 = V1 * 1.094: RETURN                   7B90
2130 V2 = V1 * 0.62137: RETURN                 7CF5
2135 V2 = V1 * 1.75: RETURN                    055F
2140 V2 = V1 * 0.0352733: RETURN               5359
2145 V2 = V1 * 2.2046: RETURN                  38C0
2150 V2 = ((V1 * 9) / 5) + 32: RETURN          279C
2155 V2 = V1 * 25.3995: RETURN                 7203
2160 V2 = V1 * 30.479: RETURN                  27C9
2165 V2 = V1 * 0.9144: RETURN                  3FC4
2170 V2 = V1 * 1.6093: RETURN                  26C5
2175 V2 = V1 * 0.5679: RETURN                  38CD
2180 V2 = V1 * 28.35: RETURN                   BD94
2185 V2 = V1 * 0.373: RETURN                   6D8F
2190 V2 = ((V1 - 32) * 5) / 9: RETURN          1C9D
2195 :                                         003A
2200 REM ***  CONSOMMATIONS  ***
2205 :                                         003A
2210 TA = 1: HOME : HTAB 27: PRINT "CALCULER
      DES CONSOMMATIONS": GOSUB 2690           61E4
2215 VTAB 7: HTAB 13: PRINT "1. --> "Calcule
     r une proportion à partir de quantités
     données": PRINT                           3890
2220 HTAB 13: PRINT "2. --> "Calculer une qu
     antité à partir d'une proportion donnée
     ": PRINT                                  D5A2
2225 HTAB 13: PRINT "3. --> "Calculer une co
     nsommation en l/100 km": PRINT            8520
2230 HTAB 13: PRINT "4. --> "Calculer le cou
     t d'un trajet": PRINT                     DFEF
2235 HTAB 11: PRINT M1$                        AC8E
2240 GOSUB 2435: IF REP < 1 OR REP > 4 THEN
     PRINT BIP$: GOTO 2240                     DD30
2245 ON REP GOTO 2250,2295,2345,2380           FC00
2250 HOME : HTAB 29: PRINT "CALCUL D'UNE PRO
     PORTION": GOSUB 2690                      3C28
2255 HTAB 12: PRINT M3$                        1791
2260 VTAB 6: INPUT "Unité 1 ? ";U1$            964A
2265 VTAB 8: INPUT "Valeur pour l'unité 1 ?
     ";V1: IF V1 < = 0 THEN PRINT BIP$: GO
     TO 2265                                   0561
2270 VTAB 10: INPUT "Unité 2 ? ";U2$           3577
2275 VTAB 12: INPUT "Valeur pour l'unité 2 ?
      ";V2: IF V2 < = 0 THEN PRINT BIP$: G
     OTO 2275                                  0B90
2280 V3 = V1 / V2:V3 = INT (V3 * 100) / 100    C92A
2285 VTAB 15: PRINT "La proportion est donc:
      ";V3;" ";U1$;"/";U2$                     518F
2290 GOSUB 2470: GOTO 2210                     CF27
2295 HOME : HTAB 29: PRINT "CALCUL D'UNE QUA
     NTITE": GOSUB 2690                        B477
2300 HTAB 12: PRINT M3$                        1791
2305 VTAB 6: INPUT "Unité 1 ? ";U1$            964A
2310 VTAB 8: INPUT "Proportion d'unité 1 ? "
     ;V1: IF V1 < = 0 THEN PRINT BIP$: GOT
     O 2310                                    BC37
2315 VTAB 10: INPUT "Unité 2 ? ";U2$           3577
2320 VTAB 12: INPUT "Proportion d'unité 2 ?
     ";V2: IF V2 < = 0 THEN PRINT BIP$: GO
     TO 2320                                   1966
2325 VTAB 14: INPUT "Quantité totale à obten
     ir ? ";V3: IF V3 < = 0 THEN PRINT BIP
     $: GOTO 2325                              54D8
2330 V4 = (V1 / V2) * V3:V4 = INT (V4 * 100
     ) / 100                                   BAD1
2335 PRINT : PRINT "Il vous faudra donc: ";V
     4;" ";U1$;" pour ";V3;" ";U2$             FBFC
2340 GOSUB 2470: GOTO 2210                     CF27
2345 HOME : HTAB 28: PRINT "CALCUL EN litres
     /100 Km": GOSUB 2690                      1C96
2350 HTAB 12: PRINT M3$                        1791
2355 VTAB 8: INPUT "Nombre de kimomètres par
     courus ? ";V1: IF V1 < = 0 THEN PRINT
     BIP$: GOTO 2355                           2235
2360 VTAB 10: INPUT "Avec combien de litres
     de carburant ? ";V2: IF V2 < = 0 THEN
     PRINT BIP$: GOTO 2360                     49F7
2365 V3 = (V2 / V1) * 100:V3 = INT (V3 * 10
     0) / 100                                  09D6
2370 VTAB 15: PRINT "La consommation est :"
     ;V3;" l/100km"                            F311
2375 GOSUB 2470: GOTO 2210                     CF27
2380 HOME : HTAB 26: PRINT "CALCULER LE COUT
      D'UN TRAJET": GOSUB 2690                 CF31
2385 HTAB 12: PRINT M3$                        1791
2390 VTAB 6: INPUT "Consommation en l/100 km
      ? ";V1: IF V1 < = 0 THEN PRINT BIP$:
     GOTO 2390                                 B1B8
2395 VTAB 8: INPUT "Prix du litre de carbura
     nt ? ";V2: IF V2 < = 0 THEN PRINT BIP
     $: GOTO 2395                              629A
2400 VTAB 10: INPUT "Nombre de kilomètres pa
     rcourus ? ";V3: IF V3 < = 0 THEN PRIN
     T BIP$: GOTO 2400                         4E58
2405 V4 = ((V1 / 100) * V3) * V2:V4 = INT (
     V4 * 100) / 100                           B47D
2410 VTAB 15: PRINT "Le trajet revient à :"
     ;V4;" F"                                  497C
2415 GOSUB 2470: GOTO 2210                     CF27
2420 :                                         003A
2425 REM ***  SAISIE N° DE CHOIX  ***
2430 :                                         003A
2435 VTAB 21: FOR I = 1 TO 80: PRINT "_";: N
```

```
     EXT : PRINT                                                    30B5
2440 VTAB 23: HTAB 30: PRINT "Indiquez votre
      choix ¯";: GET R$:REP =  VAL (R$)                             AAAC
2445 IF R$ =  CHR$ (27) OR R$ =  CHR$ (3) TH
     EN POP : GOTO 115                                              50F4
2450 RETURN                                                         63B1
2455 :                                                              003A
2460 REM ***  SAISIE REP CLAVIER  ***
2465 :                                                              003A
2470 VTAB 21: FOR I = 1 TO 80: PRINT "_";: N
     EXT : PRINT                                                    30B5
2475 VTAB 22: HTAB 25: PRINT " Imprimer les
     résultats: (I) ¯"                                              6478
2480 VTAB 23: PRINT "Appuyez sur une touche
     pour recommencer, sur (ESC) pour retour
      au menu général ";: GET R$: IF R$ =  C
     HR$ (27) OR R$ =  CHR$ (3) THEN  POP :
     HOME : GOTO 115                                                96AC
2485 IF R$ = "I" OR R$ = "i" THEN  GOTO 2645                        F461
2490 RETURN                                                         63B1
2495 :                                                              003A
2500 REM *** TRAIT. ERR. ?REENTER  ***
2505 :                                                              003A
2510 PRINT BIP$: VTAB ( PEEK (37) - 1): HTAB
     TA: RESUME                                                     A8C2
2515 :                                                              003A
2520 REM ***  AFF. PERIMETRES  ***
2525 :                                                              003A
2530 IF U = 1 AND P >  = 1000 THEN U$ = U$(3
     ):P = P / 1000                                                 8DC0
2535 IF U = 1 AND P >  = 10 THEN U$ = U$(2):
     P = P / 10                                                     C6FF
2540 IF U = 2 AND P >  = 100 THEN U$ = U$(3)
     :P = P / 100                                                   1661
2545 IF U = 3 AND P >  = 1000 THEN U$ = U$(4
     ):P = P / 1000                                                 5BC3
2550 P =  INT (P * 100) / 100: VTAB 15: HTAB
     15: PRINT T$;" = ";P;" ";U$:U$ = U$(U):
     RETURN                                                         A596
2555 :                                                              003A
2560 REM ***  AFF. SURFACES  ***
2565 :                                                              003A
2570 IF U = 1 AND S >  = 100 THEN U$ = U$(2)
     :S = S / 100                                                   4568
2575 IF U = 2 AND S >  = 10000 THEN U$ = U$(
     3):S = S / 10000                                               D32A
2580 IF U = 3 AND S >  = 1000000 THEN U$ = U
     $(4):S = S / 1000000                                           CFEC
2585 S =  INT (S * 100) / 100: VTAB 15: HTAB
     15: PRINT T$;" = ";S;" ";U$;"2":U$ = U$
     (U): RETURN                                                    D550
2590 :                                                              003A
2595 REM ***  AFF. VOLUMES  ***
2600 :                                                              003A
2605 IF U = 1 AND V >  = 1000 THEN U$ = U$(2
     ):V = V / 1000                                                 1DD1
2610 IF U = 2 AND V >  = 1000000 THEN U$ = U
     $(3):V = V / 1000000                                           C0F3
2615 IF U = 3 AND V >  = 1000000 THEN U$ = "
     million(s) de m":V = V / 1000000                               B189
2620 IF U = 4 THEN U$ = "million(s) de m"                           8125
2625 V =  INT (V * 100) / 100: VTAB 15: HTAB
     15: PRINT T$;" = ";V;" ";U$;"3":U$ = U$
     (U): RETURN                                                    975A
2630 :                                                              003A
2635 REM ***  RECOPIE D'ECRAN  ***
2640 :                                                              003A
2645 PRINT : PRINT D$;"PR£1": PRINT  CHR$ (9
     );"81N"                                                        6926
2650 FOR E = 0 TO 19: READ L                                        83FE
2655 POKE 788,L -  INT (L / 256) * 256: POKE
     789, INT (L / 256)  ,                                          C97F
2660 CALL 768: PRINT M$: NEXT                                       CC52
2665 PRINT D$;"PR£0": PRINT D$;"PR£3": RESTO
     RE : GOTO 115                                                  2CCD
2670 DATA 1024,1152,1280,1408,1536,1664,1792
     ,1920,1064,1192,1320,1448,1576,1704,183
     2,1960,1104,1232,1360,1488                                     90C7
2675 :                                                              003A
2680 REM ***  FILET HAUT  ***
2685 :                                                              003A
2690 FOR I = 1 TO 80: PRINT "=";: NEXT : PRI
     NT : RETURN                                                    503F
2695 :                                                              003A
2700 REM ***  AVERTISSEMENT  ./, ***
2705 :                                                              003A
2710 GOSUB 2690: VTAB 4: HTAB 15: PRINT M2$;
     U$: PRINT : HTAB 15: PRINT M3$: RETURN                         F0BF
2715 :                                                              003A
2720 REM ***  BYE BYE  ***
2725 :                                                              003A
2730 HOME : VTAB 8: HTAB 35: PRINT "Au revoi
     r...": VTAB 11: HTAB 11: PRINT "N'oubli
     ez pas de m'éteindre et de ranger votre
      disquette...!": PRINT BIP$: PRINT BIP$
     : END                                                          F844
```

IMP.LM

M$ doit être la première variable du programme, ce qui est le cas (voir ligne 120).

```
300:  AD 14 03 8D 1E 03 AD 15
308:  03 8D 1F 03 A2 00 A0 00
310:  8D 55 C0 BD 00 04 99 00
318:  60 C8 8D 54 C0 BD 00 04
320:  99 00 60 C8 E8 E0 28 D0
328:  E7 A5 69 18 69 02 8D 3E
330:  03 A5 6A 69 00 8D 3F 03
338:  A2 00 8D 46 03 9D 00 00
340:  E8 E0 03 D0 F5 60 50 00
348:  60
```

(extrait de TM n°1)

BSAVE IMP.LM,A$300,L$49 ■

Votre bibliothèque INFORMATIQUE

par **NESTOR**

• APPLE, LOGIQUE ET SYSTÈMES EXPERTS (René Descamps)

Pas de bonne programmation sans une logique sans faille, ce qui n'implique pas qu'un homme logique fasse forcément de la bonne programmation. Je n'irai pas plus loin et je vous laisse le soin de découvrir vous-même, dans le livre de René Descamps, comment résoudre un postulat du genre :

Seuls les anges ont des ailes
Aucun corbeau n'est un ange
Tout corbeau a des ailes...

Les programmes de cet ouvrage sont en Basic et vous seront familiers puisque réalisés sur Apple. Ils vous aideront à construire un "moteur d'inférence" et à trouver la solution d'énigmes logiques. Vous aurez d'abord lu une courte, mais instructive introduction vous conduisant à la recherche de l'intelligence (artificielle... bien sûr). Vous aurez ensuite abordé la notion de récursivité, la structure de la pile avant de passer à l'évaluation d'une expression logique. Irez-vous jusqu'à construire votre propre système expert ? Je le souhaite car, si votre programme le mérite, je suis sûr que les lecteurs de *Tremplin Micro* ne tarderont pas à le consulter dans les colonnes de leur revue préférée !

EDITIONS DU P.S.I. *(Micro et techniques), BP 86 — 77402 LAGNY/S/MARNE CEDEX, Tél. (16) 05.21.22.01 (120 F ttc).*

• DU BASIC AU LANGAGE C (Robert J. Traister)

On parle beaucoup du langage C et d'aucuns prétendent que, s'il ne devait subsister qu'un seul langage, ce devrait être le langage C. L'auteur de cet ouvrage (traduit en français par Dimitri Stoquart) est un spécialiste américain .
Il s'adresse aux utilisateurs du Basic et son but avoué est de faciliter leur passage au langage C. Si j'en juge par les premiers exemples donnés dans son livre, il semble qu'il puisse gagner son pari. C'est clairement dit et le néophyte que je suis (et reste... car j'ai seulement parcouru les dix chapitres du bouquin) a même compris plusieurs choses.

Premier constat : ce n'est pas gagné d'avance et, par rapport au Basic, on ne peut pas prétendre que la syntaxe soit facile à retenir.
Par contre, j'avoue être maintenant convaincu de la puissance du langage C.

C'est en tout cas une approche à lire avec intérêt et je ne vous cacherai pas plus longtemps que j'ai maintenant l'intention de poursuivre une étude qui devrait se révéler passionnante. J'aimerais toutefois pouvoir l'entreprendre sur mon Apple (le *GS* éventuellement) plutôt que sur un IBM ou compatible.

INTEREDITIONS *(Collection Intermicro), 87 Avenue du Maine, 75014 PARIS.*

• UNE GRAVITATION SANS GRAVITÉ (Jayant V. Narlikar)

L'auteur est professeur d'astrophysique à Bombay (Inde) et il a enseigné à l'Université de Cambridge où son travail de vulgarisation scientifique lui a valu le prix décerné par la fondation ATRE.
Pourquoi vous parler de cette traduction de Jean-Pierre Maury ? Parce que j'ai pris beaucoup de plaisir à la dévorer. Mais oui : je croyais que la gravitation universelle était un phénomène bien connu et j'apprends qu'elle reste mystérieuse. Et puis, pourquoi le cacher, j'adore que l'on me parle des planètes, comètes, trous noirs et autres trous blancs ! L'auteur se révèle non seulement concis, mais agréable à lire. Je n'ai pas tout compris (attention ! c'est un ouvrage sérieux... avec des graphiques et quelques formules !), mais je connais mieux cette gravitation... finalement maîtresse de l'Univers.

PAYOT *(Espace des Sciences), 106 boulevard Saint-Germain, 75006 PARIS (110 F ttc)*

UTILITAIRE | DOS | ProDOS

Fonctionne aussi sur l'APPLE IIGS

Yvan KOENIG

AUX.DRAW

Comment transférer une table de formes en mémoire auxiliaire... et l'utiliser ?

Yvan KOENIG — dont on ne vantera jamais assez la compétence en matière de langage machine — vous propose une solution à la fois élégante et rationnelle pour transférer facilement une table de formes en mémoire auxiliaire, puis s'en servir sans aucun problème. Notez que cet utilitaire fonctionne sous ProDOS, mais tourne aussi sous DOS 3.3 après une légère modification (lisez à ce sujet la présentation du programme source en assembleur, en haut de la page 42).

La DÉMO utilise une fonte de caractères publiée dans le numéro 1 de *Tremplin Micro* : CARAC. Attention ! ne confondez pas une telle table de formes avec les polices classiques de caractères, dont la longueur est de 768 octets (lire notre n°12).

Cet utilitaire comprend :

- **TEST.AUX.DRAW**
  Simple démo, comme vous pouvez le constater.
- **AUX.DRAW.S**
  Programme source (assembleur MERLIN, mais les utilisateurs de ProCODE ne devraient pas rencontrer de difficulté particulière pour le recopier).
- **AUX.DRAW**
  Les codes, faciles à recopier sans erreur pour celles et ceux qui disposent de la disquette *SIGNATURE*.

AJOUTEZ une ligne :

```
25 PRINT CHR$(4) "CAT/RAM"
```

TEST.AUX.DRAW

```
10 IF  PEEK (104) <  > 64 THEN  POKE 103,1:
   POKE 104,64: POKE 16384,0: PRINT  CHR$ (
   4)"RUN TEST.AUX.DRAW"                    1DD1
20 PRINT  CHR$ (21): TEXT : HOME : PRINT "P
   ATIENCE"                                 206A
30 D$ =  CHR$ (4): PRINT D$"BRUN AUX.DRAW"  42E5
40 PRINT D$"BLOAD CARAC,A$A00": &  STORE    B305
50 HCOLOR= 3: ROT= 0: SCALE= 1              D7CB
60 HGR : POKE 49234,0                       F3E6
70 FOR I = 1 TO 26: &  DRAW I AT 8 * I,10:
   NEXT                                     2913
80 FOR I = 1 TO 7: &  DRAW I + 26 AT 8 * I,
   20: NEXT                                 4D13
90 FOR I = 1 TO 26: &  DRAW I + 33 AT 8 * I
   ,30: NEXT                                9A43
100 FOR I = 1 TO 6: &  DRAW I + 59 AT 8 * I,
   40: NEXT                                 E71A
110 FOR I = 1 TO 26: &  DRAW I + 65 AT 8 * I
   ,50: NEXT                                AB4A
120 FOR I = 1 TO 5: &  DRAW I + 91 AT 8 * I,
   60: NEXT                                 C317
130 POKE 49168,0: WAIT 49152,128: POKE 49168
   ,0: TEXT : HOME                          33CB
```

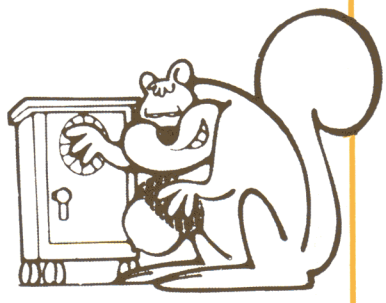
```
1
2   ************************************************************
3   *                                                          *
4   *               & DRAW, & XDRAW, & STORE                   *
5   *               ------------------------                   *
6   * & STORE marque la zone $6000-$87FF occupée en RAMdisk    *
7   *         et transfère la table de forme déposée en        *
8   *         MAINBUF vers AUXBUF en mémoire auxiliaire        *
9   * & DRAW et & XDRAW s'utilisent comme DRAW et XDRAW        *
10  *         mais utilisent la table en AUX                   *
11  *----------------------------------------------------------*
12  * Assembleur MERLIN.PRO            Roger WAGNER Editeur    *
13  *----------------------------------------------------------*
14  * Yvan KOENIG . VALLAURIS                    Le 09/12/86   *
15  ************************************************************
16  * Cette routine est écrite pour utilisation sous ProDOS    *
17  * On peut l'utiliser sous DOS 3.3 en remplaçant PROLEN     *
18  * par DOSLEN = $AA60.          La mise à jour de VOLMAP    *
19  *         est alors inutile mais elle ne perturbe rien.    *
20  ************************************************************
21          ORG   $810
22  ********************************
23
24  at      =     $C5         ;Token 'AT'
25
26  SHAPEL  =     $1A
27  SHAPEH  =     SHAPEL+1
28  from    =     $3C         ;&3D
29  end     =     $3E         ;&3F
30  dest    =     $42         ;&43
31  CHRGET  =     $B1
32  CHRGOT  =     $B7
33  DXL     =     $D0
34  DXH     =     DXL+1
35  DY      =     $D2
36  QDRNT   =     $D3
37  EL      =     $D4
38  EH      =     EL+1
39  HNDX    =     $E5
40  SCALEZ  =     $E7
41  SHAPEPNT =    $E8         ;&E9
42  COLCOUNT =    $EA
43  ROTZ    =     $F9
44
45  VOLMAP  =     $3C2
46  AMPERV  =     $3F5
47
48  MAINBUF =     $A00        ;BLOAD TABLE ,A$A00
49                            ;ce qui pourra 'écraser' HGR1
50  ************************************************
51  AUXBUF  =     $6000                             *
52  * Si l'on n'emploie pas la DHGR on peut mettre  *
53  * AUXBUF en $2000 ou même en $1000.             *
54  * Se référer à PROTECT/RAM dans Tremplin n°12   *
```

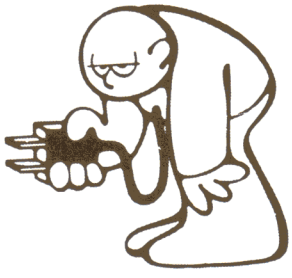
À NOS CORRESPONDANT(E)S

La Rédaction de *Tremplin Micro* et Yvan KŒNIG répondent volontiers aux questions des Lectrices et Lecteurs de la revue, mais on comprendra que certaines règles doivent absolument être respectées... et notamment :

- Ne pas nous adresser des programmes qui ne fonctionnent pas. Chercher une erreur exige parfois plusieurs heures et nous ne pouvons donc pas nous livrer à ce genre d'exercice.
- Formulation aussi claire que possible du problème (sans oublier des renseignements sur la configuration utilisée).
- Joindre une enveloppe timbrée pour la réponse.

```
                        55   * pour déterminer la VOLMAP correspondant      *
                        56   * à l'utilisation effective de la memAUX.      *
                        57   ***********************************************
                        58
                        59   PROLEN    =     $BEC8       ;Longueur du dernier BIN
                        60                               ;si on n'a pas spécifié L !!!
                        61   AUXMOVE   =     $C311       ;MAIN<->AUX, ne change pas X/Y
                        62
                        63   SYNCHR    =     $DEC0
                        64   GETBYT    =     $E6F8
                        65   GOERR     =     $F206
                        66   HPOSN     =     $F411
                        67   LRUD1     =     $F4B3
                        68   LRUD2     =     $F4B4
                        69   LRUDX1    =     $F49C
                        70   LRUDX2    =     $F49D
                        71   COSTBL    =     $F5BA
                        72   HFNS      =     $F6B9
                        73
0810: A9 20             74   INIT      LDA   #<ENTRY
0812: 8D F6 03          75             STA   AMPERV+1
0815: A9 08             76             LDA   #>ENTRY
0817: 8D F7 03          77             STA   AMPERV+2
081A: A9 4C             78             LDA   #$4C
081C: 8D F5 03          79             STA   AMPERV
081F: 60                80   Rts       RTS
                        81
0820: C9 A8             82   ENTRY     CMP   #$A8        ;STORE ?
0822: F0 1A             83             BEQ   STORE
0824: C9 95             84             CMP   #$95        ;XDRAW ?
0826: F0 0D             85             BEQ   XDRAW
0828: C9 94             86             CMP   #$94        ;DRAW ?
082A: D0 F3             87             BNE   Rts         ;Erreur
                        88
082C: 20 B1 00          89   DRAW      JSR   CHRGET
082F: 20 03 09          90             JSR   DRWPNT
0832: 4C 48 08          91             JMP   DODRAW
                        92
0835: 20 B1 00          93   XDRAW     JSR   CHRGET
0838: 20 03 09          94             JSR   DRWPNT
083B: 4C A6 08          95             JMP   DOXDRAW
                        96
083E: 20 B1 00          97   STORE     JSR   CHRGET
0841: 4C 6F 09          98             JMP   DOSTORE
                        99
0844: 86 1A            100   DRAW0     STX   SHAPEL       ;Pointeur sur la Forme
0846: 84 1B            101             STY   SHAPEH
0848: AA               102   DODRAW    TAX               ;A = valeur de ROT (0-$3F)
0849: 4A               103             LSR
084A: 4A               104             LSR
084B: 4A               105             LSR
084C: 4A               106             LSR
084D: 85 D3            107             STA   QDRNT       ;QDRNT 0=haut, 1=droite
084F: 8A               108             TXA               ; 2=bas, 3=gauche
0850: 29 0F            109             AND   #%00001111
```



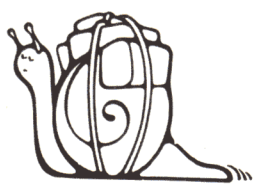


```
0852: AA          110              TAX
0853: BC BA F5    111              LDY   COSTBL,X
0856: 84 D0       112              STY   DXL          ;Place dans DXL et DY.
0858: 49 0F       113              EOR   £%00001111 ;les valeurs de COS et SIN
085A: AA          114              TAX               ;correspondant à ROT
085B: BC BB F5    115              LDY   COSTBL+1,X
085E: C8          116              INY
085F: 84 D2       117              STY   DY
0861: A4 E5       118              LDY   HNDX         ;Index par rapport
0863: A2 00       119              LDX   £0           ;à l'adresse de base
0865: 86 EA       120              STX   COLCOUNT     ;Init. compteur de collision
0867: 20 46 09    121              JSR   LOADindX     ;Lit 1er octet de la forme
086A: 85 D1       122  D2          STA   DXH
086C: A2 80       123              LDX   £$80
086E: 86 D4       124              STX   EL           ;Prépare le calcul des vecteurs
0870: 86 D5       125              STX   EH           ; Left, Right, Up, Down
0872: A6 E7       126              LDX   SCALEZ       ;Coefficient d'échelle
0874: A5 D4       127  D3          LDA   EL
0876: 38          128              SEC
0877: 65 D0       129              ADC   DXL
0879: 85 D4       130              STA   EL
087B: 90 04       131              BCC   D4
087D: 20 B3 F4    132              JSR   LRUD1        ;Déplace selon vecteur
0880: 18          133              CLC
0881: A5 D5       134  D4          LDA   EH
0883: 65 D2       135              ADC   DY
0885: 85 D5       136              STA   EH
0887: 90 03       137              BCC   D5
0889: 20 B4 F4    138              JSR   LRUD2        ;Déplace selon vecteur+90°
088C: CA          139  D5          DEX                ;Boucle selon l'échelle
088D: D0 E5       140              BNE   D3
088F: A5 D1       141              LDA   DXH
0891: 4A          142              LSR                ;Passe aux 3 bits suivants
0892: 4A          143              LSR                ; de la définition
0893: 4A          144              LSR
0894: D0 D4       145              BNE   D2           ;On n'a pas terminé / cet octet
0896: E6 1A       146              INC   SHAPEL
0898: D0 02       147              BNE   D6
089A: E6 1B       148              INC   SHAPEL+1
089C: 20 46 09    149  D6          JSR   LOADindX     ;Octet suivant de la définition
089F: D0 C9       150              BNE   D2           ;Si octet nul, c'est fini.
08A1: 60          151              RTS
                  152
                  153  * Routine XDRAW, se reporter à DRAW pour commentaires
                  154
08A2: 86 1A       155  XDRAW0      STX   SHAPEL
08A4: 84 1B       156               STY   SHAPEH
08A6: AA          157  DOXDRAW     TAX
08A7: 4A          158               LSR
08A8: 4A          159               LSR
08A9: 4A          160               LSR
08AA: 4A          161               LSR
08AB: 85 D3       162               STA   QDRNT
08AD: 8A          163               TXA
08AE: 29 0F       164               AND   £%00001111
```

```
08B0: AA          165              TAX
08B1: BC BA F5    166              LDY   COSTBL,X
08B4: 84 D0       167              STY   DXL
08B6: 49 0F       168              EOR   £%00001111
08B8: AA          169              TAX
08B9: BC BB F5    170              LDY   COSTBL+1,X
08BC: C8          171              INY
08BD: 84 D2       172              STY   DY
08BF: A4 E5       173              LDY   HNDX
08C1: A2 00       174              LDX   £0
08C3: 86 EA       175              STX   COLCOUNT
08C5: 20 46 09    176              JSR   LOADindX
08C8: 85 D1       177  XD2         STA   DXH
08CA: A2 80       178              LDX   £$80
08CC: 86 D4       179              STX   EL
08CE: 86 D5       180              STX   EH
08D0: A6 E7       181              LDX   SCALEZ
08D2: A5 D4       182  XD3         LDA   EL
08D4: 38          183              SEC
08D5: 65 D0       184              ADC   DXL
08D7: 85 D4       185              STA   EL
08D9: 90 04       186              BCC   XD4
08DB: 20 9C F4    187              JSR   LRUDX1
08DE: 18          188              CLC
08DF: A5 D5       189  XD4         LDA   EH
08E1: 65 D2       190              ADC   DY
08E3: 85 D5       191              STA   EH
08E5: 90 03       192              BCC   XD5
08E7: 20 9D F4    193              JSR   LRUDX2
08EA: CA          194  XD5         DEX
08EB: D0 E5       195              BNE   XD3
08ED: A5 D1       196              LDA   DXH
08EF: 4A          197              LSR
08F0: 4A          198              LSR
08F1: 4A          199              LSR
08F2: D0 D4       200              BNE   XD2
08F4: E6 1A       201              INC   SHAPEL
08F6: D0 02       202              BNE   XD6
08F8: E6 1B       203              INC   SHAPEL+1
08FA: 20 46 09    204  XD6         JSR   LOADindX
08FD: D0 C9       205              BNE   XD2
08FF: 60          206              RTS
                  207
0900: 4C 06 F2    208  GGERR       JMP   GOERR       ;Illegal quantity
                  209
0903: 20 F8 E6    210  DRWPNT      JSR   GETBYT      ;numéro de forme dans X
0906: A5 E8       211              LDA   SHAPEPNT
0908: 85 1A       212              STA   SHAPEL
090A: A5 E9       213              LDA   SHAPEPNT+1
090C: 85 1B       214              STA   SHAPEH
090E: 8A          215              TXA
090F: A2 00       216              LDX   £0
0911: 20 66 09    217              JSR   CMPindX     ;Y a-t-il une forme de ce num.
0914: F0 02       218              BEQ   DP1
0916: B0 E8       219              BCS   GGERR       ;NON -> Erreur          (BGE)
```

Sous DOS 3.3 prenez garde. Il existe diverses routines permettant d'utiliser la mémoire auxiliaire comme disque virtuel. Elles sont toutes différentes. Je n'ai pu en tenir compte en ce qui concerne la protection de notre table de formes.

```
0918: 0A        220 DP1     ASL             ;Détermine l'adresse de la
0919: 90 02     221         BCC   DP2       ;forme dans la table.
091B: E6 1B     222         INC   SHAPEH
091D: A8        223 DP2     TAY
091E: 20 44 09  224         JSR   LOADindY  ;  détruit C
0921: 18        225         CLC             ;Il nous faut C=0
0922: 65 1A     226         ADC   SHAPEL
0924: AA        227         TAX
0925: C8        228         INY
0926: 08        229         PHP             ;Sauve C
0927: 20 44 09  230         JSR   LOADindY  ;  détruit C
092A: 28        231         PLP             ;Récupère C
092B: 65 E9     232         ADC   SHAPEPNT+1
092D: 85 1B     233         STA   SHAPEH    ;Sauve adresse de la forme
092F: 86 1A     234         STX   SHAPEL
0931: 20 B7 00  235         JSR   CHRGOT    ;Controle
0934: C9 C5     236         CMP   £at       ;de la syntaxe
0936: D0 09     237         BNE   DP3
0938: 20 C0 DE  238         JSR   SYNCHR
093B: 20 B9 F6  239         JSR   HFNS      ;Où faut-il dessiner ?
093E: 20 11 F4  240         JSR   HPOSN
0941: A5 F9     241 DP3     LDA   ROTZ
0943: 60        242         RTS
                243
                244 LOADindY =     *         ;Remplace LDA (SHAPEL),Y /AUX
0944: 98        245         TYA
0945: 24        246         HEX   24        ;Saute instruction suivante
                247 LOADindX =     *         ;Remplace LDA (SHAPEL,X) /AUX
0946: 8A        248         TXA
0947: 18        249         CLC
0948: 65 1A     250         ADC   SHAPEL
094A: 85 3C     251         STA   from      ;Adresse de l'octet
094C: 85 3E     252         STA   end       ;à récupérer
094E: A9 00     253         LDA   £0
0950: 65 1B     254         ADC   SHAPEH
0952: 85 3D     255         STA   from+1
0954: 85 3F     256         STA   end+1
0956: A9 AF     257         LDA   £<DEPOT   ;Adresse où on le déposera
0958: 85 42     258         STA   dest
095A: A9 09     259         LDA   £>DEPOT
095C: 85 43     260         STA   dest+1
095E: 18        261         CLC
095F: 20 11 C3  262         JSR   AUXMOVE
0962: AD AF 09  263         LDA   DEPOT     ;récupère l'octet convoité
0965: 60        264         RTS
                265
                266 CMPindX  =     *         ;Remplace CMP (SHAPEL,X) /AUX
0966: 48        267         PHA             ;sauve Accu
0967: 20 46 09  268         JSR   LOADindX  ;Place (SHAPEL,X) dans DEPOT
096A: 68        269         PLA             ;Récupère Accu
096B: CD AF 09  270         CMP   DEPOT     ;Compare
096E: 60        271         RTS
                272
096F: A9 C3     273 DOSTORE LDA   £<VOLMAP+1
0971: 85 42     274         STA   dest
```



```
0973: A9 03      275            LDA   £)VOLMAP+1
0975: 85 43      276            STA   dest+1
0977: A9 B0      277            LDA   £(OURMAP    ;On va mettre en place
0979: 85 3C      278            STA   from        ;les octets de la VOLMAP
097B: A9 09      279            LDA   £)OURMAP    ;du RAMdisk correspondant
097D: 85 3D      280            STA   from+1      ;à la zone $6000-$87FF
097F: A9 BA      281            LDA   £(OURLAST   ;qui apparaitra partiellement
0981: 85 3E      282            STA   end         ;occupée, ce qui protègera
0983: A9 09      283            LDA   £)OURLAST   ;la table dans le cas où
0985: 20 A9 09   284            JSR   movaux      ;l'on emploierait le RAMdisk
                 285
0988: A9 00      286            LDA   £(AUXBUF
098A: 85 42      287            STA   dest
098C: 85 E8      288            STA   SHAPEPNT    ;Adresse de la Table
098E: A9 60      289            LDA   £)AUXBUF
0990: 85 43      290            STA   dest+1
0992: 85 E9      291            STA   SHAPEPNT+1
0994: A9 00      292            LDA   £(MAINBUF
0996: 85 3C      293            STA   from
0998: A9 0A      294            LDA   £)MAINBUF
099A: 85 3D      295            STA   from+1
099C: 18         296            CLC
099D: A9 FF      297            LDA   £(MAINBUF-1
099F: 6D C8 BE   298            ADC   PROLEN
09A2: 85 3E      299            STA   end
09A4: A9 09      300            LDA   £)MAINBUF-1
09A6: 6D C9 BE   301            ADC   PROLEN+1
09A9: 85 3F      302  movaux    STA   end+1
09AB: 38         303            SEC
09AC: 4C 11 C3   304            JMP   AUXMOVE     ;MAIN->AUX
                 305
09AF: 00         306  DEPOT     DS    1
                 307
                 308  ********************************
                 309  *    Protège  $6000 - $87FF    *
                 310  ********************************
                 311
09B0: FF         312  OURMAP    DFB   %11111111   ;blocs $08-$0F
09B1: FF         313            DFB   %11111111   ;blocs $10-$17
09B2: FF         314            DFB   %11111111   ;blocs $18-$1F
09B3: FF         315            DFB   %11111111   ;blocs $20-$27
09B4: D0         316            DFB   %11010000   ;blocs $28-$2F
09B5: 00         317            DFB   %00000000   ;blocs $30-$37
09B6: 08         318            DFB   %00001000   ;blocs $38-$3F
09B7: FF         319            DFB   %11111111   ;blocs $40-$47
09B8: FF         320            DFB   %11111111   ;blocs $48-$4F
09B9: FF         321            DFB   %11111111   ;blocs $50-$57
09BA: FF         322  OURLAST   DFB   %11111111   ;blocs $58-$5F
                 323
                 324  ************************************
                 325  * Pour adapter selon besoins réels *
                 326  *  voir PROTECT.RAM Tremplin n°12  *
                 327  ************************************

--End assembly, 427 bytes, Errors: 0
```

AUX.DRAW

Les valeurs indiquées en couleur ne doivent pas être tapées. Elles sont destinées à contrôler la saisie du programme, en utilisant la disquette *SIGNATURE* mise au point par Yvan KŒNIG (voir notre bulletin de commande).

Pour sauver cette routine :
BSAVE AUX.DRAW, A$810,L427

```
0810: A9 20 8D F6 03 A9 08 8D F7 03 A9 4C 8D F5 03 60   4E61
0820: C9 A8 F0 1A C9 95 F0 0D C9 94 D0 F3 20 B1 00 20   9DE7
0830: 03 09 4C 48 08 20 B1 00 20 03 09 4C A6 08 20 B1   8370
0840: 00 4C 6F 09 86 1A 84 1B AA 4A 4A 4A 4A 85 D3 8A   E7B7
0850: 29 0F AA BC BA F5 84 D0 49 0F AA BC BB F5 C8 84   BA5B
0860: D2 A4 E5 A2 00 86 EA 20 46 09 85 D1 A2 80 86 D4   65AE
0870: 86 D5 A6 E7 A5 D4 38 65 D0 85 D4 90 04 20 B3 F4   0582
0880: 18 A5 D5 65 D2 85 D5 90 03 20 B4 F4 CA D0 E5 A5   66A2
0890: D1 4A 4A 4A D0 D4 E6 1A D0 02 E6 1B 20 46 09 D0   E665
08A0: C9 60 86 1A 84 1B AA 4A 4A 4A 4A 85 D3 8A 29 0F   5654
08B0: AA BC BA F5 84 D0 49 0F AA BC BB F5 C8 84 D2 A4   C899
08C0: E5 A2 00 86 EA 20 46 09 85 D1 A2 80 86 D4 86 D5   1F93
08D0: A6 E7 A5 D4 38 65 D0 85 D4 90 04 20 9C F4 18 A5   D2CD
08E0: D5 65 D2 85 D5 90 03 20 9D F4 CA D0 E5 A5 D1 4A   C2E9
08F0: 4A 4A D0 D4 E6 1A D0 02 E6 1B 20 46 09 D0 C9 60   2473
0900: 4C 06 F2 20 F8 E6 A5 E8 85 1A A5 E9 85 1B 8A A2   92C8
0910: 00 20 66 09 F0 02 B0 E8 0A 90 02 E6 1B A8 20 44   A0C2
0920: 09 18 65 1A AA C8 08 20 44 09 28 65 E9 85 1B 86   8D23
0930: 1A 20 B7 00 C9 C5 D0 09 20 C0 DE 20 B9 F6 20 11   2716
0940: F4 A5 F9 60 98 24 8A 18 65 1A 85 3C 85 3E A9 00   CFFC
0950: 65 1B 85 3D 85 3F A9 AF 85 42 A9 09 85 43 18 20   F2D7
0960: 11 C3 AD AF 09 60 48 20 46 09 68 CD AF 09 60 A9   4746
0970: C3 85 42 A9 03 85 43 A9 B0 85 3C A9 09 85 3D A9   0635
0980: BA 85 3E A9 09 20 A9 09 A9 00 85 42 85 E8 A9 60   32E7
0990: 85 43 85 E9 A9 00 85 3C A9 0A 85 3D 18 A9 FF 6D   0442
09A0: C8 BE 85 3E A9 09 6D C9 BE 85 3F 38 4C 11 C3 00   D50B
09B0: FF FF FF FF D0 00 08 FF FF FF FF                  41D0
```

L'INFORMATIQUE... et les livres

• L'ORTHOGRAPHE À L'HEURE DE L'INFORMATIQUE *

Les micro-ordinateurs rentrent dans nos classes. Il faut espérer qu'ils seront demain en nombre suffisant et que tous, maîtres et élèves auront reçu d'ici là la formation nécessaire pour en tirer le meilleur parti. On se pose actuellement de nombreuses questions pour savoir ce que l'on va faire de ces machines, si elles vont remplacer le maître ou en quoi elles peuvent nous aider. Comme toujours dans les périodes de transition, le désordre et le trouble accompagnent les innovations et le besoin d'information est grand. S'il est toutefois un domaine où l'on attend beaucoup de l'informatique, c'est bien celui de l'orthographe. Travailler sur l'écran c'est avoir pour soi tout seul bien plus d'une armée de scribes, bien plus qu'une imprimerie : maniable, interactif, l'écran d'ordinateur permet l'autocorrection, l'autonomie, l'invention permanente, toute une métamorphose de valeurs, prometteuse d'un meilleur apprentissage.

Les auteurs de cet ouvrage, Jean-Claude Lallias et Anne Delgado nous emmènent au pays des graphèmes et nous proposent en même temps un rappel des acquis les plus actuels en matière d'approche de l'orthographe. C'est un tour d'horizon sur ce qui se fait dans ce secteur avec l'aide de l'informatique et une présentation de la série des "ORTHOBASE". Cumulant de façon nouvelle différents manuels de classe, à la fois dictionnaires, listes de fréquence, choix de tris graphiques grammaticaux et thématiques etc... ORTHOBASE est, comme son nom l'indique, une base de travail, un instrument de recherche... "il apprend à apprendre".

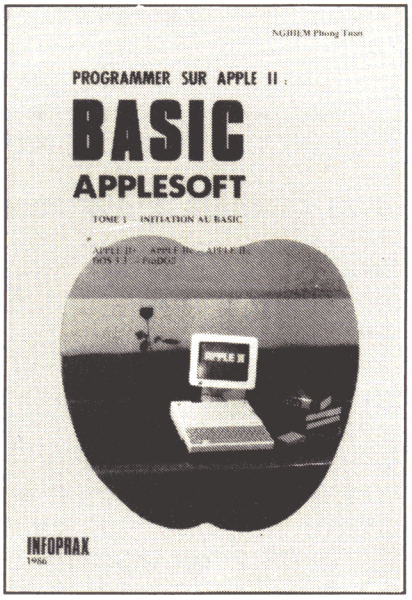
** Par J.-C. LALLIAS et A. DELGADO. (160 pages — 115 F TTC) CEDIC/NATHAN.*

• PROGRAMMER SUR APPLE II : BASIC APPLESOFT *

Ce livre apprend plus que le Basic. C'est une introduction sérieuse à l'informatique. Il montre comment utiliser ce langage pour écrire des programmes intéressants. Le lecteur est conduit pas à pas pour mettre lui-même ces programmes au point. Il apprend le BASIC au passage.

Ce premier tome initie au travail sur l'ordinateur APPLE *II*, et étudie les problèmes de base : l'utilisation des fichiers sur disquette et la gestion de la conversation qui passe par le clavier et l'écran, entre le programme et l'utilisateur, avec traitement des incidents qui peuvent se produire. Il est destiné, on s'en doute, à des personnes ne connaissant pas encore les ressources de l'Applesoft et pas davantage celles de leur machine.

** Par Ph. T. NGHIEM (324 pages — 170 F TTC) EYROLLES*



Fonctionne aussi sur l'APPLE IIGS

Clément RENARD

CADRES 40/80 COLONNES

Tracer un cadre, en mode texte, à partir d'un programme classique en Basic, est facile, mais relativement lent... d'où l'écriture de multiples routines en langage machine.
Amusez-vous à étudier CADREØ et ses variantes .

CADRE.DÉMO

```
10 PRINT CHR$ (21): TEXT : HOME
15 GOSUB 60: PRINT "CADREØ": PRINT CHR$ (4)"BLOAD CADREØ": GOSUB 50
20 GOSUB 60: PRINT "CADRE1": PRINT CHR$ (4)"BLOAD CADRE1": GOSUB 50
25 PRINT CHR$ (4)"PR£3": PRINT : PRINT CHR$ (4)"BLOAD CADRE2"
30 GOSUB 60: PRINT "CADRE2 EN 80 COLONNES ": GOSUB 50
35 GOSUB 60: PRINT "(M)ENU DE DISQUETTE ";: GET R$: PRINT
40 IF R$ = "M" THEN PRINT CHR$ (4)"RUN MENU,D1"
45 HOME : END
50 FOR I = 1 TO 5: GOSUB 55: HOME : GOSUB 55: CALL 768: NEXT: RETURN
55 CALL - 198: POKE 49168,0: WAIT 49152,128,127: POKE 49168,0: RETURN
60 VTAB 12: HTAB 12: RETURN
```

CADREØ (BSAVE CADREØ,A$300,L$28)

4Ø COLONNES — Peut être utilisé par un **BRUN CADREØ**.

```
300 :  20 58 FC    JSR  $FC58      HOME efface l'écran.
303 :  A2 17       LDX  £$17       Registre X = nombre de lignes.
305 :  8A          TXA             Numéro de ligne passé dans A avant
306 :  20 24 FC    JSR  $FC24      d'appeler VTAB+2 (positionne le curseur).
309 :  A0 27       LDY  £$27       Registre Y = longueur de la ligne (Ø à 27).
30B :  A9 20       LDA  £$20       CARACTÈRE : Espace mode inverse.
30D :  91 28       STA  ($28),Y    On trouve en $28-29 l'adresse de la ligne.
30F :  88          DEY             Y = Y — 1.
310 :  10 FB       BPL  $030D      Boucle jusqu'à Y = Ø inclus.
312 :  CA          DEX             X = X — 1.
313 :  8A          TXA             X dans A (ne modifie pas X).
314 :  F0 F0       BEQ  $0306      Si X = Ø, tracer la ligne entière.
316 :  30 0F       BMI  $0327      On a terminé...
318 :  20 24 FC    JSR  $FC24      VTAB positionne le curseur.
31B :  A0 27       LDY  £$27       Ecriture du premier octet
31D :  A9 20       LDA  £$20       et du dernier de chacune
31F :  91 28       STA  ($28),Y    des lignes $16 à 1.
321 :  A0 00       LDY  £$00
323 :  91 28       STA  ($28),Y
325 :  10 EB       BPL  $0312      Et ça repart !
327 :  60          RTS             Fin ou retour au Basic.
```

(suite page 50)

CADRE1 (VARIANTE) 40 COLONNES

```
300 : A2 17       LDX £$17
302 : 8A          TXA
303 : 20 24 FC    JSR $FC24
306 : A0 27       LDY £$27
308 : D0 0A       BNE $0314
30A : 98          TYA
30B : F0 07       BEQ $0314
30D : 8A          TXA
30E : F0 04       BEQ $0314
310 : E0 17       CPX £$17
312 : D0 04       BNE $0318
314 : A9 20       LDA £$20
316 : D0 02       BNE $031A
318 : A9 A0       LDA £$A0
31A : 91 28       STA ($28),Y
31C : 88          DEY
31D : 10 EB       BPL $030A
31F : CA          DEX
320 : 8A          TXA
321 : 10 E0       BPL $0303
323 : 85 25       STA $25
325 : 60          RTS
```

BSAVE CADRE1,A$300,L$26

CADRE2 80 COLONNES

```
300 : A2 17       LDX £$17
302 : 8A          TXA
303 : 20 24 FC    JSR $FC24
306 : A0 27       LDY £$27
308 : D0 0A       BNE $0314
30A : 98          TYA
30B : F0 07       BEQ $0314
30D : 8A          TXA
30E : F0 04       BEQ $0314
310 : E0 17       CPX £$17
312 : D0 04       BNE $0318
314 : A9 20       LDA £$20
316 : D0 02       BNE $031A
318 : A9 A0       LDA £$A0
31A : 2C 55 C0    BIT $C055
31D : 91 28       STA ($28),Y
31F : 2C 54 C0    BIT $C054
322 : 91 28       STA ($28),Y
324 : 88          DEY
325 : 10 E3       BPL $030A
327 : CA          DEX
328 : 8A          TXA
329 : 10 D8       BPL $0303
32B : 85 25       STA $25
32D : 60          RTS
```

BSAVE CADRE2,A$300,L$2E

CADRE3 40/80 COLONNES

La routine se comporte différemment selon le mode d'affichage : 40 ou 80 colonnes.

BSAVE CADRE3,A$300,L$33

```
300 : A2 17       LDX £$17
302 : 8A          TXA
303 : 20 24 FC    JSR $FC24
306 : A0 27       LDY £$27
308 : D0 0A       BNE $0314
30A : 98          TYA
30B : F0 07       BEQ $0314
30D : 8A          TXA
30E : F0 04       BEQ $0314
310 : E0 17       CPX £$17
312 : D0 04       BNE $0318
314 : A9 20       LDA £$20
316 : D0 02       BNE $031A
318 : A9 A0       LDA £$A0
31A : 2C 1F C0    BIT $C01F
31D : 10 08       BPL $0327
31F : 2C 55 C0    BIT $C055
322 : 91 28       STA ($28),Y
324 : 2C 54 C0    BIT $C054
327 : 91 28       STA ($28),Y
329 : 88          DEY
32A : 10 DE       BPL $030A
32C : CA          DEX
32D : 8A          TXA
32E : 10 D3       BPL $0303
330 : 85 25       STA $25
332 : 60          RTS
```

Si l'on n'est pas en 80 colonnes, saut.

Ecriture en MÉMOIRE AUXILIAIRE.

Retour en MÉMOIRE PRINCIPALE.

Indispensable pour avoir le curseur en haut de l'écran. ■

DIDACTIQUE ProDOS exclusivement

Fonctionne aussi sur l'APPLE IIGS

Robert CAZENAVE

QUESTIONNAIRE

Simple, mais fort intéressant

Vous allez vous-même créer vos questionnaires, puis inviter vos proches (ou vos élèves) à se soumettre aux petits tests que vous aurez ainsi préparés. Un conseil : inscrivez-vous en tête de la liste des cobayes et vous constaterez qu'il n'est pas toujours facile d'éviter les petits pièges que l'on a soi-même tendus. A l'usage, vous conviendrez qu'il ne faut pas dépasser un nombre raisonnable de questions : 6 à 8, par exemple.

COMMENT CONCEVOIR UN QUESTIONNAIRE ?

Voici un exemple, beaucoup plus parlant que de longues explications. Imaginons que le thème choisi soit la CHANSON. Vous répondrez ainsi aux deux premières interrogations du programme :

- Qui chante PARIS S'ÉVEILLE ?
- Jacques DUTRONC.

Ensuite, vous devrez fournir, dans l'ordre qui vous conviendra le mieux, les identités de deux autres vedettes (qui ne chantent pas PARIS S'ÉVEILLE) plus celle du véritable interprète : Jacques DUTRONC. Le même processus se répétera pour chaque question.

```
10 REM --------------------
15 :                                  003A
20 REM QUESTIONS / REPONSES
25 REM 16/12/1986
30 :                                  003A
35 REM --------------------
40 :                                  003A
45 TEXT : CLEAR : HOME                4051
50 DIM FICHIER$(52):S$ =  CHR$
   (7):V1 = 200:V2 = 255:V3 = 9
   0:D$ =  CHR$ (4)                   E877
55 DIM TI$(12),REP$(12),ES$(12,
   3)                                 48E1
60 :                                  003A
65 REM **** MENU PRINCIPAL  ***
   *
70 :                                  003A
75 SPEED= V2                          3C31
80 GOSUB 665:TI$ = "* TESTONS
   NOS CONNAISSANCES *": VTAB
   1: GOSUB 670                       53F0
85 VTAB 9                             3EDB
90 PRINT "1- Création d'un ques
   tionnaire": PRINT                  FE30
95 PRINT "2- Jouer": PRINT            DDF6
100 PRINT "3- Modifier un questi
   onnaire": PRINT                    748B
105 PRINT "4- Ajouter une questi
   on": PRINT                         F02D
110 PRINT "5- Quitter"                4DAE
115 HTAB 1: VTAB 20: INPUT "Votr
   e choix:";C: IF C > 5 THEN
   PRINT S$: GOTO 115                 3689
120 ON C GOTO 490,135,290,415,12
   5                                  0A55
125 TEXT : HOME : END                 5514
130 :                                 003A
135 TI$ = " * * NOMS DES FICHIER
   S DU JEU * *": GOSUB 665: VT
   AB 1: GOSUB 670: GOSUB 770:
   GOSUB 665:TI$ =  MID$ (FICHI
   ER$(CH),4,13): VTAB 1: GOSUB
   670                                1536
140 GOSUB 605: REM ** LECTURE DU
    QUESTIONNAIRE                     1B4B
145 GOSUB 690: REM *** AFFICHAGE
    DU SCORE                          074F
150 D = 2: REM * NOMBRE D'ESSAIS      4F46
```

QUESTIONNAIRE (suite)

```
155 :                                                    003A
160 REM * BOUCLE D'INTERROGATION
     *
165 :                                                    003A
170 FOR N = 1 TO NQ                                      9D30
175 GOSUB 690                                            074F
180 HOME : HTAB 1: VTAB 7: SPEED
    = V1                                                 3815
185 F = F + 1                                            1955
190 PRINT "QUESTION N° ";N: PRIN
    T TI$(N)                                             313C
195 VTAB 10                                              6A03
200 PRINT "REPONSE:";: INPUT R$                          2DC3
205 SPEED= V3                                            3A32
210 IF R$ = REP$(N) THEN  PRINT
    "REPONSE EXACTE ":SC = SC +
    1: GOSUB 690: GOTO 255                               A0AE
215 PRINT : PRINT "REPONSE FAUSS
    E": PRINT S$;: PRINT S$: FOR
    K = 1 TO 60: NEXT                                    DDF0
220 IF F < D THEN  PRINT "NOUVEL
     ESSAI": GOTO 180                                    71B6
225 HTAB 1: VTAB 10: CALL  - 958
    : PRINT "Pour vous aider:":
    PRINT                                                F7D7
230 FOR J = 1 TO 3: PRINT J;"-";
    " ";ES$(N,J): NEXT                                   9D11
235 HTAB 1: VTAB 20: INPUT "QUEL
     NUMERO:";NU: IF NU > 3 THEN
    235                                                  3F36
240 ES$ = ES$(N,NU)                                      05B6
245 IF ES$ = REP$(N) THEN  PRINT
    "REPONSE EXACTE":SC = SC + 1
    : GOSUB 690: GOTO 255                                81D4
250 PRINT : PRINT "La bonne répo
    nse était ";REP$(N): GOSUB 7
    00                                                   F6D6
255 F = 0: SPEED= V2: NEXT                               6C6D
260 TEXT : HOME : VTAB 10: HTAB
    1: INPUT "On continue (O/N)?
    ";R$: IF R$ <  > "O" AND R$
    <  > "N" THEN  PRINT S$;: GOT
    O 260                                                D09B
265 IF R$ = "N" THEN  RUN                                34F5
270 N = 0: GOTO 135                                      28CC
275 :                                                    003A
280 REM * MODIFICATION *
285 :                                                    003A
290 TI$ = " * * FICHIERS A MODIF
    IER **": GOSUB 665: VTAB 1:
    GOSUB 670: GOSUB 770: GOSUB
    665:TI$ = "MODIFICATION DE :
    "" +  MID$ (FICHIER$(CH),4,1
    3): VTAB 1: GOSUB 670: GOSUB
    605                                                  33FA
295 PRINT : PRINT                                        15AE
300 FOR K = 1 TO NQ: PRINT K;"";
    "-";"";TI$(K)                                        80AE
305 PRINT REP$(K)                                        6261
310 FOR J = 1 TO 3: PRINT ES$(K,
    J): NEXT                                             E03E
315 GOSUB 700: HOME                                      D018
320 NEXT                                                 0582
325 VTAB 10: INPUT "N° de la que
    stion à modifier (0=menu):";
    K: IF K > NQ THEN  PRINT S$;
    : GOTO 325                                           368D
330 IF K = 0 THEN  RUN                                   4D68
335 HOME :TI$ = "Modifications p
    ossibles:": VTAB 4: GOSUB 67
    0                                                    0614
340 PRINT : PRINT                                        15AE
345 PRINT "QUESTION.....Q"                               42C1
350 PRINT "REPONSE.....R"                                7294
355 PRINT "CHOIX.....C"                                  A540
360 PRINT "LES 3 ENSEMBLE.....E"                         5FCB
365 PRINT "SUPPRESSION.....S"                            3F2C
370 VTAB 20: INPUT "VOTRE CHOIX:
    ";C$: HOME : VTAB 5                                  EEEF
375 IF C$ = "Q" THEN  PRINT "Que
    stion initiale:";: PRINT TI$
    (K): PRINT "Nouvelle questio
    n:": GOSUB 720: PRINT : PRIN
    T : GOTO 450                                         4245
380 IF C$ = "R" THEN  PRINT "Rép
    onse initiale:";: PRINT REP$
    (K): PRINT "Nouvelle réponse
    :": GOSUB 730: PRINT : PRINT
    : GOTO 450                                           BD45
385 IF C$ = "C" THEN  PRINT "Cho
    ix initial:": FOR J = 1 TO 3
    : PRINT ES$(K,J): NEXT : PRI
    NT : PRINT "Nouveau choix:":
    GOSUB 740: PRINT : PRINT : G
    OTO 450                                              0693
390 IF C$ = "E" THEN  PRINT "Que
    stion et réponse:": PRINT TI
    $(K): PRINT : PRINT REP$(K):
    PRINT "Choix:": FOR J = 1 TO
    3: PRINT ES$(K,J): NEXT : PR
    INT "Nouvelle question et ré
    ponses:": GOSUB 720: GOSUB 7
    30: GOSUB 740: PRINT : PRINT
    : GOTO 450                                           6435
395 IF C$ = "S" THEN NQ = NQ - 1
    : FOR J = K TO NQ:TI$(J) = T
```

```
    I$(J + 1):REP$(J) = REP$(J +
    1): FOR I = 1 TO 3:ES$(J,I)
    = ES$((J + 1),I): NEXT : NEX
    T : PRINT : PRINT : GOTO 450  7294
400 :                             003A
405 REM * ADD. D'UNE QUESTION *
410 :                             003A
415 TI$ = " * * QUESTIONS SUPPLE
    MENTAIRES * *": GOSUB 665: V
    TAB 1: GOSUB 670: GOSUB 770:
    GOSUB 605                     E7B5
420 HOME : IF NQ = 10 AND FL = 0
    THEN  VTAB 10: PRINT S$;:TI$
    = "FICHIER COMPLET": GOSUB 6
    70: GOSUB 700: RUN            BE45
425 IF FL = 1 AND NQ = 10 THEN 4
    50                            4F3A
430 FL = 1:NQ = NQ + 1:K = NQ: G
    OSUB 720: GOSUB 730: GOSUB 7
    40                            FB54
435 IF NQ = 10 THEN TI$ = "FICHI
    ER COMPLET": VTAB 20: GOSUB
    670: GOTO 450                 7A87
440 VTAB 22: HTAB 1: INPUT "Autr
    e question (O/N)?";R$: IF R$
    <  > "O" AND R$ <  > "N" THE
    N PRINT S$;: GOTO 440         FC01
445 IF R$ = "O" THEN 420          85E0
450 HTAB 1: VTAB 22: CALL  - 958
    : INPUT "Confirmation (O/N)
    ?";R$: IF R$ <  > "O" AND R$
    <  > "N" THEN  PRINT S$;: GO
    TO 450                        77C7
455 IF R$ = "N" THEN  RUN         34F5
460 HOME : VTAB 10:TI$ = "ENREGI
    STREMENT EN COURS": GOSUB 67
    0                             C3F5
465 PRINT D$;"DELETE";FI$         6F42
470 GOTO 545                      1F49
475 :                             003A
480 REM * CREATION QUESTIONNAIRE
     *
485 :                             003A
490 TI$ = " * * CREATION DU QUES
    TIONNAIRE * *": GOSUB 665: V
    TAB 1: GOSUB 670              EC51
495 VTAB 4: HTAB 1: INPUT "Nombr
    e de questions (maximum 10):
    ";NQ: IF NQ > 10 THEN 495     E870
500 FOR K = 1 TO NQ               CB2D
505 PRINT : PRINT                 15AE
510 GOSUB 720: GOSUB 730          04CD
515 GOSUB 740                     E74B
520 NEXT : PRINT : PRINT          016A
525 VTAB 23: HTAB 1: INPUT "NOM
    DU FICHIER (0=MENU):";FI$: I
    F FI$ = "" THEN  PRINT S$;:
    GOTO 525                      A061
530 IF FI$ = "0" THEN  RUN        9514
535 HOME : VTAB 10: PRINT " ENRE
    GISTREMENT DU FICHIER "       3F4B
540 FI$ = "JX" + FI$: REM * NOM
    DU FICHIER                    09E4
545 PRINT D$;"OPEN";FI$           1EC1
550 PRINT D$;"WRITE";FI$          5A1A
555 PRINT NQ                      6159
560 FOR K = 1 TO NQ               CB2D
565 PRINT TI$(K): PRINT REP$(K)   29B2
570 FOR J = 1 TO 3: PRINT ES$(K,
    J): NEXT J                    3688
575 NEXT K                        A1CD
580 PRINT D$;"CLOSE"              C517
585 RUN                           59AC
590 :                             003A
595 REM * LECTURE QUESTIONNAIRE
    *
600 :                             003A
605 PRINT D$;"OPEN";FI$           1EC1
610 PRINT D$;"READ";FI$           FCAB
615 INPUT NQ                      2523
620 FOR K = 1 TO NQ               CB2D
625 INPUT TI$(K),REP$(K)          EFB4
630 FOR J = 1 TO 3: INPUT ES$(K,
    J): NEXT                      0808
635 NEXT                          0582
640 PRINT D$;"CLOSE"              C517
645 RETURN                        63B1
650 :                             003A
655 REM * CENTRAGE DU TITRE *
660 :                             003A
665 HOME : VTAB 1: INVERSE : PRI
    NT SPC( 40)"";: POKE 34,2: R
    ETURN                         CFE2
670 HTAB (21 -  LEN (TI$) / 2):
    PRINT TI$: NORMAL : RETURN    487C
675 :                             003A
680 REM * AFFICHAGE SCORE *
685 :                             003A
690 SPEED= V2: VTAB 24: HTAB 1:
    CALL - 958: PRINT "Nombre de
     points: ";SC;: POKE 35,23:
    RETURN                        5D87
695 :                             003A
700 SPEED= V2: VTAB 24: HTAB 1:
    PRINT "... """Appuyez sur un
    e touche SVP ""...";: POKE 4
    9168,0: WAIT  - 16384,128: G
    ET R$: RETURN                 D365
```

(suite page 54)

QUESTIONNAIRE (suite)

```
705 :                                          003A
710 REM * CREATION/MODIFIC.*
715 :                                          003A
720 PRINT "QUESTION N° ";K: PRIN
    T : INPUT TI$(K): PRINT : RE
    TURN                                       69D3
725 :                                          003A
730 INPUT "REPONSE:";REP$(K): RE
    TURN                                       42EB
735 :                                          003A
740 FOR J = 1 TO 3: INPUT "CHOIX
     DE REPONSES:";ES$(K,J): NEX
    T                                          5274
745 IF ES$(K,1) <  > REP$(K) AND
    ES$(K,2) <  > REP$(K) AND ES
    $(K,3) <  > REP$(K) THEN  PR
    INT "ERREUR DANS LES CHOIX D
    E REPONSES": PRINT S$;: GOTO
    740                                        D018
750 RETURN                                     63B1
755 :                                          003A
760 REM * LECT. FICHIERS DE JEU
    *
765 :                                          003A
770 TI$ = "... ""Veuillez patien
    ter SVP ""...": VTAB 10: GOS
    UB 670                                     16C2
775 PRINT D$;" PREFIX": INPUT PR
    $                                          F713
780 PRINT D$;"OPEN";PR$;",TDIR":
    PRINT D$;"READ";PR$                        07AA
785 INPUT A$,B$,C$                             8C0E
790 N = N + 1                                  7F65
795 INPUT FICHIER$(N):FICHIER$ =
    FICHIERS$(N)                               3179
800 IF FICHIERS$ = "" THEN N = N
    - 1: GOTO 820                              16DB
805 NOM$ =  MID$ (FICHIER$,2,15)               C827
810 IF  LEFT$ (NOM$,2) <  > "JX"
    THEN N = N - 1                             BF02
815 GOTO 790                                   2F4B
820 PRINT D$;"CLOSE"                           C517
825 :                                          003A
830 REM * ABSENCE DE FICHIER *
835 :                                          003A
840 HOME                                       2F97
845 IF N = 0 THEN  VTAB 10: PRIN
    T S$: PRINT "ABSENCE DE FICH
    IER SUR CETTE DISQUETTE": GO
    SUB 700: RUN                               EB07
850 FOR K = 1 TO N: PRINT K;" ";
    "-";" "; MID$ (FICHIER$(K),4
    ,13): NEXT                                 FAE1
855 HTAB 1: VTAB 23: INPUT "Numé
    ro du fichier (0=menu):";CH:
    IF CH > N THEN 855                         93F1
860 IF CH = 0 THEN  RUN                        1CA8
865 FI$ =  MID$ (FICHIER$(CH),2,
    15)                                        3BA8
870 RETURN                                     63B1
```

François GALLET

LA SOURIS ET L'ASSEMBLEUR

PREMIÈRE PARTIE : Définitions

Nous sommes de plus en plus nombreux à utiliser et à apprécier ce périphérique merveilleux que représente *Dame Souris* . Malheureusement, si les récents logiciels apparaissant sur le marché en tirent parti, c'est rarement le cas pour l'amateur écrivant ses propres applications. En effet, celui-ci se contente très souvent du Basic, voire du Pascal, et il faut avouer que la gestion de la Souris n'y est pas des plus simples, ni des plus rapides (sauf peut-être en Pascal).

L'efficacité passe donc par le langage machine, mais voilà... comment faire ? Il faut reconnaître que la littérature traitant de ce sujet est des plus limitée, et souvent délicate à mettre en œuvre. Ce premier article vous familiarisera avec le principe de gestion de la Souris en Assembleur, les différents microprogrammes implantés dans la carte étant expliqués. Les articles suivants mettront en pratique ces explications par le biais de routines en Assembleur et vous permettront d'utiliser pleinement la Souris pour vos propres créations.

La première étape dans l'élaboration d'un programme, consiste évidemment à savoir si la carte Souris est présente, et si elle l'est, à connaître le connecteur où elle est implantée, afin de s'y référer d'une manière relative (ceci évitant de faire appel à un slot déterminé, pour le cas où l'on voudrait changer la carte de place).

Comment trouver cette carte ?...

Celle-ci possède 2 octets de signature, dont les valeurs hexadécimales $20 et $D6, sont implantées respectivement dans le système aux adresses $Cn0C et $CnFB, "**n**" allant de 1 à 7 (N° du slot).

Par exemple : La carte est dans le slot n°4 (normalement)
n = 4 d'où $Cn0C = $C40C doit contenir $20
$CnFB = $C4FB doit contenir $D6

Passez sous Moniteur pour le vérifier. La routine Machine balayant les slots à la recherche de la carte viendra dès la prochaine fois.

Une fois la carte trouvée, il reste à s'en servir. Un moyen pratique consiste à utiliser les interruptions de type IRQ (pour plus de détail sur celles-ci, consulter l'ouvrage *"PROGRAMMATION DU 6502"* de RODNAY ZAKS). La Souris en génère 60 par seconde, ce qui permet de passer autant de fois dans le programme de traitement.

Mais avant toute chose, il convient d'initialiser la Souris. En premier lieu, vous devez déterminer l'endroit en mémoire où se placera la routine de traitement. Une fois que c'est fait (ex. : $0300), il faudra placer l'octet haut de l'adresse ($03) en $3FF et l'octet bas ($00) en $3FE, afin d'initialiser le vecteur d'interruption, vecteur qui permettra au 6502 d'aller là où vous le désirez à chaque IRQ.

Ensuite, il faut mettre la Souris en service et préciser ce qui active les interruptions, en utilisant les microprogrammes appelés **INITMOUSE** et **SETMOUSE**.

INITMOUSE (code $19) : initialise les valeurs par défaut de la Souris et synchronise le sous-système avec le cycle d'extinction verticale de l'écran.

SETMOUSE (code $12) : au départ, la Souris est inactive et ses registres sont placés aux valeurs par défaut (voir *Initmouse*). Pour la rendre active, on utilise un octet, appelé *"octet de mode"*, dont la valeur, chargée dans le registre A et transmise à *Setmouse*, permettra la mise en service en précisant la cause de l'interruption.

Exemple :

```
LDY £$19      ; Code de la routine INITMOUSE.
JSR CALLCARD  ; Appel et exécution...
LDA £$XX      ; Octet de mode dans A.
LDY £$12      ; Code de la routine SETMOUSE
JSR CALLCARD  ; Appel et exécution...
...
```

(suite page 56)

Quelles valeurs peut prendre XX et quelles en sont les significations ?

Un octet se composant par définition de 8 bits, chacun de ces bits est utilisé comme suit :

Bit Ø* : met *Dame Souris* en fonction
Bit 1 : interruption sur déplacement de la Souris
Bit 2 : interruption sur enfoncement du bouton
Bit 3 : interruption à chaque rafraîchissement de l'écran (6Ø fois par seconde)
Bit 4 à 7 : réservés.

** (le plus à droite)*

Exemple : XX = $Ø9

Détaillons l'octet bit à bit : $Ø9 = ØØØØ1ØØ1

Le bit Ø valant 1, on met donc la Souris en fonction. Les bits 1 et 2 valant Ø, il n'y aura pas d'interruptions à cause d'un déplacement ou de l'appui du bouton. Par contre, le bit 3 étant à 1, on aura une interruption à chaque extinction verticale de l'écran, et donc 6Ø branchements à notre routine de traitement, ceci en une seule seconde.

Il est évident que toute combinaison peut être demandée (mettez tout de même le bit Ø à 1, afin d'avoir la Souris en service, sinon...). Vous pouvez activer les interruptions pour tout en même temps (pas vraiment utile) en mettant l'octet de mode à $ØF (valeur à ne pas dépasser, sous peine d'empiéter sur les bits réservés, ce qui conduirait SETMOUSE à vous retourner une erreur en mode illégal).

Il existe un mode, appelé *"mode passif"*, où l'octet de mode a la valeur $Ø1, et pour lequel il n'y a pas d'interruptions. Ce mode ne présente que peu d'intérêt (ce qui n'empêche pas de l'utiliser), car on est obligé de se brancher en permanence à une routine de lecture des registres de la Souris, ce qui pénalise la rapidité et l'occupation mémoire.

INITMOUSE et SETMOUSE ne sont pas les seuls micro-programmes implantés dans la carte. Il en existe d'autres. En voici la liste :

SERVEMOUSE (code $13) : la carte Souris n'étant pas, et de loin, la seule à envoyer des interruptions au 6502 parmi la multitude de cartes existantes sur le marché, Servemouse permet de déterminer si c'est la Souris qui en est à l'origine. Servemouse met à jour l'octet d'état ($778 + n) et positionne le Carry à Ø si la Souris est la cause de l'interruption, et à 1 si elle n'y est pour rien. Servemouse ne modifie pas les trous d'écran (voir plus loin).

READMOUSE (code $14) : c'est une routine importante. Elle vous servira à transférer les données en cours des registres de la Souris vers les trous d'écran. Elle positionne à Ø les bits 1, 2 et 3 de l'octet d'état ($778 + n).

CLEARMOUSE (code $15) : positionne à Ø les valeurs en X (abscisse) et Y (ordonnée) à la fois dans la carte Souris et dans les trous d'écran. L'octet d'état ($778 + n) est inchangé.

POSMOUSE (code $16) : positionne les registres de la carte Souris en fonction des valeurs X et Y trouvées dans les trous d'écran.

HOMEMOUSE (code $18) : positionne les registres de la carte Souris à leur valeur la plus basse. Les trous d'écran ne sont pas modifiés. Si vous désirez mettre les trous d'écran aux valeurs correspondantes, faites suivre *Homemouse* par *Readmouse.*

CLAMPMOUSE (code $17) : le microprogramme important par excellence. C'est lui qui fixe les valeurs limites par défaut des positions en X et Y. Au départ ces valeurs sont $ØØØØ et $Ø3FF, soit une excursion de Ø à 1Ø23 en X aussi bien qu'en Y. Si vous voulez modifier ces limites et je parie que vous le ferez voici comment procéder :

- mettez d'abord l'accumulateur à la valeur correcte selon que vous désirez modifier les limites de X (abscisse) ou Y (ordonnée) :
  A = Ø, les limites des coordonnées X sont modifiées
  A = 1, les limites des coordonnées Y sont modifiées
- les nouvelles limites seront prélevées dans les trous d'écran ou vous les aurez placées auparavant :
  $478 : Trou d'écran contenant l'octet le moins significatif de la limite inférieure.
  $578 : Trou d'écran contenant l'octet le plus significatif de la limite inférieure.
  $4F8 : Trou d'écran contenant l'octet le moins significatif de la limite supérieure.
  $5F8 : Trou d'écran contenant l'octet le plus significatif de la limite supérieure.

Exemple : Imaginons un X variant de Ø à 79 et un Y de Ø à 23.

MODIFICATION DES LIMITES EN X

```
LDA £$ØØ      ; Ø dans ...
STA $478      ; ...
STA $578      ; ...
STA $5F8      ; ...
LDA £$4F      ; 79 dans ...
STA $4F8      ; ...
LDA £$ØØ      ; A = Ø, pour modifier les limites de
                l'abscisse
LDY £$17      ; Code de la routine CLAMPMOUSE
JSR CALLCARD  ; Appel et exécution
...
```

MODIFICATION DES LIMITES EN Y

```
LDA £$ØØ      ; Ø dans ...
STA $478      ; ...
STA $578      ; ...
STA $5F8      ; ...
LDA £$17      ; 23 dans ...
STA $4F8      ; ...
LDA £$Ø1      ; A = 1, pour modifier les limites de
                l'ordonnée
LDY £$17      ; Code de la routine CLAMPMOUSE
JSR CALLCARD  ; Appel et exécution.
...
```

Bien entendu, ceci est correct en théorie, mais en pratique, vous bougeriez à peine votre Souris que le pointeur serait déjà en fin d'écran. En fait, il faut établir les limites beaucoup plus loin en prenant des multiples de 79 et de 23 et déterminer un coefficient de réduction ramenant aux bornes de l'écran, ce qui permet de moduler la sensibilité dans le déplacement. Par exemple, on peut établir les limites en X à 632 et Y à 115, ce qui donne des coefficients respectifs de 8 et de 7. Cela signifie que l'on aura 1 seul déplacement en X à l'écran pour 8 déplacements en X par la Souris, et 1 en Y à l'écran pour 7 par la Souris.

ATTENTION : CLAMPMOUSE détruisant le contenu des trous d'écran de position en X et en Y ($478 + n, $4F8 + n, $578 + n et $5F8 + n), il suffit de la faire suivre par READMOUSE pour les restaurer.

Que sont les trous d'écran ?

Il s'agit en fait de cases mémoire situées dans la zone réservée à l'écran. Cette zone, de 1 ko, n'est pas totalement utilisée pour l'affichage de données, car 960 octets seulement sont nécessaires (24 lignes × 40 colonnes). Il reste donc de la place disponible pour stocker des paramètres, place réservée en général pour les cartes d'extension.

QUELS SONT CEUX UTILISÉS PAR LA SOURIS ?

$478 + n : Octet le moins significatif de la position en X.
$4F8 + n : Octet le moins significatif de la position en Y.
$578 + n : Octet le plus significatif de la position en X.
$5F8 + n : Octet le plus significatif de la position en Y.
$678 + n : Réservé.
$6F8 + n : Réservé.
$778 + n : Etat du bouton et des interruptions.
$7F8 + n : Mode en cours.

"n" étant le numéro du connecteur où la carte Souris est implantée. Si, par exemple, elle se trouve dans le slot 4, l'octet le moins significatif de la position en X sera stocké en $478 + 4 soit $47C.

Quelques explications sur l'octet réservé à l'état du bouton et des interruptions ($778 + n).

A chaque interruption, alors que le 6502 vient se brancher sur votre routine de traitement des déplacements de la Souris, il est plus que probable que vous aurez à tester votre Souris. L'octet d'état vous donnera des informations que vous n'aurez plus qu'à interpréter en fonction de vos désirs. Voici ces renseignements :

Bit 0* : Réservé.
Bit 1 : Interruption causée par le mouvement (X ou Y).
Bit 2 : Interruption sur enfoncement du bouton.
Bit 3 : Interruption due au rafraîchissement de l'écran (60 fois par seconde).
Bit 4 : Réservé.
Bit 5 : X ou Y ont varié depuis la dernière lecture.
Bit 6 : Le bouton était enfoncé lors de la dernière lecture.
Bit 7 : Le bouton est actuellement enfoncé.

** (le plus à droite)*

Bien entendu, chacune de ces propositions n'est vraie que lorsque le bit correspondant est à 1. Les bits 6 et 7 permettent, en associant leurs tests, de déterminer un relachement ou un enfoncement du bouton.

Voilà, il est maintenant temps de nous quitter, car la prochaine fois commenceront les choses sérieuses avec les routines en Assembleur. Je vous conseille de réviser les mnémoniques, leurs fonctions, ainsi que leurs modes d'adressage.

François GALLET

Yvan KŒNIG

vous répond...

• Lecteur passionné de Tremplin Micro, je voudrais savoir si le logiciel Toolkit dont vous faites si souvent mention est bien livré avec l'imprimante Imagewriter II (je vais en acheter une) et, dans le cas contraire, quel est son prix ?
Est-on obligé d'avoir "ProDOS" sur son Apple pour utiliser "ProCODE" ? Et peut-on se procurer "ProCODE" à un prix raisonnable ? Et y a-t-il des assembleurs fonctionnant sous DOS 3.3. et efficaces ?

D.P. (66000 PERPIGNAN)

RÉPONSE D'Y. KŒNIG :

Il existe deux disquettes TOOLKIT, imprimante dans la gamme APPLE *II*. Le Toolkit livré avec l'IMW1 était en DOS 3.3 et c'est à lui qu'il est souvent fait allusion dans *Tremplin Micro* (il y en a peut-être eu une version ProDOS lors de l'introduction du *IIc*).

A ma connaissance, il n'est plus distribué officiellement depuis la disparition de l'IMW1 mais votre distributeur devrait pouvoir vous en faire une copie.

Pour l'IMW2 il existe un Toolkit sous ProDOS. APPLE ne le fournit pas d'office car cette imprimante n'est pas spécifique à la gamme APPLE *II*. Votre distributeur, s'il est agréé, **DOIT** vous en faire une copie sur une disquette que vous lui fournirez.

Vous pouvez demander à votre distributeur de vous fournir ProDOS, il s'agit alors du produit "Kit utilisateur ProDOS" réf. F2D-2010.

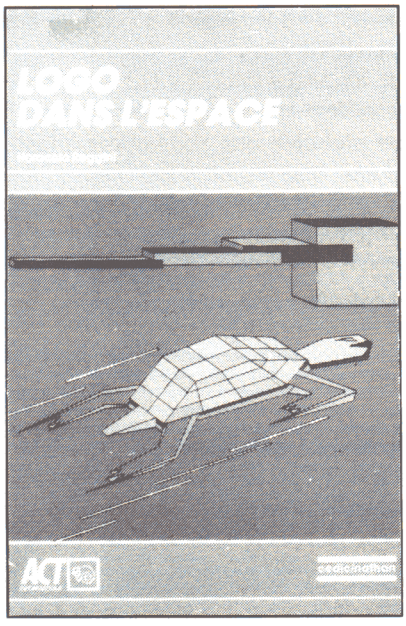
Quant à ProCODE, il forme un tout et comprend notamment le système ProDOS.

ProCODE est vendu aux environs de 1000 francs et c'est le seul produit français (Version.Soft).

Si l'utilisation d'un assembleur en Anglais ne vous gêne pas, le choix est large. Mon choix personnel est MERLIN dans ses diverses versions. ■

Votre bibliothèque INFORMATIQUE

par NESTOR



• LOGO DANS L'ESPACE
Horacio Reggini adapté de l'espagnol par Act Informatique

Il ne faut pas se demander si les outils techniques conditionnent les manières de vivre, mais plutôt réfléchir sur les immenses possibilités que ces outils offrent aux individus... pour pouvoir développer leur intuition, raisonner, décider, agir. Les outils techniques ne limitent pas l'activité humaine bien au contraire, ils peuvent élargir son champ et l'enrichir s'ils sont correctement employés. L'outil technique utilisé dans ce livre est l'ordinateur, la modalité d'utilisation est Logo et l'activité en jeu est la description et la réalisation de formes géométriques spatiales.

Ce livre invite le lecteur à participer activement au travail et au rôle de "constructeur de formes", qu'il soit artiste — peintre ou sculpteur —, artisan — menuisier ou forgeron —, architecte ou ingénieur, en lui suggérant une excursion intellectuelle dans le domaine de la création et du maniement des formes tridimensionnelles. En apprenant à l'ordinateur à produire des objets tridimensionnels, le lecteur acquiert un plus grand discernement et une meilleure compréhension de l'élégance et de la complexité des formes dans l'espace. L'ouvrage est organisé en ETUDES ou "exercices de l'esprit" qui montrent au lecteur l'utilisation des commandes permettant de résoudre des cas simples et intéressants. Au sommaire on trouvera de nombreux objets, parmi lesquels : l'escalier en colimaçon, la bicyclette, l'immeuble, la lampe, les engrenages, la cage, l'oignon, le moulin à vent, bref, une variété de thèmes pour un large public... Il est donc possible, à partir du langage Logo, de faire décoller la tortue et de lui faire tracer sur l'écran des objets en trois dimensions. Il suffit simplement d'enrichir le langage de procédures dont la liste est donnée en fin d'ouvrage. Il devient alors très simple d'imaginer et de commander des déplacements de la tortue dans l'espace. Ce livre propose de nombreux exemples qui permettront de diversifier les activités de programmation en Logo en leur donnant une dimension supplémentaire.

CEDIC/NATHAN — Idées et formes — 6, boulevard Jourdan, 75014 PARIS
Tél. (1) 45.65.06.06. 160 pages — 115 F ttc.

• PRATIQUE DE LA MISE EN PAGE SUR MACINTOSH
(A. Garnier et J.-L. Van Impe)

Cet ouvrage est destiné à tous ceux qui débutent en la matière et pour lesquels la solution offerte par Apple (Macintosh et LaserWriter) est actuellement une des plus simples à mettre en œuvre.

A travers un panorama des principaux logiciels de traitement de texte et de mise en page, les auteurs ont choisi de présenter, à l'aide d'exemples concrets, les manipulations de base qui permettront à chacun de réaliser des documents, simples ou complexes, et de maîtriser cet outil de travail.

EYROLLES — 61, bd Saint-Germain, 75240 PARIS
Tél. (1) 46.34.21.99. 160 pages — 200 F ttc.

• SUPERFORCE (Paul Davies) (traduit par A. Bouquet)

Vous arrive-t-il de lire et, ce faisant, de vous intéresser aux grandes questions scientifiques de notre époque ? Je suis sûr que oui. La micro-informatique mène à tout, c'est bien connu, à condition d'en sortir !

Pendant trente ans, Albert Einstein a poursuivi un rêve : mettre sur pied une théorie unifiée (des champs) qui puisse, en un cadre unique, englober l'infiniment petit et l'infiniment grand. Hélas ! malgré une belle obstination, il n'a jamais été en mesure de concrétiser son grand projet. Les développements actuels de la physique sont peut-être en train de réaliser le vieux rêve du père de la Relativité. Dans les infimes fractions de temps qui ont suivi le Big Bang, l'Univers, dans son entier, aurait été sous la domination exclusive d'une force unique, une *Superforce.* Suivez Paul Davies et vous en saurez davantage sur l'histoire et le détail théorique du projet des théoriciens : réussir l'unification des quatre forces fondamentales qui régissent toute chose : la gravitation, l'électromagnétisme et les deux interactions, fort et faible.

PAYOT, 106, bd Saint-Germain, 75006 PARIS
324 pages — 140 F ttc.

INITIATION | DOS | ProDOS

Fonctionne aussi sur l'APPLE IIGS

Clément RENARD

DÉSASSEMBLAGE

S'il est facile, par programme, de désassembler A instructions (pourquoi **A** ? parce que le nombre d'instructions à traiter était habituellement mis dans l'**A**ccumulateur), la même opération pose un problème sur l'Apple *IIGS*. En effet, jusqu'à plus ample informé, il apparaît que celui-ci ignore la routine $FE63 (LIST2), qui se chargeait de tout le travail.

Heureusement, il reste possible, en utilisant $F8D0 (INSTDSP) de désassembler une instruction. Il suffit alors de s'offrir un petit compteur, puis d'incrémenter le PC ($3A-$3B) de la longueur de l'instruction que l'on vient d'afficher (on trouve celle-ci en $2F... mais elle est égale à L - 1) pour remplacer LIST2 défaillante.

La démonstration ci-après permet de visualiser les trois méthodes (la routine 65C02 et GS est plus courte d'un octet car elle utilise $1A... qui incrémente l'accumulateur). N'oubliez pas les virgules, à la fin des lignes 260 et 270 (16 pour la première et 1 pour la seconde).

DESASS.BAS

```
100 TEXT : PRINT CHR$ (21): HOME
110 VTAB 12: PRINT "DESASSEMBLAGE DE A INSTRUCTIONS"
120 PRINT "Routine Apple // 6502-65C02 et GS": GOSUB 220
130 D = 812: GOSUB 230: REM Apple // 6502-65C02 et GS
140 HOME : VTAB 12: PRINT "ANCIENNE ROUTINE COURTE": PRINT : PRINT "utilisant $FE63": PRINT "DESASSEMBLAGE DE A INSTRUCTIONS": GOSUB 220
150 GOSUB 230: REM Routine 6502-65C02 (de 768 à 796)
160 HOME : VTAB 12: PRINT "ROUTINE POUR APPLE //-65C02 OU GS": PRINT "AFFICHAGE AVEC ROUTINE 1 ET CALL 8192"
170 PRINT : INVERSE : PRINT "1A";: NORMAL : PRINT " INCREMENTE L'ACCUMULATEUR"
180 GOSUB 220: GOSUB 240
190 HOME : VTAB 12: PRINT "(M)ENU DE DISQUETTE ";: CALL - 198: GET R$
200 PRINT : IF R$ = "M" THEN PRINT CHR$ (4)"RUN MENU,D1"
210 HOME : END
220 CALL - 198: POKE 49168,0: WAIT 49152,128,127: POKE 49168,0: PRINT : RETURN
230 FOR I = 768 TO D: READ R: POKE I,R: NEXT : CALL 768: RETURN
240 FOR I = 768 TO D: READ R: POKE I,R: NEXT : RESTORE : FOR I = 8192 TO 8243: READ R: POKE I,R : NEXT : CALL 8192: RETURN
250 DATA 169,24,133,6,160,0,162,3,134,59,132,58,32,208,248,24,230,47,165,47,101,58,168,144,2,230,59,198,6,208,235,44,16,192,173,0,192,16,251,44,16,192,76,88,252
260 DATA 169,24,133,6,160,0,162,3,134,59,132,58,32,99,254,44,16,192,173,0,192,16,251,44,16,192,76,88,252,,,,,,,,,,,,,,,
270 DATA 169,24,133,6,160,0,162,3,134,59,132,58,32,208,248,24,165,47,26,101,58,168,144,2,230,59,198,6,208,236,44,16,192,173,0,192,16,251,44,16,192,76,88,252,
```

DESAS.6502-65C02

```
300 : A9 18     LDA $18
302 : 85 06     STA $06
304 : A0 00     LDY £$00
306 : A2 03     LDX £$03
308 : 86 3B     STX $3B
30A : 84 3A     STY $3A
30C : 20 63 FE  JSR $FE63
30F : 2C 10 C0  BIT $C010
312 : AD 00 C0  LDA $C000
315 : 10 FB     BPL $0312
317 : 2C 10 C0  BIT $C010
31A : 4C 58 FC  JMP $FC58
```

Nombre d'instructions à désassembler dans A.

Adresse de départ dans $3A-$3B.

LIST2 fait tout le travail.

Attente d'une touche et HOME.

ATTENTION !
Ne pas utiliser $FE63 sur le nouvel Apple *IIGS !*

DESAS.GS (et 65C02)

```
00/0300 : A9 18     LDA £18
00/0302 : 85 06     STA 06
00/0304 : A0 00     LDY £00
00/0306 : A2 03     LDX £03
00/0308 : 86 3B     STX 3B
00/030A : 84 3A     STY 3A
00/030C : 20 D0 F8  JSR F8D0
00/030F : 18        CLC
00/0310 : A5 2F     LDA 2F
00/0312 : 1A        INC
00/0313 : 65 3A     ADC 3A
00/0315 : AB        TAY
00/0316 : 90 02     BCC 031A
00/0318 : E6 3B     INC 3B
00/031A : C6 06     DEC 06
00/031C : D0 EC     BNE 030A
00/031E : 2C 10 C0  BIT C010
00/0321 : AD 00 C0  LDA C000
00/0324 : 10 FB     BPL 0321
00/0326 : 2C 10 C0  BIT C010
00/0329 : 4C 58 FC  JMP FC58
```

Nombre d'instructions dans A et dans compteur $6

Adresse de départ (partie basse et partie haute).

PC = adresse de départ

INSTDSP désassemble une instruction.
Annulation de la retenue (obligée).
La longueur de l'instruction (- 1) est dans $2F.
A = A + 1 pour longueur vraie.
Partie basse PC.
A passe dans Y.
Saut si pas de retenue.
Sinon partie haute PC incrémentée.

Compteur décrémenté et suite jusqu'à 1 inclus.

On attend une touche pour nettoyer l'écran.

Jean-Noël GORGE
prend la direction du marketing chez APPLE COMPUTER FRANCE

Depuis 17 ans dans l'informatique, Jean-Noël GORGE est devenu — le 1er décembre — Directeur du marketing d'APPLE COMPUTER FRANCE.

Après avoir passé 6 ans chez Bull, 6 ans chez Cap Gemini Sogeti, entrecoupé de 5 ans chez Digital Equipement, il apporte avec lui une solide expérience marketing bien sûr, mais également une expérience du terrain où il a régulièrement exercé des responsabilités importantes.

Sous l'autorité directe de Jean CALMON, Directeur Général, Jean-Noël GORGE assume la direction du département qui regroupe la publicité, les relations presse, les relations publiques, la promotion, le développement et le marketing des produits, ainsi que les relations directes avec les utilisateurs par l'intermédiaire du club Apple.

Originaire de La Rochelle, HEC 1969, père de 3 garçons, Jean-Noël GORGE est un passionné de rugby, de voile et de bricolage. ■

EFFETS GRAPHIQUES

Petite routine permettant de jolis effets de barres parallèles verticales... surtout quand on possède un écran couleur.
Pour un arrêt momentané, maintenir la touche POMME OUVERTE enfoncée, puis pressez n'importe quelle touche pour continuer.

Que fait $F3D8 ?

```
F3D8 : 2C 55 C0    BIT $C055      PAGE 2
F3DB : 2C 52 C0    BIT $C052      GRAPHIQUE
F3DE : A9 40       LDA £$40       40 dans A et
F3E0 : D0 08       BNE $F3EA      saut
F3E2 : A9 20       LDA £$20
F3E4 : 2C 54 C0    BIT $C054      Concerne HGR
F3E7 : 2C 53 C0    BIT $C053
F3EA : 85 E6       STA $E6        $40 dans $E6
F3EC : AD 57 C0    LDA $C057      Haute résolution
F3EF : AD 50 C0    LDA $C050      graphique
F3F2 : A9 00       LDA £$00       0 dans A
F3F4 : 85 1C       STA $1C        et $1C
F3F6 : A5 E6       LDA $E6        $40 (ou $20)
F3F8 : 85 1B       STA $1B        dans $1B
F3FA : A0 00       LDY £$00       0 dans $1A
F3FC : 84 1A       STY $1A
F3FE : A5 1C       LDA $1C        La zone
F400 : 91 1A       STA ($1A),Y    $4000 à 5FFF
F402 : 20 7E F4    JSR $F47E      (ou $2000 à
F405 : C8          INY            $3FFF dans le
F406 : D0 F6       BNE $F3FE      cas de $F3E2,
F408 : E6 1B       INC $1B        c'est-à-dire
F40A : A5 1B       LDA $1B        HGR) est ini-
F40C : 29 1F       AND £$1F       tialisée (tous
F40E : D0 EE       BNE $F3FE      les octets à 0).
F410 : 60          RTS
```

EFF.HGR.B

```
100 PRINT CHR$ (21): HOME : GOSUB 170
110 FOR I = 0 TO 255: POKE 772,I : CALL 768:
    FOR J = 1 TO 15: BZZ = PEEK (49200)
120 IF PEEK (49249) > 127 THEN GOSUB 180
130 NEXT : NEXT
140 GOSUB 180: TEXT
150 VTAB 22: PRINT "(M)ENU DE DISQUETTE ";: GET R$: PRINT : IF R$ = "M"
    OR R$ = "m" THEN PRINT CHR$ (4)"RUN MENU,D1"
160 HOME : END
170 FOR I = 768 TO 791: READ R: POKE I,R:
    NEXT : RETURN
180 CALL - 198: POKE 49168,0: WAIT 49152,
    128,127: POKE 49168,0: RETURN
190 DATA 32,216,243,169,255,162,32,160,0,132,
    229,145,229,200,192,248,144,249,230,230,
    202,208,240,96
```

EFF.HGR2

```
300 : 20 D8 F3    JSR $F3D8      HGR2.
303 : A9 FF       LDA £$FF       Valeur poquée à partir du Basic.
305 : A2 20       LDX £$20       X et $20.
307 : A0 00       LDY £$00       Y = 0 pour initialiser $E5.
309 : 84 E5       STY $E5
30B : 91 E5       STA ($E5),Y    On dépose la valeur de A en $E5-E6 + Y.
30D : C8          INY            Y = Y + 1.
30E : C0 F8       CPY £$F8       On va jusqu'à $F7 inclus.
310 : 90 F9       BCC $0307
312 : E6 E6       INC $E6        Adresse haute majorée.
314 : CA          DEX            Pointeur décrémenté.
315 : D0 F0       BNE $0307      Jusqu'à 1 inclus, on continue.
317 : 60          RTS            Basic.
```

■

Votre bibliothèque INFORMATIQUE

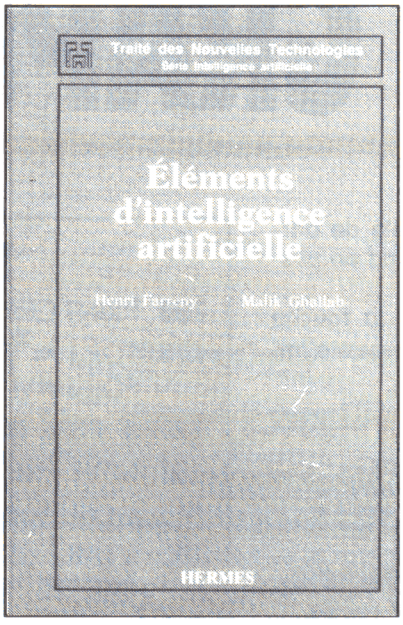
par NESTOR

• MACASTUCES (2e édition) par Hervé Thiriez

Hervé Thiriez, professeur au centre HEC-ISA, est par ailleurs le distingué rédacteur en chef de notre confrère POM'S... lequel s'est toujours intéressé à l'enfant chéri d'Apple : le célèbre Mac. Hervé Thiriez est donc particulièrement qualifié pour aider les utilisateurs de la bête à en tirer le meilleur parti, même si — convivialité oblige — utiliser des astuces paraît superflu avec une machine aussi performante.

Mais il y a ce que l'on sait et ce que l'on ignore. La plupart des gens lisant peu ou mal la documentation concernant leur matériel, il est certain que la majorité des personnes travaillant sur Macintosh n'en connaissent pas la moitié des ressources. L'ouvrage d'Hervé Thiriez leur permettra de les découvrir en moins de 300 pages illustrées par de nombreux écrans. Les moins courageux pourront d'ailleurs s'offrir une disquette d'accompagnement que nous n'avons pas été en mesure de tester.

EDITIONS DU P.S.I.
BP 86, 77402 LAGNY s/MARNE
288 pages — 160 F ttc.



• ÉLÉMENTS D'INTELLIGENCE ARTIFICIELLE (H. Farreny et M. Ghallab)

Henri Farreny enseigne l'intelligence artificielle depuis une douzaine d'années. Malik Ghallab, chargé de recherche au CNRS, est responsable de l'activité IA au sein du groupe Robotique et intelligence artificielle du Laboratoire d'automatique et d'analyse des systèmes. La compétence des auteurs est reconnue et ce n'est pas par hasard que leur livre constitue l'un des volets d'une série très sérieuse : *Traité des Nouvelles Technologies, Série Intelligence artificielle.* Et c'est bien d'un traité qu'il s'agit, complet et structuré, bourré d'exemples et de remarques, illustré aussi souvent qu'il le faut, sérieux comme il convient à tout ouvrage de cette nature, mais pas du tout rébarbatif.

Je ne crois pas utile d'en détailler la table des matières : c'est un survol méthodique de toutes les questions suscitées par des initiales un peu magiques : IA... comme intelligence artificielle. On n'explique pas un livre comme celui-ci : il est de loin préférable de l'ouvrir et d'en commencer la lecture, crayon en main. Félicitations aux Editions Hermès pour la présentation sobre et solide de ce traité. Un bon investissement que ces 368 pages de cours magistral !

EDITIONS HERMES, 51, rue Rennequin, 75017 PARIS
368 pages sous couverture cartonnée — 240 F ttc.

• LES SECRETS DE VOTRE MACINTOSH Equipe Soft-Loisirs

L'avertissement au lecteur est explicite : "Ce manuel contient uniquement des informations pratiques et concrètes". C'est vrai. SOFT-LOISIRS diffuse des logiciels du domaine public moyennant le versement d'une somme très raisonnable (50 F pour ne rien vous cacher). En vérité, c'est un peu plus compliqué que cela, mais je suis persuadé que les animateurs de Soft-Loisirs se feront un plaisir de vous fournir une documentation détaillée sur un système de diffusion tout à fait original. En tout cas, la brochure que j'ai sous les yeux (je ne suis pas un spécialiste du Macintosh) me paraît intéressante. Elle ne compte que 48 pages, mais fournit un maximum de renseignements sur les accessoires de bureau, le bureau, le Finder, MacWrite, MS.Word, MacPaint, MacDraw, Page Maker, etc. Elle nous prouve aussi qu'il est facile, avec un Mac et une LaserWriter, de concevoir une brochure prête à imprimer.

SOFT-LOISIRS, 4, avenue Jean-Jaurès, 94200 IVRY/SUR/SEINE

TURBO PASCAL

est disponible sur le MAC

À quand une version APPLE IIGS ?

Quand on commence à découvrir le nouvel Apple *IIGS*, on comprend très vite que, décidément, tout ce qui concerne le Macintosh pourrait trouver des prolongements sur le *GS*. On peut en tout cas le souhaiter quand on constate l'extraordinaire rapidité de compilation du TURBO PASCAL de BORLAND INTERNATIONAL.

Là, au moins, on ne se moque pas du monde ! TURBO PASCAL comprend non seulement les deux indispensables disquettes 3, 5 pouces (copiables pour votre usage personnel), mais une vraie documentation, avec de très nombreux exemples. Pour l'instant, c'est encore en anglais, mais la version française arrive et l'échange sera peut-être possible au moment où paraîtront ces lignes. Bravo ! BORLAND !

OUI, TURBO PASCAL POUR LE MAC EST DISPONIBLE !

L'utilisateur peut compiler et exécuter jusqu'à huit programmes différents — ou huit versions du même programme — en même temps lors du développement en Turbo Pascal. La possibilité d'observer le fonctionnement de plusieurs programmes ou de diverses versions du même programme, dans des fenêtres séparées, permet d'accélérer le développement et d'améliorer l'efficacité des programmeurs.

Turbo Pascal pour le Mac dispose d'instructions spéciales pour la gestion d'une tortue graphique. Les nouvelles possibilités graphiques permettent une manipulation plus facile des objets sur l'écran du Mac. Le concept de tortue graphique découle d'une idée de Seymour Papert et de ses collaborateurs, au Massachussets Institute of Technology (MIT). Papert voulait simplifier la création de graphiques pour ceux qui sont peu habitués aux "coordonnées cartésiennes" standard. Il inventa donc une "tortue" capable d'"avancer" d'une certaine distance sur l'écran et de tourner d'un angle donné, en laissant une trace de son passage.

La tortue graphique opère dans une fenêtre occupant la totalité de l'écran, d'une façon assez semblable à ce que font les programmes de Quickdraw.

Les procédures de gestion de la fenêtre de la tortue peuvent faire, de n'importe quelle partie de l'écran du Mac, la zone active, en prévenant l'effacement du reste de l'écran.

La tortue peut aussi être utilisée en combinaison avec Quickdraw.

Les instructions de gestion de la tortue pour le Mac comprennent PenUp (Lèvecrayon), PenDown (Baissecrayon), Forwd (Avance), Back (Recule), Heading (Fixecap), TurnLeft (Gauche), TurnRight (Droite) et TurtleWindow (Fenêtre).

L'ensemble Turbo Pascal pour le Mac comprend aussi des exemples concernant le programme de gestion du réseau local AppleTalk, le système de synthèse de parole Macintalk, une pendule accessoire du bureau, un programme d'impression, et bien d'autres encore. Tous les programmes exemples sont fournis complets avec leur code source.

Pour MAC, BORLAND INTERNATIONAL offre également un gestionnaire de bureau électronique, SideKick et un système de gestion de base de données relationnelle : Reflex pour le Mac.

TURBO PASCAL pour le Mac est disponible sur le marché français et il est vendu au prix de 995 francs hors taxes jusqu'au mois de mars 1987. *(Suite page 64).*

TURBO PASCAL pour le Mac : fiche technique

Le système de développement Turbo Pascal pour le Mac utilise toutes les possibilités et caractéristiques de l'environnement Mac, y compris la structure hiérarchique des fichiers. Il dispose de menus déroulants et de boîtes de dialogue guidant l'utilisateur à travers les choix et les options. Les caractéristiques principales de Turbo Pascal pour le Mac sont :

- Vitesse de compilation de 12000 lignes par minute.
- Une "Unité de structure", où la compilation séparée de sous-programmes crée des modules ou unités, qui peuvent être liés à n'importe quel programme en Turbo Pascal. Cette caractéristique unique permet de créer des programmes modulaires, ainsi que des bibliothèques de sous-programmes faciles à intégrer dans les projets plus importants. Grâce à sa structure modulaire, Turbo Pascal pour le Mac utilise efficacement la mémoire du Mac, et diminue donc l'espace mémoire nécessaire pour exécuter des programmes importants.
- Fenêtres d'édition multiples, permettant de passer facilement d'une fenêtre à l'autre, et de modifier, compiler ou exécuter séparément le contenu de chacune des fenêtres.
- Options de compilation, présentées sur un menu, et permettant de choisir la compilation sur disque ou en mémoire, ou la compilation suivie immédiatement d'exécution. Les étapes de développement et de mise au point sont très fortement accélérées, puisqu'il n'y a plus à changer de programme pour compiler, exécuter ou modifier un programme en développement.
- Compatibilité avec le Pascal Lisa ; les sous-programmes créés par chacun des systèmes de développement peuvent être compilés et exécutés par l'autre avec un minimum de modifications.
- Compatibilité avec le système hiérarchique de gestion des fichiers du Mac. De plus, il est possible de préciser le volume et le dossier par défaut pour les fichiers de compilation et simplifier la modification des sous-programmes.
- Capacité à utiliser toute la mémoire disponible sur le Mac, sans aucune limite.
- Possibilité d'appeler les sous-programmes de la boîte à outils du Mac.

JEU VISUEL

INITIATION

OBSERVATION

Jeu visuel sans prétention, mais utilisant les ressources d'un écran couleur éventuel. Il fait appel à une table de formes qui ne contient que les codes d'un carré. Le principe du jeu est simple et on aura tout compris après avoir lu les lignes 600 et 610... puis tapé un RUN expérimental. On notera que le programme n'interroge pas le joueur et ne mémorise donc pas telle ou telle réponse. On se contente d'observer, puis de comparer le résultat de ces observations à la solution fournie par le micro. Tel qu'il se présente, ce jeu est donc individuel, mais il est facilement modifiable. Ne vous privez donc pas du plaisir de l'améliorer !

ROT donne la position angulaire de la forme tracée par DRAW (0 à 255). On peut l'utiliser en mode direct.

SCALE est l'échelle de la forme (vecteur*s). Va de 1 à 255, mais 0 correspond à l'agrandissement maximum.

HCOLOR permet de définir la couleur. On compte 8 couleurs, dont 2 blancs et 2 noirs. Les couleurs varient suivant que la colonne est **PAIRE** ou **IMPAIRE**.

PAIRE	IMPAIRE
1 VIOLET	2 VERT
5 BLEU	6 ROUGE
0 et 4 =	NOIR
3 et 7 =	BLANC

LOMEM devra être fixé à 16384 (au-dessus de la page HGR) si vous désirez utiliser le jeu un grand nombre de fois. Tapez une ligne :
90 LOMEN:16384

```
100 TEXT : NORMAL : PRINT  CHR$ (21): HOME : R
    OT= 0: GOSUB 550                          C346
110 HGR : HCOLOR= 1                           F58E
120 :                                         003A
130 REM Tracé de tous les carrés    *****
140 :                                         003A
150 POKE 49234,0: SCALE= 60                   C654
160 DRAW 1 AT 65,65                           0A8C
170 S1 = 19 +  INT ( RND (1) * 3): REM Coté va
    riable                                    3504
180 SCALE= S1: HCOLOR= 5                      1A1E
190 FOR I = 1 TO 5: DRAW 1 AT (50 * I) - S1 +
    7,160: NEXT                               6619
200 I = 1 +  INT ( RND (1) * 3): RESTORE : FOR
    J = 1 TO I: READ S: NEXT                  EA58
210 D = 1: SCALE= S:H = 2: IF S = 40 THEN D =
    0                                         5A65
220 HCOLOR= H                                 FFDA
230 DRAW 1 AT 200 + D,81                      A3BD
240 HOME : VTAB 23: PRINT "Le grand carré est
    ";                                        6157
250 IF S > 40 THEN  PRINT " TROP GRAND";: GOTO
    280                                       C0A0
260 IF S < 40 THEN  PRINT " TROP PETIT";: GOTO
    280                                       87BC
270 PRINT " CORRECT";                         776B
280 VTAB 24: HTAB 1: PRINT "Les petits carrés
    sont ";                                   8FF3
290 IF S1 > 20 THEN  PRINT " trop grands";: GO
    TO 330                                    421E
300 IF S1 < 20 THEN  PRINT " trop petits";: GO
    TO 330                                    193A
310 PRINT " corrects";                        2CBE
320 :                                         003A
```

```
330 GOSUB 460: POKE 49235,0: GOSUB 460: GOSUB
    470: POKE 49234,0: GOSUB 460: POKE 49235,0   ECD8
340 :                                            003A
350 REM Encore un tour, ou terminé  *****
360 :                                            003A
370 HOME : VTAB 23: PRINT "(S)UITE "(T)ERMINEE
     "(M)ENU DISQUETTE ";: CALL  - 198: GET R$   D56D
380 IF R$ = "M" THEN  TEXT : HOME : PRINT : PR
    INT CHR$ (4)"RUN MENU,D1"                    6125
390 IF R$ = "S" THEN 420                         C9E4
400 IF R$ <  > "T" THEN 370                      A9B9
410 TEXT : HOME : END                            5514
420 HCOLOR= 6: HPLOT 0,0: CALL  - 3082: GOTO 1
    10                                           64F4
430 :                                            003A
440 REM Sous-programmes d'affichage *****
450 :                                            003A
460 CALL  - 198: POKE 49168,0: WAIT 49152,128,
    127: POKE 49168,0: RETURN                    B419
470 SCALE= S1: HCOLOR= 1:D = S1 - 20: FOR I =
    1 TO 3: DRAW 1 AT S1 + 5,((40 + D) * I) -
    15: NEXT                                     C191
480 FOR I = 2 TO 3: DRAW 1 AT ((40 + D) * I) -
    15,105 + D: NEXT                             9D62
490 IF S = 40 THEN H = 1                         C241
500 SCALE= S: HCOLOR= H: DRAW 1 AT 85,45         C4C6
510 RETURN                                       63B1
520 :                                            003A
530 REM Table de formes (le carré!) *****
540 :                                            003A
550 POKE 768,1: POKE 769,0: POKE 770,4: POKE 7
    71,0: POKE 772,33: POKE 773,63: POKE 774,5
    4: POKE 775,45: POKE 776,28: POKE 777,0: P
    OKE 232,0: POKE 233,3                        9723
560 :                                            003A
570 REM Petite présentation du jeu  *****
580 :                                            003A
590 VTAB 6: HTAB 14: INVERSE : PRINT "  "
    "": FOR I = 7 TO 9: VTAB I: HTAB 14:
    PRINT " ";: HTAB 28: PRINT " ": NEXT : HTA
    B 14: PRINT "  "": NORMAL : VT
    AB 8: HTAB 16: PRINT "OBSERVATION"            1D30
600 VTAB 15: PRINT "Cinq petits carrés et un p
    lus grand vontfigurer près d'un septième..
    .dans lequelils doivent exactement s'encas
    trer."                                       B10B
610 PRINT : PRINT "Hélas! toutes les pièces du
     puzzle, saufle cadre original, peuvent se
     révéler outrop grandes, ou trop petites,
    ou encorecorrectes. A VOUS D'Y VOIR CLAIR!
    "                                            7980
620 GOSUB 460: RETURN                            8235
630 :                                            003A
640 DATA 37,40,43 "                              5650
```

DRAW va dessiner toute forme dont on lui fournit le numéro. Il est indispensable de renseigner la machine sur l'adresse où se trouve la table de formes ($E8-$E9). C'est ce que font les deux derniers POKE de la ligne 550.

RND fournit un nombre aléatoire. Pour obtenir un nombre compris entre 0 et 5, on utilise la formule **AL = INT(RND(1) * 6).**

Pour obtenir un nombre entre 1 et 6 :
AL = 1 + INT(RND(1) * 6).
ou encore
AL = INT(RND(1) * 6 + 1)

CARRÉ il correspond aux codes suivants :

$01 00 04 0D
$21 3F 36 2D 1C 00

La première ligne indique le nombre de formes puis le nombre d'octets séparant la forme du début de la table. La seconde représente les codes de la forme 1 (le carré). On pourrait partir d'une forme double en utilisant :

$12 3F 24 24 2D 2D 36 36 3F 00

PETITS CARRÉS

SCALE varie de 19 à 21 (20 étant la valeur correcte). Cela se passe à la ligne 170.

CARRÉ MOYEN

Ce sont les valeurs de la ligne 640 (DATA) qui sont aléatoirement employées (ligne 200). La variable D (ligne 210) permet d'afficher le carré dans des colonnes autorisant l'utilisation de la couleur choisie. ■

Nicole BRÉAUD-POULIQUEN

L'ADRESSAGE DYNAMIQUE DU GS

INTRODUCTION

Parmi les ORIGINALITÉS du microprocesseur 65816, nous avons remarqué :

- un bus d'adressage sur 24 bits, c'est-à-dire un espace d'adresses compris entre $000000 et $FFFFFF soit 16 777 216 octets en accès direct ;
- un registre Banc de programme sur 8 bits contenant les poids forts des adresses des instructions du programme en cours d'exécution ;
- un registre Banc de Données de 8 bits contenant les poids forts des adresses des données à exploiter ;
- en plus des modes d'adressage particuliers classiques : IMMÉDIAT, INHÉRENT, ACCUMULATEUR ;
- des extensions des modes généraux d'adressage affinés par l'Indirection et l'Indexation :
  - 5 modes en adressage ABSOLU ;
  - 5 modes en adressage PAGE ZÉRO ou DIRECT ;
  - 2 modes en adressage RELATIF ;
  - 2 modes d'adressage RELATIF dans la Pile ;
- des registres d'INDEX X et Y sur 8 ou 16 bits
- un registre pointeur de PILE sur 16 bits
- un registre DIRECT pointeur de PAGE ZÉRO sur 16 bits.

Ces deux derniers registres permettant l'implantation DYNAMIQUE de la PILE et le la page ZÉRO dans le premier banc de mémoire de $000000 à $00BFFF.

Il s'agit donc d'un système d'adressage très souple permettant une SEGMENTATION des programmes : un segment de Données ou un segment de Code est exploitable facilement quel que soit le banc de mémoire où il est chargé ou déplacé.

Le problème intéressant qui découle de ces remarquables atouts du microprocesseur et de cet attrait de la programmation par segmentation est celui de la GESTION dynamique de la mémoire.

Nous allons confier cette tache au MEMORY MANAGER, un outil disponible en MEM, qui est un système de réservation de places de mémoire en temps réel. Au niveau de l'implantaion des segments de programmes, le maître d'œuvre est le System LOADER qui fait partie du système d'exploitation ProDOS16.

DÉFINITIONS

Chiffre Binaire ou Bit : Noté bi de valeur 0 ou 1.

Chiffre Hexadécimal : Noté Hi de valeur 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E ou F. Equivalent à une combinaison de 4 bits.

Octet : Combinaison de 8 bits notée b7b6b5b4b3b2b1b0.
exemple : %00101011

ou BYTE : 2 chiffres hexadécimaux $HpHm
(- - Hp - -) (- - Hm - -) $2B
Hp = 4 bits les Plus significatifs
Hm = 4 bits les Moins significatifs

Mot : 2 octets (WORD) ou 4 octets (LONG)
exemples : $0100 $00E06028
MpMm Lp Lm

Adresse : 24 bits (6 chiffres hexadécimaux)
b23--b16 Banc ou Badr
b15--b8 Adresse Haute ou adH
b7----b0 Adresse Basse ou adL
exemple : $E100A8 : adL = $A8, adH = $00, Badr = $E1

Pointeur : 2 ou 4 octets ayant pour valeur une adresse.
exemple : $00FE013F (Hp = $00) *(Suite page 68)*

Handle : 4 octets ayant pour valeur une adresse, laquelle contient un pointeur :

exemple : $00E06028 qui contient $00000C00, l'adresse de début d'une page ZÉRO.

Segment : Bloc d'une certaine taille constitué d'octets de mémoire adjacents. L'adresse du 1er octet de ce bloc est enregistrée dans le pointeur du bloc, appelé *POINTEUR-MAÎTRE ;* la position en mémoire du pointeur-maître ne change pas alors que le segment peut être déplacé. Ce qui conduira au simple changement de valeur du pointeur-maître. Pour le programmeur chaque bloc ou segment est défini par :

- sa longueur (un mot de 4 octets) ;
- l'adresse de son pointeur-maître appelée handle du bloc (4 octets) ;
- son numéro d'identification (un mot de 2 octets) ;
- ses attributs : fixe ou non, aligné sur une page, purgeable, etc... (16 bits).

Page Zéro : Segment de 256 octets situé dans le banc $00 dont l'adresse du 1er octet est contenue dans le registre Direct lorsque cette page est en activité. Les données sont exploitées avec le mode d'adressage Direct où seulement 8 bits sont nécessaires pour leur localisation dans cette page.

Pile : Segment situé dans le banc $00 dont l'accès est normalement fait sur le mode dernier entré/premier sorti grâce aux instructions d'empilement et de dépilement. Le registre Stack Pointer pointe la première position disponible sur la pile. Le remplissage de la pile se fait par adresses décroissantes.

Appels des fonctions d'Outils

Soit une fonction (ou un sous-programme) appelée par un autre programme qui lui transmet des paramètres pour que celle-ci lui renvoie un ou des résultats. Ces données d'entrée et de sortie d'une fonction peuvent transiter dans la pile : c'est le cas des fonctions d'outils disponibles dans le *GS* (en MEM et en MEV).

Pour appeler une fonction d'outil, il est essentiel de faire d'abord une place suffisante sur la pile où la fonction tiendra à notre disposition les résultats demandés, puis d'empiler les paramètres d'entrée ; l'appel lui-même consiste à charger le registre X du code de la fonction demandée avant de faire un saut au sous-programme débutant en E1/0000 qui est l'unique point d'entrée de TOUTES les fonctions. C'est l'instruction JSL qu'il faut utiliser puisque le programme appelant peut se situer dans n'importe quel autre banc de mémoire vive.

```
LDX $code
JSL E10000
```

Après l'exécution de la fonction, il faudra tester le bit Carry pour être sûr qu'il n'y a pas eu de problème. Le code de l'erreur est dans l'accumulateur si C=1. Si C=0, alors le résultat, s'il est attendu, se trouvera sur le sommet de la pile. Par exemple un pointeur sera récupéré sur la pile par :

```
PLA
STA Adr
PLA
STA Adr+2
```

les octets les plus significatifs étant les derniers sortis. Une fois les résultats désempilés, le pointeur de pile se doit d'être à la valeur qu'il avait avant de débuter l'empilement des paramètres qui préparait l'appel de la fonction.

Le Memory Manager

Supposons qu'un programme d'application ait été chargé en mémoire vive par les soins du System Loader sous ProDOS16. Le Memory Manager lui a réservé un bloc et ce bloc où il est implanté porte un numéro d'Identification (1003 par exemple). Ce numéro doit absolument être connu par le programme si celui-ci veut réserver d'autres blocs de données pendant son exécution, blocs qui porteront donc le numéro du programme qui les a demandés au Memory Manager.

Par souci d'efficacité, ce numéro est justement renvoyé par la fonction $0202 ou _MMStartup, fonction qu'il est indispensable d'appeler au début du programme d'application. Voici cet appel :

```
PEA 0000      on empile 2 octets de valeur 0 pour que
              la fonction puisse y stocker le N°.
LDX £0202
JSL E10000
PLA           Les 2 octets sont désempilés et mis dans
              l'accumulateur.
STA id        id contiendra le N° d'identification du
              programme en cours.
```

Par exemple la fonction NewHandle H:4 (LG:4 ID:2 AT:2 AD:4) renvoie un handle H pour le segment identifié ID, de taille LG, ayant les attributs AT et éventuellement une adresse AD. Cette fonction qui appartient au Memory Manager va permettre à celui-ci de gérer dynamiquement l'espace-mémoire pour tous les programmes — système ou utilisateur — qui veulent réserver de la place en mémoire au fur et à mesure de leurs besoins.

Réservation d'une page zéro

```
PEA $0000     H:4 octets sur la pile pour y mettre
```

```
PEA $0000      le résultat.
PEA $0000      LG : Longueur du segment sur 4 octets.
PEA $0100
LDA ID         ID : N° d'Identification du segment sur 16 bits.
PHA
PEA $C001      AT : Attributs du segment sur 16 bits.
PEA $0000      ADL : Adresse du segment sur 4 octets.
PEA $0000      ADH (0 si à déterminer par le Memory Manager).
LDX £$902      Code de la fonction NewHandle du Memory Manager.
JSL $E10000    Exécution de cette fonction.
BCS Erreur
PLA
STA 00         Recopie des 2 octets les moins significatifs
PLA            du Handle résultant, aux positions $00 et $01
STA 02         et des 2 octets les plus significatifs en $02
               et $03 de la page Zéro pointée par D.
LDA °00§       Chargement du contenu de l'adresse contenue dans
               $00,$01,$02,$03 c'est-à-dire les 2 octets les moins
               significatifs de l'adresse de début du bloc réservé
               (mode d'adressage Indirect Direct Long).
STA 00         Recopie de cette valeur en $00,$01 (les 2 octets les
               moins signifactifs suffisent à connaître l'emplacement
               de ce bloc, puisqu'il s'agit d'une page Zéro — banc
               $00 —).
```

Cet exemple met en évidence les particularités suivantes :

- Les paramètres des fonctions d'Outils sont transmis sur la pile, le résultat de la fonction étant disponible en haut de la pile à l'aboutissement de la fonction ; c'est au programme appelant de réserver la place du résultat sur la pile avant d'empiler les paramètres d'entrée de la fonction.

- L'instruction PEA est précieuse pour empiler une valeur immédiate sur la pile sans passer par l'accumulateur. La programmation en Macro-assembleur permettrait une écriture plus évoluée :

```
PushWord  £val   qui empile 2 octets en mode immédiat
                 (PEA val)
PushWord  Adr    qui empile le contenu de Adr (LDA Adr,
                 PHA)
PushLong  £val   est transformée en PEA valp, PEA valm
PushLong  Adr    est transformée en LDA Adr, PHA, LDA
                 Adr+2, PHA
```

- La fonction NewHandle est habilitée à réserver de la place dans tout l'espace-mémoire mais chaque segment alloué est muni des attributs suivants :

— Les attributs permanents :

Fixe en cas de compactage	(bit 14)
Ne pouvant être à cheval sur 2 bancs	(bit 4)
Non situé dans les bancs 0,1 et les zones-écran	(bit 3)
Aligné sur une page	(bit 2)
Implanté à une adresse définitive	(bit 1)
Situé dans un banc donné	(bit 0)

— Les attributs temporaires :

Verrouillé (immuable provisoirement)	(bit 15)
Un niveau de priorité en cas de purge	(bits 8 et 9)

— Par exemple :

C000 : segment fixe et verrouillé (zones-système)

C115 : fixe, verrouillé, niveau 1 de purge, sur un banc, aligné et dans un banc donné (pour les pages Zéro des outils)

C013 : verrouillé, fixe, implanté à une adresse donnée (ProDOS)

C010 : verrouillé, fixe, tenant dans un banc (programme-utilisateur en cours)

4310 : fixe, purgeable, tenant dans un banc.

- La fonction NewHandle a besoin de connaître le N° d'identification du segment à réserver en mémoire, c'est-à-dire quelle catégorie (0 à F) de programme il servira et à quel N° d'ordre nn de programme dans cette catégorie. Ce N° d'identification est attribué automatiquement par le programme-pilote de chargement et il est lisible par la fonction GetID de l'outil Miscellaneous.

Voici les modèles de N° d'Identification de segment :

0xxx	Memory Manager
1xnn	Programme d'application (ex : 1002)
2xnn	Programme-Pilote
3000	ProDOS
4xnn	Outils
5xnn	Accessoire de bureau
6xnn	Code-objet chargeable en cours d'exécution
7xnn	Chargeur de programmes
8xnn	Fonction-système
9xnn	Tool Locator, outil N°1 de localisation des autres
Axnn	Setup File, fichier-programme chargé au démarrage (le paramètre x est fourni au Memory Manager dans l'ordre croissant des demandes pour un même type de segment).

- L'adresse de début du bloc reste valable aussi longtemps que le bloc n'a pas été déplacé. Si l'on se sert de cette adresse pour exploiter les données de ce bloc à la place du Handle, on dit que le Handle est déréférencé.

Visualisation des réservations avec le Mangler

Pour mieux se rendre compte des emplacements attribués aux segments de programmes-systèmes ou utilisateur, il existe un accessoire de bureau le MANGLER que l'on peut appeler à tout moment (par les touches Esc-Ctrl-PO) et qui répond à la commande LIST en affichant la liste des handles sous la forme suivante (voir tableau page 70).

La colonne ID permet de repérer le type de segments, ainsi 1003 identifie un programme-utilisateur, ou 400E, l'outil Window Manager. *(Suite page 70)*

LISTE DES HANDLES (lire page 69)							
N°	Ptr-Maître	Adresse	Attribut	ID	Longueur	Précédent	Suivant
01	E11700	000000	C000	0000	000800	000000	E11A70
02	E11A70	000800	C115	1002	000400	E11700	E06028
03	E06028	000C00	C105	1002	000800	E11A70	E1191C
04	E1191C	009500	C013	3001	00035C	E06028	E118F4
05	E118F4	00985C	C013	3001	0026A3	E1191C	E118F4

La troisième colonne donne l'adresse d'implantation du segment qui est le contenu du Pointeur-Maître dont l'adresse est dans la deuxième colonne. C'est la valeur de la deuxième colonne qui est présente dans la pile après l'exécution de la fonction NewHandle. La commande MON conduit au moniteur et on en sort par Ctrl-Y.

Essayons de suivre la liste des handles :

*E1/1700.1713

E1/1700 : 00 00 00 00 00 C0 00 00 00 0800 00 00 00 00 00
E1/1710 : 70 1A E1 00

Chaque élément de la liste des blocs réservés comprend donc successivement :

- l'adresse du début du bloc : 4 octets
- les attributs : 2 octets
- l'ID du programme : 2 octets
- la longueur du bloc : 4 octets
- l'adresse de l'élément suivant : 4 octets
- l'adresse de l'élément précédent : 4 octets

Les mots sont toujours implantés avec l'octet le moins significatif devant. Les adresses doivent être lues de droite à gauche, exemple : E1/1A70.

Les autres commandes intéressantes du Mangler sont :

TOTALMEM	renvoie le nbre total d'octets de mémoire vive	00 14 00 00
FREEMEM	renvoie le nbre d'octets libres	00 0E 8F 05
MAXBLOCK	renvoie le bloc le plus grand disponible	00 0B BC B3

Les résultats obtenus sont ceux d'une configuration du *GS* avec une carte d'extension de 1 MegaOctets, le système APW et son Editeur en ligne.

Exemple de blocs réservés pendant l'exécution

Les Window-Records de Window Manager sont des blocs de données utilisés par le gestionnaire des fenêtres mais le programme d'application n'est pas supposé y avoir accès directement.

Les fonctions appliquées à des fenêtres demandent en effet comme argument le pointeur qui aura été renvoyé par la fonction de création d'une nouvelle fenêtre :

```
Allocation  START
            Using Globales
            PushLong £0
            PushLong £WindParams
            _NewWindow
            pla
            sta WindPtr
            pla
            sta WindPtr+2
            création d'autres fenêtres
            rts
WindParams  anop
            dc i2'EndWind-WindParams'
            dc i2'%1100000010100000'
            ...(tous les paramètres)
EndWind     anop
            END
```

La fonction NewWindow crée un bloc appelé *Window Record* à partir des paramètres définissant la fenêtre et l'environnement et l'intègre dans la liste des fenêtres. La structure du Window Record comprend en tête un pointeur sur le Window Record de la fenêtre suivante, puis la structure du Port Graphique de cette fenêtre appelé *Window Port*, puis des handles des différentes régions et des contrôles et un entier caractérisant le type de cadre. NewWindow renvoie dans la pile l'adresse du Window Port et c'est elle que l'on doit utiliser comme argument d'entrée pour toutes les fonctions applicables à cette fenêtre. La variable correspondante WinPtr sera déclarée comme globale, c'est-à-dire que son emplacement sera réservé dans un segment de données que tous les segments du programme pourront utiliser grâce à la directive Using Globales.

```
Globales    DATA
Id          ds 2
WinPtr      ds 4
            autres données
            END
Activation  START
            Using Globales
            PushLong WindPtr
            _SelectWindow   ┌ rend active la fenêtre
            etc.            │ désignée par l'adresse
            END             └ de son port.
```

Rappels sur les instructions de réservation et de déclaration de variables

En Macro-Assembleur

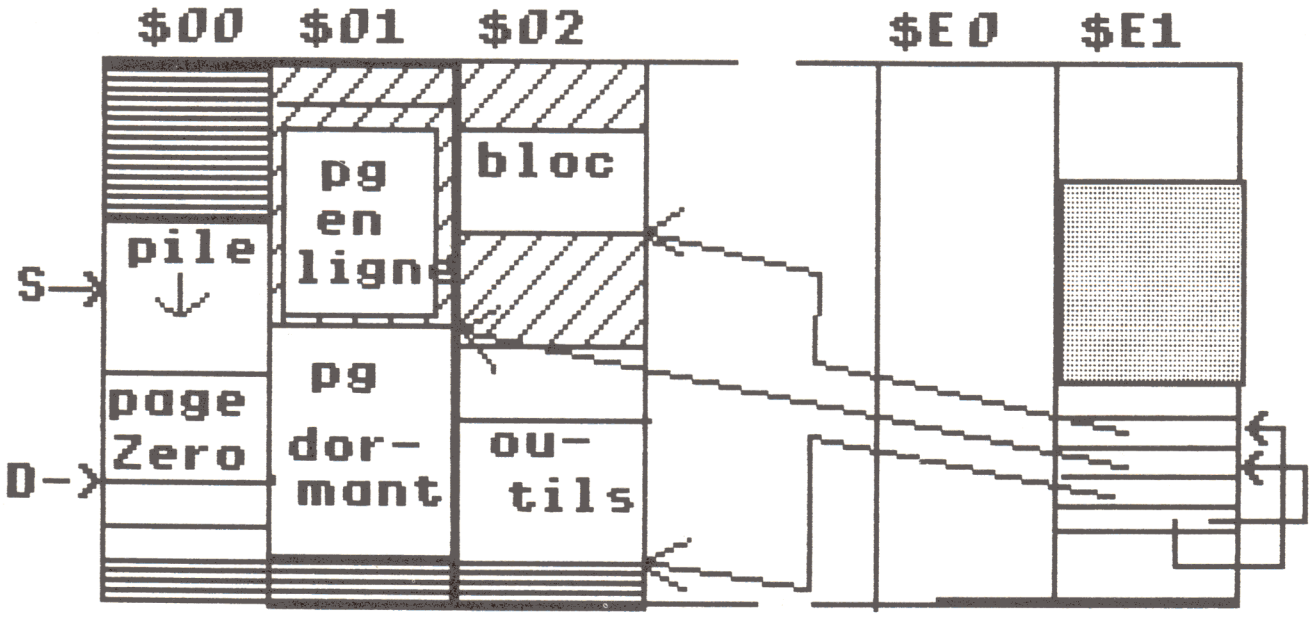
```
variable entière        x       dc    i2'0'
pointeur sur x          xptr    dc    i4'x'

chaîne de caractères    nom     dc    c'Un Titre'
pointeur sur chaîne     nptr    dc    i4'nom"
handle d'un bloc        handle  gequ  00
                        le programme utilisera l'adresse $00
                        de la page Zéro en cours pour stocker
                        les 4 octets d'un handle fourni par
                        NewHandle.
pointeur                wptr    ds    4
                        la valeur du pointeur sera enregistrée
                        en wptr où 4 octets lui sont réservés.
```

En C

```
variable entière        int x           (type)
pointeur sur x          int * xptr      (type)
                        xptr = &x       (valeur)
                        char nom°8§
chaîne de caractères    strcpy(nom,"Un titre")
```

Perspectives

Voir figure ci-dessous.



ATTENTION !

A la suite d'une réorganisation des services rédactionnels de *TREMPLIN MICRO,* il ne nous est plus possible de fournir des renseignements ***techniques*** par téléphone.

Vous pouvez par contre nous écrire, mais n'oubliez pas de joindre une enveloppe timbrée pour la réponse. MERCI ! ■

Yvan KOENIG répond à nos lecteurs

Question : *Nous avons un problème : nous voulons, au cours d'un programme en PASCAL, charger des illustrations réalisées à l'aide du logiciel de dessin "Extasie". Mais, après le rappel du dessin sauvé en binaire sur disquette, nous n'arrivons pas à connecter la double haute résolution nécessaire au bon affichage des pages ("Extasie" fonctionne en double haute résolution et les dessins doivent donc être affichés dans ce mode). Pouvez-vous nous indiquer la marche à suivre en BASIC et, si possible en PASCAL U.C.S.D. (sinon, nous traduirons). Merci d'avance ?*

Emmanuel R. (club micro du collège St-Joseph du Parchamp).

R Comme vous êtes lecteur de *Tremplin Micro*, cela me conduit à vous conseiller de vous reporter à l'article Double Haute Résolution, publié dans *Tremplin Micro* n°10.

Dans cet article, le chargement et l'affichage d'images DHGR était expliqué. Cependant, le programme en question étant destiné à un *IIe*, il n'était pas fait état d'une opération indispensable sur le *IIc*.

POKE 49165,0 STA $C00D	Active 80COL, le 'firmware' activera 80STORE.
POKE 49236,0 STA $C054	Retour page1 (MAIN) par sécurité.
POKE 49232,0 STA $C050	Active mode graphique.
POKE 49235,0 STA $C053	Graphique seul.
POKE 49239,0 STA $C057	Active haute résolution.
POKE 49278,0 STA $C07E	Active IOUDIS, utile sur *IIc*.
POKE 49246,0 STA $C05E	Active DHGR mais n'opère sur *IIc* qu'après avoir fait STA $C07E

Questions : *1° Existe-t-il un moyen pour savoir sur quelle machine se trouve un programme ; j'ai développé un programme sur II+ destiné à un IIc qui possède les flèches haut et bas. Je désirerais donc charger un fichier "FLECHE2C" lorsque le programme tourne sur IIc et "FLECHE2+" quand il tourne sur II+, et ceci de façon transparente pour l'utilisateur (pas de conversationnel !).*

2° Toujours avec les fichiers : J'ai rempli une première fois des enregistrements d'un fichier à accès direct de longueur 265 et cela sur 10 enregistrements, par exemple. J'ai modifié l'enregistrement n°3 et il contient bien les nouvelles valeurs. Est-il possible de savoir, en examinant finement la structure du fichier ou plus généralement le disque, si l'enregistrement 3 a été effectivement modifié et d'autre part, peut-on retrouver les anciennes valeurs ? Même question si il y a plusieurs modifications (dans le temps) sur un même enregistrement.

Philippe D. (92240 MALAKOFF)

R **IDENTIFICATION DES MACHINES**

Cela commence à devenir compliqué :

	FBB3	FB1E	FBC0	FBBF
APPLE II	$38			
APPLE II+	$EA	$AD		
APPLE IIe	$06		$EA	
APPLE IIe+	$06		$E0	
APPLE IIc 5"	$06		$00	
APPLE IIc 3.5"	$06		$00	$FF
APPLE III émul	$EA	$8A		$00

A mon avis, pour votre problème, il y a mieux à faire. Si j'ai bien compris vous souhaitez utiliser les flèches droite-gauche sur *II+*, haut-bas sur *IIe* ou *IIc*. Faites donc les deux en vous souvenant que flèche haut = Ctrl K et flèche bas = Ctrl J.

Ce qui se code ainsi :

```
CMP  £$85      |  CMP  £$88
BEQ  down      |  BEQ  up
CMP  £$8A      |  CMP  £$8B
BEQ  down      |  BEQ  up
down LDA £$8A...  up LDA £$8B...
```

Je vous suggère d'homogénéiser ainsi le contenu de l'accumulateur, si vous devez refaire un test ultérieurement, pour savoir quelle commande est active.

MODIFICATION D'UN ENREGISTREMENT

Désolé ! lorsque vous modifiez un enregistrement SANS effacer la totalité du fichier (ce qui est normal) le nouveau contenu s'écrit SUR l'emplacement de l'ancien (heureusement). Tout ce que vous pouvez espérer récupérer, avec TDUMP ou PROTYPC par exemple, c'est éventuellement la fin de l'ancienne fiche si la nouvelle chaîne est plus courte que l'ancienne. **Exemple** : ancienne = Anticonstitutionnellement/ (le / représente ici le (CR))

nouvelle = Z/.

Sur disque vous avez alors disque = Z/ ticonstitutionnellement/ ce qui ne gêne pas en accès normal mais peut être 'surprenant' en lecture par TDUMP ou PROTYPC.

J'utilise cette méthode lorsque je suis conduit à annuler une fiche. Je mets alors un DELETE ($7F) en tête. Ça permet, lors des tris, de repousser les annulés en fin de liste et je peux si nécessaire récupérer la chaîne (sauf les 2 premiers caractères). Cela se rapproche du mode effacement de fichier sous DOS 3.3 où le numPiste TSL est mis à 0 MAIS est sauvé en fin du titre fichier.

Question : *Je possède un APPLE IIe muni d'une carte 80 colonnes étendues (carte CHAT MAUVE). J'ai écrit un programme en BASIC permettant de gérer des cartons de LOTO. Afin d'accélérer la recherche des numéros tirés, les cartons sont placés dans un tableau en mémoire vive. Si cette solution me donne entière satisfaction au niveau de la vitesse d'exécution ; elle est néan-*

moins coûteuse en place mémoire. Si bien que je ne puis introduire plus de 250 cartons. Au-delà, le message OUT OF MEMORY apparaît. Pourriez-vous m'indiquer une solution qui me permettrait de dépasser la limite de 250 cartons, comme par exemple : utilisation de la mémoire auxiliaire de la carte CHAT MAUVE ou reloger le D.O.S. dans la carte langage ? Disposez-vous de programmes permettant d'effectuer ces opérations ? Par avance, je vous remercie de bien vouloir consacrer quelques instants à l'étude de mon problème.

Michel M. (12300 DECAZEVILLE)

R

Vous êtes avare en informations lorsque vous posez votre problème et de ce fait, il est difficile de trouver une solution. Je vais tenter de vous donner des pistes.

Quel système d'exploitation utilisez-vous, DOS 3.3 ou ProDOS ? Quel type de tableaux utilisez-vous : flottants, entiers ou chaînes ?

On peut reloger DOS 3.3 en carte langage, cf. Call APPLE in depth All About DOS 3.3 (288 pages en anglais), normalement disponible chez SIVEA. Il n'est par contre pas possible de déplacer ProDOS.

Si vous travaillez sur des tableaux de chaînes il existe un utilitaire EXTRA.K chez BEAGLE BROS qui permet de mettre les variables chaînes dans la MemAUX. Cet utilitaire est peut-être disponible chez SIVEA.

Si vous utilisez actuellement des tableaux flottants, les remplacer par des entiers ralentira le programme, mais occupera 2 octets par numéro au lieu de 5.

Si vous employez déjà des entiers, il vous serait possible de loger 2 numéros dans un seul octet avec la formule NN = 100 * N1 + N2 qui ne fonctionne que si les numéros sont toujours inférieurs à 100 mais je crois que c'est le cas. Enfin, il est possible d'envisager de renoncer aux fichiers TEXT et d'employer un gros fichier BIN dans lequel chaque numéro de 0 à 255 occuperait un seul octet. L'inconvénient de cette formule est que l'on doit à chaque fois charger et sauver le fichier complet, mais un BIN se charge beaucoup plus vite qu'un TEXT.

Enfin, il n'est peut-être pas absolument indispensable d'avoir tous les cartons en même temps en mémoire. Pourquoi ne pas lire des groupes de 50, les analyser, charger un autre groupe et ainsi de suite. Cette formule permettrait de réaliser un traitement indépendant du système d'exploitation et, éventuellement, l'acquisition d'une carte extension mémoire de 256 ou 512 K vous permettrait d'accéder plus vite encore aux fichiers chargés en RAMdisk.

J'espère que vous comprendrez qu'il m'est impossible d'être plus précis.

CONFIGURATION IMAGEWRITER

Travaillant actuellement sur un Apple IIc, j'ai pris quelques jolies colères à cause du mode de gestion de l'interface série. Fatigué de taper PRINT CHR$(9) "96N" avant chaque listage et ne souhaitant pas ajouter cette séquence dans mes programmes, j'ai mis en place une courte routine qui, sans passer par les utilitaires, permet de CONFIGURER l'interface série sur la largeur de votre choix (il faut bien entendu que cette largeur soit compatible avec la police adoptée, ex. : 96 maxi en ELITE). La routine elle-même comporte 25 octets, mais je me suis amusé à la mettre en place dans un petit programme BASIC qui montre en prime comment ne pas trop se fatiguer. Tapez la routine HEXA, tapez ensuite les lignes 1, 3, 4, 10, 20, 30, 40, 50, 70. En recopiant la ligne 3 à l'aide du curseur (ESC I.....) et bien entendu en changeant au passage quelques détails vous aurez la ligne 1.

Faites RUN, les lignes 1140 et 1150 que vous n'avez pas tapées vont s'afficher sur l'écran où vous pourrez les relire à l'aide du curseur (il y aura un : en trop à la fin de chaque ligne).

```
1     GOTO 10
2     PRINT " 1140";: FOR I = 0 TO 11: PRINT "POKE"768 + I","
      PEEK (768 + I)":";: NEXT : PRINT
3     PRINT " 1150";: FOR I = 12 TO 24: PRINT "POKE"768 + I","
      PEEK (768 + I)":";: NEXT : PRINT
4     END
10    TEXT : HOME : D$ = CHR$ (4)
20    GOSUB 1140
30    FOR I = 1 TO 6: POKE 776,16 * I
40    CALL 768
50    PRINT D$"PR£1": LIST 1140,1150: PRINT D$"PR£0": NEXT
70    END
1140  POKE 768,173: POKE 769,24: POKE 770,192: POKE 771,72:
      POKE 772,141: POKE 773,1: POKE 774,192: POKE 775,169:
      POKE 776,96: POKE 77,141: POKE 778,85: POKE 779,192
1150  POKE 780,141: POKE 781,123: POKE 782,4: POKE 783,141:
      POKE 784,84: POKE 785,192: POKE 786,104: POKE 787,48:
      POKE 788,3: POKE 789,141: POKE 790,0: POKE 791,192:
      POKE 792,96
1160  RETURN
```

```
300 : AD 18 C0    LDA $C018
303 : 48          PHA
304 : 8D 01 C0    STA $C001
307 : A9 60       LDA £$60      C'est lui que l'on ajuste.
309 : 8D 55 C0    STA $C055
30C : 8D 7B 04    STA $047B
30F : 8D 54 C0    STA $C054
312 : 68          PLA
313 : 30 03       BMI $0318
315 : 8D 00 C0    STA $C000
318 : 60          RTS   BSAVE SETLARGEUR,A$300,L$19
```

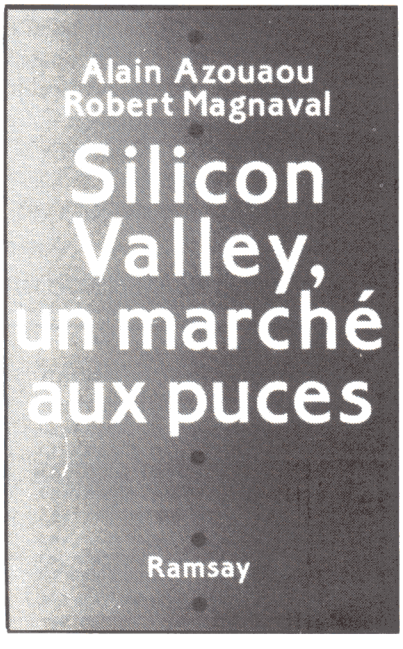
Votre bibliothèque INFORMATIQUE

par NESTOR

• SILICON VALLEY, UN MARCHÉ AUX PUCES (A. Azouaou et R. Magnaval)

De la fameuse Silicon Valley, nous avons tous entendu parler. Pour la plupart des jeunes Français (et aussi pour quelques "vieux" mordus de l'informatique personnelle — j'en suis !), c'est le nouvel Eldorado. Un gigantesque temple du futur, et en tout cas la matérialisation, pour de jeunes Américains ambitieux et doués, de rêves fabuleux. Chez les Applemaniaques, on pense évidemment aux deux Steve, mais ce ne sont que deux grandes figures parmi tant d'autres. La Silicon Valley, cela représente d'abord 4000 entreprises de pointe regroupées sur un site aux dimensions de Paris. Est-il exact que la majorité des entreprises naissent dans des garages ? C'est une autre histoire... et c'est précisément là que Robert Magnaval et Alain Azouaou interviennent. Leur livre est la première étude française sur ce sujet. Il vous permettra, c'est sûr, de mieux comprendre le modèle "Silicon Valley".

EDITIONS RAMSAY,
9, rue du Cherche-Midi, 75006 PARIS
240 pages — 92 F ttc.



• TROIS ÉTAPES VERS L'INTELLIGENCE ARTIFICIELLE SUR PC ET COMPATIBLES (René Descamps)

Oui, je sais, vous ne possédez pas d'IBM PC et vous ne connaissez pas le Basic Microsoft. Mais ce n'est vrai que pour une partie du lectorat de *Tremplin Micro*. En effet, depuis quelque temps — et notamment depuis environ un an —, les utilisateurs d'un Apple sont fréquemment amenés à travailler sur IBM ou compatible. La chose deviendra d'ailleurs possible, prochainement, sur le nouvel Apple *IIGS* (il nous doit bien cela !).

Alors, si vous en avez la possibilité, ouvrez le bon bouquin de René Descamps et essayez, en sa compagnie, de résoudre les puzzles de Taquin, de simuler un pilote automatique ou de mieux cerner la programmation de jeux aussi complexes que les dames et les échecs.

A noter que les paresseux peuvent commander la disquette d'accompagnement, mais j'ignore comment elle se présente, ne l'ayant pas eue en main. Je crois en tout cas être capable, avec l'aide de René Descamps (c'est un maître en la matière), de créer mes propres systèmes experts, sur Apple bien sûr, mais en m'inspirant des programmes proposés sur IBM.

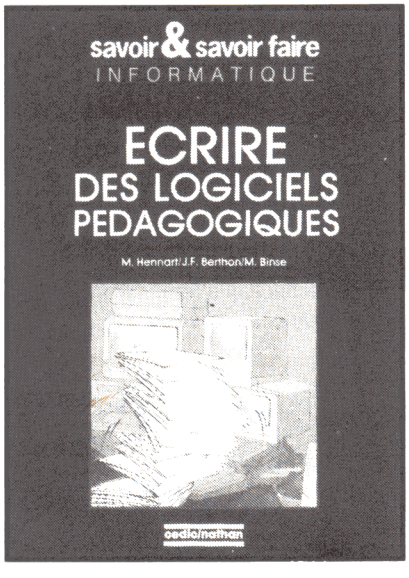
EDITIONS DU P.S.I.
BP 86, 77402 LAGNY-SUR-MARNE
270 pages — 205 F ttc.

• ÉCRIRE DES LOGICIELS PÉDAGOGIQUES (M. Hennart, J.-F. Berthon et M. Binse)

Ce nouveau volet de la collection *"Savoir & Savoir-faire"* est un guide pratique essentiellement destiné aux enseignants. Il devrait les aider à définir et à réaliser des logiciels pédagogiques, mais il peut aussi présenter un intérêt pour toute personne désirant créer des didacticiels solides et utiles.

C'est un ouvrage de professionnels. Sa lecture est édifiante. Les exemples ne tournent pas, on s'en doute, sur le matériel Apple, mais il n'est pas interdit de les adapter.

CEDIC/NATHAN, *6, bd Jourdan, 75014 PARIS*
Tél. (1) 45.65.06.06.



N°13

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°13 ☐ 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 ☐ N°13 ☐	 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**

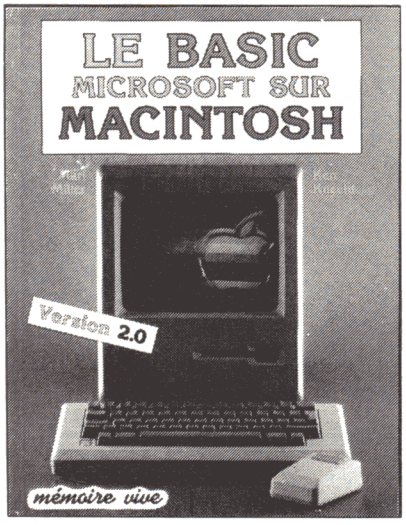
Votre bibliothèque INFORMATIQUE

par NESTOR

• LE BASIC MICROSOFT SUR MACINTOSH (Merl Miller et Ken Knecht)

C'est un ouvrage de *MÉMOIRE VIVE*, éditeur québécois bien connu, diffusé en France par le P.S.I. Il a été traduit de l'américain par Gilles Saintonge, Yves Leclerc (j'ignore s'il est parent de Félix !) et Robert Davies. C'est un livre copieux, complet, clair et tout à fait indispensable pour programmer en Basic, sur Macintosh. Je vous signale au passage qu'il se révèle également irritant quand on est possesseur d'un Apple *IIGS* et de son misérable Appiesoft. A quand le Basic Microsoft sur cette machine ?

Tenez ! j'en rêve la nuit et je me demande parfois pourquoi je ne me suis pas offert un Mac d'occasion plutôt qu'un GS neuf. "Patience !", m'a récemment recommandé Guy-Hachette (qui n'en pense pas moins). Il attend, semble-t-il, un nouveau Basic... à moins qu'il ne s'agisse d'un C performant, mais les Borland et autres génies ne semblent pas disposés à investir dans le GS. Mais revenons à notre bouquin pour constater que les possesseurs de Mac ont bien de la chance. Ils disposent, avec le Basic de Microsoft, d'un langage réellement performant. Le guide édité par "Mémoire Vive" concerne la version 2.0 et fournit réellement tous les éclaircissements sur un Basic de plus en plus répandu. Fenêtres, sourigraphie, animation et effets sonores, fichiers et gestion de fichiers, boîte à outils, commandes et fonctions.... et puis les références thématiques, les erreurs communes du Basic, les mots réservés... et tout et tout !



EDITIONS DU P.S.I. *386 pages — 250 F ttc.*

• LA PROGRAMMATION SANS PANNE (Y. Tallineau, N. Moine et I. Dall'O)

Cet ouvrage constitue plutôt un cours magistral qu'un livre ordinaire sur l'informatique. On peut en conseiller la lecture — ou plutôt l'étude — à toute personne désirant approfondir la démarche structuraliste et son application à l'écriture des programmes. Un rapide survol de la table des matières vous donnera une idée du contenu de la méthode : les quatre points d'appui, la démarche structuraliste, les normes de réalisation, les outils, l'organisation du travail, les aspects psychologiques, la formation, les chiffres repères, l'avenir, un exemple de description d'un algorithme, l'écriture de programmes et de procédures...

Bonne présentation, très aérée, dans un format inhabituel (le format A4).

EDITIONS D'INFORMATIQUE,
99, bd Jean-Jaurès, 92100 BOULOGNE
250 pages — 450 F ttc.

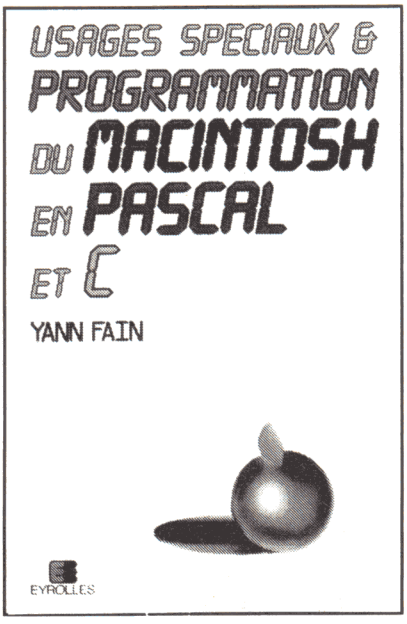
• USAGES SPÉCIAUX ET PROGRAMMATION DU MACINTOSH EN PASCAL ET C (Yann Fain)

L'auteur est avant tout un mathématicien, mais il programme depuis 1970 et dirige actuellement une société d'informatique. Il a aussi animé de nombreux cours de développement sur diverses machines, dont le Macintosh... auquel il s'intéresse spécialement dans ce livre.

Son objectif : montrer à ses lecteurs comment mieux se servir de leur micro favori et, surtout, comment le programmer en Pascal, en C, en Forth et en Basic. Bien entendu, il est impossible de tout enseigner en quelque 300 pages, mais il est par contre possible d'éclairer une personne curieuse sur bien des points.

Par de nombreux exemples commentés, Yann Fain vous apprendra comment personnaliser vos programmes, faire passer vos données d'une application à une autre sans passer par les fastidieux Couper-Coller, réparer un Finder défectueux, bref, comment vous tirer de situations parfois délicates. Dans son avertissement, l'auteur précise que (nous le citons) même si vous n'êtes pas un programmeur chevronné, vous avez en main un livre qui vous permet aussi d'écrire des applications, de comprendre les menus, les fenêtres, les icônes, les rapports entre celles-ci et les noms des applications, les ressources, les événements, et de composer du code sur votre Mac. Il ajoute que les programmes fournis sont opérationnels. *Tremplin Micro* attend le même guide... pour l'Apple *IIGS*. A vous de jouer, Yann Fain ! Si l'aventure vous tente, vous bénéficierez, c'est promis, de notre appui et vous comblerez les désirs de nombreux lecteurs !

EYROLLES, *61, bd Saint-Germain, 75240 PARIS CEDEX 05* *324 pages — Prix non connu*



Chasseur d'Images

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

Chez votre marchand de journaux !