

# tremplin micro

**Un logiciel complet  
de dessin sur Apple**

**Le prix  
du coup de fil**

**Les pokes  
en Applesoft**

**Programmer ?  
Mais c'est  
très simple !**

**LE  
PRIX  
DU  
COUP  
DE  
FIL**



M 1631 - 15 - 33,00 F



3791631033006 00150

N° 15 - Bimestriel - Troisième année  
5 Juillet - 4 Septembre 1987  
254 FB - 11 FS - **33 F**



# tremplin micro 15

# SOMMAIRE

Avec la collaboration de :

ARGOS, Claude AUBRY, Nicole BRÉAUD-POULIQUEN, Marc BROCHA,  
Yves BUONO, Maurice CHAVELLI, Marcel COTTINI, François GALLET  
et Thierry FILLAUT, NESTOR, Pierre PRIVAT et Clément RENARD.

Apple et ProDOS (noms et logos) sont des marques déposées d'Apple Computer, Inc.

## 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.

TREMPIN 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.

La Disquette TREMPIN MICRO contient tous les programmes du numéro, ainsi que les sources trop longs pour être publiés dans les colonnes de la revue.

GS, MON BEAU SOUCI .....	2
MINUPHONE (Combien coûte une communication ?) .....	3

<b>SPÉCIAL</b> LE BASIC APPLESOFT	
<b>GS</b> PARLE AUX OUTILS DU GS .....	7

<b>C TRÈS SIMPLE</b> Claude AUBRY poursuit son amusante initiation ...	11
(langage C sur le GS) Et Nestor améliore LE BON NOMBRE .....	15

CATAVISU (Petit utilitaire ProDOS permettant de visualiser l'occupation de la disquette) .....	17
POKE, COMMANDE APPLESOFT (Tout ce qu'il faut savoir) .....	21
D'UN BASIC À UN AUTRE (DE L'APPLESOFT AU GWBASIC) .....	28
STARTUP (Intéressant utilitaire ProDOS) .....	31

<b>SPÉCIAL</b> GS RAM ET GS RAM PLUS (extensions mémoire pour Apple IIGS) ....	39
<b>GS</b> HORLOGE GS (Attention ! il faut ProDOS 8 en mémoire !) .....	40

GUIDES DE FORMATION (Comment découvrir Excel ou PageMaker ?) .....	44
---	----

<b>GRAPHISME</b> Logiciel de dessin mis au point par l'un de nos lecteurs, <b>Yves BUONO</b> ARC.EN.CIEL mérite de retenir votre attention .....	45
---	----

EDUCATION ET JEUX .....	53
GRAPH.EXPRESS (Routine en mode TEXT) .....	54

<b>LA SOURIS ET L'ASSEMBLEUR</b> (Troisième partie) .....	55
---	----

<b>LE TRAITEMENT DE TEXTE DU CÔTÉ DES COMPATIBLES</b> .....	60
---	----

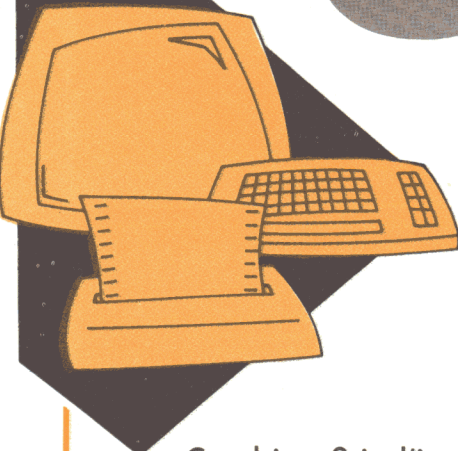
YVAN KOENIG RÉPOND À NOS LECTEURS .....	63
METTEZ UN GS DANS VOTRE APPLE II .....	69
NESTOR (SAGESSE OBLIGE !) .....	71
QUESTIONS et MEA CULPA (n°14) .....	72
QUOI EST OÙ ? .....	73
QUESTIONS... RÉPONSES (suite) .....	74
BULLETIN D'ABONNEMENT .....	75
LES LIVRES .....	27,30,70,76



# GS

## MON BEAU SOUCI

---



Nos lectrices et lecteurs sont perplexes. Certains (peu, eu égard au nombre considérable de machines vendues en France) ont choisi de troquer leur vieil Apple contre un GS flambant neuf. D'autres ont opté pour un MACINTOSH ou pour un compatible IBM.

Combien ? je l'ignore. Par contre, je sais que la plupart des lectrices et lecteurs de *Tremplin Micro* ont préféré conserver leur Apple II+ ou IIe, en attendant d'autres offres, plus alléchantes. On ne saurait leur donner tort.

Il est possible, actuellement, sur le marché de l'occasion, de s'offrir une configuration complète pour une somme relativement modique. Par ailleurs, grâce à des cartes comme la TRANSWARP d'*Applied Engineering*, un Apple IIe devient instantanément un Apple Turbo, capable — le son et le graphique en moins, bien sûr — de rivaliser avec un GS de la première couvée.

Les gens qui programment (ils sont nombreux parmi nos correspondants) ou désirent s'initier à ce jeu captivant qu'est la "domestication" d'un microprocesseur, prennent autant de plaisir à tapoter sur le clavier d'un Apple IIe ou d'un Apple IIc que sur celui d'un GS.

Et puis, avouons-le, il est plus agréable de travailler sous ProDOS 8 que sous ProDOS 16 — certes plus performant (on ne saurait raisonnablement le nier), mais terriblement gourmand en temps de chargement.

Enfin, une fois de plus, Apple mise surtout sur les nouveaux MAC (oh ! les belles machines !) parce que la firme de Cupertino espère pénétrer en force dans les entreprises. Comme nous aimons bien la pomme, nous lui souhaitons de réussir, mais en regrettant qu'il ne reste plus, pour les vieux de la vieille, que de rares pépites... concédés aux prix des pépites.

Bonnes vacances et profitez donc d'icelles pour vous adonner à l'étude passionnante de l'assembleur !

**Guy-HACHETTE.**



# MINUPHONE

**V** OICI un petit programme qui pourra rendre d'éminents services aux plus bavards utilisateurs du téléphone. Vous en connaissez sûrement dans votre entourage ! Gageons que Pierre Privat et son *Minuphone* vous aideront à réduire très sensiblement le montant de vos factures téléphoniques. En attendant, suiez donc un peu sur la frappe du programme !

*Rappelons qu'il ne faut surtout pas taper les valeurs indiquées en couleur : elles permettent aux heureux possesseurs de notre disquette **SIGNATURE** (une idée d'Yvan KOENIG) de contrôler chacune des lignes de leur programme.*

## SIGNATURE

est indispensable. Ce logiciel fait gagner un temps précieux en évitant à nos lectrices et lecteurs de fastidieuses recherches en cas d'erreur (et qui n'en commet jamais ?).

## SIGNATURE

ne coûte que 30 F (bulletin de commande à la fin de la revue).

(suite page 4)

```

110 D$ = CHR$(4):BIP$ = CHR$(7): TEXT      84DF
115 ONERR GOTO 805                          6CED
120 PRINT D$;"PR£3": PRINT : HOME         A25E
125 :                                       003A
130 REM ** LECTURE TAXE DE BASE **
135 :                                       003A
140 PRINT D$;"OPEN TAXE": PRINT D$;"READ TAXE": I
      NPUT T: PRINT D$;"CLOSE TAXE": PRINT  3117
145 :                                       003A
150 REM ** MENU GENERAL **
155 :                                       003A
160 HOME : GOSUB 400                        3C15
165 VTAB 6: HTAB 28: PRINT "1 --> "Lundi au Vendr
      edi"                                  E416
170 PRINT : HTAB 28: PRINT "2 --> "Samedi"  D809
175 PRINT : HTAB 28: PRINT "3 --> "Dimanches et F
      ètes"                                  F36B
180 PRINT : HTAB 28: PRINT "4 --> "Changer le pri
      x de la taxe"                          4E0C
185 PRINT : HTAB 28: PRINT "5 --> "Terminer"  BBBF
190 MX = 5: GOSUB 430                       6E2B
195 ON R GOTO 215,280,345,805,785          DB70
200 :                                       003A
205 REM ** TARIF SEMAINE **
210 :                                       003A
215 HOME : GOSUB 400                        3C15
220 VTAB 5: HTAB 30: INVERSE : PRINT " LUNDI au V
      ENDREDI ": NORMAL : PRINT            0BC8
225 PRINT "TARIF EN VIGUEUR:"              0CA8
230 PRINT : HTAB 20: PRINT "1 --> "ROUGE (8h00-12
      h30 et 13h30-18h00)"                 3F81
235 PRINT : HTAB 20: PRINT "2 --> "BLANC (12h30-1
      3h30 et 18h00-21h30)"               BB8E
240 PRINT : HTAB 20: PRINT "3 --> "BLEU (6h00-8h0
      0 et 21h30-22h30)"                  F1F6
245 PRINT : HTAB 20: PRINT "4 --> "BLEU NUIT (22h

```



```

30-6h00)"
250 MX = 4: GOSUB 430
255 ON R GOSUB 475,480,485,490
260 GOTO 510
265 :
270 REM ** TARIF SAMEDI **
275 :
280 HOME : GOSUB 400
285 VTAB 5: HTAB 36: INVERSE : PRINT " SAMEDI " :
NORMAL : PRINT
290 PRINT "TARIF EN VIGUEUR:"
295 PRINT : HTAB 20: PRINT "1 --> "ROUGE (8h00-12
h30)"
300 PRINT : HTAB 20: PRINT "2 --> "BLANC (12h30-1
3h30)"
305 PRINT : HTAB 20: PRINT "3 --> "BLEU (6h00-8h0
0 et 13h30-22h30)"
310 PRINT : HTAB 20: PRINT "4 --> "BLEU NUIT (22h
30-6h00)"
315 MX = 4: GOSUB 430
320 ON R GOSUB 475,480,485,490
325 GOTO 510
330 :
335 REM ** TARIF DIMANCHE **
340 :
345 HOME : GOSUB 400
350 VTAB 5: HTAB 30: INVERSE : PRINT " DIMANCHES
et FETES " : NORMAL : PRINT
355 PRINT "TARIF EN VIGUEUR:"
360 PRINT : PRINT : HTAB 24: PRINT "1 --> "BLEU (
6h00-22h30)"
365 PRINT : PRINT : HTAB 24: PRINT "2 --> "BLEU N
UIT (22h30-6h00)"
370 MX = 2: GOSUB 430
375 ON R GOSUB 485,490
380 GOTO 510
385 :
390 REM ** AFFICHAGE HAUT **
395 :
400 VTAB 1: PRINT "Taxe: ";T;" F";: HTAB 22: PRIN
T "MINUPHONE - Pierre PRIVAT, Mars 1987"
405 FOR I = 1 TO 80: PRINT "=";: NEXT I: PRINT
410 RETURN
415 :
420 REM ** AFFICHAGE BAS **
425 :
430 VTAB 19: FOR I = 1 TO 80: PRINT "_";: NEXT I
435 VTAB 21: HTAB 26: PRINT "<ESC> pour retour au
menu"
440 VTAB 22: HTAB 33: PRINT "Votre choix ? ";: GE
T R$
445 IF R$ = CHR$(27) OR R$ = CHR$(3) THEN 160
450 R = VAL (R$): IF R < 1 OR R > MX THEN PRINT
BIP$: GOTO 440

```

2163  
512A  
55B4  
0341  
003A

003A  
3C15

F3DE  
0CA8

115B

3A69

ACF7

2163  
512A  
55B4  
0341  
003A

003A  
3C15

E401  
0CA8

B4F8

C559  
9728  
0C20  
0341  
003A

003A

C007  
589D  
63B1  
003A

003A  
4911

FA3C

4BC7  
1A6E

7E97

# MINUPHONE

(suite)

## LIGNE 115 (PAGE 3)

Utile lors de la première utilisation (ou encore si le fichier a été effacé par erreur). Cet ONERR GOTO envoie directement à la création du fichier TXT: **TAXE.**

## LIGNE 140 (PAGE 3)

Lit la taxe de base PTT en vigueur dans le fichier **TAXE.**

## LIGNE 190 (PAGE 3)

**MX** est une variable qui, en fonction des différents menus, limite le chiffre-réponse au nombre de réponses possibles.

## LIGNES 220 à 375 et LIGNES 475 à 490

Bien sûr à modifier si l'Administration décide de changer l'actuel système de tarification. Le coefficient **TX** est multiplicateur de la durée en secondes par unité de base (voir la ligne 615).



## LIGNES 560 à 590

A modifier si les PTT changent la durée des communications (**UN** correspond au nombre de secondes).

## LIGNE 670

Il est possible de régler la précision en faisant varier la valeur-limite de **Z**.

## LIGNES 675 et 695

La touche «espace» a-t-elle été pressée ?

## LIGNES 820 à 850

L'utilisation d'une instruction **INPUT**, évidemment plus simple, aurait forcément conduit à des erreurs de type "EXTRA IGNORED", en raison du risque d'utilisation d'une virgule à la place du point traditionnel (taxe souvent en francs et centimes... et utilisateurs plus ou moins au courant de la syntaxe Applesoft !).

```
455 RETURN 63B1
460 : 003A
465 REM ** COEFFICIENTS TARIFS **
470 : 003A
475 TX = 1:TX$ = "ROUGE": RETURN CF38
480 TX = 1.5:TX$ = "BLANC": RETURN 3E79
485 TX = 2:TX$ = "BLEU": RETURN 3BDF
490 TX = 3:TX$ = "BLEU NUIT": RETURN 3D40
495 : 003A
500 REM ** MENU ZONES **
505 : 003A
510 HOME : GOSUB 400 3C15
515 VTAB 5: PRINT "Numéro de zone (voir carte annuaire) :": PRINT 7B21
520 PRINT : HTAB 22: PRINT "1 --> "Communication locale" ECF5
525 PRINT : HTAB 22: PRINT "2 --> "Communication voisinage (A)" 3F9D
530 PRINT : HTAB 22: PRINT "3 --> "Communication voisinage (B)" 7E9F
535 PRINT : HTAB 22: PRINT "4 --> "Communication voisinage (C)" 99A1
540 PRINT : HTAB 22: PRINT "5 --> "Communication distante" B4A5
545 MX = 5: GOSUB 430 6E2B
550 ON R GOTO 570,575,580,585,590 907B
555 : 003A
560 REM ** COEFFICIENTS ZONES **
565 : 003A
570 UN = 360:UN$ = "LOCALE": GOTO 610 454D
575 UN = 72:UN$ = "VOISINAGE (A)": GOTO 610 11C4
580 UN = 45:UN$ = "VOISINAGE (B)": GOTO 610 54C5
585 UN = 24:UN$ = "VOISINAGE (C)": GOTO 610 BFC3
590 UN = 12:UN$ = "DISTANTE": GOTO 610 C3C3
595 : 003A
600 REM ** CALCULS ET AFFICHAGE HORLOGE **
605 : 003A
610 HE = 0:MI = 0:SE = 0:SC = 0:P = 0:U = 1: HOME : GOSUB 400: GOSUB 710 90EA
615 UX = UN * TX 5896
620 VTAB 4: PRINT "Communication: ";UN$;: HTAB 62 : PRINT "Tarif: ";TX$ 21C6
625 PRINT : HTAB 22: PRINT "(Une unité toutes les ";UX;" secondes)": PRINT : FOR I = 1 TO 80: PRINT ".":; NEXT I: PRINT CAF9
630 VTAB 20: FOR I = 1 TO 80: PRINT "_":; NEXT I 4809
635 VTAB 22: HTAB 18: PRINT "<RET> pour démarrer et <ESPACE> pour arrêter ";: GET R$: IF R$ < > CHR$(13) THEN PRINT BIP$: GOTO 635 FB23
640 HE$ = STR$(HE):MI$ = STR$(MI):SE$ = STR$(SE) 1C65
645 IF SE < 10 THEN SE$ = "0" + SE$ 62BF
650 IF MI < 10 THEN MI$ = "0" + MI$ 44B9
655 IF HE < 10 THEN HE$ = "0" + HE$ 539E
```

(suite page 6)



```

660 VTAB 17: HTAB 8: PRINT HE$;" ":"MI$;" ":"SE$;;
    HTAB 72: PRINT U
665 P = U * T: VTAB 12: HTAB 40: PRINT P
670 FOR Z = 1 TO 67: REM ** Réglage horloge **
675 IF PEEK ( - 16368) = 160 THEN 705
680 NEXT Z:SE = SE + 1:SC = SC + 1:U = INT (SC /
    UX) + 1
685 IF SE = 60 THEN MI = MI + 1:SE = 0
690 IF MI = 60 THEN HE = HE + 1:MI = 0
695 IF PEEK ( - 16368) = 160 THEN 705
700 GOTO 640
705 VTAB 22: CALL - 958: HTAB 21: PRINT "Appuyer
    sur une touche pour continuer "; GET R$: PR
    INT : GOTO 740
710 VTAB 17: HTAB 1: PRINT "Temps: 00:00:00"; HT
    AB 63: PRINT "Unités : "0 "
715 VTAB 12: HTAB 32: PRINT "Prix : "*****"
720 RETURN
725 :
730 REM ** MENU DE FIN **
735 :
740 HOME : GOSUB 400: VTAB 8: PRINT "Voulez-vous:"
745 VTAB 10: HTAB 22: PRINT "1 --> "Recommencer a
    vec les mêmes paramètres"
750 PRINT : HTAB 22: PRINT "2 --> "Recommencer en
    changeant les paramètres"
755 PRINT : HTAB 22: PRINT "3 --> "Quitter"
760 MX = 3: GOSUB 430
765 ON R GOTO 610,160,785
770 :
775 REM ** BYE BYE **
780 :
785 HOME : VTAB 10: HTAB 36: PRINT "Terminé...":
    END
790 :
795 REM ** SAISIE/ENREGISTREMENT TAXE DE BASE **
800 :
805 HOME : GOSUB 400
810 VTAB 21: FOR I = 1 TO 80: PRINT "_"; NEXT :
    PRINT
815 VTAB 10: HTAB 25: PRINT "Prix de la taxe de b
    ase ? "; CALL - 868
820 GET TA$: IF TA$ = CHR$ (13) THEN 855
825 IF TA$ = "," THEN TA$ = "." : GOTO 845
830 IF TA$ = "." THEN 845
835 IF TA$ = CHR$ (8) OR TA$ = CHR$ (127) THEN
    T2$ = "": GOTO 815
840 IF ASC (TA$) < 48 OR ASC (TA$) > 57 THEN 820
845 PRINT TA$;
850 T2$ = T2$ + TA$: GOTO 820
855 T = VAL (T2$): IF T = 0 THEN 815
860 PRINT : PRINT D$;"OPEN TAXE": PRINT D$;"WRITE
    TAXE": PRINT T: PRINT D$;"CLOSE TAXE": PRINT
865 GOTO 160

```

73D2  
724A  
0B0A  
BB78

A3C8  
5B06  
11F0  
BB78  
0345

9350

185E  
FAEF  
63B1  
003A

003A  
FAFA

8A26

8C2C  
EE7F  
9129  
82DB  
003A

003A

89D3  
003A

003A  
3C15

30B5

EE50  
15E9  
72EB  
690D

90D4  
A731  
CFAE  
2524  
FFA1

BAB0  
3942

# MINUPHONE

(suite)

Les programmes  
paraissant dans

## TREPLIN MICRO

restent la propriété  
de leurs auteurs.

Leur reproduction est  
évidemment  
autorisée, mais ils ne  
peuvent être  
commercialisés.

Leurs auteurs sont  
seuls habilités à en  
autoriser l'éventuelle  
publication dans un  
recueil.

## AVIS À NOS LECTEURS

**TREPLIN MICRO** invite  
ses Lectrices et Lecteurs à  
adopter **ProDOS**, système  
d'exploitation beaucoup  
plus performant que le  
DOS 3.3.

A propos de **ProDOS**, se  
méfier des anciennes ver-  
sions et utiliser, sinon la  
plus récente, du moins  
celle portant la référence  
1.1.1.

# LE BASIC APPLESOFT

## — parle aux outils du GS —

### 1. Quel environnement de programmation ?

Supposons que vous ne disposiez pas du système de développement APW pour pouvoir écrire efficacement en assembleur — et avec des macro-instructions — un programme avec une interface-utilisateur de type Mac (le Bureau Electronique sur l'écran Super Haute Résolution avec la Souris).

Avec APW, on écrirait cette application en la décomposant en plusieurs segments de programmes et de données, puis en une simple commande ASML PG.SRC, on obtiendrait l'assemblage et l'édition de liens de ces segments sous forme d'un fichier chargeable et relogeable automatiquement sous le système d'exploitation ProDOS 16.

Notre hypothèse de développement ici est le BASIC.SYSTEM que vous aurez mis sur une disquette SYSTEM contenant au minimum les fichiers suivants :

PRODOS	SYS
SYSTEM	DIR
BASIC.SYSTEM	SYS
STARTUP	BAS

Le sous-répertoire ou dossier SYSTEM contenant au minimum :

P8	SYS
P16	\$F9
START	S16
SYSTEM.SETUP	DIR
TOOLS	DIR
DESK.ACCS	DIR
FONTS	DIR

où on trouvera les Outils à charger en MEV à la demande de l'application, une collection d'accessoires de bureau et une collection de polices de caractères. Le programme START est celui qui va être exécuté au démarrage.

Il s'agit en général du lanceur d'application (Apple System Launcher). Avec une disquette système MOUSE.DESK, vous mettrez BASIC.SYSTEM en œuvre en cliquant sur son icône.

En appelant BASIC.SYSTEM, on se place automatiquement sous ProDOS 8, grâce au fichier P8 qui contient la version ProDOS 1.1.1 déjà connue, mais que l'on appellera dorénavant ProDOS 8.

Pour assembler un sous-programme en langage machine, entrez dans le mini-assembleur par le classique CALL - 151 suivi, dès la sollicitation de l'\* , par un !

Une fois les instructions mnémoniques tapées, retournez au Moniteur en tapant simplement 'Return'.

Pour sauvegarder sur disquette, rappelez BASIC par Ctrl-C, puis tapez **BSAVE PG.OBJ,A\$adr, L\$long.**

Le fichier ainsi obtenu pourra être rechargé par un **BLOAD PG.OBJ.**

Il est prévu dans APW une commande qui permet de transformer :

— un fichier relogeable et exécutable sous ProDOS 16 (type EXE ou S16) en :

— un fichier équivalent en adresse absolue exécutable sous ProDOS 8 (donc chargeable dans un programme en BASIC). C'est la commande MAKEBIN nom du fichier objet.

C'est pourquoi dans cet article vous trouverez le listing du source PG.SRC écrit sous APW ; la commande ASML PG.SRC a généré PG.OBJ grâce à la directive KEEP PG.OBJ ; puis la commande MAKEBIN PG.OBJ a généré le même code que celui obtenu grâce au mini-assembleur.

Le source PG.SRC a été écrit délibérément sans macro-instructions pour pouvoir le lire et le taper sous la même forme en mini-assembleur. Le mini-assembleur ne vous oblige pas à taper le signe \$ pour les données en hexadécimal comme l'exige APW.

### 2. L'exemple

Nous avons choisi d'implanter le programme BASIC à sa place normale (à partir de \$800) en supposant qu'il n'ira pas au-delà de \$2000 pour son code et ses données numériques. En \$2000 nous chargeons PG.OBJ le sous-programme de dialogue avec les outils, en supposant qu'il n'ira pas au-delà de \$3000 ; cette dernière adresse a été choisie arbitrairement comme début des pages Zéro que réclament les outils pour démarrer leur travail. QuickDraw en utilise trois et EventManager une. (suite page 8)



**PG.OBJ**

\$CALL-151

\*0=x

\*0=m

0=m 0=x 1=LCbank (0/1)

```
00/2000: 18      CLC
00/2001: FB      XCE
00/2002: C2 30   REP £30
00/2004: 20 12 20 JSR 2012
00/2007: 20 7A 20 JSR 207A
00/200A: 20 B3 20 JSR 20B3
00/200D: 38      SEC
00/200E: FB      XCE
00/200F: E2 30   SEP £30
00/2011: 60      RTS
```

```
00/2012: A2 01 02 LDX £0201
00/2015: 22 00 00 E1 JSL E10000
00/2019: F4 00 00   PEA 0000
00/201C: A2 02 02 LDX £0202
00/201F: 22 00 00 E1 JSL E10000
00/2023: 68      PLA
00/2024: 8D E4 20 STA 20E4
00/2027: F4 00 00   PEA 0000
00/202A: F4 00 00   PEA 0000
00/202D: F4 E1 00   PEA 00E1
00/2030: F4 00 20   PEA 2000
00/2033: A2 02 1A LDX £1A02
00/2036: 22 00 00 E1 JSL E10000
00/203A: A2 02 10 LDX £1002
00/203D: 22 00 00 E1 JSL E10000
00/2041: F4 00 30   PEA 3000
00/2044: F4 00 00   PEA 0000
00/2047: F4 A0 00   PEA 00A0
00/204A: AD E4 20 LDA 20E4
00/204D: 48      PHA
00/204E: A2 04 02 LDX £0204
00/2051: 22 00 00 E1 JSL E10000
00/2055: F4 00 33   PEA 3300
00/2058: F4 14 00 PEA 0014
00/2058: F4 00 00   PEA 0000
00/205E: F4 40 01   PEA 0140
00/2061: F4 00 00   PEA 0000
00/2064: F4 C8 00   PEA 00C8
00/2067: AD E4 20 LDA 20E4
00/206A: 48      PHA
00/206B: A2 06 02 LDX £0206
00/206E: 22 00 00 E1 JSL E10000
```

**PG.SRC**

TITLE 'PG.SRC'

LIST ON

ABSADDR ON

SYMBOL ON

ORG \$002000

KEEP PG.OBJ

PG

START

```
CLC ;mise en mode natif
XCE ;avec les registres
REP £$30 ;sur 16 bits
JSR DEMARR ;démarrage des Outils
JSR BOUCLE ;boucle de gestion des évènements
JSR FERMER ;fermeture
SEC
XCE ;mise en mode émulation
SEP £$30 ;remise des registres
RTS ;en mode 8 bits
END
```

DEMARR START

USING COMMUN

```
LDX £$201 ;TLStartup
JSL $E10000
PEA $0000
LDX £$202 ;MMStartup
JSL $E10000
PLA
STA ID
PEA $0000
PEA $0000
PEA $00E1
PEA $2000
LDX £$1A02 ;FindHandle de $E10000
JSL $E10000
LDX £$1002 ;DisposeHandle
JSL $E10000
PEA $3000 ;début des Pages Zéros QuickDraw
PEA $0080 ;SCB:620 pixels/ligne
PEA $00A0 ;nbre d'octets/ligne
LDA ID
PHA
LDX £$204 ;QDStartup
JSL $E10000
PEA $3300 ;Page Zéro Event Manager
PEA $0014 ;long. maxi file d'évènements
PEA $0000 ;X min
PEA $0280 ;X max du curseur
PEA $0000 ;Y min
PEA $00C8 ;Y min
LDA ID
PHA
LDX £$206 ;EMStartup
JSL $E10000
```

```

00/2072: A2 04 91   LDX £9104
00/2075: 22 00 00 E1 JSL E10000
00/2079: 60        RTS

```

```

LDX ££9104   ;ShowCursor
JSL $E10000
RTS

```

END

BOUCLE START

```

00/207A: F4 00 00   PEA 0000
00/207D: F4 FF FF   PEA FFFF
00/2080: A9 00 00   LDA £0000
00/2083: 48        PHA
00/2084: A9 E6 20   LDA £20E6
00/2087: 48        PHA
00/2088: A2 06 0A   LDX £0A06
00/208B: 22 00 00 E1 JSL E10000
00/208F: 68        PLA
00/2090: F0 E8     BEQ 207A £-18£
00/2092: AD E6 20   LDA 20E6
00/2095: C9 01 00   CMP £0001
00/2098: D0 13     BNE 20AD £+13£

```

```

USING COMMUN
PEA $0000
PEA $FFFF   ;tous les évènements accessibles
LDA £$0000
PHA
LDA £EVENT   ;l'adresse de EVENT sur 4 octets
PHA
LDX £$A06   ;GetNextEvent
JSL $E10000
PLA
BEQ BOUCLE   ;aucun évènement?
LDA EVENT   ;lequel?
CMP £$0001   ;bouton-souris enfoncé?
BNE NOT1

```

;simple exemple de traitement:

```

00/209A: F4 00 00   PEA 0000
00/209D: A2 04 86   LDX £8604
00/20A0: 22 00 00 E1 JSL E10000
00/20A4: A2 04 15   LDX £1504
00/20A7: 22 00 00 E1 JSL E10000

```

```

PEA $0000
LDX ££8604   ;Random renvoie sur la pile
JSL $E10000   ;un nombre aléatoire sur 2 octets
LDX ££1504   ;ClearScreen met ce nombre dans
JSL $E10000   ;tous les octets associés aux pixels

```

```

00/20AB: 80 CD     BRA 207A £-33£
00/20AD: C9 03 00   CMP £0003
00/20B0: D0 C8     BNE 207A £-38£
00/20B2: 60        RTS

```

```

BRA BOUCLE
NOT1 CMP £$0003   ;touche appuyée
BNE BOUCLE
RTS

```

END

FERMER START

```

00/20B3: A2 06 03   LDX £0306
00/20B6: 22 00 00 E1 JSL E10000
00/20BA: A2 04 03   LDX £0304
00/20BD: 22 00 00 E1 JSL E10000
00/20C1: F4 00 00   PEA 0000
00/20C4: F4 00 00   PEA 0000
00/20C7: F4 00 00   PEA 0000
00/20CA: F4 00 80   PEA 8000
00/20CD: AD E4 20   LDA 20E4
00/20D0: 48        PHA
00/20D1: F4 13 C0   PEA C013
00/20D4: F4 E1 00   PEA 00E1
00/20D7: F4 00 20   PEA 2000
00/20DA: A2 02 09   LDX £0902
00/20DD: 22 00 00 E1 JSL E10000
00/20E1: 68        PLA

```

```

USING COMMUN
LDX ££306   ;EMShutDown
JSL $E10000
LDX ££304   ;QDShutDown
JSL $E10000
PEA $0000
PEA $0000
PEA $0000   ;taille du bloc à réserver
PEA $8000   ; $008000
LDA ID
PHA
PEA $C013   ;fixe,verrouillé,sur un seul banc
PEA $00E1   ; en $E12000
PEA $2000   ;
LDX ££902   ;NewHandle
JSL $E10000
PLA

```

(suite page 10)



00/20E2: 68      PLA  
00/20E3: 60      RTS

PLA  
RTS

END

Nous avons choisi cette présentation, peu orthodoxe il est vrai, pour rendre ce programme accessible à tous les utilisateurs de l'APPLE IIGS, y compris ceux qui ne disposent pas du système APW (relire l'article de Nicole BRÉAUD-POULIQUEN).

```
COMMUN DATA            ;données communes
ID            DS 2        ;N°d'identité sous lequel le
              ;            Memory Manager connaît ce programme
EVENT        DS 2        ;résultats de GetNextEvent
TOUCHE       DS 4
QUAND        DS 4
OU            DS 4
MODIFIER     DS 2
              END
```

## PR.BASIC

```
10 PRINT "Le but de cette application est de montrer comment faire"
20 PRINT "appel aux OUTILS du GS pour utiliser l'environnement type Mac"
30 PRINT "-le bureau electronique en page Super Haute Résolution"
40 PRINT "en restent COMPATIBLE avec BASIC APPLESOFT et ProDOS 8"
45 PRINT CHR$(4)"BLOAD PG.0BJ"
50 PRINT "Le S/P est chargé en $002000"
60 PRINT "Les pages Zéro nécessaires aux Outils sont en $003000"
62 PRINT : INVERSE : PRINT " 620 OU 340 " ; : NORMAL
63 SCB = 8261: REM $2045
64 MAXX = 8287: REM $205F-2060
65 INPUT N: IF N = 0 THEN END
66 IF N = 620 THEN POKE SCB,128: POKE MAXX + 1,2: POKE MAXX,128
67 IF N = 340 THEN POKE SCB,0: POKE MAXX + 1,1: POKE MAXX,64
70 CALL 8192: REM $2000
75 ID = 8420: REM $20E4
80 PRINT "-démarrage du Tool Locator"
90 PRINT "-démarrage du Memory Manager"
100 TYPE = PEEK (ID + 1) / 16: MAINID = PEEK (ID)
110 PRINT "N° d'identification du bloc courant "; TYPE: "00": MAINID
120 PRINT "ProDOS alias $3001 possède la mémoire de l'écran Super Haute Résolution"
130 PRINT "dont QuickDraw devra se servir"
140 PRINT "on va chercher le handle de cette zone $E10000"
150 PRINT "puis le désallouer pour laisser QuickDraw s'en servir:"
160 PRINT "-FindHandle"
170 PRINT "-DisposeHandle "
180 PRINT "-démarrage de QuickDraw"
190 PRINT "-démarrage de l'EventManager"
220 PRINT "La boucle principale du S/P traite les événements par:"
230 PRINT "-GetNextEvent pour les faire connaître au programme"
240 PRINT "qui attend que le bouton-souris soit enfoncé pour "
250 PRINT "faire varier la couleur du fond"
260 PRINT "et qui termine la boucle dès qu'une touche est pressée"
290 PRINT "Le retour au programme BASIC appelant nécessite:"
292 PRINT "-fermeture des outils sauf le Memory Manager"
294 PRINT "-réallocation de la mémoire Super Haute Résolution par:"
296 PRINT "-NewHandle "
300 GOTO 62
```

### 3. Perspectives

Sachant qu'il faut laisser aux outils des zones de travail ou Pages Zéro dans le banc \$00, qu'un programme en BASIC APPLESOFT ne s'exécute que s'il est dans le banc \$00, que pour le Memory Manager, c'est ProDOS 8 qui est propriétaire des blocs :

\$000800-\$00B800 et  
\$010800-\$01B800 puis  
\$E02000-\$E06000 et  
E12000-E1A000,

le travail de recherche des meilleures adresses d'implantation est ouvert.

Les outils seraient donc tous utilisables en cohabitation avec du BASIC ? A vous maintenant de faire travailler Window Manager, Menu Manager, Control Manager, Dialog Manager, etc. ...

# C TRÈS SIMPLE ! (suite)

## C PAS POSSIBLE

Imaginez une course d'ordinateurs : un MO5 sorti tout droit de chez Thomson et de l'Éducation nationale, mon vieil APPLE IIe tout vermoulu, avec PASCAL UCSD, mon tout-nouveau-tout-beau GS équipé de son C de compétition.

Même parcours pour tout le monde : calculer 10 fois (pour faire bonne mesure) les nombres premiers apparaissant dans la suite d'entiers de 1 à 10 000.

### Résultat des courses

**MO5 + BASIC** : à 56 minutes, il a fallu abattre l'ordinateur : il souffrait trop.

**APPLE IIe + PASCAL** : 14 minutes, pas mal, l'APPLE IIe !... on a quand même dû refaire la peinture.

**APPLE II GS + C** : 6 secondes !

Des commentaires, des questions, oui, non, boff ! OK ! je vous sens plein de courage : voici la saga des variables, l'épopée des objets informatiques.

## C COURAGEUX

Fini de s'amuser ! aujourd'hui, on attaque les choses sérieuses et ceux qui liront cet article jusqu'au bout ne seront plus jamais les mêmes : une majesté certaine illuminera leurs traits fatigués. Quant aux ÉLUS qui auront tout compris, un C d'argent brillera sur leur noble front.

Une façon élégante d'aborder sérieusement un langage consiste à définir les objets (les variables) qu'il manipule et la façon dont il les manipule.

Le très vénérable SAINT NIKLAUS WIRTH, auteur du très regretté PASCAL, nous apprend que ALGORITHM + DATABASE = PROGRAM. Ce qui, en bon français, signifie à peu près : si tu sais écrire PRINTF "HELLO" et que tu ne sais pas comment STRUCTURER tes données, tes programmes font rire ou pleurer, selon qu'on les regarde ou qu'on les utilise.

## C LA RENCONTRE DES QUATRE TYPES

La puissance d'un langage informatique est très

liée à la façon dont il peut manipuler les objets (les variables) qui lui sont confiés.

BASIC, lui, n'est pas méchant. Je rappelle aux inconscients, à ceux qui se seraient aventurés sur ces lignes bourbeuses sans repasser leur BASIC ÉLÉMENTAIRE, que ce langage délicieux possède en gros quatre types prédéfinis : ENTIERS, RÉELS, CHAÎNES DE CARACTÈRES, et un quatrième discret et caché comme une taupe, le type LOGIQUE.

Ces objets, Basic sait les organiser en tableau et en fichier, donc en structures linéaires simples.

La déclaration du type d'un objet Basic se fait au moment de son utilisation par la présence d'un suffixe \$, %, ou par une absence de suffixe selon que l'on veut déclarer une chaîne, un entier ou un réel.

Seule la structure TABLEAU échappe à la règle et doit être déclarée avant l'utilisation par l'instruction DIM.

Ce petit rappel ayant distrait les BASICOIS et les BASICOISES, passons vite à C que je vais définir par ses différences.

(suite page 12)



On pourrait dire que le langage est d'autant plus puissant et efficace qu'il possède de types d'objets différents, et de possibilités d'organiser ces objets en structures complexes.

## C TYPÉ

**C** est un langage "typé" il permet l'utilisation d'objets de types divers selon que l'on veut privilégier la vitesse d'exécution, l'encombrement en mémoire ou encore la puissance de calcul.

A la différence du BASIC, toute variable devant être utilisée dans le cours du programme, doit être déclarée à l'avance. Ceci permet une gestion efficace de mémoire et un gain de temps appréciable pour le compilateur.

En Basic nous écrivons :	tandis qu'en C :
10 X% = 5	int X;
20 .....	X = 5;
30 .....	.....

Remarquons au passage que l'instruction d'affectation (=) est la même en BASIC et en **C**. Elle se prononce X reçoit 5.

L'inconvénient de l'écriture **C** est qu'elle ne permet pas d'identifier immédiatement le type d'une variable dans le cours d'un programme : il faut remonter à sa déclaration et celle-ci est parfois lointaine.

Son avantage principal est de pouvoir limiter la durée de vie (on dit plutôt la visibilité ou la portée) d'une variable à la zone de programme strictement nécessaire et de limiter ainsi les mélanges de variables bien connus des programmeurs BASIC.

Dans le programme ci-dessous il existe deux variables x distinctes et elles possèdent une valeur différente selon que l'on se trouve dans le bloc 1 ou 2.

```

main ()
é          /* un premier bloc */
int x = 5; /* on fabrique une variable entière */
printf("%d ", x); /* et on l'écrit... */
           /* on reviendra sur la syntaxe */
é          /* un deuxième bloc */
int x = 7; /* une deuxième variable x */
printf("%d ", x); /* la première n'est plus VISIBLE */
é          /* fin du deuxième bloc retour au 1 */

printf(' %dçn",x); /* coucou la variable 1 est revenue */
é          /* fin du bloc 1 */
    
```

En **C** les /\* et \*/ délimitent les lignes de commentaire.

A part la syntaxe farfelue de printf (farfelue, mais extrêmement pratique), si vous avez exécuté ce programme vous avez tout compris.

Il produit l'affichage 5 7 5.

On dit que les variables x sont des variables LOCALES. On distingue en **C** les variables LOCALES et les variables GLOBALES qui appartiennent à tout un programme.

Malgré cela — ou peut-être même à cause de cela —, les langages "typés" font toujours un peu peur aux programmeurs BASIC. Ces derniers considèrent comme une injuste punition ces déclarations "inutiles".

Je vais essayer de leur prouver que, loin d'être pénalisante, cette nécessité permet une grande souplesse dans les techniques de programmation.

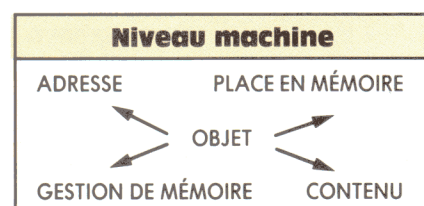
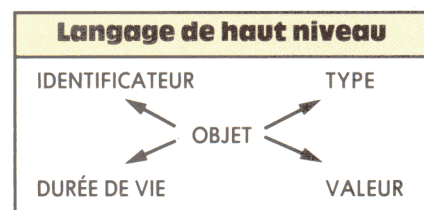
## LA DÉCLARATION DES VARIABLES EN C

Pour exister, dans la majorité des langages un objet informatique (une variable par exemple) doit être défini par quatre éléments :

1. Un identificateur, le "nom" de l'objet (a, b, toto).
2. Un type associé (nombre entier, réel, chaîne de caractères).
3. Une valeur associée (10, 15, 12, "machin").
4. Une "classe de mémorisation" définissant la durée de vie.

(En Basic, la classe de mémorisation est une et indivisible puisqu'une variable est définie pour tout le programme).

Cette structure quadri-parties correspond à la réalité physique de l'organisation mémoire dans les entrailles profondes de la machine (pouah !).



Le langage **C** est pratiquement le seul à pouvoir passer de l'un à l'autre niveau de représentation sans problème, ce qui lui a valu sa réputation d'assembleur haut niveau.

Nous allons passer en revue les différents constituants tels qu'ils existent en **C** APW.

## LES IDENTIFICATEURS

Il faut bien le reconnaître, on prend de mauvaises habitudes en BASIC APPLESOFT. Comme ce langage ne considère que deux lettres significatives pour nommer les variables, comme de plus il n'apprécie pas tellement les identificateurs contenant des mots réservés, on en vient très vite à programmer avec des A\$ des I et des N qui ne rappellent en rien l'objet manipulé.

**C** lui, considère TOUS les caractères de l'identificateur comme significatifs et distingue même majuscules et minuscules : Toto et toto sont des identificateurs différents en **C**.

Il faut en profiter pour nommer intelligemment les

variables que l'on utilise. Nous verrons par la suite que le type pointeur constitue à lui seul un outil prodigieux pour rendre un programme totalement incompréhensible ; raison de plus pour bien choisir les identificateurs et écrire TVA plutôt que Z.

## LES TYPES PRÉDÉFINIS

Le **C** de l'APPLE IIGS se distingue par une opulence de types assez remarquable.

Les types numériques en particulier sont très nombreux : alors que le type réel de Basic Applesoft présente juste assez de précision pour me permettre de calculer mes droits d'auteur sur ces articles, **C** offre pour les types numériques une représentation entière, signée, ou non-signée, courte, longue, étendue, flottante, double, énumération, etc. j'en passe et des plus spécialisés. On doit pouvoir calculer la distance de la Terre à la Lune en centimètres !

Le tableau ci-dessous résume les principaux types de **C** APW je vous laisse la joie de les découvrir.

TYPE	VALEURS		Nb OCTETS
char	- 128	+ 128	1
unsigned char	0	+ 255	1
short	- 32767	+ 32767	2
unsigned short	0	+ 65535	2
int	- 32767	+ 32767	2
unsigned int	0	+ 65535	2
long	- 2147483648	+ 2147483647	4
unsigned long	0	+ 4294967295	4
enum	dépend de l'énumération		1, 2, 4
*	pointeur vers une variable		4
float	IEEE simple précision		4
double	IEEE double précision		8
comp	Valeur entière format ROMTOOL		8
extended	Valeur réelle format ROMTOOL		10

Certains d'entre vous ont dû le remarquer : pas de type chaîne de caractères prédéfini en **C**. En fait, **C** considère qu'une chaîne de caractères ressemble trop à un tableau de caractères pour s'encombrer d'un type spécifique.

**C** n'est pas très regardant quant aux mélanges de types : le type CHAR (caractère) s'accommode par exemple très bien de transporter des valeurs entières.

Le programmeur en **C** doit être vigilant : s'il recherche l'efficacité maximale et partant, le code

le plus court possible, il peut être rapidement amené à réaliser des programmes générateurs de migraine.

A titre de démonstration le petit programme (que vous trouverez en haut de la page 14) affiche les lettres de A à Z et leur code ASCII en décimal et en hexadécimal, amusez-vous à changer le type **char** en type **int** par exemple et voyez ce qui se passe.

Une petite explication pour ceux qui s'endorment au fond de la salle : ce programme... (suite page 14)



## PROGRAMME :

```
/* Mélange de types */  
main ( )  
{  
    unsigned char i;  
  
    for (i='A'; i <= 'Z'; i++);  
    printf("%c %d %x %n", i, i, i);  
  
}
```

utilise une instruction très proche du FOR... NEXT du Basic :

FOR (x=début ; x <= fin ; x++) é...bloc...è  
Le bloc délimité par é et è sera exécuté (fin -début) fois, x étant incrémenté à chaque boucle grâce à l'opérateur unaire ++.

L'instruction **printf** que nous avons déjà rencontrée permet un affichage formaté des données il s'agit d'une sorte de super PRINTUSING.

**printf(contrôle, arg1, arg2, ...);**

**arg** est une variable dont le mode d'affichage sera conditionné par les caractères de conversion contenus dans la chaîne contrôle.

Caractère de conversion	Affichage de arg
%d	décimal
%o	octal
%x	hexadécimal
%u	décimal non signé
%c	caractère
%s	chaîne
%e	notation scientifique
%f	notation décimale
%g	choisit e ou f (la plus courte)

## LE TYPE POINTEUR

Le type pointeur fait les délices et les souffrances du programmeur en **C**. Il provoque plus de dégâts que l'alcool et le tabac réunis dans notre communauté élitiste : avec les pointeurs on peut tout faire.

Un pointeur est un entier long désignant des adresses de variables. Supposez l'existence d'une variable de type entier nommée "Objet" et déclarée par **int Objet = 5;**

L'opérateur & (ampersand) permet de déterminer l'adresse de cette variable.

**Po = &Objet** affecte à la variable **Po** l'adresse de **Objet**. On dit que **Po** est un pointeur sur la variable **Objet**.

L'opérateur \* (astérisque) placé devant le nom

du pointeur donne la valeur de la variable pointée : dans notre exemple \*Po est égal à 5 et peut remplacer **Objet** dans toutes ses applications.

Nous reviendrons longuement sur les pointeurs dans les articles suivants... Pour l'instant, souvenez-vous seulement qu'un pointeur non initialisé sur une variable, pointe n'importe où et que ça peut faire très mal au ProDOS.

## Dans notre prochain numéro :

*Un pointeur fou a pu s'introduire dans l'espace mémoire. Tombera-t-il entre les griffes du trop célèbre docteur MALLOC ou pourra-t-il rejoindre TOPLEVEL dans le Kernel ?... Vous le saurez en lisant le prochain article de :*

**C** très simple, mais ça se complique, dans **TREMLIN MICRO**.  
A bientôt... **Claude AUBRY.**

Le premier volet de cette série d'articles a paru dans le numéro 14 de **TREMLIN MICRO**

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

# Le bon nombre (version 1)

Reprenons l'exercice n°1 (Tremplin Micro n°14), mais en essayant de l'améliorer sur 3 points :

- Ouverture du programme sur un écran vide (home) ;
- Modification de la fourchette en fonction des limites déterminées par les rentrées successives de nombres ;
- Limitation du nombre de coups.

## EN BASIC

Cela se traduirait par les lignes suivantes :

Micro-programme élémentaire.

```

10 PRINT CHR$(4)"PR£3": PRINT
15 X = 69:C = 0:S = 20:A = 1:Z = 100
20 HOME
25 S = S - 1: IF S > 0 THEN PRINT " ____ ";: GOTO 25
30 PRINT : PRINT :S = 20:C = C + 1
35 IF C = 10 THEN PRINT "VOUS AVEZ PERDU!": END
40 PRINT "TROUVEZ UN NOMBRE COMPRIS ENTRE "A" ET "Z";
45 INPUT " ";N: PRINT
50 IF N = X THEN PRINT "C'EST LE NOMBRE - TERMINE": END
55 IF N > X THEN 70
60 PRINT "IL EST TROP PETIT": IF N > (A) THEN A = N + 1:
  GOTO 25
65 GOTO 75
70 PRINT "IL EST TROP GRAND": IF N < Z THEN Z = N - 1:
  GOTO 25
75 C = C - 1: GOTO 25
  
```

## REMARQUES

La ligne 25\* pourrait être avantageusement remplacée par :

```
FOR S = 1 TO 19 : PRINT " ____ ";: NEXT
```

Mais la ressemblance avec le programme en **C** se révélerait moins évidente. Pourquoi le **C = C - 1** de la ligne 75 ? Simplement pour ne pas comptabiliser un nombre sans influence sur la fourchette proposée. Si cette dernière est 65, 70 et que l'on rentre un nombre plus petit ou égal 65... ou plus grand ou égal 75, on aboutira à la ligne 75 : un coup pour rien !

(suite page 16)

\* Elle affiche 19 fois 4 soulignements. Pour une ligne complète (80 caractères), il faudrait S = 21.



## ÉQUIVALENCE EN LANGAGE C

```
main( )
é
```

```
int x, n, c, s, a, z;
x = 69;
c = 0;
s = 20;
a = 1;
z = 100;
```

```
CL:
```

```
printf("çf");
```

```
LG:
```

```
--s;
    if (s > 0) é
        printf("_____");
        goto LG;
è
printf("çnçn");
s = 20;
++c;
    if (c == 10) é
        printf("VOUS AVEZ PERDU!çn");
        exit (0);
è
printf("TROUVEZ UN NOMBRE COMPRIS ENTRE %d ET %d -> ", a, z);
scanf("%d", &n);
if (n == x) é
    printf("C'EST LE NOMBRE - TERMINEçn");
    exit (0);
è
if (n < x) é
    printf("IL EST TROP PETITçn");
    if (n > a) é
        a = n + 1;
è
    goto LG;
è
if (n > x) é
    printf("IL EST TROP GRANDçn");
    if (n < z) é
        z = n - 1;
è
    goto LG;
è
n = 0;
--c;
    goto LG;
è
```

### PROCESSUS

- **CL** : Nouvel écran.
- **LG** : Une ligne de soulignement suivie par deux retours.
- **S** est réinitialisé à 20.
- **C** est incrémenté. Quand il est égal à 10, message et fin de programme.
- Choix du nombre
- Si le nombre choisi est égal à x, c'est terminé !
- S'il est plus petit que x, on l'annonce.
- Si, dans ce cas-là, n est plus grand que a, on fait a = n + 1. Retour à LG.
- Si n est plus grand que x, on l'annonce.
- Si, dans ce cas-là, n est plus petit que z, on fait z = n - 1.
- Sinon, n = 0 et C = C - 1, puis goto LG.

**printf("çf");** vide l'écran.

**--s;** signifie S = S - 1.

**+ +c;** signifie C = C + 1.

**çn** indique un retour ligne.

**çnçn** en génère donc 2.

Fonctionne aussi  
sur l'APPLE IIGS

# CATAVISU

Cet utilitaire permettra aux heureux utilisateurs de ProDOS de visualiser l'occupation de la disquette présente dans le lecteur (port n°6). Le programme baptisé "CATAVISU" affiche à l'écran une table d'occupation avec :

- Sur la ligne horizontale supérieure les pistes numérotées de 00 à \$22 ce qui fait de 00 à 34 en décimal soit 35 pistes.
- Sur la colonne la plus à gauche les numéros de blocs numérotés de 0 à 7 soit 8 blocs.

## TABLE D'OCCUPATION DE LA DISQUETTE

```

000000000000000000001111111111111111222
0123456789ABCDEF0123456789ABCDEF012
0
1  .«---- indique un bloc libre
2
3
4  *«---- indique un bloc occupé
5
6
7

```

L'utilitaire débute à l'adresse \$9000 avec l'affi-

chage des lignes de présentation. Puis, grâce au MLI et à la commande READ BLOCK qui effectue la lecture d'un bloc de 512 octets, le programme va chercher sur la disquette le VTOC (ou BIT-MAP) la piste 00 bloc 06. Là, il trouve, sous la forme de 35 octets, l'occupation de la disquette avec pour règle :

- Un bloc libre représenté par un 1 ;
- Un bloc occupé représenté par un 0.

Il suffit donc de lire ces octets et de les transférer dans une partie de la mémoire de l'APPLE (\$2000) pour les traiter. Ce traitement consiste à prendre un octet et à l'analyser selon les règles ci-dessus, puis à afficher un "." ou un "\*" suivant le cas. ATTENTION ! cet utilitaire ne fonctionne qu'avec ProDOS. En effet la notion de blocs n'est utilisée que par ce système d'exploitation (1 bloc = 2 pistes).

Pour utiliser le programme, il suffit de taper :

- BRUN CATAVISU.C
- CALL 1013
- et d'utiliser alors le "&" pour appeler l'utilitaire.

## CATAVISU.S

### ASSEMBLAGE

### par ProCODE

#### SAUVEGARDE :

BSAVE CATAVISU.C,A\$9000,L300

0	BASCALC	EQU	\$FBC1	
1	BLOCK	EQU	\$80	
2	CH	EQU	\$24	
3	CV	EQU	\$25	
4	HOME	EQU	\$FC58	
5	MLI	EQU	\$BF00	;Adresse du MLI
6	PAGE1	EQU	\$C054	
7	STRUT	EQU	\$DB3A	
8	VTAB	EQU	\$FC22	
9	*			
10		ORG	\$9000	
11	*			

(suite page 18)



```

9000: A9 00      12      LDA £$00      ; Change HIMEM de valeur pour
9002: 85 73      13      STA $73       ; protéger le programme.
9004: A9 94      14      LDA £$94      ;
9006: 85 74      15      STA $74       ;
          16      *
9008: A9 13      17      LDA £<DEBUT   ; Permet d'utiliser "&" pour
900A: 8D F6 03   18      STA $3F6      ; appeler
900D: A9 90      19      LDA £>DEBUT   ; la fonction CATAVISU.
900F: 8D F7 03   20      STA $3F7
9012: 60         21      RTS
          22      *
          23      *
9013: 20 58 FC   24      DEBUT JSR HOME   ; Efface l'écran.
9016: A9 07      25      LDA £$07      ; Ligne 7. Prép. l'affich. vert.
9018: 8D 2B 91   26      STA LINDEP    ; des nbrs représ. les blocs.
901B: A2 B0      27      LDX £$B0      ; Stocke le caractère "0" dans X.
901D: A9 01      28      LDA £$01      ; Colonne n°1.
901F: 2C 54 C0   29      BIT PAGE1    ; Active la mém. princ. d'écran.
9022: 2A         30      ROL
9023: 4A         31      LSR           ; Divise par 2.
9024: A8         32      TAY
9025: 5A         33      VERTIC PHY
9026: AD 2B 91   34      LDA LINDEP    ; N° de ligne d'affic. dans A.
9029: 20 C1 FB   35      JSR BASCALC
902C: 7A         36      PLY
902D: 8A         37      TXA
902E: 91 28      38      STA (£28),Y  ; Affic. du carac. de A.
9030: E8         39      INX           ; Prépare affic. carac. suivant.
9031: AD 2B 91   40      LDA LINDEP
9034: C9 0E      41      CMP £$0E     ; Dern. ligne pour l'affichage ?
9036: F0 05      42      BEQ SUITE    ; Oui, alors on affiche le titre.

9038: EE 2B 91   43      INC LINDEP    ; Non: on descend d'une ligne...
903B: 80 E8      44      BRA VERTIC   ; et retour pour nouvel affic.
          45      *
903D: A9 02      46      SUITE LDA £$02      ; Positionne le curseur vert.
903F: A0 04      47      LDY £$04     ; Positionne le curseur horiz.
9041: 20 AE 90   48      JSR POS
9044: A9 08      49      LDA £<CH3    ; Affiche la chaîne de caractère.
9046: A0 91      50      LDY £>CH3    ; "TABLE D'OCCUPATION DE LA DIS-
          51      ; QUETTE"
9048: 20 3A DB   52      JSR STROUT   ;
          53      *
904B: A9 05      54      LDA £$05     ; On recommence avec de nouvelles
          55      ; positions.
904D: A0 04      56      LDY £$04     ;
904F: 20 AE 90   57      JSR POS      ;
9052: A9 C0      58      LDA £<CH1    ; "000000000000000011111111111111
          59      ; 11222"
9054: A0 90      60      LDY £>CH1    ;
9056: 20 3A DB   61      JSR STROUT   ;
          62      *
9059: A9 06      63      LDA £$06     ;
905B: A0 04      64      LDY £$04     ;

```

```

905D: 20 AE 90 65 JSR POS ;
9060: A9 E4 66 LDA £<CH2 ; "0123456789ABCDEF0123456789ABCD
67 ; EF012"
9062: A0 90 68 LDY £>CH2 ;
9064: 20 3A DB 69 JSR STROUT ;
70 *
9067: 20 00 BF 71 JSR MLI ; Appel du MLI.
906A: 80 72 DFB BLOCK ; Appel fonction READ BLOCK MLI.
906B: B6 90 73 DA PARAM ; Fournit les paramètres.
74 *
906D: A2 00 75 LDX £$00
906F: 86 FE 76 STX $FE ; Prépare le compteur permettant
77 ; la lecture des blocs.
9071: A9 04 78 LDA £$04
9073: 85 4C 79 STA $4C ; Positionne horizontalement le
80 ; curseur pour afficher.
9075: A9 07 81 ENC LDA £$07
9077: 85 4D 82 STA $4D ; Positionne vert. le curseur
83 ; pour l'affichage.
84 *
9079: A6 FE 85 COUCOU LDX $FE ; Stocke la valeur du pointeur
86 ; d'octets dans X.
907B: 1E 00 20 87 ASL $2000,X ; Décale vers la gauche.
907E: 90 0C 88 BCC SUITE1 ; Si C=0: bloc occupé.
9080: 20 AA 90 89 JSR POS1 ; Sinon saut pour prép. l'affic.
9083: A9 BE 90 LDA £<CH5 ; Affiche...
9085: A0 90 91 LDY £>CH5 ; alors
9087: 20 3A DB 92 JSR STROUT ; un "."
908A: 80 0A 93 BRA CONT ; Opération suivante.
94 *
908C: 20 AA 90 95 SUITE1 JSR POS1 ; Le bloc est occupé.
908F: A9 BC 96 LDA £<CH4 ; On prépare l'affichage.
9091: A0 90 97 LDY £>CH4 ;
9093: 20 3A DB 98 JSR STROUT ; d'un "*".
99 *
9096: E6 4D 100 CONT INC $4D ; On incrémente la ligne
9098: A5 4D 101 LDA $4D ; et compare avec la
909A: C9 0F 102 CMP £$F ; dernière ligne possible.
909C: D0 DB 103 BNE COUCOU ; <> on a d'autres blocs à affic.
909E: E6 FE 104 INC $FE ; sinon on incrémente pour
105 ; passer à la piste suivante.
90A0: E6 4C 106 INC $4C ; Incr. pos. hor. du curseur
90A2: A5 4C 107 LDA $4C ; et comparaison avec
90A4: C9 27 108 CMP £$27 ; la dernière position possible
90A6: F0 0D 109 BEQ FIN ; Si différent, on a d'autres
90A8: 80 CB 110 BRA ENC ; pistes à traduire.
111 *
90AA: A5 4D 112 POS1 LDA $4D
90AC: A4 4C 113 LDY $4C
114 *
90AE: 85 25 115 POS STA CV
90B0: 20 22 FC 116 JSR VTAB
90B3: 84 24 117 STY CH
118 *

```

(Suite page 20)



90B5: 60           119 FIN           RTS  
                   120 \*  
                   121 \*  
 90B6: 03           122 PARAM        HEX 03  
 90B7: 60           123 UNIT         HEX 60  
 90B8: 00 20       124 BUFFER       DA \$02000  
                   125  
 90BA: 06 00       126 BLOCS        HEX 0600  
                   127 \*  
 90BC: AA           128 CH4         ASC "\*"   
 90BD: 00           129               HEX 00  
 90BE: AE           130 CH5         ASC "."   
 90BF: 00           131               HEX 00

**VOUS POUVEZ AUSSI TAPER DIRECTEMENT  
LES CODES, EN PASSANT PAR CALL -151**

90C0: B0 B0 B0  
       B0 B0 B0  
       B0 B0 B0  
       B0 B0 B0  
       B0 B0 B0  
       B0 B1 B1  
       B1 B1 B1  
       B1 B1 B1  
       B1 B1 B1  
       B1 B1 B1  
       B1 B1 B2

9000: A9 00 85 73 A9 94 85 74 A9 13 8D F6 03 A9 90 8D  
 9010: F7 03 60 20 58 FC A9 07 8D 2B 91 A2 B0 A9 01 2C  
 9020: 54 C0 2A 4A A8 5A AD 2B 91 20 C1 FB 7A BA 91 28  
 9030: E8 AD 2B 91 C9 0E F0 05 EE 2B 91 80 E8 A9 02 A0  
 9040: 04 20 AE 90 A9 08 A0 91 20 3A DB A9 05 A0 04 20  
 9050: AE 90 A9 C0 A0 90 20 3A DB A9 06 A0 04 20 AE 90  
 9060: A9 E4 A0 90 20 3A DB 20 00 BF 80 B6 90 A2 00 86

3EDF  
 7FEF  
 9C8C  
 D97A  
 52EB  
 6ABD  
 70BF

90E3: 00           132 CH1         ASC "00000000000000001111111111111111222"  
                   133               HEX 00

90E4: B0 B1 B2  
       B3 B4 B5  
       B6 B7 B8  
       B9 C1 C2  
       C3 C4 C5  
       C6 B0 B1  
       B2 B3 B4  
       B5 B6 B7  
       B8 B9 C1  
       C2 C3 C4  
       C5 C6 B0  
       B1 B2

9070: FE A9 04 85 4C A9 07 85 4D A6 FE 1E 00 20 90 0C  
 9080: 20 AA 90 A9 BE A0 90 20 3A DB 80 0A 20 AA 90 A9  
 9090: BC A0 90 20 3A DB E6 4D A5 4D C9 0F D0 DB E6 FE  
 90A0: E6 4C A5 4C C9 27 F0 0D 80 CB A5 4D A4 4C 85 25  
 90B0: 20 22 FC 84 24 60 03 60 00 20 06 00 AA 00 AE 00  
 90C0: B0 B0 B0 B0 B0 B0 B0 B0 B0 B0 B0 B0 B0 B0 B0  
 90D0: B1 B1 B1 B1 B1 B1 B1 B1 B1 B1 B1 B1 B1 B1 B1

B57C  
 31B3  
 BFAD  
 ECE7  
 4827  
 C200  
 0710

9107: 00           134 CH2         ASC "0123456789ABCDEF0123456789ABCDEF012"  
                   135               HEX 00

9108: D4 C1 C2  
       CC C5 A0  
       C4 A7 CF  
       C3 C3 D5  
       D0 C1 D4  
       C9 CF CE  
       A0 C4 C5  
       A0 CC C1  
       A0 C4 C9  
       D3 D1 D5  
       C5 D4 D4  
       C5

90E0: B2 B2 B2 00 B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 C1 C2  
 90F0: C3 C4 C5 C6 B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 C1 C2  
 9100: C3 C4 C5 C6 B0 B1 B2 00 D4 C1 C2 CC C5 A0 C4 A7  
 9110: CF C3 C3 D5 D0 C1 D4 C9 CF CE A0 C4 C5 A0 CC C1  
 9120: A0 C4 C9 D3 D1 D5 C5 D4 D4 C5 00 00

98A6  
 0AA2  
 D518  
 C84B  
 F9DB

912A: 00           136 CH3         ASC "TABLE D'OCCUPATION DE LA DISQUETTE"  
                   137               HEX 00  
 912B: 00           138 LINDEP       HEX 00



# POKE, COMMANDE APPLESOFT

## L'INSTRUCTION POKE DU BASIC APPLESOFT

POKE est une instruction faisant partie de l'interpréteur du Basic AppleSoft. L'adresse \$D072 pointe \$E77A qui est l'entrée (adresse-1) de la routine POKE. C'est une instruction qui permet de placer à partir d'un programme Basic par exemple, une valeur dans un emplacement mémoire dont l'adresse est spécifiée. L'exécution d'un POKE revient à sélectionner une adresse correspondant à un emplacement mémoire et à y placer une valeur déterminée. L'instruction POKE possède par ailleurs la caractéristique de sélection d'un commutateur logique du système. Si l'adresse déclarée après le POKE n'existe pas, l'instruction sera ignorée. POKE peut être déclaré en mode direct (clavier) ou en mode indirect (programme Basic).

### Syntaxe :

POKE ADRESSE, VALEUR où  
ADRESSE est une expression arithmétique comprise entre 0 et 65535 inclus, et VALEUR une expression arithmétique comprise entre 0 et 255 inclus.

### APPLESOFT ROUTINE POKE :

E77B - JSR \$E746 Transfert de l'adresse du POKE dans \$50-\$51.  
Au retour du sous-programme, la valeur du POKE se trouvera dans le registre X.

E77E - TXA Copie la valeur du POKE dans A.

E77F - LDY \$500 Initialise le registre Y,

E781 - STA (\$50),Y et copie la valeur du POKE à l'emplacement mémoire spécifié.

E783 - RTS

Retour au sous-programme appelant.

### Remarques :

- Une adresse négative est l'équivalent d'une adresse positive obtenue en lui additionnant la valeur 65536.
- L'équivalent du POKE en langage machine est STA (STore Accumulator).
- En programmation Basic, les valeurs déclarées après POKE devront être en base 10, tandis que le langage machine requiert les valeurs en base 16.
- POKE place dans l'adresse indiquée un nombre binaire codé sur 8 bits, équivalent au nombre décimal déclaré.
- Si l'adresse déclarée ne correspond pas à une adresse mémoire valide, alors l'instruction sera ignorée. Si le POKE est adressé à une mémoire morte (ROM), celui-ci restera sans effet.
- Si les paramètres déclarés après POKE sont hors des limites fixées par l'AppleSoft, un message d'erreur sera retourné : ILLEGAL QUANTITY ERROR.
- POKE peut modifier le contenu d'une adresse ou commuter des switches, encore appelés commutateurs logiques. Ce dernier mode est sélectionné en écrivant ou en lisant, à certaines adresses servant de point d'entrée à ces commutateurs logiques.

## POKE DE LA PAGE ZÉRO

La page zéro contient un certain nombre de valeurs et pointeurs utilisés soit par l'interpréteur AppleSoft, soit par le système d'exploitation. Une totale maîtrise et une manipulation astucieuse de ces vecteurs... (suite page 22)



peut dans bien des cas simplifier la programmation. La suite des pokes n'est nullement exhaustive, mais permet au débutant d'avoir à sa portée un outil de travail non négligeable.

### Transmission du pointeur pour l'appel du USR

Avec la fonction AppleSoft USR, qui appelle un sous-programme en langage machine, l'argument doit être transmis à travers l'accumulateur flottant. Il est alors nécessaire de sauvegarder l'adresse d'appel en page zéro, adresses \$000B-\$000C, avec un JMP (4C) dans \$000A. Le résultat est, par la suite, placé dans l'accumulateur flottant. Cette transmission de paramètres peut se faire par une série de pokes aux adresses précitées, de la page zéro :

POKE 10,76: POKE 12,ADR/256 : POKE 11,ADR  
- PEEK (12) \* 256 où ADR est la valeur de l'adresse mémoire.

### Code couleur en graphique GR

La valeur de la couleur en mode GR est stockée à l'adresse \$0030 (48 - COLOR). Cette valeur peut être modifiée en changeant le code de la couleur :

POKE 48,COLOR \* 17 où Color équivaut au code couleur choisi.

### Mode d'affichage AppleSoft

Le mode d'affichage vidéo est sous contrôle de la fonction INVFLG, qui conditionne le mode d'affichage de l'Apple. D'après la valeur contenue dans l'adresse \$32 (50), l'affichage passe en mode normal (£\$FF), inverse (£\$3F) ou clignotant (£\$7F).

POKE 50,255      Mode NORMAL  
POKE 50,63        Mode INVERSE  
POKE 50,127      Mode FLASH

### Instructions interdites en mode direct

Certaines des commandes AppleSoft comme OPEN, READ, WRITE, APPEND ou POSITION, ne peuvent être adressées en mode direct (clavier) sans rencontrer le message d'erreur NOT DIRECT COMMAND. Pour modifier cette situation imposée, placez en mode direct, avant un ordre GOTO ou GOSUB et sur la même ligne, à l'emplacement où le programme contient une des instructions précitée : POKE 51,128.

Par la suite, l'Apple se positionne de lui-même dans le mode initial par l'intermédiaire du Monitor.

### Sauvegarde des registres PC, A, X et Y

Lorsqu'il est nécessaire de sauvegarder des adresses du compteur ordinal (registre PC) et des regis-

tres A, X et Y, avant l'appel d'un sous-programme du Monitor, il faudra effectuer la série suivante de pokes :

POKE 58,Adr1 : POKE 59,Adr2 : POKE 69,Adr3 :  
POKE 70,Adr4 : POKE 71,Adr5

Les adresses 58,59 désignent le pointeur de l'adresse de sauvegarde du PC, sous la forme PC octet poids faible, et PC octet poids fort. Les adresses 69,70 et 71 sont respectivement destinées à la sauvegarde de l'accumulateur (A), du registre X, et du registre Y.

### Paramètres avant l'appel d'une routine du Monitor

Les adresses \$3C-\$43 (60 à 67) sont utilisées pour transmettre des paramètres à une routine du Monitor, pour déplacer ou vérifier un espace mémoire par exemple (MOVE, VERIFY etc.). Avant l'exécution de la routine, il sera nécessaire de mettre le registre d'index Y à zéro. L'instruction CALL - 610 effectue cette mise à zéro. Exemple de transmission de paramètres et de mise à zéro préalable du registre Y :

CALL - 610 : POKE 60... 67, Valeur : CALL - 468  
où CALL - 468 est l'appel au sous-programme MOVE (\$FE2C) pour déplacer une zone mémoire vers une autre.

### Mise à zéro du registre d'état (PS)

L'adresse \$48 (72), status du registre d'état PS, est utilisée pour annuler le registre d'état PS au retour d'un sous-programme du Monitor par exemple. Avec l'utilisation de la routine de S.H.Lam, il sera nécessaire par la suite de mettre à zéro le registre d'état du microprocesseur : POKE 72,0.

### Début du programme Basic

TXTTAB est l'adresse de début du programme AppleSoft, fixée à la valeur \$0801 par défaut. Pour l'utilisation d'une adresse autre, placez en début de votre programme AppleSoft la ligne suivante :

POKE 103,1 : POKE 104,Adr/256 : POKE Adr,0

Votre programme AppleSoft sera ainsi logé à l'adresse Adr.

### Modification de LOMEM

LOMEM, adresse de début de la zone des variables simples et fin du programme Basic, se laisse facilement déplacer. Pour ce faire, modifiez les adresses \$69 et \$6A (105 et 106).

POKE 106,Adr/256  
POKE 105,Adr-PEEK (106) \* 256  
Fixe la valeur de LOMEM à l'adresse décimale Adr.

## Modification de HIMEM

HIMEM, adresse de début de la zone des variables de chaînes de caractères et fin du DOS 3.3, se laisse facilement déplacer. Il faudra modifier les adresses \$73 et \$74 (115 et 116) :

```
POKE 116,Adr/256
```

```
POKE 115,Adr-PEEK (116) * 256
```

Fixe la valeur de HIMEM à l'adresse décimale Adr.

## Mode direct

CURLIN se trouve aux adresses \$75-\$76 et contient la valeur \$FF en mode direct (à partir du clavier). Pendant l'exécution d'un programme Basic AppleSoft, le numéro de la ligne courante en exécution se trouve réactualisé à cette adresse. En modifiant le contenu de cette adresse, il devient alors possible d'exécuter en mode direct, des instructions telles que INPUT ou GET. En adressant un POKE 118,0 sur la même ligne que l'instruction à transmettre en mode direct, l'Apple ne retournera pas de message d'erreur :

```
POKE 118,0 : INPUT "ESSAI !! ";E$
```

## Fin du programme AppleSoft

Pour modifier le pointeur de la fin du programme Basic, lors d'un ajout d'un programme en langage machine par exemple, il faudra modifier les adresses \$AF et \$B0 (175 et 176) :

```
POKE 176,Adr/256
```

```
POKE 175,Adr-PEEK (176) * 256
```

Fixe la fin du programme Basic à sa nouvelle valeur décimale Adr.

## Annule ONERR

POKE 216,0 annule l'instruction ONERR déclarée au début d'un programme Basic.

## HGR ou HGR2 ?

En mode graphique haute résolution, il est intéressant de pouvoir utiliser l'une ou l'autre des deux pages graphiques, sans pour autant devoir l'afficher. Si un programme doit modifier une partie d'une image, sans pour autant l'afficher, il est possible d'intervenir directement sur la page graphique en modifiant le contenu de l'adresse \$E6.

L'adresse \$E6 (230) est utilisée pour mémoriser le numéro de la page graphique où doivent agir les tracés des instructions en cours :

```
POKE 230,32 sélectionne la page HGR ($2000)
```

```
POKE 230,64 sélectionne la page HGR2 ($4000)
```

```
POKE 230,96 sélectionne une page fictive à l'adresse $6000. Cette page ne pourra être affichée directement, mais devra au préalable être déplacée en mémoire soit dans la page HGR, soit dans la page HGR2.
```

## POKE DE LA PAGE 2

La page 2 occupe les adresses \$0200-\$02FF, et sert de tampon d'entrée de caractères frappés au clavier. La routine de S.H.Lam utilise le tampon clavier par l'intermédiaire de pokes pour installer en mémoire des sous-programmes en langage machine. Cette même routine s'utilise aussi à partir d'un programme Basic pour rentrer des instructions via le Monitor :

```
10 Y$ = EXEMPLE$
```

```
20 Y$ = Y$ + " N D9C6G"
```

```
30 FOR I = 1 TO LEN (Y$) : POKE 511,ASC (MID$ (Y$,I,1)) + 128 : NEXT
```

```
40 POKE 72,0 : CALL - 144
```

où EXEMPLE\$ pourrait être "2000 < 6000.7FFM", et copie la page graphique fictive débutant à partir

de l'adresse \$6000 et se terminant à l'adresse \$7FFF, dans la page HGR (PAGE 1, adresses \$2000-\$3FFF).

### ou encore :

```
EXEMPLE$ = "CO81 N CO81 N D000 < D000.FFFM"  
copiera la ROM dans la carte d'extension mémoire.
```

### ou encore :

```
EXEMPLE$ = "300: 20 3A FF 20 3A FF 20 3A FF 60"  
copiera à l'adresse $0300 un sous-programme quelconque. Dans l'exemple cité, CALL 768 exécute le sous-programme, l'Apple émettra 3 bips.
```

La liste n'est pas exhaustive, et toute commande du Monitor pourra être exécutée par ce procédé.

## POKE DE LA PAGE 3

Cette page réserve une partie de ses adresses au programmeur (\$0300-\$03CF), et l'autre partie... (suite page 24)



se partage entre le DOS (\$03D0-\$03EF) et la ROM Monitor (\$03F0-\$03FF).

### Branchement en cas de BRK

Changement de l'adresse de branchement en cas de BReaK du moniteur :

POKE 1009,Adr/256  
POKE 1008,Adr-PEEK (1009) \* 256

### Branchement en cas de RESET

Changement de l'adresse de branchement en cas d'un RESET:

POKE 1011,Adr/256  
POKE 1010,Adr-PEEK (1011) \* 256

ou encore : POKE 1012,0 qui provoque un démarrage à froid du système, après un RESET.

### Branchement en cas du & ampersand

Modifier l'adresse de saut par le & ampersand est possible en modifiant les pointeurs aux adresses \$03F6 et \$03F7 (1014 et 1015). Normalement, pointe l'adresse \$FF58.

POKE 1015,Adr/256  
POKE 1014,Adr-PEEK (1015) \* 256

### Touches Ctrl-Y

L'association des touches CTRL-Y permet, lorsque vous êtes en mode Monitor de faire un saut au sous-programme, dont l'adresse est stockée à \$03F9 et \$3FA (1017 et 1018). Normalement, pointe l'adresse \$FF65.

POKE 1018,Adr/256  
POKE 1017,Adr-PEEK (1018) \* 256

## POKE DES PAGES 4 à 11

Les pages 4 à 7 sont réservées à l'espace mémoire de la page 1 texte, tandis que les pages 8 à 11 sont celles réservées à la page 2 texte. Elles représentent l'affichage de l'écran vidéo. Il est possible d'affecter un point quelconque de la surface de l'écran, en adressant un POKE à la bonne adresse. L'adressage se fait donc directement en page 1 texte aux points des coordonnées x et y. La valeur x étant comprise entre 0 et 39 et y entre 0 et 23. Le mode d'affichage pouvant lui-même être modifié d'après la variable MOD dans notre exemple.

MOD entre 0 et 63 = affichage INVERSE

MOD entre 64 et 127 = affichage FLASH  
MOD entre 128 et 191 = affichage NORMAL  
MOD entre 192 et 255 = affichage MINUSCULE

Dans le cas d'une écriture texte en page 2, il suffira de remplacer 1024 (\$400) par la valeur \$2048 (\$800). Dans ce cas le programme AppleSoft devra être relogé, en modifiant les adresses 103 et 104 (\$67 et \$68). Le début ne pourra pas se trouver avant l'adresse \$0C00, étant donné que la fin de la page 2 texte débute à l'adresse \$0BFF.

POKE 1024 + Adr + 128 \* Y - 984 \* INT (Y/8),MOD

## POKE POUR LE DOS 3.3

Les valeurs données par la suite sont uniquement valables pour un DOS 3.3 logé dans l'espace mémoire d'un système de 64 Ko de RAM. Pour une configuration différente, il suffit de retrancher 1024 octets aux adresses indiquées pour chaque Ko en moins.

Par défaut, le système d'exploitation termine le boot d'une disquette sur un fichier du type Basic (A). En modifiant simplement un octet à l'adresse 40514 du DOS, on peut booter sur un fichier binaire ou EXEC. Normalement, cet octet a comme valeur 06 pour un fichier Basic.

Pour initialiser sous DOS 3.3 une disquette avec un fichier de boot différent, tapez à partir du clavier le

poke en conséquence. Cette modification peut très bien se réaliser au niveau de la disquette même à l'aide d'un utilitaire dans le genre DISKFIXER ou autre. Pour ce faire, il faudra modifier le byte à la position relative \$42 (66), piste \$00 secteur \$0D de ladite disquette.

POKE 40514,52 permet un boot du système sur un fichier BINAIRE.  
POKE 40514,20 permet un boot du système sur un fichier EXEC.  
POKE 43140,73 : POKE 43160,68 : POKE 43212,82 vous ramène à un DOS standard, après modification de certains pointeurs.



## DRIVE et SLOT

- POKE 43624,D Sélectionne le drive ayant la valeur D (D = 1 pour le lecteur 1).
- POKE 43624,3 - PEEK (43624) Change de lecteur de disquettes.
- POKE 43626,S Sélectionne le slot de l'Apple (S = 6 pour slot 6).

## TABLE DU DOS

POKE 44452,255	Elimine la pause durant un catalogue (CATALOG).
POKE 44452,22	Restaure la pause lors du CATALOG.
POKE 43140,96	Annule la commande INIT du DOS.
POKE 43140,73	Restaure la fonction INIT du DOS.
POKE 43160,96	Annule la fonction DELETE du DOS.
POKE 43160,68	Restaure la fonction DELETE du DOS.
POKE 43212,96	Annule la commande RENAME du DOS.
POKE 43212,82	Restaure la fonction RENAME du DOS.
POKE 43144,96	Annule la fonction LOAD.
POKE 43144,76	Restaure la fonction LOAD.
POKE 43148,96	Annule la fonction SAVE.
POKE 43148,83	Restaure la fonction SAVE.
POKE 43152,96	Annule la fonction RUN.
POKE 43152,82	Restaure la fonction RUN.
POKE 43166,96	Annule la fonction LOCK.
POKE 43166,76	Restaure la fonction LOCK.
POKE 43170,96	Annule la fonction UNLOCK.
POKE 43170,85	Restaure la fonction UNLOCK.
POKE 43218,96	Annule la fonction CATALOG.
POKE 43218,67	Restaure la fonction CATALOG.
POKE 43247,96	Annule la fonction FP.
POKE 43247,70	Restaure la fonction FP.
POKE 43249,96	Annule la fonction INT.
POKE 43249,73	Restaure la fonction INT.
POKE 43257,96	Annule la fonction BLOAD.
POKE 43257,66	Restaure la fonction BLOAD.
POKE 43262,96	Annule la fonction BRUN.
POKE 43262,66	Restaure la fonction BRUN.
POKE 43266,96	Annule la fonction VERIFY.
POKE 43266,86	Restaure la fonction VERIFY.

## TABLE IOB

La table IOB est utilisée par la routine RWTS pour l'accès direct à un secteur donné de la disquette. La table IOB se trouve par défaut à l'adresse 47080 (\$B7E8) pour un DOS booté sur une configuration de 64 Ko. Ces paramètres peuvent être modifiés par la suite pour des besoins spécifiques.

IOB = 47080

POKE IOB+1,SLT \* 16 modifie le numéro du Slot actuel (modulo 16).

- POKE IOB+2,Dve modifie le numéro du drive actuel (Valeur de 1 à 2).
- POKE IOB+3,Vol modifie le numéro du volume actuel. Pour lire n'importe quel volume de disquette, il faudra donner à Vol la valeur 0 (Valeur de 0 à 254).
- POKE IOB+4,Piste modifie le numéro de la piste actuelle (Valeur de 0 à 34).
- POKE IOB+5,Secteur modifie le numéro du secteur actuel (Valeur de 0 à 15).
- POKE IOB+6,DCTLW modifie l'octet de poids faible de l'adresse de la table des caractéristiques du DOS (DCT).
- POKE IOB+7,DCTHG modifie l'octet de poids fort de l'adresse de la table des caractéristiques du DOS (DCT).
- POKE IOB+8,BUFFLW modifie l'octet de poids faible du tampon utilisé par le DOS.
- POKE IOB+9,BUFFHG modifie l'octet de poids fort du tampon utilisé par le DOS.
- POKE IOB+11,BYT Sert à sélectionner le nombre d'octets à modifier sur un secteur. Prend comme valeur \$00 pour modifier les 256 octets d'un secteur.
- POKE IOB+12,Mode modifie le travail de la tête de lecture :
- Mode = 0 positionne la tête de lecture.
  - Mode = 1 lecture.
  - Mode = 2 écriture.
  - Mode = 4 formatage.

## QUELQUES TRUCS EN VRAC

### Meilleure lisibilité en mode 80 colonnes

POKE - 16205,0 : POKE - 16203,0

### Impression des minuscules

POKE 243,32 force l'imprimante à l'impression de minuscules. Faire précéder le PRINT par POKE 243,32. Après l'instruction PRINT à mettre en minuscules, tapez NORMAL. L'adresse 243 (\$F3) ou ORMASK sert à l'APPLE-SOFT pour forcer le mode FLASH. *(suite page 26)*

## Sortie sur enregistrement cassette

POKE - 16352,0

## POKE sur deux octets

POKE adresse +1,V / 256 : POKE adresse,V - 256 \* PEEK (adresse +1) où adresse +1 est l'octet de poids fort, adresse est l'octet de poids faible, et V est la valeur à modifier.

## POKE divers

- POKE 44033,P met la VTOC du disk avant initialisation sur la piste numéro P (P = 13, met la VTOC sur la piste 13, secteur 0).
- POKE 37656,X effectue un saut de X lignes dans un programme.
- POKE 37658,X durée d'attente :
- si x 128 augmente la durée.
  - si x 128 diminue la durée.

## Langage Integer

Pour éviter lors d'un Boot à chaud de recharger à chaque fois le langage Integer dans la carte langage ou la carte d'extension mémoire, il faudra effectuer au préalable les trois POKE suivants :

POKE 49107,234 : POKE 40108,234 :  
POKE 49109,234

## Accès à une piste interdite

Si vous voulez pour une raison ou une autre, interdire l'accès à certaines pistes de votre disquette, il suffira de faire les POKE suivants :

POKE 44723,A \* 4 : POKE 44725,B \* 4  
où les variables A et B représentent les limites extrêmes des pistes accessibles pour la sauvegarde de vos fichiers.

## Modifier l'emplacement de la VTOC

La VTOC se trouve normalement sur la piste \$11 secteur \$00, et le Directory sur la piste \$11 secteurs \$15 à \$01 (DOS standard). Pour modifier cette disposition, il sera nécessaire de faire les POKE suivants :

POKE 44033,3 : POKE 44723,16 : POKE 44741,12 :  
POKE 44764,3 : POKE 46012,3

Dans notre exemple, la VTOC et le Directory seront placés sur la piste \$03, après initialisation d'une disquette. Cette dernière ne pourra plus être lue, ni bootée par un DOS 3.3 standard. Les programmes de copie usuels comme FID ou COPYA ne fonctionnent plus. Les paramètres pour un DOS classique sont respectivement 17, 12, 68, 17 et 17.

# APPLE // ProDOS GUIDE DU PROGRAMMEUR APPLESOFT

Par Marcel COTTINI

Vous ne tirerez pleinement profit de votre Apple que si vous connaissez parfaitement les multiples ressources de son nouveau système d'exploitation : ProDOS.

Après avoir lu l'excellent ouvrage de Marcel COTTINI, vous trouverez en ProDOS l'un de vos meilleurs alliés pour une programmation avancée.

### Extrait de la table des matières :

- Familiarisation du lecteur avec ProDOS.
- Les vecteurs particuliers sous ProDOS.
- Système DOS 3.3 ou ProDOS ?
- Monitor, interpréteur Applesoft et ProDOS.
- Le boot d'une disquette ProDOS.
- Occupation type d'une configuration Apple.
- Commandes typiques du Basic.System.
- La Basic System Global page.
- et bien d'autres sujets !

BULLETIN DE COMMANDE PAGE 75.



# Vient de paraître

Les Editeurs nous  
communiquent :

## • **IMAGES NUMÉRIQUES** (Jean-Baptiste Touchard)

D'abord cantonnées à certains secteurs de pointe, en raison de l'importance des calculs nécessaires pour les mettre en œuvre, les images numériques envahissent aujourd'hui les écrans des téléviseurs, des cinémas... et des micro-ordinateurs.

Ce livre se veut très pragmatique, voire terre à terre face à ce phénomène. Il a pour ambition de donner au lecteur les éléments directement utilisables. Le panorama présenté dans ce livre (vidéotex, petit formulaire de géométrie, les trois dimensions, la croisée des dimensions, les objets graphiques, leur saisie, puis leur traitement) n'a certes pas la prétention d'être exhaustif dans un domaine extrêmement vaste et mouvant. Il tente cependant de faire ressortir le fait que les algorithmes fondamentaux, utilisés en infographie, font rarement appel à des concepts très élaborés. Les exemples présentés dans le livre, bien qu'étant écrits dans des langages divers, s'inscrivent dans des syntaxes claires et facilement transposables.

Un bagage mathématique de bachelier scientifique permet de comprendre le plus grand nombre des concepts pré-

sentés. Bien que l'ouvrage présente l'image de synthèse de manière approfondie, il offre cependant différents niveaux de lecture :

- de nombreuses illustrations et légendes apportent, d'une part, une meilleure compréhension pour le débutant ;
- d'autre part, de multiples ouvertures vers des systèmes plus sophistiqués, comme les palettes graphiques, sont proposées au lecteur plus passionné.

**CEDIC/NATHAN**, 6-10, boulevard JOURDAN —  
75014 PARIS. (336 pages — 240 F)

## • **INITIATION À L'ALGORITHMIQUE ET AUX STRUCTURES DE DONNÉES**

(J. Courtin et I. Kowarski)

### 1. *Programmation structurée et structures de données élémentaires.*

Ce premier tome est consacré à la construction d'algorithmes fondamentaux sur les fichiers séquentiels et les vecteurs (structures à accès direct). Il s'adresse aux étudiants de premier cycle (IUT, BTS, DEUG) et plus généralement à tous les lecteurs désirant s'initier à la construction d'algorithmes corrects, qui sont la base de toute bonne programmation.

Les algorithmes sont écrits dans un langage proche du Pascal et leur traduction dans un langage classique (Pascal, Fortran, Cobol, ...) est aisée. De très nombreux exemples et exercices permettent au lecteur de se familiariser avec la démarche proposée et de vérifier que la méthode a bien été assimilée.

Le deuxième tome (parution fin 1987) aborde l'étude des structures des données avancées que sont les listes chaînées, les piles, les tables et les arbres. Les auteurs y mettent en évidence la notion de récursivité qui constitue une bonne initiation à l'utilisation des langages de l'intelligence artificielle (Lisp, Prolog).

Coll. "Dunod Informatique", 298 pages, broché, 120 F

## • **LA SÉCURITÉ DES RÉSEAUX** (J.-M. Lamère, Y. Leroux, J. Tourly) *Méthodes et techniques*

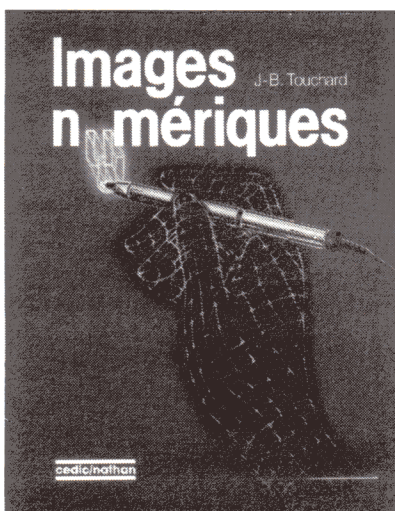
Faisant suite à un premier ouvrage de sensibilisation, *la Sécurité informatique*, le présent titre fournit aux praticiens tous les outils méthodologiques aujourd'hui disponibles pour l'analyse et la réduction des risques informatiques.

Dans cet esprit, les auteurs développent les méthodes MARION-AP (sites locaux et réseaux simples) et MARION-RSX (réseaux d'ordinateurs) avec l'ensemble des éléments quantitatifs : ratios, questionnaire d'audit, bases de référence, etc., permettant une mise en œuvre effective. Un exemple illustre la démarche pratique au travers de fiches méthodologiques. Un chapitre important est consacré aux moyens spécifiques de la sécurité des télécommunications : chiffrement, authentification, signature...

Cet ouvrage s'adresse à tous ceux qui ont conscience de l'émergence d'une nouvelle fonction liée à la sécurité informatique, aussi bien les utilisateurs et les professionnels que les étudiants de l'enseignement supérieur.

Collection "Dunod Informatique",  
374 pages, broché, 290 F

**DUNOD**, 17, rue Rémy-Dumoncel, B.P. 50,  
75661 PARIS CEDEX 14





# D'un Basic à un autre

**Q** UI ne connaît pas le GWBASIC, langage employé sur la plupart des micros considérés comme compatibles avec le modèle IBM ? Vous ? pas possible ! Je vous propose, à titre d'exercice — et simplement pour le plaisir — de transformer un programme écrit en GWBASIC... en routine APPLESOFT. Nous supposons, au départ, que vous disposez d'un Apple récent, équipé d'une carte 80 colonnes et autorisant l'utilisation des caractères souris, mais ce détail n'est important que pour le cadre... et encore ! Il est toujours possible d'afficher un cadre constitué par 204 astérisques.

## OBJET DU PROGRAMME

C'est un peu simplet. On vous demande seulement de taper votre nom et votre prénom. Toutes les lettres de l'alphabet sont acceptées, ainsi que l'espace, l'apostrophe et le tiret. Si une erreur est commise, le caractère erroné apparaît en mode inverse et il faut renouveler la saisie. Ensuite, si vous avez commis l'erreur de taper votre nom en oubliant les majuscules, on vous corrige gentiment... et c'est tout !

## TRADUCTION

*Seules sont expliquées les lignes posant un problème.*

**100** En Applesoft, on n'est pas forcément en 80 colonnes, d'où le libellé de cette ligne.

**CLS** est remplacé par **HOME** et **BEEP** par **PRINT CHR\$(7)** ou **CALL - 198**. La **REMARQUE** doit être précédée par **REM** (en GWBASIC, on peut utiliser la simple apostrophe).

**105** Applesoft, au contraire de GWBASIC, ne reconnaît pas les **à ê î ô û î ï** et **ë...** ce qui est dommage.

**110** **LOCATÉ 5,14** est remplacé par **VTAB 5 : HTAB 14**, **COLOR 0,7** par **INVERSE** et **COLOR 7,0** par **NORMAL**. Nous ne reviendrons plus là-dessus.

*Quelques textes ont été sensiblement modifiés, mais c'est sans rapport avec la programmation.*

**185** Le programme en GWBASIC comporte un **200** bug. Celui-ci est semble-t-il éliminé dans

la routine Applesoft. Comparez les deux versions (c'est du Basic, classique !) et essayez de comprendre !

**220** APPLESOFT ne connaît pas **INKEY\$**, mais la ligne d'attente habituelle fait parfaitement notre affaire.

**235** C'est évidemment le tracé du cadre qui à différencie le plus les deux machines...

**250** encore que ! nous disposons, sur le compatible IBM, de caractères semi-graphiques qui autorisent bien des acrobaties. Sur Apple, les caractères souris sont insuffisants et on ne le déplorera jamais assez. On peut néanmoins procéder comme nous l'avons fait ou tracer un cadre classique, soit en utilisant l'espace en mode inverse, soit un caractère comme l'astérisque... On peut aussi passer par une petite routine LM, mais cela nous éloigne du Basic traditionnel !

Pour éviter le scrolling, il est en tout cas indispensable de poquer le contenu de la case 2039 (HTAB 80, VTAB 24). Notez le **POKE 33,78** de la ligne 250, indispensable pour empêcher l'effacement du cadre, au moment de l'INPUT.

## Les caractères semi-graphiques

Voici les cinq caractères permettant le tracé (parfait) du cadre en GWBASIC :

201	⌘	205	=	187	⌘
200	⌘			188	⌘

## GW BASIC

```
100 CLS : BEEP : GOSUB 235 'Tracé du ca
dre
105 V$="abcdefghijklmnopqrstuvwxyzaéeëüâé
îôûîèç ABCDEFGHIJKLMNOPQRSTUVWXYZ-"
110 LOCATE 5,14: COLOR 0,7:PRINT" SAVEZ-
VOUS ORTHOGRAPHER VOTRE NOM ET VOTR
E PRENOM ? ":COLOR 7,0
115 LOCATE 10,5 : C = 0 : P= 0: BEEP
120 INPUT"Indiquez-moi votre nom et votr
e prénom ";N$
125 N1$="":'Nom corrigé (majuscules) éve
ntuellement
130 FOR I = 1 TO LEN (N$): M$= MID$ (N$,I
,1) : IF M$ = " " THEN C=I
135 IF M$="-" OR M$="'" THEN P=I
140 FOR J = 1 TO LEN (V$) : IF M$ = MID$
(V$,J,1) THEN 150
145 NEXT : LOCATE 10,45+I : COLOR 0,7 :
PRINT M$ : COLOR 7,0 : GOTO 115
150 IF (I-1 = C OR I-1=P) AND M$ > "Z" A
ND M$ < CHR$(123) THEN M$=CHR$(ASC(M
$)-32)
155 N1$=N1$+M$
160 NEXT I
165 LOCATE 12,5 : IF C THEN 175
170 PRINT "J'avais pourtant précisé: nom
ET prénom!": GOTO 115
175 PRINT "Ainsi, vous vous nommez "; LE
FT$(N$,C-1); " ... et votre prénom e
st";MID$(N$,C,LEN(N$)+1-C)
180 LOCATE 14,5
185 IF LEFT$(N$,1) > "Z" OR P < C THEN P
RINT "Une lettre capitale ne saurait
dévaluer votre NOM... la preuve": L
OCATE 15,15: PRINT "- "; LEFT$(N1$,C
-1) :GOTO 195
190 PRINT "Votre NOM semble correctement
orthographié..."
195 LOCATE 17,5
200 IF MID$( N$,C+1,1) >"Z" OR P >C THEN
PRINT"Votre prénom serait mieux orth
ographié ainsi":LOCATE 18,15: PRINT
"-";RIGHT$(N1$,LEN(N$)+1-C):GOTO 210
205 PRINT "Votre PRENOM ne paraît pas po
ser de problème..."
210 PRINT:PRINT:BEEP
215 GOSUB 220:CLS : END
220 N$=INKEY$ : IF LEN(N$)=0 THEN 220
225 RETURN
230 'tracé d'un cadre utilisant les cara
ctères graphiques
235 PRINT CHR$(201);: FOR I = 2 TO 79 :
PRINT CHR$ (205);: NEXT : PRINT CHR$
(187)
240 FOR I = 2 TO 23 : LOCATE I,1: PRINT
CHR$(186): LOCATE I,80: PRINT CHR$(1
86):NEXT
245 LOCATE 24,1: PRINT CHR$(200);: FOR I
= 2 TO 79 : PRINT CHR$ (205);: NEXT
: PRINT CHR$(188);
250 RETURN
```

## APPLESOFT

```
100 D$ = CHR$(4): PRINT D$"PRÉ3": PRINT
: HOME : PRINT CHR$(7): GOSUB 235: R
EM Tracé du cadre
105 V$ = "abcdefghijklmnopqrstuvwxyzaéeëüçA
BCDEFGHIJKLMNOPQRSTUVWXYZ-"
110 VTAB 5: HTAB 14: INVERSE : PRINT " SAV
EZ-VOUS ORTHOGRAPHER VOTRE NOM ET VOT
RE PRENOM ? ": NORMAL
115 VTAB 10: HTAB 5:C = 0:P = 0: CALL - 1
98
120 INPUT "Nom et votre prénom svp: ":N$
125 N1$ = "": REM Nom corrigé (majuscules)
éventuellement
130 FOR I = 1 TO LEN (N$):M$ = MID$(N$,
I,1): IF M$ = " " THEN C = I
135 IF M$ = "-" OR M$ = "'" THEN P = I
140 FOR J = 1 TO LEN (V$): IF M$ = MID$(
V$,J,1) THEN 150
145 NEXT : PRINT : HTAB 29 + I: VTAB 10: I
NVERSE : PRINT M$: NORMAL : GOTO 115
150 IF (I - 1 = C OR I - 1 = P) AND M$ > "
Z" AND M$ < CHR$(125) THEN M$ = CHR
$(ASC (M$) - 32)
155 N1$ = N1$ + M$
160 NEXT I
165 VTAB 12: HTAB 5: IF C THEN 175
170 PRINT "J'avais pourtant précisé: nom e
t prénom!": GOTO 115
175 PRINT "Nom: "; LEFT$(N$,C - 1);", pré
nom: "; MID$(N$,C, LEN (N$) + 1 - C)
180 VTAB 14: HTAB 5
185 R$ = LEFT$(N1$,C - 1): IF R$ < > L
EFT$(N$,C - 1) THEN PRINT "Une lettr
e capitale ne saurait dévaluer un NOM.
.. la preuve": VTAB 15: HTAB 15: PRIN
T "- ";R$: GOTO 195
190 PRINT "Votre NOM semble correctement o
rthographié..."
195 VTAB 17: HTAB 5
200 R$ = RIGHT$(N1$, LEN (N$) + 1 - C):
IF R$ < > RIGHT$(N$, LEN (N$) + 1 -
C) THEN PRINT "Votre prénom serait mi
eux orthographié ainsi": VTAB 18: HTA
B 15: PRINT "-";R$: GOTO 210
205 PRINT "Votre PRENOM ne paraît pas pose
r de problème..."
210 PRINT : PRINT CHR$(7)
215 GOSUB 220: TEXT : HOME : END
220 POKE 49168,0: WAIT 49152,128: POKE 491
68,0
225 RETURN
230 REM Tracé d'un cadre utilisant les car
actères souris
235 INVERSE : PRINT CHR$(27);: FOR I = 1
TO 80: PRINT "L";: NEXT
240 FOR I = 2 TO 23: VTAB I: POKE 1403,0:
PRINT "_";: POKE 1403,79: PRINT "Z";:
NEXT
245 POKE 1403,0: VTAB 24: FOR I = 1 TO 79:
PRINT "S";: NEXT
250 NORMAL : VTAB 1: PRINT CHR$(24): POK
E 2039,83: POKE 33,78: RETURN
```



# Votre bibliothèque INFORMATIQUE

par NESTOR

## • CLEFS POUR C (François Piette)

Au moment où je rédige ces lignes, les candidats à l'apprentissage du langage C sur Apple IIGS restent sur leur faim. Aucun ouvrage — à ma connaissance du moins — n'a traité la question, mais on connaît l'une des grandes qualités du C : son excellente portabilité.

Les amateurs de C (ils se révèlent de plus en plus nombreux) tireront donc profit de ce nouveau volet de la collection LANGAGES aux Editions du P.S.I.

En fait, CLEFS POUR C est un manuel de référence capable de fournir de bonnes informations à tout programmeur. Ses exemples (nombreux) ont été écrits en C standard. Ils sont en principe destinés aux utilisateurs de PC et compatibles, mais aussi à ceux de l'Atari ST et de l'Amstrad PC. Notons au passage que les programmeurs sur Atari ST ne sont pas aussi démunis que les applemaniaques armés d'un GS tout neuf.

Mais revenons aux CLEFS POUR C. L'auteur ne se contente pas, dans les 290 pages de son bouquin, d'une banale énumération d'instructions que d'autres ont déjà explicitées. Il étudie les deux versions les plus répandues du C : le

LATTICE C (c'est, dans ses grandes lignes, celui du GS) et le Digital Research C, principal outil de programmation sous GEM, sur Atari ST. A noter que les utilisateurs de Microsoft C peuvent aussi se reporter au chapitre consacré au Lattice C : ce sont des produits identiques jusqu'à la version 3.0.

EDITIONS DU PSI, BP 86 77402 LAGNY-S/MARNE — CEDEX (300 pages environ — 220 F TTC).

## • C ET SES FICHIERS (J. Boisgontier et J.-P. Lagrange)

Cet ouvrage s'adresse aux utilisateurs déjà familiarisés avec le langage C, mais les auteurs ont tout de même pris la peine de revenir sur les principales notions et fonctions du C. Toute la première partie du livre est consacrée à ce rappel. C'est concis, clair et précis. Machines concernées : PC et compatibles, Atari, Amiga, Macintosh, etc. La deuxième partie, plus intéressante à mes yeux parce que traitant un sujet moins connu, nous aide à découvrir la gestion de fichiers en C... ou à approfondir nos connaissances sur la question. Après l'avoir lue, vous saurez (presque) tout sur l'ouverture, la fermeture, l'écriture et la lecture des fichiers à accès direct et à accès séquentiel.

Les auteurs auraient pu en rester là, mais ils n'ont pas voulu nous quitter sans aborder des sujets aussi intéressants que : les méthodes de tri, l'allocation dynamique, les bases de données, etc.

EDITIONS DU PSI — (170 pages, 170 F)

## • ASTRONOMIE ET ORDINATEUR (Guy Sérane)

Si vous vous intéressez à l'astronomie, gageons que le livre de Guy Sérane ne vous laissera pas indifférent. Son objet est le calcul des positions astronomiques

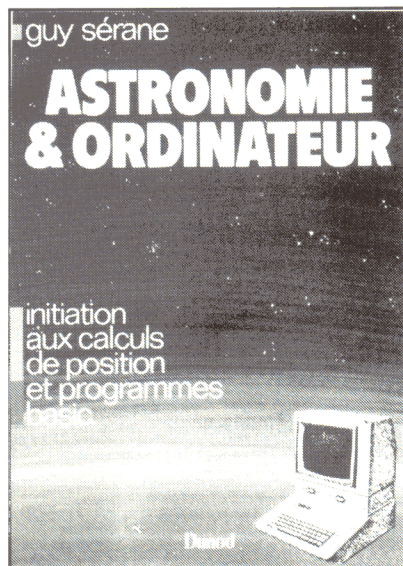
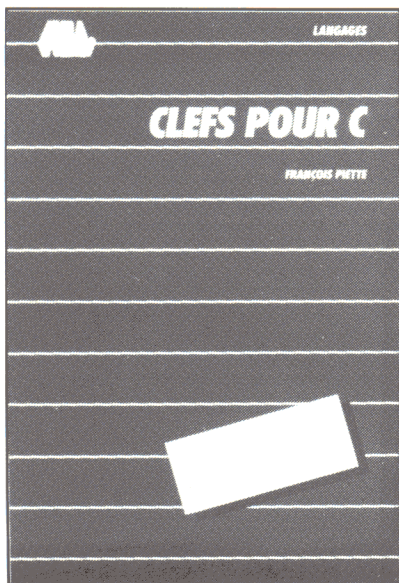
par des méthodes simples, accompagnées d'exemples numériques... une simplicité qui ne nuit en rien à la précision.

L'auteur traite des problèmes fondamentaux de la mécanique céleste :

- Positions des étoiles et des planètes à n'importe quelle date ;
- Positions et trajectoires des comètes et des satellites artificiels.
- Préviation et calculs de phénomènes astronomiques divers (éclipses, occultations, élongations...).
- Navigation : point astronomique, méridienne.
- Calculs de trajectoires de vaisseaux spatiaux.

L'originalité de l'ouvrage réside dans le fait que tous les sujets abordés au cours des différents chapitres font l'objet d'applications numériques et de programmes écrits en BASIC standard, mais peuvent être facilement adaptés à tout micro-ordinateur. C'est dire que tout lecteur ayant à sa disposition un micro-ordinateur peut utiliser ces programmes, sans avoir à connaître les démonstrations et développements, d'aspect parfois complexe, fournis dans le corps du livre.

DUNOD, 17, rue Remy-Dumoncel, B.P. 50, 75661 PARIS CEDEX 14 (264 pages — 160 F)





# STARTUP

Ce programme présente un double intérêt : **utilitaire** et **didactique**.

## 1. Aspect utilitaire :

Pour la réalisation de ce STARTUP, je suis parti avec les objectifs suivants :

- Occupation d'un seul bloc sur la disquette.
- Rapidité d'exécution (rien n'est plus pénible que ces "HELLO" qui n'en finissent plus de charger le catalogue).
- Lecture des catalogues secondaires (type DIR).
- Possibilité de lancer un programme de type BAS, BIN, REL, SYS, TXT (en tant que fichier EXEC).

Le plus difficile a été de tout faire entrer dans les 512 octets impartis (ne vous attendez donc pas à trouver des fioritures !), pour le reste il n'y a pas eu de gros problèmes. Le programme sélectionné est lancé avec le dash "-", pour un fichier type TXT qui n'est pas un fichier EXEC, vous aurez droit à une avalanche de SYNTAX ERROR.

## 2. Aspect didactique :

L'analyse du programme vous montrera les deux types d'appel de ProDOS à partir de l'assembleur. Vous serez amené également à comprendre l'organisation des catalogues sur les disquettes ProDOS. Vous pouvez aussi utiliser le programme sans vous poser de question sur son fonctionnement...

### Les commandes

- Les flèches droite et gauche permettent de sélectionner le fichier.
- La touche RETURN essaie de le lancer, dans le cas d'un fichier DIR le PREFIX est mis à jour (ne pas dépasser 40 caractères pour éviter les surprises !) et le nouveau catalogue est affiché.
- La touche ESCAPE permet de relire le catalogue principal et la touche ESC vous envoie directement dans le BASIC.

```

1 * ***** *
2 * * STARTUP par Maurice Chavelli * *
3 * *      pour Tremplin-Micro      * *
4 * ***** *
5
6 PTR      =    $6          ; Pointeurs
7 ADD      =    $8
8 SCH      =   $18          ; Sauvegarde CH
9 COMPT    =   $1B          ; Compteur de fichiers
10 WNDTOP   =   $22          ; Haut de la fenetre TEXTE
11 CH       =   $24          ; Pointeur horizontal du curseur
12 CV       =   $25          ; Pointeur vertical du curseur
13 BASL     =   $28          ; Pointeur adresse ligne courante
14 BASH     =   $29
15 IN       =  $200          ; Pointeur du buffer d'entrée
16 DOSCMD   =  $BE03        ; Entrée page globale BASIC-SYS

```

(suite page 32)

```

17 MLI      =    $BF00      ; Entrée page globale de ProDOS
18 SETTXT  =    $FB39      ; équivalent du TEXT du BASIC
19 TABV    =    $FB5B      ; Positionne curseur ligne A
20 BASCALC =    $FBC1      ; Calcule BASL et BASH ligne A
21 VTAB    =    $FC22      ; Positionne curseur ligne CV
22 HOME    =    $FC58      ; Equivalent du HOME du BASIC
23 CROUT   =    $FD8E      ; Envoie un retour chariot
24 COUT    =    $FDED      ; Envoie caractère code dans A
25 BELL    =    $FF3A      ; Un petit Bip!
26
27          ORG    $7000
28
29 DEBUT    LDA    $C0BB      ; Affichage vert CHAT MAUVE
30          LDA    £0        ; Initialisations
31          STA    ADD
32          STA    NBR+1
33          STA    SCH
34          LDA    £2        ; Le catalogue principal
35          STA    NBR        ; commence au Block 2
36
37 * Chargement du catalogue en $7600. Le catalogue principal
38 * occupe les 4 blocs de 2 à 5. Les catalogues secondaires
39 * occupent 1 ou plusieurs blocs qui se situent n'importe
40 * où, mais qui sont chaînés. On trouve d'abord, sur 2 oc-
41 * tets, le pointeur arrière, et sur les 2 octets suivants
42 * le pointeur avant.
43
44 CHARGE   LDA    £$76      ; Buffer en $7600
45          STA    BUF+1
46 CHARG    JSR    MLI        ; Appel standard du MLI
47          HEX    80        ; Code de la commande READ BLOCK
48          DA    PARAM      ; Pointeur des paramètres
49          BEQ    CH1
50          JMP    BELL       ; Impossible à lire!
51 CH1      LDA    BUF+1      ; Lecture réussie
52          STA    ADD+1
53          INC    BUF+1      ; 1 bloc occupe 2 pages mémoire
54          INC    BUF+1
55          LDY    £2
56          LDA    (ADD),Y    ; Récupère le pointeur avant
57          STA    NBR        ; et le sauve en NBR et NBR+1
58          INY
59          LDA    (ADD),Y
60          STA    NBR+1
61          BNE    CHARG      ; S'il n'est pas nul on continue
62          LDA    NBR
63          BNE    CHARG
64
65 * Préparation commande PREFIX. C'est une deuxième manière
66 * d'appeler ProDOS à partir de l'Assembleur: elle exige
67 * la présence du BASIC.SYSTEM.
68
69          LDY    £5        ; Inscrit PREFIX dans buffer IN

```

```

70  PREF      LDA  PREFIX,Y
71           STA  IN,Y
72           DEY
73           BPL  PREF
74
75  * Affichage et commande PREFIX
76
77           JSR  HOME      ; Efface la fenetre d'écran
78           JSR  SETTXT   ; TEXT
79           LDA  £0       ; Positionne curseur sur ligne 0
80           JSR  TABV
81           LDA  £4       ; Pointe PTR sur le nom du Volume
82           STA  PTR
83           LDA  £#76
84           STA  PTR+1
85           LDA  SCH      ; Ajuste CH pour écriture
86           STA  CH       ; correcte du PREFIX
87           LDA  £"/     ; Commence par afficher un /
88           JSR  COUT
89           JSR  COM      ; Affiche PREFIX et -> IN
90           JSR  DOSCMD   ; Exécute commande dans buffer IN
91           LDA  £2       ; Diminue la fenetre
92           STA  WNDTOP
93           LDA  £0       ; Initialise compteur de fichiers
94           STA  COMPT
95
96  * Affichage des noms de fichiers
97
98  TRAIT     JSR  HOME
99           LDX  £22      ; C'est parti pour les 22 lignes
100  T1       JSR  PLUS     ; Avance d'un fichier
101           JSR  ECRIT   ; Affiche le nom du fichier
102           LDA  CV      ; Bas de la page?
103           CMP  £23
104           BEQ  T7      ; Oui, on arrete
105           JSR  CROUT   ; Non, passe à la ligne suivante
106  T7       LDA  £7625   ; Récupère nombre total fichiers
107           CMP  COMPT   ; Le compare avec COMPT
108           BEQ  T2      ; S'il est égal on arrete
109           DEX         ; Sinon on décrémente X
110          BNE  T1       ; S'il n'est pas nul on remet ça
111  T2       LDA  £1      ; Remet COMPT à 1
112           STA  COMPT
113           LDA  £#2B    ; Remet PTR sur £7600
114           STA  PTR
115           LDA  £#76
116           STA  PTR+1
117           LDA  £2      ; Remet le curseur sur la ligne 2
118  T3       JSR  TABV
119
120  * Traitement du clavier
121
122  T4       LDY  £39     ; Affiche inverse fichier courant

```

(suite page 34)



```

123 T5      LDA  (BASL),Y
124        AND  £$3F      ; Masque pour l'inversion
125        STA  (BASL),Y
126        DEY
127        BPL  T5
128 T6      LDA  $C000    ; Attend une frappe de touche
129        BPL  T6
130        STA  $C010
131        CMP  £$A0      ; Est-ce la barre d'espace?
132        BNE  T8        ; Non
133        JMP  DEBUT     ; Oui, on recommence tout à zéro
134 T8      CMP  £$9B      ; Est-ce la touche ESC?
135        BNE  T11     ; Non
136        JSR  SETTXT   ; Oui, fenetre normale
137        JMP  HOME     ; on efface et on abandonne tout!
138 T11     CMP  £$95      ; Est-ce la flèche à droite?
139        BEQ  BAS      ; Oui
140        CMP  £$88      ; Est-ce la flèche à gauche?
141        BEQ  HAUT     ; Oui
142        CMP  £$8D      ; Est-ce le touche RETURN?
143        BNE  T5        ; Non, alors c'est une erreur!
144        INY
145        LDA  (PTR),Y   ; Récupère le type de fichier
146        AND  £$F0
147        CMP  £$D0      ; Type DIR?
148        BEQ  SOUS     ; Oui, lire nouveau catalogue
149        JMP  HOOK     ; Vecteur extension éventuelle
150 T10     JSR  SETTXT   ; TEXT
151        JSR  HOME     ; HOME
152        LDA  £$A0      ; Code espace dans A
153        LDY  £4        ; 4 espaces dans le buffer
154 T9      STA  IN,Y     ; c'est pour utiliser le
155        DEY          ; meme sous-programme COM!
156        BPL  T9
157        LDA  £"-      ; Inscrit le dash dans buffer
158        STA  IN+5
159        JSR  COM      ; Affiche nom fichier -> IN
160        JMP  DOSCMD   ; C'est parti pour ce fichier!
161
162 * Lecture sous-catalogue
163
164 SOUS     LDY  £17      ; Récupère numéro Key-Block
165        LDA  (PTR),Y   ; du sous-catalogue et le
166        STA  NBR      ; sauve dans PTR
167        INY
168        LDA  (PTR),Y
169        STA  NBR+1
170        JMP  CHARGE   ; C'est parti nouveau catalogue!
171
172 * Traitement vers le bas
173
174 BAS      JSR  NORM    ; Affichage normal
175        LDA  COMPT    ; Récupère numéro fichier

```

```

176          CMP    £7625      ; Le compare total fichiers
177          BEQ    T4          ; Si égalité ne va pas plus loin
178          JSR    PLUS        ; Incrémente PTR et COMPT
179          INC    CV          ; Incrémente CV
180          LDA    CV
181          CMP    £24         ; Sort de la page?
182          BNE    T3          ; Non, alors tout va bien
183          DEC    CV          ; Oui, alors on décrémente CV
184          JSR    CROUT       ; On monte les fichiers d'un cran
185          JSR    ECRIT       ; On écrit nom du petit nouveau
186          JMP    T4
187
188 * Traitement vers le haut
189
190 HAUT      JSR    NORM        ; Affichage normal
191          DEC    COMPT        ; Un fichier plus "bas"
192          BNE    H1          ; Si on est pas à 0 on continue
193          INC    COMPT        ; Sinon il n'y a plus rien à voir
194          BEQ    H1
195          JMP    T4
196 H1        JSR    MOINS       ; Décrémente PTR et COMPT
197          DEC    CV          ; Décrémente CV
198          LDA    CV
199          CMP    £1          ; Ligne 1?
200          BEQ    H2          ; Oui, alors il faut réagir
201          JMP    T3          ; Non, alors c'est parfait
202 H2        LDA    £24         ; Ici petit SCROLL inverse
203          STA    CV          ; Je vous laisse le découvrir
204 H3        DEC    CV
205          DEC    CV
206          BEQ    H5          ; La sortie est ici!
207          JSR    VTAB
208          LDA    BASL
209          STA    ADD
210          LDA    BASH
211          STA    ADD+1
212          INC    CV
213          JSR    VTAB
214          LDY    £39
215 H4        LDA    (ADD),Y
216          STA    (BASL),Y
217          DEY
218          BPL    H4
219          BMI    H3
220 H5        INY
221          STY    CH          ; Curseur sur première colonne
222          LDA    £2          ; Curseur sur la ligne 2
223          JSR    TABV
224          JSR    ECRIT       ; Écrit le nom du fichier
225          JMP    T4
226
227 * Paramètres pour chargement de Bloc
228

```

(suite page 36)



```

229 PARAM    HEX  03          ; 3 paramètres
230          HEX  60          ; Slot 6 et Drive 1
231 BUF      DA  $7600       ; Buffer de réception
232 NBR      DA  0           ; Numéro du Bloc
233
234 * Routine avance d'un titre
235
236 PLUS      INC  COMPT      ; Incrémente COMPT
237 P0        CLC             ; Effectue PTR+39, chaque fichier
238          LDA  PTR         ; occupe 39 octets dans le
239          ADC  £39         ; Volume Directory
240          STA  PTR
241          BCC  P1
242          INC  PTR+1
243 P1        CMP  £$FF       ; Attention changement de Bloc!
244          BNE  P2         ; Pas de changement
245          LDA  £4          ; Changement de Bloc ajuste PTR
246          STA  PTR
247          INC  PTR+1
248 P2        LDY  £0         ; Fichier effacé?
249          LDA  (PTR),Y
250          BEQ  P0         ; Oui, alors on passe au suivant
251          RTS            ; Non, alors tout va bien...
252
253 * Routine recule d'un titre
254
255 MOINS      SEC             ; Effectue PTR-39
256          LDA  PTR
257          SBC  £39
258          STA  PTR
259          BCS  M1
260          DEC  PTR+1
261 M1        CMP  £$DD       ; Changement de Bloc?
262          BNE  M2         ; Non
263          LDA  £$D8       ; Oui, alors on ajuste PTR
264          STA  PTR
265 M2        LDY  £0         ; Fichier effacé?
266          LDA  (PTR),Y
267          BEQ  MOINS      ; Oui, alors on passe au suivant
268          RTS            ; Non, alors tout va bien...
269
270 * Routine d'écriture nom de fichier
271
272 ECRIT      LDY  £0         ; Récupère longueur nom fichier
273          LDA  (PTR),Y
274          AND  £$F
275          STA  EC2+1       ; et la sauve dans EC2+1
276 EC1       INY             ; Affiche le nom à l'écran
277          LDA  (PTR),Y
278          ORA  £$80        ; Bit 7 à 1 pour affichage
279          JSR  COUT
280 EC2       CPY  £0
281          BNE  EC1

```

```

282         LDY  CH           ; Récupère CH pour PREFIX
283         RTS
284
285 * Routine affichage normal
286
287 NORM     LDY  £39
288 N1       LDA  (BASL),Y
289         ORA  £$80         ; Masque pour affichage normal
290         STA  (BASL),Y
291         DEY
292         BPL  N1
293         RTS
294
295 * Routine pour commande
296
297 COM      JSR  ECRIT        ; Ecrit le nom
298         STY  SCH          ; Sauvgarde CH pour le PREFIX
299         LDY  £0           ; Inscrit le nom dans buffer IN
300 C1       LDA  (BASL),Y
301         STA  IN+6,Y
302         INY
303         CPY  SCH          ; Jusqu'à SCH contient longueur
304         BNE  C1
305         LDA  £$8D         ; Inscrit un retour chariot
306         STA  IN+6,Y
307         RTS
308
309 * Textes
310
311 PREFIX   ASC  "PREFIX"
312
313 * Vecteur pour modification éventuelle, par exemple trai-
314 * tement complet pour les fichiers: RENAME, DELETE... mais
315 * le programme débordera du bloc.
316
317 HOOK     JMP  T10

```

*Si vous ne possédez pas  
d'Assembleur, passez à la page  
suivante pour saisir les codes du  
programme STARTUP de  
Maurice CHAVELLI.*

## SIGNATURE

Cette disquette *TREMLIN MICRO* permet une vérification facile des programmes (Basic et LM) insérés dans la revue. Bon nombre de nos lecteurs l'utilisent déjà et s'en félicitent.

C'est à leur intention que nous publions le petit rectificatif ci-après, modification qui autorise la vérification des fichiers binaires sur 24 octets par ligne (au lieu de 16).

Concerne **BIN.CHECK.PRO** et **BIN.CHECK.DOS**

```

515 NK = PEEK (ENTREE + 21): VTAB 12: PRINT "Nombre de colonnes >"NK;: HTAB 26: INPUT "";X$: IF X$ > ""
    THEN X = VAL (X$): ON X < > 16 AND X < > 24 AND X < > 32 GOTO 515: POKE ENTREE + 21,X
517 IF SLOT = 3 OR NOT SLOT THEN RETURN
2022 HOME

```



## STARTUP (LES CODES)

/TMF/STAR/C,A\$7000,L\$01FC\*

BSAVE NOM,A\$7000,L\$01FC

7000:	AD BB C0 A9 00 85 08 8D 7F 71 85 18 A9 02 8D 7E 71 A9 76 8D 7D 71 20 00	4B59
7018:	BF 80 7A 71 F0 03 4C 3A FF AD 7D 71 85 09 EE 7D 71 EE 7D 71 A0 02 B1 08	AFDE
7030:	8D 7E 71 C8 B1 08 8D 7F 71 D0 DB AD 7E 71 D0 D6 A0 05 B9 F3 71 99 00 02	FBC4
7048:	88 10 F7 20 58 FC 20 39 FB A9 00 20 5B FB A9 04 85 06 A9 76 85 07 A5 18	E816
7060:	85 24 A9 AF 20 ED FD 20 DC 71 20 03 BE A9 02 85 22 A9 00 85 1B 20 58 FC	2C68
7078:	A2 16 20 80 71 20 B8 71 A5 25 C9 17 F0 03 20 8E FD AD 25 76 C5 1B F0 03	1075
7090:	CA D0 E7 A9 01 85 1B A9 2B 85 06 A9 76 85 07 A9 02 20 5B FB A0 27 B1 28	9D96
70A8:	29 3F 91 28 88 10 F7 AD 00 C0 10 FB 8D 10 C0 C9 A0 D0 03 4C 00 70 C9 9B	3AE1
70C0:	D0 06 20 39 FB 4C 58 FC C9 95 F0 3F C9 88 F0 5B C9 8D D0 D2 C8 B1 06 29	C593
70D8:	F0 C9 D0 F0 1E 4C F9 71 20 39 FB 20 58 FC A9 A0 A0 04 99 00 02 88 10 FA	982F
70F0:	A9 AD 8D 05 02 20 DC 71 4C 03 BE A0 11 B1 06 8D 7E 71 C8 B1 06 8D 7F 71	BA44
7108:	4C 11 70 20 D0 71 A5 1B CD 25 76 F0 8F 20 80 71 E6 25 A5 25 C9 18 D0 81	73ED
7120:	C6 25 20 8E FD 20 B8 71 4C A4 70 20 D0 71 C6 1B D0 07 E6 1B F0 03 4C A4	463C
7138:	70 20 9E 71 C6 25 A5 25 C9 01 F0 03 4C A1 70 A9 18 85 25 C6 25 C6 25 F0	D89F
7150:	1B 20 22 FC A5 28 85 08 A5 29 85 09 E6 25 20 22 FC A0 27 B1 08 91 28 88	9C19
7168:	10 F9 30 DF C8 84 24 A9 02 20 5B FB 20 B8 71 4C A4 70 03 60 00 76 00 00	632B
7180:	E6 1B 18 A5 06 69 27 85 06 90 02 E6 07 C9 FF D0 06 A9 04 85 06 E6 07 A0	6DC1
7198:	00 B1 06 F0 E5 60 38 A5 06 E9 27 85 06 B0 02 C6 07 C9 DD D0 04 A9 D8 85	5369
71B0:	06 A0 00 B1 06 F0 E7 60 A0 00 B1 06 29 0F 8D CA 71 C8 B1 06 09 80 20 ED	8700
71C8:	FD C0 00 D0 F4 A4 24 60 A0 27 B1 28 09 80 91 28 88 10 F7 60 20 B8 71 84	C547
71E0:	18 A0 00 B1 28 99 06 02 C8 C4 18 D0 F6 A9 8D 99 06 02 60 D0 D2 C5 C6 C9	F9C9
71F8:	D8 4C E0 70	B874

\* Cette référence concerne la disquette TREMLIN MICRO, mais vous pouvez donner à cette routine le titre de votre choix : STARTUP.C par exemple.

## CONSEILS AUX DÉBUTANTS :

Rappelons que, pour taper un programme en langage machine (sans assembleur), il suffit d'accéder au Moniteur par un **CALL - 151**, puis de taper textuellement chacune des lignes ci-dessus. Lorsque cette tâche — ô combien épuisante — est terminée, sortir du Moniteur par CTRL-C (touche CTRL + C), puis taper le **BSAVE NOM,A\$7000,L\$01FC** indiqué en tête de programme.

### PRÉCAUTION :

Avant de rentrer les codes d'un programme, s'assurer que l'on a bien accès au DOS ou à ProDOS en essayant d'obtenir le catalogue de la disquette. Eventuellement (cela n'engage à rien !) essayer de sauver la routine avant même de l'avoir tapée.

Rien n'est plus désagréable que de se retrouver devant un écran rempli de codes, après vingt minutes de frappe laborieuse... sans le DOS !

**Attention !** Ne tapez surtout pas les valeurs indiquées en couleur, à la fin de chaque ligne : elles sont destinées aux utilisateurs de la disquette SIGNATURE et permettent de localiser très rapidement les éventuelles erreurs de saisie (lire, à ce sujet le bas de la page précédente).

# GS RAM et GS RAM PLUS

## extensions mémoire pour Apple IIGS

*Connaissez-vous les produits d'Applied Engineering ? et notamment GS RAM et GS RAM PLUS, extensions mémoire pour Apple IIGS ? Vous pensez sans doute que ces cartes sont encore réservées au marché américain ? Erreur : l'importateur existe et je l'ai rencontré. C'est la société lyonnaise BREJOUX. J'ajoute que le prix de vente de ces cartes est raisonnable (si mes renseignements sont bons, il correspond à un cours normal du dollar, ce qui est assez rare pour mériter d'être signalé). Qui plus est, elles sont garanties 5 ans. Parfaitement : vous avez bien lu.*

### GS RAM PLUS OU GS RAM ?

La première supporte des DRAM Chips de 1MO, permettant d'obtenir 6 MEGAS sur la carte principale. La seconde supporte des RAM Chips de 256 K. Elle est disponible en différentes versions (de 512 K à 1,5 MEGA), mais une carte additionnelle "piggy back" peut, en venant s'enficher sur elle, étendre sa mémoire à 8 MEGAS... reconnu par l'Apple IIGS. Naturellement, GS RAM et GS RAM PLUS sont compatibles avec tous les logiciels du GS, AppleWorks inclus, tout comme le Basic, ProDOS, DOS 3.3, PASCAL, C et CPM.

### POUR QUOI FAIRE ?

Par exemple pour passer de 60000 à 250000 lignes en traitement de texte. GS RAM repousse les limites d'AppleWorks et vous offre, par-dessus le marché, un buffer d'impression qui vous autorise à continuer la saisie pendant l'impression. Capacité mémoire oblige : GS RAM auto-segmente vos fichiers importants pour les sauver sur deux disquettes ou plus. Pratique, non ?

### UN PATCH

Craignant un retard dans la diffusion de la version AppleWorks 2.0 en Europe (ou sa non commercialisation), la Société Bréjoux avait lancé l'étude d'un patch sur la version française 1.4. La première monture de ce programme supporte la version 2.0 américaine et la version française 1.4 ProDOS 8 et 16. Elle a la particularité de reconnaître l'ordinateur lors de la configuration, ainsi que les diverses cartes d'extension : RamWorks pour le IIe, RamFactor pour le II Plus, IIe et IIGS... GS RAM et GS RAM PLUS pour l'Apple IIGS.

### INSTALLATION

Une notice on ne peut plus claire (en français) indique exactement la marche à suivre. A moins de s'obstiner à la lire la tête en bas et les yeux fermés, il est impossible de se tromper. Quand la carte est en place et le GS en marche, il suffit de consulter le tableau de bord pour savoir que l'on dispose d'une jolie réserve de mémoire !

### APPLEWORKS 2 EXPANDER

Là encore, l'utilisateur dispose d'une documentation en français... que je lui conseille de lire (ce que je n'ai pas fait... d'où un patch intempestif de ma version d'AppleWorks... mais j'en possédais heureusement une copie). Il convient de suivre exactement les instructions, mais c'est enfantin. *Précision utile* : le support technique de la Société Bréjoux n'est pas avare de précisions éventuelles.

Documentation et renseignements : Société Bréjoux — J.-M. BRESARD  
29, rue de Montriblond 69009 LYON — Tél. 78.36.52.69.

# HORLOGE.GS

Cette routine en langage machine vous permettra de travailler avec la date et l'heure dans vos programmes en Basic. Pour qu'elle fonctionne correctement il est absolument nécessaire d'avoir PRODOS 8 en mémoire.

## MODE D'EMPLOI :

Votre programme Basic devra contenir les lignes suivantes :

```
5 PRINT CHR$(4);"-HORLOGE.GS"
10 DA$="00/00/00": Rem Initialisation de la
   variable qui contiendra la date
20 HE$="00:00": REM Initialisation de la varia-
   ble qui contiendra l'heure
```

30 Rem Ces deux réservations sont indis-  
pensables

40 & DA\$,HE\$

50 PRINT "Nous sommes le ";DA\$

60 PRINT "HEURE = ";HE\$

Vous pourrez répéter la ligne 40 autant de fois que vous le désirerez dans votre programme.

## COMMENT ÇA MARCHE ?

Le programme Assembleur retient les deux adresses des variables DA\$ et HE\$. Ensuite, appel est fait à la routine GET TIME (\$82) de PRODOS qui place la date et l'heure en mémoire de la manière suivante :

### DATE

49041 (\$BF91)

49040 (\$BF90)

7	6	5	4	3	2	1	0
Année							

7	6	5	4	3	2	1	0
Mois				Jour			

### TIME

49043 (\$BF93)

49042 (\$BF92)

7	6	5	4	3	2	1	0
Heure							

7	6	5	4	3	2	1	0
Minute							

Le reste du programme n'est qu'un décodage de ces données et une conversion au format texte de manière à entrer dans DA\$ et HE\$.

**Remarque :** Il est primordial que DA\$ ait au moins une longueur de 8 caractères et HE\$ une longueur de 5.



```

1 *****
2 *
3 *           HORLOGE.GS (PRODOS8 UNIQUEMENT) *
4 *
5 *****
6 *
7 *           Routine utilitaire permettant l'utilisation *
8 *           de l'horloge interne du 2 GS. *
9 *           Cette routine est appellable par Basic : *
10 *           10 DA$="00/00/00":HE$="00:00" *
11 *           20 & DA$,HE$ *
12 *           (Voir exemple BASIC) *
13 *-----*
14 *   Assembleur MERLIN.PRO           Roger WAGNER Editeur *
15 *-----*
16 *
17 *   Marc BROCHA                       Le 21/03/87 *
18 *
19 *****
20
21          ORG   $300
22
23 PTRGET  =   $DFE3   ;Routine Applesoft de recherche de descripteur
24 SNGFLT  =   $E301   ;Convertit l'entier en Y en flottant dans FAC
25 FOUT    =   $ED34   ;Convertit FAC en une chaine de caracteres
26 FBUFFER =   $100    ;Buffer utilise par FOUT
27 CHKCOM  =   $DEBE   ;Recherche d'une virgule
28 STRDATE =   $18     ;Pointeur page 0 de DATE (DA$)
29 STRTIME =   $1A     ;Pointeur page 0 de TIME (HE$)
30 STR     =   $1C     ;Pointeur intermediaire
31 AMPER   =   $3F5
32 MLI     =   $BF00   ;Entree PRODOS
33 DATE    =   $BF90   ;Adresse de sauvegarde de Date par Prodos
34 TIME    =   $BF92   ;Adresse de sauvegarde de l'Heure par Prodos
35
36 *   Initialisation d'Ampersand
37
0300: A9 4C 38
0302: 8D F5 03 39
0305: A9 10 40
0307: 8D F6 03 41
030A: A9 03 42
030C: 8D F7 03 43
030F: 60 44
45
46 *   Corps du programme
47
0310: 20 E3 DF 48 DEBUT JSR PTRGET ;Saisie de l'adresse Y,A de DA$
0313: 85 18 49 STA STRDATE ;Sauvegarde de Y,A
0315: 84 19 50 STY STRDATE+1

```

(Suite page 42)

```

0317: A0 01    51      LDY  £1          ;Sauvegarde de l'adresse réelle de DA$
0319: B1 18    52      LDA  (STRDATE),Y; dans STRDATE
031B: 48       53      PHA
031C: C8       54      INY
031D: B1 18    55      LDA  (STRDATE),Y
031F: 85 19    56      STA  STRDATE+1
0321: 68       57      PLA
0322: 85 18    58      STA  STRDATE
                    59
0324: 20 BE DE 60      JSR  CHKCOM      ;Recherche de la vigule
0327: 20 E3 DF 61      JSR  PTRGET      ;Saisie de l'adresse de HE$ dans Y,A
032A: 85 1A    62      STA  STRTIME     ;Sauvegarde de Y,A
032C: 84 1B    63      STY  STRTIME+1
032E: A0 01    64      LDY  £1          ;Sauvegarde de l'adresse réelle de HE$
0330: B1 1A    65      LDA  (STRTIME),Y; dans STRTIME
0332: 48       66      PHA
0333: C8       67      INY
0334: B1 1A    68      LDA  (STRTIME),Y
0336: 85 1B    69      STA  STRTIME+1
0338: 68       70      PLA
0339: 85 1A    71      STA  STRTIME
                    72
033B: 20 00 BF 73      JSR  MLI         ;Appel de la routine GET_TIME
033E: 82 00 00 74      HEX  82,00,00   ;code $82 sans paramètres
                    75
0341: AD 90 BF 76      LDA  DATE        ;On isole le numéro du jour
0344: 29 1F    77      AND  £$1F       ; dans JOUR
0346: 8D BB 03 78      STA  JOUR
0349: AD 91 BF 79      LDA  DATE+1     ;On isole le numéro de l'année
034C: 4A       80      LSR             ; dans ANNEE
034D: 8D BD 03 81      STA  ANNEE
0350: AD 90 BF 82      LDA  DATE        ;On isole le numéro du mois
0353: 6A       83      ROR             ; dans MOIS
0354: 4A       84      LSR
0355: 4A       85      LSR
0356: 4A       86      LSR
0357: 4A       87      LSR
0358: 8D BC 03 88      STA  MOIS
035B: A2 03    89      LDX  £3         ;Initialisation de LOOPX
035D: 8E BF 03 90      STX  LOOPX      ;3 données Jour,Mois,Année
0360: A6 18    91      LDX  STRDATE    ;On travaille avec DA$
0362: A4 19    92      LDY  STRDATE+1
0364: 20 80 03 93      JSR  LOOP0      ;On convertit les 3 bytes dans DA$
                    94
0367: AD 93 BF 95      LDA  TIME+1     ;On isole l'heure
036A: 8D BB 03 96      STA  JOUR       ; dans JOUR
036D: AD 92 BF 97      LDA  TIME      ;et les minutes
0370: 8D BC 03 98      STA  MOIS      ; dans MOIS
0373: A2 02    99      LDX  £2         ;Initialisation de LOOPX
0375: 8E BF 03 100     STX  LOOPX     ;2 données Heure,Minute
0378: A6 1A    101     LDX  STRTIME    ;On travaille avec HE$

```

```

037A: A4 1B 102 LDY STRTIME+1
037C: 20 80 03 103 JSR LOOP0 ;On convertit les 2 bytes dans HE$
037F: 60 104 RTS
105
106 * Sous programme de Conversion
107
0380: 86 1C 108 LOOP0 STX STR ;On sauvegarde l'adresse de travail
0382: 84 1D 109 STY STR+1 ;dans STR
0384: A2 00 110 LDX £0
0386: BE BE 03 111 STX PTRX ;Initialisation de PTRX, compteur de boucle
0389: BD BB 03 112 LOOP LDA JOUR,X ;On prend la donnée pointée par JOUR,X
038C: AB 113 TAY ;On la transfère dans Y
038D: 20 01 E3 114 JSR SNGFLT ;On convertit Y en flottant dans FAC
0390: 20 34 ED 115 JSR FOUT ;On décharge FAC en format texte dans FBUFFER
0393: AD BE 03 116 LDA PTRX ;Calcul de l'index de sauvegarde dans la
0396: 0A 117 ASL ;chaîne de caractères
0397: 6D BE 03 118 ADC PTRX ;Y <-- PTRX * 3
039A: AB 119 TAY
039B: A9 30 120 LDA ££30 ;Initialisation à 00 de la zone de 2 digits
039D: 91 1C 121 STA (STR),Y
039F: C8 122 INY
03A0: 91 1C 123 STA (STR),Y
03A2: AD 01 01 124 LDA FBUFFER+1 ;Saisie du digit de poids faible
03A5: F0 03 125 BEQ LOOP1 ;Si 0 digit de poids fort reste à 0
03A7: 91 1C 126 STA (STR),Y ;sinon sauvegarde du digit de poids faible
03A9: 88 127 DEY
03AA: AD 00 01 128 LOOP1 LDA FBUFFER ;Saisie du digit du poids fort
03AD: 91 1C 129 STA (STR),Y ;et sauvegarde
03AF: EE BE 03 130 INC PTRX ;PTRX = PTRX + 1
03B2: AE BE 03 131 LDX PTRX
03B5: EC BF 03 132 CPX LOOPX ;Si PTRX <> LOOPX on continue
03B8: D0 CF 133 BNE LOOP
134
03BA: 60 135 RTS ;Sinon Retour
136
137 * Réserve des variables
138
03BB: 00 139 JOUR HEX 00
03BC: 00 140 MOIS HEX 00
03BD: 00 141 ANNEE HEX 00
03BE: 00 142 PTRX HEX 00
03BF: 00 143 LOOPX HEX 00

```

SI VOUS POSSEDEZ "SIGNATURE"  
contentez-vous de taper les  
codes ci-après. Sauvegarde:  
BSAVE HORLOGE.GS,A\$300,L\$C0

```

0380: A9 4C 8D F5 03 A9 10 8D F6 03 A9 03 8D F7 03 60 20 E3 DF 85 18 84 19 A0 9D08
0318: 01 B1 18 48 C8 B1 18 85 19 68 85 18 20 BE DE 20 E3 DF 85 1A 84 1B A0 01 A1C3
0330: B1 1A 48 C8 B1 1A 85 1B 68 85 1A 20 00 BF 82 00 00 AD 90 BF 29 1F 8D BB 093A
0348: 03 AD 91 BF 4A 8D BD 03 AD 90 BF 6A 4A 4A 4A 8D BC 03 A2 03 8E BF 03 C666
0360: A6 18 A4 19 20 80 03 AD 93 BF 8D BB 03 AD 92 BF 8D BC 03 A2 02 8E BF 03 E4A6
0378: A6 1A A4 18 20 80 03 60 86 1C 84 1D A2 00 8E BE 03 BD BB 03 A8 20 01 E3 2ADD
0390: 20 34 ED AD BE 03 0A 6D BE 03 A8 A9 30 91 1C C8 91 1C AD 01 01 F0 03 91 AFBD
03A8: 1C 88 AD 00 01 91 1C EE BE 03 AE BE 03 EC BF 03 D0 CF 60 00 00 00 00 84CA

```



# GUIDES DE FORMATION

Comment découvrir rapidement un logiciel comme *PageMaker* ou *Excel* ? Il existe plusieurs solutions. Chacun ses goûts, pas vrai ? En voici quatre :

- Utiliser immédiatement la disquette, sans consulter le manuel. C'est plein d'aléas... et ça peut vous coûter une disquette non protégée. Le ludographe que je suis affectionne beaucoup cette méthode car elle ressemble assez — sous certains aspects — à un jeu d'esprit. Je la déconseille néanmoins à celles et à ceux qui désirent se servir professionnellement d'un logiciel.
- Lire attentivement la documentation et le manuel de références, puis réaliser tranquillement, devant le clavier, les exemples (malheureusement peu nombreux) qui y sont proposés. C'est une bonne démarche, mais relativement longue. Certains points de la doc resteront forcément dans l'ombre. On devra sans doute acheter des ouvrages complémentaires.
- Suivre un cours de formation : il en existe d'excellents, de bons et de détestables. Ne pas hésiter à nous écrire quand c'est totalement "bidon". Cela arrive encore, hélas ! Nous publierons volontiers les lettres des mécontents, mais aussi celles des participants satisfaits.
- Les guides de formation : j'ai récemment pu compulsier à loisir ceux mis au point par EDIDACOM et j'avoue avoir été conquis par leur clarté et par le sens pédagogique qui a guidé leurs auteurs.

## VERSION SOFT

### dans une place forte gauloise !

Mais oui, ces champions du logiciel Apple s'installent du côté d'Alésia. Notez leur nouvelle adresse :

VERSION SOFT

44, rue d'Alésia 75014 PARIS Tél. : (16-1) 43.21.38.21

## EXCEL SUR MACINTOSH

Environ 140 pages d'exemples illustrés, allant de l'élaboration d'un tableau simple à la création d'un journal des ventes à double entrée, en passant par le grapheur d'*Excel* et les calculs statistiques... sans oublier les macro-commandes : de quoi satisfaire les plus exigeants. Une disquette d'accompagnement est évidemment fournie. Naturellement, on ne peut en tirer profit que si l'on possède la version originale d'*EXCEL*.

## PAGEMAKER SUR MACINTOSH

Qui ne rêve pas de publier son propre journal... sous forme de bulletin "de santé" ? On sait que c'est possible, avec *PageMaker*, un Macintosh et une imprimante LaserWriter. Mais que l'on ne s'y trompe pas : tout n'y est pas d'une évidente simplicité !

Le guide de formation d'Edidacom tend pourtant à prouver le contraire.

## D'AUTRES PROJETS

D'autres guides, réalisés sur le même modèle, sont prévus (notamment pour *WriterPlus*) et je crois qu'il existe aussi un guide de formation pour *APPLEWORKS* (mais sur Apple IIe). Nous ne manquerons pas de signaler, lors de leur parution, les intéressants ouvrages de cet éditeur.

G.H.

EDIDACOM 11, rue Marceau B.P.40 78800 HOUILLES — Tél. (1) 39.68.92.51

# ARC-EN-CIEL

Voulez-vous concevoir votre propre logiciel de dessin, sur Apple... et en couleurs ? Gageons que la réponse est oui. Le programme (c'est presque un logiciel !) d'Yves BUONO devrait répondre à votre attente. S'il ne brille pas par ses explications (mais les sources figurent dans la disquette *TREMLIN MICRO*), il a le mérite de fonctionner. Naturellement, il vous faut un Apple IIc ou IIe avec souris (un GS fera semble-t-il l'affaire)... et beaucoup de talent. Notez que les dessins reproduits page 51 (ils sont plus jolis sur un écran en couleurs) ont été obtenus à partir du programme, par l'auteur lui-même. Naturellement, **Arcenciel** est modifiable et perfectible. A vous de jouer !

Attention ! pour lancer **Arcenciel**, il est indispensable de passer par un startup qui installe le programme au-dessus des pages graphiques. Le voici :

## ARC.START

```
10 TEXT : NORMAL : PRINT CHR$
   (21) 93EF
15 POKE 24576,0: POKE 103,1: PO
   KE 104,96 2A24
20 PRINT CHR$ (4)"-ARCENCIEL" 161D
```

## ARCENCIEL

```
100 GOSUB 568 F753
102 GOTO 116 3F43
104 INPUT " ";X,Y,S D45F
106 IF X > XX THEN X = XX FB20
108 IF X < 2 THEN X = 2 9D26
110 IF Y > YY THEN Y = YY 0A26
112 IF Y < 1 THEN Y = 1 4026
114 RETURN 63B1
116 GOSUB 224: REM appel de la s
   ouris 0D48
118 HPLLOT 0,YL TO 0,0 TO XL,0 TO
   XL,YL TO 0,YL C49A
120 HPLLOT 1,0 TO 1,YL: HPLLOT XL
   - 1,0 TO XL - 1,YL B7DA
```

```
122 HOME : VTAB 21: PRINT "AVEZ-
   VOUS UNE IMAGE A CHARGER ?(O
   /N) ": GET K$ A903
124 ON K$ = "O" GOTO 130 20C5
126 ON K$ = "N" GOTO 222 A3C6
128 HOME : PRINT CHR$ (7): VTAB
   21: PRINT "Répondez par (O)u
   i ou (N)on. Merci": FOR I =
   0 TO 1500: NEXT : GOTO 122 F20E
130 HOME :H = 1: VTAB 21: INPUT
   "NOM DE L'IMAGE A CHARGER : "
   ";FI$ D8C9
132 ON FI$ = "" GOTO 138 E2C2
134 ONERR GOTO 138 88EC
136 GOTO 140 2D40
138 FLASH : PRINT "JE NE TROUVE
   PAS D'IMAGE DE CE NOM.": FOR
   I = 0 TO 2000: NEXT : NORMAL
   : GOTO 122 322D
140 PRINT D$;"BLOAD";FI$,"A$2000
   ": HOME : VTAB 21: PRINT "Vo
   us pouvez dessiner. Couleur
   Trace = ";CT 44EE
142 PRINT "ATTENTION ! Cliquet t
   enu ou appelez (T)": VTAB 23
   : PRINT "pour positionner le
   curseur sans tracer" 5CAB
144 VTAB 1: PRINT D$;"IN£4" 4D9C
146 NB = 0 2E90
```

## ARCENCIEL

148 GOSUB 104	0B45	:Y = 0: POKE VS,0: GOTO 116	C55A
150 IF NOT NB THEN IF S > 1 THEN HPLLOT X,Y:NB = 1	59C6	198 ON A\$ = "I" AND NOT Z GOTO 346	C0AB FC54
152 IF S > 1 THEN HPLLOT TO X,Y : VTAB 1: GOTO 148	AB84	200 GOSUB 596	
154 IF S = 1 THEN VTAB 1: SCALE = 1: ROT= 0: XDRAW 1 AT X,Y: FOR I = 1 TO 10: NEXT : XDRAW 1 AT X,Y: GOTO 144	B143	202 IF A\$ = "M" AND Z = 0 THEN PRINT D\$;"BSAVE/RAM/IMA0,A\$2000,L\$1FF8": GOTO 454	0B1B
156 HOME : NORMAL	946E	204 IF A\$ = "T" THEN GOSUB 302: GOTO 222	25FE
158 VTAB 1: PRINT D\$;"IN€0"	3598	206 ON A\$ = "R" GOTO 338	B0C8
160 VTAB 21: PRINT "(C)ouleur "(F)in """"(E)ffacer """"	D073	208 ON A\$ = "P" GOTO 236	B4C3
162 PRINT "(P)eindre "(S)auver "(I)rappel Image """	7399	210 ON A\$ = "G" GOTO 428	57BD
164 PRINT "(R)etouche (T)rait ""(G)omme ""CT = ";CT;	306A	212 GOTO 182	3946
166 IF F1 = 1 THEN VTAB 21: HTAB 38: FLASH : PRINT "X": NORMAL	D569	214 HOME : VTAB 22: HTAB 1: PRINT "COULEUR ? (0 à 7)": PRINT "( Liste des couleurs par ' / )": GET K\$	F5D5
168 VTAB 24: HTAB 1: PRINT "(M)odification image "(V)inVersion """"	BC0A	216 IF K\$ = "?" THEN GOSUB 586: GOTO 214	BF02
170 VTAB 21: HTAB 2: INVERSE : PRINT "C";: HTAB 13: PRINT "F";: HTAB 24: PRINT "E": NORMAL	030C	218 IF K\$ > "7" OR K\$ < "0" THEN CALL GL: HOME : VTAB 22: FLASH : PRINT "ERREUR ! 0 à 7 SEULEMENT !": FOR I = 0 TO 1500: NEXT : NORMAL : GOTO 214	7DB7 0B6F
172 VTAB 22: HTAB 2: INVERSE : PRINT "P";: HTAB 13: PRINT "S";: HTAB 24: PRINT "I": NORMAL	EC2B	220 CT = VAL (K\$): HCOLOR= CT	
174 VTAB 23: HTAB 2: INVERSE : PRINT "R";: HTAB 13: PRINT "T";: HTAB 24: PRINT "G": VTAB 24: HTAB 2: PRINT "M";: HTAB 24: PRINT "V";: NORMAL	9930	222 HOME : VTAB 21: PRINT "Vous pouvez dessiner. "Couleur Trace = ";CT: PRINT "(Pressez 'ESPACE' pour avoir le menu)": GOTO 144	0DAB
176 PRINT D\$;"IN€4"	E58F	224 REM	
178 VTAB 1: INPUT "";Z1,Z2,S: IF ABS (S) < 3 THEN 178	B699	226 PRINT D\$;"PR€4": PRINT CHR\$(1)	0DF7 5F96
180 PRINT D\$;"IN€0"	4E8B	228 PRINT D\$;"PR€0"	
182 VTAB 24: HTAB 40: GET A\$	8299	230 HOME :CT = 3: VTAB 24: PRINT "Couleur trace = ";CT;	7EFD
184 ON A\$ = "" OR A\$ = " " GOTO 182	49DA	232 HCOLOR= CT	3D29
186 ON A\$ = "C" GOTO 214	4CB2	234 RETURN	63B1
188 ON A\$ = "S" GOTO 326	59C6	236 GOSUB 556	0650
190 IF A\$ = "V" THEN & VX,Y: GOTO 222	9F9D	238 POKE VS,0: HOME : VTAB 21: PRINT "PLEIN OU TRAME (P/T)": PRINT "(Avez vous fermé les zones à peindre ?": PRINT "Si non, (Q) pour quitter !)": GET K\$: IF K\$ = "P" THEN POKE 214,0: GOTO 258	9280
192 IF A\$ = "X" THEN F1 = 0: GOTO 534	F83A	240 ON K\$ = "T" GOTO 246	5CD2
194 IF A\$ = "F" THEN TEXT : HOME : GOTO 374	FA0D	242 ON K\$ = "Q" GOTO 156	B7CF
196 IF A\$ = "E" THEN HGR :X = 0		244 HOME : VTAB 21: PRINT "REPONDEZ PAR 'P','T' ou 'Q'. MERCI": CALL GL: FOR I = 0 TO 1500: NEXT : GOTO 238	0267
		246 POKE VS,0: HOME : VTAB 21: HTAB 1: PRINT "Trame Simple,	



Double/fond blanc": PRINT "o			
u Simple/fond Noir (S/D/N)":			
GET K\$: IF K\$ = "S" THEN HO			
ME : VTAB 21: HTAB 1: PRINT			
"Pointillé simple, fond blan			
c": POKE 206,127: POKE 214,1			
: GOTO 254	CFE1		
248 IF K\$ = "D" THEN POKE 206,2			
55: POKE 214,2: HOME : VTAB			
21: HTAB 1: PRINT "Pointillé			
double / fond blanc": GOTO			
254	6AD9		
250 IF K\$ = "N" THEN POKE 206,1			
27: POKE 214,3: HOME : VTAB			
21: HTAB 1: PRINT "Pointillé			
sur fond noir :":PT = 1: GO			
TO 254	ADE3		
252 POKE VS,0: HOME : VTAB 21: H			
TAB 1: PRINT "REPONDEZ PAR S			
, D OU N. MERCI": CALL GL: F			
OR I = 0 TO 1500: NEXT : GOT			
O 246	2A85		
254 VTAB 22: HTAB 1: PRINT "VERT			
(1) "VIOLET (2)"	3A18		
256 VTAB 23: HTAB 1: PRINT "ORAN			
GE (5) "BLEU (6)": G			
OTO 402	BC2C		
258 POKE VS,0: HOME : VTAB 21: H			
TAB 1: PRINT "Rapport des to			
ns :"(1)/1 "(2)/1 "(3)/1": G			
ET K\$: FOR I = 1 TO 3	6592		
260 IF K\$ = STR\$(1) THEN POKE			
215,I + 1: VTAB 21: HTAB 14			
+ I * 7: INVERSE : PRINT I:			
NORMAL : GOTO 266	F6E2		
262 NEXT I	9DCB		
264 CALL GL: HOME : VTAB 21: HTA			
B 1: PRINT "Erreur ! Vérifie			
z votre choix. Merci": FOR I			
= 0 TO 1500: NEXT : GOTO 258	070C		
266 PRINT "LISTE COULEURS PAR ?"			
: GET K\$	AEDA		
268 ON K\$ = "?" GOSUB 586	95C9		
270 HOME : VTAB 21: HTAB 1	67D7		
272 INPUT "COULEUR 1 = ";C1\$: IF			
C1\$ = "0" OR C1\$ = "4" THEN			
CALL GL: HOME : VTAB 21: HTA			
B 1: PRINT "PAS DE NOIR EN C			
OULEUR 1. MERCI": FOR I = 0			
TO 1500: NEXT : GOTO 272	E7D2		
274 IF C1\$ > "7" OR C1\$ = "" OR			
LEN (C1\$) > 1 THEN GOSUB 40			
0: FOR I = 0 TO 1500: NEXT :			
GOTO 272	A772		
276 CALL 768, VAL (C1\$)		342B	
278 INPUT "COULEUR 2 = ";C2\$: IF			
C2\$ > "7" OR C2\$ = "" OR LE			
N (C2\$) > 1 THEN GOSUB 400:			
FOR I = 0 TO 1500: NEXT : GO			
TO 272	C839		
280 CALL 779, VAL (C2\$)		432E	
282 PRINT "CONFIRMATION SVP (O/N			
) :"		5737	
284 GET K\$		852D	
286 ON K\$ = "0" GOTO 296		22D2	
288 ON K\$ = "N" GOTO 300		C7C3	
290 HOME : VTAB 21: HTAB 1: INVE			
RSE : PRINT "CONFIRMATION SV			
P (O/N)": NORMAL		799D	
292 VTAB 22: HTAB 1: PRINT "COUL			
EUR 1 = ";C1\$: PRINT "COULEU			
R 2 = ";C2\$: GOTO 284		B47C	
294 INVERSE : PRINT K\$: NORMAL :			
PT = 0		9CB6	
296 REM			
298 & PX,Y		2DDC	
300 GOSUB 224: GOTO 222		3CC3	
302 IF HH = 1 THEN HOME : GOTO			
306		1A17	
304 HOME : VTAB 21: PRINT "Dépla			
cer le curseur cliquet relac			
hé": PRINT "Pour un trait: c			
liquet tenu": PRINT "puis re			
lâché en bout de trait": FOR			
I = 0 TO 3000: NEXT : HOME		69E3	
306 CALL GL		091F	
308 VTAB 22: PRINT "(T)rait"		2C13	
310 PRINT D\$;"IN£4": VTAB 1: GOS			
UB 104		FE1B	
312 SCALE= 1: ROT= 0		E3CC	
314 XDRAW 1 AT X,Y: FOR I = 1 TO			
10: NEXT : XDRAW 1 AT X,Y		8FED	
316 IF S = 2 THEN HPLLOT X,Y: &			
NX,Y		8E4A	
318 IF S = 3 THEN HPLLOT PEEK (			
8) + 256 * PEEK (7), PEEK (			
6) TO X,Y		A191	
320 IF S < 1 THEN RETURN		6E77	
322 HH = 1		1491	
324 GOTO 310		273F	
326 REM SAUVEGARDE EN RAM			
328 B\$ = "BSAVE/RAM/IMA": HOME :			
VTAB 21: PRINT "N" DE L'IMAG			
E A SAUVER EN RAM (1 OU 2)":			
GET R\$		12C0	
330 NI = VAL (R\$)		1913	
332 IF NI > 2 OR NI < 1 THEN CA			
LL GL: FLASH : VTAB 21: HTAB			



426 HOME : VTAB 21: HTAB 1: PRINT "ATTENTION AU N° DE COULEUR. MERCI": PRINT CHR\$(7): FOR I = 0 TO 1500: NEXT : GO TO 238	C216	10: NEXT : XDRAW 1 AT X,Y	6AEC
428 GOSUB 556: HOME : VTAB 21: PRINT "GOMME agit entre 2 points blancs"	BD12	490 IF S < 0 THEN GOSUB 224: POKE VS,0: GOTO 156	9886
430 PRINT D\$;"IN£0"	4E8B	492 IF S = 2 THEN &OX,Y:EB = PEEK(8) + 256 * PEEK(7):YB = PEEK(6)	2844
432 INPUT "Couleur de SUBSTITUTION ? (0 à 7) ";Z\$	F1FD	494 IF S < > 3 THEN VTAB 1: GO TO 472	ADEC
434 IF Z\$ > "7" OR Z\$ = "" OR LEN(Z\$) > 1 THEN CALL GL: HOME : VTAB 21: PRINT "ATTENTION AU N° DE COULEUR. MERCI": FOR I = 0 TO 1000: NEXT : GO TO 428	BB20	496 ON EB = X OR YB = Y GOTO 500	0D35
436 CT = VAL(Z\$): CALL 768,CT	0549	498 GOSUB 548: GOTO 504	19CF
438 HOME : VTAB 21: PRINT "Cliquer pour QUITTER": VTAB 21: HTAB 30: PRINT "Couleur = ";CT	5BEF	500 CALL GL: HOME : VTAB 21: PRINT "FENETRE NULLE.": PRINT "Déplacez le curseur. Merci": FOR I = 0 TO 2000: NEXT	B2EF
440 PRINT D\$;"IN£4"	E58F	502 GOSUB 224: GOTO 460	4BC7
442 VTAB 1: GOSUB 104	2952	504 HOME : VTAB 21: PRINT "POUR RECTIFIER VOTRE IMAGE, PRESSEZ 'Z'": VTAB 22: PRINT "OU BIEN PRESSEZ Q POUR QUITTER ' '": NORMAL	D278
444 SCALE= 1: ROT= 0	E3CC	506 PRINT D\$;"IN£0": GET K\$	2DF2
446 XDRAW 1 AT X,Y: FOR I = 0 TO 10: NEXT : XDRAW 1 AT X,Y	6AEC	508 IF K\$ = "Z" THEN Z = 1: GOTO 514	B028
448 & AX,Y	71CD	510 IF K\$ = "Q" THEN GOSUB 548: GOTO 156	5017
450 ON ABS(S) < > 2 GOTO 442	4C43	512 CALL GL: FLASH : GOTO 504	0176
452 POKE VS,0: GOTO 156	7A3F	514 POKE 230,64: CALL 879	6852
454 REM MODIFICATION DE L'IMAGE	185A	516 POKE VS,0	81BE
456 Z = 0	EA3C	518 IF X = 279 THEN GOSUB 546: GOTO 522	9108
458 PRINT D\$;"BLOAD/RAM/IMA0,A#4000,L#1FF8"	A1F5	520 CD = EB / 7:LD = YB:CA = INT(X / 7):LA = Y:C1 = CD:L1 = LD:C2 = CA:L2 = LA	3745
460 HOME : VTAB 21: PRINT "Définissez une fenetre avec la souris """;	5EEB	522 HGR	2391
462 PRINT "Positionnez le curseur puis, cliquez, """;	5EF2	524 & VX,Y	99E2
464 PRINT "déplacez cliquet tenu, relachez. """;	8B02	526 CALL 821,LD,CD,CA,LA	6CFF
466 PRINT " (Pressez 'ESPACE' pour avoir le Menu)";	B1FD	528 F1 = 1	CA78
468 CT = 3: HCOLOR= CT	4D9C	530 HOME : VTAB 21: PRINT "VOUS POUVEZ MODIFIER VOTRE DESSIN. """;	E492
470 VTAB 1: PRINT D\$;"IN£4"	D45F	532 PRINT "Votre modification finie, pressez (X). """: FOR I = 0 TO 2500: NEXT : GOTO 156	5262
472 INPUT "";X,Y,S	9D08	534 ON Z = 1 GOTO 538	B95A
474 IF X > XL THEN X = XL	6322	536 CALL GL: GOTO 156	4EA0
476 IF X < 0 THEN X = 0	260C	538 POKE 230,64: CALL 62450: POKE 230,32: CALL 879	3132
478 IF Y > YL THEN Y = YL	A124	540 CALL 821,L1,C1,C2,L2	B6BB
480 IF Y < 0 THEN Y = 0	E8DF	542 PRINT D\$;"BLOAD/RAM/IMA0,A#2000,L#1FF8"	962C
482 IF X = XL THEN 486	D2A7	544 & WX,Y: GOSUB 224:Z = 0: GOTO 156	FC7A
484 X = INT(X / 7) * 7	E3CC	546 CD = EB / 7:LD = YB:CA = 40: LA = Y:C1 = CD:L1 = LD:C2 =	



## ARCENCIEL

<pre> CA:L2 = LA: RETURN 548 IF EB &gt; X THEN TT = X:X = EB       :EB = TT 550 IF YB &gt; Y THEN TT = Y:Y = YB       :YB = TT 552 FOR I = EB TO X: XDRAW 1 AT       I,YB: NEXT : FOR I = YB TO Y       : XDRAW 1 AT X,I: NEXT 554 FOR J = YB TO Y: XDRAW 1 AT       EB,J: NEXT : FOR J = EB TO X       : XDRAW 1 AT J,Y: NEXT : RET       URN 556 HOME : VTAB 21: PRINT "Posit       ionnez le curseur, puis Cliq       uez" 558 PRINT D\$;"IN&amp;4": POKE VS,0 560 VTAB 1: GOSUB 104 562 SCALE= 1: ROT= 0: XDRAW 1 AT       X,Y: FOR I = 0 TO 10: NEXT :       XDRAW 1 AT X,Y: IF ABS (S)       &lt; &gt; 2 THEN 560 564 VTAB 1: INPUT " ";Z1,Z2,S: IF       ABS (S) &lt; 3 THEN 564 566 PRINT D\$;"IN&amp;0": POKE VS,0:       RETURN           </pre>	<p>8C16 1011 6B50 A9A2 DC7E FA34 E587 2952 7182 8C98 9D6E</p>	<pre> 568 D\$ = CHR\$ (4): DIM IM(6):IA       = 3:GL = - 1052 570 PRINT D\$;"BRUN PR19G.C" 572 PRINT D\$;"BLOAD TEINT.CAR.C" 574 PRINT D\$;"BLOAD TBADR.C" 576 PRINT D\$;"BLOAD TRANSIT8.C" 578 XL = 279:YL = 159:VS = - 16       368:H = 0:HH = 0 580 XX = XL - 2:YY = YL - 1: HGR 582 POKE 232,26: POKE 233,3 584 RETURN 586 HOME : VTAB 21: PRINT " (0)       "Noir0" TAB( 22)"(4) "Noir1" 588 PRINT " (1) "Vert" TAB( 22)"       (5) "Orange" 590 PRINT " (2) "Violet" TAB( 22       )" (6) "Bleu" 592 PRINT " (3) "Blanc0" TAB( 22       )" (7) "Blanc1ctK"; 594 FOR I = 0 TO 3000: NEXT : RE       TURN 596 IF Z = 1 THEN RETURN 598 B\$ = "BSAVE/RAM/IMA" + STR\$       (IA) + ",A\$2000,L\$1FF8": PRI       NT D\$;B\$:IM(IA) = 1 600 IA = IA + 1: IF IA &gt; 6 THEN       IA = 3 602 H = 1: RETURN           </pre>	<p>4476 A1BC AE1C C701 48B6 2834 7045 67CE 63B1 2916 2034 4834 4CA2 AAF5 E07D 2B20 EFA4 1334</p>
--	---	--	--

/TMF/ARC/TEINT.CAR.C,A\$0300,L\$0032

0300: 20 B1 00 20 E9 F6 A5 E4 85 CE 60 20 B1 00 20 E9	A5E6
0310: F6 A5 E4 85 CF A5 CE 85 E4 60 01 00 04 00 96 0A	93B4
0320: 2D 25 24 0C 18 24 3C 3F 1F 3F 37 36 16 36 2E 2D	7DAB
0330: 05 00	1E05

/TMF/ARC/TBADR.C,A\$036F,L\$0023

036F: A5	4BA5
0370: FE 8D 34 03 A9 00 85 FE A5 FE 20 11 F4 A4 FE A5	36FD
0380: 26 99 00 90 A5 27 05 E6 99 C0 90 E6 FE C0 BF D0	5D22
0390: E7 60	2F47

/TMF/ARC/TRANSIT8.C,A\$0335,L\$003A

0335: 20 F5 E6 86 06 20 F5 E6 86 07 20	E92F
0340: F5 E6 86 08 20 F5 E6 86 09 A6 06 BD 00 90 85 F9	F26A
0350: 85 CE BD C0 90 85 FA 49 60 85 CF A4 07 B1 F9 91	30C2
0360: CE C8 C4 08 D0 F7 E8 E4 09 F0 03 4C 4B 03 60	D0EB

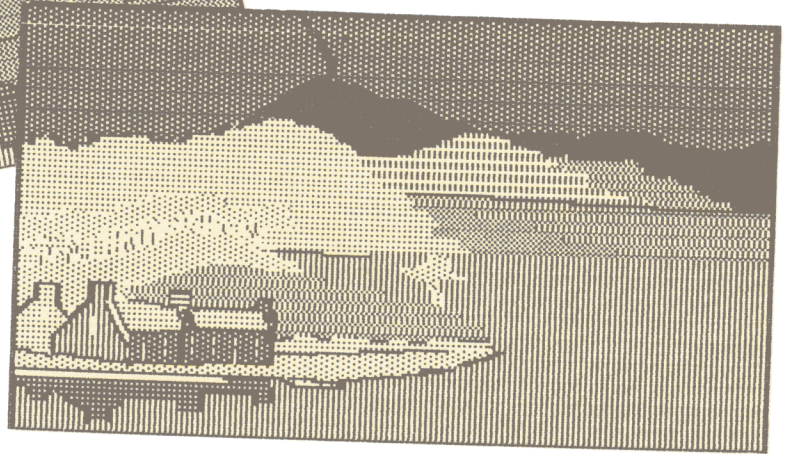
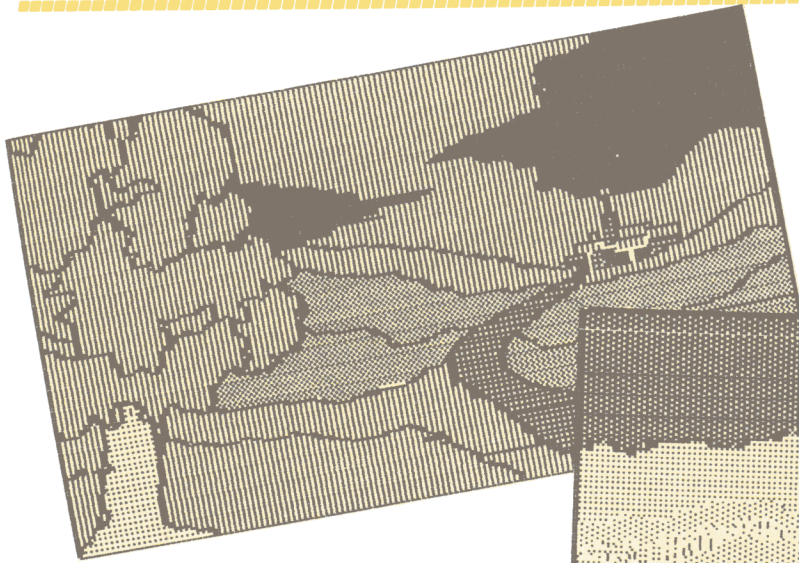
*Tapez aussi ces 3 petites routines, mais n'oubliez pas le programme LM principal (page suivante).*

*Si vous êtes intéressé(e) par les sources de ces programmes, commandez tout simplement la disquette TREMLIN MICRO n°15. Bulletin spécial à la fin de la revue.*



# ARC-EN-CIEL

Routine LM principale  
PR19G.C



Deux exemples — malheureusement en noir et blanc — de dessins réalisables avec **ARCENCIEL**.

/TMF/ARC/PR19G.C,A#0C00,L#0683

0C00:	A9 4C 8D F5 03 A0 10 8C F6 03 A0 0C 8C F7 03 60 85 0B 20 02 E1 A5 A0 85	499E
0C18:	07 85 4B A5 A1 85 08 85 4C 20 F5 E6 86 06 86 4A A5 0B C9 57 F0 13 C9 41	05AF
0C30:	F0 12 C9 50 F0 15 C9 52 F0 0D C9 56 D0 0C 4C 0C 11 4C 2B 11 4C CC 10 4C	0698
0C48:	96 0F 60 20 65 11 85 4D A9 04 85 FF 85 0C E6 06 20 13 0D F0 13 20 5E 0F	F6EB
0C60:	D0 0B 20 5E 0F D0 03 20 5D 0D 20 71 0F 20 71 0F C6 06 C6 06 20 13 0D F0	19CD
0C78:	13 20 5E 0F D0 0B 20 5E 0F D0 03 20 5D 0D 20 71 0F 20 71 0F E6 06 20 6A	BC1B
0C90:	0F 20 13 0D F0 C0 20 71 0F A5 07 85 1B A5 08 85 1A A5 4B 85 07 A5 4C 85	DD29
0CA8:	08 E6 06 20 13 0D F0 13 20 64 0F D0 0B 20 64 0F D0 0B 20 64 0F D0 03 20 66	3937
0CC0:	20 6A 0F C6 06 C6 06 20 13 0D F0 13 20 64 0F D0 0B 20 64 0F D0 0B 20 64 0F	79CE
0CD8:	0D 20 6A 0F 20 6A 0F E6 06 20 71 0F 20 13 0D F0 C0 20 BC 0D A5 4A C5 06	895E
0CF0:	B0 10 90 04 A5 4A 85 06 E6 06 20 0B 0D F0 24 4C 8A 0D C6 06 20 0B 0D D0	C8BD
0D08:	EB F0 35 A5 4B 85 07 A5 4C 85 08 20 7A 11 A5 30 29 7F 31 26 F0 02 A9 01	8925
0D20:	85 09 60 20 6A 0F 20 13 0D 20 71 0F A5 09 F0 0D 20 71 0F 20 13 0D 20 6A	897C
0D38:	0F A5 09 D0 4D 4C 56 0C 20 6A 0F 20 13 0D 20 71 0F A5 09 F0 0D 20 71 0F	244C
0D50:	20 13 0D 20 6A 0F A5 09 D0 9A 4C 56 0C 20 71 0F 20 6F 0D 4C 6A 0F 20 6A	BC2A
0D68:	0F 20 6F 0D 4C 71 0F A6 4D E0 FF F0 14 A5 08 9D 83 11 E8 A5 07 9D 83 11	4AF0
0D80:	E8 A5 06 9D 83 11 E8 86 4D 60 A6 4D F0 25 CA BD 83 11 85 06 85 4A CA BD	79E3
0D98:	83 11 85 4B 85 07 CA BD 83 11 85 08 85 4C 86 4D 20 13 0D F0 03 4C 23 0D	BBEB
0DB0:	4C 56 0C A5 D6 C9 03 90 D0 4C 03 0E A5 D6 D0 07 A5 D7 85 0C 4C 39 0F A5	A54A
0DC8:	CE 20 4D 0F A5 D6 C9 02 D0 03 4C B2 0E C9 01 F0 74 A4 EE 98 18 69 05 A5	CBF2
0DE0:	EF 69 00 C9 20 F0 A2 A5 08 20 7A 0F A5 07 20 7A 0F A5 1A 20 7A 0F A5 1B	68A6
0DF8:	20 7A 0F A5 06 20 7A 0F 84 EE 60 A5 EE 8D 32 03 A5 EF 8D 33 03 20 65 11	8811
0E10:	F0 0E A5 EF CD 33 03 D0 07 A5 EE CD 32 03 F0 E2 A4 EE B1 1E 85 08 20 7C	195D
0E28:	0F B1 1E 85 07 20 7C 0F B1 1E 85 1A 20 7C 0F B1 1E 85 1B 20 7C 0F B1 1E	A217
0E40:	85 06 20 7C 0F 84 EE 20 E5 0E 4C 12 0E A5 CF C9 2A F0 0C C9 AA F0 08 C9	21BE
0E58:	55 F0 2C C9 D5 F0 28 A9 D5 85 FD A5 06 4A 90 10 A9 00 F0 0E 20 6A 0F 20	611C
0E70:	84 0F D0 08 4C 71 0F EA A9 02 85 19 20 73 10 A5 EC C5 19 D0 E7 F0 12 A9	3CDE
0E88:	AA 85 FD A5 06 4A 90 05 A9 03 D0 E6 EA A9 01 D0 E1 A5 CF 25 FD 85 E4 20	FB7C
0EA0:	74 11 A6 FF 20 6A 0F 20 84 0F D0 01 60 CA D0 F4 F0 ED A5 06 85 EC A9 00	C3D7
0EB8:	20 79 10 A5 CF C9 2A F0 0C C9 AA F0 08 C9 55 F0 10 C9 D5 F0 0C A9 D5 85	4032
0ED0:	FD A5 EC C9 02 B0 A0 90 8F A9 AA 85 FD A5 EC C9 02 90 B1 B0 AB A5 CF C9	0DD2
0EE8:	2A F0 0C C9 AA F0 08 C9 55 F0 35 C9 D5 F0 31 A9 D5 85 FD A5 06 4A 90 0F	A327

(Suite page 52)

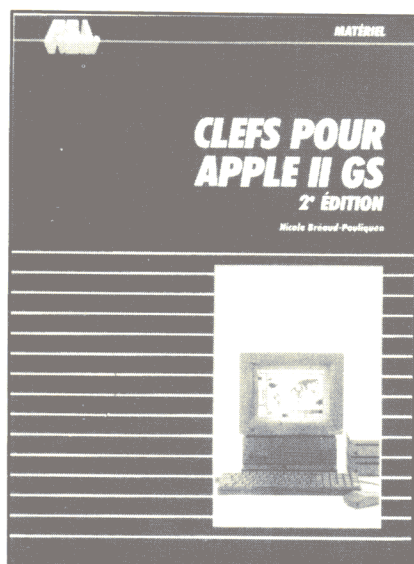


## ARC-EN-CIEL (suite et fin de PR19G.C)

0F00:	A9 00 F0 0D 20 6A 0F 20 84 0F D0 07 4C 71 0F A9 02 85 19 20 73 10 A5 EC	FD12
0F18:	C5 19 F0 E8 A5 CF 25 FD 85 E4 20 74 11 4C 04 0F A9 AA 85 FD A5 06 4A 90	9013
0F30:	04 A9 03 D0 DC A9 01 D0 D8 A5 06 85 EC A9 00 20 79 10 A5 EC F0 05 A5 CE	8915
0F48:	4C 4D 0F A5 CF 85 E4 20 6A 0F 20 7A 11 A5 1A A6 1B A4 06 4C 3A F5 20 6A	CEF8
0F60:	0F 4C 13 0D 20 71 0F 4C 13 0D E6 08 D0 02 E6 07 60 A5 08 D0 02 C6 07 C6	35A6
0F78:	08 60 91 1E C8 D0 04 E6 1F E6 EF 60 A5 08 C5 1A D0 06 A5 07 C5 1B F0 03	7CCE
0F90:	A9 01 60 A9 00 60 2C 10 C0 2C 00 C0 10 FB AD 00 C0 85 0A C9 51 F0 10 A5	BDC1
0FA8:	0A C9 C7 F0 0D C9 C4 F0 2E C9 C2 F0 12 D0 DF 4C 84 FE A9 06 20 5E 11 A9	F333
0FC0:	C7 20 50 11 4C D9 0F A9 17 20 5E 11 A9 C2 20 50 11 F0 04 A9 FF D0 02 A9	14CE
0FD8:	7F 20 19 10 4C 21 10 A9 0E 20 5E 11 A9 C4 20 80 FE 20 F0 FD 2C 10 C0 2C	87CB
0FF0:	00 C0 10 FB AD 00 C0 C9 B1 F0 12 C9 B2 F0 0E C9 B5 F0 0A C9 B6 F0 06 C9	E1E3
1008:	D1 F0 AC D0 D2 29 07 AA 20 EC F6 20 1B 10 4C 21 10 85 E4 20 74 11 4C 66	AF73
1020:	10 2C 10 C0 2C 00 C0 10 FB 20 63 10 AD 00 C0 C9 D1 F0 D6 C9 95 F0 0E C9	6F88
1038:	88 F0 13 C9 8B F0 1A C9 8A F0 1B D0 06 E6 08 D0 02 E6 07 4C A7 0F A5 08	3479
1050:	D0 02 C6 07 C6 08 4C A7 0F C6 06 4C A7 0F E6 06 4C A7 0F 20 7A 11 A9 01	7680
1068:	85 E7 A9 00 A2 1E A0 03 4C 5D F6 A5 08 85 EC A5 07 85 ED 20 B2 10 A5 EC	ECC6
1080:	C5 0C 90 0E 38 E5 0C 85 EC E6 F9 D0 F1 E6 FA 4C 7E 10 A5 ED F0 35 18 A5	B4D7
1098:	F9 65 FE 85 F9 90 02 E6 FA 18 A5 EC 65 FB 85 EC 90 02 E6 ED C6 ED F0 CE	AA9C
10B0:	D0 E0 A9 00 85 F9 85 FA 85 FE 85 FB A9 FF 38 E5 0C E6 FE C5 0C B0 F8 69	66F0
10C8:	01 85 FB 60 EA 20 13 0D F0 08 20 6A 0F 20 13 0D D0 06 20 6A 0F 4C CD 10	2374
10E0:	20 71 0F 20 71 0F A5 07 85 1B A5 08 85 1A A5 4B 85 07 A5 4C 85 08 20 71	C563
10F8:	0F 20 13 0D F0 F8 20 71 0F 20 13 0D F0 F0 20 6A 0F 4C 4F 0F A9 00 85 06	2B6E
1110:	A2 00 A0 00 20 11 F4 B1 26 49 FF 91 26 C8 C0 28 D0 F5 E6 06 A5 06 C9 A0	57B2
1128:	D0 E6 60 A9 20 85 07 A9 40 85 09 A9 00 85 06 85 08 A8 A2 20 EA B1 08 F0	09A0
1140:	04 31 08 91 06 C8 D0 F5 E6 07 E6 09 CA D0 ED 60 20 80 FE 20 F0 FD 20 7A	5C69
1158:	11 B1 26 29 80 60 85 24 A9 15 4C C1 FB A9 83 85 1E A9 12 85 1F A9 00 85	FCBC
1170:	EE 85 EF 60 20 7A 11 4C 5A F4 A5 06 A6 08 A4 07 4C 11 F4 00 00 00 00 00	2D5C
1188:	00 00	0000
11A0:	00 00	0000
11B8:	00 00	0000

Continuez avec des zéros jusqu'à \$1282 inclus. **Sauvegarde** : BSAVE PR19G.C, A\$C00, L\$683

## Voici enfin la seconde édition des **CLEFS POUR LE GS**



Ouvrage indispensable, même s'il ne constitue pas une panacée. Dans cette réédition, Nicole Bréaud-Pouliquen a revu sa copie et tenu largement compte des compléments d'information dont elle a pu disposer. C'est dire que le contenu est beaucoup plus conforme à la machine actuelle que celui de la première édition (qui concernait surtout le prototype du GS). La présentation de ces **CLEFS POUR L'APPLE II GS** bénéficie aussi des améliorations que l'on note aujourd'hui dans la plupart des ouvrages du PSI.

Qui doit lire (pardon : étudier) le livre de Nicole Bréaud-Pouliquen ? Toute personne qui, possédant, un GS, souhaite le programmer elle-même. Le lecteur (ou la lectrice) disposera non seulement de la liste des instructions du 65C816, mais de celle (interminable) des outils. Il apprendra comment utiliser l'Assembleur... Bref, ces **CLEFS** lui permettront de ne plus avoir l'impression d'être devant une belle fille — le GS — aveugle, sourde... et muette.

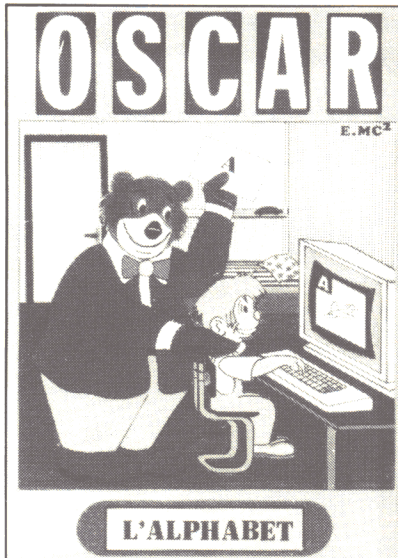
Nestor.



# ÉDUCATION

et

# JEUX



## OSCAR L'ALPHABET

Les logiciels de la série OSCAR fonctionnent sur Apple II muni de deux lecteurs de disquettes et d'un joystick. Un moniteur couleur est souhaitable d'autant plus que le graphisme est soigné. Pour celui-ci, on pourrait se passer du second lecteur, mais à condition de négliger le module "L'alphabet en images", ce qui serait dommage. Pour les deux autres, un seul lecteur est suffisant. Utilisation simple, réellement à la portée d'enfants de 3 à 8 ans. Documentation succincte, mais suffisante. Des pastilles autocollantes de couleur sont fournies pour différencier les deux boutons du joystick.

## OSCAR

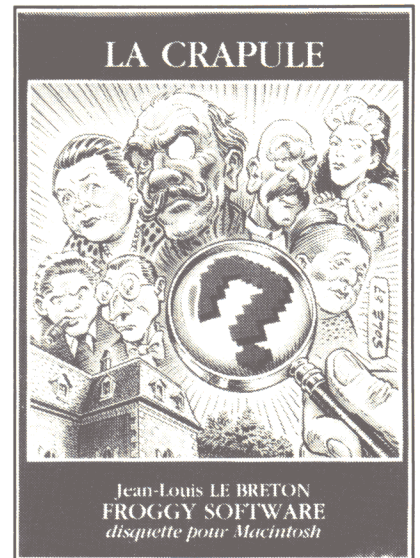
### L'ALBUM enchanté

Même auteur que le précédent (L. Villain). C'est un logiciel d'éveil pour les petits : reconnaissance d'animaux, d'objets etc. à travers un album qui parle (synthèse vocale). Il est utilisable à partir du joystick ou du clavier et comporte une vingtaine de puzzles élémentaires.

## OSCAR PUZZLES

Deux niveaux de difficulté : 4 ou 6 pièces à remettre en ordre. Mode d'emploi on ne peut plus simple. Où il est prouvé qu'il est possible de donner des "leçons" de logique à de très jeunes enfants.

API, 12 chemin du Petit Etang 86000 POITIERS



## LA CRAPULE MAC

Si vous possédez un MAC 512 ou MAC+, transformez-vous en détective et suivez la piste... fléchée pour vous par Jean-Louis LE BRETON.

Je vous préviens, c'est coton ! Heureusement, si vous tournez trop longtemps en rond, vous aurez la possibilité d'écrire au docteur Froggy pour obtenir la solution. Voici son adresse :

**Docteur Froggy**  
34, rue Henri Chevreau  
75020 PARIS

C'est aussi celle de FROGGY SOFTWARE... où vous obtiendrez tous les renseignements sur les productions de Jean-Louis LE BRETON (Tél. [16-1] 43. 58.25.98).



# GRAPH.EXPRESS

ROUTINE  
DEMANDÉE  
MODE TEXT

```

100 TEXT : NORMAL :D$ = CHR$ (4): PRINT D$"PRÉ3": PRINT : HOME      7A9B
105 DIM M(12),AN(12)                                               51F6
110 PRINT D$"RESTORE GRAPH.VAR"                                    AF33
115 G$ = "*****"                                                DF31
120 VTAB 23: PRINT "<T>erminé (A)bréger (C)orriger la valeur précédent
    e (RET) Valeur non modifiée"; VTAB 1: PRINT                    AE78
125 I = 0: PRINT : FOR J = 1 TO N: VTAB 6 + J: PRINT AN(J);: HTAB 7: P
    RINT M(J): NEXT                                               186E
130 I = I + 1                                                       805B
135 GOSUB 275                                                       ED4E
140 GOSUB 280: INPUT "":A$: IF NOT LEN (A$) THEN 165              BF92
145 IF A$ = "C" AND I > 1 THEN GOSUB 275:I = I - 1: GOTO 135      2FA5
150 IF A$ = "T" THEN GOSUB 275:I = N: GOTO 200                   C0A4
155 IF A$ = "A" THEN CALL - 958:I = I - 1: GOTO 200              2F33
160 AN(I) = VAL (A$)                                               1B94
165 GOSUB 275                                                       ED4E
170 GOSUB 290: INPUT "":M$: IF NOT LEN (M$) THEN 190              3EA9
175 IF M$ = "C" THEN 135                                           CAD2
180 IF M$ = "T" OR M$ = "A" THEN A$ = M$: GOTO 150               D3FF
185 M(I) = VAL (M$)                                               2D5E
190 GOSUB 275                                                       ED4E
195 IF I < 12 THEN 130                                             B082
200 FOR J = 1 TO I: IF M(J) > G THEN G = M(J)                       197E
205 NEXT                                                            0582
210 D = G / 70                                                      548D
215 VTAB 22: CALL - 958: INPUT "TITRE :<u>tG ":R$: IF LEN (R$) THEN TX
    $ = R$                                                         6F62
220 HOME : VTAB 5: IF LEN (TX$) THEN HTAB (80 - LEN (TX$)) / 2: PRI
    NT TX$: PRINT                                                  2B28
225 FOR J = 1 TO I: PRINT AN(J);" "": IF NOT M(J) THEN 235        CF01
230 HTAB 17 - LEN ( STR$ (M(J))): PRINT M(J);: HTAB 20: PRINT LEFT$
    (G$, INT (M(J) / D))                                           3840
235 NEXT                                                            0582
240 PRINT : PRINT : PRINT "Echelle: 1/" INT (D)                   4CFA
245 N = I: PRINT D$"STORE GRAPH.VAR"                               073D
250 VTAB 22: PRINT : CALL - 198: PRINT "(C)ORRIGER (M)ENU DE DISQUETT
    E (A)PPLESOFT ";: GET R$                                       8B20
255 IF R$ = "C" THEN 125                                           57D6
260 IF R$ = "M" THEN PRINT D$"RUN MENU"                            CAF8
265 IF R$ = "A" THEN HOME : END                                    8F8D
270 GOTO 250                                                        1142
275 GOSUB 280: GOTO 285                                           F5CE
280 VTAB 5 + I: PRINT : RETURN                                     95C7
285 CALL - 868: PRINT AN(I);: HTAB 7: PRINT M(I): RETURN         D32D
290 VTAB 5 + I: PRINT : HTAB 7: RETURN                            D9CE

```





# LA SOURIS ET L'ASSEMBLEUR

## TROISIÈME PARTIE : Traitement de l'interruption et limites

*Et voici venir un épisode supplémentaire dans la conquête de la Souris (rassurez-vous, cela se terminera bientôt... ce n'est pas Dallas, que diable !).*

### Au menu de ce jour :

- Traitement de l'interruption avec tests du bouton (INTERUP).
- Etablissement des limites d'excursion de la Souris.
- Recherche des coordonnées.
- Réduction des coordonnées pour affichage du pointeur (la prochaine fois) à l'écran.

Le nouveau source étant arrivé (mais pas le Beaujolais !), mettez-le directement à la suite de celui de la dernière fois, à partir de l'adresse \$60D8 (début de INTERUP). N'oubliez pas, pour ceux qui utilisent un Assembleur (ProCODE), de rajouter les labels définis au début, à ceux existant déjà.

De plus, il faudra insérer les lignes suivantes pour remplacer une partie des quelques octets (NOP) réservés précédemment (ligne 38), afin d'utiliser les nouvelles routines :

601A :	20 2B 61	JSR LIMITES	; Appel de la routine fixant les limites
601D :	A0 18	LDY £HOME	; Offset de HOMEMOUSE
601F :	20 34 60	JSR CALLCARD	; ...
6022 :	A0 14	LDY £READ	; Offset de READMOUSE
6024 :	20 34 60	JSR CALLCARD	; ...
6027 :	20 63 61	JSR NEWCOORD	; Coordonnées actuelles

### Passons maintenant aux commentaires :

A chaque extinction verticale de l'écran (60 fois par seconde), le 6502 viendra se brancher sur la routine INTERUP, le vecteur d'interruption (\$3FE-\$3FF) étant initialisé à cette adresse.

Reportez-vous au listing pour voir et comprendre la démarche suivie dans les tests sur le bouton (4 cas possibles), en vous aidant, bien évidemment, de ce que je vous ai expliqué dans la première partie.

### Détermination des limites :

Souvenez-vous : lors du premier article, je vous avais précisé que les limites devaient être établies beaucoup plus loin que celles de l'écran. En fait, et comme exemple, elles ont été définies comme suit :

\$0000	← Abscisse	← \$022F	ou
0	← Abscisse	← 559	
\$0000	← Ordonnée	← \$0167	ou
0	← Ordonnée	← 359	

Les facteurs de réduction respectifs sont donc de 7 ((7 × 80) - 1) pour X et de 15 ((15 × 24) - 1) pour Y. Il vous est loisible de modifier ces paramètres, et vous pouvez le faire en adoptant la démarche suivante :

- Modifier la valeur des limites en X et Y en tenant compte du ou des nouveaux facteurs de réduction.
- Modifier la valeur de ces facteurs en ligne 114 du nouveau source.

(suite page 56)



## Détermination des nouvelles coordonnées et réduction de celles-ci dans les limites de l'écran...

C'est le sous-programme NEWCOORD qui se charge d'aller chercher les nouvelles coordonnées dans les trous d'écran correspondants, et de les transmettre à REDUIT pour traitement. REDUIT effectue une division 16 bits (coordonnée trou d'écran) / 8 bits (facteur), et renvoie un paramètre (dans le registre A) compris dans les bornes définies, en X ou Y, de l'écran.

Le résultat est stocké selon X ou Y dans les mémoires nommées QUOTX et QUOTY.

Pour plus de précisions sur les divisions en langage Machine, je vous renvoie, comme la première fois, à l'ouvrage "LA PROGRAMMATION DU 6502" de RODNAY ZAKS.

Cette fois, il se passera quelque chose de significatif. En effet, si vous appuyez sur le bouton, vous entendrez un bip. Ceci n'a évidemment que valeur de démonstration, et vous trouverez, j'en suis sûr, les traitements les plus divers pour l'utilisation du bouton.

Je vous laisse maintenant assimiler cette nouvelle étape dans la connaissance de la Souris, car la prochaine fois, oh merveille !, vous verrez enfin bouger le pointeur sur l'écran.

A bientôt...

François GALLET

**NOTA :** Si vous travaillez sous Moniteur, chargez d'abord le SOURIS.C que vous avez sauvé la dernière fois, puis rajoutez les octets de l'extension d'aujourd'hui, et sauvegardez enfin le tout par :

**BSAVE SOURIS.C,A\$6000,L436**

Lancez par **CALL 24576** en Basic ou **\$6000G** sous Moniteur.

```

0 *****
1 *
2 *   Programmation de la Souris : Troisième partie *
3 *
4 *****
5 *
6 *
7 BELL      EQU   $FF3A      :Emet un bip
8 CLAMP     EQU   $17        :Fixe de nouvelles limites aux coord. X et Y
9 ETAT      EQU   $77C      :Etat du bouton et des interruptions
10 HB       EQU   $1A        ;
11 HBX      EQU   $57C      :Octet de poids fort de l'abscisse retournée par
12          :la souris (voir aussi LBX)
13 HBY      EQU   $5FC      :Octet de poids fort de l'ordonnée retournée par
14          :la souris (voir aussi LBY)
15 HOME     EQU   $18        :Positionne les registres de position de la souris
16          :aux valeurs inférieures des limites , soit en
17          :haut à gauche . Doit être suivi de READ
18 LB       EQU   $19        ;
19 LBX      EQU   $47C      :Octet de poids faible de l'abscisse retournée par
20          :la souris (voir aussi HBX)
21 LBY      EQU   $4FC      :Octet de poids faible de l'ordonnée retournée par
22          :la souris (voir aussi HBY)
23 READ     EQU   $14        :Transfère les données en cours de la souris
24          :vers les trous d'écran
25 SERVE    EQU   $13        :Déetecte si la souris à causé l'interruption
26 SET      EQU   $12        :Initialise la souris avec le mode adéquat
27 *
28          ORG   $60D8
29 *
38 *
39 *
40 *   Sous programme : INTERUP
41 *

```

```

42 *
43 *
44 INTERUP EQU *
45 SEI ;On interdit les interruptions
46 *
47 PHA ;Sauvegarde
48 PHX ;...du
49 PHY ;contexte
50 *
51 LDY $SERVE ;Offset de SERVEMOUSE
52 JSR CALLCARD ;...
53 BCC CLICK ;Si C = 1 , la souris n'est pas la cause de
54 ;l'interruption
55 JMP RET.INT
56 CLICK LDY $READ ;Offset de READMOUSE
57 JSR CALLCARD ;...
58 *
59 LDA ETAT ;Etat du bouton et des interruptions
60 ASL ;Test du bit 7 pour savoir si le bouton est
61 ;enfoncé (présentement , là , dis-donc !)
62 BCS DOWN ;C = 0 si relâché ou 1 si enfoncé
63 JMP UP
64 DOWN ASL ;Le click est enfoncé , mais l'était-il la fois
65 ;précédente ? (là est la question !)
66 BCC D.BEF ;C = 0 si relâché ou 1 si enfoncé
67 JMP UP.BEF
68 *
69 * Cas : Enfoncé (maintenant) / Relâché (avant)
70 *
71 D.BEF JSR BELL
72 NOP
73 NOP
74 NOP
75 JMP RET.INT ;et bye-bye
76 *
77 UP ASL ;Test du bit 6 pour savoir s'il l'était auparavant
78 BCS DOWN.BEF ;C = 0 si relâché ou 1 si enfoncé
79 *
80 * Cas : Relâché (maintenant) / Relâché (avant)
81 *
82 NOP
83 NOP
84 NOP
85 JMP RET.INT ;et retour
86 *
87 * Cas : Relâché (maintenant) / Enfoncé (avant)
88 *
89 DOWN.BEF JSR BELL
90 NOP
91 NOP
92 NOP
93 JMP RET.INT ;Retour d'interruption
94 *
95 * Cas : Enfoncé (maintenant) / Enfoncé (avant)

```





```

96 *
97 UP.BEF JSR BELL
98 NOP
99 NOP
100 NOP
101 JMP RET.INT
102 *
103 RET.INT PLY ;Restauration
104 PLX ;...du
105 PLA ;contexte
106 CLI ;Rétablit les interruptions
107 RTI ;...et retour d'interruption
108 *
109 *
110 * Stockage de paramètres
111 * -----
112 *
113 COL40 HEX 00 ;Mémorise l'abscisse modulo 40
114 FACTEUR HEX 070F ;Facteur 07 pour X et 0F pour Y
115 MEMORY HEX 00 ;Mémorise le caractère écrasé par le pointeur
116 QUOT HEX 0000 ;Mémoire des nouvelles abscisses et ordonnées
117 QUOTX HEX 00 ;Résultat de la division des coord. pour X
118 QUOTY HEX 00 ;Résultat de la division des coord. pour Y
119 *
120 *
121 * Sous programme : LIMITES
122 * -----
123 *
124 LIMITES LDA f$00 ;...
125 STA $478 ;...LB de la lim. inf.
126 STA $578 ;...HB de la lim. inf.
127 LDA f$2F ;...
128 STA $4F8 ;...LB de la lim. sup.
129 LDA f$02 ;...
130 STA $5F8 ;...HB de la lim. sup.
131 LDA f$00 ;Modification des limites de l'abscisse
132 LDY fCLAMP ;Offset de CLAMPMOUSE
133 JSR CALLCARD ;...
134 *
135 LDA f$00 ;...
136 STA $478 ;...LB de la lim. inf.
137 STA $578 ;...HB de la lim. inf.
138 LDA f$67 ;...
139 STA $4F8 ;...LB de la lim. sup.
140 LDA f$01 ;...
141 STA $5F8 ;...HB de la lim. sup.
142 LDA f$01 ;Modification des limites de l'ordonnée
143 LDY fCLAMP ;Offset de CLAMPMOUSE
144 JSR CALLCARD ;...
145 LDY fREAD ;Offset de READMOUSE
146 JSR CALLCARD ;...
147 RTS ;Retour d'appel
148 *
149 *

```



# DU CÔTÉ DES COMPATIBLES

## LE TRAITEMENT DE TEXTE

Quand on a la bonne habitude de travailler avec un traitement de texte aussi simple et convivial que GRIBOUILLE, EPISTOLE... ou APPLE WRITER (l'un des plus connus dans l'univers Apple), il n'est pas toujours facile de s'adapter aux logiciels fonctionnant sur IBM PC et compatibles. Plaignons lectrices et lecteurs que leurs occupations obligent pourtant à accomplir cette petite gymnastique.

**CLAVIER** Côté clavier, constatons que nous ne sommes pas particulièrement gâtés avec nos Apple II. Une seule "bête" est enfin dotée d'un clavier digne de ce nom : le GS. Ce n'est pas le summum du confort, mais cela vaut bien les "accessoires" qui équipent bon nombre de compatibles. Il est vrai que nous sommes logés à la même enseigne avec l'Apple IIc. Disons que, malheureusement, le clavier reste trop souvent le point faible des ordinateurs personnels qui se veulent par ailleurs des machines de traitement de texte. Bien... mais le clavier, aussi mauvais soit-il, on s'y fait. L'homme s'adapte à tout, y compris au pire, pas vrai ?

**COMMANDES** Il en va autrement pour les commandes. Ah ! les commandes ! L'absence de touches de fonction fait que, sur Apple, tout traitement de texte prend immédiatement des allures de langue étrangère. Et on sait que les Français ne sont pas tellement doués pour "causer" dans une autre langue que la leur. Néanmoins, de logiciel en logiciel, nous avons pris l'habitude de jongler avec les contrôles- ceci-cela, puis avec les Pommes (ouvertes ou fermées), sans oublier la touche Escape, ô combien sollicitée !

**SOURIS** Certains d'entre nous sont tellement attachés à ce système qu'ils se refusent à caresser la souris, remède à tous les maux dont souffraient les premiers traitements de texte. Et c'est vrai que l'usage de la souris — prolongement naturel de la pensée ? — pose parfois plus de problèmes qu'il n'en résout : essayez plutôt de taquiner l'animal sur un bureau encombré ! Il en va de même pour les menus dits déroulants et pour ces multiples fenêtres,

agréables à la vue, mais dont l'ouverture et la fermeture peuvent se révéler fastidieuses.

**L'IDÉAL** Alors, existe-t-il un traitement de texte idéal ? Bien sûr que oui : le vôtre, et uniquement celui-ci. Vous l'avez élu pour des raisons qui vous sont personnelles et vous tenez à lui parce qu'il tient lui-même ses promesses. Il n'y a rien à ajouter. Je connais des inconditionnels d'Apple Writer pour lesquels tout le reste n'est que pâle copie.

Reste que — profession oblige — ces mêmes personnes se retrouvent parfois (je crois pouvoir écrire "souvent") devant le clavier d'un IBM ou d'un compatible (que cet autre univers est donc compliqué !), et obligés de "faire avec ce qu'on leur donne". Vous savez aussi bien que moi que, dans une administration (mais ce n'est pas différent dans les petites, moyennes et grandes entreprises), les documentations sont généralement introuvables. Il y a toujours un Pierre ou une Brigitte qui l'a provisoirement emportée à la maison afin de l'étudier à tête reposée : ça vous classe quelqu'un, non ?

Et c'est à ce moment-là que l'on apprécie la convivialité des logiciels conçus pour Apple. On s'aperçoit alors que les traitements de texte les plus simples ne sont pas forcément les plus difficiles à mettre en œuvre. Tenez, rien que pour le plaisir, j'en ai testé trois et je vous livre mes réflexions en vrac. Je vous signale au passage que mon compatible (un DONATEC) est équipé d'une très belle imprimante EPSON LQ 2500, malheureusement affligée d'un défaut de jeunesse : trouver des rubans pour en faire réellement une imprimante opérationnelle relève du tour de force.



## ● **PHILOTEXTE JUNIOR** (Editions Dortec)

*Philotexte Junior* est diffusé par le Réseau Planétaire qui l'annonce comme le moins cher des traitements de textes professionnels sur PC (quelques billets de cent francs).

Le manuel d'utilisation (80 pages) est évidemment en français. Il comporte un double index : mots-clefs et touches, ce qui constitue une bonne idée et facilite grandement la recherche. Faut-il pour autant négliger la lecture de cette bonne documentation ? Sûrement pas. C'est le meilleur moyen de savoir où l'on va et surtout comment y aller... et en sortir !

Pour le reste *Philotexte Jr* n'a pas grand-chose à envier aux autres traitements de texte de sa catégorie et il supporte même la comparaison avec des logiciels plus ambitieux. Qu'on en juge :

- Copie et déplacement de blocs de texte à l'intérieur d'un document ou encore de document à document ;
- Recherche et remplacement automatique ;
- Travail simultané sur deux textes avec fenêtrage d'écran ;
- Tous types de justification et de formatage avec utilisation des possibilités typographiques de l'imprimante ;
- En-têtes et notes de bas de page, tables d'index, sommaires...

Ajoutons que neuf écrans d'aide détaillée sont disponibles à tout moment.

Le programme d'impression passe par un fichier "texte-règle" avec lequel il convient de se familiariser. On peut en effet utiliser le même texte-règle pour l'éditeur et pour l'impression, mais il est souvent plus pratique de concevoir des versions spéciales pour l'un et l'autre de ces usages.

*Philotexte Jr* constitue une excellente initiation à *PHILOTEXTE II* (la nouvelle version sera disponible quand paraîtront ces lignes), également distribué par le Réseau Planétaire, mais plus complet (et aussi plus cher, tout en restant très bon marché). Je ne voudrais pas trop vous mettre l'eau à la bouche, mais :

- Un correcteur d'orthographe avec dictionnaire de 100 000 mots ;
- Mailing avec inclusion conditionnelle d'adresses ;
- Gestionnaire de bureau (logiciel intégré en co-résident pour calculs, agenda, notes à la volée...);

Ça vous classe un traitement de texte, pas vrai ?

Réseau Planétaire — Raffy Queyrières — 43260 Saint-Julien-Chapteuil — Tél. 71.08.73.49

## ● **ALLEGRO** (Cedic/Nathan)

Très belle boîte rouge. Vous me voyez venir et vous pensez déjà que je vais comparer le contenu au contenant. Erreur, même s'il est exact que j'ai été surpris de ne découvrir dans ce joli coffret qu'une brochure de 50 pages... évidemment accompagnée d'une disquette.

Autre surprise : des addenda précisent comment effectuer une sauvegarde ou une édition partielle du texte et comment créer un texte entièrement nouveau. On se dit que si le logiciel n'est pas plus sérieux que sa documentation, on va en être pour ses frais (précision utile : *Allegro* est réellement bon marché, mais ce n'est pas une raison suffisante pour jeter son argent par les fenêtres !).

Rassurez-vous : je n'ai rien perdu. D'abord parce que cet *Allegro* m'avait été confié par le Service de Presse de Cedic/Nathan et ensuite parce qu'il en vaut beaucoup d'autres. Inutile de revenir sur l'emballage, sinon pour préciser qu'il est inutile : mieux vaudrait une disquette supplémentaire... ou une chemise protectrice. Le prix constitue un argument de vente suffisant, du moins à mes yeux.

La documentation est certes succincte, mais d'une lecture agréable. On constate rapidement qu'elle est pratiquement aussi inutile que l'emballage : on retrouve la plupart des explications sur la disquette. Sans être particulièrement doué, j'ai réussi à copier le logiciel sur disque dur, à lancer le programme et à imprimer mon premier texte en moins d'une heure. J'ajoute que ce texte était le listage d'une routine en Basic, repris en traitement de texte.

*Allegro*, traitement de texte en français, se révèle à la fois simple et rapide, même si l'on peut contester la manière dont sont utilisées les touches de fonction (au début, c'est assez déroutant, mais on s'y habitue très vite). Il est bon de lire les 50 pages de la documentation pour avoir immédiatement une vue d'ensemble des possibilités offertes par le logiciel, mais il est plus pratique, en cours d'utilisation, de faire appel à la fonction d'aide.

J'ai noté que les fonctions les plus attendues sont bien présentes : pagination, justification, reconnaissance de paragraphes, alignement sur la virgule décimale, fusion de textes, réalisation d'un index, importation de fichiers.

Cedic/Nathan, 6 bd Jourdan — 75014 PARIS — Tél. (1) 45.65.06.06.

# LE TRAITEMENT DE TEXTE



## • NATHALIE 2 PLUS (EBP INFORMATIQUE)

Nombreuses ressemblances avec le PC.Write américain... et pour cause ! Cette version française a été baptisée *NATHALIE* par EBP Informatique, mais le Réseau Planétaire en diffuse une autre sous le nom de *PHILOTEXTE II* (lire page précédente)... avec un dictionnaire de 100 000 mots. Celui de *NATHALIE* n'en compte que 50 000 (mais sans doute 70 000 au moment où vous lisez ces lignes, et avec un certain nombre d'améliorations).

Il s'agit là d'un traitement de texte professionnel, ambitieux (possibilité d'utiliser une imprimante de type LaserJet, mais je ne m'y suis pas risqué), et pas aussi facile à utiliser — dès que l'on sort des commandes classiques — que le souhaiterait sans doute l'usager moyen.

La documentation est en français et je manquerais d'objectivité si je la considérais comme "mince" ou "incomplète". Il lui manque pourtant quelque chose

puisqu'il m'est assez souvent arrivé de me poser des questions, notamment à propos des problèmes de mise en page et d'impression.

Ceci dit, *NATHALIE* est un traitement de texte auto-documenté, comme bon nombre de logiciels concurrents d'ailleurs. L'utilisateur moyen maîtrisera parfaitement la plupart des commandes classiques en quelques heures, et en particulier celles permettant d'obtenir les menus d'aide.

S'il s'agit seulement de taper une lettre, un rapport, pas de problème : le Rédacteur est aussi convivial que beaucoup d'autres. Il tire un bon parti des touches de fonction des IBM et compatibles. Créer un texte, le corriger et le sauver s'accomplit sans douleur. L'impression coule de source.

## Le dictionnaire

*NATHALIE* permet de contrôler, en temps réel, le dernier mot tapé : un bip retentit si le dictionnaire n'en connaît pas l'orthographe... ce qui est assez fréquent. Il est nécessaire de lire attentivement le mode d'emploi du correcteur pour bien savoir où l'on met les pieds. C'est un outil intéressant, un plus incontestable, mais on peut être pour ou contre son utilisation. On ne saurait le considérer comme un juge infaillible.

Personnellement, je recommanderais surtout son utilisation aux personnes ayant à taper de nombreux termes techniques. Naturellement elles devraient d'abord créer leur propre dictionnaire, mais pourraient ensuite l'enrichir au fur et à mesure de la découverte de mots nouveaux.

## L'impression

*NATHALIE* reconnaît un nombre important d'imprimantes (y compris les plus récentes). Un apprentissage s'impose avant d'être en mesure de réaliser des impressions complexes. N'étant pas un spécialiste de la question, je suppose qu'il en va de même pour la plupart des "imprimeurs". Nous nous heurtons aux mêmes difficultés en photocomposition, des difficultés que quelques jours de pratique font oublier !

Si je devais commercialiser un logiciel de traitement de texte, j'insisterais davantage sur les problèmes de formatage et d'impression. Les documentations actuelles, même correctes (ce qui est le cas de celle de *NATHALIE*) sont à mes yeux insuffisantes.

E.B.P., 55 c Rue de Port Royal 78470 ST-REMY-LÈS-CHEVREUSE



# Yvan KOENIG

## répond à nos lecteurs

**Questions :** Pouvez-vous m'aider à résoudre quelques petits problèmes car je sèche lamentablement.

### • PROBLÈME N°1

Il se situe sur la disquette SIGNATURE, face 2 (version ProDOS). La disquette utilise le ProDOS 1.1 et mon Apple utilise la version 1.0. Comment rendre compatible mon Apple et la disquette SIGNATURE ?

### • PROBLÈME N°2

Il concerne le programme "TREIZE" (TREMLIN MICRO N°10). Après quelques changements (ajout de 500 mots, j'ai bien dit 500, et pas un de plus !) il y a des problèmes avec ce mini dico. Après avoir modifié ainsi la ligne 110 :

```
110 HOME:NM=500 au lieu de
```

```
110 HOME:NM=7, le programme se plante sur deux lignes :
```

```
485 V=PEEK(768): IF V=NM OR NOT V THEN  
POKE 768,1: POKE 769,CH : GOTO 500
```

```
495 NEXT: POKE 769 + V, CH: POKE 768,V+1
```

Comment modifier ces deux lignes ? Sinon c'est l'ILLEGAL QUANTITY ERROR qui m'attend.

### • PROBLÈME N°3

Comment écrire en HGR, et en grand format (le même que celui des lettres) un compte à rebours qui partirait de 45 et finirait à 0 ?

M. T. (75020 PARIS)

**R** Ce n'est pas votre APPLE, mais VOUS qui utilisez ProDOS 1.0. Demandez à votre distributeur habituel de vous faire une mise à niveau ProDOS 2.0, BASIC.SYSTEM 1.1 et tout ira bien. Enfin presque ! Il faudra en effet supprimer la ligne qui contrôle ProDOS (30 si je ne m'abuse). Seul le test sur BASIC.SYSTEM est utile. Vous trouverez quelques aménagements à SIGNATURE dans ce numéro de Tremplin, page 37.

Pour faire un POKE, il faut que la valeur à poker soit inférieure à 256. Puisque vous utilisez 500 mots, vous êtes en infraction car la ligne 480 définit la valeur à poker (CH) en fonction de ce nombre de mots. Limite pour NM : 255 maxi, mais comme on stocke les numéros des mots déjà employés en page 3, il

**R**

ne faut pas dépasser \$3D0 sous peine d'un plantage du système. Cela limite en fait NM à 206... Si ces limitations vous chagrinent, réécrivez la routine 480-520 pour traiter des numéros sur 2 octets et placez la table en partie haute de la mémoire.

Pour faire un compte à rebours je vous suggère :

```
1000 for TIME=45 to 0 STEP - 1
```

```
1010 CH$=right$("00"+str$(TIME),2)
```

```
1020 N = asc(left$(CH$,1)) - 64: draw N at  
X,Y
```

```
1030 N = asc(right$(CH$,1)) - 64: draw N at  
X+21,Y
```

```
1040 for I = 1 to 100 : next (à vous d'ajuster)
```

```
1050 next : return
```

Bien entendu, je suppose que vous avez initialisé le mode graphique, chargé la table de formes et défini X et Y pour positionner votre affichage.

**Question :** Je suis dans un club informatique où, pour des raisons pratiques, je laisse ma disquette de Basic (sous DOS 3.3) sur place. J'ai constaté, la semaine dernière, que plusieurs de mes programmes avaient été modifiés à mon insu. Je voudrais donc protéger ma disquette, en Basic de préférence, pour éviter ces surprises. J'ai conçu un programme "Hello" qui demande un code à l'utilisateur, lequel code n'est connu que par moi. Si celui-ci est exact, le catalogue arrive à l'écran, sinon le système est relancé. J'ai protégé le programme contre "contrôle-reset" et "contrôle-C" par :

```
POKE 1010,0: POKE 1011,3: POKE 1012,166: POKE  
768,76: POKE 769,0: POKE 770,3
```

Malheureusement, en chargeant le DOS avec une disquette système, puis en la remplaçant par la mienne, on peut accéder à toutes mes routines. Il y a aussi le POKE 214,128 qui ne me convient guère car il m'interdit également le listing de mes propres programmes.

Je vous lance donc un S.O.S. en vous précisant que je possède un Apple IIc 128 Ko.

**R**

La défunte revue GOLDEN avait publié dans son numéro 16, sous la signature de Ph. GUIOCHON, GOLDCRYPTOR qui répondrait à votre demande sous DOS 3.3. POM'S propose, dans son numéro 29... (suite page 64)



**R** le programme KRUPLOS qui fait le même genre de manipulation sous ProDOS.

**Question :** J'aimerais savoir comment créer un fichier de type SYSTEM et comment transformer un fichier BINAIRE en fichier SYSTEM. J'aimerais le savoir pour pouvoir booter directement un programme de communication de type BIN sans avoir à passer par le BASIC.SYSTEM. Je peux loader un programme SYSTEM en langage machine en entrant la commande suivante :

**BLOAD nom, TSYS, A\$xxx, L\$xxx**

Il me "load" le programme à l'adresse indiquée, et je peux donc modifier un programme SYSTEM, mais je n'arrive pas à la "resaver" en type SYS. J'ai bien essayé la commande :

**BSAVE nom, TSYS, A\$xxx, L\$xxx**

mais il me fait une erreur en me disant : NOT A TYPE SYS FILE. *Anthony R. (93600 AULNAY sous BOIS)*

**R** Votre question appelle une réponse simple. ProDOS — en fait, BASIC.SYSTEM — ne sait créer de manière automatique qu'un petit nombre de types de fichiers. Ce sont les types BAS, BIN et TXT. Ne pleurez pas : les 253 autres types sont tout à fait accessibles. Ils requièrent simplement votre intervention.

Il faut créer le fichier avant de pouvoir le sauver, exactement comme lorsque vous créez un "sous-catalogue". Dans votre cas, il faudra faire :

**CREATE truc.SYSTEM,TSYS** (slot drive éventuels)

**BSAVE truc.SYSTEM,TSYS,A\$2000,L\$xxxx**

Vous aviez en fait la réponse dans SLOADC (*Tremplin Micro* n°11) où je CREAteais un fichier de type \$F1 pour l'écran compacté.

**Questions :** • Est-il possible qu'un fichier "exec" puisse ouvrir un fichier "TEXT" pour le travailler ? Si oui, pouvez-vous me donner un exemple (mes essais n'aboutissent qu'à un "NO DIRECT COMMAND").

- A quoi sert l'instruction "FLUSH" et quelle est son utilité ?
- Pourriez-vous dire un mot sur les fichiers dits "Relogeables" ?
- Et pour terminer, à corriger dans T.M. n°13 :

"Brico" en 9300 remplacer "IF LO <= 0" par "IF LA <= 0" et en 1100 V = (((D/2) ^ 2) \* PI) \* HC ...

*Denis B. (25110 BAUME LES DAMES)*

**R** ProDOS ou, pour être plus exact, BASIC.SYSTEM contient une table où sont stockées des paires d'octets déterminant les règles de fonctionnement de chaque commande.

L'interprétation de ces octets se fait comme suit, bit par bit en partant du bit 8 :

octet 1	PFIX	% 10 00 00 00	préfixe attendu, pathname optionnel
	SLOT	% 01 00 00 00	numslot seul (PRE ou INE)
	RRUN	% 00 10 00 00	commande différée
	FNOPT	% 00 01 00 00	nom de fichier optionnel
	CRFLG	% 00 00 10 00	création de fichier autorisée
	T	% 00 00 01 00	type fichier autorisé
	FN2	% 00 00 00 10	2 <sup>e</sup> titre attendu (rename)
	FN1	% 00 00 00 01	titre attendu

octet 2	AD	% 10 00 00 00	Adresse autorisée
	B	% 01 00 00 00	Byte autorisé
	E	% 00 10 00 00	End autorisé
	L	% 00 01 00 00	Longueur autorisée
	LINE	% 00 00 10 00	(à) numligne autorisé
	SD	% 00 00 01 00	Slot / Drive autorisés
	F	% 00 00 00 10	Field autorisé
	R	% 00 00 00 01	Record autorisé
	V	% 00 00 00 00	V toujours ; autorisé est ignoré

Pour OPEN et APPEND le 1<sup>er</sup> octet est \$2D (%00101101) tandis que pour READ et WRITE il est de \$21 (%00100001). Je vous laisse faire l'analyse complète mais vous pouvez constater que le bit 6 est à 1 dans tous les cas ce qui interdit l'emploi de ces commandes en mode direct. Faire \$B93F et \$B955:0D, \$B941 et \$B953:01 supprimerait cette interdiction mais ce serait À VOS RISQUES ET PÉRILS.

Je vous rappelle que lorsque vous êtes dans un fichier EXEC, tout se passe comme si vous étiez en mode direct. Attention, la routine INPUT n'est pas évidente à employer en mode direct (cf. *Tremplin Micro* n°9).

FLUSH provoque le transfert sur disque de toutes les zones tampon allouées aux fichiers préalablement ouverts. Si l'option "titre" est utilisée, seule la zone tampon du fichier spécifié est transférée. Tout se passe comme pour la commande CLOSE à ceci près que les fichiers restent ouverts et que les pointeurs de fichiers ne sont pas altérés.

**R** La commande FLUSH est très utile afin de préserver l'intégrité des données sur disque. Un programme qui risque d'être interrompu accidentellement doit régulièrement utiliser la commande FLUSH de façon à perdre le moins de données possible en cas d'interruption.

Cette définition est reprise du fascicule Addendum aux TRAVAUX PRATIQUES APPLE-SOFT la programmation en Basic avec ProDOS qui est livrée avec le disque OFFICIEL ProDOS.

Je suppose que votre 3<sup>e</sup> question porte sur les fichiers de type REL. Ce sont des fichiers qui comportent non seulement le code actif, mais aussi un ensemble d'informations qui permettra de les employer sans contrainte d'adresse, ils seront ajustés par le "loader" en fonction de leur implantation effective. Ils sont générés par l'assembleur EDASM d'APPLE. MERLIN.PRO génère pour sa part des fichiers RELogeables de structure et d'utilisation différente. MINIE et le BUREAU de MINIE sont des assemblages de fichiers de type REL MERLIN.

Merci pour vos remarques sur BRICO.

**Question :** Je suis possesseur d'un IIe version 65C02 et d'une imprimante IMAGEWRITER II avec interface super-sérial. Je voudrais utiliser les programmes parus dans T.M n°6 permettant de charger des polices de caractères et pouvoir m'en servir avec des traitements de texte genre APPLEWORKS. En utilisant CARAC.CHARG je parviens à les charger et à les imprimer, mais malheureusement je ne trouve pas la "combine" pour utiliser les trois qualités d'impression de l'IIII. Je ne peux imprimer qu'en qualité brouillon. *Jean-Louis D. (06604 ANTIBES cedex)*

**R** Une fois encore, voici un lecteur confronté à un mauvais fonctionnement du circuit de distribution APPLE.

Les manuels de références techniques ne sont pas publiés par APPLE (TVA 18,60%) mais par un éditeur, INTEREDITIONS 18 av. du Maine 75014 PARIS (TVA 7%).

Cette formule judicieuse s'accompagne hélas d'effets pervers. Ces manuels sont pratiquement absents des rayons des "distributeurs agréés" (problèmes de marge ?). De plus, j'ai pu constater à plusieurs reprises que lorsqu'un utilisateur a des problèmes, on ne

**R** lui signale même pas l'existence de ces documents. Lorsque le numéro 15 de Tremplin Micro sera paru, il est probable que la version française du "TECHNICAL REFERENCE MANUAL IMWII" sera disponible.

La réponse à votre question est simple, lorsqu'on lui demande d'imprimer des caractères personnalisés, l'IIWII passe automatiquement dans le mode STANDARD (le mode intermédiaire, voyant de droite allumé). Elle se remet dans le mode où elle était au début dès que vous quittez les caractères personnalisés.

Si vous souhaitez vous rapprocher de la densité d'impression du mode quasi-courrier, passez en mode gras chaque fois que vous appelez les personnalisés. J'envoie pour ma part systématiquement la séquence ESC ' ESC ! pour appeler, et ESC " ESC \$ pour quitter les caractères personnalisés.

**Question :** Je vous écris pour un petit renseignement, le troisième depuis votre naissance !

Si je vous dis : LDA £\$30  
LDY £\$10  
JSR \$3D9

Vous me répondez routine RWTS avec l'adresse de I.O.B. D'accord ?

Faisons un tour à l'adresse :

310 : 01 60 01 00 11 : je connais la signification. MAIS...

00 91 03 00 95 00 00  
01 00 00 60 01 00  
01 EF D8 00 ] ? AïE

Cette routine est très utilisée et je m'étonne que vous n'avez pas encore donné la signification de la table des paramètres. Je suis sûr que les lecteurs seraient contents d'en saisir le sens.

T.M. n°1 page 5 (nombre de secteurs libres) un petit bug... supprimer de \$371 à \$375 JSR \$ED24 et ça fonctionne. *Didier M. (72100 LE MANS)*

**R** Non : il n'y a pas de bug dans la routine que vous signalez. Son auteur a choisi d'imprimer un "Carriage Return" (retour chariot \$0D) après le nombre de secteurs. Vous préférez ne pas en mettre, c'est votre droit mais le choix de l'auteur était tout à fait respectable. JSR \$ED24 exécute la routine ASOFT implantée à cette adresse... *(suite page 66)*



**R** à savoir LINPRT qui imprime en décimal l'entier dont l'octet bas est dans (X) et l'octet haut dans (A).

En ce qui concerne la RWTS, je ne suis pas sûr que *Tremplin Micro* n'ait pas déjà publié les infos demandées. Enfin, allons-y ! Désassemblons une table RWTS, celle contenue dans DOS 3.3 par exemple :

org \$B7E8

TBLTYPE	hex 01	IOB type
SNUM16	dfb 6*16	num slot * 16
DNUM	hex 01	num lecteur
VOLEXPT	hex FF	num volume attendu
TNUM	hex 0C	num piste
SNUM	hex 08	num secteur
	da DCTTBL	adresse de la table DCT
USRBUF	da \$9600	adresse du buffer lecture
BYTCNT	dw \$100	nombre d'octets par secteur
CMDCODE	hex 02	code de la commande
		00 positionne la tête 01 lecture
		02 écriture 04 format
ERRCODE	hex 07	code d'erreur
VOLFND	hex FE	n° de volume du dernier accès
SLOTFND	dfb 6*16	n° de slot .....
DRVFND	hex 01	n° de lecteur .....
	hex 00,00	apparemment inutilisés
DCTTBL	hex 00	code du périphérique (0 pour lecteur 5")
	hex 01	phase par piste (1 obligatoire)
	hex EF,D8	compteurs pour mise en route du moteur (valeurs intouchables)

**Question :** Ayant eu à ma disposition un APPLE IIe pendant un bon bout de temps, je m'étais acheté plusieurs logiciels. Maintenant, je possède un IIc et plusieurs de ces logiciels refusent de démarrer normalement. Il paraîtrait que ceci est dû à un octet qui est testé à chaque démarrage. On m'a parlé du \$C600. Une solution serait possible en déplaçant le programme BOOT qui est implanté dans cette partie de la mémoire...

Malgré pas mal de tentatives, je n'arrive absolument pas à faire tourner ces logiciels. Pourriez-vous m'aider ? **Gérard C. (78960 VOISINS le BRETONNEUX)**

**R** Apparemment votre IIc ignore aussi où prendre le timbre demandé pour la réponse.

Sauf si vous avez acheté un appareil défectueux, je ne pense pas que vos problèmes viennent de la ROM de boot.

Si vous réussissez à déplacer un programme inscrit dans les ROMs de la machine, faites-moi signe : vous aurez gagné un rayon de soleil !...

Si vous aviez pris la précaution de me fournir les titres des programmes capricieux, j'aurais peut-être pu vous renseigner, mais pour l'instant : impossible !

Etes-vous sûr qu'il ne s'agit pas de logiciels tournant sous CPM et qui pouvaient donc fonctionner sur un IIe équipé d'une carte Z80 ?

Je connais quelques logiciels qui plantent sur le IIc, mais ça n'a rien à voir avec la ROM de boot. C'est en général parce qu'ils sont protégés en utilisant une "singularité" du 6502 qui n'existe pas sur le 65C02 équipant le IIc et le IIe nouveau. Pour savoir à quoi vous en tenir, essayez les programmes en question sur un IIe nouvelles roms. S'ils plantent c'est probablement à cause de cela (à moins bien entendu que ce soient des programmes CPM).

**NDLR :** Gérard C. est peut-être — tout simplement — un petit malin qui cherche à utiliser des logiciels piratés...

## DONATEC : 1986, une année "DÉMONIAQUE"

Pour sa troisième année d'existence, DONATEC, constructeur français de micro-ordinateurs compatibles et de matériels périphériques, confirme sa progression en réalisant en 1986 un chiffre d'affaires de 38 millions de Francs, soit 70 % d'augmentation par rapport à l'année précédente, dont le C.A. était de 22,5 MF. En 1986, DONATEC a vendu 4 000 micro-ordinateurs.

Avec de grands changements et un chiffre d'affaires en croissance constante, la société DONATEC évolue très vite et s'impose désormais sur le marché de la micro-informatique. En 1986, DONATEC a changé de locaux, ouvert un point de vente, augmenté son capital et ses effectifs, et renforcé son équipe technique. Créé avec quatre personnes, en février 1985, DONATEC en emploie actuellement trente. Son capital de départ de 250.000 F atteint aujourd'hui 2.250.000 F. Son point de vente ouvert depuis le 1<sup>er</sup> juillet 1986, lui sert également de show-room et lui permet de conseiller et d'orienter les utilisateurs dans leurs différents choix.

Après s'être lancé dans la distribution de produits hauts de gamme pour la Publication Assistée par Ordinateur, tels le moniteur MDS Genius, l'imprimante Kyocera F-1010, les logiciels Personnal Publisher et Ventura, et le scanner Microtek, DONATEC poursuit ses efforts de couverture des marchés.

La société DONATEC, présente sur le marché OEM, chez les revendeurs et également auprès des grands comptes (Citroën, Technip, Volvo, BMW, Creusot Loire, Ministère de la Défense, Ministère de la Coopération, CNRS, Compagnie Générale des Eaux,...) par le biais de son point de vente, affirme une fois de plus sa volonté de performance avec un souci permanent de précéder sinon d'accompagner les progrès de la micro-informatique. Diable !...



**Question :** Comment réaliser des copies d'un écran qui contient des caractères affichés en inverse vidéo... en utilisant la routine qui a paru dans le cadre du programme *Brico de T.M. n°13* ?

M. R. (38100 GRENOBLE)

**R** Le remède me semble simple\*. Il faut contrôler le caractère lu sur l'écran avant de l'incorporer à la chaîne générée par la routine. Faites BLOAD IMP.LM,A\$300, puis passez en moniteur par CALL - 151. Tapez :

```
316 : 20 49 03
320 : 20 49 03
349 : 09 80      cad
                   ORA £$80 pour forcer ASCII négatif
34B : C9 A0      CMP £" "
34D : B0 02      BCS $351 ce n'est pas un contrôle
34F : 69 40      ADC £$40 contrôle majuscule normale, vous pourriez mettre A9 A0 qui remplacerait le caractère gênant par un espace.

351 : 99 00 60   STA $6000,Y
354 : 60         RTS
```

Retour au Basic par Ctrl C puis, BSAVE NEWIMP,A\$300,L\$55

\* Le programme IMP80.IW2 (NOUVELLES ROUTINES POUR LE 65C02) règle en fait ce problème. Il imprime en gras ce qui est en inverse sur l'écran et tient également compte des caractères souris.

**Questions :** La routine de tri en langage machine TRI.1.2 de Tremplin Micro n°11 (page 48) se plante magistralement avec EXTRA.VARIABLES de la disquette EXTRA.PRODOS (programme permettant d'accroître la mémoire utilisateur à 60 K environ). Pourrait-on la faire tourner correctement dans cette configuration et dans ce cas quelles modifications faudrait-il lui apporter ?

La commande PRODOS STORE sauve sur disque toutes les variables du programme en cours. Par quelques pokes serait-il possible de sauver seulement une ou deux variables (dimensionnées) de son choix ?

Jean-Paul T. (93300 AUBERVILLIERS)

**R** Il est tout à fait normal que TRI.1 ne fonctionne pas avec EXTRA.K qui stocke les variables en MemAUX. La routine TRI.1 est faite pour travailler avec un stockage normal en MemPrincipale. PROTECT.RAM et AUX.DRAW dans Tremplin

**R** Micro n°12 et n°13 vous donnent deux formules pour sauver et récupérer des infos en MemAux. En vous inspirant de ces exemples il vous serait peut-être possible de modifier TRI.1 en liaison avec la méthode de stockage employée par EXTRA.K, méthode que je ne connais pas.

Il n'est pas possible de réaliser un transfert sélectif de variables par quelques POKes. Vous pourriez trouver dans NIBBLE EXPRESS vol2 un programme de CORNELIS BONGERS qui, sous le titre AMPER STORE and RECALL opérait la sauvegarde et la récupération d'un tableau numérique (programme repris dans HAIFA publié par POM'S). Ecrit sous DOS 3.3, ce programme devrait être remanié pour travailler sous ProDOS. Il faudrait le placer entre ProDOS et ses buffers et changer la méthode d'envoi des commandes BSAVE et BLOAD.

**Question :** J'ai un problème que l'article de M. Cotini dans le n°12 de T.M. n'a pas résolu. Il s'agit des transferts entre mémoire principale et mémoire auxiliaire avec carte 80 colonnes.

J'ai écrit une petite routine en Langage Machine pour faire des fenêtres dans l'écran 80 colonnes de mon IIc ou de mon II GS et je voudrais transférer le contenu en mémoire auxiliaire, par exemple à l'adresse \$4000. Cela marche très bien pour écrire en Mem. Aux. en actionnant \$C005, en revanche impossible de le faire dans l'autre sens, c'est-à-dire de LIRE en Mem. Aux. par STA \$C003 (RAMRD), qui est le corollaire de RAMWRT. Pouvez-vous me dépanner ?

D'autre part, je vous précise que la routine publiée dans un des premiers numéros de T.M. pour transférer un écran 80 colonnes en mémoire auxiliaire et le rappeler ne fonctionne pas sur le GS (il doit s'agir, je pense, des Trous d'écrans qui n'aiment sans doute pas être "ballottés"). Enfin, la plupart des routines publiées en Langage Machine sont données à l'adresse \$300, lorsqu'elles sont courtes, il existe également une autre place en ProDOS, à l'adresse \$BB4A qui n'est touchée ni par le chargement d'autres programmes ni par l'ouverture de fichiers. J'y mets personnellement une routine de gestion de la souris qui s'y trouve très bien et qui n'affecte pas la marche du système.

Pierre G. (42170 ST-JUST-ST-RAMBERT)

**R** Nous sommes tous des auteurs extérieurs, et c'est ce qui garantit l'indépendance... (suite page 68)

**R** de Tremplin Micro. PROTECT.RAM et AUX.DRAW dans Tremplin Micro n°12 et n°13 vous donnent deux formules pour sauver et récupérer des infos en Mem. Aux. En vous inspirant de ces exemples, il vous serait peut-être possible de faire fonctionner votre programme. Je rappelle simplement que, si vous activez le commutateur AUXREAD, vous basculez en mode lecture en AUX la mémoire \$200 à \$FFFF. C'est pourquoi, dans PROTECT.RAM, je place une routine de "lecture AUX" dans la page zéro qui elle n'est pas basculée par AUXREAD. Dans AUX.DRAW j'emploie systématiquement la routine MOVAUX mise à notre disposition par APPLE.

Pour un transfert d'écrans 80 colonnes sans problème, je vous conseille d'étudier de près SLOADC dans Tremplin Micro n°11. Il est peu prudent de s'implanter dans le trou de BASIC-SYSTEM (302 octets en BB4C) pour des simples routines car :

- 1° Ce trou n'est pas garanti. Il est d'ailleurs moins grand dans BASIC.SYSTEM 1.0.
- 2° NIBBLE et Call APPLE y ont implanté des patches corrigeant des "bugs" de BASIC-SYSTEM (que vous êtes nombreux à appeler à tort PRODOS qui lui, se charge au-dessus de \$D000).

Je regrette d'ailleurs qu'APPLE n'ait pas profité de l'introduction d'une nouvelle version de PRODOS (PRODOS v2.0 ou P8 v1.2 et maintenant 1.3) pour corriger également BASIC-SYSTEM qui n'a ABSOLUMENT pas changé dans la tourmente.

mode proportionnel, de rétablir cet accent à la place qu'il mérite et qu'il n'aurait jamais dû quitter ? Ma question va peut-être vous sembler naïve ! Vous seriez bien aimable si vous pouviez me répondre où me faire connaître les références des documents que je pourrais consulter sur ce sujet.

Jean M. (75007 PARIS)

**R** Rassurez-vous (ou inquiétez-vous), vos programmes ne sont pas en cause. Le problème a sa source dans l'imprimante IWill. APPLE France a été mis au courant depuis plusieurs mois. Je leur ai même balisé la route pour l'analyse du bug. Le "recul" est court sur l'IWill, ce qui permet de se repositionner correctement sur un i. Le manuel de référence technique, qui doit sortir sous peu en français, affirme, dans sa version anglaise, que l'on peut envoyer plusieurs "reculs" en séquence, ce qui aurait dû permettre de se repositionner tout aussi correctement sur un u, un e ou un a. Hélas ! l'imprimante n'est apparemment pas conforme au manuel.

Irrité par l'apparent immobilisme d'APPLE France j'ai écrit directement à J.-L. GASSÉE en espérant que sa situation de Français de l'étranger le pousserait à faire en sorte que les imprimantes de "sa" marque traitent correctement SA langue. C'était le 8 février et je n'ai pas reçu le moindre soupçon de réponse.

Je pense que seule la multiplication des réclamations a des chances de faire bouger les choses.

Yvan KOENIG

**Question :** Tout à fait débutant en programmation, je suis surtout un utilisateur de traitement de texte et de gestionnaire de fichiers. Je travaille sur APPLE IIc et imprimante IMAGEWRITER 2, système dont je suis parfaitement content, sauf en ce qui concerne l'impression en mode proportionnel. Ce mode est, pourtant, celui qui me séduirait le plus..., s'il ne présentait pas un défaut réhébitorique : en mode proportionnel, il est absolument impossible de placer l'accent circonflexe (même chose pour le tréma), exactement au-dessus de la lettre qu'il est censé "coiffer". Il est toujours situé à droite, ce qui fait le plus mauvais effet... Que ce soit avec les traitements de texte "Epistole" ou "Appleworks" ou même avec les petits éditeurs de texte, imaginés par moi, les résultats restent toujours les mêmes. Y-a-t-il une astuce quelconque qui permettrait, en

Indispensables :

**LES NOUVELLES  
ROUTINES  
POUR 65C02  
(et 6502)**

Bulletin de commande page 75



# METTEZ UN GS

## — dans votre Apple // —

*J'ai essayé, sans trop y croire, je l'avoue, la carte accélérateur **TRANSWARP**. D'où des surprises d'autant plus agréables ! Ce produit d'**Applied Engineering**, bien connu aux Etats-Unis, est importé par la société lyonnaise **BREJOUX** (qui diffuse aussi **GS RAM** et **GS RAM PLUS** — voir l'article de la page 39).*

*Que nous promet-il ? Tout simplement d'accroître la vitesse de notre vieil Apple II ou ILe jusqu'à 3,6 MGz ! Vous avez bien lu : on passe de 1 à plus de 3 MGz et c'est évidemment spectaculaire. Il suffit de lister un programme pour en être immédiatement convaincu !*

### QUE SE PASSE-T-IL ?

TRANSWARP met le 6502 ou le 65C02 de votre Apple en sommeil et travaille alors avec sa propre mémoire et son microprocesseur personnel... qui est aussi un 65C02, mais cadencé à 3,6 MGz.

TRANSWARP accélère non seulement la mémoire vive mais aussi la mémoire auxiliaire (256 K de RAM ultra rapide). Mais alors, comment vont se comporter certains périphériques et logiciels, prévus pour des vitesses beaucoup plus "raisonnables" ? Apprenez d'abord que cette carte est entièrement transparente avec tous les logiciels (aucune disquette de mise en route) et compatible avec toutes les cartes d'interface ou d'extension Apple.

Sachez aussi que trois vitesses sont possibles : 1 et 3,6 MGz par contrôle au clavier et 1,7 MGz par contrôle sur la carte. Une extension 16 bits est même disponible sur demande, mais je ne l'ai pas testée.

### UTILISATION

TRANSWARP peut être installé dans n'importe quel slot disponible. Une notice en français, succincte, mais TRÈS CLAIRE, indique exactement la marche à suivre pour configurer les interrupteurs de la carte... selon la nature des périphériques en service. Ensuite, plus rien à faire ! A la mise en route, TRANSWARP est activé automatiquement : un bip

retentit et le logo "TRANSWARP" apparaît sur l'écran (il suffit de presser la touche ESCAPE pendant l'affichage du logo pour désactiver complètement la carte et retrouver le 6502 ou le 65C02 "escargot" de l'Apple II classique). Mais il est également possible, avec un seul POKE, de faire varier la vitesse du processeur à partir d'un programme personnel. Les accès à la RAM et à la ROM de l'Apple sont limités, on le sait, à 1 MGz, mais TRANSWARP charge la ROM de l'Apple dans sa propre RAM et utilise le reste de cette RAM pour émuler à la fois la mémoire principale et la mémoire auxiliaire. Par rapport à un Apple ordinaire, c'est le jour et la nuit ! On se demande même pourquoi Apple n'a pas offert cette accélération à tous ses clients. Peut-être à cause des cartes Z-80 qui, même avec TRANSWARP activé, fonctionnent à leur vitesse normale ?

Nous avons tous regretté, lors de l'apparition du 65C02, que celui-ci soit plus ou moins bridé. Cette accélérateur — un produit que je vous recommande — prouve que nous avons raison. Gageons qu'une carte identique, mais avec un 65C816 débridé, autorisera bientôt — sur l'Apple IIGS — des cadences encore plus surprenantes !

Documentation et renseignements : Société Bréjoux — J.-M. BRESARD  
29, rue de Montriblond 69009 LYON — Tél. 78.36.52.69.



# Vient de paraître

Les Editeurs nous  
communiquent :

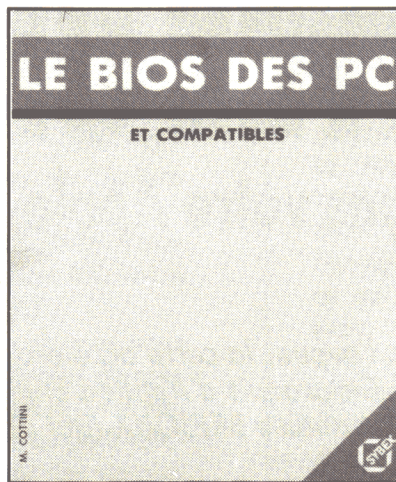
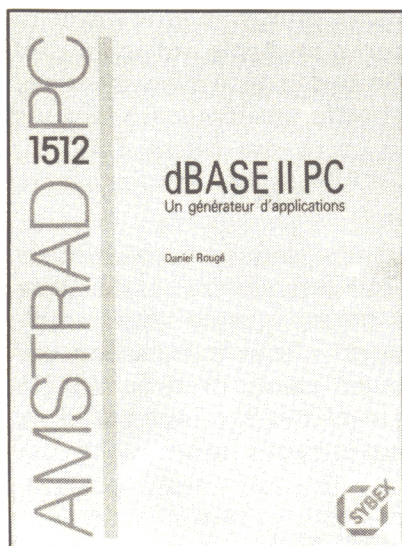
- **AMSTRAD PC 1512  
dBASE II PC  
un générateur d'applications  
(Daniel Rougé)**

*dBASE II* est un logiciel puissant, mais difficile à aborder pour un utilisateur non programmeur.

Cet ouvrage développe la construction d'un *générateur d'applications* permettant de créer et d'utiliser des applications simples ou complexes de *dBASE II*. Il s'adresse à tous, du débutant au programmeur expérimenté qui pourra adapter les programmes fournis à ses propres besoins. Après une introduction aux bases de données, sont développés la conception et le mode d'emploi du générateur d'applications. La dernière partie contient les programmes, accompagnés de toutes les explications (structures, commentaires, variables) nécessaires à leur compréhension.

Destiné aux utilisateurs de *dBASE II PC* sur *Amstrad PC 1512* et compatibles, il n'oublie pas les possesseurs d'*Amstrad PCW 8256*, *PCW 8512* et *CPC 6128*.

Livre broché de 416 pages (168 F).  
SYBEX, 6-8 impasse du Curé 75018 PARIS



- **LE BIOS DES PC  
ET COMPATIBLES  
(M. Cottini)**

*Le BIOS de l'IBM PC* est un ouvrage destiné à tout programmeur sur PC, qu'il soit professionnel, débutant ou simple "hobbyiste". Le but recherché est une mise à niveau du contenu du ROS (*Résident Operating System*, système d'exploitation résident) implanté d'origine dans la "machine" et des fonctions essentielles du BIOS (*Block Input Output System*, modules d'entrées/sorties du système) dans son environnement.

Bon nombre d'ouvrages spécialisés traitent de l'IBM PC, mais très rares sont ceux destinés aux systèmes d'exploitation en général (BIOS + MS-DOS) ; dans ce cas les éléments essentiels sont tenus généralement sous silence. En rédigeant *Le BIOS de l'IBM PC*, l'auteur a voulu réhabiliter l'IBM PC très souvent copié au détriment de l'utilisateur. C'est ainsi que les nombreuses recherches et vérifications entreprises au niveau du PC se sont concrétisées par la naissance d'un ouvrage technique de haut niveau.

Six chapitres sont consacrés à l'IBM PC : de la mise au point aux notions fondamentales nécessaires au débutant, en passant par l'étude des princi-

paux composants électroniques et de la structure d'une disquette souple 5 pouces 1/4, sans oublier les interruptions du BIOS (*INT Interrupt*) et du DOS, qu'elles soient externes ou internes, de type NMI, INTR ou RESET. Le chapitre 5 nous dévoile le ROS (BIOS + table des paramètres + routines auto-tests) regroupant tous les modules du BIOS. Les mécanismes des différentes routines et sous-programmes sont décrits clairement, le BIOS tenant une place prépondérante dans le système. Des annexes mettent en valeur certains petits trucs, "must" destiné à parfaire votre savoir-faire.

Livre broché de 414 pages (198 F).  
SYBEX, 6-8 impasse du Curé 75018 PARIS

- **GUIDE PRATIQUE DE FICHES  
ET DOSSIERS  
(Michel Garrard)**

Bien des gens sont confrontés aujourd'hui à des problèmes de classement et de tri de fiches. Un gestionnaire de fiches comme *Fiches et Dossiers* vient soulager l'emploi du temps et le plan de travail. La disquette remplace ici le rustique "bac à fiches" et les traditionnelles opérations de classement, de sélection, d'ajout ou de suppression, de modification s'opèrent très facilement à l'aide du clavier et du crayon optique (ou de la souris).

Cet ouvrage se veut un guide pratique abondamment illustré. On y trouve de nombreux exemples ainsi que des modèles pratiques qui pourront être utilisés tels quels ou modifiés au gré de vos besoins. La seule limite est celle de l'imagination. Les exemples ici présentés concernent à la fois le domaine familial et le domaine professionnel, ce qui montre le vaste champ des illustrations possibles de ce logiciel.

Dans la première partie, l'auteur décrit la création, la gestion, l'impression du fichier ainsi que le passage de *Fiches et Dossiers* à *Paragraphe*. On trouve ensuite dans la deuxième partie de nombreux exemples tels que le budget familial, les résultats scolaires, le carnet d'adresses, la gestion d'un portefeuille d'actions, etc...

codic/nathan, 6-10 boulevard Jourdan 75014 PARIS



# NESTOR

## SAGESSE OBLIGE

### IL Y A UN BUG

• Je suis en train d'essayer d'écrire une petite routine en langage machine et, pour économiser des octets, je fais le plus souvent possible appel à des routines du moniteur... et notamment à \$F94C :

F94C :	20 ED ED	JSR \$FDED
F94F :	CA	DEX
F950 :	D0 F8	BNE \$F94A
F952 :	60	RTS

Normalement, si j'ai :

300 :	A9 AA	LDA £\$AA
302 :	A2 FF	LDX £\$FF
304 :	20 4C F9	JSR \$F94C

Je dois obtenir l'affichage de 255 astérisques. Or, bizarrement, je n'obtiens qu'un seul astérisque suivi de plusieurs lignes d'espaces... et j'y perds le peu d'assembleur que je connais.

**NESTOR** : Je ne suis pas un génie, mais à votre place, je regarderais tout de même ce qui se trouve en \$F94A, adresse à laquelle vous renvoie le BNE de \$F950. Vous constaterez qu'il s'agit d'un très bel :

```
$F94A A9 A0 LDA £$A0
```

Or, \$A0, jusqu'à nouvel avis, c'est un espace. Au premier tour de la boucle, vous affichez effectivement l'astérisque alors contenu dans l'Accumulateur, mais dès le second tour, la routine affiche X espaces... et c'est bien pour cela qu'elle est là, je vous le garantis.

Pour afficher x fois l'\* , contentez-vous de :

300 :	A2 FF	LDX £\$FF (ou autre chose)
302 :	A9 AA	LDA £\$AA
304 :	20 ED FD	JSR \$FDED
307 :	CA	DEX
308 :	D0 FA	BNE \$304
30A :	60	RTS

Ça ne vous mangera que 3 octets supplémentaires et ça marchera, c'est promis !

### FAUT-IL COMMENCER PAR L'ABC ?

• Vous avez l'air de vous jouer des difficultés et je rêve de pouvoir, moi aussi, pratiquer le langage machine et accélérer ainsi les programmes en Basic dont la lenteur me désespère. Dois-je absolument commencer par l'ABC ? J'avoue me contenter, depuis quelques mois, de recopier les routines sans les comprendre. Puis-je acheter vos bouquins en confiance ?

**NESTOR** : Etant donné mon grand âge, j'ai dû passer de longues soirées sur les programmes des autres avant de réussir à y comprendre quelque chose, mais je m'y suis bien amusé et c'est l'essentiel. De plus, il m'arrive d'étonner mes enfants et petits-enfants, ce qui n'est pas fait pour me déplaire ! Faut-il commencer par l'ABC ? Personnellement, je n'ai pas su me plier à cette discipline, mais on dit que c'est la meilleure démarche.

**Le 6502 PAS À PAS** (c'est l'œuvre de deux lecteurs de *Tremplin Micro*) m'a beaucoup plu parce qu'il permet réellement de s'initier en concevant de petites routines performantes. Bien sûr, on fait mieux plus tard, mais pour un début, c'est extra.

Quant à nos bouquins de routines, je vous les recommande tous pour trois raisons :

- Ils sont accompagnés de leur disquette.
- Ils comportent souvent plusieurs développements sur une seule idée.
- Tout y est expliqué.

Comme vous le savez, ils reprennent des routines parues dans la revue, mais comprennent aussi des inédits. ■

# Questions

• Comment incorporer — en Basic — un mot dans une liste alphabétique ?

**RÉPONSE :** Il faut d'abord vérifier que la variable concernée a été suffisamment dimensionnée... et revoir DIM le cas échéant. Vous pouvez vous inspirer de la petite DÉMO que voici :

## INC

```
10 TEXT :D$ = CHR$(4): PRINT D
   $"PR£3": PRINT : HOME      65C4
15 DIM A$(40)                  DDA0
20 HTAB 17: INVERSE : PRINT "INC
   ORPORER UN MOT DANS UNE LISTE
   ALPHABETIQUE": NORMAL : VTAB
   5                            8750
25 FOR I = 1 TO 26:A$(I) = CHR$(
   (I + 64) + "-MOT-" + STR$(I
   ): PRINT A$(I),: NEXT      9327
30 VTAB 21: PRINT : INPUT "MOT -
   > ";M$: IF M$ = "" THEN 60  C3D8
35 N = I                       2E67
40 FOR I = N TO 1 STEP - 1: IF
   A$(I - 1) < = M$ THEN A$(I)
   = M$: GOTO 50              3FDB
45 A$(I) = A$(I - 1): NEXT    3F84
50 VTAB 5: CALL - 958         B70C
55 FOR I = 1 TO N: PRINT A$(I),:
   NEXT : GOTO 30            71FD
60 VTAB 21: PRINT : CALL - 868:
   PRINT "(M)ENU DE DISQUETTE (A
   )PPLESOFT ->_c_tG ";: GET R$: 9D21
65 IF R$ = "M" OR R$ = "m" THEN
   PRINT D$"RUN MENU"        4E9D
70 IF R$ < > "A" AND R$ < > "a
   " THEN 60                 3FDA
75 HOME                       2F97
```

• Je travaille toute la journée sur un IBM PC... et je m'amuse le soir sur un Apple IIc. Je suppose que bon nombre de vos lecteurs sont dans le même cas. Ne serait-il pas possible de transformer les commandes CAT de ProDOS et CATALOG du DO 3.3 en DIR, mais sans modifier réellement les noms de ces commandes,

afin qu'elles restent compatibles avec les divers logiciels tournant déjà sur la machine ?

**RÉPONSE :** Si je comprends bien, vous désirez seulement créer un programme EXEC vous permettant de lire le catalogue par un simple — DIR (sous ProDOS) ? Voici un exemple qui, par-dessus le marché, vous renvoie ensuite au menu de la disquette.

## EXDIR

```
10 TEXT :D$ = CHR$(4): PRINT D
   $"PR£3": PRINT : HOME      65C4
15 PRINT D$"OPEN DIR"        6A97
20 PRINT D$"WRITE DIR"      E0F0
25 PRINT "PREFIX,S6"        9781
30 PRINT "CAT"              1FD6
35 PRINT "CALL-198:POKE49168,0:W
   AIT49152,128:POKE49168,0"  1EC6
40 PRINT "RUN MENU"         B148
45 PRINT D$"CLOSE"         DBDC
```

Sous DOS 3.3, il faudrait remplacer CAT par CATALOG et taper EXEC DIR... et je ne vois pas où serait l'avantage !

## MEA CULPA (n°14)

**Page 9 :** CHKCOM, c'est bien \$DEBE, mais il faut inverser l'adresse quand on l'utilise en langage machine (BE DE). A corriger deux fois...

**Page 12 :** En \$32D, il faut lire 09 et non \$7... qui fait boucler la routine VIDEDEC indéfiniment.

**Page 53 :** A la ligne 455, on a collé un rectificatif qui a masqué le 1 qui se trouvait à la suite du +.

### ... et aussi Yvan KOENIG

#### TREMLIN MICRO N°13 :

**Page 72 :** Il aurait fallu dire :  
POKE 49235,0 STA \$C053 Graphique mixte.

#### TREMLIN MICRO N°14 :

**Page 41 (Double basse résolution) :** J'ai parlé d'un bug dans le IIc, mais c'est moi qui suis buggué ! En \$F1FD, on trouve, dans IIc un CPX £\$50 qui permet d'accéder à tout l'écran. C'est dans le IIGS que l'on trouve CPX £\$30 et c'est pour cela que les concepteurs d'APPLE ont dupliqué la routine.

**Page 72 (colonne de droite) :** J'ai écrit "En \$60AF essayez de remplacer..." et j'ai donné les modifs en partant de \$60AD ce qui n'est pas très futé.



Fonctionne aussi sur l'APPLE IIGS

# Quoi est où ?

## QUESTION

Comment savoir quelle lettre est à un endroit donné de l'écran texte ?

Si vous connaissez l'adresse exacte, aucun problème : **L = PEEK (ADR)** vous fournira le code ASCII du caractère affiché. Si vous ne connaissez que la position (HTAB - VTAB), utilisez plutôt la petite routine que voici (valable sur 40 colonnes exclusivement). Elle place le code ASCII à l'adresse 6 ou vous le lirez donc par un **L = PEEK (6)**.

**SYNTAXE :** CALL 768, HTAB, VTAB (où HTAB et VTAB peuvent évidemment être des variables numériques). Ne pas utiliser 40,24 (ligne 75) qui provoquerait un scrolling et fausserait le résultat.

```

10 FOR I = 768 TO 791: READ R: POKE I,R: NEXT
20 DATA 32,76,231,202,218,32,76,231,202,138,32,71,248,250,138,101,38,133,38,177,38,
133,6,96
30 TEXT : NORMAL : PRINT CHR$ (21): HOME
40 PRINT "Q U O I E S T O U ?"
50 PRINT " _____ ": PRINT
60 PRINT "HTAB3:VTAB1, IL Y A: ": CALL 768,3,1: PRINT CHR$ ( PEEK (6))
70 PRINT : PRINT "HTAB5:VTAB4, IL Y A: ": CALL 768,5,4 : PRINT CHR$ ( PEEK (6))
75 FOR I = 1 TO 10: VTAB 8: HTAB 1: INPUT "H, V ";H,V: CALL - 958: HTAB H: VTAB
V: INVERSE : PRINT "C": VTAB 9: NORMAL : PRINT : CALL 768,H,V: PRINT PEEK (6):
NEXT
80 VTAB 22: PRINT "(M)ENU DISQUETTE ": GET R$: IF R$ = "M" THEN PRINT CHR$
(4)"RUN MENU"

```

```

300 : 20 4C E7 JSR $E74C
303 : CA DEX
304 : DA PHX
305 : 20 4C E7 JSR $E74C
308 : CA DEX
309 : 8A TXA
30A : 20 47 F8 JSR $F847
30D : FA PLX
30E : 8A TXA
30F : 65 26 ADC $26
311 : 85 26 STA $26
313 : B1 26 LDA ($26),Y
315 : 85 06 STA $06
317 : 60 RTS

```

COMBYTE teste la virgule et évalue X.  
 X = X - 1 (HTAB = 1 correspond à CH = 0).  
 X entassé sur la pile (pas avec le 6502 !).  
 COMBYTE pour le second paramètre  
 X = X - 1 (VTAB = 1 correspond à CV = 0).  
 X dans A.  
 GBASCALC place l'adresse de la ligne en \$26-27  
 X récupéré sur la pile...  
 et mis dans A.  
 CH additionné à partie basse de l'adresse.  
 Lecture (Y = 0 à la sortie de GBASCALC).  
 Résultat dans 6.  
 Retour.



## QUESTIONS

• Si je fais une série de :

10 INPUT "Première valeur "; A

20 INPUT "Deuxième valeur "; B etc.

et si je veux revoir ces valeurs, comment faire pour qu'un simple RETURN valide la première valeur, si elle est bonne, sans avoir à la ressaisir ? Et aussi sans passer par des PEEK et POKE ?

Roger C.

## RÉPONSES

R

Si un seul PEEK ne vous donne pas d'eczéma et si vous n'êtes pas hostile à une saisie un peu plus rationnelle, je vous propose les quelques lignes ci-après. Libre à vous, dans la suite du programme, de transformer les variables alphanumériques en variables numériques par une boucle du type :

FOR I = 1 TO 10 : A(I) = VAL (A\$(I)) : NEXT

```
10 TEXT : NORMAL : PRINT CHR$(4)"PR£3": PRINT CHR$(17): REM Supprimer la
   dernière instruction pour rester en 80 colonnes
15 FOR I = 1 TO 10
20 VTAB (I * 2) - 1: HTAB 1: PRINT "VALEUR "I" -> "; H = PEEK (1403): PRINT
   A$(I); REM Remplacer 1403 par 36 sur Apple IIe
25 POKE 1403,H: CALL - 868: REM CALL -958 effacerait toutes les lignes plac
   ées sous la lignes en cours
30 INPUT " "; R$
35 IF R$ = "" THEN POKE 36,H: VTAB (I * 2) - 1: PRINT A$(I): GOTO 45
40 A$(I) = R$
45 NEXT
50 VTAB 22: PRINT "(C)ORRECTION ""(S)UITE ""-> "; GET R$: PRINT
55 IF R$ = "C" OR R$ = "c" THEN 15
60 IF R$ = "S" OR R$ = "s" THEN 70
65 GOTO 50
70 END
```

DANS CET EXEMPLE, LA LIGNE 70 EST FICTIVE

• Après avoir tapé le programme *Histo.text* publié dans le n°8 de votre revue ainsi que la modification couleur du n°11, je me heurte à un mauvais fonctionnement. Comme vous pouvez le constater sur le feuillet joint, l'impression n'est pas bonne. Au niveau matériel j'utilise un Apple IIe avec kit 65C02, une carte super série Apple et une imprimante Imagewriter II. Si vous pouviez m'indiquer quelle erreur j'ai pu commettre, je vous en serais très reconnaissant. Dans cette attente...

François F (10000 TROYES)

• Je viens d'acheter un Apple IIGS. Avec une belle remise. Heureusement car les résultats se révèlent plutôt décevants : que faire de cette belle machine sans logiciels et sans documentation ? Comment écrire et lire une information dans un fichier direct ? Comment utiliser l'assembleur, le son, les fameux langages (C notamment) ? Pas de bouquins, si ce n'est un livre auquel je ne comprends rien d'une collaboratrice de *Tremplin Micro\**, rien dans la presse informatique sur le GS. Le néant. Un ami du Club m'a donné votre adresse et une disquette, mais ça ne marche pas. De qui se moque-t-on ?

Pierre N. (33000 BORDEAUX)

\* Pierre fait allusion à Nicole Bréaud-Pouliquen. Une nouvelle édition de ses CLEFS POUR L'APPLE IIGS vient de paraître au PSI. Ce n'est évidemment pas réservé aux débutants, mais le sujet y est bien traité.

R

Je viens de terminer la lecture des *Misérables*, en quatre volumes, une œuvre magistrale de Victor Hugo, mais je me heurte à un problème : il y a quelque chose que je n'ai pas bien compris. J'utilise pourtant des lunettes "JE-VOIS-CLAIR", une lampe de chevet "ECLAIRE-BIEN" et un matelas "TOUT-A-RESSORT". Si vous pouviez m'indiquer à quel moment je me suis endormi... sur mes *Misérables* (et aussi le mot de la ligne... de la page... du volume), je vous en serais reconnaissant. Dans cette attente...

Ben voyons ! et tout cela sans timbre pour la réponse !

R

C'est vrai que les déçus du GS sont relativement nombreux, mais la documentation existe. Les programmeurs en Applesoft seraient bien inspirés en la lisant d'un bout à l'autre. Les autres ne resteront pas très longtemps sur leur faim. Les logiciels arrivent et les livres aussi, mais laissons aux auteurs le temps de les écrire sérieusement. Pour le reste, Pierre mon ami, vous exagérez.

Lisez *Tremplin Micro* et vous constaterez que, à défaut d'encenser Apple et le GS, nous commençons à y parler de la machine. Je vous signale au passage que le fichier direct a été traité dans notre numéro 13. Mais oui ! Quant à nos disquettes, il est sûr que, sans la revue, elles sont à peu près inutilisables. ■





# Votre bibliothèque INFORMATIQUE

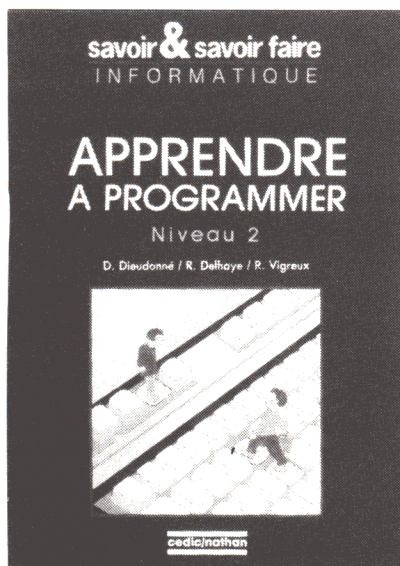
par NESTOR

- **APPRENDRE  
À PROGRAMMER (NIVEAU 2)**  
(D. Dieudonné, R. Delhaye,  
R. Vigreux)

Après le Tome 1, ce nouveau volet propose un approfondissement et un élargissement des méthodes d'analyse et de programmation. Deux notions essentielles sont notamment traitées : les procédures et les fichiers. On y trouve aussi un intéressant chapitre sur les méthodes de tri (attention ! il ne s'agit pas d'Apple-soft). La première partie donne la description d'outils et l'explication des notions permettant une programmation plus raisonnée et plus efficace. La seconde partie est un ensemble d'exemples codés en Basic, Logo et LSE afin de mettre l'accent sur l'algorithmique.

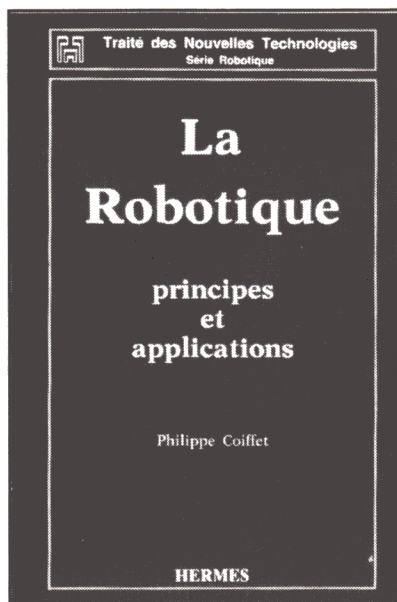
Cedic/Nathan

6, boulevard Jourdan, 75014 PARIS  
192 pages — 119 F TTC.



- **LA ROBOTIQUE**  
(Philippe Coiffet)

Dans la série des *Traité des Nouvelles Technologies*, les Editions Hermès nous proposent un maître-ouvrage signé par un spécialiste de la question. Philippe Coiffet est directeur de Recherche au



CNRS, professeur à l'Institut National des Sciences et Techniques Nucléaires, professeur à l'Université de Californie, centre de Santa Barbara, etc. Nous sommes effectivement en présence d'un gros traité consacré aux principes et applications de la Robotique. Cet ouvrage est surtout destiné aux étudiants en Génie électrique et Génie mécanique, mais il pourra évidemment intéresser toute personne capable d'en comprendre les solides exposés. Notons au passage que la présentation de cette collection est très soignée (couverture cartonnée et pelliculée). De la bonne besogne !

Editions Hermès

51, rue Rennequin, 75017 PARIS  
436 pages — 180 F TTC.

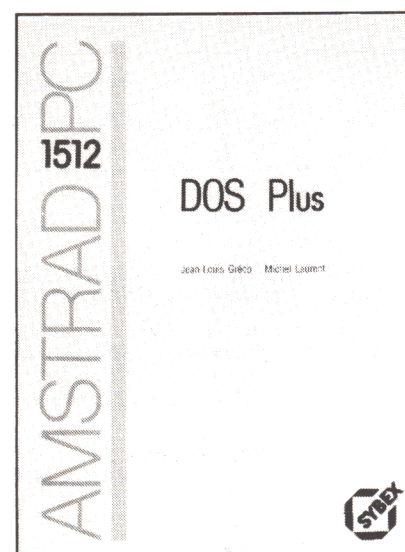
- **AMSTRAD PC 1512  
DOS PLUS**  
(J.-L. Gréco et M. Laurent)

Chacun d'entre nous connaît quelqu'un qui connaît quelqu'un qui a rencontré un copain utilisant un AMSTRAD PC 1512. Votre ami Nestor ne saurait faire exception à la règle, mais son opinion — récente — passe par un seul intermédiaire. Et je suis au regret de vous infor-

mer que cette machine est semble-t-il aussi compatible que beaucoup d'autres avec celle du Géant IBM. Il paraît même, si j'en crois mon informateur — il est en train de découper la bête en tranches —, qu'Amstrad n'a pas à rougir des performances de son petit dernier. Amstrad aura en tout cas le mérite, (du moins à mes yeux), d'amener un grand nombre d'utilisateurs à programmer un compatible. Et ici, je veux dire à Tremplin Micro, ça nous cause un très vif plaisir : à un moment où d'aucuns aimeraient transformer tous les usagers — convivialité oblige — en mouse-men ou mouse-women esclaves de logiciels super-intelligents, la naissance d'une nouvelle race de programmeurs nous plonge dans une joie sans mélange.

Je m'aperçois que je n'ai pas beaucoup parlé du bouquin. C'est apparemment un bon petit manuel, sûrement utile — sinon indispensable — à tous les utilisateurs qui veulent tirer un bon parti de l'Amstrad sous DOS Plus, système qui revendique une compatibilité maximale tant vis-à-vis de MS-DOS (version 2.1) que vis-à-vis de CP/M 86. Et oui, DOS Plus se révèle capable de lire les disquettes écrites sous MS-DOS, l'inverse n'étant pas nécessairement vrai...

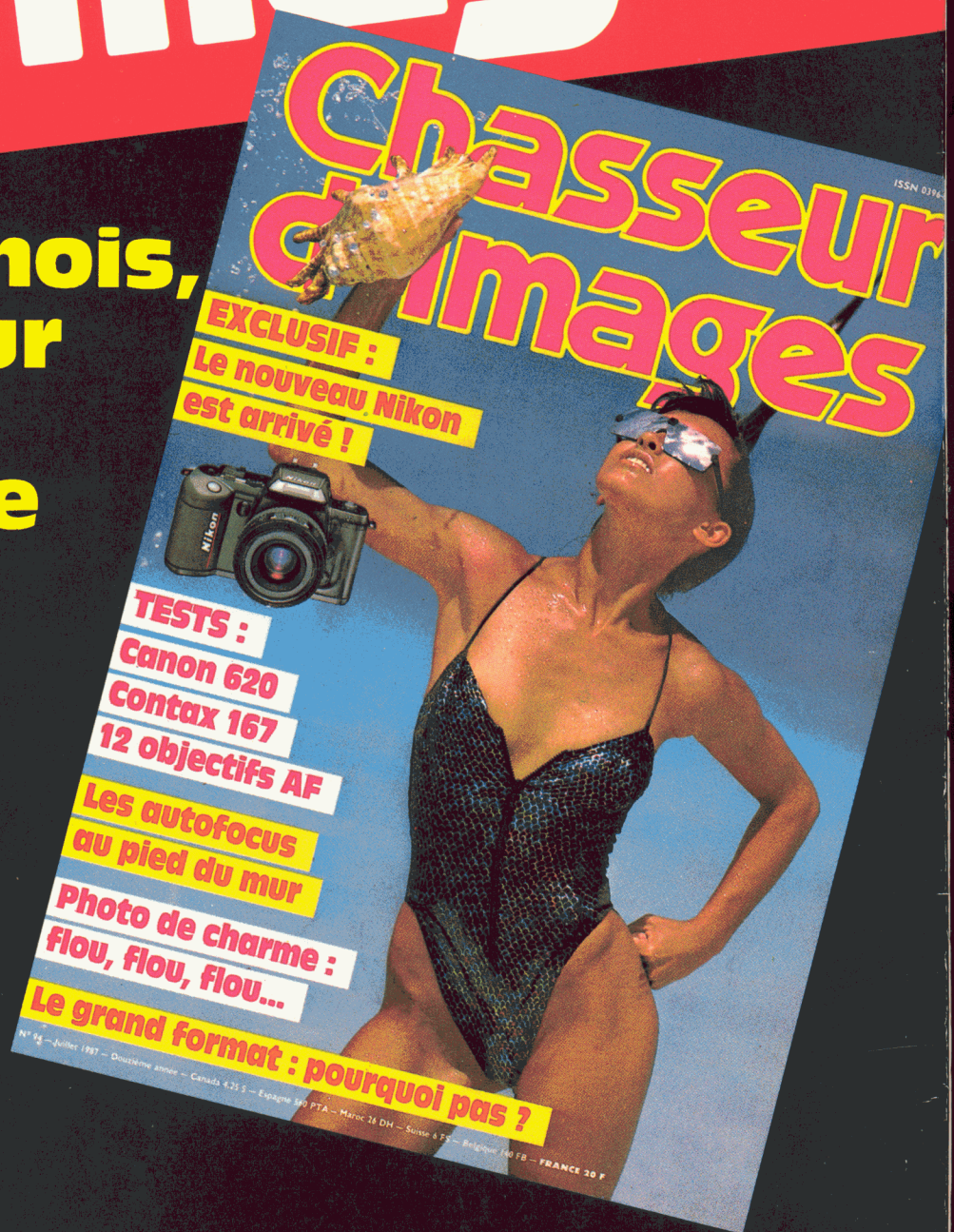
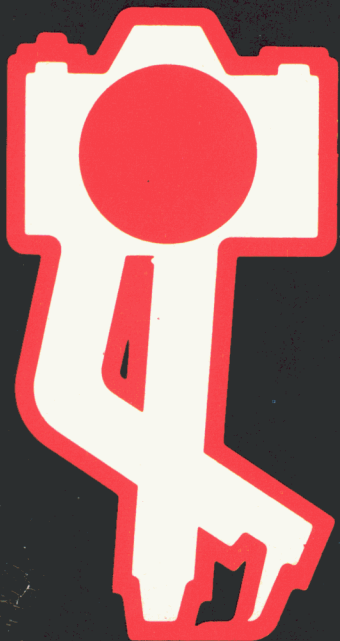
SYBEX, 6-8 Impasse du Curé, 75881 PARIS CEDEX 18  
232 pages — 128 F TTC.





# Chasseur d'Images

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



Chez votre  
marchand de journaux !