pom's

La revue francophone des utilisateurs de l'Apple

ProDOS et BUGBYTER à l'essai

Un jeu d'adresse en Pascal

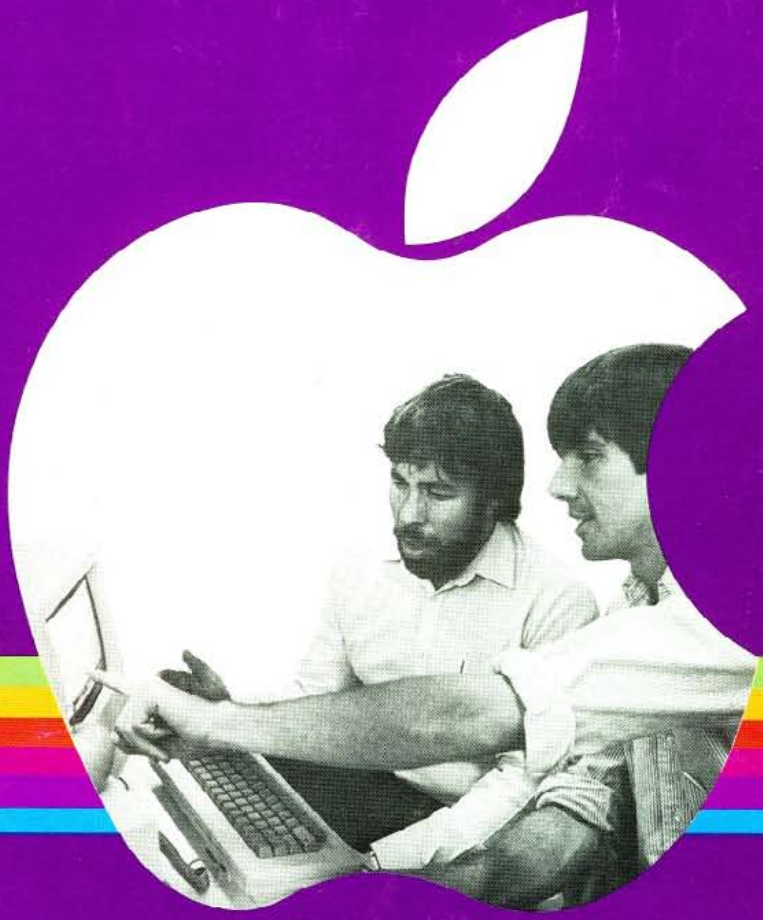
Impression des variables

PEEKs à gogo

Réduction d'images HGR

Initiation à l'assembleur (3)

Thinktank à l'essai



NUMERO 13 • PRIX 40 F

M2366-13-40 F

ISSN : 0294-6068

SPiD PRÉSENTE

LE N°2

LISTE DES POINTS DE VENTE

06000 - MAD'S - NICE - (93) 88.04.79
06210 - ÉVOLUTION 2000 - MANDELIEU - (93) 49.81.61
08600 - MICRO-BOUTIQUE JCR - GIVET - (24) 55.01.93
10000 - MICROPOLIS - TROYES - (25) 72.03.79
11000 - I ÉLEC VIDEO CLUB - CARCASSONNE - (68) 47.08.94
11000 - R 2 I INFORMATIQUE - NARBONNE - (68) 65.15.83
12000 - BASE 2 SOCODETI - RODEZ - (65) 42.50.05
13004 - ALLIANCE MARSEILLE - (91) 06.35.99
13005 - ELP INFO - MARSEILLE - (91) 94.91.13
13006 - MD SYSTÈME - JCR BOUTIQUE - MARSEILLE - (91) 37.62.33
13200 - LUDO - ARLES - (90) 96.79.03
14000 - OMB VASSARD TILLIETTE - CAEN - (31) 93.48.09
16000 - S.A. LHOMME - ANGOULÊME - (45) 92.97.37
18000 - AVENIR INFORMATIQUE - BOURGES (48) 65.16.57
19100 - MICROMATIC - BRIVE - (55) 87.15.17
19100 - INFORMATIC 19 - BRIVE - (55) 87.77.08
21000 - O.M.G. MICRO LEADER - DIJON - (80) 30.12.70 +
24100 - MICRO CYRANO INFORMATIQUE - BERGERAC - (16) 56.06.06.12 +
25206 - ITA MONTBELIARD - MONTBELLIARD CEDEX - (81) 94.50.65
26000 - DOMICA - VALENCE - (75) 41.14.75
26500 - ECA ÉLECTRONIQUE - BOURG-LES-VALENCE - (75) 42.68.88
29000 - L'ORDINATEUR 29 - QUIMPER - (98) 95.92.70
30000 - DISCOUNT INFORM. SERVICE - NIMES - (66) 23.74.21
31000 - MICRO DIFFUSION - TOULOUSE - (61) 22.81.17
33000 - MICRO DIFFUSION - BORDEAUX - (56) 81.11.99
33800 - ETS COCA - BORDEAUX - (61) 92.91.78
34000 - PIB - JCR BOUTIQUE - MONTPELLIER - (67) 58.84.37
34900 - BUREAU ORGANISATION - SÈTE - (67) 74.34.10
34500 - MARCELLEC - BÉZIERS - (67) 31.37.65
37170 - LIM - CHAMBRAY-LES-TOURS - (47) 27.29.00
38500 - MICRO AVENIR - VOIRON - (76) 65.72.35
39000 - MICRO 39 - JEAN-PIERRE-ANDRÉ LONS'LE SAUNIER (84) 24.45.39
41500 - T.I.M. - MER - (54) 81.62.47
42000 - DÉTROIT INFORMATIQUE - SAINT-ÉTIENNE - (77) 33.58.59
42100 - SAINT-ÉTIENNE COMPOSANTS - SAINT-ÉTIENNE (77) 33.50.14
42300 - MICRO SYSTÈME RHONE-ALPES - ROANNE - (77) 68.67.99 +
44100 - SILICONE VALLÉE - NANTES - (40) 73.21.67
45000 - TÉLÉPHONIE BIS ORLÉANS (38) 54.34.34
47000 - JULIEN ÉLECTRONIQUE - AGEN - (58) 66.55.64
49000 - TEMPS X - ANGERS - (41) 88.95.07
49300 - CHOLET INFORMATIQUE - CHOLET - (41) 46.02.40
54000 - SÉREC NANCY - (8) 332.12.60
56000 - L'ORDINATEUR 56 - VANNES - (97) 49 59 90
56100 - L'ORDINATEUR 56 - LORIENT - (97) 64.52.54
57504 - ARGO INFORMATIQUE - SAINT-AVOLD - (87) 92.54.84 +
57800 - CMI FREYMING MERLEBACH - (87) 81.14.89
59000 - ETS BOULANGER - LILLE - (20) 54.98.75
59000 - BÉCY INFORMATIQUE - LILLE - (20) 92.33.06
59400 - MICROSHOP - CAMBRAI - (27) 81.98.09 +
59500 - CID - DOUAI - (27) 88.47.20
59800 - M.B.D.C. - LILLE - (20) 57.91.87
60108 - QUENEUTTE - CREIL - (4) 425.04.26
60200 - LARDET S.A. - COMPIÈGNE - (4) 423.07.86
63000 - IMPACT - CLERMONT-FERRAND - (73) 92.17.55
64110 - ESPACE MICRO 64 - BAYONNE - (59) 59.41.55
64600 - INFORMATIQUE BASCO LANDAISE - ANGLET - (59) 31.96.05
66000 - SÉRIE INFORMATIQUE - PERPIGNAN - (68) 34.00.11
67150 - ETS A. FRITSCH - ERSTEIN - (88) 98.03.51
68000 - F.I.R. - COLMAR (89) 23.68.35
69003 - B.I.M.P. - LYON (7) 860.84.27
69400 - MICRO INFORM.BEAUJOLAISE - VILLEFRANCHE-S/SAONE - (74) 68.44.92
70000 - ÉLECTRO BOUTIQUE - VESOUL - (84) 76.49.52 +
71100 - AVENIR ÉLECTRONIQUE - CHALON/SAONE - (85) 48.73.35
71400 - C.H.B. ÉLECTRONIQUE - AUTUN - (85) 52.70.26
72000 - MICROTIQUE AESCULAPPLE - LE MANS - (43) 24.97.80
73100 - L'ORDINATEUR - AIX-LES-BAINS (79) 88.19.07
74102 - D.S.A. MICRO - ANNEMASSE - (50) 38.31.40
75001 - VIDEO SHOP - PARIS - (1) 296.93.95
75005 - HACHETTE - PARIS - 633.84.68
75006 - DURIEZ S.A. - PARIS - 329.05.60
75008 - ÉNERGY 8 - PARIS - 293.41.33
75009 - LE JEU ÉLECTRONIQUE - PARIS - 526.62.93 / 874.43.20
75009 - LPS BUREAU - PARIS - 878.26.~~55~~
75009 - J.C.R. ÉLECTRONIQUE PARIS - 282.19.00
75010 - GÉNÉRAL VIDEO - PARIS - 206.50.50
75010 - LOGIC STORE - PARIS - 206.72.28
75011 - COCCINOL INFORMATIQUE - PARIS - 355.63.00
75011 - P.I.T.B. PARIS 254.38.01
75012 - ELLIX - PARIS - 307.65.58
75014 - MIDEF - PARIS - 539.98.68
75015 - J.C.S. COMPOSANTS - PARIS - 355.96.22
75015 - ILLEL CENTRE - PARIS - 554.97.48
75016 - PENTASONIC - PARIS - 524 93.16
75016 - ANTIGONE - PARIS - 743.13.41
76600 - MICRO MAX - LE HAVRE - (35) 41.77.47
76600 - V.P.C. BUREAU LE HAVRE - (35) 42.49.21
76600 - L'ORDINATEUR - LE HAVRE - (35) 91 54 55
77000 - ÉPSILON - MELUN - 437.51.95
80000 - LOGIC - AMIENS - (22) 95.54.84
83000 - P.S.I. ÉLECTRONIQUE - TOULON - (94) 93.11.20
86011 - LISTE INFORMATIQUE - POITIERS CEDEX - (49) 41.43.86
87000 - MICROLIM - LIMOGES - (55) 34.10.12 +
89100 - MINI LOISIRS - SENS - (86) 64.41.91
89100 - LASOBIKOR YONNE - SENS (86) 64.51.26 +
91210 - VIDEOTRONIC - DRAVEIL - 940.28.30
92100 - AXIOME - BOULOGNE - 604.09.21
92100 - OLIG - BOULOGNE BILL - (1) 605.05.59
94100 - DIXMA - SAINT-MAUR - 885.98.22
98000 - MICROTEK 2 - MONACO - (93) 30.67.67 +
88002 - A.V.M. - ÉPINAL (29) 82.14.97

SUCCÈS OBLIGE

Le deuxième d'une longue série de guide des logiciels.
Plus d'un tiers de nouveautés.

AU SOMMAIRE :

— **Une sélection de 416 programmes en Anglais ou en Français pour :**
APPLE - ATARI - COMMODORE V20 et C64 - EPSON HX 20 - ORIC 1 et ORIC ATMOS - IBM PC - SINCLAIR ZX81 et SPECTRUM TRS 80 - THOMSON TO 7 - HECTOR.

— **Les fiches techniques de chaque programme comprenant :**
La description précise du programme.
Son prix moyen constaté.
Sa compatibilité avec tel ou tel micro.

— **En plus vous trouverez :**
Des conseils pour choisir et acheter le programme que vous cherchez.
Des index pour trouver facilement ce que vous cherchez.

EN VENTE 15 F CHEZ VOTRE DISTRIBUTEUR OU 15 F + 5 F DE PORT EN RENVOYANT LE COUPON CI-DESSOUS.

SPiD
LA HAUTE FIABILITÉ

BON DE COMMANDE A RENVOYER A **SPID** - 39, RUE V.-MASSÉ - 75009 PARIS

Je désire recevoir le "GUIDE DES LOGICIELS" Printemps 1984
Je joins 20 F en chèque (15 F + 5 F de port) en règlement.

Nom .....

Adresse .....

Code et ville .....

pom's n°13 JUILLET 1984

Sommaire

* Langage : B(asic) - A(ssembleur) - P(ascal). (B) signifie : relatif au BASIC.
** Niveau : D(ébutant) - M(oyen) - P(rofessionnel) - T(ous).
P-T signifie : programme utilisable par les débutants, mais dont la compréhension est de niveau "Professionnel".

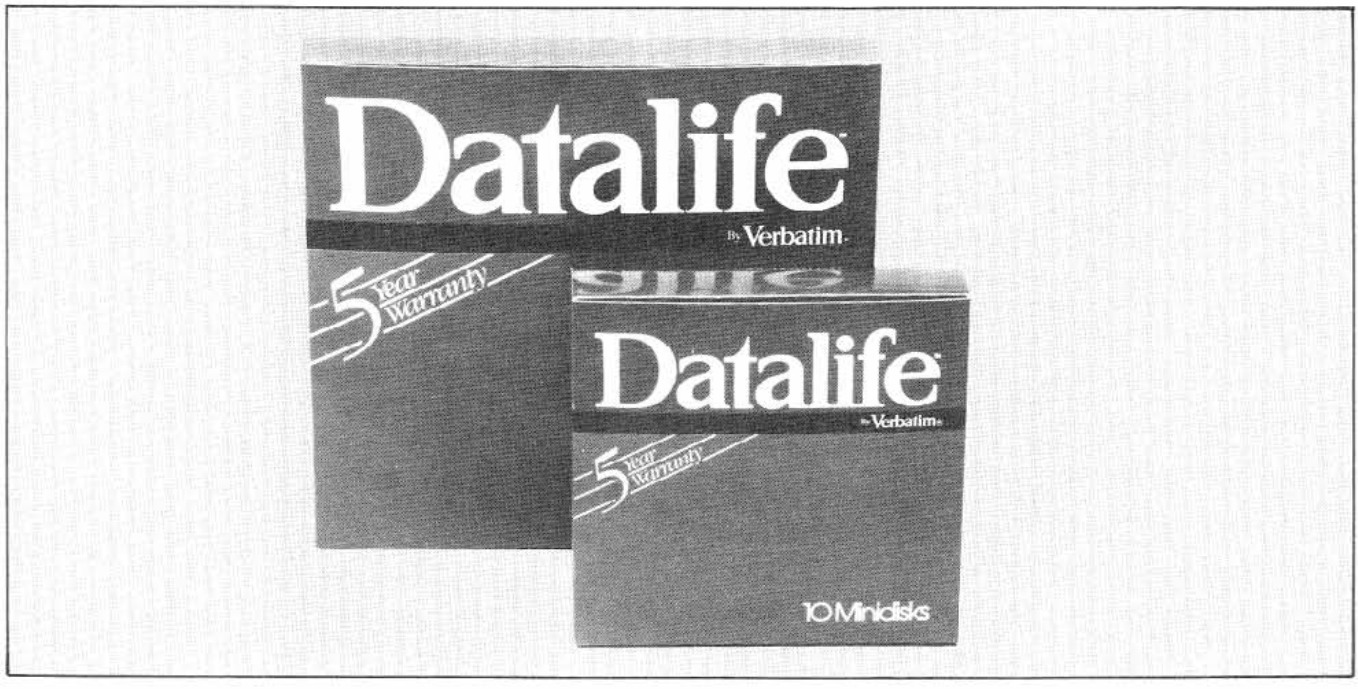
Les annonceurs

Éditions MEV — 64-70 rue des Chantiers — 78000 Versailles
Directeur de la publication : Hervé Thiriez. Imprimerie Rosay, 94300 Vincennes. Imprimé en France. Dépôt légal : 3e trimestre 1984.

Datalife

BY Verbatim

DISQUETTES ET MINI DISQUETTES TOUTES CONFIGURATIONS



- Certification unitaire 100 % sans erreur.
- Durée de vie : 30 millions de révolutions (standard de l'Industrie 3,5 millions de révolutions).
- Anneau de renforcement en standard sur le 5 1/4 ".
- 5 1/4 " en 48 et 96 TPI, simple et double face.

BFI ELECTRONIQUE - 9 RUE YVART -
75015 PARIS.
Tél. 533-01-37.

Editorial

Ouf ! Exposer dans trois salons sur la micro-informatique en six semaines, c'est franchement trop ... Certes, cela nous a donné l'occasion de rencontrer de nombreux lecteurs anciens et nouveaux, mais il est difficile de gérer trois salons puis d'exploiter leurs résultats sans prendre de retard.

Le Sicob Printemps s'est très bien passé, avec un monde fou autour des stands d'Apple et de Pom's. Un peu moins de monde à Micro Expo, dans un style plus "professionnel". Et, pour couronner le tout, ce tout premier Apple Expo très réussi, où nous avons eu le plaisir de rencontrer Steve Wozniak (Woz pour les intimes), Bill Budge (créateur entre autres de Raster Blaster, Pinball Construction Set et Mouse-Paint) et Howard Marks (créateur de Jane), parmi d'autres artistes. Quand pourrons-nous ainsi faire la connaissance d'un créateur de matériel ou d'un auteur de logiciel français devenu milliardaire à moins de 25 ans ? Dans longtemps, hélas, je le crains.

Les deux premières revues françaises de micro-informatique spécialisées sur un matériel étaient Pom's et La Commode, en septembre 1981. Il y a aujourd'hui cinq revues francophones sur l'IBM PC/XT, deux sur l'Apple, ... au total une bonne quinzaine. Nous fêtons la naissance récente de Théophile, la revue pour utilisateurs du TO7 de Thomson. Nous avons bien ri quand nous avons appris que la nouvelle version du TO7 s'appelait le MO5 : nous aurons alors peut-être une revue baptisée Hémophile, dont le slogan pourrait être "La revue des vampires de l'informatique".

*Dans ce numéro, vous trouverez tout d'abord de nombreux bancs d'essai. **Guy Lapautre** vous présente Thinktank et le résultat de ses tests sur la compatibilité entre l'Apple //c et son "grand frère" //e. **Alexandre Duback** a vu pour vous LIGHT 1, et **Alexandre Avrane** vous livre ses premières réflexions sur ProDOS et Bugbyter. Dans le domaine du matériel, **Vincent Plassard** nous fait vivre sa rencontre avec les lecteurs 1 méga de Micro-Expansion, tandis que **Kriss Grafiti** et moi-même vous donnons quelques nouvelles du Macintosh.*

***Gérard Michel** poursuit la série d'initiation à l'assembleur et vous présente également le BASICIUM, un Applesoft enrichi pour la gestion d'écran et la saisie de variables. En toute logique, **Roland Jost** vous offre des PEEKs à gogo après les POKEs du Pom's 11. Encore de nombreux utilitaires dans ce numéro, avec la documentation de tables de shapes par **Erick Ringot**, l'impression des variables d'un programme Basic par **Laurent Esnault**, des routines de HOME présentées par **Philippe François**, un programme de réduction des images HGR par **Patrice Neveu** et un petit outil d'analyse de la VTOC par **Guy d'Herbemont**.*

***Dominique Bernardi** vous permet d'exercer votre adresse avec Ordralphabetix et **Jérôme Leclercq** propose un bloc-notes à tous ceux d'entre vous qui perdent leurs "petits papiers"... Les micro-informations du jour vous seront présentées par **Jean-Michel Gourévitch** et vous trouverez également dans ce numéro 13 un sommaire thématique des articles publiés par Pom's de ses origines jusqu'au numéro 12.*

Depuis ce dernier numéro s'est créé le club Apple; il compte plus de 2000 membres en moins de deux mois. Il est vrai que ce club bénéficie du soutien total d'Apple Seedrin, ce qui n'est pas négligeable. Comme notre revue, il semble avoir pour but d'apprendre aux utilisateurs Apple à tirer un meilleur parti de leur matériel. Nous ne pouvons qu'encourager un tel objectif.

A tout hasard, nous vous rappelons que Pom's recherche toujours un collaborateur (temps partiel ou temps plein) connaissant très bien l'Apple, au moins au niveau du Basic et de l'assembleur, capable d'écrire clairement et sans faute d'ortographe. Aucune contrainte de sexe ou d'âge ...

Pour finir, nous vous donnons rendez-vous fin septembre, au stand 172 de la boutique micro du Sicob, sur le parvis de La Défense.

Hervé Thiriez

En couverture : STEVE WOZNIAK et BILL BODGE à Apple-expo
(photo : J.L. DESNOS)

Ont collaboré à ce numéro : Alexandre Avrane – Dominique Bernardi – Alexandre Duback – Laurent Esnault – Guy d'Herbemont – Philippe François – Jean-Michel Gourévitch – Kriss Graffiti – Roland Jost – Guy Lapautre – Jérôme Leclerq – Gérard Michel – Patrice Neveu – Vincent Plassard – Erick Ringot – Hervé Thiriez. **Rédacteurs :** Alexandre Avrane – Gérard Michel. **Dessins :** Laurent Bidot. **Directeur de la publication - rédacteur en chef :** Hervé Thiriez.
Siège social : Editions MEV - 49, rue Lamartine - 78000 Versailles - Tél. : (3) 951.24.43.
Diffusion N.M.P.P. : Sophie Marnez - Tél. : (1) 240.22.01.
Composition : Télécompo 13 15, avenue du Petit Parc - 94300 Vincennes - Tél. : 328.18.63.
Impression : Rosay - 47, avenue de Paris - 94300 Vincennes - Tél. 328.18.63.

Sommaire thématique de Pom's

Mes premiers émois

Kriss Graffiti

Il y a des femmes qui marchent à l'ambition, d'autres à l'argent, d'autres à la sécurité ; moi je marche à l'affection, et j'ai tout l'équipement qui va avec : un grenier plein de trucs dont je ne sais pas me débarasser, un chien, quelques vrais amies, des vrais Jules et des Jules vrais, et bien sûr, un collaborateur, OLIBRIUS INOXYDABLE.

Comme on passe beaucoup de temps ensemble, il faut qu'il soit sympathique, solide, agréable à regarder, capable d'abattre un travail de titan (mais un petit peu moins que moi quand même, histoire de garder le leadership dans mon entreprise), et puis surtout il faut qu'il m'aime d'un amour ergonomique. Cet amour régule les rapports de travail beaucoup mieux que les syndicats, mais il comporte un risque important : et si tout à coup Inoxydable tombait amoureux de quelqu'un d'autre ? Son rendement risquerait d'être considérablement diminué. C'est pourquoi j'en suis arrivée à la conclusion qu'il fallait lui trouver une autre passion, et la lui trouver moi-même. Mais qui ? Mais quoi ?

C'est à ce moment là que les trompettes célestes ont soufflé leurs notes électroniques et que Jean-Louis Gassée, le directeur d'Apple France, m'est apparu en rêve, nageant la brasse dans une piscine d'octets phosphorescents. Naturellement, comme je ne connaissais rien à rien, cette première vision resta pour moi totalement énigmatique.

Ensuite, j'ai eu d'autres visions tout aussi sybillines, mais de plus en plus fréquentes. Une deuxième, où mon Inoxydable, habillé en prestidigitateur, sortait des pommes et des souris d'un écossais débonnaire et compréhensif... Et enfin une troisième, un écossais qui soufflait dans une cornemeuse faite avec deux très belles pommes grignotées...

Je suis obligée d'être un peu sévère avec le service d'apparitions d'Apple, mais je n'y comprenais toujours rien, jusqu'au jour où j'ai rencontré le capitaine de l'informatique légère, Jean-Louis Gassée, en chair et en os, et où je lui ai raconté mes rêves. Il était assis sur un trône doré, très beau, et il m'a dit « Quand le rêve reviendra, n'oubliez pas de prononcer cette phrase + J'INITIALISE LA DISQUETTE + ». C'est ce que j'ai fait et, en me réveillant le lendemain matin, j'ai vu, à côté de mon lit, un Macintosh qui avait envie de me faire de l'œil, et je lui ai donné l'autorisation de le faire. Depuis ce jour, je suis une cheftesse d'entreprise radieuse. Inoxydable est amoureux de Mac et travaille deux fois plus et deux fois plus rapidement. Quant à ma vertu informatique, elle est définitivement compromise. Il faut dire qu'informatiquement parlant, j'étais totalement vierge et que l'initialisation me semblait une épreuve insurmontable dès que je jetais un regard inquiet en direction des ordinateurs à langage BASIC, Pascal, ou autres... Avec Mac, pas besoin de passer par ces chemins épineux pour bénéficier de ses avantages. C'est un professionnel du charme, et c'est lui qui s'adapte à vous. Vous voulez couper, coller, copier ? Il vous suffit de pointer la souris dans un menu écrit en français français, et de choisir.

C'est d'abord la suppression du bruit qui m'a séduite. Depuis des années, Inoxydable et moi écrivions dans la même pièce, sur deux machines électriques qui faisaient un raffut d'enfer, et, au-delà d'un certain nombre d'heures de travail, on était obligé de mettre des boules Quiès, ce qui nuisait gravement à la communication.

Ensuite, mon bureau, qui avait une réputation quasi internationale de foutoir, est devenu clair et ordonné. Ce n'est pas très informatique, ce que je vous dis là, mais l'atmosphère qui règne dans un bureau a une influence considérable sur la qualité du travail. Au lieu d'être en lutte permanente contre les stress du bruit, du dossier illisible, du papier perdu, du carbone sale et des archives envahissantes, l'espace-boulot a pris un air de vacances et on peut même y travailler en musique, avantage appréciable dans la profession que j'exerce.

Bien sûr, au début, Macintosh a un petit côté « hallucinogène à la californienne » et la distorsion qu'il exerce sur le temps n'est pas sans danger. S'il est vrai que mon temps de travail est réduit de plus d'un tiers par les avantages considérables du traitement de texte, je dois avouer que les joies de MacPaint et MacWrite et le confort nouveau qu'il m'offre font que j'oublie totalement l'heure et qu'il n'est pas rare que je me retrouve à 6 heures du matin en train de peaufiner un texte que j'aurais abandonné depuis longtemps si j'avais eu à le retaper entièrement. Mais, ce n'est pas moi qui vous l'apprendrai, quand on passe des nuits blanches avec quelqu'un, c'est que l'échange d'information est agréable.

Travailler au mot près, à la virgule près, offre une liberté de création totalement révolutionnaire et dame le pion à la paresse. L'esprit, dégagé de toutes sortes de contraintes pénibles, retrouve la joie pure de l'écriture.

En plus, c'est joli. Les caractères sont variés, dans leur forme et dans leur taille, l'écran est clair, on peut en régler la luminosité, on peut dessiner et intégrer les dessins aux textes, ou l'inverse, et quand on est vraiment accro, on peut mettre le beau Mac dans son sac de voyage et l'emporter en week-end. Mais oui, mais oui, ça m'est arrivé !

Ah, une dernière question, quel est le sexe de Mac ?

Inoxydable prétend que c'est une femme, à cause de la pénétration des disquettes indispensable avant toute communication vraiment intime, mais moi je prétends que c'est un homme qui a l'extrême délicatesse de mettre un KILT et de montrer ses jambes, qu'il a fort jolies, en cette saison propice aux amours.

Maintenant que je suis initialisée, j'envisage sérieusement de faire un lexique des mots tendres ou injurieux que les utilisateurs d'ordinateurs emploient vis-à-vis de leur nouveau compagnon, et qui se répercutent dans la vie quotidienne. Par exemple, quand un homme me fait la cour d'une manière qui me déplait ? Je lui réponds SYNTAX ERROR... Sans doute parce qu'avec Macintosh je n'ai jamais le chagrin d'en arriver à de telles extrémités. ■

HOME, sweet HOME

Philippe François

L'instruction HOME est une des instructions les plus utilisées dans les programmes. Elle consiste à nettoyer l'écran texte : HOME en Applesoft et Page(Output) en Pascal font appel à la routine du moniteur située en $FC58 (−936 en décimal).

Voici une série de routines qui nettoient l'écran de manière plus originale en remplissant l'écran avec le caractère de votre choix.

Le programme se situe en $9418 sous les buffers du DOS et comporte 6 points d'entrée :

- CALL 37910 : effacement très rapide (5437 cycles machine contre 15323 généralement);
- CALL 37944 : effacement vers le bas;
- CALL 38008 : effacement vers la gauche;
- CALL 38051 : effacement vers le centre;
- CALL 38113 : effacement en ciseaux;
- CALL 38184 : effacement en spirale : spectaculaire mais malheureusement très long.

Vous trouverez ci-dessous les programmes Applesoft et Pascal de démonstration et les listings source en Big Mac.

N'oubliez pas d'initialiser HIMEM à 37910 et de "poker" le code ASCII du caractère de remplissage en 37911 : ainsi POKE 37911,36 vous couvrira de dollars ... ■

```
1           ORG  $9418
2  *
3  ********************************
4  *                              *
5  *      HOME, SWEET HOME..      *
6  *                              *
7  ********************************
8  *
9  *********
10 * CONST *          Big Mac
11 *********
12 *
13 ESPACE   =    $A0       ;ascii espace
14 ZERO     =    $00
15 *
16 *******
17 * VAR *
18 *******
19 *
20 ADRLIGNE =    $28
21 LIGNE    =    $06
22 COMPTEUR =    $FE
23 *
24 *---------------------------------
25 *
26 HCNT     =    $00
27 VCNT     =    HCNT+1
28 XL       =    VCNT+1
29 XR       =    XL+1
30 YT       =    XR+1
31 YB       =    YT+1
32 *
33 *---------------------------------
34 *
35 *********************
36 * EFFACEMENT RAPIDE *
37 *********************
38 *
39          LDA  #ESPACE
40          EOR  #$80
41          LDY  #119
42 SUITE    STA  $400,Y   ;lines 0  8 16
43          STA  $500,Y   ;      2 10 18
44          STA  $600,Y   ;      4 12 20
45          STA  $700,Y   ;      6 14 22
46          STA  $480,Y   ;      1  9 17
47          STA  $580,Y   ;      3 11 19
48          STA  $680,Y   ;      5 13 21
49          STA  $780,Y   ;      7 15 23
50          DEY
51          BPL  SUITE
52          RTS
53 *
54 *
55 *
56 *************************
57 * EFFACEMENT VERS LE BAS *
58 *************************
59 *
60          LDA  #24
61          STA  $FE
62 *
63 ENCORE   LDX  #23       ;derniere ligne
64          LDA  ADRL,X   ;chargement de la partie basse
65          STA  ADRLIGNE ;on stocke dans adrligne
66          LDA  ADRH,X   ;chargement partie haute
67          STA  ADRLIGNE+1
68 NEXT     LDA  ADRLIGNE
69          STA  LIGNE
70          LDA  ADRLIGNE+1
71          STA  LIGNE+1
72          DEX
73          LDA  ADRL,X
74          STA  ADRLIGNE
75          LDA  ADRH,X
76          STA  ADRLIGNE+1
77 *
78          LDY  #39       ;on part de la droite
79 DEPLACE  LDA  (ADRLIGNE),Y :deplace caract. vers bas
80          STA  (LIGNE),Y
81          DEY
82          BPL  DEPLACE
83 *
84          CPX  #00       ;a-t-on atteint le bas?
85
86          BEQ  EFFSOM
87          BPL  NEXT      : = jmp next
88 *
89 EFFSOM   LDY  #39       ; coin droit
90          LDA  #ESPACE
91 EFFACE   STA  (ADRLIGNE),Y
92          DEY
93          BPL  EFFACE
94 *
95          DEC  COMPTEUR
96          BNE  ENCORE
97          RTS
98 *
99 ***************************
100 * EFFACEMENT VERS LA GAUCHE *
101 ***************************
102 *
103         LDA  #40       ;40 caracteres par ligne
104         STA  COMPTEUR
105 *
106 ENCORE2  LDX  #ZERO     ;ligne sommet
107 NXTLIGN  LDY  #ZERO     ;colonne de gauche
108         LDA  ADRL,X
109         STA  ADRLIGNE
110         LDA  ADRH,X
111         STA  ADRLIGNE+1
112 CONT     INY
113         LDA  (ADRLIGNE),Y
114         DEY
115         STA  (ADRLIGNE),Y
116         INY
117         CPY  #39       ;derniere colonne?
118         BNE  CONT
119 *
120 * DERNIERE COLONNE *
121 *
122         LDA  #ESPACE
123         STA  ($28),Y
124         INX
125         CPX  #24       ;derniere ligne?
126         BNE  NXTLIGN
127 *
128         DEC  COMPTEUR
129         BNE  ENCORE2
130         RTS
131 *
132 ***************************
133 * EFFACEMENT PAR LE CENTRE *
134 ***************************
135 *
136         LDY  #20       ;initial. compteur de boucle
137         STY  COMPTEUR
138 *
139 BOUCLE   LDX  #ZERO     ;initialisation ligne
140 *
141 NEXTLIGN LDY  #16       ;initialisation colonne
142         LDA  ADRL,X
143         STA  ADRLIGNE
144         LDA  ADRH,X
145         STA  ADRLIGNE+1
146 *
```

```
147 DEPLACD  LDA  (ADRLIGNE),Y ;recuper. d'un caractere
148          INY
149          STA  (ADRLIGNE),Y ;on le deplace d'un rang
150          DEY
151          BEQ  GAUCH     ;on sort si derniere col
152          DEY
153          BPL  DEPLACD   ;=JMP
154 *
155 GAUCH    LDA  #ESPACE
156          STA  (ADRLIGNE),Y
157          LDY  #21       ;initialisation col
158 *
159 DEPG     LDA  (ADRLIGNE),Y;on place un caractere
160          DEY
161          STA  (ADRLIGNE),Y
162          INY
163          CPY  #39       ;est-on a la derniere col
164          BEQ  DROIT     ;si oui on sort
165          INY
166          BPL  DEPG      ;=JMP
167 *
168 DROIT    LDA  #ESPACE
169          STA  (ADRLIGNE),Y
170          INX
171          CPX  #24       ;derniere ligne?
172          BNE  NEXTLIGN
173 *
174          DEC  COMPTEUR
175          BNE  BOUCLE
176          RTS
177 *
178 *
179 ******************************************
180 * EFFACEMENT GAUCHE-DROITE & DROITE-GAUCHE *
181 ******************************************
182 *
183          LDX  #40
184          STX  COMPTEUR
185 BOUCLE2  LDX  #ZERO
186 GAUCHE   LDA  ADRL,X
187          STA  ADRLIGNE
188          LDA  ADRH,X
189          STA  ADRLIGNE+1
190          LDY  #$01
191 DEPLACG  LDA  (ADRLIGNE),Y
192          DEY
193          STA  (ADRLIGNE),Y
194          INY
195          CPY  #39
196          BEQ  BLANCD
197          INY
198          BPL  DEPLACG
199 *
200 BLANCD   LDA  #$A0
201          STA  (ADRLIGNE),Y ;a ce moment y=39
202          INX
203          LDA  ADRL,X
204          STA  ADRLIGNE
205          LDA  ADRH,X
206          STA  ADRLIGNE+1
207 *
208 DEPLACDT DEY
209          LDA  (ADRLIGNE),Y
210          INY
211          STA  (ADRLIGNE),Y
212          DEY
213          BEQ  BLANCL
214          BNE  DEPLACDT
215 *
216 BLANCL   LDA  #ESPACE
217          STA  (ADRLIGNE),Y;a ce moment y-0
218          INX
219          CPX  #24       ; termine?
220          BNE  GAUCHE
221          DEC  COMPTEUR
222          BNE  BOUCLE2
223          RTS
224 *
225 ********************************
226 *       "HOME" EN SPIRALE      *
227 ********************************
228 *
229 DEBUT    LDA  #24       ; pour 24 lignes
230          STA  VCNT
231 DEB1     LDA  #40       ; pour 40 colonnes
232          STA  HCNT
233 DEB2     JSR  ROTATION
234          LDA  #ESPACE
235          STA  $628+12   ; au milieu de l'ecran
236 *
237          DEC  HCNT
238          BNE  DEB2
239          DEC  VCNT
240          BNE  DEB1
241          RTS            ; c'est fini !
242 *
243 ROTATION
244          LDA  #$00
245          STA  XL        ; bord gauche
246          STA  YT        ; sommet
247          LDA  #23
248          STA  YB        ; bas
249          LDA  #39
250          STA  XR        ; bord droit
251 *
252          JSR  BAS       ; on part du cote gauche
253          BEQ  ROT2
254 ROT1     JSR  BAS
255          DEC  YB
256 ROT2     JSR  GAUCHE1
257          INC  XL
258          JSR  HAUT
259          INC  YT
260          JSR  DROIT1
261          DEC  XR
262          LDA  YT
263          CMP  YB
264          BCC  ROT1
265          RTS
266 *
267 ********************************
268 *ON SE DEPLACE VERS LA GAUCHE ET*
269 *          VERS LE BAS         *
270 ********************************
271 *
272 BAS      LDY  XL
273          LDX  YB
274 BAS1     DEX
275          JSR  CALCULE
276          LDA  (ADRLIGNE),Y
277          PHA
278          INX
279          JSR  CALCULE
280          PLA
281          STA  (ADRLIGNE),Y
282          DEX
283          CPX  YT
284          BNE  BAS1
285          RTS
286 *
287 ********************************
288 *ON SE DEPLACE VERS LA DROITE ET*
289 *         VERS LE HAUT         *
290 ********************************
291 *
292 HAUT     LDY  XR
293          LDX  YT
294 HAUT1    INX
295          JSR  CALCULE
296          LDA  (ADRLIGNE),Y
297          PHA
298          DEX
299          JSR  CALCULE
300          PLA
301          STA  (ADRLIGNE),Y
302          INX
303          CPX  YB
304          BNE  HAUT1
305          RTS
306 *
307 ********************************
308 *ON SE DEPLACE VERS LE SOMMET ET*
309 *       VERS LA GAUCHE         *
310 ********************************
311 *
312 GAUCHE1  LDX  YT
313          JSR  CALCULE
314          LDY  XL
315 GAUCHE2  INY
316          LDA  (ADRLIGNE),Y
317          DEY
318          STA  (ADRLIGNE),Y
319          INY
320          CPY  XR
321          BNE  GAUCHE2
322          RTS
323 *
324 ********************************
325 * ON SE DEPLACE VERS LE BAS ET  *
326 *       VERS LA DROITE         *
327 ********************************
328 *
329 DROIT1   LDX  YB
330          JSR  CALCULE
331          LDY  XR
332 DROIT2   DEY
333          LDA  (ADRLIGNE),Y
334          INY
335          STA  (ADRLIGNE),Y
336          DEY
337          CPY  XL
338          BNE  DROIT2
339          RTS
340 *
341 ************************
342 *  CALCUL DE L'ADRESSE  *
343 ************************
344 *
```

```
345 CALCULE  LDA  ADRH,X
346          STA  ADRLIGNE+1
347          LDA  ADRL,X
348          STA  ADRLIGNE
349          RTS
350 *
351 * PARTIE BASSE DES ADRESSES  PAGE TEXT *
352 *
353 ADRL     HEX  0000000000000080 ;lignes 0-7
354          HEX  28A828A828A828A8 ;lignes 8-15
355          HEX  50D050D050D050D0 ;lignes 16-23
```

```
356 *
357 * PARTIE HAUTE DES ADRESSES PAGE TEXT *
358 *
359 *
360 ADRH     HEX  0404050506060707 ;lignes 0-7
361          HEX  0404050506060707 ;lignes 8-15
362          HEX  0404050506060707 ;lignes 16-23
363 *
364 *
365          LST  OFF
```

```
1  REM   ** ESSAI SWEE
      T HOME **
5  HOME : VTAB 1: HTAB
      11: PRINT "HOM
      E, SWEET HOME..
      "
10 HIMEM: 37910: PRIN
      T  CHR$ (4)"BLO
      AD SWEET HOME"
20 INPUT "DONNEZ LE C
      ARACTERE ";A$:
      POKE 37911, ASC
       (A$)
30 CALL 37910: CALL 3
      7944: REM  EFFA
      CEMENT PAR LE B
      AS
40 CALL 37910: CALL 3
      8008: REM  EFFA
      CEMENT PAR LA G
      AUCHE
50 CALL 37910: CALL 3
      8051: REM  EFFA
      CEMENT PAR LE C
      ENTRE
60 CALL 37910: CALL 3
      8113: REM  EFFA
      CEMENT GAUCHE D
      ROITE-DROITE GA
      UCHE
70 CALL 37910: CALL 3
      8184: REM  EFFA
      CEMENT EN SPIRA
      LE
80 END
```

Récapitulation HOME

```
9416- A9 3F
9418- 49 80 A0 77 99 00 04 99
9420- 00 05 99 00 06 99 00 07
9428- 99 80 04 99 80 05 99 80
9430- 06 99 80 07 88 10 E5 60
9438- A9 18 85 FE A2 17 BD D0
9440- 95 85 28 BD E8 95 85 29
9448- A5 28 85 06 A5 29 85 07
9450- CA BD D0 95 85 28 BD E8
9458- 95 85 29 A0 27 B1 28 91
9460- 06 88 10 F9 E0 00 F0 02
9468- 10 DE A0 27 A9 A0 91 28
9470- 88 10 FB C6 FE D0 C5 60
9478- A9 28 85 FE A2 00 A0 00
9480- BD D0 95 85 28 BD E8 95
9488- 85 29 C8 B1 28 88 91 28
9490- C8 C0 27 D0 F5 A9 A0 91
9498- 28 E8 E0 18 D0 E0 C6 FE
94A0- D0 DA 60 A0 14 84 FE A2
94A8- 00 A0 12 BD D0 95 85 28
94B0- BD E8 95 85 29 B1 28 C8
94B8- 91 28 88 F0 03 88 10 F5
94C0- A9 A0 91 28 A0 15 B1 28
94C8- 88 91 28 C8 C0 27 F0 03
94D0- C8 10 F3 A9 A0 91 28 E8
94D8- E0 18 D0 CD C6 FE D0 C7
94E0- 60 A2 28 86 FE A2 00 BD
94E8- D0 95 85 28 BD E8 95 85
94F0- 29 A0 01 B1 28 88 91 28
94F8- C8 C0 27 F0 03 C8 10 F3
9500- A9 A0 91 28 E8 BD D0 95
9508- 85 28 BD E8 95 85 29 88
9510- B1 28 C8 91 28 88 F0 02
9518- D0 F5 A9 A0 91 28 E8 E0
9520- 18 D0 C4 C6 FE D0 BE 60
9528- A9 18 85 01 A9 28 85 00
9530- 20 41 95 A9 A0 8D 34 06
9538- C6 00 D0 F4 C6 01 D0 EC
9540- 60 A9 00 85 02 85 04 A9
9548- 17 85 05 A9 27 85 03 20
9550- 6F 95 F0 05 20 6F 95 C6
9558- 05 20 9F 95 E6 02 20 87
9560- 95 E6 04 20 B2 95 C6 03
9568- A5 04 C5 05 90 E6 60 A4
9570- 02 A6 05 CA 20 C5 95 B1
9578- 28 48 E8 20 C5 95 68 91
9580- 28 CA E4 04 D0 ED 60 A4
9588- 03 A6 04 E8 20 C5 95 B1
9590- 28 48 CA 20 C5 95 68 91
9598- 28 E8 E4 05 D0 ED 60 A6
95A0- 04 20 C5 95 A4 02 C8 B1
95A8- 28 88 91 28 C8 C4 03 D0
95B0- F5 60 A6 05 20 C5 95 A4
95B8- 03 88 B1 28 C8 91 28 88
95C0- C4 02 D0 F5 60 BD E8 95
95C8- 85 29 BD D0 95 85 28 60
95D0- 00 80 00 80 00 80 00 80
95D8- 28 A8 28 A8 28 A8 28 A8
95E0- 50 D0 50 D0 50 D0 50 D0
95E8- 04 04 05 05 06 06 07 07
95F0- 04 04 05 05 06 06 07 07
95F8- 04 04 05 05 06 06 07 07
```

```
PROGRAM ESSAI_HOME;

VAR
  CAR : CHAR;

PROCEDURE HOME(CARACTERE:CHAR); EXTERNAL;
PROCEDURE EFFACE_BAS; EXTERNAL;
PROCEDURE EFFACE_GAUCHE; EXTERNAL;
PROCEDURE EFFACE_CENTRE; EXTERNAL;
PROCEDURE EFFACE_GD_DG; EXTERNAL;
PROCEDURE EFFACE_SPIRALE; EXTERNAL;

BEGIN
  WRITE('Donnez un caractere ');READ(CAR);
  HOME(CAR);
  EFFACE_BAS;
  HOME(CAR);
  EFFACE_GAUCHE;
  HOME(CAR);
  EFFACE_CENTRE;
  HOME(CAR);
  EFFACE_GD_DG;
  HOME(CAR);
  EFFACE_SPIRALE;
END.
```

```
;---------------------------------
;   macro pop & push
;
              .MACRO POP
              PLA
              STA %1
              PLA
              STA %1+1
              .ENDM

              .MACRO PUSH
              LDA %1+1
              PHA
              LDA %1
              PHA
              .ENDM
```

```
;
;*********************************
;*                               *
;*      HOME, SWEET HOME..       *
;*                               *
;*********************************
;
              .PROC HOME.1

              .REF ADRL, ADRH
;
;******
; VAR *
;******
;
RETURN        .EQU 00
```

```
ADRLIGNE        .EQU 28
LIGNE           .EQU 06
COMPTEUR        .EQU OFE
;
;*********************
;* EFFACEMENT RAPIDE *
;*********************
;
                POP RETURN

                PLA       : recuperation du parametre
                EOR #80

                LDY #119.
SUITE           STA 400,Y ;lines 0  8 16
                STA 500,Y ;       2 10 18
                STA 600,Y ;       4 12 20
                STA 700,Y ;       6 14 22
                STA 480,Y ;       1  9 17
                STA 580,Y ;       3 11 19
                STA 680,Y ;       5 13 21
                STA 780,Y ;       7 15 23
                DEY
                BPL SUITE

                PUSH RETURN

                RTS
;
;
;
;*************************
;* EFFACEMENT VERS LE BAS *
;*************************
;
                .PROC EFFACE_BAS

                .REF ADRL, ADRH
;
;******
; VAR *
;******
;
RETURN          .EQU 00
ADRLIGNE        .EQU 28
LIGNE           .EQU 06
COMPTEUR        .EQU OFE

                POP RETURN

                LDA #24.
                STA OFE
;
ENCORE          LDX #23. ;derniere ligne
                LDA ADRL,X ;chargement partie basse
                STA ADRLIGNE ;on stocke dans adrligne
                LDA ADRH,X ;chargement partie haute
                STA ADRLIGNE+1
NEXT            LDA ADRLIGNE
                STA LIGNE
                LDA ADRLIGNE+1
                STA LIGNE+1
                DEX
                LDA ADRL,X
                STA ADRLIGNE
                LDA ADRH,X
                STA ADRLIGNE+1
;
                LDY #39. ;on part de la droite
DEPLACE         LDA @ADRLIGNE,Y ;le caract. vers bas
                STA (LIGNE),Y
                DEY
                BPL DEPLACE
;
                CPX #00 ;a-t-on atteint le bas?

                BEQ EFFSOM
                BPL NEXT ;  =  jmp next
;
EFFSOM          LDY #39. ; coin droit
                LDA #0A0
EFFACE          STA @ADRLIGNE,Y
                DEY
                BPL EFFACE
               ;
                DEC COMPTEUR
                BNE ENCORE

                PUSH RETURN

                RTS
;
;*****************************
;* EFFACEMENT VERS LA GAUCHE *
;*****************************
;
                .PROC EFFACE_GAUCHE

                .REF ADRL, ADRH
;
;******
; VAR *
;******
;
RETURN          .EQU 00
ADRLIGNE        .EQU 28
LIGNE           .EQU 06
COMPTEUR        .EQU OFE

                POP RETURN

                LDA #40. ;40 caracteres par ligne
                STA COMPTEUR
;
ENCORE2         LDX #00 ;ligne sommeT
NXTLIGN         LDY #00 ;colonne de gauche
                LDA ADRL,X
                STA ADRLIGNE
                LDA ADRH,X
                STA ADRLIGNE+1
CONT            INY
                LDA @ADRLIGNE,Y
                DEY
                STA @ADRLIGNE,Y
                INY
                CPY #39. ;derniere colonne?
                BNE CONT
;
; DERNIERE COLONNE *
;
                LDA #0A0
                STA (28),Y
                INX
                CPX #24. ;derniere ligne?
                BNE NXTLIGN
;
                DEC COMPTEUR
                BNE ENCORE2

                PUSH RETURN

                RTS
;
;*****************************
;* EFFACEMENT PAR LE CENTRE *
;*****************************
;
                .PROC EFFACE_CENTRE

                .REF ADRL, ADRH

;
;******
; VAR *
;******
;
RETURN          .EQU 00
ADRLIGNE        .EQU 28
LIGNE           .EQU 06
COMPTEUR        .EQU OFE

                POP RETURN

                LDY #20. ;initial. compteur de boucle
                STY COMPTEUR
;
BOUCLE          LDX #00 ;initialisation ligne
;
NEXTLIGN        LDY #18. ;initialisation colonne
                LDA ADRL,X
                STA ADRLIGNE
                LDA ADRH,X
                STA ADRLIGNE+1
;
DEPLACD         LDA @ADRLIGNE,Y ;recup. d'un caractere
                INY
                STA @ADRLIGNE,Y :le deplace d'un rang
                DEY
                BEQ GAUCH ;on sort si derniere col
                DEY
                BPL DEPLACD ;.EQUJMP
;
GAUCH           LDA #0A0
                STA @ADRLIGNE,Y
                LDY #21. ;initialisation col
;
DEPG            LDA @ADRLIGNE,Y;on place un caractere
                DEY
                STA @ADRLIGNE,Y
                INY
                CPY #39. ;est-on a la derniere col
                BEQ DROIT ;si oui on sort
```

```
                INY
                BPL DEPG :.EQUJMP
;
DROIT           LDA #0A0
                STA @ADRLIGNE,Y
                INX
                CPX #24. ;derniere ligne?
                BNE NEXTLIGN
;
                DEC COMPTEUR
                BNE BOUCLE

                PUSH RETURN

                RTS
;
;
;*****************************************
;* EFFACEMENT GAUCHE-DROITE & DROITE-GAUCHE *
;*****************************************
;
                .PROC EFFACE_GD_DG

                .REF ADRL, ADRH

;
;******
; VAR *
;******
;
RETURN          .EQU 00
ADRLIGNE        .EQU 28
LIGNE           .EQU 06
COMPTEUR        .EQU 0FE
                POP RETURN

                LDX #40.
                STX COMPTEUR
BOUCLE2         LDX #00
GAUCHE          LDA ADRL,X
                STA ADRLIGNE
                LDA ADRH,X
                STA ADRLIGNE+1
                LDY #01
DEPLACG         LDA @ADRLIGNE,Y
                DEY
                STA @ADRLIGNE,Y
                INY
                CPY #39.
                BEQ BLANCD
                INY
                BPL DEPLACG
;
BLANCD          LDA #0A0
                STA @ADRLIGNE,Y ;a ce moment y= 39
                INX
                LDA ADRL,X
                STA ADRLIGNE
                LDA ADRH,X
                STA ADRLIGNE+1
;
DEPLACDT        DEY
                LDA @ADRLIGNE,Y
                INY
                STA @ADRLIGNE,Y
                DEY
                BEQ BLANCL
                BNE DEPLACDT
;
BLANCL          LDA #0A0
                STA @ADRLIGNE,Y;a ce moment y-0
                INX
                CPX #24. ; termine?
                BNE GAUCHE
                DEC COMPTEUR
                BNE BOUCLE2

                PUSH RETURN

                RTS
;
;*********************************
;*       "HOME" EN SPIRALE       *
;*********************************
;
                .PROC EFFACE SPIRALE

                .DEF ADRL, ADRH
;
;--------------------------------
;
RETURN          .EQU 00
HCNT            .EQU 0A
VCNT            .EQU HCNT+1
XL              .EQU VCNT+1
XR              .EQU XL+1
YT              .EQU XR+1
YB              .EQU YT+1
ADRLIGNE        .EQU 28
;
;--------------------------------

                POP RETURN

DEBUT           LDA #24. ; pour 24 lignes
                STA VCNT
DEB1            LDA #40. ; pour 40 colonnes
                STA HCNT
DEB2            JSR ROTATION
                LDA #0A0
                STA 628+12. ; au milieu de l'ecran
;
                DEC HCNT
                BNE DEB2
                DEC VCNT
                BNE DEB1

                PUSH RETURN

                RTS  ; c'est fini !
;
ROTATION
                LDA #00
                STA XL ; bord gauche
                STA YT ; sommet
                LDA #23.
                STA YB ; bas
                LDA #39.
                STA XR ; bord droit
;
                JSR BAS ; on part du cote gauche
                BEQ ROT2
ROT1            JSR BAS
                DEC YB
ROT2            JSR GAUCHE1
                INC XL
                JSR HAUT
                INC YT
                JSR DROIT1
                DEC XR
                LDA YT
                CMP YB
                BCC ROT1
                RTS
;
;*********************************
;*ON SE DEPLACE VERS LA GAUCHE ET*
;*          VERS LE BAS          *
;*********************************
;
BAS             LDY XL
                LDX YB
BAS1            DEX
                JSR CALCULE
                LDA @ADRLIGNE,Y
                PHA
                INX
                JSR CALCULE
                PLA
                STA @ADRLIGNE,Y
                DEX
                CPX YT
                BNE BAS1
                RTS
;
;*********************************
;*ON SE DEPLACE VERS LA DROITE ET*
;*         VERS LE HAUT          *
;*********************************
;
HAUT            LDY XR
                LDX YT
HAUT1           INX
                JSR CALCULE
                LDA @ADRLIGNE,Y
                PHA
                DEX
                JSR CALCULE
                PLA
                STA @ADRLIGNE.Y
                INX
                CPX YB
                BNE HAUT1
                RTS
;
;*********************************
;*ON SE DEPLACE VERS LE SOMMET ET*
;*       VERS LA GAUCHE          *
;*********************************
;
GAUCHE1         LDX YT
                JSR CALCULE
```

```
                LDY XL
GAUCHE2         INY
                LDA @ADRLIGNE,Y
                DEY
                STA @ADRLIGNE,Y
                INY
                CPY XR
                BNE GAUCHE2
                RTS
;
;*******************************
;* ON SE DEPLACE VERS LE BAS ET *
;*       VERS LA DROITE         *
;*******************************
;
DROIT1          LDX YB
                JSR CALCULE
                LDY XR
DROIT2          DEY
                LDA @ADRLIGNE,Y
                INY
                STA @ADRLIGNE,Y
                DEY
                CPY XL
                BNE DROIT2
                RTS
```

```
;
;************************
;*  CALCUL DE L'ADRESSE  *
;************************
;
CALCULE         LDA ADRH,X
                STA ADRLIGNE+1
                LDA ADRL,X
                STA ADRLIGNE
                RTS
;
; PARTIE BASSE DES ADRESSES  PAGE TEXT *
;
ADRL    .BYTE 00,80,00,80,00,80,00,80 ;lignes 0-7
        .BYTE 28,0A8,28,0A8,28,0A8,28,0A8 ;lignes 8-15
        .BYTE 50,0D0,50,0D0,50,0D0,50,0D0 ;lignes 16-23
;
; PARTIE HAUTE DES ADRESSES PAGE TEXT *
;
;
ADRH    .BYTE 04,04,05,05,06,06,07,07 ;lignes 0-7
        .BYTE 04,04,05,05,06,06,07,07 ;lignes 8-15
        .BYTE 04,04,05,05,06,06,07,07 ;lignes 16-23
;
;
        .END
```

Bugbyter à l'essai

Alexandre Avrane

L'Apple ne connaît qu'un seul langage, celui du processeur 6502, et pourtant peu d'utilisateurs l'exploitent; la plupart se considèrent définitivement brouillés, après la lecture des manuels Apple, avec la programmation en assembleur.

Bugbyter est un outil de programmation facilitée en assembleur : il assiste l'utilisateur dans la mise au point des programmes en 6502, en lui offrant des possibilités originales par rapport à ce qu'offrent le moniteur de l'Apple ou les divers assembleurs disponibles.

Mettre au point un programme, c'est avant tout pouvoir l'exécuter instruction par instruction (STEP/TRACE) pour déceler l'origine d'une erreur. Il est certes possible de reprendre les routines de STEP/TRACE disponibles sur le moniteur d'origine de l'Apple II (avant le II Plus) ou avec le Moniteur Etendu (Pom's 8), mais les informations affichées restent rudimentaires (en particulier la valeur des bits du registre d'état).

Bugbyter s'est donné les moyens d'un "debugging" agréable ; en effet l'écran affiche simultanément :

- la douzaine d'instructions précédant et suivant la position actuelle;
- les valeurs en hexadécimal ou en binaire des registres (A/X/Y/P/S) du 6502 et leur évolution;
- la position courante du compteur de programme (PC);
- la situation actuelle de la pile du 6502, ainsi que le pointeur de pile;
- le nombre de cycles-machine exécutés;
- les valeurs d'une demi-douzaine d'adresses stratégiques que l'utilisateur aura précisées,
- la situation des points d'arrêts, également spécifiés par l'utilisateur, et le nombre de fois qu'ils ont été atteints.

Il est également possible d'observer l'évolution des zones d'écran texte ou haute résolution, page 1 ou 2, et de faire varier la vitesse d'exécution, éventuellement à l'aide d'un paddle.

Bugbyter fonctionne comme un interpréteur des instructions du 6502, ce qui ralentit évidemment le déroulement du programme; il est néanmoins possible de préciser une zone mémoire qui devra être exécutée à pleine vitesse, par exemple la routine RWTS du DOS. Entièrement relogeable, Bugbyter peut également être implanté sur une carte langage.

Les problèmes de contention (collision) sont réduits au minimum et Bugbyter ne réserve en fait qu'une trentaine d'octets sur la pile du 6502; toutes les adresses en page zéro sont sauvegardées et restaurées; l'affichage vidéo peut être supprimé; il est également possible de spécifier que tous les caractères du clavier sauf un seront reçus par le programme exécutant, le dernier servant à arrêter le TRACE.

Si les caractéristiques de Bugbyter s'arrêtaient là, ce serait déjà un excellent produit. Mais il offre en prime :

- un mini-assembleur résident, plus pratique que celui d'Apple;
- un traducteur décimal/hexadécimal;
- la possibilité de visualiser la mémoire par dump hexadécimal et ASCII;
- la modification facile des quatre fenêtres d'affichage à l'écran;
- l'installation de points d'arrêts (breakpoints) virtuels (un BRK n'est pas inséré);
- l'exécution sur Apple II, II+, //e, et /// en mode émulation;
- non protection et fonctionnement sous DOS 3.2 ou 3.3, ce qui permet de l'utiliser conjointement avec un assembleur, Big Mac par exemple, sans devoir recharger l'un des deux programmes. Un seul regret : il est impossible d'obtenir une sortie sur imprimante des informations produites.

En rapport performance/prix, un seul concurrent sérieux : le Moniteur Etendu du Pom's 8 qui offre de nombreuses possibilités semblables, ainsi que certaines facilités en plus ou en moins par rapport à Bugbyter. Le prix du numéro et de la disquette : 95 F.

En guise de conclusion, Bugbyter est, à un coût raisonable (environ 450 FF, mais difficile à trouver), une nécessité à la fois pour le novice qui veut apprendre l'assembleur sans douleur, et pour l'utilisateur plus expérimenté qui gagnera un temps précieux de mise au point et osera aborder certains programmes très complexes. La programmation en assembleur n'est plus un art avec ses moments d'intuition et de désespoir, mais devient une technique. Est-ce dommage ?

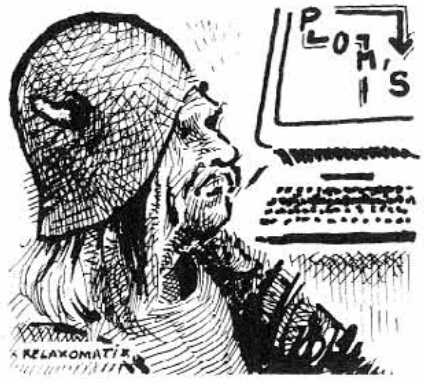
■

Un jeu d'adresse en Pascal : ORDRALPHABETIX

Dominique Bernardi

J'ai voulu voir s'il était vraiment possible d'utiliser l'assembleur UCSD. En fait, ça l'est et on découvre même à l'usage que le passage de paramètres entre le Pascal et son assembleur est nettement plus simple qu'en Basic (pour la mise au point, c'est autre chose !). En particulier, on peut dans le programme Pascal redéfinir les touches de direction utilisées par la routine assembleur, grâce à la directive .PUBLIC qui montre toutes les qualités de l'assembleur (ou plutôt de l'éditeur de liens UCSD).

Toutes les explications nécessaires sont données par le programme Pascal. Par contre, je n'ai pas eu le temps (ou le courage) de documenter la procédure externe. De toute façon, il serait sans doute plus astucieux de faire appel aux routines de Turtlegraphics (si on savait où elles se trouvent) ou d'écrire sa propre routine de HPLOT, plutôt que de faire appel à la ROM Applesoft comme je le fais. Enfin, comme le disait Galilée, ça tourne ...



ORD1 est le programme Pascal d'ORDRALPHABETIX et ORD2 le programme assembleur.

■

```
PROGRAM ORDRALPHABETIX;

{* ECRIT PAR DOMINIQUE BERNARDI, SUR UNE
   IDEE DE JEAN-LOUIS MOISY ET HELENE PAIGNAM *}

USES TURTLEGRAPHICS, APPLESTUFF;

TYPE POINT=RECORD Y,X:INTEGER END;
     CHOIXDECA=SET OF CHAR;

VAR  TABLE:ARRAY[0..26] OF POINT;
     PT,DEPART:POINT;
     LETTRES,VITESSE,SCOREMAX,BUT,CRASH,
     DELTAX,DELTAY,TEMPORISATION:INTEGER;
     HAUT,BAS,GAUCHE,DROITE,A:CHAR;

PROCEDURE COURS; EXTERNAL;

FUNCTION PCAR(S:CHOIXDECA):CHAR;
 VAR A:CHAR;
 BEGIN
  REPEAT READ(A) UNTIL A IN S;
  PCAR:=A
 END;

FUNCTION OUI:BOOLEAN;
 BEGIN
  OUI:=PCAR(['O','N']) ='O'
 END;

FUNCTION ENTIER:INTEGER;
 VAR A:CHAR;
     I:INTEGER;
 BEGIN
  A:=PCAR(['0'..'9']);
  I:=ORD(A)-ORD('0');
  A:=PCAR(['0'..'9',' ']);
  IF A<>' ' THEN I:=10*I+ORD(A)-ORD('0');
  ENTIER:=I
 END;

PROCEDURE PARAMETRES;
 VAR I,J:INTEGER;
 BEGIN
  REPEAT
   PAGE(OUTPUT);
   WRITELN('              PARAMETRES');
   WRITELN; WRITELN;
   WRITELN('ACTUELLEMENT: ',LETTRES,' LETTRES, VITESSE ',VITESSE);
   WRITELN; WRITELN;
   WRITE('LETTRES (1..26) -->');
   REPEAT I:=ENTIER UNTIL I IN [1..26];
   WRITELN;
   WRITE('VITESSE (1..20) -->');
   REPEAT J:=ENTIER UNTIL J IN [1..20];
   WRITELN; WRITELN;
   WRITELN(I,' LETTRES, VITESSE ',J);
   WRITELN;
   WRITE('OK ? ')
  UNTIL OUI;
  LETTRES:=I; VITESSE:=J
 END;

PROCEDURE TOUCHES;
 VAR A1,A2,A3,A4:CHAR;
 BEGIN
  REPEAT
   PAGE(OUTPUT);
   WRITELN('        TOUCHES DIRECTIONNELLES');
   WRITELN; WRITELN;
   WRITE('HAUT --> '); READ(A1);
   WRITELN;
   WRITE('BAS --> '); READ(A2);
   WRITELN;
   WRITE('GAUCHE  > '); READ(A3);
   WRITELN;
   WRITE('DROITE --> '); READ(A4);
   WRITELN; WRITELN;
   WRITE('OK ? ')
  UNTIL OUI;
  HAUT:=A1; BAS:=A2; GAUCHE:=A3; DROITE:=A4
 END;

PROCEDURE CHOISIT;
 VAR PT:POINT;
     I:INTEGER;
 PROCEDURE HASARD;
  BEGIN
   WITH PT DO BEGIN
    X:=4+(RANDOM MOD 267);
    Y:=10+(RANDOM MOD 178)
   END
  END;
 FUNCTION POSSIBLE:BOOLEAN;
  VAR J:INTEGER;
  BEGIN
   J:=1;
   WHILE J<I DO BEGIN
     IF (ABS(TABLE[J].X-PT.X)<5.5 )
      AND (ABS(TABLE[J].Y-PT.Y)<6.5)
      THEN BEGIN POSSIBLE:=FALSE; EXIT(POSSIBLE) END;
     J:=J+1
    END;
   POSSIBLE:=TRUE
  END;
 BEGIN (*CHOISIT*)
  FOR I:=1 TO LETTRES DO
   BEGIN
    REPEAT HASARD UNTIL POSSIBLE;
    TABLE[I]:=PT
   END;
  REPEAT HASARD UNTIL POSSIBLE;
  DEPART:=PT
 END;

PROCEDURE TRACE(I:INTEGER);
 BEGIN
  MOVETO(TABLE[I].X,190-TABLE[I].Y);
  WCHAR(CHR(64+I));
 END;

PROCEDURE DESSINE;
 VAR I:INTEGER;
 BEGIN
  INITTURTLE;
  VIEWPORT(4,275,4,187);
  FILLSCREEN(WHITE);
  VIEWPORT(0,279,0,191);
  FILLSCREEN(REVERSE);
  FOR I:=1 TO LETTRES DO TRACE(I);
 END;
```

```
PROCEDURE JEU;
 VAR SCORE:INTEGER;
     FINI:BOOLEAN;
     A:CHAR;
 BEGIN
  DESSINE;
  SCORE:=0;
  BUT:=1;
  CRASH:=0;
  DELTAX:=0;
  DELTAY:=0;
  PT:=DEPART;
  TEMPORISATION:=241-12*VITESSE;
  REPEAT
   COURS;
   IF CRASH = 0
    THEN BEGIN
           WRITE(CHR(7));
           SCORE:=SCORE+VITESSE;
           TRACE(BUT);
           BUT:=BUT+1;
           FINI:=BUT>LETTRES
         END
    ELSE BEGIN
           NOTE(4,100);
           FINI:=TRUE;
         END
  UNTIL FINI;
  IF SCORE>SCOREMAX THEN SCOREMAX:=SCORE;
  READ(A);
  PAGE(OUTPUT);
  TEXTMODE;
  WRITELN('VOTRE SCORE EST DE ',SCORE,' POINTS');
  WRITELN('LE SCORE MAXIMUM ATTEINT AUJOURD''HUI');
  WRITELN('EST DE ',SCOREMAX,' POINTS');
  WRITELN;
  READ(A)
 END;

PROCEDURE INIT;
 BEGIN
  RANDOMIZE;
  LETTRES:=26;
  VITESSE:=10;
  SCOREMAX:=0;
  CHOISIT;
  HAUT:='I';
  BAS:='M';
  GAUCHE:='J';
  DROITE:='K';
  CHARTYPE(6)
 END;
```

```
PROCEDURE INSTRUCTIONS;
 VAR A:CHAR;
 BEGIN
  PAGE(OUTPUT);
  WRITELN('              INSTRUCTIONS');
  WRITELN; WRITELN;
  WRITELN('VOUS DEVEZ RAYER DANS L''ORDRE LES');
  WRITELN('LETTRES DE L''ALPHABET, EN DIRIGEANT LE');
  WRITELN('TRAIT GRACE AUX TOUCHES DE DIRECTION QUI');
  WRITELN('SONT, PAR DEFAUT, <I> <J> <K> ET <M>.');
  WRITELN;WRITELN;
  WRITELN('ATTENTION, UNE FOIS QU''IL A DEMARRE, LE');
  WRITELN('TRAIT NE S''ARRETE QU''A LA VICTOIRE');
  WRITELN('FINALE OU A LA COLLISION AVEC LE BORD DU');
  WRITELN('LE RECTANGLE 6X7 CONTENANT UNE LETTRE');
  WRITELN('ERRONEE.');
  WRITELN;
  WRITELN; READ(A)
 END;

PROCEDURE MENU;
 BEGIN
  PAGE(OUTPUT);
  WRITELN('           ORDRALPHABETIX');
  WRITELN; WRITELN;
  WRITELN('I)NSTRUCTIONS');
  WRITELN('T)OUCHES DE DIRECTION');
  WRITELN('P)ARAMETRES DU JEU');
  WRITELN('D)ISPOSITION DES LETTRES');
  WRITELN('J)EU');
  WRITELN('F)IN');
  WRITELN; WRITELN;
  WRITE('VOTRE CHOIX --> ')
 END;

BEGIN (* ORDRALPHABETIX *)
 INIT;
 REPEAT
  MENU;
  A:=POS(['I','T','P','D','J','F']);
  CASE A OF
   'I':INSTRUCTIONS;
   'T':TOUCHES;
   'P':PARAMETRES;
   'D':CHOISIT;
   'J':JEU
  END
 UNTIL A='F';
 PAGE(OUTPUT);
 GOTOXY(14,12);
 WRITELN('AU REVOIR')
END.
```

Programme ORD2

```
.PROC COURS
.PUBLIC TABLE,LETTRES,BUT
.PUBLIC TEMPORISATION,CRASH
.PUBLIC DELTAX,DELTAY,PT
.PUBLIC HAUT,BAS,GAUCHE,DROITE

STOCK     .EQU 0
HIRES     .EQU 0E0
COLOR     .EQU 0E4
PAGE      .EQU 0E6
KEYBOARD  .EQU 0C000
KEYSTROB  .EQU 0C010
PLOT      .EQU 0F457
RAM       .EQU 0C088
ROM       .EQU 0C08A

INTRO     LDY #0F
$0        LDA HIRES,Y
          STA BLOC,Y
          DEY
          BPL $0
          LDA #7F
          STA COLOR
          LDA #20
          STA PAGE
          STA ROM

ALLUME    LDY PT+1
          LDX PT
          LDA PT+2
          JSR PLOT
          JSR DELAI

          LDA KEYBOARD
          BPL AVANCE
          STA KEYSTROB
          LDX #0
          AND #7F
          CMP HAUT
          BEQ MONTE
          CMP BAS
          BEQ DESCEND
          CMP GAUCHE
          BEQ AGAUCHE
          CMP DROITE
          BEQ ADROITE
          JMP AVANCE

MONTE     STX DELTAX
          STX DELTAX+1
          LDA #0FF
          STA DELTAY
          JMP AVANCE

DESCEND   STX DELTAX
          STX DELTAX+1
          LDA #1
          STA DELTAY
          JMP AVANCE

AGAUCHE   STX DELTAY
          LDA #0FF
          STA DELTAX
          STA DELTAX+1
          JMP AVANCE

ADROITE   STX DELTAY
          STX DELTAX+1
          LDA #1
          STA DELTAX

AVANCE    CLC
          LDA PT
          ADC DELTAX
          STA PT
          LDA PT+1
          ADC DELTAX+1
          STA PT+1
          CLC
          LDA PT+2
          ADC DELTAY
          STA PT+2

CHECKX    LDA PT+1
          BEQ CHECKG
          LDA PT
          CMP #20.
          BCS PERDU
          JMP CHECKY

CHECKG    LDA PT
          CMP #4
          BCC PERDU

CHECKY    LDA PT+2
          CMP #188.
          BCS PERDU
          CMP #4
          BCC PERDU

PILE      LDA BUT
          ASL A
          ASL A
          TAX
          JSR CARRE
          BEQ CORRECT

DEPILE    CLC
          TXA
          ADC #4
          TAX
          LSR A
          LSR A
          CMP LETTRES
          BEQ $0
          BCC $0
          JMP ALLUME
$0        JSR CARRE
          BEQ PERDU
          JMP DEPILE

CORRECT   LDA #0
          STA CRASH
          JMP EXIT

PERDU     LDA #1
          STA CRASH
          JMP EXIT

CARRE     SEC
          LDA PT
          SBC TABLE,X
          STA STOCK
          LDA PT+1
          SBC TABLE+1,X
          BNE NON
          LDA STOCK
          CMP #6
          BCS NON
          SEC
          LDA TABLE+2,X
          SBC PT+2
          CMP #7
          BCS NON
OUI       LDA #0
          RTS
NON       LDA #1
          RTS

DELAI     LDY TEMPORISATION
$1        DEY
          BEQ FIN
          LDX #10.
$2        DEX
          BNE $2
          JMP $1
FIN       RTS

EXIT      STA RAM
          LDY #0F
$3        LDA BLOC,Y
          STA HIRES,Y
          DEY
          BPL $3
          RTS
BLOC      .BLOCK 10

          .END
```

ProDOS à l'essai

Alexandre Avrane

Avec l'arrivée du Macintosh, Apple a annoncé la création de la famille "Apple 32", regroupant les matériels tournant sur processeur 32 bits. Etrangement, peu de mots sur la gamme "Apple 8" ont été prononcés, comme si les Apple II (Plus, //e et //c) et /// devaient éternellement continuer à s'ignorer et rester quasiment incompatibles l'un avec l'autre. Heureusement, voici ProDOS.

Rendu possible par l'abaissement du coût de la mémoire, ProDOS est le nouveau système d'exploitation destiné à remplacer, à terme, le DOS 3.3 et éventuellement le S.O.S., et donc à offrir enfin une certaine compatibilité entre la famille Apple II et l'Apple ///.

L'évaluation qui va suivre est divisée en trois grandes parties volontairement succintes :

- une description générale des caractéristiques du ProDOS;
- l'étude de ses commandes et la comparaison à celles du DOS 3.3;
- un premier regard sur son fonctionnement interne.

Caractéristiques du ProDOS

ProDOS a un double objectif : rendre possible l'utilisation d'un disque dur sur Apple II de manière aussi simple que celle d'une disquette, et permettre à l'Apple /// d'utiliser les logiciels de l'Apple II sans devoir passer en mode émulation. En bref, les fichiers de l'un peuvent être lus par l'autre.

ProDOS ne nécessite aucune modification hardware (pas d'ajout de PROM comme lors du passage DOS 3.2 vers DOS 3.3), mais requiert un minimum de 64K de RAM (le DOS 3.3 tient en 16K). La quasi-totalité de la carte langage lui est réservée; nous avons par conséquent la douleur de vous faire part du décès, à l'âge de 8 ans, du Basic Integer qui n'est pas supporté par ProDOS...

Des disques durs d'une capacité maximum de 32 mega-octets (Mo) peuvent être utilisés (le ProFile offre 5 Mo); un fichier individuel peut enfin avoir une taille de 16 Mo, soit l'équivalent de 114 disquettes !

C'est énorme ! Pour pouvoir s'y retrouver rapidement, ProDOS utilise le principe des catalogues hiérarchisés : un catalogue général fournit une liste de sous-catalogues qui, eux-mêmes, peuvent contenir des sous-sous-catalogues, et ainsi de suite. Relativement simple à utiliser, cette méthode permet de structurer le contenu d'un disque, mais pourra dérouter l'utilisateur habitué à des structures plus simples. Sans cette structure hiérarchique, il faudrait, si chaque fichier avait une taille de 10 K, 21 pages d'écran pour lire le catalogue d'un ProFile !

ProDOS gère automatiquement tous les sous-catalogues et, en particulier, les dates de création et de dernière modification d'un fichier, sa longueur en octets et son adresse de chargement. Pour l'instant, il reconnaît seulement la carte d'horloge Thunderclock, mais les autres fabricants vont certainement proposer dans peu de temps des patches pour leurs matériels.

Un des principaux points forts de ProDOS réside dans sa gestion beaucoup plus rapide des fichiers, par rapport non seulement à celle du DOS 3.3, mais également à l'égard des versions "dopées" telles que ZDOS, ProntoDOS, Diversi-DOS, etc. J'ai obtenu les temps suivants (en secondes) avec un fichier binaire de 32 K :

	DOS 3.3	ZDOS 2.0	ProDOS 1.0
BSAVE	46	39	18
BLOAD	33	8	5

Enfin, ProDOS est livré avec plusieurs utilitaires équivalents à FID et COPYA, ainsi qu'avec un programme de conversion des fichiers séquentiels DOS 3.3 : les fichiers à accès direct doivent d'abord être transposés en fichiers séquentiels, convertis en ProDOS, puis retransposés en fichiers à accès direct ... Call Apple a publié en avril 1984 un programme effectuant cette double transposition.

En conclusion partielle, ProDOS est le quatrième grand système d'exploitation pour Apple II disponible en France. Hormis le DOS 3.3, il va devoir compter avec le Pascal, le CP/M, et le MEM/DOS à qui il pourrait faire du tort.

Les commandes du ProDOS

Pour des raisons de place, cette étude se borne à indiquer les différences entre les commandes du ProDOS et celles du DOS 3.3, et nécessite donc une connaissance pratique de ce dernier. Les commandes du ProDOS sont, en effet, volontairement similaires à celles du DOS 3.3; il n'y a donc pas nécessité d'apprendre une nouvelle gamme de commandes, et l'utilisateur rôdé au DOS 3.3 s'accoutumera très rapidement à ce nouveau système d'exploitation.

Sept types de fichiers existent sous ProDOS : les fichiers de programmes Basic, textes, et binaires nous sont déjà familiers. S'y ajoutent les fichiers systèmes (ProDOS lui-même constitue un fichier, contrairement au DOS 3.3 qui se trouve toujours sur les 3 premières pistes d'une disquette), les fichiers binaires relogeables (l'équivalent des fichiers R du DOS ToolKit), les fichiers de variables (qui contiennent les variables d'un programme Applesoft), et enfin les sous-catalogues (considérés comme des fichiers à part entière et pouvant donc être lus ou écrits). En outre, l'utilisateur a la possibilité de définir 8 types supplémentaires de fichiers.

Hiérarchie des catalogues

La hiérarchisation des catalogues est facilitée par le PREFIX : supposons que sur un disque, baptisé DISQUE1 lors de son initialisation, existent des fichiers de gestion, de jeux, des utilitaires, etc. On peut créer un sous-catalogue nommé GESTION contenant tous les fichiers de gestion, puis un sous-sous-catalogue baptisé IMPOT qui contient plusieurs fichiers, mettons un par année. Pour appeler un de ces fichiers, il faut une commande telle que, par exemple :

```
OPEN DISQUE1/GESTION/IMPOT/ANNEE1
```

Remarquez que l'on ne précise ni le slot ni le drive, ProDOS identifiant (comme Pascal) le nom logique du volume (DISQUE1) avec son adresse physique (slot, drive). Les "/" (prononcez slash) servent de délimiteurs entre les identifiants qui doivent comporter au plus 15 caractères (moins que sur DOS 3.3, mais plus que sur Pascal ou CP/M, et franchement mieux que les 8 caractères reconnus par l'IBM PC ...) pris parmi les lettres, les chiffres ou le point. La longueur totale de la commande, hors l'instruction OPEN, ne peut excéder 64 caractères. Pour alléger la syntaxe, il est possible de préciser un préfixe par défaut avec l'instruction :

```
PREFIX /DISQUE1/GESTION/IMPOT/
```

et la commande ci-dessus devient plus simplement :

```
OPEN ANNEE1
```

Les commandes disparues

Six commandes du DOS 3.3 ont disparu : INIT (il faudra charger et exécuter un programme de formatage), INT et FP (le Basic Integer n'est pas supporté). De plus, MAXFILES (ProDOS gère directement les allocations mémoire et tolère 8 fichiers ouverts simultanément), MON et NOMON (en contrepartie, la commande TRACE de l'Applesoft fonctionne correctement) sont purement et simplement ignorés par ProDOS.

Les commandes nouvelles ou modifiées

Plusieurs instructions concernent la manipulation des catalogues :

CATALOG garde son sens habituel et affiche les fichiers dépendant du préfixe donné, avec indication de la date de dernière modification et des capacités libre et occupée du volume : les tailles sont exprimées en blocs correspondant à 2 secteurs, soit 512 octets. En effet, ProDOS est (comme le SOS) de filiation Pascal, ce qui explique la coïncidence de la taille de bloc.
CAT fournit une version plus complète du catalogue, sur 80 colonnes.
PREFIX, déjà mentionné, permet de modifier le préfixe.
CREATE permet de créer un fichier catalogue.
– : cette nouvelle commande très pratique (prononcez "dash") peut être utilisée à la place de RUN, BRUN ou EXEC.
FLUSH équivaut à un CLOSE, mais laisse le fichier ouvert.
FRE est une commande de nettoyage mémoire beaucoup plus rapide que celle de l'Applesoft.
STORE et *RESTORE* permettent respectivement de stocker sur disque et de rappeler les variables d'un programme Applesoft.

Enfin, ProDOS autorise l'adjonction de nouvelles commandes à son vocabulaire, à la différence du DOS 3.3 où il fallait pour cela savoir jongler en assembleur.

Les commandes améliorées

Les autres instructions de ProDOS sont semblables à celles du DOS 3.3, mais comportent de très nombreuses améliorations qu'il serait trop long de détailler ici; ainsi, il devient possible de préciser l'adresse de fin d'un programme binaire (BSAVE ECRAN, A$2000,E$3FFF), ou de charger un programme Applesoft puis de l'exécuter à partir d'une ligne précise (RUN PROGRAMME, @200).

Les instructions APPEND et CHAIN fonctionnent correctement sous ProDOS. Enfin, les commandes IN# et PR# peuvent préciser directement l'adresse d'une routine d'entrée ou de sortie.

Conversion des programmes Applesoft

La programmation sous ProDOS est donc sensiblement la même que sous DOS 3.3 et la conversion de programmes en Applesoft ne devrait pas poser de problèmes majeurs.

Il existe néanmoins une différence fondamentale dans l'interception des commandes entre le DOS 3.3 et ProDOS. Le DOS 3.3 contrôle si, au début d'une ligne, on veut imprimer le caractère Ctrl-D. ProDOS, pour sa part, contrôle chaque instruction à travers la routine TRACE; lorsqu'il rencontre une instruction PRINT, il vérifie si le premier caractère est un Ctrl-D. Conclusion : l'habitude de certains (bons) programmeurs de faire précéder un Ctrl-D par un retour chariot, en définissant D$ = CHR$(13) + CHR$(4), est maintenant à bannir. Deuxième implication : cette méthode d'interception ralentit quelque peu l'exécution d'un programme (environ 5 à 8%) mais, en contrepartie, le nettoyage de la mémoire et la gestion des fichiers sont beaucoup plus rapides avec ProDOS.

Fonctionnement interne

Cette dernière partie s'adresse aux habitués de l'assembleur; il n'est pas question d'indiquer dans cette première présentation l'organisation détaillée des routines en mémoire ou des pointeurs physiques sur disque de ProDOS, mais plutôt de fournir un aperçu de son fonctionnement.

Historiquement, on divise le DOS 3.3 en trois modules distincts : le "Captain" interprète les commandes, le "File Manager" gère l'allocation des fichiers sur une disquette, et "RWTS", enfin, assure la lecture, l'écriture ou le formatage d'un secteur.

RWTS est sensiblement identique sous ProDOS, sauf qu'il est incapable de formater un disque et a été rebaptisé "Disk Driver". C'est d'ailleurs la même routine qui est utilisée sous Pascal et CP/M; seul son équivalent sous DOS 3.2, avec ses pistes de 13 secteurs, diffère. Bien évidemment, le Disk Driver est un module différent lors de l'emploi d'un disque dur.

ProDOS raisonne en blocs, chaque bloc étant une unité logique de 2 secteurs physiques. Il y a donc 35 * 16 / 2 = 280 blocs sur une disquette normale.

A l'inverse du DOS 3.3 qui réside intégralement en mémoire, ProDOS se compose de deux parties distinctes, la deuxième étant chargée en fonction des besoins.

Noyau de ProDOS : le Kernel

Le ProDOS Kernel contient l'équivalent du File Manager et de RWTS; il occupe l'ensemble de la carte langage (à l'exception de la zone $D000 $D0FF de la Bank 2), ainsi qu'une petite zone ($BF00-$BFFF) destinée à "switcher" entre les différentes parties hautes de la mémoire et à recevoir des appels de programmes externes.
Cet ensemble est également appelé M.L.I. (pour Machine Language Interface).

Il peut être appelé directement par un programme en assembleur. Son équivalent sous DOS 3.3 serait un point d'entrée unique pour l'appel de RWTS et du File Manager, avec une table des paramètres également unique. La syntaxe d'appel est :

```
JSR MLI ;MLI=$BF00
DFB CMDNUM ;type de commande
    de fichiers (sauf Basic)
DA PARAM ;adresse de la table des
    paramètres
BNE ERREUR ;en retour, accumula-
    teur=type d'erreur
```

Les commandes disponibles concernent la lecture ou l'écriture d'un bloc ou d'un fichier texte, le préfixe, etc. Deux utilitaires livrés avec ProDOS facilitent l'apprentissage : l'EXERCISER permet d'étudier l'exécution des appels et BUGBYTER (voir l'analyse dans ce numéro) permet de "debugger" un programme assembleur.

Interface avec l'Applesoft : BASIC.SYSTEM

Le noyau de ProDOS permet de gérer tous les fichiers qui ne sont pas des programmes Applesoft. En effet, pour utiliser ces derniers, ainsi que l'interpréteur de commandes, il est nécessaire de charger une deuxième partie de ProDOS, qui porte le nom BASIC.SYSTEM et occupe les adresses $9600-$BEFF.

Notons que HIMEM se trouve alors à la même position qu'avec le DOS 3.3; de toute façon, il est préférable avec ProDOS de ne pas mettre d'instruction HIMEM dans un programme Applesoft (et surtout pas un HIMEM qui ne correspondrait pas à une limite de page mémoire), ProDOS gérant lui-même l'allocation de la RAM par sections de 256 octets.

Oubliez HELLO, apprenez STARTUP. Lors du boot d'une disquette, ProDOS n'exécute pas automatiquement un programme Applesoft précisé lors de son initialisation, mais recherche dans le catalogue principal un fichier nommé STARTUP. S'il le trouve, celui-ci est exécuté quelque soit son type (Applesoft, assembleur,

fichier Exec); il est de la responsabilité de ce programme de charger ou non le BASIC.SYSTEM.

Certaines adresses de page zéro sont utilisées par ProDOS : $3A-$3F par RWTS; $40-$4E par d'autres routines qui les rétablissent avant de rendre la main. Les vecteurs de la page 3 sont également présents sous ProDOS.

Format d'une disquette

Rappelez-vous que, sous ProDOS, il faut penser en blocs de 512 octets, et non plus en secteurs de 256 octets. Les 280 blocs d'une disquette sont numérotés dans l'ordre croissant des pistes : 0 à 7 sur la piste 0, 8 à 15 sur la piste 1, etc.

Les deux premiers blocs contiennent le "Loader", c'est-à-dire le programme, chargé et exécuté par la routine du contrôleur de la disquette, qui lui-même charge le ProDOS Kernel.

Les blocs 2 à 5 contiennent le catalogue principal de la disquette (on dit aussi table des matières), à raison de 13 fichiers par bloc (d'où une limite de 52 fichiers dans le catalogue principal, étant entendu que certains des fichiers peuvent être des sous-catalogues).

Le bloc 6 correspond à la carte du volume ("Volume Bit Map") qui fonctionne comme la VTOC du DOS 3.3; chaque bit du bloc indique si l'un des blocs du volume est libre ou occupé. Sur disque dur, il peut exister jusqu'à 16 blocs formant la carte du volume et représentant 16 * 512 * 8 =65536 blocs donc 33.554.432 octets ! Sur une disquette, seuls les 35 premiers octets, représentant les 280 blocs, sont utilisés.

Sous DOS 3.3, un fichier utilisait systématiquement au moins un secteur, appelé TSL ("Track Sector List"), qui donnait les adresses des secteurs de données de ce fichier. Une philosophie moins gourmande en place a été adoptée avec ProDOS :

- un fichier de moins de 512 octets (1 bloc) est directement accessible;
- un fichier d'une taille comprise entre 2 et 256 blocs nécessite un bloc d'index;
- un fichier d'une taille supérieure utilise un bloc d'index principal qui pointe vers au plus 128 blocs d'index secondaires, eux-mêmes pointant vers les blocs de données (ouf!).

Il n'est pas question ici d'examiner individuellement la valeur des compteurs et pointeurs de ces blocs systèmes (il faudrait y consacrer un article entier). Néanmoins ceux-ci devront être étudiés plus attentivement par ceux qui souhaitent développer des utilitaires : il faudra en particulier analyser les commandes UNDELETE ("restoration" d'un fichier DELETé) et VERIFY (similaire à celle du DOS 3.3; ProDOS ne vérifie pas l'intégrité physique d'un fichier, mais uniquement son existence).

Conclusion

ProDOS offre incontestablement des améliorations considérables par rapport au DOS 3.3, même si on reste un peu sur sa faim (pas de gestion automatique des fichiers séquentiels indexés, que l'on était pourtant en droit d'attendre). Apple livre maintenant les //e et //c uniquement avec ProDOS, et souhaite en faire le nouveau standard. Il ne faut cependant pas espérer (craindre ?) la disparition prochaine du DOS 3.3, d'une part car ProDOS est très gourmand en mémoire (il occupe trois fois plus de place), d'autre part car plus de 10.000 logiciels pour Apple tournent sur DOS 3.3 (soit dit en passant, plus que CP/M) : à ma connaissance, des programmes vedettes tels que VisiCalc ou Apple Writer ne supportent pas (encore) ProDOS.

■

La compatibilité de l'Apple //c

Guy Lapautre

Des bancs d'essai de l'Apple //c figurent dans toutes les revues (Pom's a publié 2 "points de vue" dans son précédent numéro). Des démonstrations sont faites dans toutes les boutiques. Le constructeur propose une documentation bien fournie. Aussi avons-nous choisi d'aborder ici l'Apple //c sous un angle un peu particulier.

Vous êtes un familier de l'Apple //e.
Vous possédez de nombreux logiciels du commerce (et d'autres que vous avez écrits).
Vous disposez de périphériques classiques : lecteurs de disque, moniteur, imprimante (par exemple la Dot Matrix Printer d'Apple).
Vous êtes également accoutumé à utiliser des périphériques moins classiques : téléviseur couleur relié à une carte RGB, poignées de jeu.
Vous travaillez couramment sur écran 80 colonnes.
Vous avez également une carte avec modem intégré, qui vous simplifie bien la vie.

A quoi pouvez-vous vous attendre si vous décidez de "passer à l'Apple //c" ? C'est à cette question, à notre sens fondamentale, que nous allons essayer de répondre.

Le premier contact

La surprise du poids et de la taille. Vous vous y attendiez bien sûr, mais cette petite boîte au design agréable surprend toujours. Autre surprise, moins agréable celle-là : l'alimentation, séparée, relativement lourde (1,4 kg environ), qui ne tient pas dans la sacoche. C'est déjà moins portable...

Il y a aussi le problème de l'écran. Faisons confiance à Apple, qui annonce pour la fin de l'année un écran plat rabattable de 24 lignes par 80 colonnes.

Soulevons le couvercle... Non, mauvaise habitude ! On n'ouvre pas l'Apple //c. C'est un système de type fermé, et, en principe, vous pouvez tout faire sans l'aide des fameux connecteurs d'extension de l'Apple //e. Habituez-vous à brancher directement vos périphériques sur la batterie de connecteurs spécialisés, à l'arrière de l'appareil.

Branchons. C'est simple, avec le lecteur intégré et les connecteurs externes. Petite désillusion, si vous n'y aviez pas fait attention, ni votre Dot Matrix, ni vos lecteurs de disques de l'Apple //e ne sont utilisables.
La Dot Matrix est une imprimante parallèle, la sortie imprimante de l'Apple //c est une "série". Il va vous falloir acheter une autre imprimante, l'Image Writer d'Apple (la même que pour Mac Intosh) par exemple.
Le contrôleur de disques n'a pas les mêmes caractéristiques que celui de l'Apple //e.

Allons un peu plus loin dans la découverte

Votre carte RGB ne trouve évidemment pas sa place, puisqu'il n'y a pas de connecteurs d'extension. Mais

vous trouverez, en série, un adaptateur RGB fonctionnant sur prise Péritel. Pas de difficulté de ce côté. Même le son est renvoyé sur votre téléviseur, ce qui peut donner des effets assez spectaculaires sur certains jeux. Si vous possédez la disquette Pom's No 11, essayez-la, vous serez surpris de la relative netteté de la voix synthétique. Quant au fonctionnement sur moniteur noir et blanc, pas de problème. Vous utilisez le même cordon de raccordement.

A propos de moniteur, la position de notre constructeur préféré semble être du style "faites ce que je dis, ne faites pas ce que je fais". Vous trouvez dans le manuel d'installation : "ATTENTION - ne posez pas le téléviseur (ou quoi que ce soit d'autre) sur l'ordinateur, ce serait trop lourd et bloquerait les ouvertures de ventilation". Et la couverture du manuel Utilitaires Système représente... un Apple //c avec un moniteur qui semble posé dessus (certes incliné pour dégager les ouvertures, mais quand même...). En fait, il s'agit du nouveau moniteur Apple, avec son support "étudié pour".

Vos manettes de jeu ou votre joystick seront utilisés sans difficulté SI vous avez une version se branchant sur le connecteur arrière de votre Apple //e : la prise est identique.

Pas de place non plus pour votre carte 80 colonnes, mais c'est un dispositif qui se trouve en série sur l'Apple //c. Avec commutation manuelle par poussoir sur l'ordinateur ou reconnaissance directe par le programme.

Mais vous allez tomber de haut en abordant le chapitre des communications : bien sûr, votre carte modem n'a pas sa place, mais de plus, RIEN d'analogue ne peut la remplacer. Vous voici revenu au Moyen Age, au temps du modem acoustique par exemple. Pour un portable...

Le clavier

Doux, agréable à manier, il vous réservera quelques surprises.

Fini le double clavier AZERTY/QWERTY de l'Apple //e, si discuté. Vous avez un AZERTY normalement constitué. Les touches sont disposées EXACTEMENT de la même façon que sur le modèle //e, à l'exception de la touche RESET.

Une originalité en ce qui concerne le fonctionnement majuscules/minuscules. Sur l'Apple //e, la touche "Caps Lock" ne verrouillait que les 26 lettres de l'alphabet, mais ni les chiffres ni les signes, ce qui est particulièrement énervant quand on doit frapper beaucoup de valeurs numériques (ce n'est pas rare en informatique...). Utiliser par exemple un tableur en AZERTY relève plus du numéro de haute voltige que du modèle de convivialité ! Ici, "Caps Lock" verrouille tout, ce qui est très intéressant mais présente néanmoins des inconvénients, si on veut cette fois utiliser couramment "-" "=" ":" ou "," par exemple. Solution adoptée : les touches majuscules fonctionnent comme des inverseurs. Si Caps Lock est enfoncée, elles font passer en minuscules. Sinon, elles font passer en majuscules. Il faut un peu de temps pour s'y habituer, mais c'est finalement assez commode.

NDLR : Cela vous posera des problèmes cependant si vous utilisez aussi un IBM PC/XT, qui se comporte différemment.

Autre originalité : vous pouvez maintenant taper les commandes, ainsi que les instructions de vos programmes Applesoft, en minuscules aussi bien qu'en majuscules : la conversion est faite automatiquement en majuscules. Vous n'avez plus à craindre le méchant "SYNTAX ERROR" si vous tapez "catalog" au lieu de "CATALOG".

Et pour les fanatiques du QWERTY ? Ils ne sont pas totalement oubliés car une touche permet de simuler un clavier QWERTY sur le clavier AZERTY. Mais ce n'est qu'une roue de secours : la signification QWERTY des touches n'étant pas indiquée sur celles-ci, il faut taper en aveugle. Tout le monde ne peut pas le faire.

Systèmes d'exploitation et langages

Le système d'exploitation standard de l'Apple //c est PRODOS, qui peut aussi équiper l'Apple //e. De même, l'Apple //c accepte le DOS 3.3 (mais pas les versions antérieures du DOS. Vous devrez convertir, ce qui est simple).

Vous pouvez aussi travailler en Pascal, en SUPER-PILOT ou en un LOGO nouveau, permettant une utilisation optimale de la mémoire centrale. Quelques frais à prévoir donc (après essai sommaire, il semble que Edi-Logo, dans sa version écrite pour le //e, fonctionne sur le //c).

Mais - et c'est là où le bât blessera de très nombreux utilisateurs - pas question de Soft Card, donc de CP/M. Finis les logiciels prestigieux comme dBASE II (que rien d'approchant ne vient remplacer), ou WORDSTAR. Bons à jeter aussi vos propres logiciels en Basic Microsoft. Même "punition" pour les adeptes de MEM/DOS, au moment où les auteurs de ce système annoncent précisément plusieurs nouveautés.

L'Apple //c est un peu un ordinateur pour père de famille peu soucieux d'innovation et d'utilisation non standard. A ce titre, certains le considèreront comme une régression par rapport au modèle //e, malgré ses nombreux progrès technologiques et sa capacité de mémoire étendue. C'est sans doute pour cela que, paraît-il, le constructeur vise pour une bonne part le marché du "premier ordinateur".

C'est aussi pourquoi nous n'avons pas testé des logiciels très particuliers - dont sont souvent friands les lecteurs de Pom's - tels que HAIFA, PLE ou la gestion de masques, par exemple.
Bien entendu, nous n'avons pas testé non plus les utilitaires classiques tels que copie, renumérotation, fusion de programmes, ... mais pour une autre raison : une version Apple //c accompagne la machine (en ProDOS, sur la disquette Utilitaires Système).

Les langages utilisés dans les programmes testés se répartissent entre Applesoft, Pascal UCSD, assembleur et même Basic Entier, bien que ce dernier soit fortement tombé en désuétude.

Vos programmes

Selon le constructeur, "90 à 95 % des programmes écrits pour Apple II ou Apple //e sont compatibles Apple //c". Si l'on excepte les logiciels fonctionnant sous un système d'exploitation non supporté, cela paraît vrai quantitativement. Nous avons fait un certain nombre d'essais et aucun logiciel essayé ne s'est avéré totalement impropre à fonctionner sur //c, mais certains logiciels de premier plan sont pratiquement inutilisables.

Nous aurons l'occasion d'examiner un certain nombre de dysfonctionnements, cas par cas.

Par ailleurs, nous avons limité nos tests à des logiciels n'exigeant qu'un seul lecteur de disques (ce qui élimine par exemple Business Graphics ou Think Tank).
Enfin, ne perdons jamais de vue que les logiciels testés sont des logiciels écrits pour l'Apple //e. D'ores et déjà, de nombreux auteurs proposent des logiciels nouveaux, ou des versions nouvelles, spécialement écrits pour le //c.

Nous n'avons rencontré aucune difficulté tenant à la reconnaissance de la "carte" 80 colonnes, ni au niveau de l'adaptateur RGB. Relativement à ce dernier point, nous ne notons pas d'amélioration par rapport aux résultats obtenus avec la carte "Chat Mauve" sur Apple //e pour ce qui est de la qualité du texte sur écran graphique. C'est toujours très mauvais,

sauf pour les logiciels dont les auteurs ont eu la bonne idée de choisir des solutions d'écriture monochrome.

Les résultats des tests qui suivent ne visent nullement à constituer des bancs d'essai des logiciels correspondants sur Apple //c. Notre objectif était de vérifier si ces logiciels étaient acceptés, et si des dysfonctionnements évidents se produisaient. Il est fort possible que certaines options posent des problèmes que nous n'avons pas rencontrés au cours de ce bref tour d'horizon.

Les tableurs

L'éternel Visicalc subit l'épreuve avec succès, dans sa version 40 colonnes. Il en est de même pour Magicalc, y compris dans son option 80 colonnes.

Multiplan en revanche se tire moins bien d'affaire. Certes, les options que nous avons testées fonctionnent correctement. Mais il existe un défaut rédhibitoire : les caractères sous le curseur sont totalement illisibles (que ce curseur se trouve à l'intérieur du tableau ou sur la ligne d'entrée de données). Une version spéciale Apple //c est donc nécessaire (elle existe peut-être). Vous pouvez, utilisateur du //e, remiser votre disquette système au magasin des accessoires inutiles. A moins que le fournisseur ne propose un échange...

Pas de Calcstar bien sûr (CP/M).

Les traitements de texte

Magic Window II (Autotexte en Français) fonctionne normalement, ainsi que son complément Magic Mailer (Autocourrier).

Il y a en revanche des problèmes pour Apple Writer //e. La ligne d'état du système est perturbée par des taches bizarres (interférence, paraît-il, avec le système de gestion des icônes spécial à l'Apple //c). Toutefois, cet inconvénient n'empêcherait pas d'utiliser le logiciel (la chance veut que les dites tâches ne recouvrent pas les mentions essentielles telles que position du curseur ou capacité de mémoire utilisée), si ne s'y ajoutait le même défaut que dans le cas de Multiplan : caractères sous curseur illisibles (y compris le nom du fichier qui est toujours totalement illisible). A noter qu'on dispose de 46845 caractères, au lieu de 27645 sur le //e.
Même remarques donc que pour Multiplan, ce qui met hors course une seconde vedette. Et bien sûr pas de Wordstar...

Les gestionnaires de fichiers

Tout va bien pour les propriétaires de Visifile et de CX Multigestion (en 40 colonnes), d'Omnis (malgré la jonglerie d'utilisation avec un seul disque), de P.F.S., de Visidex. Même le vieux Doc Database (écrit en Basic entier, sous DOS 3.2, donc après transposition en DOS 3.3) passe sans problème.

Mais voilà. Pas de dBASE II (CP/M). Et pas de produit de remplacement. Certes, Omnis 3 est annoncé; mais il nécessitera 3 lecteurs de disques, et ne pourra donc pas tourner sur Apple //c, qui ne peut être muni que de 2 lecteurs (sauf à travailler sur disque dur ?). A quand une VRAIE base de données sous ProDOS ?

Les éditeurs graphiques

Nous avions quelques craintes, compte tenu des différences relativement importantes dans le système de gestion de l'écran haute résolution. Elles n'étaient pas fondées.

Nous n'avons pas pu faire l'essai de Business Graphics, qui exige 2 lecteurs de disques. Mais Visiplot (qui donne des tracés couleur sur écran de bonne qualité) aussi bien que Charts Unlimited passent la rampe sans difficulté. De même que Matgraph (Pom's No 10).

Quelques programmes divers (écrits par l'auteur) faisant appel aux pages graphiques haute résolution 1 ou 2 n'ont pas posé d'avantage de problèmes.

Programmes de simulation et jeux

Tous ceux que nous avons essayés ont fonctionné correctement, ce qui ne veut pas dire qu'il en sera toujours de même.

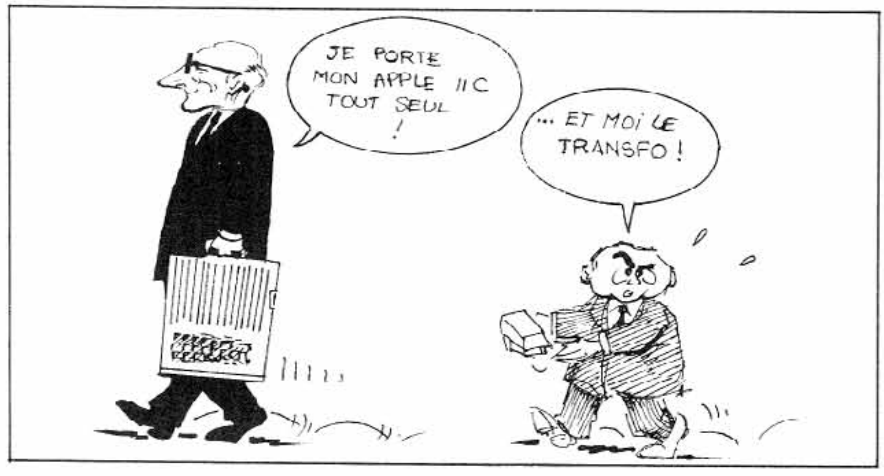
C'est dans ce domaine que la connexion double avec le téléviseur (RGB plus son) joue pleinement son rôle. De belles images (comme par exemple dans Adventure to Atlantis ou Twerps), et des bruits réalistes (on peut monter la "sono", ce qui doit plaire à nos chères têtes blondes...).

Des simulations à vocation pédagogique écrites par l'auteur ont également donné des résultats satisfaisants.

En résumé

L'Apple //c apparaîtra aux possesseurs d'Apple //e comme une machine agréable à utiliser et que l'arrivée d'un écran plat rendra vraiment portable. A ceci s'ajoute la prochaine disponibilité de divers types de batteries rechargeables, dont sans doute un modèle léger qui tiendrait dans la sacoche de transport.

Chacun appréciera le clavier, la liaison Péritel en série, le lecteur de disquette intégré, avec l'avantage d'une totale compatibilité avec le //e au niveau des supports, mais en regrettant parfois d'en rester aux 140K. Cet inconvénient devrait être rapidement compensé par la possibilité de connecter un disque dur, ou un mini-disque souple 3"1/2 (type Macintosh).



L'utilisation des logiciels écrits sous DOS ou Pascal ne posera guère de problèmes, avec une restriction pour deux des logiciels classiques les plus prisés : Multiplan et Apple Writer. Mais n'oublions pas la venue d'Apple Works, qui combine, avec l'utilisation de la souris, tableur, traitement de texte et gestionnaire de fichiers, dans des conditions satisfaisantes grâce à la capacité mémoire augmentée (ainsi bien entendu que de nombreux logiciels nouveaux, écrits spécifiquement pour l'Apple //c).

Les "classiques", qui ne sont pas à la recherche de solutions plus ou moins bricolées, devraient être de futurs utilisateurs heureux de l'Apple //c, avec comme seul vrai regret, pour nombre d'entre eux, l'absence de CP/M et du cortège de logiciels de premier plan écrits sous ce système d'exploitation.

Les "bidouilleurs" ... feraient mieux de s'abstenir.

■

Analyse de la VTOC

Guy d'Herbemont

Voici un petit programme d'inspection de la VTOC (Volume Table Of Contents) d'une disquette en DOS 3.3; il pourra, je l'espère, être utile aux débutants qui souhaitent comprendre les mécanismes internes du DOS, ainsi qu'aux programmeurs chevronnés qui désirent une routine rapide illustrant la place restante disponible sur disque.

Pour rappel, la VTOC est la table des matières d'une disquette, et le point d'entrée obligatoire pour toutes les opérations sur les fichiers.

Le programme affiche d'abord un dump de l'en-tête de la VTOC (pour plus de détail, consultez le Manuel DOS d'Apple, page 125), puis le schéma d'occupation des pistes et secteurs. Enfin, il indique les nombres de secteurs libres et occupés.

Le fonctionnement du programme est très simple et se prête à toutes sortes d'ajouts possibles; il est numéroté à partir de 61000 (pour pouvoir être utilisé comme sous-programme) et commence par un appel au DOS pour lire la VTOC actuelle. Après avoir copié celle-ci (par la routine de S.H.Lam - Voir Pom's 2) pour qu'elle débute à une limite de page et facilite ainsi la lecture avec le manuel, le programme enchaîne vers l'affichage des pistes 0 à 17, puis 18 à 34 sur les colonnes de droite.

La sous-routine d'affichage calcule pour chaque secteur son statut (libre si le bit est à 1, occupé sinon), met à jour le compteur de secteurs libres, et affiche le tout.

A noter que ce programme ne modifie pas le DOS. Il ne fonctionnera cependant pas avec des DOS placés à une adresse inhabituelle (32K ou carte langage), mais pourra néanmoins examiner des VTOC situées à des endroits exotiques, pourvu que le DOS chargé en mémoire connaisse leurs adresses ... ■

Exécution du programme : Premier écran

```
9700- 04 11 0F 03 00 00 FE 00
9708- 00 00 00 00 00 00 00 00
9710- 00 00 00 00 00 00 00 00
9718- 00 00 00 00 00 00 00 00
9720- 00 00 00 00 00 00 00 7A
9728- 00 00 00 00 00 00 00 00
9730- 1F 01 00 00 23 10 00 01
```

Exécution du programme : Second écran

```
   0123456789ABCDEF          0123456789ABCDEF

0  0000000000000000      18  0000000000000000
1  0000000000000000      19  0000000000000000
2  0000000000000000      20  1111111111111000
3  0000000000000000      21  1111111111110000
4  0000000000000000      22  0000000000000000
5  0000000000000000      23  0000000000000000
6  0000000000000000      24  0000000000000000
7  0000000000000000      25  0000000000000000
8  0000000000000000      26  0000000000000000
9  0000000000000000      27  0000000000000000
10 0000000000000000      28  0000000000000000
11 0000000000000000      29  0000000000000000
12 0000000000000000      30  0000000000000000
13 0000000000000000      31  0000000000000000
14 0000000000000000      32  0000000000000000
15 0000000000000000      33  1111000000000000
16 0000000000000000      34  0000000000000000
17 0000000000000000          1=SECTEUR LIBRE

OCCUPES=531, LIBRES=29, TOTAL=560
```

```
1  REM  *** PROGRAMME DISQ.VTOC ***
2  REM  ** AFFICHE VTOC DOS 3.3 **
61010  CALL 45047: POKE 72,0: TEXT : HOM
       E : REM BUFFER=46011
61020  DEF  FN P(X) = 256 *  PEEK (X) +
        PEEK (X + 1)
61030  DEF  FN B(X) = X > 2 *  INT (X >
       2): REM 0=PAIR,1=IMPAIR
61039  REM  ** DEBUT DE LA VTOC **
61040 A$ = "9700<B3BB.B3FEM N9700.9737 N
       D823G": FOR I = 1 TO  LEN (A$): P
       OKE 511 + I, ASC ( MID$ (A$,I,1))
        + 128: NEXT : POKE 72,0: CALL  -
        144: PRINT : GET A$: HOME : REM
       ROUTINE S.H.LAM
61049  REM  ** TABLE DES SECTEURS **
61050 A$ = "0123456789ABCDEF":N = 0
61060  PRINT  SPC( 3)A$: PRINT :A = 4606
       7:B = 46135:S = 1:T = 4: GOSUB 61
       200: REM PISTES 0-17
61070  VTAB 1: HTAB 24: PRINT A$: PRINT
       :A = 46139:B = 46203:S = 21:T = 2
       4: GOSUB 61200: REM PISTES 18-34
61080  HTAB T: PRINT "1=SECTEUR LIBRE"
61090  PRINT : PRINT "OCCUPES="560 - N",
        LIBRES="N", TOTAL=560": END
61199  REM  ** ROUTINE D'AFFICHAGE **
61200  FOR I = A TO B STEP 4:X =  FN P(I
       ):X$ = "": FOR J = 1 TO 16:X$ = X
       $ +  STR$ ( FN B(X)):N = N +  FN
       B(X):X =  INT (X / 2): NEXT : HTA
       B S: PRINT (I - 46067) / 4;: HTAB
        T: PRINT X$: NEXT : RETURN
```

Bloc-notes

Jerôme Leclercq

Si vous perdez votre temps à rechercher les petits bouts de papier sur lesquels vous inscrivez des informations très précieuses, par exemple votre mot de passe pour DB Master, ce programme est pour vous : il permet, en effet, de stocker dans un fichier bien propre des notes courtes, et toujours disponibles.

Ce programme a été créé initialement sur un Apple II Plus muni de la carte minuscules ROM LC (NDLR : ROM Lower Case, pour avoir les minuscules sur un Apple II Plus) et d'une imprimante Centronics 739. Cependant, il est assez simple, en changeant quelques lignes, de l'adapter à d'autres configurations qui ne permettent pas l'introduction de minuscules ou possèdent une autre imprimante. C'est ce qui est réalisé en partie par le programme BLOC-NOTES-DEBUT, qui donne le choix entre les configurations Apple II Plus, II Plus avec ROM LC, //e et //c.

Description du programme

BLOC-NOTES est un programme de saisie de notes non ordonnées d'une longueur maximale de 120 caractères, soit 3 lignes d'écran, qui permet la lecture de toutes les notes ou d'une seule, la modification, la suppression ou l'édition dans l'ordre alphabétique ou de saisie.

Au début de l'exécution, le programme charge un utilitaire de tri, l'utilitaire PROGR48K publié dans le Pom's 5, et initialise une routine de génération de son et la routine "DOS 7" du Pom's 2 qui permet de connaître le nombre de secteurs encore disponibles sur la disquette à chaque CATALOG. Ensuite, après avoir indiqué son nom et la date, on arrive dans un menu constitué de deux pages.

Pour pouvoir écrire des notes, il faut d'abord "créer un fichier" (choix 3 de la page 2 du menu) en indiquant le nom que l'on souhaite lui donner; cela crée un fichier annexe (suffixé par .NB) qui contient le nombre de fiches contenues dans le fichier principal.

Ensuite, il existe deux possibilités :

- écrire une note en prenant le choix 1 de la première page du menu;
- faire le choix 1 de la deuxième page du menu, la "SAISIE RAPIDE"; cette solution, lors d'une longue saisie, est très avantageuse.

Les autres choix sont explicités par leur titre même :

LIRE TOUT LE FICHIER - LIRE UNE NOTE - EFFACER UNE NOTE - CORRIGER UNE NOTE - CATALOGUE DE LA DISQUETTE (nombre de secteurs libres) - EDITER TOUTES LES NOTES (en 132 caractères par ligne) - DETRUIRE UN FICHIER - COMPACTER TOUT LE FICHIER : une note effacée n'est pas supprimée mais remplacée par des caractères blancs; il faut donc réorganiser périodiquement le fichier.

Sur le plan ergonomique, le programme permet, au lieu de répéter à chaque entrée le nom du fichier, de faire apparaître en tapant RETURN le titre du fichier en cours. Dans certains cas, pour éviter de détruire intempestivement des notes ou même le fichier entier, on est obligé de retaper le nom du fichier.

Configurations diverses

Les opérations effectuées pour mettre en oeuvre le programme avec diverses configurations sont indiquées ci-dessous. Elles sont réalisées automatiquement par le programme BLOC-NOTES-DEBUT.

Apple II Plus et ROM LC

Après avoir fait le BRUN PATCH qui permet, quand on possède la ROM LC, de brancher les "minuscules", on lance le programme par RUN BLOC-NOTES.

Apple II Plus sans ROM LC

Le programme MINMAJ de conversion minuscule/majuscule réalisé par Alexandre Avrane permet de transformer automatiquement, une fois qu'il est chargé, toutes les minuscules en majuscules. Il suffit de faire préalablement "BRUN MINMAJ" pour utiliser le bloc-notes avec un Apple II Plus sans ROM LC. Ce programme est désactivé par le RESET. Bien entendu, vous pouvez utiliser MINMAJ avec tout programme comportant des minuscules : les minuscules paraîtront alors sous la forme de majuscules, y compris les minuscules accentuées et le "ç".

Apple //e ou //c

Dans le cas de l'Apple //e, il n'est pas nécessaire de "brancher" les minuscules, il faut donc supprimer les POKEs qui avaient permis de changer CTRL-A en ESC pour le passage Majuscule/minuscule (lignes 102, 247, 905, 1045, 7001 et 9030). D'autre part, le résumé des lettres avec accent (é, è, à et ù) et de la cédille (ç) n'est plus utile : on peut donc supprimer les lignes 158 à 175.

Modification de l'imprimante

Ceux qui ne possèdent pas la Centronics 739 doivent modifier les codes à envoyer à l'imprimante dans les lignes 1785 (132 caractères de large), 1787 (écriture en condensé et en double largeur) et 1788 (fin d'écriture en double largeur). Ceux qui ne possèdent pas d'imprimante en 132 colonnes ont intérêt à modifier les lignes 190, 210, 560, 950, 1010, 1160, 1350, 1433, 1440, 1750, 2540 et 6060.

■

Note aux auteurs

En raison de l'affluence des contributions spontanées reçues par Pom's, nous prions les lecteurs-auteurs de nous pardonner s'il nous faut quelque temps pour analyser leurs créations : à certaines périodes, nous recevons jusqu'à un article par jour ...

Nous souhaitons aussi vous rappeler certains principes de base qui faciliteront le travail d'analyse et de préparation des articles. L'idéal est de nous faire parvenir une disquette contenant :

- le(s) fichier(s) du programme proposé;
- un ou deux fichiers d'exemples, si le programme utilise des fichiers externes;
- un texte sous forme de fichier TEXT en DOS 3.3 (au verso si le recto est dans un autre système d'exploitation), de préférence en Apple Writer; n'oubliez pas d'indiquer dans ce texte, le cas échéant, l'assembleur utilisé ainsi que le type de matériel nécessaire;

Pensez aussi à mettre une étiquette avec vos nom, adresse et numéro(s) de téléphone, ainsi qu'une indication du système d'exploitation (DOS 3.3, ProDOS, Pascal, ...) utilisé sur chaque face. Il est utile d'ajouter un listing du catalogue de la disquette avec une brève explication du rôle de chaque fichier.

Programme BLOC-NOTES.DEBUT

```
0  REM   BLOC-NOTES DEBUT
1  REM AA 10/6/84
10  TEXT : HOME : INVERSE : SPEED= 255:
    NOTRACE : VTAB 5: HTAB 16: PRINT
    " BLOC-NOTES ": NORMAL :D$ =  CHR
    $ (4)
```

```
20  VTAB 15: PRINT : PRINT "1 - APPLE II
    +": PRINT "2 - APPLE II+ AVEC ROM
     LC": PRINT "3 - APPLE IIE/IIC"
30  VTAB 21: PRINT : PRINT "VOTRE CONFIG
    URATION? -->";: GET A$:A =  VAL (
    A$): ON A GOTO 100,110,120: GOTO
    30
40  NORMAL
45  PRINT : PRINT
50  PRINT "AVANT DE LANCER L'EXECUTION D
    U PROGRAMMEBLOC-NOTES, SI VOUS PO
    SSEDEZ LA ROM LC  FAITES BRUN PAT
    CH"
70  NEW
```

```
100  PRINT : PRINT D$"BRUN SHIFT.$9A00";
       GOTO 120
110  PRINT : PRINT D$"EXEC BLOC-NOTES.II
       +LC": END
120  PRINT : PRINT D$"RUN BLOC-NOTES"
```

Fichier EXEC : BLOC-NOTES //+LC

```
BRUN PATCH
RUN BLOC-NOTES
```

Programme BLOC-NOTES

```
10  GOSUB 10000
20  GOSUB 9000
50  GOTO 20
100  REM   **********************
           *                    *
           *     écriture       *
           *                    *
           **********************
101 I$ = "":CT = 0
102  POKE 40067,155
103 X$ = ""
105  TEXT
110  HOME : PRINT "              ECRITURE"
120  PRINT : PRINT "Quel est le nom du fichie
       r? ";: &  INPUT I$: IF I$ <  > "" THEN
        NOM$ = I$
130  IF NOM$ = "" THEN  POKE 768,255: POKE 76
       9,208: CALL 770: GOTO 20
135  PRINT NOM$: FOR T = 1 TO 100: NEXT T: HO
       ME
142  IF NB <  > 0 THEN 150
145  GOSUB 4000
150  IF NB > 500 THEN  HOME : INVERSE : PRINT
       "CE FICHIER CONTIENT PLUS DE 500 ENTR
      EES": NORMAL : PRINT " On ne peut plus
       rentrer de nouvelle fiche. ": POKE 76
      8,255: POKE 769,208: CALL 770: FOR T =
       1 TO 400: GOTO 20
158  HTAB 1: VTAB 19: PRINT "_____________
      ________________________": VTAB 21: HT
      AB 1
160  PRINT "SHIFT 8: é   SHIFT 9: è   '/': c
            minuscules/MAJUSCULES: ESC
              SHIFT P: à   SHIFT 1: ù"
165  PRINT "éèéèéèéèéèéèéèéèéèéèéèéèéèéèéè
       éèéèéèéè"
170  VTAB 1: HTAB 1
175  PRINT "éèéèéèéèéèéèéèéèéèéèéèéèéèéèéè
       éèéèéèéè"
178  PRINT "Fichier ";NOM$
180  PRINT "Entrez 120 caractères au maximum
              (= 3 lignes )"
183  PRINT "Fiche numéro ";: INVERSE : PRINT
       NB: NORMAL
185  PRINT
190  &  INPUT X$,L120
195  IF X$ = "" THEN 20
200  IF  LEN (X$) > 120 THEN  HOME : INVERSE
       : PRINT "CHAINE TROP LONGUE ERREUR": F
       OR T = 1 TO 400: NEXT T: GOTO 20
210  PRINT OP$ + NOM$ + ",L121"
220  PRINT WT$ + NOM$ + ",R";NB
230  PRINT X$
240  PRINT CL$ + NOM$
245  GOSUB 4100
246  IF CT = 1 THEN 7030
247  POKE 40067,129
```

des trémas à l'écran

```
250  GOTO 20: REM  FIN DE LA ROUTINE D'ECRITU
       RE
500  REM   **********************
           *                    *
           *      lecture       *
           *                    *
           **********************
501  HOME : VTAB 12
502  PRINT "Nétoyage mémoire: un peu de patie
       nce SVP":AP =  FRE (0)
505 I$ = ""
510  TEXT : HOME
520  HTAB 16: INVERSE : PRINT "LECTURE": NORM
       AL : PRINT : PRINT "Quel est le nom du
        fichier à lire?";: &  INPUT I$: IF I$
        <  > "" THEN NOM$ = I$
525  IF NOM$ = "" THEN  POKE 768,255: POKE 76
       9,208: CALL 770: GOTO 20
530  PRINT NOM$: FOR T = 1 TO 300: NEXT T
540  GOSUB 4000: REM * NB=? *
560  PRINT OP$ + NOM$ + ",L121"
570  FOR X = 1 TO NB - 1
580  PRINT RD$ + NOM$ + ",R";X
590  &  INPUT RP$(X)
600  NEXT X
610  PRINT CL$ + NOM$
615  HOME
616  PRINT "Lecture de ";NOM$
617  PRINT : POKE 34,2
620  FOR Y = 1 TO NB - 1
625  IF RP$(Y) = "" THEN  PRINT "-(";Y;")";:
       FLASH : PRINT "NOTE EFFACEE !!": NORMA
       L : NEXT Y
630  PRINT "-(";Y;")";RP$(Y)
640  NEXT Y
642  PRINT : PRINT
645  INVERSE : PRINT "> TAPEZ UNE TOUCHE APRE
       S LECTURE <": NORMAL : GET QL$$
646 NB = 0
650  GOTO 20
900  REM   **********************
           *                    *
           *    correction      *
           *                    *
           **********************
902 NOM$ = ""
905  POKE 40067,155
910  TEXT : HOME
920  HTAB 15: INVERSE : PRINT "CORRECTION": N
       ORMAL
923  PRINT : PRINT : PRINT : PRINT
925  PRINT : PRINT "Quel est le nom du fichie
       r? ";: &  INPUT NOM$
926  IF NOM$ = "" THEN  POKE 768,255: POKE 76
       9,208: CALL 770: GOTO 20
927  PRINT NOM$
930  GOSUB 4000: INPUT "Quel est le numéro de
        la note que  vous voulez corriger?";N
940  IF N >  = NB THEN  HOME : PRINT "Ce numé
       ro n'existe pas": POKE 768,255: POKE 7
       69,208: CALL 770: FOR T = 1 TO 400: NE
```

```
      XT T: GOTO 20
950   PRINT OP$ + NOM$ + ",L121"
960   PRINT RD$ + NOM$ + ",R";N
970   &  INPUT DA$
980   PRINT CL$ + NOM$
985   HOME
990   PRINT "CORRECTION: ": PRINT DA$
1000  VTAB 2: HTAB 1: &  INPUT TT$
1002  IF TT$ = "" THEN 20
1003  IF  LEN (TT$) > 120 THEN  HOME : PRINT
       "Chaine trop longue erreur": POKE 768,
       255: POKE 769,208: CALL 770: FOR T = 1
        TO 300: NEXT T: GOTO 20
1010  PRINT OP$ + NOM$ + ",L121"
1020  PRINT WT$ + NOM$ + ",R";N
1030  PRINT TT$
1040  PRINT CL$ + NOM$
1045  POKE 40067,129
1050 NB = 0: GOTO 20
1100  REM  **********************
           *                    *
           *  lecture de note   *
           *                    *
           **********************
1105  HOME
1110  HTAB 10: INVERSE : PRINT "LECTURE D'UNE
       NOTE": NORMAL : PRINT : PRINT : PRINT
       : PRINT
1120  &  INPUT I$,"Quel est le nom du fichier
       à lire? ": IF I$ <  > "" THEN NOM$ =
       I$
1130  IF NOM$ = "" THEN  POKE 768,255: POKE 7
       69,208: CALL 770: GOTO 20
1135  PRINT NOM$
1140  PRINT : INPUT "Quel est le numéro de la
       note? ";NN
1145  GOSUB 4000
1150  IF NN > NB THEN  HOME : PRINT "Ce numér
       o n'existe pas": POKE 768,255: POKE 76
       9,208: CALL 770: FOR T = 1 TO 400: NEX
       T T: GOTO 20
1160  PRINT OP$ + NOM$ + ",L121"
1170  PRINT RD$ + NOM$ + ",R";NN
1180  &  INPUT SY$
1190  PRINT CL$ + NOM$
1200  INVERSE : PRINT "NOTE ";NN;":": NORMAL
1205  IF SY$ = "" THEN  FLASH : PRINT "NOTE E
       FFACEE !!": NORMAL : PRINT : PRINT : P
       RINT : PRINT : PRINT : GOTO 1215
1210  PRINT SY$: PRINT : PRINT : PRINT : PRIN
       T : PRINT
1215  VTAB 23
1220  INVERSE : PRINT ">TAPEZ  UNE  TOUCHE  A
       PRES  LECTURE<": NORMAL : GET LS$
1230 NB = 0: GOTO 20
1300  REM  **********************
           *                    *
           *    compactage      *
           *                    *
           **********************
1305 NOM$ = ""
1310  TEXT : HOME
1320  HTAB 15: INVERSE : PRINT "COMPACTAGE":
       NORMAL
1330  PRINT : PRINT "Quel est le nom du fichi
       er à compacter ? ";: &  INPUT NOM$: IF
        NOM$ = "" THEN  POKE 768,255: POKE 76
       9,208: CALL 770: GOTO 20
1335  PRINT NOM$
1340  GOSUB 4000
1350  PRINT OP$ + NOM$ + ",L121"
1360  FOR X = 1 TO NB - 1
1370  PRINT RD$ + NOM$ + ",R";X
1380  &  INPUT RP$(X): NEXT X
1390  PRINT CL$ + NOM$
1400  REM  **********************
           *                    *
           * RP$(Y) -> BA$(Q)   *
           *                    *
           **********************
1405 Q = 1
1410  FOR Y = 1 TO NB - 1
1420  IF RP$(Y) <  > "" THEN BA$(Q) = RP$(Y):
       Q = Q + 1
1430  NEXT Y
1433  PRINT OP$ + NOM$ + ",L121": PRINT DE$ +
       NOM$
1440  PRINT OP$ + NOM$ + ",L121"
1450  FOR K = 1 TO Q - 1
1460  PRINT WT$ + NOM$ + ",R";K
1470  PRINT BA$(K)
1480  NEXT K
1490  PRINT CL$ + NOM$
1500  REM  **********************
           *                    *
           *     NB = Q         *
           *                    *
           **********************
1530  PRINT OP$ + NOM$ + ".NB?,L10"
1540  PRINT WT$ + NOM$ + ".NB?,R1"
1550  PRINT Q: PRINT CL$ + NOM$ + ".NB?"
1560 NB = 0: GOTO 20
1700  REM  **********************
           *                    *
           *     édition        *
           *                    *
           **********************
1703  HOME : VTAB 12: PRINT "Nétoyage mémoire
       : un peu de patience SVP":AP =  FRE (0
       )
1705 NOM$ = ""
1710  TEXT : HOME
1720  HTAB 16: INVERSE : PRINT "EDITION": NOR
       MAL : PRINT
1725  &  INPUT YT$,"Quel est le nom du fichie
       r à éditer ?   ": IF YT$ <  > "" THEN
       NOM$ = YT$
1726  PRINT NOM$: IF NOM$ = "" THEN  POKE 768
       ,255: POKE 769,208: CALL 770: GOTO 20
1727  &  INPUT NM$,"Numero devant chaque entr
       ée ? ": IF  LEFT$ (NM$,1) = "O" THEN N
       M = 1
1728  IF  LEFT$ (NM$,1) <  > "O" THEN NM = 0
1731  PRINT : &  INPUT DI$,"Double interligne
        ? ": &  INPUT TC$,"Tiret devant chaqu
       e entrée ? ": IF NOM$ = "" THEN 20
1732  IF  LEFT$ (DI$,1) = "O" THEN DI = 1
1733  IF  LEFT$ (DI$,1) <  > "O" THEN DI = 0
1734  IF  LEFT$ (TC$,1) = "O" THEN TC = 1
1735  IF  LEFT$ (TC$,1) <  > "O" THEN TC = 0
1736  IF DI$ = "" THEN 20
1737  IF TC$ = "" THEN 20
1740  GOSUB 4000: REM  NB=?
1750  PRINT OP$ + NOM$ + ",L121"
1760  FOR X = 1 TO NB - 1: PRINT RD$ + NOM$ +
       ",R";X
1770  &  INPUT RP$(X): NEXT
1780  PRINT CL$ + NOM$
1781  GOSUB 5000
1783  PRE 1
1785  PRINT  CHR$ (9); CHR$ (177); CHR$ (179)
       ; CHR$ (178); CHR$ (206)
1787  PRINT  CHR$ (27); CHR$ (20): PRINT  CHR
       $ (27); CHR$ (14); TAB( 18)"NOTES DU F
       ICHIER ";NOM$
1788  PRINT  CHR$ (27); CHR$ (15)
1790  FOR P = 1 TO NB - 1
1800  IF RP$(P) = "" THEN  NEXT
1810  IF TC = 1 AND NM = 0 THEN  HTAB 10: PRI
       NT "-";" ";RP$(P)
1811  IF TC = 1 AND NM = 1 THEN  HTAB 10: PRI
       NT "(";P;")- ";RP$(P)
1812  IF TC = 0 AND NM = 0 THEN  HTAB 10: PRI
       NT RP$(P)
1813  IF TC = 0 AND NM = 1 THEN  HTAB 10: PRI
       NT "(";P;") ";RP$(P)
1815  IF DI = 1 THEN  PRINT
1820  NEXT
1830  PRINT : PRINT : PRINT "          ";N$;"
       le ";DT$
```

```
1835  FOR WE = 1 TO 5: PRINT  CHR$ (10): NEXT

1840  PR# 0
1841  PRINT : &  INPUT PQ$,"Voulez-vous une a
      utre impression ? ": IF  LEFT$ (PQ$,1)
       = "O" THEN 1783
1850 NB = 0: GOTO 20
2100  REM  **********************
           *                    *
           * création fichier   *
           *                    *
           **********************
2110 NOM$ = ""
2120  TEXT : HOME
2130  INVERSE : PRINT " CREATION D'UN FICHIER
       ": NORMAL
2140  PRINT : PRINT : FLASH : PRINT "ATTENTIO
      N": POKE 768,255: POKE 769,112: CALL 7
      70: NORMAL
2150  PRINT "Lorsque l'on crèe un fichier, si
       un fi- chier du meme nom existe déjà,
       il sera  définitivement effacé."
2160  PRINT : &  INPUT NO$,"Voulez-vous vraim
      ent créer un fichier ? "
2170  IF  LEFT$ (NO$,1) <  > "O" THEN  POKE 7
      68,255: POKE 769,208: CALL 770: GOTO 2
      0
2180  &  INPUT NOM$,"Quel est le nom du fichi
      er ? ": IF NOM$ = "" THEN 20
2190 NB = 0: GOSUB 4100
2200  GOTO 20
2500  REM  **********************
           *                    *
           * destruction fich.  *
           *                    *
           **********************
2502 NOM$ = ""
2505  TEXT
2510  HOME : HTAB (14): INVERSE : PRINT "DEST
      RUCTION": NORMAL : PRINT
2520  &  INPUT NO$,"Voulez-vous vraiment détr
      uire un fichier ? ": IF  LEFT$ (NO$,1)
       <  > "O" THEN  POKE 768,255: POKE 769
      ,208: CALL 770: GOTO 20
2530  PRINT : &  INPUT NOM$,"Quel est le nom
      du fichier ? "
2535  IF NOM$ = "" THEN  POKE 768,255: POKE 7
      69,208: CALL 770: GOTO 20
2540  PRINT OP$ + NOM$ + ",121"
2550  PRINT DE$ + NOM$
2560  PRINT OP$ + NOM$ + ".ND?"
2570  PRINT DE$ + NOM$ + ".ND?"
2580  POKE 768,255: POKE 769,112: CALL 770:NB
       = 0: GOTO 20
2900  REM  **********************
           *                    *
           *    catalogue       *
           *                    *
           **********************
2905  TEXT
2910  HOME : HTAB 14: INVERSE : PRINT "CATALO
      GUE": NORMAL
2911  PRINT : PRINT "Drive 1 ou 2 ? ";: &  IN
      PUT DR$:DR =  VAL (DR$)
2913  IF DR < 1 OR DR > 2 THEN DR = 1
2925  POKE 34,2
2930  PRINT  CHR$ (4)"CATALOG,D"DR
2935  PRINT
2940  INVERSE : PRINT ">TAPEZ UNE TOUCHE APRE
      S LECTURE<": NORMAL
2950  GET SD$
2960  POKE 34,0
2990  POKE 768,255: POKE 769,112: CALL 770:NB
       = 0: GOTO 20
3000  REM  **********************
           *                    *
           *      erreurs       *
           *                    *
           **********************
3001  CALL 803: TEXT : HOME
3002 NOM$ = "":NB = 0
3003 L =  PEEK (222): IF L = 5 OR L = 6 OR L
      = 11 THEN  VTAB 11: PRINT "ERREUR DE D
      ISQUETTE": POKE 768,255: POKE 769,208:
       CALL 770: FOR T = 1 TO 500: NEXT : GO
      TO 20
3004  IF L = 4 THEN  VTAB 11: PRINT "DIQUETTE
       PROTEGEE": POKE 768,255: POKE 769,208
      : CALL 770: FOR T = 1 TO 500: NEXT T:
      GOTO 20
3005  IF L = 8 THEN  VTAB 11: PRINT "PORTE DI
      SQUETTE OUVERTE": POKE 768,255: POKE 7
      69,208: CALL 770: FOR T = 1 TO 600: NE
      XT T: GOTO 20
3006  IF L = 9 THEN  VTAB 11: PRINT "DISQUETT
      E PLEINE": POKE 768,255: POKE 769,208:
       CALL 770: FOR T = 1 TO 600: NEXT T: G
      OTO 20
3007  IF L = 10 THEN  VTAB 11: PRINT "FICHIER
       VEROUILLE": POKE 768,255: POKE 769,20
      8: CALL 770: FOR T = 1 TO 600: NEXT T:
       GOTO 20
3008  IF L = 7 THEN  VTAB 11: PRINT "VOLUME I
      NCOMPATIBLE": POKE 768,255: POKE 769,2
      08: CALL 770: FOR T = 1 TO 600: NEXT T
      : GOTO 20
3010  INVERSE : PRINT "                    ERREU
      R                 ": NORMAL
3015  PRINT : PRINT : PRINT
3021  PRINT "Erreur en ligne "; PEEK (218) +
       PEEK (219) * 256;" de code erreur ";L
      ;".
3022  PRINT "Voulez-vous sortir du programme?
       ";: &  INPUT PO$: IF  LEFT$ (PO$,1) <
        > "O" THEN NB = 0: GOTO 9000
3030  HOME : PRINT "    AU REVOIR !!"
3999  END
4000  REM  **********************
           *                    *
           *       NB = ?       *
           *                    *
           **********************
4010  PRINT OP$ + NOM$ + ".NB?,L10"
4020  PRINT RD$ + NOM$ + ".NB?,R1"
4030  INPUT NB
4040  PRINT CL$ + NOM$ + ".NB?"
4050  RETURN
4100  REM  **********************
           *                    *
           *    NB = NB + 1     *
           *                    *
           **********************
4110 NB = NB + 1
4140  PRINT OP$ + NOM$ + ".NB?,L10"
4150  PRINT WT$ + NOM$ + ".NB?,R1"
4160  PRINT NB
4170  PRINT CL$ + NOM$ + ".NB?"
4180  RETURN
4300  REM  **********************
           *                    *
           *        fin         *
           *                    *
           **********************
4310  HOME
4320  GOTO 3022
5000  REM  **********************
           *                    *
           *    tri oui/non     *
           *                    *
           **********************
5001  &  INPUT FH$,"Voulez-vous l'ordre alpha
      bétique? ": IF  LEFT$ (FH$,1) <  > "O"
       THEN  RETURN
5010 CD = NB - 1
5020  FOR Y = 1 TO CD:RP%(Y) = Y: NEXT Y
5030 T%(0) = CD
5040 T%(1) =  USR ( VAL (RP$(0)))
5050 T%(2) =  USR (RP%(0))
5060 T%(3) = 3
5070 T%(4) = 2
```

```
5080 CD = T%(0)
5090  CALL 31232
5099  RETURN
6000  REM  ********************
           *                  *
           * destruction note *
           *                  *
           ********************
6010  HOME : HTAB 14: INVERSE : PRINT "DESTRU
      CTION": NORMAL
6020  &  INPUT NOM$,"Quel est le nom du fichi
      er ? "
6030  IF NOM$ = "" THEN  POKE 768,255: POKE 7
      69,208: CALL 770: GOTO 20
6040  GOSUB 4000: INPUT "Quel est le numéro d
      e la note que vous  voulez détruire ?
      ";N
6050  IF N >  = NB THEN  POKE 768,255: POKE 7
      69,208: CALL 770: GOTO 20
6055 RI$ = ""
6056 RP$(N) = ""
6060  PRINT OP$ + NOM$ + ",L121"
6070  PRINT WT$ + NOM$ + ",R";N
6080  PRINT RI$
6090  PRINT CL$ + NOM$
6100 NB = 0: GOTO 20
7000  REM  ********************
           *                  *
           *   saisie rapide  *
           *                  *
           ********************
7001  POKE 40067,155
7002  HOME : HTAB 13: INVERSE : PRINT "SAISIE
       RAPIDE": NORMAL : PRINT : PRINT : PRI
      NT : PRINT
7005 CT = 1:I$ = ""
7011  &  INPUT I$,"Quel est le nom du fichier
      ? ": IF I$ <  > "" THEN NOM$ = I$
7012  IF NOM$ = "" THEN  POKE 768,255: POKE 7
      69,208: CALL 770: GOTO 20
7013  PRINT NOM$
7015  GOSUB 4000: REM  NB=?
7017  HOME
7020  VTAB 2: GOTO 150
7030  HOME : PRINT "Pour arreter la saisie ra
      pide, taper ESCsinon tapez une touche
      quelconque"
7035 X$ = ""
7040 S =  PEEK ( - 16384): POKE  - 16368,0
7045  IF S < 128 THEN 7040
7050  IF S = 155 THEN  GOTO 20
7060  GOTO 7020
8999  END
9000  REM  ********************
           *                  *
           *       menu       *
           *                  *
           ********************
9005 CT = 0
9010  TEXT : HOME
9030  POKE 40067,129
9070 V = 1: GOSUB 9490: FOR V = 1 TO 22: HTAB
       1: PRINT "■";: HTAB 39: PRINT "▒": NE
      XT V: GOSUB 9490
9080  VTAB 2: HTAB 10: INVERSE : PRINT "BLOC-
      NOTES PAGE 1": NORMAL
9090  VTAB 3: HTAB 2: FOR I = 2 TO 38: PRINT
      "à";: NEXT I
9100  FOR I = 1 TO 7: HTAB 3: VTAB 4 + 2 * I:
       PRINT I;".";M$(I): NEXT I
9120  VTAB 21: HTAB 15: PRINT "VOTRE CHOIX ";
       CHR$ (91);" "; CHR$ (93): VTAB 21: HT
      AB 28: GET C$: PRINT C$
9130 V =  VAL (C$): IF V = 0 THEN 9120
9135  IF V > 7 THEN 9120
9140  FOR I = 0 TO 7 - V: VTAB 18 - 2 * I: HT
      AB 4: PRINT ">": IF I > 0 THEN  VTAB 2
      0 - 2 * I: HTAB 4: PRINT "."
9150  NEXT I: IF V = 6 THEN  FOR I = 1 TO 200
      : NEXT
9160  VTAB 4 + 2 * V: HTAB 5: INVERSE : PRINT
       M$(V): NORMAL
9170  FOR I = 1 TO 100: NEXT I
9180  ON V GOTO 100,500,1100,6000,900,2900,92
      00
9200  REM  ********************
           *                  *
           *    menu page 2   *
           *                  *
           ********************
9210  HOME :V = 1: GOSUB 9490: FOR V = 1 TO 2
      2: HTAB 1: PRINT "■";: HTAB 39: PRINT
      "▒": NEXT V: GOSUB 9490
9220  VTAB 2: HTAB 9: INVERSE : PRINT "BLOC-N
      OTES PAGE 2": NORMAL
9230  VTAB 3: HTAB 2: FOR I = 2 TO 38: PRINT
      "è";: NEXT I
9240  FOR V = 8 TO 14: VTAB 2 * V - 10: HTAB
      3: PRINT (V - 7);".";M$(V): NEXT V
9260  VTAB 21: HTAB 15: PRINT "VOTRE CHOIX ";
       CHR$ (91);" "; CHR$ (93): VTAB 21: HT
      AB 28: GET C$: PRINT C$:V =  VAL (C$)
9270  IF V = 0 THEN 9260
9280  FOR I = 0 TO 7 - V: VTAB 18 - 2 * I: HT
      AB 4: PRINT ">": IF I > 0 THEN  VTAB 2
      0 - 2 * I: HTAB 4: PRINT "."
9290  NEXT I: IF V = 7 THEN  FOR I = 1 TO 200
      : NEXT
9300  VTAB 4 + 2 * V: HTAB 5: INVERSE : PRINT
       M$(V + 7): NORMAL
9310  FOR I = 0 TO 100: NEXT I
9320  ON V GOTO 7000,1700,2100,2500,1300,4300
      ,9000
9490  VTAB V: HTAB 2: FOR H = 2 TO 38: PRINT
      "è";: NEXT H: RETURN
9500  REM  ********************
           *                  *
           *  secteurs libres *
           *                  *
           ********************
9530 Y$ = "A884:5B N ADC3:20 00 B6 N B600:A2
       0C 20 4A F9 A9 00 85 40 85 41 A0 C8 18
       B9 F2 B3 F0 0E 0A 90 FB 48 E6 40 D0 0
      2 E6 41 68 18 90 F0 88 D0 E9 A6 40 A5
      41 AC 00 E0 C0 20 D0 07 20 1B E5 20 2F
       AE 60 20 24 ED 20 2F AE 60"
9540  GOSUB 9550
9545  RETURN
9550 Y$ = Y$ + " N D9C6G"
9560  FOR I = 1 TO  LEN (Y$)
9570  POKE 511 + I, ASC ( MID$ (Y$,I,1)) + 12
      8
9580  NEXT : POKE 72,0: CALL  - 144
9590  RETURN
10000  REM ********************
           *                  *
           *  initialisation  *
           *                  *
           ********************
10001  TEXT : HOME : VTAB 10: PRINT "initiali
       sation de la routine de tri,     de la
       routine de génération de son,      et d
       e la routine permettant de savoir   le
       nombre de secteurs libres à chaque
       CATALOGUE"
10003  HIMEM: 31230: PRINT  CHR$ (4)"CLOSE"
10004  GOSUB 20000
10005  GOSUB 40000
10008  GOSUB 45000
10009  GOSUB 9500: HOME : VTAB (10): PRINT "c
       hargement de PROGR48K, et de TRIL.M3"
10010  PRINT  CHR$ (4)"BRUNPROGR48K"
10020  PRINT  CHR$ (4)"BLOADTRIL.M3"
10050 D$ = ""
10060 OP$ = D$ + "OPEN"
10070 RD$ = D$ + "READ"
10080 CL$ = D$ + "CLOSE"
10090 WT$ = D$ + "WRITE"
10095 DE$ = D$ + "DELETE"
10100  NOTRACE
```

```
10105  ONERR  GOTO 3000
10107  DIM RP$(500),BA$(500),M$(14)
10108  DIM RP%(500),T%(4)
10110  HOME : PRINT "Le programme est chargé
       ": PRINT : PRINT "Retirez la disquette
        programme": PRINT : PRINT "et mettez
       la disquette données "
10115  POKE 768,255: POKE 769,212: CALL 770
10120  PRINT : PRINT "Quel est votre nom ? ";
       : &  INPUT N$
10125  IF N$ = "" THEN  VTAB 6: CALL  - 868:
       GOTO 10120
10130  PRINT : PRINT "Quelle est la date? ";:
        &  INPUT DT$
10135  IF DT$ = "" THEN  VTAB 8: CALL  - 868:
        GOTO 10130
10160  FOR I = 1 TO 14: READ M$(I): NEXT
10300  GOTO 20
20000  REM **********************
           *                    *
           * routine d'erreurs  *
           *                    *
           **********************


20010 RE$ = "323:68 A8 68 A6 DF 9A 48 98 48 6
       0 ND9C6G": FOR I = 1 TO  LEN (RE$): PO
       KE 511 + I, ASC ( MID$ (RE$,I,1)) + 12
       8: NEXT : POKE 72,0: CALL  - 144
20020  RETURN : REM
```

```
40000  REM **********************
           *                    *
           *  init. rout. tri   *
           *                    *
           **********************
40010  POKE 10,76: POKE 11,26: POKE 12,3
40020  POKE 794,165: POKE 795,132
40030  POKE 796,164: POKE 797,131
40040  POKE 798,76: POKE 799,242: POKE 800,22
       6
40055  RETURN
45000  REM **********************
           *                    *
           *  init. rout. son   *
           *                    *
           **********************
45010 MU$ = "300:02 02 AD 30 C0 88 D0 05 CE 0
       1 03 F0 09 CA D0 F5 AE 00 03 4C 02 03
       60 ND9C6G": FOR I = 1 TO  LEN (MU$): P
       OKE 511 + I, ASC ( MID$ (MU$,I,1)) + 1
       28: NEXT : POKE 72,0: CALL  - 144
45020  RETURN : REM

50300  DATA  ECRIRE,LIRE TOUT LE FICHIER,LIRE
        UNE NOTE,EFFACER UNE NOTE,CORRIGER UN
       E NOTE,CATALOGUE DE LA DISQUETTE,SUITE
        MENU
50310  DATA SAISIE RAPIDE,EDITER TOUTES LES N
       OTES,CREER UN FICHIER,DETRUIRE UN FICH
       IER,COMPACTER TOUT LE FICHIER,FIN,RETO
       UR DEBUT MENU
```

Programme PATCH

```
9C00- A9 1D 8D F2 03 A9 9C 8D
9C08- F3 03 20 6F FB A9 28 85
9C10- 38 A9 9C 85 39 A9 9B 8D
9C18- 01 9D 4C D3 03 A9 28 85
9C20- 38 A9 9C 85 39 4C BF 9D
9C28- 8E D6 9C 08 48 E0 00 D0
9C30- 03 8E D4 9C BA BD 07 01
9C38- C9 77 D0 11 BD 08 01 C9
9C40- FD D0 0A A9 B6 9D 07 01
9C48- A9 9C 9D 08 01 68 AE D6
9C50- 9C 8D D7 9C A4 24 A9 20
9C58- 91 28 A9 10 20 A8 FC E6
9C60- 4E D0 0F E6 4F B1 28 CD
9C68- 57 9C D0 EA AD D7 9C 4C
9C70- 58 9C AD D7 9C 2C 00 C0
9C78- 10 E0 91 28 AD 00 C0 2C
9C80- 10 C0 C9 81 D0 0D AD D4
9C88- 9C 49 FF 8D D4 9C B1 28
9C90- 4C 51 9C 2C D4 9C 10 1D
9C98- 8C D5 9C A0 0A 88 30 08
9CA0- D9 C0 9C D0 F8 B9 CA 9C
9CA8- AC D5 9C C9 DB B0 06 C9
9CB0- C1 90 02 09 20 28 60
```

Programme MINMAJ

```
9A00- A9 0F 85 36 A9 9A 85 37
9A08- A9 00 85 48 4C EA 03 8D
9A10- 4A 9A 08 8A 48 AD 4A 9A
9A18- A2 07 DD 3A 9A F0 0C CA
9A20- 10 F8 C9 E0 90 08 38 E9
9A28- 20 D0 03 BD 42 9A 8D 4A
9A30- 9A 68 AA 28 AD 4A 9A 4C
9A38- F0 FD C0 DC E0 FB FC FD
9A40- FE FF C1 C3 A1 C5 D5 C5
9A48- AD A1 00 FF
```

Programme PROGR 48K

```
9300- A9 4C 8D F5 03 A9 18 8D
9308- F6 03 A9 93 8D F7 03 A9
9310- 00 85 73 A9 93 85 74 60
9318- 02 BE 80 C9 84 F0 1D C9
9320- AB F0 13 C9 B0 F0 12 C9
9328- B5 F0 08 C9 AA D0 03 4C
9330- 4E 94 60 4C 2F 94 4C FF
9338- 93 4C 05 94 A4 25 84 07
9340- A4 24 84 08 A9 00 85 09
9348- 85 06 20 B1 00 20 E3 DF
9350- 20 6C DD 85 85 84 86 A0
9358- 00 B1 B8 C9 2C D0 35 20
9360- B1 00 C9 4C F0 14 C9 56
9368- F0 18 C9 48 F0 1D C9 22
9370- D0 05 E6 06 4C 94 93 4C
9378- 8D 94 20 F5 E6 86 09 4C
9380- 57 93 20 F5 E6 CA 86 07
9388- 4C 57 93 20 F5 E6 CA 86
9390- 08 4C 57 93 A5 07 85 25
9398- A5 08 85 24 20 22 FC A5
93A0- 06 F0 12 38 20 81 DE 20
93A8- 3D DB EA EA EA EA EA EA
93B0- EA EA 20 B7 00 A6 09 F0
93B8- 19 A5 32 48 20 84 FE A9
93C0- AE 20 ED FD CA D0 FA A6
93C8- 09 20 10 FC CA D0 FA 68
93D0- 85 32 20 6F FD 8A F0 1E
93D8- BD FF 01 C9 83 F0 1A BD
93E0- FF 01 29 7F 9D FF 01 CA
93E8- D0 F5 A9 00 A0 02 A2 8D
93F0- 20 E9 E3 20 9A DA A2 00
93F8- 60 20 3A FF 4C D0 03
```

Programme TRIL. M 3

```
7A00- A0 00 B1 83 85 94 C8 B1
7A08- 83 85 93 C8 B1 83 85 96
7A10- C8 B1 83 85 95 C8 B1 83
7A18- 85 9A C8 B1 83 85 99 C8
7A20- C8 B1 83 85 8A C8 C8 B1
7A28- 83 85 8B A0 00 A6 8A E0
7A30- 02 F0 0D E0 03 F0 13 E0
7A38- 05 F0 13 A9 FF 85 8A 60
7A40- A9 80 91 95 98 C8 91 95
7A48- D0 0E A9 00 F0 02 A9 FF
7A50- A4 8A 88 91 95 88 10 FB
7A58- 18 A5 8A 65 95 85 95 90
7A60- 02 E6 96 18 A5 8B 65 99
7A68- 85 99 90 02 E6 9A A5 93
7A70- D0 02 C6 94 C6 93 D0 07
7A78- A5 94 D0 03 85 8A 60 18
7A80- A5 95 85 97 65 8A 85 95
7A88- A5 96 85 98 69 00 85 96
7A90- 18 A5 99 85 9B 65 8B 85
7A98- 99 A5 9A 85 9C 69 00 85
7AA0- 9A 20 CA 7A 20 F0 7A 10
7AA8- 1B 20 36 7B 38 A5 97 E5
7AB0- 8A 85 97 B0 02 C6 98 38
7AB8- A5 9B E5 8B 85 9B B0 E4
7AC0- C6 9C 90 E0 20 69 7B 4C
7AC8- 6E 7A A4 8A C0 05 F0 0B
7AD0- 88 B1 95 99 9D 00 88 10
7AD8- F8 30 07 A4 96 A5 95 20
7AE0- F9 EA A4 8B F0 09 88 B1
7AE8- 99 99 A5 00 88 10 F8 60
7AF0- A6 8A E0 03 F0 1C 90 09
7AF8- A4 98 A5 97 20 B2 EB A8
7B00- 60 A0 01 38 A5 9E F1 97
7B08- 88 A5 9D F1 97 50 02 49
7B10- FF 60 A0 02 B1 97 99 A0
7B18- 00 88 10 F8 C8 C4 A0 F0
7B20- F0 C4 9D F0 0B B1 9E D1
7B28- A1 90 05 D0 06 C8 D0 ED
7B30- A9 FF 60 A9 00 60 A0 00
7B38- B1 97 48 C8 C4 8A D0 F8
7B40- 98 18 0A A8 88 68 91 97
7B48- 88 C4 8A B0 F8 A5 8B F0
7B50- 17 A0 00 B1 9B 48 C8 C4
7B58- 8B D0 F8 98 18 0A A8 88
7B60- 68 91 9B 88 C4 8B B0 F8
7B68- 60 18 A5 97 65 8A 85 97
7B70- 90 02 E6 98 A4 8A C0 05
7B78- F0 0B 88 B9 9D 00 91 97
7B80- 88 10 F8 30 06 A4 98 AA
7B88- 20 2B EB A4 8B F0 14 88
7B90- 18 A5 9B 65 8B 85 9B 90
7B98- 02 E6 9C B9 A5 00 91 9B
7BA0- 88 10 F8 60
```

Impression des variables

Laurent Esnault

Lors de la mise au point d'un programme Basic, on a souvent besoin de connaître le contenu des variables. Il faut pour cela faire d'innombrables PRINTs pour les variables simples et les tableaux : des boucles FOR-NEXT qui ont comme désavantage d'être longues et fastidieuses à l'usage ainsi que muettes sur le rang des variables dans le tableau. De plus, un autre problème peut se présenter : supposons que le programme Basic crée une variable, V par exemple, qui s'annule durant le traitement. Comment savoir si le déroulement du programme a été normal : si l'on tape "PRINT V" après l'exécution, l'Applesoft répondra par un "0" difficilement exploitable.

PRNVAR (PRiNt VARiables) permet de faire disparaître ce genre de difficultés, tout en fournissant des renseignements supplémentaires, avec l'aide du fameux "&" (ampersand) de l'Applesoft déjà fréquemment mis à contribution dans Pom's.

Ce programme permet aussi de mieux comprendre l'organisation des variables en Basic, grâce à l'option "P" en particulier. J'ai ainsi appris, en tapant "XG!A." au clavier, non seulement que cela engendre une SYNTAX ERROR, mais aussi que l'on crée par cette "instruction" une variable réelle nommée "XG".

Utilisation de PRNVAR

Ce programme affiche, selon les options qu'on lui fournit : les variables alphanumériques, réelles et entières, dimensionnées (tableaux) ou non, ainsi que leur adresse d'implantation et, pour les tableaux, leurs dimensions.
En outre, les fonctions pourront être signalées.

La syntaxe de cette nouvelle commande ajoutée au Basic est fort simple :

- "&" seul, affiche toutes les variables, quelle que soit leur nature.
- "&)" affiche tous les tableaux.
- "&R" (resp. "&%", "&$" ou "&#"), affiche les variables simples réelles (resp. entières, alphanumériques ou représentant une fonction).
- "&(R)" (resp. "&(%)" ou "&($)"), affiche les tableaux réels (resp. entiers ou alphanumériques).

Remarque : la commande "&(#)" n'aurait aucun sens, car les tableaux de fonctions n'existent pas, et sera donc ignorée.

On peut aussi regrouper des commandes :
"&(R%)$" aura pour effet d'afficher les tableaux réels et entiers ainsi que les variables simples alphanumériques.

La place des caractères n'a aucune importance :
"&$(%R)" aura le même effet que précédemment.

Cependant, il est bien évident que :
"&($)%R" n'aura pas le même effet.

Options pratiques

Il y a en outre deux options possibles :

- 0 : la présence de ce caractère dans la commande précise que vous voulez voir apparaître les variables nulles (celles-ci étant ignorées autrement). On peut toutefois inverser sa fonction, si on le désire, en remplaçant aux lignes 221 et 327 "BNE" par "BEQ".
- P : la présence de ce caractère précise que vous voulez voir apparaître l'adresse d'implantation du contenu de chaque variable à côté du contenu de celle-ci. Cette adresse correspondra en fait à l'endroit exact où est implanté le contenu de la variable. On obtient l'adresse du nom d'une variable simple en soustrayant 2 à l'adresse du contenu. Pour les tableaux, après par exemple DIM A(10), il apparaîtra en premier "TABLEAU A(11) EN : HHLL", HHLL correspondant à l'adresse d'implantation du nom du tableau. Remarquons au passage que ce n'est pas A(10) mais A(11) qui est affiché, cela provient du fait que DIM A(10) génère 11 éléments (de 0 à 10).
  Enfin, pour les variables représentant une fonction, viendra s'ajouter la ligne Basic où celle-ci a été définie.
  Encore une précision : pour des raisons de présentation, seuls les douze premiers caractères des variables alphanumériques seront affichés.

Pour obtenir l'une et/ou l'autre de ces deux options, il suffit d'introduire "0" et/ou "P" dans la commande, à n'importe quelle place.

Remarque : tout autre caractère que "R", "%", "$", "#", "(", ")", "0" ou "P" est considéré comme une demande de tout afficher dans la catégorie des variables prises en compte (simples et/ou dimensionnées). Ainsi, "&X" affiche toutes les variables simples sans les tableaux.
Enfin, on peut obtenir une pause dans l'affichage en appuyant sur n'importe quelle touche du clavier et le relancer en faisant de même. On revient au Basic par CTRL-C.

Description générale

Ce programme fait 880 octets et peut donc être placé juste avant le DOS (HIMEM : 9600), en $9290. Pour la sauvegarde sur disquette, LISA utilisant la page mémoire $9500-9600, il faut opérer ainsi :

1 - Taper le programme et l'assembler (!); le code objet sera alors implanté en $800.
2 - Faire CTRL-D "BSAVE PRNVAR,A$800,L$365".
3 - Booter une disquette sous Applesoft (Basic).
4 - Faire "BLOAD PRNVAR, A$9290".
5 - Faire enfin "BSAVE PRNVAR, A$9290,L$370".

Sa mise en fonction se fera ensuite simplement par "BRUN PRNVAR", ceci ayant pour effet de charger le programme en mémoire et de mettre à jour le vecteur & et les pointeurs HIMEM, FRETOP, ARYTAB et STREND. Les variables seront alors effacées. Il est donc préférable de faire "BRUN PRNVAR" avant l'exécution d'un programme Basic si l'on veut connaître le contenu des variables que celui-ci aura créées.

Passons au programme en lui-même. Il est constitué de trois parties principales :

1 - TRTAMP : traitement de la commande, c'est-à-dire l'analyse des caractères suivant l'ampersand et la détermination des choix et options. On y utilise CHRGET, sous-programme d'acquisition de caractères.
2 - TRAITM : traitement des variables simples. On effectue un balayage de la zone des variables simples (LOMEM→ARYTAB), à la recherche d'un certain type de variables (entières, réelles, alphanumériques ou représentant une fonction); lorsqu'on en a trouvé une, on l'affiche en tenant compte des options éventuelles. Quand ce premier balayage est effectué, on passe à un second type de variables (si l'utilisateur l'a demandé). N'oublions pas qu'une variable simple occupe 2 octets pour son nom et 5 octets quel que soit son type.
Il faut aussi remarquer que les varia-

bles représentant les fonctions (implantées uniquement dans la zone des variables simples) contiennent l'adresse d'implantation de la fonction dans le programme ainsi que l'adresse pointant sur le contenu de la variable prise en compte dans la fonction (exemple : X pour DEF FN A(X)=...).

3 - TABLEAU : traitement des tableaux. Il est similaire au traitement des variables simples, mis à part le fait que l'on affiche les dimensions de chaque élément du tableau pris en compte (balayage de ARYTAB à STREND).

L'organisation des variables est la suivante :

- 2 octets pour le nom;
- 2 pour la longueur du tableau;
- 1 pour le nombre de dimensions (inférieur à 89);
- 2 pour chaque dimension.

Plus :

- 5 octets par élément pour les réels;
- 2 pour les entiers;
- 3 pour les alphanumériques (1 pour la longueur de la chaîne et 2 pour l'adresse d'implantation).

(Pour plus de détails, voir le manuel de référence de l'Apple II.)

Viennent enfin les sous-programmes : quatre principaux, dont les REMs du listing expliquent la fonction.

Voilà, je pense avoir tout dit. Vous êtes maintenant en possession d'une commande supplémentaire qui, insérée dans le programme aux points "stratégiques" et associée à la fonction TRACE, constitue une arme très efficace pour la chasse aux "bugs".

La disquette de Pom's comporte un second programme de démonstration, appelé DEMO2.

■

LISA 2.5

```
1           NLS
2           TTL "AFFICHAGE DE VARIABLES PAR L'&"
3 ;         ****************************
4 ;         *                          *
5 ;         *        PRNVAR            *
6 ;         *                          *
7 ;         *   Par Laurent ESNAULT    *
8 ;         *                          *
9 ;         *        APPLE II          *
10 ;        *                          *
11 ;        *                  04/1984 *
12 ;        ****************************
13 ;
14 ;
15 ;        ****************************
16 ;        *                          *
17 ;        *  Definition des adresses *
18 ;        *                          *
19 ;        ****************************
20 ;
21 ;  1) EN PAGE ZERO :
22 ;
23 ADVAR    EPZ $FB             ;Pointeur general
24 IMPLDIM  EPZ $F9
25 LOMEM    EPZ $69
26 HIMEM    EPZ $73
27 HTAB     EPZ $24
28 V0       EPZ $CF
29 V1       EPZ $CF
30 FRETOP   EPZ $6F
31 STREND   EPZ $6D
32 ARYTAB   EPZ $6B
33 CHRGET   EPZ $B1
34 ;
35 ;  2) ABSOLUES :
36 ;
37 VECTEUR  EQU $3F5
38 KBD      EQU $C000
39 KEYSTR   EQU $C010
40 ADDIM    EQU $300
41 MOVFM    EQU $EAF9           ;(Y,A) -->FAC
42 COUT     EQU $FDED           ;Affiche CHRS(A)
43 STROUT   EQU $DB3A           ;Affiche la chaine (A,Y)
44 PRNFAC   EQU $ED2E           ;Affiche FAC
45 CROUT    EQU $FD8E           ;Retour chariot
46 PRBL2    EQU $F94A           ;X espaces
47 PRNTAX   EQU $F941           ;Affiche A et X
48 LINPTR   EQU $ED24           ;Affiche X*256+A
49 GIVAYF   EQU $E2F2
50 BELL     EQU $FF3A
51 ;
52 ;
53          ORG $9290
54          OBJ $800
55 ;
56 ;        ---------------------------------
57 ;        -                               -
58 ;        -   Stocke le vecteur & et      -
59 ;        -   Les nouveaux pointeurs.     -
60 ;        -          (=CLEAR)             -
61 ;        ---------------------------------
62 ;
63          LDA #$4C
64          STA VECTEUR
65          LDA #TRTAMPO
66          LDY /TRTAMPO
67          STA VECTEUR+1
68          STY VECTEUR+2
69          STA HIMEM
70          STY HIMEM+1
71          STA FRETOP
72          STY FRETOP+1
73          LDA LOMEM
74          LDY LOMEM+1
75          STA ARYTAB
76          STY ARYTAB+1
77          STA STREND
78          STY STREND+1
79          RTS
80 ;
81 ;
82 ;        ************************
83 ;        *                      *
84 ;        * TRAITEMENT DE L'&    *
85 ;        *                      *
86 ;        ************************
87 ;
88 TRTAMPO  LDY #0
89          STY AMPERS
90          STY OPNUL
91          STY OPPOIN
92 TRTAMP   TAX                 ;Si c'est ":"
93          BEQ FIN             ;Ou "00" alors
94          CPX #':'            ;Fin de la
95          BEQ FIN             ;Commande
96 ;
97          CPX #'('
98          BNE SUITE1
99          LDY #1
100         BNE NEWBCL
101 SUITE1  CPX #')'
102         BNE COMPAR
103         CPY #1
104         BEQ SUITE3
105         LDA #%11110000
106         BNE RET1
107 ;
108 SUITE3  LDY #0
109         BEQ NEWBCL
110 ;
111 ;       --------------------------------
112 ;       -                              -
113 ;       - On compare X avec les codes  -
114 ;       - Ascii de '0'et 'P' (options)-
115 ;       -  Puis de 'R','S','%' et '#'. -
116 ;       -                              -
117 ;       --------------------------------
118 COMPAR  LDA #1
119         CPX #'0'
120         BNE PAS0
121         STA OPNUL
122         BEQ NEWBCL
123 PAS0    CPX #'P'
124         BNE PASP
125         STA OPPOIN
126         BEQ NEWBCL
127 ;
128 PASP    CPX #'R'
129         BEQ FINBCL
130         ASL
131         CPX #'S'
132         BEQ FINBCL
133         ASL
134         CPX #'%'
135         BEQ FINBCL
136         ASL
137         CPX #'#'
138         BEQ FINBCL
139         LDA #%00001111
140 ;
141 FINBCL   CPY #1
142         BNE RET1
```

```
143             ASL
144             ASL                     ;On
145             ASL                     ;Modifie
146             ASL                     ;Ampers
147 RET1        ORA AMPERS              ;En
148             STA AMPERS              ;Consequence
149 ;
150 NEWBCL      JSR CHRGET
151             JMP TRTAMP
152 ;
153 FIN         LDA AMPERS
154             BNE TRAITM
155             LDA #%11111111
156             STA AMPERS
157             JMP TRAITM
158 ;
159 ;           -------------------------------
160 ;
161 NOM         DFS 2
162 TYPE        DFS 1
163 NUL         HEX 00
164 NBRDIM      DFS 1
165 CHOIX       DFS 1
166 LENVAR      DFS 1
167 OPPOIN      DFS 1                   ;<>0 si option 'P'
168 OPNUL       DFS 1                   ;<>0 si option '0'
169 AMPERS      DFS 1
170 V2          DFS 1
171 V3          DFS 1
172 ;
173 MESSAGE     HEX 8D
174             ASC "Tableau "
175             HEX 00
176 MESSAGE0    ASC " en "
177             HEX 00
178 MESSAGE1    ASC " en ligne "
179             HEX 00
180 MESSAGE2    ASC "fonction"
181             HEX 00
182 TYPE0       ASC " %& "
183 ;
184 ;           ---------------------------
185 ;           ************************
186 ;           *                      *
187 ;           * TRAITEMENT DES       *
188 ;           *   VARIABLES SIMPLES  *
189 ;           *                      *
190 ;           ************************
191 ;
192 TRAITM      LDA #4
193             STA CHOIX
194 ;
195 AUTRE       DEC CHOIX               ;On passe
196             BMI TABLEAU             ;A un autre
197             LSR AMPERS              ;Type de
198             BCC AUTRE               ;Variable
199 ;
200             LDA LOMEM               ;On commence
201             STA ADVAR               ;Le balayage
202             LDA LOMEM+1             ;A partir
203             STA ADVAR+1             ;De LOMEM
204 ;
205 GO          CMP ARYTAB+1            ;On verifie
206             BCC TEST                ;Que l'on
207             LDA ADVAR               ;N'est pas sorti
208             CMP ARYTAB              ;De la zone des
209             BCS AUTRE               ;Variables
210 ;
211 TEST        JSR INPUT               ;On regarde si
212             LDA #2                  ;Cette variable
213             JSR POIN.A              ;Convient
214             CPX CHOIX
215             BNE SUITE
216 ;
217             LDY CHOIX               ;Si fonction,
218             BEQ B1                  ;On passe.
219             JSR IFAFNUL             ;On teste
220             LDX OPNUL               ;Sa valeur
221             BNE B1                  ;Pour
222             TAX                     ;L'option '0'
223             BEQ SUITE
224 ;
225 B1          LDA #NOM                ;On envoie
226             LDY /NOM                ;Son nom
227             JSR STROUT              ;A l'affichage
228             JSR PRNVAR              ;Avec son contenu
229 ;
230 SUITE       LDA #5                  ;On passe
231             JSR POIN.A              ;A la variable
232             JMP GO                  ;Suivante
233 ;
234 ;           ************************
235 ;           *                      *
236 ;           *   TRAITEMENT DES     *
237 ;           * VARIABLES DIMENSIONNEES *
238 ;           *                      *
239 ;           ************************
240 ;
241 TABLEAU     LDA #4
242             STA CHOIX
243 ;
244 AUTRE2      DEC CHOIX
245             BNE CONT
246             RTS                     ;Retour au BASIC
247 CONT        LSR AMPERS
248             BCC AUTRE2
249 ;
250             LDA ARYTAB              ;On commence
251             STA ADVAR               ;Le balayage
252             LDA ARYTAB+1            ;A partir
253             STA ADVAR+1             ;De ARYTAB
254 ;
255 GO1         LDA ADVAR+1             ;On verifie
256             CMP STREND+1            ;Que l'on
257             BCC TEST2               ;N'est pas
258             LDA ADVAR               ;Sorti de
259             CMP STREND              ;La zone des
260             BCS AUTRE2              ;Tableaux .
261 ;
262 TEST2       JSR INPUT               ;On teste le
263             CPX CHOIX               ;Tableaux pointe
264             BEQ B2                  ;S'il ne convient
265             JMP SUITE2              ;Pas:SUITE2
266 ;
267 B2          LDY #4                  ;On stocke
268             LDA (ADVAR),Y           ;Le nbre de
269             ASL                     ;Dimensions
270             STA NBRDIM              ;Mult. par 2
271 ;
272             LDA #5
273             JSR POIN.A
274 ;
275             LDA ADVAR               ;On stocke
276             STA IMPLDIM             ;L'adresse
277             LDA ADVAR+1             ;D'implantation
278             STA IMPLDIM+1           ;Des dimensions
279 ;
280             LDA OPPOIN
281             BEQ NONOP
282 ;
283             LDA #MESSAGE            ;L'option 'P'
284             LDY /MESSAGE            ;A
285             JSR STROUT              ;Ete prise...
286             LDA #NOM
287             LDY /NOM
288             JSR STROUT
289             LDX #0                  ;...On
290             LDY NBRDIM              ;Affiche
291             DEY
292 B3          INX
293             LDA (IMPLDIM),Y         ;Les
294             STA ADDIM,X             ;Dimensions
295             DEY
296             DEX
297             LDA (IMPLDIM),Y
298             STA ADDIM,X
299             INX
300             INX
301             DEY
302             BPL B3                  ;Maximums...
303             JSR AFFDIM              ;...Et le
304             LDA #MESSAGE0           ;Pointeur
305             LDY /MESSAGE0           ;Pointeur
306             JSR STROUT              ;Initial.
307             JSR AFPOIN0
308 ;
309 NONOP       LDX NBRDIM              ;Remise
310             LDA #0                  ;A zero
311 B4          DEX                     ;Des
312             STA ADDIM,X             ;Compteurs
313             BNE B4                  ;De dimension
314 ;
315             LDA NBRDIM              ;On pointe
316             JSR POIN.A              ;sur la variable
317 ;
318             LDX CHOIX               ;On determine
319             INX                     ;Le nbre
320             CPX #4                  ;D'octets
321             BNE NON                 ;Qu'elle
322             INX                     ;Occupe et on
323 NON         STX LENVAR              ;Le stocke
324 ;
325 BCLE        JSR IFAFNUL
326             LDX OPNUL
327             BNE AFF
328             TAX
329             BEQ B5
330 ;
331 AFF         LDA #NOM                ;On affiche
332             LDY /NOM                ;:
333             JSR STROUT              ;Le nom
334             JSR AFFDIM              ;Les DIMS
335             JSR PRNVAR              ;La variable
336 ;
337 B5          LDA LENVAR              ;On pointe sur
338             JSR POIN.A              ;La prochaine
```

```
339 ;
340           LDX #0
341           LDY NBRDIM
342           DEY
343           DEY
344 ADD       INX                     ;On
345           CLC                     ;Met
346           LDA ADDIM,X             ;A jour
347           ADC #1                  ;Les
348           STA ADDIM,X             ;Compteurs
349           DEX                     ;De dimensions
350           LDA ADDIM,X             ;Et...
351           ADC #0
352           STA ADDIM,X
353           CMP (IMPLDIM),Y
354           BNE BCLE
355           INX
356           INY
357           LDA ADDIM,X
358           CMP (IMPLDIM),Y         ;...On
359           BNE BCLE                ;Continue
360           LDA #0                  ;Si le
361           DEX                     ;Tableau
362           STA ADDIM,X             ;N'est pas
363           INX                     ;Termine...
364           STA ADDIM,X
365           INX
366           DEY
367           DEY
368           DEY                     ;...Si oui
369           BPL ADD                 ;On passe
370           JMP GO1                 ;Au suivant
371 ;
372 SUITE2     LDY #2                  ;On pointe
373           CLC                     ;Sur
374           LDA (ADVAR),Y           ;Le tableaux
375           ADC ADVAR               ;Suivant
376           TAX                     ;Et...
377           INY
378           LDA (ADVAR),Y
379           ADC ADVAR+1
380           STA ADVAR+1             ;...On
381           STX ADVAR               ;Retourne
382           JMP GO1                 ;En GO1
383 ;
384 ;         ****************************
385 ;         *                          *
386 ;         * DEBUT DES SOUS-PROGRAMMES *
387 ;         *                          *
388 ;         ****************************
389 ;
390 ;         --------------------------------
391 ;         -                              -
392 ;         - INPUT determine la nature de -
393 ;         - La variable pointee et range -
394 ;         -      son nom.                -
395 ;         --------------------------------
396 ;
397 INPUT     LDX #1                  ;1er octet
398           LDY #0                  ;Superieur
399           LDA (ADVAR),Y           ;A $80 -->
400           BMI ENTIER              ;Entier ou fonction.
401           INX
402 ENTIER    STA NOM
403           INY                     ;2eme octet
404           LDA (ADVAR),Y           ;Superieur
405           BMI CHAINE              ;A $80 -->
406           INX                     ;Chaine ou entier
407           CPX #2
408           BNE CHAINE
409           LDX #0                  ;FONCTION
410 CHAINE    AND #%01111111
411           BNE OK
412           LDA #$A0
413 OK        STA NOM+1               ;Stocke
414           LDA TYPE0,X             ;Le nom
415           STA TYPE                ;Et le type.
416           RTS                     ;Puis retour
417 ;Resultat dans X:
418 ;1:Entier .2:Chaine .3:Reel .0:Fonction
419 ;         --------------------------------
420 ;         -                              -
421 ;         -   PRNVAR affiche la variable -
422 ;         -    Pointee et les pointeurs  -
423 ;         -   Si option 'P', puis        -
424 ;         -   Fait un test-clavier .     -
425 ;         -                              -
426 ;         --------------------------------
427 PRNVAR    LDA #"="
428           JSR COUT
429 ;
430           LDA CHOIX
431           BEQ FONCTIO
432           CMP #2
433           BNE NONALPH
434 ;
435           LDY #1                  ;On affiche
436           LDA (ADVAR),Y           ;La variable
437           STA V0                  ;Alpha. si
438           INY                     ;Cela en
439           LDA (ADVAR),Y           ;Est une
440           STA V1
441           LDY #0
442           LDA #""""
443 B8        JSR COUT
444           CPY V3
445           BEQ PRNFIN
446           LDA (V0),Y
447           ORA #$80
448           INY
449           BNE B8
450 PRNFIN    LDA #""""
451           JSR COUT
452           JMP AFPOINT
453 ;
454 FONCTIO    LDA #MESSAGE2
455           LDY /MESSAGE2
456           JSR STROUT
457           LDA OPPOIN
458           BEQ NONPOIN
459           LDY #0                  ;Si option 'p',
460           LDA (ADVAR),Y           ;On
461           STA V0                  ;Cherche
462           TAX                     ;La ligne
463           INY                     ;BASIC
464           LDA (ADVAR),Y           ;Ou
465           STA V1                  ;Est
466           LDA #MESSAGE1           ;Implante
467           LDY /MESSAGE1           ;La
468           JSR STROUT              ;Fonction.
469           DEC V1
470           LDY #255
471 PROCHE    DEY
472           LDA (V0),Y
473           BNE PROCHE
474           DEY
475           DEY
476           DEY
477           DEY
478           DEY
479 VERIF     INY
480           LDA (V0),Y
481           BNE VERIF
482 TROUVE    INY                     ;Une
483           INY                     ;Fois
484           INY                     ;Trouvee,
485           LDA (V0),Y              ;On l'affiche
486           TAX                     ;Et
487           INY                     ;On envoie
488           LDA (V0),Y              ;A l'affichage
489           JSR LINPTR              ;Des
490           JMP AFPOINT             ;Pointeurs.
491 ;
492 NONALPH    CMP #3
493           BEQ REEL
494 ;
495           LDY #0                  ;On
496           LDA (ADVAR),Y           ;Affiche
497           TAX                     ;L'entier.
498           INY
499           LDA (ADVAR),Y
500           TAY
501           TXA
502           JSR GIVAYF
503           JMP AFFAC
504 ;
505 REEL      LDY ADVAR+1             ;On envoie
506           LDA ADVAR               ;Le reel
507           JSR MOVFM               ;Dans FAC et
508 AFFAC     JSR PRNFAC              ;On l'affiche
509 ;
510 AFPOINT   LDA OPPOIN             ;On affiche
511           BEQ NONPOIN             ;Les pointeurs
512           LDA #19                 ;Si option 'P'
513           SEC                     ;En respectant
514           SBC HTAB                ;La mise
515           BCS AFPOIN1             ;En page
516           LDA #2
517 AFPOIN1   TAX
518           JSR PRBL2
519 ;
520 AFPOINO   LDA #"@"
521           JSR COUT
522           LDX ADVAR
523           LDA ADVAR+1
524           JSR PRNTAX
525 ;
526 NONPOIN   JSR CROUT               ;On effectue
527           LDA KBD                 ;Le test-clavier
528           BPL NOPAUSE
529           LDX KEYSTR              ;Si
530           CMP #$83                ;C'est CTL-C
531           BEQ RETBASIC            ;On retourne
532 S1        LDA KBD                 ;Au BASIC par BELL
533           BPL S1                  ;Sinon on fait
534           LDX KEYSTR              ;Une pause
```

```
535           CMP #$83
536           BEQ RETBASIC
537 NOPAUSE   RTS
538 RETBASIC  PLA
539           PLA
540           JMP BELL
541 ;
542 ;         ---------------------------------
543 ;         -                               -
544 ;         - AFFDIM affiche les dimensions -
545 ;         -     Stockees en $300...       -
546 ;         -                               -
547 ;         ---------------------------------
548 AFFDIM    LDY #0
549           LDA #"("
550 AGAIN     JSR COUT
551           LDA ADDIM,Y
552           INY
553           LDX ADDIM,Y
554           STY V2
555           JSR LINPTR
556           LDA #","
557           LDY V2
558           INY
559           CPY NBRDIM
560           BNE AGAIN
561           LDA #")"
562           JMP COUT
563 ;
564 ;         ---------------------------------
565 ;         -                               -
566 ;         -  Additionne l'accumulateur    -
567 ;         -        Avec ADVAR .           -
568 ;         -                               -
569 ;         ---------------------------------
570 ;
571 POIN.A    CLC
572           ADC ADVAR
573           STA ADVAR
574           LDA ADVAR+1
575           ADC #0
576           STA ADVAR+1
577           RTS
578 ;
579 ;         ---------------------------------
580 ;         -                               -
581 ;         -   IFAFNUL test la valeur      -
582 ;         -    De la variable pointee     -
583 ;         -  En vue de l'option 'O'       -
584 ;         -    Et stocke la longueur      -
585 ;         -   De la chaine si cela en     -
586 ;         -     Est une .                 -
587 ;         -                               -
588 ;         ---------------------------------
589 ;
590 IFAFNUL   LDY #0
591           LDA (ADVAR),Y
592           INY
593           CPY CHOIX
594           BEQ ENTIER1
595           INY
596           CPY CHOIX
597           BNE RET0
598           LDX OPPOIN          ;Si option 'P'
599           BEQ RET2            ;On limite la
600           CMP #13             ;Longueur de la
601           BCC RET2            ;Chaine pour
602           LDA #12             ;L'affichage
603 RET2      STA V3
604           RTS
605 ENTIER1   CLC
606           ADC (ADVAR),Y
607           BCC RET0
608           TYA
609 RET0      RTS
610 ;
611           END
```

Programme PRNVAR (BIN)

```
9290- A9 4C 8D F5 03 A9 B4 A0
9298- 92 8D F6 03 8C F7 03 A5
92A0- 73 84 74 85 6F 84 70 A5
92A8- 69 A4 6A 85 6B 84 6C 85
92B0- 6D 84 6E 60 A0 00 8C 31
92B8- 93 8C 30 93 8C 2F 93 AA
92C0- F0 59 E0 3A F0 55 E0 28
92C8- D0 04 A0 01 D0 47 E0 29
92D0- D0 0C C0 01 F0 04 A9 F0
92D8- D0 35 A0 00 F0 37 A9 01
92E0- E0 30 D0 05 8D 30 93 F0
92E8- 2C E0 50 D0 05 8D 2F 93
92F0- F0 23 E0 52 F0 11 0A E0
92F8- 24 F0 0C 0A E0 25 F0 07
9300- 0A E0 23 F0 02 A9 0F C0
9308- 01 D0 04 0A 0A 0A 0A 0D
9310- 31 93 8D 31 93 20 B1 00
9318- 4C BF 92 AD 31 93 D0 3B
9320- A9 FF 8D 31 93 4C 5B 93
9328- CF C4 C5 00 B1 AC C1 A4
9330- C4 B0 B0 B0 8D D4 E1 E2
9338- EC E5 E1 F5 A0 00 A0 E5
9340- EE A0 00 A0 E5 EE A0 EC
9348- E9 E7 EE E5 A0 00 E6 EF
9350- EE E3 F4 E9 EF EE 00 A0
9358- A5 A4 A0 A9 04 8D 2D 93
9360- CE 2D 93 30 46 4E 31 93
9368- 90 F6 A5 69 85 FB A5 6A
9370- 85 FC C5 6C 90 06 A5 FB
9378- C5 6B B0 E4 20 B0 94 A9
9380- 02 20 CE 95 EC 2D 93 D0
9388- 1A AC 2D 93 F0 0B 20 DA
9390- 95 AE 30 93 D0 03 AA F0
9398- 0A A9 28 A0 93 20 3A DB
93A0- 20 D8 94 A9 05 20 CE 95
93A8- 4C 72 93 A9 04 8D 2D 93
93B0- CE 2D 93 D0 01 60 4E 31
93B8- 93 90 F5 A5 6B 85 FB A5
93C0- 6C 85 FC A5 FC C5 6E 90
93C8- 06 A5 FB C5 6D B0 E1 20
93D0- B0 94 EC 2D 93 F0 03 4C
93D8- 9C 94 A0 04 B1 FB 0A 8D
93E0- 2C 93 A9 05 20 CE 95 A5
93E8- FB 85 F9 A5 FC 85 FA AD
93F0- 2F 93 F0 33 A9 34 A0 93
93F8- 20 3A DB A9 28 A0 93 20
9400- 3A DB A2 00 AC 2C 93 88
9408- E8 B1 F9 9D 00 03 88 CA
9410- B1 F9 9D 00 03 E8 E8 88
9418- 10 EE 20 AA 95 A9 3E A0
9420- 93 20 3A DB 20 7D 95 AE
9428- 2C 93 A9 00 CA 9D 00 03
9430- D0 FA AD 2C 93 20 CE 95
9438- AE 2D 93 E8 E0 04 D0 01
9440- E8 8E 2E 93 20 DA 95 AE
9448- 30 93 D0 03 AA F0 0D A9
9450- 28 A0 93 20 3A DB 20 AA
9458- 95 20 D8 94 AD 2E 93 20
9460- CE 95 A2 00 AC 2C 93 88
9468- 88 E8 18 BD 00 03 69 01
9470- 9D 00 03 CA BD 00 03 69
9478- 00 9D 00 03 D1 F9 D0 C4
9480- E8 C8 BD 00 03 D1 F9 D0
9488- BB A9 00 CA 9D 00 03 E8
9490- 9D 00 03 E8 88 88 88 10
9498- D0 4C C3 93 A0 02 18 B1
94A0- FB 65 FB AA C8 B1 FB 65
94A8- FC 85 FC 86 FB 4C C3 93
94B0- A2 01 A0 00 B1 FB 30 01
94B8- E8 8D 28 93 C8 B1 FB 30
94C0- 07 E8 E0 02 D0 02 A2 00
94C8- 29 7F D0 02 A9 A0 8D 29
94D0- 93 BD 57 93 8D 2A 93 60
94D8- A9 BD 20 ED FD AD 2D 93
94E0- F0 2A C9 02 D0 67 A0 01
94E8- B1 FB 85 CE C8 B1 FB 85
94F0- CF A0 00 A9 A2 20 ED FD
94F8- CC 33 93 F0 07 B1 CE 09
9500- 80 C8 D0 F1 A9 A2 20 ED
9508- FD 4C 6B 95 A9 4E A0 93
9510- 20 3A DB AD 2F 93 F0 71
9518- A0 00 B1 FB 85 CE AA C8
9520- B1 FB 85 CF A9 43 A0 93
9528- 20 3A DB C6 CF A0 FF 88
9530- B1 CE D0 FB 88 88 88 88
9538- 88 C8 B1 CE D0 FB C8 C8
9540- C8 B1 CE AA C8 B1 CE 20
9548- 24 ED 4C 6B 95 C9 03 F0
9550- 10 A0 00 B1 FB AA C8 B1
9558- FB A8 8A 20 F2 E2 4C 68
9560- 95 A4 FC A5 FB 20 F9 EA
9568- 20 2E ED AD 2F 93 F0 19
9570- A9 13 38 E5 24 B0 02 A9
9578- 02 AA 20 4A F9 A9 A4 20
9580- ED FD A6 FB A5 FC 20 41
9588- F9 20 8E FD AD 00 C0 10
9590- 13 AE 10 C0 C9 83 F0 0D
9598- AD 00 C0 10 FB AE 10 C0
95A0- C9 83 F0 01 60 68 68 4C
95A8- 3A FF A0 00 A9 A8 20 ED
95B0- FD B9 00 03 C8 BE 00 03
95B8- 8C 32 93 20 24 ED A9 AC
95C0- AC 32 93 C8 CC 2C 93 D0
95C8- E5 A9 A9 4C ED FD 18 65
95D0- FB 85 FB A5 FC 69 00 85
95D8- FC 60 A0 00 B1 FB C8 CC
95E0- 2D 93 F0 15 C8 CC 2D 93
95E8- D0 15 AE 2F 93 F0 06 C9
95F0- 0D 90 02 A9 0C 8D 33 93
95F8- 60 18 71 FB 90 01 98 60
```

Programme DEMO 1

```
5  PRINT  CHR$ (4)"BRUN PRNVAR(BIN)"
10  TEXT : HOME : PRINT "      TEST DE L
      A FONCTION PRNVAR"
15  LIST 15,80
20  DIM A$(1,2,3):V$ = ",": DEF  FN B(X)
      = 1 / X
30  FOR I = 0 TO 1
40  FOR J = 0 TO 2
```

```
50  FOR K = 0 TO 3
60 A$(I,J,K) =  STR$ (I) + V$ +  STR$ (J
      ) + V$ +  STR$ (K)
70  NEXT K,J,I
80  GET R$: & OP
```

Exécution de DEMO 1

```
X  =0                          $08E3
I  =2                          $08EA
J  =3                          $08F1
K  =4                          $08F8
V $=","                          $08D5
R $=" "                          $08FF
B  =fonction en ligne 20                  $08DC

Tableau A $(2,3,4) en $0909
A $(0,0,0)="0,0,0"                        $090F
A $(1,0,0)="1,0,0"                        $0912
A $(0,1,0)="0,1,0"                        $0915
A $(1,1,0)="1,1,0"                        $0918
A $(0,2,0)="0,2,0"                        $091B
A $(1,2,0)="1,2,0"                        $091E
A $(0,0,1)="0,0,1"                        $0921
A $(1,0,1)="1,0,1"                        $0924
A $(0,1,1)="0,1,1"                        $0927
A $(1,1,1)="1,1,1"                        $092A
A $(0,2,1)="0,2,1"                        $092D
A $(1,2,1)="1,2,1"                        $0930
A $(0,0,2)="0,0,2"                        $0933
A $(1,0,2)="1,0,2"                        $0936
A $(0,1,2)="0,1,2"                        $0939
A $(1,1,2)="1,1,2"                        $093C
A $(0,2,2)="0,2,2"                        $093F
A $(1,2,2)="1,2,2"                        $0942
A $(0,0,3)="0,0,3"                        $0945
A $(1,0,3)="1,0,3"                        $0948
A $(0,1,3)="0,1,3"                        $094B
A $(1,1,3)="1,1,3"                        $094E
A $(0,2,3)="0,2,3"                        $0951
A $(1,2,3)="1,2,3"                        $0954
```

Documentation de tables de shapes

Erick Ringot

Ce programme permet de documenter des tables de formes, en précisant :

- La longueur de la table.
- Le nombre de formes.
- Pour chaque forme, l'adresse relative par rapport au début de la table.

Il permet également de représenter les formes à l'écran ou sur imprimante.

Principe du programme

- La table est chargée en $4000 = 16384, soit en page HGR2.
- Le premier octet est N, nombre de formes composant la table.
- Pour chaque forme, on lit l'adresse relative (groupe de deux octets consécutifs).
- On vérifie que l'octet précédant cette adresse est bien nul (sinon, on n'a pas affaire à une table de formes).
- Pour déterminer la longueur de la table, on parcourt les octets de la dernière forme, jusqu'à l'obtention d'un zéro terminant à la fois cette forme et la table.

Variables

A = 16384 (adresse de chargement)
ADRESSE(i) : tableau des adresses relatives des formes
AH,AL : octets fort et faible de l'adresse relative courante
D : adresse de fin de table
ER : code de l'erreur éventuelle
IF%,IR% : indicateurs d'usage de sous-programmes d'édition
I,F,L : indices de boucle
LL : longueur de la table de formes
LSB%,MSB% : octets faible/fort de la longueur LL
MESSAGE$: message renseignant sur le déroulement du programme
MX,MY : marges de la page haute résolution
N : nombre de formes
N$: nom de la table
O%(i), O$(i), i variant de 1 à 4. Quartets composant LL, en décimal, en hexa.
PX,PY : pas horizontal, vertical dans la page HGR
Q,P : nombre de lignes, de colonnes dans la page HGR
X,Y : coordonnées de la forme courante dans la page HGR
ZE : octet normalement nul si le fichier N$ est bien une table

Remarque : le sous-programme d'impression est écrit pour l'imprimante SEIKOSHA GP-100a, connectée sur le slot 1. La signification des codes de contrôle utilisés est la suivante :

CHR$(20) <CTRL-T> : Empêche l'écho à l'écran
CHR$(14) <CTRL-N> : Ecriture double largeur
CHR$(15) <CTRL-O> : Rétablit la simple largeur
CHR$(17) <CTRL-Q> : Effectue la copie de la page haute résolution

■

```
100  REM ---------------------
110  REM  DOCUMENTATION
120  REM  DES TABLES DE
130  REM  FORMES
140  REM ---------------------
150  REM  ***  DOC/SHAPE  ***
160  REM  -1- INITIALISATION
170  LOMEM: 24576: REM APRESHGR2
180  ONERR  GOTO 1150
190  TEXT : HOME
200  PRINT "NOM DE LA TABLE...."
210  INPUT "";N$: IF N$ = "" THEN  END
220  GOTO 1010
230  REM  -2- MESSAGE
240  VTAB 20: INVERSE : PRINT MESSAGE$: FOR
       I = 1 TO 1000: NORMAL : NEXT : VTAB
       20: CALL  - 868: RETURN
250  REM  -3- VERIFICATION TTABLE
260 A = 4 * 16 ^ 3: POKE 232,0: POKE 233,64
       : REM ADRESSE TABLE
270 MESSAGE$ = "CHARGEMENT DE " + N$: GOSUB
       240
280  PRINT  CHR$ (4)"BLOAD"N$",A"A
290 N =  PEEK (A): REM NOMBRE DE FORMES
300  DIM ADRESSE(N): REM TABLEAU DES ADRESS
       ES RELATIVES
310  FOR I = 1 TO N
320 AL =  PEEK (A + 2 * I): REM ADRESSE REL
       ATIVE OCTET FAIBLE
330 AH =  PEEK (A + 2 * I + 1): REM ADRESSE
       RELATIVE OCTET FORT
340 ADRESSE(I) = AH * 256 + AL: IF A + ADRE
       SSE(I) > 2 ^ 16 THEN 1210: REM  ADRE
       SSE RELATIVE
350 ZERO = 0: IF I > 1 THEN ZERO =  PEEK (A
       + ADRESSE(I) - 1): REM  DERNIER OCT
       ET DE LA FORME PRECEDENTE
360  IF ZERO <  > 0 THEN  GOTO 1210: REM CE
       N'EST PAS UNE TABLE DE FORMES !
370  NEXT I
380 MESSAGE$ = N$ + " EST UNE TABLE": GOSUB
       240
390  RETURN
400  REM  -4- LONGUEUR DE LA TABLE
410 D = A + ADRESSE(N): REM ADRESSE DERNIER
       E FORME
420  IF  PEEK (D) <  > 0 THEN D = D + 1: GO
       TO 420: REM LA TABLE SE TERMINE PAR
       UN ZERO
430 LL = D - A + 1: REM LGUEUR DECIMALE
440  REM  * CONVERSION HEXA *
450 MSB% = LL / 256:LSB% = LL - 256 * MSB%
460 O%(1) = MSB% / 16:O%(2) = MSB% - 16 * O
       %(1):O%(3) = LSB% / 16:O%(4) = LSB%
       - 16 * O%(3): REM QUARTETS COMPOSANT
       L'ADRESSE
470  FOR I = 1 TO 4:O$(I) =  STR$ (O%(I))
480  IF O%(I) > 9 THEN O$(I) =  CHR$ (O%(I)
       + 55)
490  NEXT
500  RETURN
510  REM  -5- REPRESENTATION
520  IF IR% = 1 THEN  POKE  - 16297,0: POKE
       - 16302,0: POKE  - 16304,0: GOTO 7
       10
530  HGR : POKE  - 16302,0
540 IR% = 1
550  HCOLOR= 3: SCALE= 1: ROT= 0
560 Q =  INT ( SQR (N)): REM NB.LIGNES
570 PY =  INT (192 / Q): REM INTERVALLE EN
       Y
580 MY = (192 - (Q - 1) * PY) / 2: REM MARG
       E EN Y
590 P =  INT (N / Q): IF N > P * Q THEN P =
       P + 1
600 PX =  INT (280 / P)
610 MX = (280 - (P - 1) * PX) / 2
620  FOR L = 1 TO Q
630  FOR F = 1 TO P
640 I = F + (L - 1) * P
650  IF I > N THEN 710
660 Y = MY + (L - 1) * PY: REM  COTE
670 X = MX + (F - 1) * PX: REM ABSCISSE
680  DRAW I AT X,Y
690  NEXT F
700  NEXT L
710  GET A$: TEXT
720  RETURN
730  REM  -6- EDITION IMPRIMANTE
740  REM  SEIKOSHA GP100 SLOT 1
750  PRINT  CHR$ (13) CHR$ (4)"PR#1"
760  PRINT  CHR$ (20)
770  PRINT  CHR$ (14) TAB( 20 -  LEN (N$) /
       2)N$ CHR$ (15): PRINT
780  GOSUB 860
790  PRINT
800  POKE  - 12317,0
810  PRINT  CHR$ (17)
820  PRINT  CHR$ (4)"PR#0"
830  RETURN
840  REM  -7- TABLEAU D'ADRESSES
850  TEXT : HOME :IE% = 1
860  PRINT "LONGUEUR, OCTETS"
870  PRINT "DECIMALE       -> "LL
880  PRINT "HEXADECIMALE   -> $"O$(1)O$(2)
       O$(3)O$(4)
890  PRINT : PRINT "NOMBRE DE FORMES  "N
900  VTAB 9
910  PRINT "FORME"; TAB( 15);"ADRESSE"
920  POKE 34,10: VTAB 11
930  FOR I = 1 TO N
940  PRINT  TAB( 5 -  LEN ( STR$ (I)));I; T
       AB( 20 -  LEN ( STR$ (AD(I))));ADRES
       SE(I)
950  IF IE% = 0 THEN 970
960  IF I = N OR I = 10 *  INT (I / 10) THE
       N  GET A$: HOME
970  NEXT
980 IE% = 0: TEXT
990  RETURN
1000  REM  -0- PROGRAMME PPAL
1010  GOSUB 260
1020  GOSUB 410
1030  HOME
1040  PRINT "COMMANDES": PRINT
1050  PRINT "DESSIN DES FORMES......1"
1060  PRINT "TABLE D'ADRESSES.......2"
1070  PRINT "IMPRESSION.............3"
1080  PRINT "AUTRE TABLE............4"
1090  PRINT "TERMINE................5"
1100  GET A$:A% =  VAL (A$)
1110  ON A% GOSUB 520,850,750,1120,1130: GO
       TO 1030
1120  RUN
1130  TEXT : HOME : END
1140  REM  -8- TRAITEMENT ERREURS
1150 ER =  PEEK (222):LI =  PEEK (218) +  P
       EEK (219) * 256
1160  IF ER = 6 THEN MESSAGE$ = N$ + " INCO
       NNU": GOSUB 240: RUN
1170  TEXT
1180  PRINT  CHR$ (13) CHR$ (4)"PR#0"
1190  HOME : PRINT "ERREUR TYPE "ER: PRINT
       "LIGNE       "LI: END
1200  REM  -9- FICHIER NON-TABLE
1210 MESSAGE$ = N$ + " N'EST PAS UNE TABLE"
       : GOSUB 240: RUN
```

FAITES VIVRE VOTRE APPLE II

DES LIVRES ET DES DISQUETTES POUR VOTRE APPLE

Les disquettes d'accompagnement des Editions du P.S.I. sont destinées aux personnes qui n'ont pas le temps de taper elles-mêmes les programmes. Chaque disquette constitue la fidèle adaptation sur Apple II des listings proposés dans l'ouvrage de référence. Ces disquettes doivent être considérées comme une aide au lecteur et non comme un progiciel.
Attention! les consignes d'utilisations des programmes enregistrés sur les disquettes sont précisées dans les ouvrages de référence. Pour chaque disquette, l'achat du livre est donc indispensable.
La disquette seule: 195,00 FF (port et emballage compris)

Pour Apple II, II plus, //e - Dos 3.3 - version 48 K ou plus :

Disquette : **"Nouvelle comptabilité sur Apple II"** tome 1
Disquette : **"Nouvelle comptabilité sur Apple II"** tome 2
Disquette : **"La paie et ses annexes"**
Disquette : **"Outils financiers et comptables"**
Disquette : **"Modèles pratiques de décision"** tome 1
Disquette : **"Modèles pratiques de décision"** tome 2
Disquette : **"L'Apple et ses fichiers"** tome 1.
Disquette : **"Modèles d'expression graphique"**
Disquette : **"Mathématiques et Statistiques"**
Disquette : **"Méthodes de calcul numérique"** tome 1 et 2
Disquette : **"Le Basic et l'école"** tome 1
Disquette : **"Le Basic et l'école"** tome 2
Disquette : **"Visicalc sur Apple II"**
Attention : Disquette maîtresse Visicalc indispensable.

Pour Apple II plus, //e - Dos 3.3 - version 64 K ou plus

Disquette : **"Multiplan pour Apple II"**
Attention : Disquette maîtresse Multiplan indispensable.

Pour Apple II, II plus, //e avec Système Pascal

Disquette : **"Bibliothèque scientifique en Pascal"**

NOUVEAU PLAN COMPTABLE

Nouvelle comptabilité sur Apple II tome 1

par Serge et Gérard Lilio

Voici un ensemble complet de programmes de Comptabilité sur Apple II adapté au Nouveau Plan Comptable, pour petites entreprises, professions libérales, artisans, commerçants. On y trouve des programmes permettant l'édition des livre-journal, grand-livre, balance, bilan; la possibilité d'éditer les livres de banque, de TVA, de recettes, ainsi qu'un programme de détection des erreurs de frappe. Capacité du système : compte à 6 chiffres, 400 lignes par plan.

Le livre : 184 pages - 120,00 FF
La disquette : 195,00 FF

Nouvelle comptabilité sur Apple II tome 2 NOUVEAU

par Serge et Gérard Lilio

Ce second tome contient trois programmes complémentaires des programmes de comptabilité du tome 1 : - visualisation graphique tridimensionnelle des recettes et dépenses avec calcul des ratios. - tableaux des amortissements. - calcul d'impôts, ainsi qu'un programme de comptabilité générale pour les utilisateurs de disques 8 pouces, disques durs ou d'une seule unité 5 pouces 1/4.

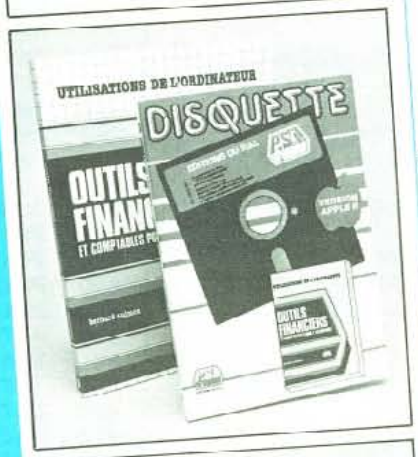
Le livre : 144 pages - 110,00 FF
La disquette : 195,00 FF

La paie et ses annexes

par Jean Michel Jégo

Cet ouvrage présente l'analyse et la programmation des problèmes de la paie et des charges salariales : fichiers de personnel, calcul des cotisations, élaboration d'un bulletin de salaire avec congés payés. En fin d'ouvrage on aboutit à la tenue d'un cahier des charges salariales tel qu'il peut se présenter dans une petite entreprise. Attention : beaucoup de simplifications ont été apportées par rapport aux situations très diverses rencontrées par les professionnels de la paie.

Le livre : 136 pages - 110,00 FF
La disquette : 195,00 FF



Outils financiers et comptables pour l'entreprise

par Bernard Sulmon

Sont traités dans ce livre : l'analyse des coûts marginaux, le calcul de seuils de rentabilité, le Direct Costing, le calcul des ratios ou du Fond de Roulement minimum, la rentabilité des investissements et la gestion des stocks. Chaque programme est proposé avec un exemple d'application.

Le livre : 156 pages - 100,00 FF
La disquette : 195,00 FF

Multiplan pour Apple II plus et //e NOUVEAU

par Hervé Thiriez

Multiplan est un progiciel qui permet de gérer plusieurs tableaux simultanément; cet ouvrage sera pour les possesseurs d'ordinateurs Apple II Plus ou //e un véritable guide d'utilisation de Multiplan grâce à des exemples progressifs et à de nombreux cas d'application (gestion de portefeuilles, de copropriété, feuille de paie, impôts, tableaux de bord, etc.).

Le livre : 216 pages - 100,00 FF
La disquette : 195,00 FF

Visicalc sur Apple

par Hervé Thiriez

Après une présentation progressive du modèle Visicalc, l'ouvrage étudie de nombreux cas d'application : feuille d'impôt, gestion de copropriété, paye, facturation..., permettant d'introduire les différentes instructions et astuces d'utilisation.

Le livre : 176 pages - 90,00 FF
La disquette : 195,00 FF

L'Apple et ses fichiers

par Jacques Boisgontier

Pour apprendre progressivement la programmation des applications utilisant les fichiers l'ouvrage commence par une présentation concise et illustrée des commandes du Système d'Exploitation Disque et des instructions du Basic Applesoft. Les instructions des fichiers séquentiels et à accès direct sont ensuite décrites ainsi que leur utilisation. Des méthodes pratiques, souvent mal connues, montrent comment utiliser au mieux des fichiers à accès direct : accès indexé, liste inverse. Une vingtaine de programmes illustrent l'utilisation de ces techniques.

Le livre : 176 pages - 90,00 FF
La disquette : 195,00 FF

Modèles pratiques de décision
Tome 1

par Jean-Pierre Blanger

Cet ouvrage vise l'automatisation du processus de la prise de décision. Les différentes techniques exposées sont complétées d'un exemple et d'un programme en Basic pour permettre au lecteur une rapide maîtrise des modèles présentés et leur intégration à de nombreuses applications (simulation, gestion, intelligence artificielle...).

Le livre : 144 pages - 90,00 FF
La disquette : 195,00 FF

Modèles pratiques de décision
Tome 2

par Jean-Pierre Blanger

Ce tome 2 de Modèles pratiques de décision offre un nouvel éventail de techniques visant l'automatisation du processus de la prise de décision. Chacun des vingt modèles présentés donne lieu à un bref exposé, un exemple et un programme en Basic standard.

Le livre : 176 pages - 90,00 FF
La disquette : 195,00 FF

Modèles d'expression graphique

par Jean-Pierre Blanger

Cet ouvrage expose un ensemble de techniques visant la mise en œuvre des possibilités graphiques des ordinateurs individuels. Il permet au débutant comme à l'amateur chevronné d'aborder la résolution de problèmes de plus en plus complexes (tracés d'ellipse, rotation de polygone, hachurage de surface...). Les modèles d'expression graphique, écrits en Basic Applesoft, sont abondamment commentés et facilement adaptables à d'autres ordinateurs individuels.

Le livre : 232 pages - 130,00 FF
La disquette : 195,00 FF

Mathématiques et statistiques

par Hervé Haut

Cet ouvrage est un recueil de 16 logiciels de base (niveau supérieur) tant en mathématiques qu'en statistiques. Chaque problème traité comporte une introduction numérique, un exposé de la technique de programmation utilisée, un organigramme détaillé et un programme complet en Basic suivi d'un exemple d'utilisation.

Le livre : 272 pages - 100,00 FF
La disquette : 195,00 FF

Bibliothèque scientifique en Pascal

par Hervé Haut

Les procédures proposées dans cet ouvrage sont conçues pour être implantées dans le System Library du langage. L'utilisateur disposera d'une bibliothèque enrichie, permettant une résolution aisée et performante d'un grand nombre de problèmes mathématiques et statistiques souvent rencontrés dans les programmes scientifiques.

Le livre : 152 pages - 90,00 FF
La disquette : 195,00 FF

Méthodes de calcul numérique

par Claude Nowakowski

Tome 1 : Equations non linéaires, polynômes, calcul matriciel, interpolation, intégration et équations différentielles: pour chaque problème les différentes méthodes de calcul numérique sont étudiées. Ces algorithmes sont illustrés par un organigramme, un programme en Basic et un exemple d'exécution. Le Tome 2 donne des algorithmes plus élaborés ou plus subtils et aborde de nouveaux thèmes : approximation des fonctions, problèmes aux limites, équations aux dérivées partielles.
Chaque algorithme est programmé en Basic et en Pascal.

Tome 1 : 144 pages - 90,00 FF
Tome 2 : 184 pages - 110,00 FF
La disquette pour les 2 tomes : 195,00 FF

Le Basic et l'école - Tome 1

par Jacques Gouet

Le livre décrit un ensemble de programmes en Basic, destinés aux professeurs, aux parents et aux élèves. Grammaire, mathématiques, conjugaisons française, anglaise, sont autant d'exemples d'application qui font de cet ouvrage un véritable outil d'enseignement et d'initiation, à l'école ou à la maison.

Le livre : 192 pages - 120,00 FF
La disquette : 195,00 FF

Le Basic et l'école - Tome 2

par Jacques Gouet

Pourvu que vous possédiez un lecteur de disquette, vous allez pouvoir créer et gérer vous-même vos Questionnaires à Choix Multiples et vos exercices, suivre les notes obtenues, et concevoir des carnets de vocabulaire étranger, à l'école comme à la maison.

Le livre : 160 pages - 110,00 FF
La disquette : 195,00 FF

P.S.I. SUISSE
Case postale
Route neuve 1
1701 Fribourg
SUISSE
Tél. : (037) 23.16.28
CCP 17.5684

au Canada
SCE Inc
65, Avenue Hillside
Montreal (Westmount)
Quebec H3Z 1W1
Tel. : (514) 935 13 14

P.S.I. DIFFUSION
BP 86 - 77402 Lagny-S/Marne Cedex
FRANCE
Téléphone (6) 006.44.35

P.S.I. BENELUX
5, avenue de la Ferme Rose
1180 Bruxelles
BELGIQUE
Téléphone (2) 345.08.50

Table de conversions en francs belges et francs suisses
90 FF = 605 FB - 28.40 FS
100 FF = 770 FB - 31.50 FS
110 FF = 850 FB - 34.60 FS
120 FF = 925 FB - 37.60 FS
130 FF = 1000 FB - 40.60 FS
DISQUETTES 195 FF = 1500 FB - 61.50 FS

Envoyer ce bon accompagné de votre règlement à P.S.I. DIFFUSION ou, pour la Belgique et le Luxembourg à P.S.I. BENELUX ou, pour la Suisse à P.S.I. SUISSE.

DESIGNATION	NOMBRE	PRIX
	TOTAL	

par avion : ajouter 8 FF (75 FB) par livre

Signature (obligatoire pour paiement par carte de crédit)

☐ Paiement par chèque joint ☐ Paiement en FF par carte bleue VISA (à P.S.I. DIFFUSION uniquement) paiement supérieur à 50 FF

N° _____ Date d'expiration ____

NOM _____ PRENOM _____

rue _____ n° ____

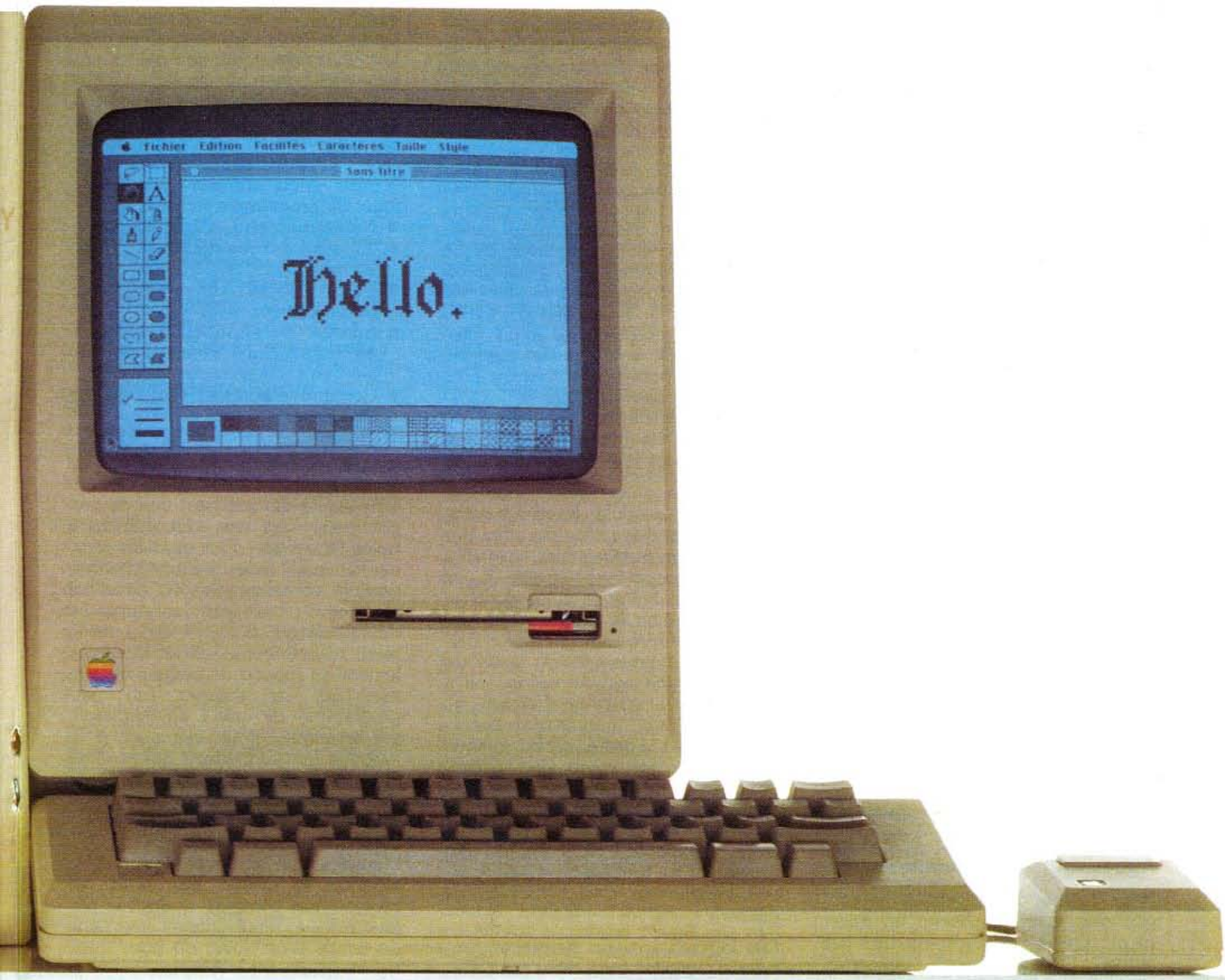
Code postal _____ Ville _____

AGRAM

Apple Computer, Inc. est le licencié de la marque Macintosh.

Il était temps qu'un capitaliste

fasse une révolution.



N'apprenez plus à devenir une machine, Apple a inventé Macintosh.

Pom's a vu LIGHT 1

Alexandre Duback

LIGHT 1 est un ensemble de logiciels d'aide à la programmation diffusé par la société Les Années Nouvelles. Si certains de ses modules sont à la portée de tous, ou presque (catalogue général, recherche de Tokens...), l'exploitation des méthodes de "pistage" de programmes qu'il propose sera vraisemblablement réservée aux utilisateurs "professionnels" des Apple II et //e.

Le produit se présente sous la forme d'une disquette "logiciels", accompagnée d'une documentation imprimée. L'apprentissage de son fonctionnement peut être complété par des stages de deux jours organisés par Les Années Nouvelles. Nous présenterons ici rapidement les différentes fonctions offertes, en insistant sur celles qui nous paraissent les plus utiles ou les plus originales.

Accès direct aux disquettes

Le module baptisé ZAPPLE 7 permet toutes les opérations de manipulation de disquettes (formatées en DOS 3.3) au niveau du secteur :

- Lecture et affichage du contenu d'un secteur octet par octet, avec possibilité de modification et de ré-écriture.
- Analyse et manipulation du directory.
- Etat d'occupation d'une disquette.
- Recherche de fichiers et récupération possible de fichiers détruits.
- Identification, nettoyage et transfert de secteurs (d'un disque à un autre).
- Recherche de suites d'octets ou de caractères sur la disquette, pour savoir, par exemple, dans quels octets de quel secteur a été enregistrée la séquence "DUPONT".
- Impression de suites de secteurs, ainsi que des différents secteurs occupés par un fichier donné.

Ce programme peut donc aider à la récupération de disquettes endommagées sur le plan logique, mais aussi à la mise au point de logiciels dont les opérations de lecture et écriture sur disquette ne produisent pas les résultats espérés.

Autres utilitaires "ponctuels"

Nous regroupons ici différents programmes assurant chacun un traitement précis et indépendant des autres. Les traitements concernés sont :

- Constitution et exploitation d'un catalogue général de disquettes (liste alphabétique des fichiers, recherche d'un fichier donné, liste des fichiers dont le nom commence par une séquence donnée, tri selon le type de fichiers, liste des noms affectés aux disquettes enregistrées dans le catalogue, le tout à l'écran ou sur imprimante).
- Affichage des informations de chargement de fichiers binaires (adresse de départ et longueur).
- Recherche de Tokens, de suites de caractères (y compris caractères de contrôle) et de variables dans un programme en Applesoft.
- Recherche de séquences d'octets en mémoire centrale.
- Impression désassemblée de codes hexadécimaux et recherche d'octets inutilisés dans une portion de mémoire.
- Comparaison de deux séquences d'octets en mémoire ou sur disquette.

Le pistage de programmes

Ce traitement très complet (et relativement complexe) peut être appliqué à des programmes en Applesoft ou en langage machine. Il permet très exactement de suivre pas à pas les étapes de déroulement d'un programme et de savoir quelles adresses et quelles routines du système sont utilisées, de suivre l'appel et l'exécution des différentes séquences du programme lui-même, de connaître le contenu des registres après chaque instruction...

Un premier module, baptisé Ariane, construit sur disquette une table des adresses d'instructions par lesquelles passe le programme étudié. Il est possible de définir la zone dont on veut ainsi pister le déroulement : fenêtre d'une adresse à une autre, mémorisation uniquement à partir du passage à une adresse précise ou à un numéro d'instruction désigné - les instructions sont numérotées par le logiciel dans l'ordre chronologique d'apparition au cours de l'exécution... On peut de même "masquer" une séquence d'adresses pour le pistage, afin de ne pas encombrer la table avec le compte-rendu du déroulement de routines non significatives (attente au clavier, boucle de délai...).

Des logiciels annexes permettent ensuite d'exploiter cette table pour établir une première "radiographie" du programme étudié : impression de la succession des adresses d'instructions empruntées, tableau des adresses distinctes et de leur nombre d'occurrences (combien de fois passe-t-on par les routines CHRGET, NEWSTT, CHKNUM...), recherche des numéros d'instructions où apparaît telle ou telle adresse remarquable... Pour faciliter l'interprétation de ces résultats, vous pouvez constituer des tables d'adresses remarquables en fonction des opérations sur lesquelles se centre votre intérêt. Une table de cette nature permet d'affecter un nom à chaque adresse d'instruction que vous jugez particulièrement significative, par exemple, CHRGET pour l'adresse $B1, NEWSTT pour l'adresse $D7D2...

Cette première exploitation des résultats d'Ariane vous guidera ensuite dans l'utilisation du module Chalut, qui réalise le pistage pas à pas proprement dit. Sur les séquences critiques d'exécution précédemment isolées, Chalut vous donnera un compte-rendu complet du déroulement du programme :

- Numéro d'instruction
- Contenu des registres A, X, Y et S en hexa.
- Valeur des bits du registre d'état du processeur.
- Adresse de l'instruction et libellé en hexadécimal et en désassemblé.
- Contenu d'adresses remarquables désignées.

L'ensemble de ces modules produit donc finalement un rapport très détaillé sur le déroulement de vos programmes. Ceci vise tout d'abord à isoler les causes d'un mauvais fonctionnement, mais peut également servir à connaître plus en détail les routines du moniteur de l'Apple ou de l'Applesoft, puisqu'il est parfaitement réalisable d'en pister le déroulement au moyen de petits programmes qui les utilisent (un petit programme de calcul en Applesoft, par exemple, pour examiner à la loupe le fonctionnement des routines "virgule flottante", isoler les causes du "bug" signalé dans Pom's 11 et le résoudre, comme l'ont fait les auteurs du logiciel...).
L'emploi d'un tel système ne semble toutefois se justifier que pour des programmes complexes, dont le pistage manuel représenterait une perte de temps prohibitive, ou pour quelqu'un éprouvant le besoin de rentrer très précisément dans l'étude du fonctionnement de l'Apple. Il sera donc vraisemblablement réservé à des usages professionnels.

L'ensemble du logiciel accompagné de sa documentation est proposé au prix de 1950 F TTC. Les journées de formation, quant à elles, reviennent à 1000 F TTC par jour; elles peuvent porter à la fois sur l'apprentissage de LIGHT 1 lui-même et sur la présentation de la structure et du fonctionnement "soft" de l'Apple, dont une connaissance minimale est de toute façon requise pour tirer un meilleur parti du produit.

Le prix, relativement élevé malgré tout, peut trouver sa justification dans l'intégration des routines utiles à l'analyse de programmes que propose LIGHT 1 (vous disposez en un seul système d'outils pour la manipulation des disquettes, la recherche de Tokens, la comparaison de séquence d'octets...). En outre, la méthode de "pistage" conduit effectivement à une résolution très fine de difficultés dont l'étude par d'autres moyens pourrait constituer une tâche particulièrement fastidieuse, sinon tout à fait décourageante.
Quant à l'apprentissage de l'utilisation du produit, la documentation fournie peut y suffire. Le recours aux journées de formation dépend finalement du temps que vous entendez y consacrer et de votre niveau de connaissances en matière d'Apple "soft".

■

Thinktank à l'essai

Guy Lapautre

Un logiciel pas comme les autres

Thinktank est très large d'idées, ce qui est bien la moindre des choses pour un logiciel de "traitement des idées". Pas besoin de classement préalable, inutile de structurer ce qu'on veut lui confier (enfin presque...). N'exagérons rien, il ne mettra quand même pas d'ordre dans un chaos originel.

Mais il est plein de ressources : on peut développer ici, condenser là, changer la hiérarchie entre les objets traités, créer des niveaux supplémentaires, trier alphabétiquement une table des matières, rechercher un quelconque vocable dans tout ou partie d'un ensemble, même particulièrement touffu (ce n'est pas toujours très rapide), provoquer des éditions ou des duplications sélectives, fusionner, communiquer avec des fichiers tiers...

Mais attention ! Si Thinktank (réservoir d'idées) permet tout cela, ne vous méprenez pas pour autant. Il n'est pas question ici d'intelligence artificielle, de démarche inductive ou déductive. Comme dans la célèbre auberge espagnole, on y trouve ce qu'on y apporte. Mais il est sûrement plus facile de l'y retrouver que dans ladite auberge.

Ce n'est pas non plus un gestionnaire de fichiers au sens classique du terme. Pas de notions d'enregistrements et de champs ou d'indexation. Et pourtant, il permet de stocker des informations et de les retrouver. Ce n'est pas davantage un traitement de texte. Cependant, il possède un éditeur et un générateur de formats d'édition. Au fond, c'est peut-être encore son nom de "réservoir d'idées" qui le définit le mieux.

La documentation comporte un "tutorial" qui explique sur un exemple les grands principes de fonctionnement. Un manuel de référence plus complet lui fait suite.

Sur le plan de la commodité d'emploi, un guide par menus enchaînés rend les principales options aisément accessibles. Si vous avez de la mémoire, vous pouvez accélérer le processus en tapant, à tout stade ou presque, la lettre initiale de la commande que vous voulez mettre en oeuvre. Et, si vous avez tout oublié (ou presque), une commande unique "/" vous permet, de tout point du programme, de passer au menu (de même que la touche ESC vous fait sortir de l'option en cours).

On pourra faire deux critiques au niveau de la commodité d'emploi. La première tient justement à cette commande ESC : elle fait généralement sortir d'une option en abandonnant ce qui n'a pas été rendu définitif. En revanche, quand vous avez fini de rédiger un "paragraphe", ESC est la sortie normale qui sauvegarde le travail (alors que RETURN ne remplit que la fonction de saut de ligne). La seconde concerne l'éditeur, peu classique (et en fait peu puissant). Par exemple, la flèche arrière efface le caractère qui précède le curseur, alors que la touche DEL efface le caractère sous le curseur.

Un exemple d'utilisation

Ma femme et moi manquons quelque peu de mémoire (en tous cas pour les besoins de la cause... mais cela doit être également un peu vrai dans la vie). Aussi avons-nous confié à Thinktank tout un lot d'idées qui nous sont passées par la tête, afin de disposer d'un pense-bête permanent pour les deux mois de vacances que nous comptons royalement nous offrir.

Grande bridgeuse devant l'éternel, ma femme a d'abord ouvert une rubrique BRIDGE, sans trop savoir ce qu'elle allait y mettre. Un peu plus tard, elle a "fourré" dans cette rubrique les noms des personnes avec lesquelles elle est accoutumée à bridger pendant les vacances, avec quelques indications générales pour chacune d'entre elles.

Elle a aussi mis "dans le même panier" quelques idées sur les compartiments du jeu où elle a envie de se perfectionner. Ce qui l'a conduit à reprendre quelques noms de joueurs pouvant l'y aider. Pourquoi pas ? Thinktank s'y retrouvera toujours, même si une information unique figure à plusieurs endroits.

De mon côté, je me suis penché sur le problème des gens qui viendraient nous voir. Entre-temps d'ailleurs, j'ai pensé qu'il était peut-être préférable de mettre un peu à part la famille (elle comporte parfois des gens susceptibles...). Un petit changement dans l'ordre des rubriques y pourvoit. J'ai en outre noté au passage une interférence avec le bridge de Madame : il y a des parents et amis qui sont aussi bridgeurs ! Certains avaient été pris en compte dans la rubrique Bridge, d'autres non. Ma femme a décidé de n'y garder que les bridgeurs habituels. De toute façon, Thinktank les retrouvera tous, si on le lui demande gentiment. Notez qu'il eût été possible de dupliquer dans la rubrique bridge les données concernant le bridge dans la famille ou chez les amis...

Vous direz peut-être que, même si Thinktank y met de l'ordre, l'écran contenant des données aussi disparates doit s'avérer difficile à lire. Une des fonctionnalités les plus originales du logiciel est "étendre / restreindre", qui permet, en se plaçant à n'importe quel niveau, d'étendre tout ce qui concerne le niveau N−1, qui apparaît alors sur l'écran, ou au contraire de le restreindre, ce qui le fait disparaître de l'écran (mais évidemment pas de la mémoire). Si vous voulez étendre ou restreindre sur plusieurs niveaux à la fois, il suffit de faire précéder la commande du nombre de niveaux voulus. La commande elle-même est bien choisie au plan mnémotechnique : "<"

pour étendre ou élargir, ">" pour restreindre ou réduire.

Quant à la commande d'impression, elle lance sur imprimante, comme titre, la rubrique sur laquelle se trouve le curseur, et comme texte, tout ce qui est d'un niveau inférieur dans cette rubrique. Si vous vous placez en tête, sur le nom du "fichier", vous éditez alors la totalité de celui-ci.

Fac Simile de quelques éditions

(1) - L'ensemble des titres
(2) - Les titres de la partie BRIDGE, après ajout d'un chapitre
(3) - Edition complète de la partie LES AMIS QUI VIENDRONT
(4) - Table des matières de cette partie
(5) - Une autre édition complète, dans une forme style "rapport"
(6) - La table des matières correspondantes ■

16-MAY-84 VACANCES D'ETE PAGE 1

BRIDGE
BERTHE
ANATOLE
CLAUDE
DENIS
LUI
ELLE
FAMILLE
ANATOLE
ERNEST
PARENTS
ENFANTS
EUX
BEBE
AMIS QUI VIENDRONT
DENIS
FRANCOIS
EN JUILLET
EN AOUT
LUI
ELLE
SI ELLE EST EN FORME
SINON
GASTON
HIPPOCAMPE
PARENTS
ENFANTS
GARCONS
FILLE

1

16-MAY-84 BRIDGE PAGE 1

BERTHE
ANATOLE
CLAUDE
DENIS
LUI
ELLE
LE TEXAS FACON JR
LA VARIANTE DYNASTIE

2

16-MAY-84 AMIS QUI VIENDRONT PAGE 1

DENIS
Elle surement - prevoir bridge.
Lui peut-etre. S'il vient, lui faire rencontrer CLAUDE

FRANCOIS
EN JUILLET
Passera seulement 1 ou 2 jours lors de sa venue au congres de pisciculture sur terre aride.
Aura sans doute envie de rencontrer des passionnes de la region.
Demander a BERTHE, qui doit etre au courant.
Si possible, organiser une reunion a la maison (s'il n'y a pas plus de 8 personnes!
Sinon, voir s'il est possible de reserver un coin tranquille dans un restaurant

EN AOUT
Ils viendront tous les deux passer une semaine

LUI
Adore la peche au tout gros - retenir bateau

ELLE
SI ELLE EST EN FORME
Prevoir moult visite de vieilles pierres

SINON
Bridge, mais cela ne sera pas drole

GASTON
Essayer de les avoir en meme temps que les FRANCOIS

3

HIPPOCAMPE
PARENTS
Verifiez s ils comptent venir en Juillet ou en Aout.
N'apprecieraient pas du tout si nous ne pouvions pas les recevoir

ENFANTS
GARCONS
Il y en a 4 - Peut-etre 5?
Trouver un endroit ou ils puissent planter leur tente.
Pas trop loin pour qu ils puissent venir facilement quand ils en ont envie.
Pas trop pres pour qu ils n aient pas sans arret leurs parents sur le dos.

FILLE
Viendra peut-etre avec une copine. Elles seront un peu serrees dans le petit lit.
A moins qu'elle ne vienne avec un "copain" ?
Voir attitude des parents...

16-MAY-84 AMIS QUI VIENDRONT PAGE 1

4

16-MAY-84 FAMILLE 5 PAGE 1

1: ANATOLE
Ne pas oublier de les inviter a manger des ecrevisses, ils adorent cela!
(elle surtout...)

2: ERNEST
2.1: PARENTS
A traiter avec tous les honneurs dus a leur rang et a leur age

2.2: ENFANTS
2.2.1: EUX
Pas de probleme. Bons copains.
Prevoir une sortie a frais partages.

2.2.2: BEBE
Acheter avant de partir le cadeau pour la naissance.
Probleme: C'est Claude. Garcon ou fille?

16-MAY-84 FAMILLE PAGE 1

6

PEEKs à gogo

Roland Jost

L'instruction PEEK permet la saisie du contenu d'une adresse en mémoire à partir d'un programme Basic. Définissons la fonction :
DEF FN PE(X) = PEEK(X) + 256 * PEEK(X+1)
qui permet la saisie d'un nombre supérieur à 255. Cette instruction devra figurer dans votre programme Basic.

Cette liste de PEEKs vous permettra d'une part de mieux comprendre certains programmes Applesoft, d'autre part de simplifier l'écriture de vos propres programmes.

Page zéro

- X = FN PE(11) : adresse du sous-programme appelé par USR.
- X = FN PE(26) : valeur du pointeur dans la table de formes. Après un DRAW ou un XDRAW, X = adresse de la prochaine forme de la table.
- X = PEEK(32) : marge gauche de la fenêtre de texte
- X = PEEK(33) : largeur de la fenêtre de texte
- X = PEEK(34) : haut de la fenêtre de texte
- X = PEEK(35) : bas de la fenêtre de texte
- X = PEEK(36) : position horizontale du curseur (0 <= X <= 39).
- Y = PEEK(37) : position verticale du curseur (0<= Y <= 23).

Essayez par exemple l'instruction suivante :
HOME : SPEED=100 : VTAB 10 : FOR X=0 TO 100 : CALL 868 : POKE 37,PEEK(37)-1 : POKE 36, PEEK(36) + 10 - LEN(STR$(X)) : PRINT X : NEXT
ou encore :
IF PEEK(37) = 23 THEN PRINT "Taper une touche pour continuer";: GET R$ .... qui permet l'arrêt momentané de l'affichage lorsque l'écran est rempli.

- X = FN PE(38) : adresse mémoire du début (colonne 0) de la ligne courante en mode GR.
- X = FN PE(40) : adresse mémoire de la colonne 0 de la ligne courante en mode TEXT. L'instruction ci-dessous permet d'afficher les adresses de début des lignes 1 à 22 de la page TEXT :
  HOME : FOR I=1 TO 22 : PRINT I" "FN PE(40) : NEXT I
- X = INT(PEEK(48)/17) : code de la couleur GR (basse résolution) : noir = 0, magenta = 1, bleu foncé = 2, pourpre = 3, vert foncé = 4, gris = 5, bleu = 6, bleu clair = 7, brun = 8, orange = 9, gris = 10, rose = 11, vert = 12, jaune = 13, transparent = 14, blanc = 15.
- X = PEEK(50) : indique le mode d'affichage texte en cours : 255 = NORMAL, 63 = INVERSE, 127 = FLASH.
- X = FN PE(54) : CSW,CSWH : adresse de la routine de sortie de caractères.

Sans DOS : X = 65008 ($FDF0)
DOS standard en MEV : X = 40637
DOS relogé en carte SATURN : X = 48961.

- X = FN PE(56) : KSW,KSWH : adresse de la routine d'entrée de caractères.

Sans DOS : X = 64795
DOS standard en MEV : X = 40577
DOS relogé en carte SATURN : X = 48966.

Il peut être intéressant de déconnecter le DOS pendant une exécution de programme et de le reconnecter plus tard. Ceci peut se faire par la suite d'instructions suivante :
FOR X = 0 TO 3: PD(X) = PEEK(54 + X) : NEXT: POKE 54,240: POKE 55,253: POKE 56,27: POKE 57,253: REM DECONNECTE LE DOS
FOR X = 0 TO 3: POKE 54 + X, PD(X): NEXT: REM RECONNECTE LE DOS

- X = PEEK(78) : nombre aléatoire entre 0 et 255.
- X = PEEK(79) : nombre aléatoire entre 0 et 255.
- X = FN PE(78) : nombre aléatoire entre 0 et 65365.
- X = FN PE(103) : TXTTAB : adresse de début du programme Applesoft (normalement $801).

IF FN PE(103) <> AD OR PEEK(AD - 1) <> 0 THEN POKE 104,AD/256 : POKE 103,AD - 256*PEEK(104) : POKE AD-1,0 : PRINT CHR$(4) "BRUN programme"
permet à un programme Applesoft de se reloger lui-même plus haut en mémoire à partir de l'adresse AD.

- X = FN PE(105) : LOMEM : adresse de début de la zone des variables.
- X = FN PE(107) : ARYTAB : adresse de début de la zone des tableaux.
- X = FN PE(107) - FN PE(105) : longueur de la zone des variables simples.
- X = FN PE(109) : STREND : adresse de fin de la zone des tableaux.
- X = FN PE(109) - FN PE(107) : longueur de la zone des tableaux.
- X = FN PE(111) : FRETOP : adresse de fin de la zone libre. De la fin de la mémoire jusqu'ici sont stockées les chaînes de caractères.
- X = FN PE(111) - FN PE(109) : nombre d'octets de MEV libres.
- X = FN PE(115) : HIMEM : fin de la zone mémoire utilisable. Adresse du début de stockage des chaînes de caractères.
- X = FN PE(117) : CURLIN : contient le numéro de la ligne de programme en cours d'exécution.
- X = FN PE(119) : OLDLIN : contient le numéro de la dernière ligne de programme exécutée.
- X = FN PE(121) : OLDTXTPR : adresse RAM de l'instruction à exécuter - 1.

Pour supprimer la ligne nnnn et les lignes précédentes dans un programme Applesoft, faire :
nnnn X = FN PE(121) + 1 : POKE 103,PEEK(X): POKE 104,PEEK(X+1).
Cela permet par exemple d'augmenter la vitesse d'exécution du programme, en déplaçant le "haut" du programme. Pour revenir aux pointeurs originaux, faire POKE 103,1 : POKE 104,8.

- X = FN PE(123) : numéro de la ligne DATA à lire ou en cours de lecture.
- X = FN PE(125) : adresse mémoire du premier octet de la prochaine donnée à lire en DATA.
- X$ = CHR$ (PEEK(129)) + CHR$ (PEEK(130)) : nom de la dernière variable utilisée. Son adresse se trouve en 131 et 132.
- X = FN PE(131) : VARPNT : adresse de la dernière variable utilisée (ou du descripteur de cette variable). Le nom de la variable se trouve aux adresses 129 et 130.

L'adresse de la variable X pourra être obtenue par X=0 : PRINT FN PE(131).

- X = FN PE(175) : PRGEND : adresse de la fin du programme Applesoft.
- X = FN PE(175) - FN PE(103) : longueur du programme Applesoft.

Pour sauver un programme Applesoft sous forme binaire, faire :
AF = FN PE(175) : XX = AF - 2048 : BSAVE nom du programme, A$800,LXX. Le programme ainsi sauvé ne pourra plus être exécuté par un RUN ni par un BRUN. Pour lancer l'exécution, faire un BLOAD puis un RUN.

- X = FN PE(218) : ERRLIN : contient le numéro de ligne de programme où une erreur s'est produite.
- X = FN PE(220) : valeur du poin-

teur dans l'instruction où s'est produite l'erreur.

- X = PEEK(222) : X sera égal au code d'erreur (Applesoft ou DOS).

X = 0 : NEXT WITHOUT FOR ERROR
X = 1 : LANGUAGE NOT AVAILABLE
X = 2 : RANGE ERROR
X = 3 : -idem-
X = 4 : WRITE PROTECTED
X = 5 : END OF DATA
X = 6 : FILE NOT FOUND
X = 7 : VOLUME MISMATCH
X = 8 : I/O ERROR
X = 9 : DISK FULL
X = 10 : FILE LOCKED
X = 11 : SYNTAX ERROR (DOS)
X = 12 : NO BUFFERS AVAILABLE
X = 13 : FILE TYPE MISMATCH
X = 14 : PROGRAM TOO LARGE
X = 15 : NOT DIRECT COMMAND
X = 16 : SYNTAX ERROR
X = 22 : RETURN WITHOUT GOSUB ERROR
X = 42 : OUT OF DATA ERROR
X = 53 : ILLEGAL QUANTITY ERROR
X = 66 : OVERFLOW ERROR
X = 77 : OUT OF MEMORY ERROR
X = 90 : UNDEF'D STATEMENT
X = 107 : BAD SUBSCRIPT ERROR
X = 120 : REDIM'D ARRAY ERROR
X = 133 : DIVISION BY ZERO
X = 149 : ILLEGAL DIRECT ERROR
X = 163 : TYPE MISMATCH ERROR
X = 176 : STRING TOO LONG ERROR
X = 191 : FORMULA TOO COMPLEX
X = 210 : CAN'T CONTINUE
X = 224 : UNDEF'D FUNCTION
X = 255 : BREAK

Exemple de sous-programme de traitement d'erreurs :

```
10 POKE 758,104 : POKE 759,168 : POKE 760,104 : POKE 761,166 : POKE 762,223 : POKE 763,154 : POKE 764,72 : POKE 765,152 : POKE 766,72 : POKE 767,96 : REM ROUTINE DE TRAITEMENT D'ERREURS (MANUEL APPLESOFT)
20 ONERR GOTO 30000
30000 CALL 758 : ER = PEEK(222) : LN = FN PE(218)
30010 PRINT"ERREUR NUMERO "ER" A LA LIGNE "LN
30030 IF ER<1 or ER>15 THEN PRINT "programme interrompu" : END
30040 .....
30050 IF ER = 4 THEN PRINT "Disquette protégée en écriture - changer de disquette " ;: GET R$ : PRINT CHR$(13) + CHR$(4) "commande DOS" : GOTO 30100
30060 .....
30070 IF ER = 9 THEN PRINT "Disquette remplie - changer de disquette";: GET R$ : ... GOTO 30100
30080 .....
30100 RESUME : REM retour à l'instruction où l'erreur a eu lieu
```

- X = PEEK(223) : valeur du pointeur de pile avant l'erreur.
- X = FN PE(224) : coordonnée horizontale du curseur HGR.
- X = PEEK(226) : position verticale du curseur HGR.
- X = PEEK(228) : code couleur HGR : noir = 0, vert = 42, violet = 85, blanc = 127, noir = 128, orange = 170, bleu = 213, blanc = 255.
- X = PEEK(230) : code de la dernière page HGR appelée. Pour la page 1, X = 32. Pour la page 2, X = 64.

Pour que les instructions HGR agissent sur la page haute résolution non affichée, faire POKE 230,96 - PEEK(230).

- X = PEEK(231) : valeur de SCALE (entre 1 et 255).
- X = FN PE(232) : adresse de début de la table de formes.

Exemple d'utilisation : IF FN PE(232) <> XXXXX THEN PRINT CHR$(4) "BLOAD Table de formes,AXXXXX"

- X = PEEK(234) : compteur de collision HGR.
- X = PEEK(241) : 256 - X = vitesse d'affichage écran. Utilisé par "SPEED=".
- X = PEEK(249) : valeur de ROT (entre 0 et 255).

Page 3

- X = FN PE(977) : si X <> 40383, le DOS n'est pas présent en MEV.
- IOB = PEEK(987) * 256 + 232 : adresse de la table des entrées/sorties utilisée par RWTS.
- X = FN PE(1008) : adresse du sous-programme gérant les interruptions BRK.
- X = FN PE(1010) : adresse de retour au langage en cours d'utilisation.
- X = FN PE(1014) : adresse d'un sous-programme appelé par & (Ampersand).
- X = FN PE(1017) : adresse d'un sous-programme appelé par CTRL-Y en mode moniteur.

Pages texte ($400-$7FF)

- CA = PEEK(1024 + X + 128*Y - 984*INT(Y/8))

CA = code ASCII du caractère se trouvant au point de coordonnées X et Y de l'écran texte (0 <= X <= 39 et 0 <= Y <= 23).

Pour la page 2 de texte (non accessible normalement en Applesoft), remplacer 1024 par 2048.

8 groupes d'adresses sont réservées à chacun des slots :

0 : $478-$47F (1144-1151)
1 : $4F8-$4FF (1272-1279)
2 : $578-$57F (1400-1407)
3 : $5F8-$5FF (1528-1535)
4 : $678-$67F (1656-1663)
5 : $6F8-$6FF (1784-1791)
6 : $778-$77F (1912-1919)
7 : $7F8-$7FF (2040-2047)

Vous pourrez y trouver certaines indications concernant les périphériques, par exemple :

X = PEEK(1144) (soit $478) : numéro de la dernière piste lue ou écrite.

X = INT(PEEK(1528)/16) (soit $5F8) : numéro du slot contenant le contrôleur qui a initialisé le DOS en activité.

Page 8

Un programme Applesoft commence normalement à l'adresse $801 (2049).

- AD = FN PE(2049) : adresse de la seconde ligne du programme Applesoft ;
- NUM = FN PE(2051) : numéro de ligne de la première ligne Applesoft.

DOS 48K ($9600-$BFFF)

Toutes les adresses sont données pour un système 48K. Pour 32K ou 16K, retrancher respectivement 16384 et 32768. Pour un DOS relogé en carte langage ou extension, ces adresses ne sont plus valables.

- X = PEEK(40185) : X=0 ou X > 2 pour le lecteur 1; X=2 pour le lecteur 2.
- X = FN PE(40192) - 595 * N +38 : adresse du Nème tampon du DOS.
- X = PEEK(43603) : adresse de la routine de sortie de caractères (cf CSW,CSWH)
- X = PEEK(43604) : adresse de la routine d'entrée de caractères (cf KSWH,KSWL)
- X = PEEK(43607) : nombre de tampons (buffers) du DOS. Fixé par MAXFILES.
- X = FN PE(43616) : longueur du dernier programme binaire chargé en MEV.
- X = PEEK(43622) : numéro de volume actuel.
- X = PEEK(43624) : numéro de lecteur actuel. Pour permuter entre deux lecteurs, faire un POKE 43624,3-PEEK(43624).
- X = PEEK(43626) : numéro de port actuel.
- X = PEEK(43628) : longueur du dernier enregistrement.
- X = PEEK(43630) : numéro du dernier enregistrement.
- X = PEEK(43632) : numéro de l'octet actuel.

- X = FN PE(43634) : adresse de début du dernier programme binaire chargé en MEV.

BLOCK IOB

IOB (INPUT/OUPUT BLOCK) : utilisé par le sous-programme RWTS (Read Write Track Sector) pour l'accès à un secteur de la disquette. L'adresse de la table IOB utilisée par le DOS est donnée par :

- IOB = 256*PEEK(FNPE(996)) + PEEK(FNPE(999))

Pour un DOS standard 48K, IOB = 47080 ($B7E8).
DOS relogé en carte SATURN : IOB = 49128.

Les informations suivantes peuvent être obtenues par un PEEK à une adresse de l'IOB :

- X = PEEK(IOB) : type de l'IOB, égal à 1 pour DOS sur disquette.
- X = INT(PEEK(IOB + 1)/16) : numéro du slot dans lequel se trouve l'interface DISK à utiliser.
- X = PEEK(IOB + 2) : numéro du lecteur de disque (1 ou 2) à utiliser.
- X = PEEK(IOB + 3) : numéro de volume attendu.
- X = PEEK(IOB + 4) : numéro de la dernière piste lue (0 - 34).
- X = PEEK(IOB + 5) : numéro du dernier secteur lu (0 - 15 en DOS 3.3, 0 - 12 en DOS 3.2).
- X = FN PE(IOB + 6) : adresse de la table des caractéristiques de la mémoire de masse utilisée.
- X = FN PE(IOB + 8) : adresse du dernier tampon du DOS utilisé.
- X = PEEK(IOB + 12)

X = 0 : mode positionnement de la tête.
X = 1 : mode lecture
X = 2 : mode écriture
X = 4 : mode formatage

- X = PEEK(IOB + 13) : contient le code d'erreur lors d'une opération écriture ou lecture :

X = 0 : pas d'erreur
X = 16 : disquette protégée en écriture ou défectueuse.
X = 32 : erreur de volume. Ne correspond pas au volume attendu en IOB + 3.
X = 64 : erreur de lecteur d'origine indéfinie.
X = 128 : erreur de lecture. Emis après 48 essais successifs infructueux.

- X = PEEK(IOB + 14) : numéro de volume de la dernière disquette lue.
- X = INT(PEEK(IOB + 15)/16) : numéro du dernier slot utilisé.
- X = PEEK(IOB + 16) : numéro du dernier lecteur utilisé (0 ou 1).

DOS relogé sur la carte d'extension Saturn 32K

- X = FN PE(48992) : longueur du dernier programme binaire chargé en MEV.
- X = FN PE(49010) : adresse du dernier programme binaire chargé en MEV.

Adresses $C000-CFFF

- X = PEEK(49152) (ou −16384) : saisie au vol d'un caractère au clavier. Si X > 127, une touche a été enfoncée et X − 128 = code ASCII du caractère frappé au clavier.

L'instruction ci-dessous permet par exemple d'interrompre temporairement l'affichage :
IF PEEK(−16384) > 127 THEN POKE −16368,0: WAIT −16384, 128 : POKE −16368,0

- X = PEEK(49184) (soit −16352) active la sortie magnétophone. Permet la sortie sur un amplificateur. Un deuxième PEEK à la même adresse désactive la sortie.
- X = PEEK(49200) +PEEK(49200) (soit −16336) : fait cliqueter le haut-parleur. Peut être remplacé par un POKE 49200, PEEK(49200).

Pour obtenir un "TOP", on pourra utiliser :
FOR I = 49200 TO 49230 : X = PEEK(I) : NEXT

Pour simuler le bruit d'une bille qui rebondit :
FOR N = 0 TO 30 : FOR I = 49200 TO 49230 − N : X = PEEK(I) : NEXT I,N

- X = PEEK(49216) (ou −16320) : impulsion utilitaire. La broche 5 du connecteur de jeu chute de +5 volts à 0 volts pendant 0,98 microsecondes, puis revient à +5 volts.
- X = PEEK(49232) (ou −16304) : sélection du mode graphique.
- X = PEEK(49233) (ou −16303) : sélection du mode TEXT.
- X = PEEK(49234) (ou −16302) : sélection du mode graphique non mixte (écran entier).
- X = PEEK(49235) (ou −16301) : sélection du mode graphique mixte (4 lignes de texte).
- X = PEEK(49236) (ou −16300) : sélection du mode graphique (HGR 1).
- X = PEEK(49237) (ou −16299) : sélection du mode graphique (HGR 2).
- X = PEEK(49238) (ou −16298) : sélection du mode graphique basse résolution.
- X = PEEK(49239) (ou −16297) : sélection du mode graphique haute résolution (HGR ou HGR2).
- X = PEEK(49232) + PEEK (49234) + PEEK(49236) + PEEK (49239) : passage en mode HGR pleine page sans effacement.
- X = PEEK(49232) + PEEK (49234) + PEEK(49237) + PEEK (49239) : passage en mode HGR2 sans effacement.
- X = PEEK(49240) (ou −16296) : sortie tout ou rien numéro 0 OFF.
- X = PEEK(49241) (ou −16295) : sortie tout ou rien numéro 0 ON
- X = PEEK(49242) (ou −16294) : sortie tout ou rien numéro 1 OFF.
- X = PEEK(49243) (ou −16293) : sortie tout ou rien numéro 1 ON.
- X = PEEK(49244) (ou −16292) : sortie tout ou rien numéro 2 OFF.
- X = PEEK(49245) (ou −16291) : sortie tout ou rien numéro 2 ON.
- X = PEEK(49246) (ou −16290) : sortie tout ou rien numéro 3 OFF.
- X = PEEK(49247) (ou −16289) : sortie tout ou rien numéro 3 ON
- X = PEEK(49248) (soit −16288) active l'entrée magnétophone. Un deuxième PEEK à cette adresse déconnecte le magnétophone en lecture.
- X = PEEK(49249) (ou −16287) : teste le bouton de la manette numéro 0. Si X>127, le bouton a été pressé.

Il n'est effectivement prévu que 3 entrées numériques sur lesquelles sont branchés les boutons poussoirs. On peut avoir une 4ème entrée en utilisant l'entrée magnétophone...

- X = PEEK(49250) (ou −16286) : teste le bouton de la manette numéro 1.
- X = PEEK(49251) (ou −16285) : teste le bouton de la manette numéro 2.
- X = PEEK(49252) (ou −16284) : lecture de la manette numéro 0.
- X = PEEK(49253) (ou −16283) : lecture de la manette 1.
- X = PEEK(49254) (ou −16282) : lecture de la manette 2.
- X = PEEK(49255) (ou −16281) : lecture de la manette 3.

Ces entrées analogiques peuvent être utilisées pour des accessoires autres que les manettes.

Les adresses 49256 à 49263 jouent un rôle identique à celui des adresses 49248-49255.

- X = PEEK(49264) (ou −16272) : remise à zéro des entrées analogiques. Les valeurs des adresses 49252 à 49255 passent à une valeur supérieure à 127.

FOR S = 0 TO 7 : PRINT PEEK(49386 + 256*S) + PEEK(49388 + 256*S) + PEEK(49390 + 256*S) : NEXT
fournit un nombre caractéristique de l'interface présente dans le slot S :
0 : carte RAM STATIQUE LEGEND
14 : carte CPS MULTIFUNCTION
113 : carte langage APPLE
350 : programmateur d'Eproms Micro-Périph
376 : interface imprimante (dépend de l'interface)

497 : interface parallèle MID P2
519 : carte SATURN 32K
583 : interface DISK DOS 3.3
632 : interface DISK DOS 3.2

On peut aussi lire à cet égard "Analyse du contenu des slots" dans le Pom's 1.

- X = PEEK(49384) (ou −16152) : arrête le moteur du lecteur.
- X = PEEK(49385) (ou −16151) : démarre le moteur du lecteur.

Gestion d'une carte d'extension MEV

Les adresses ci-dessous sont données pour une carte insérée dans le slot S (0 <= S <= 7).

- X = PEEK(49280 + 16*S) (soit −16256 + 16*S) : connecte la carte d'extension mémoire (BLOC 2) en mode lecture. Protection en écriture.
- X = PEEK(49281 + 16*S) + PEEK(49281 + 16*S) (soit −16255 + 16*S) : déconnecte le BLOC 2 en mode lecture. Dépro tège en écriture. La lecture de la ROM est possible.
- X = PEEK(49282 + 16*S) (soit −16254 + 16*S) : BLOC 2 de la carte déconnecté en lecture, protection en écriture. C'est-à-dire retour au langage en ROM.
- X = PEEK(49283 + 16*S) + PEEK(49283 + 16*S) (soit −16253 + 16*S) : le BLOC 2 de la carte d'extension est accessible en lecture ET en écriture.

Les 12K du BLOC 1 sont accessibles de la même facon par les PEEKs suivants :

- X = PEEK(49288 + 16*S) (soit −16248 + 16*S)
- X = PEEK(49289 + 16*S) (soit −16247 + 16*S)
- X = PEEK(49290 + 16*S) (soit −16246 + 16*S)
- X = PEEK(49291 + 16*S) (soit −16245 + 16*S)

Pour une carte d'extension 32K ou plus, la 2ème tranche de 16K est accessible par les PEEKs indiqués ci-dessous :

- BLOC 2 : utiliser X = PEEK (49284 + 16*S) à (49287 + 16*S).
- BLOC 1 : utiliser X = PEEK (49292 + 16*S) à (49295 + 16*S).

ROM

- X = PEEK(64255) (soit −1281) X = 0 si c'est la ROM AUTOSTART, X = 1 avec l'ancien moniteur.

NDLR : une certaine redondance existe inévitablement entre les PEEKs à gogo et les POKEs à gogo (Pom's 11). Certains sont symétriques, tels par exemple ceux qui concernent la gestion de l'écran (de 32 à 35). D'autres, tels ceux des zones de 49232 à 49249, jouent un rôle identique. Nous les avons gardés dans cet article, car il n'est pas évident a priori (sauf pour les champions) que ces PEEKs et POKEs sont de nature similaire. Remarquons par ailleurs que les PEEKs relatifs au DOS 3.3, comme les POKEs du même genre, n'ont aucune raison de fonctionner pareillement avec le ProDOS. ■

Le lecteur Micro-Expansion 1 Méga

Vincent Plassard

Depuis un an environ, on parlait de ce fameux disque 1 Méga, et j'attendais avec impatience cette nouveauté qui allait désormais agrandir considérablement la mémoire de masse de mon Apple //e. Enfin éliminées, toutes ces manipulations de disquettes 140K, trop petites pour les fichiers un peu importants.

Le mardi 6 mars 1984, j'ai acheté un ensemble lecteur Micro-Expansion G.502 chez un distributeur Apple connu de tous les Lyonnais. Il venait juste de le recevoir et je suis reparti le soir même avec mon appareil. Il m'a semblé intéressant de faire part aux lecteurs de Pom's de mes pre mières impressions d'utilisateur et de passionné d'informatique.

Le matériel

L'ensemble lecteur G.502 est livré dans un carton assez impressionnant, le boîtier étant entouré de plaques de mousse protégeant efficacement le matériel pendant le transport. L'inventaire du colis est le suivant :

- 1 boîtier intégrant 1 ou 2 lecteurs (selon le type choisi). Le boîtier est en bonne tôle solide, à 13 kg l'ensemble (plus de 3 Apple //c ...). La face avant est agréable et les voyants permettent de visualiser le fonctionnement des lecteurs.
- 1 alimentation (de bonne qualité).
- 1 ventilateur.
- 1 carte Interface/Contrôleur qui assure la liaison entre l'Apple et le boîtier. Cette carte est impressionnante, mais s'intègre parfaitement. Pour les connaisseurs, elle est vraiment "chouette".
- 1 disquette de démarrage (140 K). Cette disquette permet d'initialiser et de formater les disquettes 1 Méga sous le système d'exploita tion que vous utiliserez. Il existe actuellement des disquettes de démarrage pour DOS 3.3, MEM/DOS, Pascal et CP/M. Ces disquettes proposent différents utilitaires propres à l'utilisation des lecteurs 1 Méga.
- 1 manuel d'utilisation du système. A la livraison, on m'a remis un manuel qui me semblait un peu confus; j'en ai fait la remarque à mon revendeur. Je viens de rece voir une nouvelle documentation beaucoup plus explicite. Il est important de lire ce fascicule pour éviter des erreurs de manipulation qui pourraient être préjudiciables.
- 1 cordon secteur (alimentation 220V). La prise de terre est obligatoire (ne pas l'oublier).
- 1 nappe reliant l'interface au boîtier. Un détrompeur propre à chaque connecteur évite tout risque d'erreur dans les branchements.

Caractéristiques techniques

Caractéristiques générales

- Nbre de lecteurs par unité : 1 pour le modèle G.501, 2 pour le modèle G.502
- Nbre de têtes de lecture : 2 par lecteur
- Nbre de pistes par face : 77
- Nbre de pistes total : 154
- Nbre de secteurs par piste : 26
- Nbre d'octets par secteur : 256
- Vitesse de rotation : 360 t/m
- Temps de top moteur (maxi) : 1 sec.

Caractéristiques d'Enregistrement

- Capacité non formatée : 1.604 K.octets
- Capacité formatée : 1.025 K.octets
- Temps d'accès piste à piste : 5 ms
- Temps d'accès moyen : 91 ms
- Temps d'accès maximum : 395 ms
- Temps de stabilisation : 15 ms
- Temps moyen d'attente : 17 ms
- Vitesse de transfert : 500 K.bits/sec.

Dimensions du boîtier : hauteur 15 cm, largeur 26 cm, profondeur 36 cm.

Supports magnétiques : type BASF réf. 2 HD 71/053 ou Micro-Expansion réf. DQ 500 H.

MTBF (mean time between failures) : 8.000 heures.

Mes impressions d'utilisateur

Maintenant que vous connaissez un peu mieux l'ensemble G.501/G.502, c'est-à-dire le matériel et ses caractérisques, je vous livre mes impressions d'utilisateur. C'était d'ailleurs l'objectif que je m'étais fixé en me mettant à "pianoter" cet article.

J'ai travaillé avec les 1Mo sous DOS 3.3 et sous MEM/DOS. La première remarque que je peux faire, c'est que les 1Mo sont beaucoup plus adéquats sous MEM/DOS qu'en DOS 3.3. Le ProDOS aura certainement une meilleure efficacité avec ces lecteurs que le DOS 3.3.

Système DOS 3.3

Il faut savoir tout d'abord que, sur une disquette 1Mo, on met 7 disquettes 140K. L'équivalent d'une disquette 140K s'appellera VOLUME avec le système 1Mo. C'est déjà le cas avec les disquettes 140K, mais on ne s'y sert en général pas des numéros de volume. ATTENTION, le numero de volume qui était utilisé avec les disquettes 140K n'est pas reconnu par le système 1Mo.

Une disquette 1Mo contient 7 catalogues (D1....D7). Pour lire un programme ou un fichier qui se trouve sur le troisième volume de la disquette 1Mo, vous devrez taper "LOAD nomfic,Ss,D3". En résumé, on peut considérer qu'avec un ensemble lecteur 1Mo, nous avons :

– 7 drives 140K (D1...D7) avec le modèle G.501;
– 14 drives 140K (D1..D14) avec le modèle G.502.

Le principal avantage du système est d'éviter la manipulation de disquettes et d'avoir en permanence l'accès à 7 ou 14 drives selon le modèle.

Après avoir booté la disquette de démarrage 140K, à la mise sous tension de l'Apple, le système est utilisable et vous voyez apparaître à l'écran le menu des utilitaires, à savoir :

Formatage physique disquette 1Mo
Initialisation des Volumes
Tranfert des Programmes (le FID)
Copie d'un Volume (c'est COPYA)
Catalogue des Volumes 1Mo
Copie rapide du 1Mo sur un 1Mo
Retour au Basic

La copie rapide du 1Mo sur 1Mo est vraiment très rapide : 1'20" pour recopier l'équivalent de 7 disquettes !

En cours d'utilisation, les temps de transfert entre la disquette 1Mo et la mémoire de l'Apple sont semblables à ceux du DOS 3.3.

Dès la mise sous tension de l'ensemble lecteur 1Mo, le ventilateur de refroidissement se met à tourner. Je ne peux pas dire qu'il soit vraiment bruyant mais ce "ronron" permanent est pénible à supporter...

Dans l'immédiat, les logiciels tels que Visicalc, Multiplan ou Apple Writer ne sont pas utilisables sur disquettes 1Mo, et je le regrette beaucoup. Sinon, aucun problème d'utilisation n'est à noter sous DOS 3.3.

Système MEM/DOS

Je ne sais pas si beaucoup d'entre vous connaissent et utilisent le MEM/DOS, mais je l'emploie pour ma part très souvent, car il simplifie considérablement la programmation par l'usage de masques et grâce à sa manipulation des fichiers. Depuis que j'ai découvert MEM/DOS, je ne comprends pas comment j'ai pu m'en passer antérieurement. Par contre, pour des fichiers importants, une disquette 140K est rapidement saturée; faire une recherche ou un tri sur plusieurs disquettes devient une véritable acrobatie, voire une opération pratiquement irréalisable.

Actuellement, avec les 1Mo, le problème est résolu. Vous disposez réellement d'un méga en ligne par disquette, contrairement à la disquette 1Mo sous DOS 3.3 qui compte 1Mo fractionné en 7 volumes. Vous saisissez les perspectives qui s'ouvrent aujourd'hui aux utilisateurs. Jusqu'alors, pour résoudre un tel problème, un disque dur était indispensable, sans parler du backup nécessaire pour la sauvegarde.

Les utilitaires sous MEM/DOS sont complets et sans problème apparent. J'ai observé une grande vitesse d'exécution, phénomène qu'on ne constate pas sous DOS 3.3.

En résumé, avec un Apple //e, une carte 80 colonnes étendue, une carte MEM/DOS, et cet ensemble lecteur, vous pouvez réaliser efficacement des applications professionnelles qui ne relèvent plus d'un bricolage informatique. Cette configuration est vraiment performante pour une entreprise, pour un établissement scolaire, pour qui doit traiter la paye, la comptabilité, les plannings, la gestion, les fichiers clients, les fichiers élèves, tous les fichiers...

Conclusion

Ce n'est pas après deux semaines d'utilisation que je peux prétendre connaître à fond ce nouveau matériel mais, d'emblée, je peux dire en toute simplicité que cet ensemble lecteur 1Mo est bien conçu et qu'il correspond à un besoin réel. L'utilisateur moyen a désormais la possibilité d'accéder à une mémoire de masse très acceptable et à une possibilité de sauvegarde adéquate.

Le marché du disque dur 5Mo risque d'être touché par cette nouveauté.

Le prix

Hélas, le prix n'est pas à la portée de toutes les bourses. 13.000 F TTC pour le modèle G.501, 25.000 F pour le G.502. La disquette 1Mo revient à 130 F. Pour des applications concrètes et utilitaires, ce prix sera facilement considéré comme un investissement rentable et indispensable. Au niveau de l'amateur, il est certainement prohibitif.

Formulons l'espoir de voir un jour les prix baisser, mais nous payons le prix du sérieux et de la technicité (*NDLR : ... et d'un marché pour le moment hexagonal.*).

Bravo pour cette réalisation française ! Les 2Mo m'ont mordu...

Pour la petite histoire, chacun sait que le week-end est propice pour se consacrer à fond à de tels essais. Ma femme avait eu la délicatesse et la gentillesse de me laisser seul dans mes expériences. Quelle ne fut pas sa surprise, en rentrant le soir, de me retrouver avec un énorme pansement autour de la main gauche.
Non, rassurez-vous, les 2Mos ne m'avaient pas "véritablement" mordu, c'était simplement une grossière erreur de cuisinier étourdi ... Pour ma punition, la brûlure réduit dans l'immédiat ma vitesse de frappe, mais heureusement, je récupère facilement avec mes 2Mo le temps que j'aurais perdu en manipulation de disquettes ...

En conclusion, les 2Mos, c'est TERRIBLE.

***NDLR** : nous devons apporter un bémol à cette étude enthousiaste, où l'on reconnaît l'esprit dynamique de l'animateur du club Info-Maniaques (62 avenue Paul Santy - 69008 Lyon). Nous n'avons pas encore eu l'occasion à Pom's de tester ce lecteur et réservons donc notre opinion en attendant. Dès que Micro-Expansion nous aura confié un matériel, nous vous apporterons notre opinion en complément.*
Rappelons également, pour une meilleure analyse du marché et une meilleure information des utilisateurs intéressés par ce type de mémoire de masse, que la société I.E.F. (Paris) commercialise depuis quelques années des lecteurs de disquettes souples 1Mo, d'un prix relativement moins élevé mais aussi d'une technologie moins "lourde" (pas d'alimentation autonome, pas de ventilation...) ; par contre, la fiabilité de ce lecteur semble inférieure.
Vous pourrez consulter une présentation de ces autres lecteurs dans l'article consacré aux Mémoires de Masse dans le Pom's 4 ou le Recueil No 1. ■

Réduction d'image HGR

Patrice Neveu

Pom's vous a souvent proposé des programmes pour gérer les pages graphiques : inversion des couleurs, transferts, opérations logiques entre plusieurs pages, stroboscopie, etc. Aujourd'hui, je vous offre un ensemble de routines qui permettent de réduire une page graphique au quart de sa taille normale.

Le programme Basic de démonstration **Réduction au 1/4** prend une image stockée sur disquette et la réduit au quart de sa taille en haut à gauche de l'écran. Il utilise les routines en assembleur suivantes :

- **Réduit Hauteur** qui réduit de moitié la hauteur de l'image;
- **Réduit Largeur** fait de même, avec la largeur bien sûr;
- **Moitié 1 → Moitié 2** transfère la demi partie gauche de l'image à droite;
- **Copyleft** agit sur les demi-parties hautes et basses;
- **HGR2 → HGR** effectue un simple transfert entre les 2 pages;
- **HGR2+HGR2 → HGR** superpose les 2 pages sur la première.

Hormis les 2 dernières routines qui sont très simples, quelques mots d'explication pour les autres :

– Copyleft est une routine de Jean-François Duvivier parue dans le numéro 3 de Pom's; elle est utilisée ici pour inverser les parties hautes et basses d'un écran;

– Réduit Hauteur balaye une ligne sur deux de l'écran 1, par appel à HPOSN (module de l'interpréteur Applesoft en $F411 renvoyant l'adresse de base d'une ligne graphique) et stocke cette ligne sur l'écran 2.

– Réduit Largeur est le module le plus compliqué de l'ensemble car il s'agit de conserver un pixel sur deux, en travaillant sur des octets où seuls 7 bits nous intéressent et sont affichés en ordre inverse :
Une ligne graphique comporte 280 points (pixels) soit 20x2x7. Examinons un groupe de 2x7 pixels et codons-les individuellement :
Ecran : ABCDEFGHIJKLMN
Les octets équivalents sont au nombre de 2 et leurs bits sont :
Octets : *GFEDCBA#NMLKJIH
avec * et # les bits de couleur
Regroupons arbitrairement ces deux octets en un seul avec les bits :
Octet : *NLJHFDB
Cet octet apparaît graphiquement comme 7 pixels :
Ecran : BDFHJLN
et l'objectif de ne conserver qu'un seul pixel sur 2 est donc réalisé.

Le programme **Réduction infinie** permet de dupliquer une image en quatre plus petites, ou de juxtaposer 4 images différentes sur un même écran graphique.

Bien entendu, la notion de couleur n'est pas conservée ! Pour éviter des pertes trop sensibles de définition, il est conseillé d'avoir une image originale en HCOLOR=3 (blanc). D'autre part, rien n'empêche de recommencer la réduction au quart une nouvelle fois; on peut ainsi obtenir des réductions au 1/16. Difficile d'aller plus loin ! ■

Programme REDUCTION au 1/4

```
10  HOME
100  INPUT "NOM DE L'IMAGE A REDUIRE : "
      ;I1$
140  INPUT "NOM DE L'IMAGE FINALE : ";NF
      $
990  HGR2
1000  HGR
1005  REM  *** GRAPHIQUE TOTAL ***
1010  POKE  - 16302,0
1015  REM
1020  PRINT  CHR$ (4)"BLOAD"I1$",A$2000"
1040  PRINT  CHR$ (4)"BRUN REDUIT LARGEU
      R.OBJ"
1050  POKE  - 16300,0
1060  PRINT  CHR$ (4)"BRUN HGR2->HGR.OBJ
      "
1065  REM  *** NETTOIE PAGE 2 ***
1070  HGR2
1075  REM  *** BRANCHE PAGE 1 ***
1080  POKE 230,32
1085  REM
1090  PRINT  CHR$ (4)"BRUN REDUIT HAUTEU
      R.OBJ"
1200  IF NF$ <  > "" THEN  PRINT  CHR$ (
      4)"BSAVE"NF$",A$4000,L$1FF8"
1210  HOME : GET A$: TEXT
```

Programme REDUCTION INFINIE

```
1  REM  *** REDUCTION INFINIE ***
10  TEXT : HOME : SPEED= 255: NOTRACE
11  ONERR  GOTO 990
12  LOMEM: 32768
20 D$ =  CHR$ (4)
30 DB$ = D$ + "BRUN"
35 DL$ = D$ + "BLOAD"
36 DS$ = D$ + "BSAVE"
40  INVERSE : PRINT " DEMONSTRATION DE R
    EDUCTION D'IMAGE HGR ": NORMAL
50  PRINT : PRINT "ALEXANDRE AVRANE 05/M
    AI/1984 POUR POM'S"
55  PRINT : PRINT : PRINT "REDUCTION D'U
    NE SEULE IMAGE OU DE QUATREIMAGES
     DIFFERENTES (1/4)? ";: GET A$
56  IF A$ = "4" THEN 500
57  IF A$ <  > "1" THEN  RUN
99  REM  *** UNE SEULE IMAGE ***
100  PRINT : PRINT : INPUT "NOM DE L'IMA
    GE: ";I1$
110  HGR2 : HGR : POKE 49234,0
120  PRINT DL$I1$",A$2000"
125  GOSUB 1000
130  GOSUB 2000
140  PRINT DB$"HGR+HGR2->HGR2.OBJ"
150  POKE 49237,0
160 A$ = "2000<4000.5FFF": GOSUB 4000
170  GOSUB 3000
180  PRINT DB$"HGR+HGR2->HGR2.OBJ"
190  POKE 49237,0
200  HOME : GET A$
210  TEXT : HOME : PRINT "ON REDUIT ENCO
    RE? (O/N) ";: GET A$
220  IF A$ = "O" THEN  PRINT : POKE 4923
    9,0:A$ = "2000<4000.5FFF": GOSUB
    4000: HGR2 : GOTO 125
225  GOTO 900
499  REM  *** 4 IMAGES DIFFERENTES ***
500  PRINT : PRINT : INPUT "NOM DE L'IMA
    GE 1: ";I1$
505  HGR2 : HGR : POKE 49234,0
510  PRINT DL$I1$",A$2000"
520  GOSUB 1000
```

```
525 A$ = "6000<4000.5FFF": GOSUB 4000
527  POKE 49236,0: POKE 49233,0
530  PRINT : INPUT "NOM DE L'IMAGE 2: ";
       I2$
535  POKE 49232,0
540  PRINT DL$I2$",A$2000"
550  GOSUB 1000
580  GOSUB 2000
585 A$ = "4000<6000.7FFF": GOSUB 4000
590  PRINT DB$"HGR+HGR2->HGR2.OBJ"
600  POKE 49237,0
610 A$ = "6000<4000.5FFF": GOSUB 4000
620  POKE 49236,0: POKE 49233,0
630  PRINT : INPUT "NOM DE L'IMAGE 3: ";
       I3$
640  POKE 49232,0
650  PRINT DL$I3$",A$2000"
655  HGR2
660  GOSUB 1000
670  GOSUB 3000
675 A$ = "4000<6000.7FFF": GOSUB 4000
680  PRINT DB$"HGR+HGR2->HGR2.OBJ"
690  POKE 49237,0
700 A$ = "6000<4000.5FFF": GOSUB 4000
710  POKE 49236,0: POKE 49233,0
720  PRINT : INPUT "NOM DE L'IMAGE 4: ";
       I4$
730  POKE 49232,0
740  PRINT DL$I4$",A$2000"
750  HGR2
760  GOSUB 1000
770  GOSUB 2000
790  GOSUB 3000
800 A$ = "4000<6000.7FFF": GOSUB 4000
810  PRINT DB$"HGR+HGR2->HGR2.OBJ"
820  POKE 49237,0: GET A$
830  POKE 49236,0: POKE 49233,0
899  REM  *** SAUVEGARDE DE L'IMAGE ***
900  PRINT : PRINT : PRINT "SAUVEGARDE D
       E L'IMAGE REDUITE": INPUT "NOM (R
       ETURN SINON): ";I2$
910  IF I2$ < > "" THEN  PRINT DS$I2$",
       A$4000,L$1FF8"
920  PRINT : PRINT "ON RECOMMENCE? (O/N)
       ";: GET A$: IF A$ = "O" THEN  RU
       N
930  END
990  POKE 216,0: TEXT : HOME : RESUME
999  REM  * REDUCTION --> HAUT GAUCHE *
1000  POKE 230,32
1010  PRINT DB$"REDUIT LARGEUR.OBJ"
1020  POKE 49237,0
1030 A$ = "2000<4000.5FFF": GOSUB 4000
1040  HGR2
1050  POKE 230,32: REM PAGE=1
1060  PRINT  CHR$ (4)"BRUN REDUIT HAUTEU
       R.OBJ"
1070 A$ = "2000<4000.5FFF": GOSUB 4000
1080  RETURN
1999  REM  * TRANSFERT GAUCHE-DROITE *
2000  POKE 230,32
2010  POKE 49236,0
2020  PRINT DB$"MOITIE1->MOITIE2.OBJ"
2030  RETURN
2999  REM  ** TRANSFERT HAUT --> BAS **
3000  PRINT DL$"COPYLEFT"
3010  POKE 771,32: POKE 772,96: POKE 773
       ,1: POKE 774,0
3020  POKE 49236,0
3030  CALL 768
3040  RETURN
3999  REM  *** DEPLACE LES PAGES ***
4000 A$ = A$ + "M ND9C6G": FOR I = 1 TO
       LEN (A$): POKE 511 + I, ASC ( MI
      D$ (A$,1,1)) + 128: NEXT : POKE 7
      2,0: CALL  - 144: RETURN
```

REDUIT LARGEUR
Lisa 2.5

```
 1 ;************************
 2 ;*                      *
 3 ;* CE PROGRAMME REDUIT  *
 4 ;*  DE MOITIE DANS LA   *
 5 ;*  LARGEUR,LA PAGE 1,  *
 6 ;* ET ENVOI LE RESULTAT *
 7 ;*    SUR LA PAGE 2.    *
 8 ;*                      *
 9 ;* AUTEUR :   P . NEVEU *
10 ;*                      *
11 ;************************
12 ;
13 ;
14 ;
15 HPOSN    EQU $F411
16 HBASL    EPZ $26
17 HBASH    EPZ $27
18 P1L      EPZ $6              ;ADRESSE DE LA
19 P1H      EPZ $7              ;LIGNE,PAGE 1
20 P2L      EPZ $8              ;ADRESSE DE LA
21 P2H      EPZ $9              ;LIGNE,PAGE 2
22 ;
23          ORG $300
24 ;
25          JMP DEBUT
26 ;
27 LIGNE    DFS 1               ;NUMERO DE LIGNE
28 OCT      DFS 1               ;OCTET RESULTAT
29 OCT1     DFS 1               ;OCTETS A
30 OCT2     DFS 1               ;RACCOURCIR.
31 DEP      DFS 1
32 ARR      DFS 1
33 ;
34 DEBUT    LDA #$00
35          STA OCT
36          STA LIGNE
37 ;
38          LDA #$00
39          STA $C052           ;GRAPHIQUE TOTAL
40          STA $C055           ;PAGE 2
41 ;
42 CHERCHE  LDA LIGNE
43          LDX #$00
44          LDY #$00
45          JSR HPOSN
46 ;
47          LDY #$00
48          STY DEP
49          STY ARR
50 ;
51          LDA HBASL
52          STA P1L
53          LDA HBASH
54          STA P1H
55          CLC
56          ADC #$20
57          STA P2H
58          LDA P1L
59          STA P2L
60 ;
61 CHARGE   LDY DEP
62          LDA (P1L),Y
63          STA OCT1
64          INC DEP
65          LDY DEP
66          LDA (P1L),Y
```

```
 67            STA OCT2
 68            INC DEP
 69 ;
 70 ;2 OCTETS->1 OCTET
 71 ;
 72            LDA #$00
 73            STA OCT
 74 ;
 75            LDA OCT1
 76            AND #$80          ;BIT8
 77            CLC               ;-->8
 78            ADC OCT
 79            STA OCT
 80 ;
 81            LDA OCT1
 82            AND #$20          ;BIT6
 83            LSR               ;-->3
 84            LSR
 85            LSR
 86            CLC
 87            ADC OCT
 88            STA OCT
 89 ;
 90            LDA OCT1
 91            AND #$08          ;BIT4
 92            LSR               ;-->2
 93            LSR
 94            CLC
 95            ADC OCT
 96            STA OCT
 97 ;
 98            LDA OCT1
 99            AND #$02          ;BIT2
100            LSR               ;-->1
101            CLC
102            ADC OCT
103            STA OCT
104 ;
105            LDA OCT2
106            AND #$40          ;BIT7
107            CLC               ;-->7
108            ADC OCT
109            STA OCT
```

```
110 ;
111            LDA OCT2
112            AND #$10          ;BIT5
113            ASL               ;-->6
114            CLC
115            ADC OCT
116            STA OCT
117 ;
118            LDA OCT2
119            AND #$04          ;BIT3
120            ASL               ;-->5
121            ASL
122            CLC
123            ADC OCT
124            STA OCT
125 ;
126            LDA OCT2
127            AND #$01          ;BIT1
128            ASL               ;-->4
129            ASL
130            ASL
131            ADC OCT
132            STA OCT
133 ;
134 ;SAUVE L'OCTET
135 ;
136            LDY ARR
137            STA (P2L),Y
138            INC ARR
139            LDA ARR
140            CMP #!19
141            BEQ CONT
142            JMP CHARGE
143 ;
144 ;LIGNE SUIVANTE
145 ;
146 CONT       INC LIGNE
147            LDA LIGNE
148            CMP #!192
149            BEQ FIN
150            JMP CHERCHE
151 FIN        RTS
152            END
```

Récapitulation

REDUIT LARGEUR

```
0300- 4C 09 03 C0 60 00 A9 00
0308- 8D A9 00 8D 04 03 8D 03
0310- 03 A9 00 8D 52 C0 8D 55
0318- C0 AD 03 03 A2 00 A0 00
0320- 20 11 F4 A0 00 8C 07 03
0328- 8C 08 03 A5 26 85 06 A5
0330- 27 85 07 18 69 20 85 09
0338- A5 06 85 08 AC 07 03 B1
0340- 06 8D 05 03 EE 07 03 AC
0348- 07 03 B1 06 8D 06 03 EE
0350- 07 03 A9 00 8D 04 03 AD
0358- 05 03 29 80 18 6D 04 03
0360- 8D 04 03 AD 05 03 29 20
0368- 4A 4A 4A 18 6D 04 03 8D
0370- 04 03 AD 05 03 29 08 4A
0378- 4A 18 6D 04 03 8D 04 03
0380- AD 05 03 29 02 4A 18 6D
0388- 04 03 8D 04 03 AD 06 03
0390- 29 40 18 6D 04 03 8D 04
0398- 03 AD 06 03 29 10 0A 18
03A0- 6D 04 03 8D 04 03 AD 06
03A8- 03 29 04 0A 0A 18 6D 04
03B0- 03 8D 04 03 AD 06 03 29
03B8- 01 0A 0A 0A 6D 04 03 8D
03C0- 04 03 AC 08 03 91 08 EE
03C8- 08 03 AD 08 03 C9 13 F0
03D0- 03 4C 3C 03 EE 03 03 AD
03D8- 03 03 C9 C0 F0 03 4C 19
03E0- 03 60
```

Programme REDUIT HAUTEUR Lisa 2.5

```
 1 ;*************************
 2 ;*                       *
 3 ;* CE PROGRAMME PREND UNE *
 4 ;*  LIGNE SUR DEUX DE LA  *
 5 ;* PAGE 1 ET SAUVE SUR LA *
 6 ;*        PAGE 2.        *
 7 ;*                       *
 8 ;*************************
 9 ;
10 ;
11 HPOSN     EQU $F411
12 HBASL     EPZ $26
13 HBASH     EPZ $27
14 P1L       EPZ $6
15 P1H       EPZ $7
16 P2L       EPZ $8
17 P2H       EPZ $9
18 LIGNE1    EPZ $A
19 LIGNE2    EPZ $B
20 ;
21           ORG $300
22 ;
23 ;
24 DEBUT     LDA #$00
25           STA LIGNE1
26           STA LIGNE2
27 ;
28 CHERCHE   LDA LIGNE1
29           LDX #$00
30           LDY #$00
31           JSR HPOSN
32           LDA HBASL
33           STA P1L
```

```
34           LDA HBASH
35           STA P1H
36 ;
37           LDA LIGNE2
38           LDX #$00
39           LDY #$00
40           JSR HPOSN
41           LDA HBASL
42           STA P2L
43           LDA HBASH
44           CLC
45           ADC #$20
46           STA P2H
47 ;
48 ;
49           LDY #$00
50 BOUCLE    LDA (P1L),Y
51           STA (P2L),Y
52           INY
53           CPY #!39
54           BNE BOUCLE
55           INC LIGNE1
56           INC LIGNE1
57           INC LIGNE2
58           LDA LIGNE1
59           CMP #!192
60           BNE CHERCHE
61 ;
62 ; PASSE EN PAGE 2
63 ; POUR VOIR LE
64 ; RESULTAT
65 ;
66           LDA #$00
67           STA $C052
68           STA $C055
69 ;
70           RTS
71           END
```

Récapitulation
REDUIT HAUTEUR

```
0300- A9 00 85 0A 85 0B A5 0A
0308- A2 00 A0 00 20 11 F4 A5
0310- 26 85 06 A5 27 85 07 A5
0318- 0B A2 00 A0 00 20 11 F4
0320- A5 26 85 08 A5 27 18 69
0328- 20 85 09 A0 00 B1 06 91
0330- 08 C8 C0 27 D0 F7 E6 0A
0338- E6 0A E6 0B A5 0A C9 C0
0340- D0 C4 A9 00 8D 52 C0 8D
0348- 55 C0 60
```

Programme
MOITIE 1 → MOITIE 2
Lisa 2.5

```
 1 ;
 2 ;**************************
 3 ;*                        *
 4 ;* CE PROGRAMME TRANSFERE *
 5 ;* LA MOITIE DE LA PAGE 2 *
 6 ;*    SUR L'AUTRE MOITIE  *
 7 ;*                        *
 8 ;**************************
 9 ;
10 ;
11 HPOSN     EQU $F411
12 HBASL     EPZ $26
13 HBASN     EPZ $27
14 ;
15           ORG $300
16 ;
17           JMP DEBUT
18 ;
19 LIGNE     DFS 1
20 DEP       DFS 1
21 ARR       DFS 1
22 ;
23 DEBUT     LDA #$00
24           STA LIGNE
25 ;
26 CHERCHE   LDA LIGNE
27           LDX #$00
28           LDY #$00
29           JSR HPOSN
30 ;
31           LDA #$00
32           STA DEP
33           CLC
34           ADC #!20
35           STA ARR
36 ;
37 BOUCLE1   LDY DEP
38           LDA (HBASL),Y
39           LDY ARR
40           STA (HBASL),Y
41           LDA #$00
42           LDY DEP
43           STA (HBASL),Y
44           INC DEP
45           INC ARR
46           LDA DEP
47           CMP #!20
48           BNE BOUCLE1
49 ;
50           INC LIGNE
51           LDA LIGNE
52           CMP #!192
53           BNE CHERCHE
54           RTS
55           END
```

Récapitulation
MOITIE 1 → MOITIE 2

```
0300- 4C 06 03 C0 14 28 A9 00
0308- 8D 03 03 AD 03 03 A2 00
0310- A0 00 20 11 F4 A9 00 8D
0318- 04 03 18 69 14 8D 05 03
0320- AC 04 03 B1 26 AC 05 03
0328- 91 26 A9 00 AC 04 03 91
0330- 26 EE 04 03 EE 05 03 AD
0338- 04 03 C9 14 D0 E2 EE 03
0340- 03 AD 03 03 C9 C0 D0 C3
0348- 60
```

Programme COPYLEFT
Big Mac

```
 1 ********************************
 2 *          COPYLEFT            *
 3 ********************************
 4
 5 * Ref: Pom's #3 p.53
 6
 7 NOLIGN   =    $00
 8 NB       =    $01
 9 LIGNP    =    $02
10 SAVX     =    $04
11 SENS     =    $05
12 LIGN     =    $26
13 HLIGN    =    $E2
14 HPAG     =    $E6
15 HPOSN    =    $F411
16 HIRES1   =    $C050
17 HIRES2   =    $C052
18 HIRES3   =    $C057
19 PAGE1    =    $C054
20 PAGE2    =    $C055
21
22          ORG  $300
23
24          JMP  DEBUT
25
26 PAGE     HEX  20
27 SEQU     DFB  0,0,0,0,0,0,0,0,0,0
28          DFB  0,0,0,0,0,0,0,0,0,0
29
30 DEBUT    =    *
31          LDA  HIRES1
32          LDA  HIRES2
33          LDA  HIRES3
34
35          LDX  #0
36          STX  SAVX
37
38          LDA  PAGE1
39          LDA  PAGE
40          CMP  #$40
41          BNE  BOUCLE
42          LDA  PAGE2
43 BOUCLE   LDA  #0
44          STA  NOLIGN
45          STA  HLIGN
46          JSR  HPOSN
47
48          LDX  SAVX
49          INC  SAVX
50          INC  SAVX
51          LDA  SEQU,X
52          BEQ  FIN
53          STA  NB
54          INX
55          LDA  SEQU,X
56          STA  SENS
57
58 BOUCL2   LDA  SENS
59          BPL  BOUCL3
60          LDA  #192
61          STA  NOLIGN
62
63 BOUCL3   JSR  TOUR
64          DEC  NB
65          LDA  NB
66          BNE  BOUCL2
67
68          JMP  BOUCLE
69
70 TOUR     JSR  CLIGN
71          LDY  #$27
72 TOUR1    LDA  (LIGN),Y
73          STA  (LIGNP),Y
74          DEY
75          BPL  TOUR1
76
77          LDA  NOLIGN
78          BNE  TOUR
79          RTS
80
81 CLIGN    LDA  LIGN
82          STA  LIGNP
83          LDA  LIGN+1
84          STA  LIGNP+1
85
86          LDA  NOLIGN
87          CLC
88          ADC  SENS
89          STA  NOLIGN
90          CMP  #192
91          BNE  LIGN1
92          LDA  #0
93          STA  NOLIGN
94 LIGN1    LDA  PAGE
95          STA  HPAG
96          LDA  NOLIGN
97          STA  HLIGN
98          JSR  HPOSN
99          RTS
100
101 FIN     RTS
102         END
```

Récapitulation COPYLEFT
Pom's 3 p. 53 ou
Recueil 1 p. 140

```
0300- 4C 18 03 20 00 00 00 00
0308- 00 00 00 00 00 00 00 00
0310- 00 00 00 00 00 00 00 00
0318- AD 50 C0 AD 52 C0 AD 57
0320- C0 A2 00 86 04 AD 54 C0
0328- AD 03 03 C9 40 D0 03 AD
0330- 55 C0 A9 00 85 00 85 E2
0338- 20 11 F4 A6 04 E6 04 E6
0340- 04 BD 04 03 F0 51 85 01
0348- E8 BD 04 03 85 05 A5 05
0350- 10 04 A9 C0 85 00 20 62
0358- 03 C6 01 A5 01 D0 EF 4C
0360- 32 03 20 73 03 A0 27 B1
0368- 26 91 02 88 10 F9 A5 00
0370- D0 F0 60 A5 26 85 02 A5
0378- 27 85 03 A5 00 18 65 05
0380- 85 00 C9 C0 D0 04 A9 00
0388- 85 00 AD 03 03 85 E6 A5
0390- 00 85 E2 20 11 F4 60 60
```

Programme
HGR 2 → HGR
Lisa 2.5

```
 1 ;**************************
 2 ;*                        *
 3 ;* CE PROGRAMME TRANSFERE *
 4 ;* LA PAGE GRAPHIQUE 1->2 *
 5 ;*                        *
 6 ;**************************
 7 ;
 8 ;
 9           ORG $300
10 ;
11 DEP       EPZ $6
12 ARR       EPZ $8
13 ;
14 DEBUT     LDA #$00
15           STA DEP
16           STA ARR
17           LDA #$40
18           STA DEP+1
19           LDA #$20
20           STA ARR+1
21           LDX #!32
22           LDY #$FF
23 SUIVANT   LDA (DEP),Y
24           STA (ARR),Y
25           DEY
26           BNE SUIVANT
27 BLOCSUI   INC DEP+1
28           INC ARR+1
29           DEX
30           BMI FIN
31           BNE SUIVANT
32 FIN       RTS
33           END
```

Récapitulation
HGR 2 → HGR

```
0300- A9 00 85 06 85 08 A9 40
0308- 85 07 A9 20 85 09 A2 20
0310- A0 FF B1 06 91 08 88 D0
0318- F9 E6 07 E6 09 CA 30 02
0320- D0 F0 60
```

Programme
HGR + HGR 2 → HGR 2
Lisa 2.5

```
 1 ;
2 ;***************************
3 ;*                         *
4 ;* CE PROGRAMME SUPERPOSE  *
5 ;* LA PAGE 1 SUR LA PAGE 2 *
6 ;*                         *
7 ;***************************
8 ;
9           ORG $300
10 ;
11 P1       EPZ $6
12 P2       EPZ $8
13 ;
14          LDA #$00
15          STA P1
16          STA P2
17          TAY
18          TAX
19          LDA #$40
20          STA P2+1
21          LDA #$20
22          STA P1+1
23 ;
24 OCT      LDA (P1),Y
25          ORA (P2),Y
26          STA (P2),Y
27          INY
28          BNE OCT
29          INC P1+1
30          INC P2+1
31          LDY #$00
32          INX
33          CPX #!33
34          BNE OCT
35          RTS
36          END
```

Récapitulation
HGR + HGR 2 → HGR 2

```
0300- A9 00 85 06 85 08 A8 AA
0308- A9 40 85 09 A9 20 85 07
0310- B1 06 11 08 91 08 C8 D0
0318- F7 E6 07 E6 09 A0 00 E8
0320- E0 21 D0 EC 60
```

Initiation à l'assembleur (3)

Gérard Michel

Chose promise... Le temps semble maintenant venu de préciser la nature du "dernier résultat" maintes fois rencontré dans les deux premiers articles, notamment au sujet des instructions de test et branchement BEQ et BNE.
Au travers des problèmes déjà traités, il ressort que le "dernier résultat" se comporte en fait comme un indicateur caractéristique du résultat de la dernière opération effectuée par le processeur. Ainsi, après LDA $18, nous étions capables de savoir si l'accumulateur contenait 0 ou non, en testant la valeur du "dernier résultat" par BEQ ou BNE. De même, après CMP $25, le "dernier résultat" pouvait nous indiquer si l'accumulateur et l'adresse $25 contenaient la même valeur; dans l'affirmative, BEQ provoquait un branchement mais le programme continuait en séquence dans le cas contraire.
Avant d'aller plus loin sur cette notion "d'indicateur", nous devons toutefois examiner rapidement la façon dont sont représentées les informations pour le processeur.

La représentation des nombres

Pour votre Apple, la quantité économique d'information c'est l'octet. Ce n'est pas pour autant son langage "naturel" puisque, comme chacun sait, l'ordinateur ne dispose que de nombres binaires (bits) pour exprimer sa pensée et comprendre la vôtre...
Un octet se compose donc de huit bits, soit huit chiffres ne pouvant prendre chacun que la valeur 0 ou 1. De ce fait, l'octet peut représenter une valeur comprise entre "00000000" et "11111111". Sans entrer dans les détails de l'algèbre binaire, rappelons que le nombre "11111111" se convertit en décimal de la façon suivante, en traitant les bits de la droite vers la gauche (du bit 0 vers le bit 7) :

(1*(2^0)) + (1*(2^1)) + (1*(2^2)) + (1*(2^3)) + (1*(2^4)) + (1*(2^5)) + (1*(2^6)) + (1*(2^7)) = 1 + 2 + 4 + 8 + 16 + 32 +64 + 128 = 255

Ceci vous explique en outre pourquoi nous avons pu jadis affirmer que la valeur d'un octet, et donc d'un registre ou d'une adresse-mémoire, devait se situer entre 0 et 255 en décimal.

Mais une question se pose maintenant : et l'hexadécimal, dans tout cela ! Rassurez-vous, il n'est pas loin. Comme un octet peut également s'écrire au moyen de deux chiffres hexadécimaux, il semble naturel de supposer que le chiffre de gauche correspond aux quatre bits de gauche et celui de droite aux quatre bits de droite. Ainsi, "11111111" = "1111" "1111" = "1+2+4+8" "1+2+4+8" = "15" "15" = "F" "F" = $FF... et la boucle est bouclée. Cette explication n'est sans doute pas très mathématique, mais elle suffira pour guider la suite du propos.

En effet, lors de l'examen de l'instruction BEQ, nous avons évoqué la possibilité de déplacements "négatifs" vers le haut du programme en disant simplement que $FF correspondait à −1 en décimal, $FE à −2, et ainsi de suite jusqu'à $80 pour −128. Il s'agit donc de savoir par quel miracle un même nombre peut valoir à la fois +255 et −1 !
L'explication réside dans la pauvreté de l'octet en tant que quantité d'information, puisqu'il faut représenter au moyen de huit bits seulement des nombres positifs ET négatifs. La convention adoptée en la matière confère au bit le plus à gauche (bit 7) le rôle particulier de "bit de signe" : lorsque ce bit vaut 0, le nombre est considéré comme positif, et inversement, un nombre est considéré comme négatif lorsque son bit 7 vaut 1.
Les nombres positifs vont donc de "00000000" à "01111111", soit 0 à +127 ou encore $00 à $7F, tandis que les nombres négatifs vont de "10000000" à "11111111", soit −128 à −1 ou encore $80 à $FF.
L'ambiguïté provient finalement de ce que le signe ne présente d'intérêt que si l'on doit en tenir compte pour les besoins du programme (merci Monsieur de La Palice). $FF vaut toujours 255, évidemment, mais si l'on doit manipuler la valeur −1, il faut savoir que le processeur la code également au moyen de $FF.

Mais, direz-vous, à quoi bon se fatiguer avec des nombres négatifs sur un octet puisque la représentation des nombres réels réclamera inévitablement un système plus complexe de plusieurs octets et une codification du signe indépendante de celle de la valeur ? Certes, mais il n'en reste pas moins qu'il faut pouvoir effectuer des soustractions sur des nombres d'un seul octet, et comment calculer "5−4" si l'on ne sait calculer "5+(−4)"...

Additions et soustractions en binaire

Les règles de l'addition sont simples :
0 + 0 = 0
1 + 0 = 0 + 1 = 1
1 + 1 = 0 avec une retenue de 1
1 + 1 + (retenue) = 1 avec retenue

Quelques exemples :
00000001 + 00000001 = 00000010
00100000 + 00111111 = 01011111
11111111 + 00000001 = 00000000 avec retenue "extérieure" de 1

Dans le dernier exemple, la retenue sort de l'octet. Nous n'avons pas à en tenir compte pour le résultat arithmétique de notre calcul sur un octet ($FF + $01 = $00 de même que 9 + 1 = 0 en décimal si l'on ne compte que sur un chiffre), mais nous verrons plus loin que le processeur conserve la trace de ce neuvième bit extérieur.

Abordons maintenant le problème plus délicat des soustractions.
Pour calculer 5−3, nous devons ajouter −3 à 5, et donc déterminer tout d'abord la représentation de −3. La méthode utilisée s'appelle le "complément à 2" et suppose deux étapes dont l'une, baptisée "complément à 1", consiste à remplacer tous les 1 d'un octet par des 0 et inversement tous ses 0 par des 1 :
3 = 00000011
Complément à 1 de 3 =11111100
On ajoute 1 à la valeur précédente :
complément à 2 de 3 = 11111101

notre soustraction donne donc finalement :
00000101 + 11111101 − 00000010 avec une retenue extérieure dont on ne tient pas compte. On vérifie bien que 5−3 = 2.

Calculons maintenant (−5) + (−3) :
5 = 00000101, d'où complément à 1 : 11111010
−5 = 11111011
(−5) + (−3) = 11111011 + 11111101 = 11111000 avec retenue de 1
11111000 est un nombre négatif (bit 7 à 1), mais quel nombre ? Pour le savoir, faisons l'inverse du complément à 2, en retirant d'abord 1 à notre nombre, ce qui revient à lui ajouter −1, soit $FF = 11111111 :
11111000 + 11111111 = 11110111
Complément à 1 de 11110111 = 00001000 = 1*(2^3) = 8
11111000 est donc égal à −8 (complément à 2 de +8).
Une autre méthode pour déterminer à quel nombre négatif en décimal correspond un octet dont le bit 7 est à 1 consiste à le convertir tout d'abord en valeur absolue (sans tenir compte du signe), puis à retirer 256 à la valeur ainsi obtenue. Ici, 11111000 = 248, et 248−256 = −8.
Tout semble par conséquent aller pour le mieux dans le meilleur des mondes possibles...

Hélas, ce n'est qu'une apparence fragile ! Calculons par exemple 64+64 :
64 = 01000000
64 + 64 =01000000 + 01000000 = 10000000
10000000 est un nombre négatif, −128 plus précisément, et donc 64+64 = −128 !

Surprenant, n'est-ce pas ? Mais vous aurez noté que notre calcul est arithmétiquement juste car, si l'on ne se préoccupe pas du signe, 10000000 vaut également +128, ce qui est plus convenable. Le résultat n'est qu'algébriquement faux, en raison de la retenue qui s'est opérée du bit 6 sur le bit 7 en provoquant un changement de signe intempestif. Ce phénomène est baptisé "débordement".

Autre exemple de débordement : −127 + −62 = 10000001 + 11000010 = 01000011 avec retenue extérieure. 01000011 vaut +67 en décimal, d'où résultat erroné. Mais raisonnons une fois encore en négligeant le signe :
10000001 = 129 = $81
11000010 = 194 = $C2
10000001 + 11000010 = 129 + 194 = $81 + $C2 = 323 = $143, soit un résultat sur deux octets, avec poids fort de $01 (256) et poids faible de $43 (67). En ne tenant compte que de l'octet de poids faible, la valeur +67 est donc correcte. En d'autres termes, le débordement ne conduit réellement à des résultats faux que si le signe du résultat est pertinent pour l'utilisation qui doit être faite de ce résultat. C'est pourquoi vous avez pu trouver dans Pom's de nombreux programmes en assembleur dans lesquels étaient effectuées des additions et des soustractions sans que le programmeur semblât se soucier de l'éventualité d'un débordement. Si nous en avons parlé ici, c'est surtout parce que, au même titre que la "retenue extérieure" ou le "dernier résultat = 0", le débordement fait partie de ces indicateurs caractéristiques que nous allons maintenant examiner plus en détail.

Le registre d'état du microprocesseur

Comme les autres registres (accumulateur ou registres d'index X et Y), le registre d'état peut être considéré comme une variable interne au microprocesseur. Il s'agit donc d'un octet, dont le programmeur n'a jamais à préciser le nom ou l'emplacement mémoire, que le microprocesseur gère automatiquement et auquel certaines instructions de l'assembleur font directement référence sous le nom P (au même titre que A pour l'accumulateur).
Mais chaque bit du registre d'état revêt une signification particulière, alors que les autres registres sont plus généralement manipulés au niveau de l'octet. Plus précisément, le registre d'état doit refléter "l'état" du microprocesseur à l'issue de la dernière opération significative effectuée et, de fait, c'est lui qui nous renseigne sur les caractéristiques du "dernier résultat". Chaque bit du registre d'état joue ainsi le rôle d'indicateur, en ce qu'il conserve le statut de la dernière opération vis-à-vis des critères donnés comme significatifs pour la machine (dernier résultat nul, retenue externe, débordement dans les calculs...).

Sur les huit bits de l'octet composant le registre d'état, sept sont réellement pertinents pour le processeur de l'Apple :

- Bit 7 : indicateur de signe (baptisé N dans la littérature). Ce bit vaut 0 si la dernière opération donne un résultat positif et 1 dans le cas contraire (le bit 7 de l'octet contenant le résultat est alors à 1). Notons dès maintenant que les bits d'état sont positionnés selon la même méthode que pour les opérateurs logiques : 1 veut dire "vrai" (N vaut 1 si le dernier résultat est négatif) et 0 veut dire "faux".
- Bit 6 : indicateur de débordement (baptisé V) : vaut 1 si la dernière opération provoque un débordement (voir ci-dessus l'addition 64+64) et 0 sinon.
- Bit 5 : inutilisé.
- Bit 4 : indicateur de "BREAK". Nous en préciserons la fonction dans un article ultérieur.
- Bit 3 : indicateur "décimal codé binaire", dont nous réserverons également l'analyse pour nos prochaines rencontres.
- Bit 2 : indicateur "interruption", avec même remarque que ci-dessus car à chaque jour suffit sa peine...
- Bit 1 : indicateur "zéro" (baptisé Z). Ce bit vaut 1 lorsque la dernière opération donne un résultat nul et 0 dans le cas contraire. Vous subodorez donc déjà que cet indicateur Z doit avoir quelque rapport avec nos instructions BEQ et BNE (ne vous découragez pas pour autant si vous ne subodorez rien...).
- Bit 0 : indicateur de retenue (C pour les intimes !). Il vaut 1 lorsque la dernière opération provo-

que une retenue et 0 sinon. Nous avions dit précédemment que le processeur pouvait tenir compte des retenues "extérieures" occasionnées par des opérations sur des nombres codés en un seul octet, au travers d'un "neuvième" bit. Vous ne serez donc pas surpris d'apprendre que c'est l'indicateur C qui joue ce rôle de neuvième bit : ne vaut-il pas justement 1 lorsqu'une retenue sort de l'octet et qu'un "1" doit par conséquent être stocké quelque part dans l'hypothèse où l'opération porterait sur plus d'un octet (songez à ce que vous faisiez à l'école pour additionner 9 et 1 : "je pose 0 et je retiens 1". Le processeur retient tout simplement 1 dans le bit C de son registre d'état).

Le langage assembleur autorise deux types de manipulation sur les bits du registre d'état : test et branchement (voir BEQ et BNE), mais aussi positionnement "forcé" de certains indicateurs. Nous examinerons dans l'ordre ces deux types d'instructions.

Test et branchement

Nous limitons pour l'heure notre propos aux indicateurs N (signe), V (débordement), Z (zéro) et C (retenue).

Deux des instructions de test et branchement nous sont d'ores et déjà connues : BEQ provoque un branchement si "dernier résultat = 0", ce qui revient à dire que le bit Z vaut 1 ("dernier résultat = 0" est vrai), tandis que BNE provoque un branchement si "dernier résultat <> 0" (bit Z vaut 0).
Il existe de même un couple d'instructions pour chacun de nos trois autres bits du registre d'état.

Bit N (signe)

- BMI (30) : branchement si négatif, c'est-à-dire si le bit N vaut 1.
- BPL (10) : branchement si positif (bit N vaut 0).

Bit V (débordement)

- BVS (70) : branchement si "dernier résultat provoque un débordement" (bit V = 1).
- BVC (50) : branchement si pas de débordement (bit V = 0).

Bit C (retenue)

- BCS (B0) : branchement si "dernier résultat provoque une retenue" (bit C = 1).
- BCC (90) : branchement si pas de retenue (bit C = 0).

Bit Z (zéro)

BEQ et BNE sont, bien sûr, les deux instructions concernées.

Les règles du branchement sont les mêmes pour toutes les instructions ci-dessus que pour BEQ et BNE : il s'agit toujours d'un déplacement de X octets vers le haut ou le bas selon la valeur donnée à la suite du code de l'instruction. Rappelons en outre que c'est le microprocesseur qui positionne tout seul ses indicateurs N, V, Z et C en fonction du travail qui lui est demandé et que nous nous contentons, par des instructions du type B.., de tester la valeur de ces indicateurs après les opérations, afin de décider de la suite des traitements à effectuer.

Puisque nous parlerons moins dans la suite de l'article de l'indicateur V que des autres, précisons qu'il peut servir notamment à corriger des résultats pour lesquels le signe est significatif (ainsi, BVS devrait alors entraîner une action correctrice sur le signe du résultat - modification du bit 7), et qu'il peut également servir de "drapeau" dans les sous-routines (si, par exemple, le bit V n'a aucune chance d'être modifié à l'intérieur d'une telle routine, on peut lui donner la valeur 0 ou 1 avant l'appel de la sous-routine, selon le type d'appel, et orienter ensuite les traitements par des tests BVC ou BVS).

"Forcer" la valeur des bits d'état

Le terme "forcer" est fréquemment utilisé lorsque l'on traite de langage machine ou d'assembleur, pour indiquer que l'on impose à certaines variables caractéristiques des valeurs qui ne sont pas nécessairement celles que leur donne le processeur en fonction des opérations effectuées.
C'est bien le terme qui s'applique ici puisque le programmeur peut manipuler par certaines instructions la valeur des bits du registre d'état, et ce indépendamment de toute autre opération.

Bit C (retenue)

CLC (18) : met à 0 le bit C.
SEC (38) : met à 1 le bit C.

Bit V (débordement)

CLV (B8) : met à 0 le bit V.

Ce sont là les seules instructions "directes" dont nous disposons pour forcer les bits du registre d'état (du moins les quatre bits que nous examinons pour l'instant). On peut cependant manipuler les autres bits par des moyens indirects. Ainsi LDA #$0 met le bit Z à 1 et N à 0 (dernier résultat nul mais positif), LDA #$FF met Z à 0 et N à 1 (dernier résulat différent de 0 et négatif, puisque $FF = 11111111, avec bit 7 à 1, ce qui caractérise pour le processeur un nombre négatif). De même, ajouter #$40 à #$40 met à 1 le bit V ($40 = 64 = 01000000, 01000000 + 01000000 = 10000000, d'où retenue du bit 6 sur le bit de signe, ce qui positionnera dans le cas présent l'indicateur de débordement à 1 en raison de l'erreur au niveau du résultat : 64 + 64 = −128).

Mais, une fois encore, un murmure court dans la foule : ajouter #$40 à #$40, certes, mais comment ? Voici la réponse.

Addition d'un octet à un autre

Le microprocesseur de votre Apple dispose d'une instruction pour ajouter une valeur codée sur un octet à une autre valeur sur un octet. Le code mnémonique en est **ADC** : elle ajoute à l'accumulateur la valeur donnée après ADC (soit sous forme d'un nombre - #$10 par exemple - soit sous forme d'une variable définie par l'un des modes d'adressage que nous avons déjà étudiés). Mais elle ajoute en outre au résultat ainsi obtenu la valeur du bit C (0 ou 1). Pourquoi ? Tout simplement pour permettre les additions sur deux octets. Voyons tout d'abord les différents formats de l'instruction ADC avant de revenir sur ce point.

ADC nombre = ADC #$XX = 69 XX (où XX désigne un nombre de deux chiffres hexadécimaux)

ADC contenu d'une adresse en page zéro = ADC ADR = ADC $aa = 65 aa (par exemple, 65 18)

ADC contenu d'une adresse quelconque = ADC ADR = ADC $aaaa = 6D octet bas/octet haut

ADC ADR,X = ADC $aaaa,X = 7D octet bas/octet haut

ADC ADR,Y = ADC $aaaa,Y = 79 octet bas/octet haut

ADC adresse page zéro,X = ADC ADR,X = ADC $aa,X = 75 adresse

ADC (ADR,X) = ADC ($aa,X) = 61 adresse

ADC (ADR),Y = ADC ($aa),Y = 71 adresse

Notez bien que c'est toujours à l'accumulateur que l'instruction ADC ajoute quelque chose, le résultat de l'addition étant lui aussi stocké automatiquement dans l'accumulateur. Puisque cette addition intègre également le bit C du registre d'état, il faut veiller à ce que ce dernier soit bien à 0 avant ADC si l'on ne veut prendre en compte que la valeur de l'accumulateur et celle de l'octet qui lui est ajouté. Comme il n'est pas toujours facile de savoir comment sont positionnés les indicateurs du registre d'état à l'issue des diverses ins-

tructions exécutées précédemment, l'instruction CLC peut rendre ici de grands services.
Supposons ainsi que l'on veuille ajouter au contenu de l'adresse $18 celui de l'adresse $DD60 et conserver le résultat en $18. Le programme réalisant ce traitement serait le suivant :

LDA $18 (A5 18)
CLC (18)
ADC $DD60 (6D 60 DD)
STA $18 (85 18)

L'instruction CLC pourrait être supprimée si nous étions sûrs de la valeur 0 du bit C avant l'addition.

L'exécution de ADC fournit un "dernier résultat" significatif pour le processeur, et le registre d'état se trouve donc reconfiguré en fonction de ce dernier résultat. Sont affectés par l'opération les bits :

N : mis à 1 si le résultat de l'addition est négatif (bit 7 de l'accumulateur à 1 après ADC), à 0 sinon.
V : mis à 1 s'il y a débordement, à 0 sinon.
Z : mis à 1 si l'addition donne un résultat nul, à 0 sinon.
C : mis à 1 si l'addition des deux nombres provoque une retenue extérieure, à 0 sinon.

Cette mise à jour de la retenue C permet d'effectuer des additions avec des nombres de deux octets, sur le même principe que pour les nombres décimaux (9 + 1 = 0, et la retenue de 1 vient s'ajouter au chiffre des dizaines). Reprenons notre exemple précédent, en supposant cette fois que la valeur à laquelle on doit ajouter le contenu de l'adresse $DD60 est stockée en $18 (poids faible) et $19 (poids fort). Deux solutions sont envisageables pour réaliser l'opération.

Programme 1

LDA $18 (A5 18)
CLC (18)
ADC $DD60 (6D 60 DD)
STA $18 (85 18)
LDA $19 (A5 19)
ADC #$0 (69 00)
STA $19 (85 19)

Après avoir ajouté à l'octet d'adresse $18 la valeur de $DD60 et stocké le résultat en $18, on charge l'accumulateur avec la valeur de $19. On ne sait pas, a priori, si le premier ADC a produit ou non une retenue, mais on sait en revanche que le bit C est positionné en conséquence. En ajoutant alors 0 à l'accumulateur, sans toucher à C, on lui ajoute en fait 0+0 s'il n'y a pas de retenue, ou 0+1 s'il y en a une. La valeur de l'octet de poids fort remise ensuite en $19 tiendra donc bien compte du résultat complet de l'addition effectuée sur l'octet de poids faible.

Si, par exemple, $18 contient #$FE et $19 #$03, on trouvera dans ces deux adresses #$FF et #$03 après l'opération si $DD60 contient #$01, mais on y trouvera #$00 et #$04 si $DD60 contient #$02.

Programme 2

LDA $18 (A5 18)
CLC (18)
ADC $DD60 (6D 60 DD)
STA $18 (85 18)
BCC suite du programme (90 02)
INC $19 (E6 19)
... suite du programme

On utilise cette fois la possibilité qui nous est offerte de tester le bit C après l'addition sur l'octet de poids faible. Si C=0, il n'y a pas de retenue, l'octet de poids fort est donc correct et BCC nous amènera directement à la suite du programme. Si C=1, en revanche, BCC ne provoquera pas de branchement et, en incrémentant $19 avant de passer à la suite, on aura bien rétabli la valeur correcte du poids fort en fonction de la retenue qui s'y ajoute.
Cette solution, plus élégante, plus courte et plus rapide, sera préférée à la précédente.

Si l'on veut enfin ajouter deux nombres codés en deux octets (le premier en $18-$19 et le second en $DD60-$DD61, par exemple, toujours dans l'ordre poids faible/poids fort), en gardant le résultat dans le premier, le programme devient :

LDA $18 (A5 18)
CLC (18)
ADC $DD60 (6D 60 DD)
STA $18 (85 18)
LDA $19 (A5 19)
ADC $DD61 (6D 61 DD)
STA $19 (85 19)

Ce programme ressemble fort au programme 1 ci-dessus, mais cette fois on additionne bien les deux poids forts, plus la retenue éventuelle provenant de l'addition des poids faibles.

Soustraction d'un octet à un autre

L'instruction équivalente à ADC pour la soustraction est **SBC**. Elle retire de l'accumulateur la valeur donnée après SBC (nombre ou contenu d'une adresse), mais aussi l'opposé du bit C, c'est-à-dire 0 si C=1 et 1 si C=0. Le résultat est stocké dans l'accumulateur.

Cette prise en compte de l'opposé de la retenue doit évidemment permettre les soustractions sur deux octets.

Considérons par exemple deux octets quelconques O1 et O2, dont on sait simplement que O2 est supérieur à O1 (sans se préoccuper de signe). Nous appelerons B le bit de rang le plus élevé qui est à 1 dans O2 et à 0 dans O1 (si O1=00010011 et O2=00101111, B est le bit de rang 5, i.e. le troisième en partant de la gauche), et nous désignerons par R le rang correspondant (R=5 dans notre exemple). Sur la gauche de R, tous les bits de O1 et O2, s'il en reste (si R < 7), ont la même valeur au même rang, 0 ou 1. On veut maintenant calculer O1 −O2. Pour ce faire, on calcule d'abord le complément à 1 de O2, que nous noterons C1-O2. Dans C1-O2, B est donc à 0. Lorsqu'on ajoute ensuite 1 (00000001) à C1-O2 pour obtenir le complément à 2 de O2, noté C2-O2, deux cas sont envisageables :

- B reste à 0. Dans l'addition de C2-O2 à O1, pour calculer O1−O2, on a donc deux bits à 0 au rang R qui "absorberont" inévitablement toute retenue qui se propagerait à partir de l'addition des bits de rang inférieur à R (0 + 0 + retenue est au plus égal à 1, mais sans retenue à reporter à gauche). Quant aux bits situés à gauche de R, s'ils étaient à 1 dans O2, et donc dans O1, ils sont à 0 dans C2-O2 ; et s'ils étaient à 0 dans O2, et donc dans O1, ils sont à 1 dans C2-O2. A gauche de R, on ne peut donc ajouter que des 0 à des 1, sans retenue venant de la droite (il ne peut y en avoir au rang R) et, par conséquent, il ne peut y avoir de retenue à l'extérieur de l'octet représentant le résultat de O1−O2.

Exemple :
O1 = 00101011 (43 en décimal)
O2 = 01000100 (68) : B est le bit de rang 6
C1-O2 = 10111011
C2-O2 = 10111100 (B reste à 0)
O1 + C2-O2 = 00101011 + 10111100 = 11100111 (−25) sans retenue.

- B repasse à 1. Ceci n'est possible que si tous les bits situés à droite de R sont à 1 dans C1-O2 ou si B est de rang 0. Quand on ajoute C2-O2 à O1, on a donc, au rang R, un 0 dans O1 et un 1 dans C2-O2. Une retenue pourrait ainsi se propager au-delà de R, si elle venait de l'addition des bits situés à droite de R. Mais, si tous les bits en question étaient à 1 dans C1-O2, ils sont nécessairement à 0 dans C2-O2 (par exemple, 00001111 + 00000001 = 00010000). A droite de R, on n'ajoute donc que des 0 à autre chose (0 ou 1) et il ne peut y avoir de retenue se décalant vers la gauche jusqu'au rang R (dans le cas où B est de rang 0, le problème ne se pose même pas : on

ajoute 1 à 0 au rang 0 sans retenue). Pour les bits de rang supérieur à R, même remarque que ci-dessus; là encore, il ne peut y avoir de retenue extérieure dans le résultat de O1−O2.

Exemple :
O1 = 00101011
O2 = 01000000 (64) : B est le bit de rang 6
C1-O2 = 10111111
C2-O2 = 11000000 (B passe à 1)
O1 + C2-O2 = 00101011 + 11000000 = 11101011 (21) sans retenue.

En conclusion, après la soustraction d'un octet O2 à un octet O1 qui lui est inférieur, il n'y a jamais de retenue extérieure et C est donc toujours à 0.

Examinons maintenant le cas inverse, la soustraction O1−O2 lorsque O2 est inférieur à O1. B est le bit de rang le plus élevé (R) qui est à 0 dans O2 alors que le bit de même rang est à 1 dans O1. Dans C1-O2, B est donc à 1 et, pour C2-O2, deux hypothèses sont toujours possibles :

- B reste à 1. Dans l'addition O1 + C2-O2, on trouve au même rang R un bit à 1 dans O1 et un bit à 1 dans C2-O2, d'où apparition d'une retenue à reporter à gauche. A gauche de R, on a toujours au même rang un 0 dans O1 et un 1 dans C2-O2, ou un 1 dans O1 et un 0 dans C2-O2. Au rang R+1, s'il existe, on ajoute donc 1 + 0 + retenue de 1, la retenue se propage en R+2, et ainsi de suite jusqu'à l'extérieur de l'octet (si R=7, la retenue opérée au rang R tombe de même directement à l'extérieur de l'octet).

Exemple :
O1 = 00101011
O2 = 00010110 (22) : B est le bit de rang 5
C1-O2 = 11101001
C2-O2 = 11101010 (B reste à 1)
O1 + C2-O2 = 00101011 + 11101010 = 00010101 (21) avec retenue.

- B passe à 0. Ceci n'est possible que si une retenue s'est propagée de la droite jusqu'au rang R mais, de ce fait, elle se propage encore au moins jusqu'au rang R+1 (elle s'arrête là si le bit de rang R+1 est à 0 dans C1-O2, mais elle continue encore si ce dernier est déjà à 1). Supposons tout d'abord qu'il y ait au moins un bit à 0 de rang supérieur à R dans C1-O2, qui se retouvera donc à 1 dans C2-O2. Ce bit était à 1 dans O2, tout comme le bit correspondant dans O1. Dans l'opération O1 + C2-O2, on a donc à ce rang 1+1, d'où retenue vers la gauche qui se propagera jusqu'à l'extérieur de l'octet puisque l'on ajoute toujours pour les autres rangs 0 + 1 + retenue de 1.
  Pour qu'il n'y ait par contre aucun bit à 0 de rang supérieur à R dans C1-O2, tous les bits concernés doivent être à 0 dans O2. Par ailleurs, la retenue qui arrive en R pour mettre B à 0 dans C2-O2 provient obligatoirement de ce que tous les bits à droite de R sont à 1 dans C1-O2, ce qui implique qu'ils étaient également à 0 dans O2. Comme B est aussi à 0 par définition, tous les bits de O2 sont nuls : O2=00000000 = 0. C'est bien la seule valeur de O2 pour laquelle la retenue dégagée au rang R dans C2-O2 lorsque le bit B repasse de 1 à 0 ne conduit pas finalement à une retenue extérieure dans le résultat O1 + C2-O2, soit O1−O2.

Exemple :
O1 = 00101011
O2 = 00100000 (32) : B est le bit de rang 3
C1-O2 = 11011111
C2-O2 = 11100000 (B passe à 0)
O1 + C2-O2 = 00101011 + 11100000 = 00001011 (11) avec retenue.

En conclusion, après la soustraction d'un octet O2 à un octet O1 qui lui est supérieur, il y a toujours une retenue extérieure (C=1), sauf si O2 = 0.

Toutes ces explications n'ont d'autre but que d'aider à l'analyse de l'instruction SBC car, si l'utilisation de la retenue C paraît assez "naturelle" dans ADC, elle s'avère beaucoup moins évidente dans SBC (par assimilation avec le décimal, on s'attendrait plus à retirer la retenue elle-même que son opposé). Récapitulons. Si l'on soustrait quelque chose d'un nombre sur deux octets, le poids fort du nombre doit être diminué de 1 si la valeur retirée de son poids faible lui est supérieure (si on calcule par exemple #$0100−#$FF on doit trouver #$0001 : poids faible = #$00−#$FF = #$01 / poids fort = #$01 diminué de 1 = #$00). Si la valeur est par contre inférieure au poids faible, le poids fort ne doit pas être affecté par cette soustraction. Or nous savons maintenant que si l'on retire une valeur trop grande d'un autre octet, il n'y a jamais de retenue extérieure : à l'issue de la soustraction, C=0. Si l'on fait ensuite un SBC sur le poids fort, et ce indépendamment de ce que l'on met derrière SBC, l'instruction retirera de toute façon l'opposé de C au poids fort, soit 1 puisque C=0, et le résultat sera correct.
Inversement, si l'on retire au poids faible une valeur qui lui est inférieure, on aura C=1 à l'issue de la soustraction. Un SBC sur le poids fort retirera de ce dernier l'opposé de C, soit 0, le laissant donc correctement inchangé. Il faudrait toutefois se méfier du cas où la valeur retirée au poids faible n'est autre que 0, car il n'y a pas alors de retenue extérieure et l'opposé de C vaut 1 (on pourrait ainsi calculer que #$0101−#$00 = #$0001 !). Heureusement, le processeur prend ce problème en compte et positionne toujours C à 1 lorsque la valeur soustraite est 0.

Reste le cas particulier de l'égalité des deux valeurs : si l'on calcule O1−O1, le résultat obtenu sera bien 0, mais quid du bit C ? Il est clair que dans le complément à 1 de O1 (C1-O1), tous les bits qui étaient à 0 dans O1 se retrouvent à 1 et réciproquement. Si le bit de rang 0 est à 0 dans C1-O1, il sera à 1 dans le complément à 2 (C2-O1); s'il est à 1, il sera à 0 mais une retenue se propagera vers la gauche dans C2-O1 et cette retenue transformera à un moment donné un bit à 0 dans C1-O1 en un bit à 1 dans C2-O2 (sauf s'il n'y a que des bits à 1 dans C1-O1, ce qui revient à dire que O1−0 et nous ramène au problème spécifique du zéro). A un rang R quelconque, on aura donc dans C2-O1 un bit revenu à la même valeur 1 que dans O1. Les bits à droite de R (si R>0) seront à 0 après propagation de la retenue (ils étaient à 1 dans C1-O1 et à 0 dans O1) : pour les rangs inférieurs à R, on ajoute ainsi 0+0. Au rang R, on ajoute 1+1, d'où retenue vers la gauche qui se propagera jusqu'à l'extérieur de l'octet résultat O1−O1, puisque, aux rangs supérieurs à R, on additionne toujours "0+1 + retenue de 1" ou "1+0 + retenue de 1". A l'issue de la soustraction, C=1 : une opération comme #$2450−#$50 ou #$24FE−#$14FE produirait donc un résultat correct.

Exemple 1 :
O1 = 00101011
C1-O1 − 11010100 (bit de rang 0 à 0)
C2-O1 = 11010101
O1 + C2 O1 = 00000000 avec retenue.

Exemple 2 :
O1 = 01000100
C1-O1 = 10111011 (bit de rang 0 à 1)
C2-O2 = 10111100
O1 + C2-O1 = 00000000 avec retenue.

Ces considérations sur l'arithmétique binaire peuvent vraisemblablement vous paraître un peu trop complexes, surtout en comparaison avec les deux premiers articles d'initiation à l'assembleur. De fait, il n'est ni interdit, ni impossible, d'écrire des programmes en assembleur, même de grande taille et de haute ambition,

sans savoir ce qu'il advient, ou doit advenir, de tel ou tel bit à l'occasion de telle ou telle opération. Il suffit pour cela de bien assimiler le fonctionnement des instructions et de respecter leurs règles (savoir, par exemple, qu'il faut toujours faire SEC avant une première soustraction et que les problèmes de retenue éventuels seront pris en charge par le deuxième SBC si l'on ne fait pas SEC avant). Toutefois, si vous en avez le goût et le temps, il vous sera bénéfique de creuser un peu ces questions, ne serait-ce que pour mieux comprendre les mécanismes, mais aussi parce que d'autres instructions de l'assembleur, qu'il nous reste encore à examiner, ne "travaillent" qu'au niveau du bit (de donnée ou du registre d'état) et produisent des résultats que l'on comprend mal si l'on ne s'attache pas un tant soit peu au contenu des octets.

Voici maintenant la liste des différents formats de l'instruction SBC.

SBC valeur = SBC #$NN = E9 NN

SBC adresse en page zéro = SBC ADR = SBC $aa = E5 aa

SBC adresse quelconque = SBC ADR = SBC $aaaa = ED octet bas/octet haut

SBC "adresse page zéro",X – SBC ADR,X = SBC $aa,X = F5 aa

SBC "adresse quelconque",X = SBC ADR,X = SBC $aaaa,X = FD octet bas/octet haut

SBC "adresse quelconque",Y = SBC ADR,Y = SBC $aaaa,Y = F9 octet bas/octet haut

SBC (ADR,X) = SBC ($aa,X) = E1 aa

SBC (ADR),Y = SBC ($aa),Y = F1 aa

Rappelons que, comme pour ADC, la donnée à laquelle on retire une autre donnée (indiquée explicitement ou obtenue par l'un des modes d'adressage précisés ci-dessus) est toujours contenue dans le registre accumulateur.

Mis à part le bit C dont il a déjà été longuement question, l'exécution de SBC affecte les mêmes indicateurs du registre d'état que celle de ADC, à savoir N, V et Z, et ce exactement selon les mêmes règles.

Sachant que SBC retire l'opposé de la retenue du résultat de la soustraction proprement dite, il faut s'assurer que le bit C est bien à 1 lorsque l'on travaille sur des nombres d'un seul octet ou sur les octets de poids faible. Si un doute plane sur ce point au moment de procéder à l'opération, on utilisera l'instruction SEC pour l'éliminer.

Vous trouverez ci-après les petits programmes présentés précédemment pour ADC, adaptés au cas de soustractions par SBC.

Exemple 1

```
LDA $18 (A5 18)
SEC (38)
SBC $DD60 (ED 60 DD)
STA $18 (85 18)
```

Exemple 2 - programme 1

```
LDA $18 (A5 18)
SEC (38)
SBC $DD60 (ED 60 DD)
STA $18 (85 18)
LDA $19 (A5 19)
SBC #$00 (E9 00)
STA $19 (85 19)
```

Exemple 2 - programme 2

```
LDA $18 (A5 18)
SEC (38)
SBC $DD60 (ED 60 DD)
STA $18 (85 18)
BCS suite du programme (B0 02)
DEC $19 (C6 19)
suite du programme
```

Exemple 3

```
LDA $18 (A5 18)
SEC (38)
SBC $DD60 (ED 60 DD)
STA $18 (85 18)
LDA $19 (A5 19)
SBC $DD61 (ED 61 DD)
STA $19 (85 19)
```

Bref retour sur CMP, CPX, CPY et les autres...

Dans l'article précédent, nous avions dit que la comparaison de "quelque chose" avec l'un des registres du processeur revenait en fait à une soustraction virtuelle de ce "quelque chose" au registre concerné, avec positionnement du "dernier résultat" en conséquence. Il est clair maintenant que ce sont certains bits du registre d'état qui sont affectés par ces opérations de comparaison, comme le bit Z, par exemple, dont nous avions déjà envisagé l'utilisation en sortie d'une comparaison par les instructions BEQ et BNE.

Sont également positionnés par CMP, CPX ou CPY les indicateurs N et C. N ne donnera toutefois une information correcte que s'il n'y a pas de débordement. Le bit C est beaucoup plus intéressant pour nous : à l'issue de la comparaison, il est à 1 si le contenu du registre est supérieur ou égal au "quelque chose", et à 0 dans le cas contraire (on retrouve là le même comportement de C que dans les soustractions).
En sus de l'égalité ou de l'inégalité, on peut donc tester :
Registre < "quelque chose" par BCC
Registre >= "quelque chose" par BCS
Registre > "quelque chose" par BEQ suivi de BCS (on traite d'abord l'éventualité d'une égalité par BEQ et on ne parvient donc jusqu'à l'instruction BCS que si le contenu du registre est bien le plus grand).

La liste suivante donne, en regard de chaque instruction déjà traitée dans les articles précédents, les indicateurs d'état affectés par son exécution, lorsque celle-ci exerce effectivement une influence sur le registre d'état (STA, par exemple, n'a aucune action sur le registre) :
LDA : N, Z
LDX : N, Z
LDY : N, Z
INX : N, Z
INY : N, Z
DEX : N, Z
DEY : N, Z
TAX : N, Z
TAY : N, Z
TXA : N, Z
TYA : N, Z
INC : N, Z
DEC : N, Z
PLA : N, Z

Exemples d'application

Exemple 1

Il s'agit d'une routine d'entrée et sortie de caractères à laquelle on veut donner les caractéristiques suivantes :
- Le curseur sera un "–" clignotant.
- Les lettres de A à Z seront affichées en NORMAL.
- Les chiffres de 1 à 9 seront affichés en INVERSE.
- Tous les autres caractères seront affichés en FLASH.
- On pourra sortir de la routine par ESC.
- On refusera en saisie tous les caractères de contrôle, sauf RETURN et les flèches à droite et à gauche.

Pour analyser cette routine, il faut disposer de quelques indications sur la table des codes-écran de l'Apple :
- Le code reçu du clavier correspond toujours au code-écran "mode normal" du caractère correspondant.
- Les codes "clavier" des caractères de contrôle vont de $80 à $9F.
- Les codes "clavier" des lettres vont de $C1 à $DA.
- Les codes "clavier" des chiffres vont de $B0 à $B9 et il faut leur retirer $80 pour passer aux codes "écran" en inverse.
- Les autres caractères vont de $A0 à $AF ($40 à retirer pour passer au code "clignotant"), puis de $BA à $BF (toujours $40 à retirer), $C0 en fait également partie ($80 à retirer pour le code "clignotant"), et enfin les codes de $DB à $DF ($80 à soustraire). Nous ne nous préoccupons pas ici des minuscules.

Exemple 1 - Lisa 1.5

```
0300                 1
0300                 2  OUT    EQU $FDF0
0300                 3  ADB    EPZ $28
0300                 4  CAR    EPZ $6
0300                 5  H      EPZ $24
0300 2058FC          6         JSR $FC58
0303 204803          7  DEB    JSR GET
0306 C99B            8         CMP #$9B
0308 F03D            9         BEQ FIN
030A C98D           10         CMP #$8D
030C F033           11         BEQ ECRAN
030E C988           12         CMP #$88
0310 F02F           13         BEQ ECRAN
0312 C995           14         CMP #$95
0314 D006           15         BNE S5
0316 A424           16         LDY H
0318 B128           17         LDA (ADB),Y
031A D025           18         BNE ECRAN
031C C9A0           19  S5     CMP #$A0
031E 90E3           20         BCC DEB
0320 C9B0           21         CMP #$B0
0322 B005           22         BCS S0
0324 38             23         SEC
0325 E940           24         SBC #$40
0327 D018           25         BNE ECRAN
0329 C9BA           26  S0     CMP #$BA
032B B005           27         BCS S1
032D 38             28  S4     SEC
032E E980           29  S3     SBC #$80
0330 D00F           30         BNE ECRAN
0332 C9C0           31  S1     CMP #$C0
0334 F0F7           32         BEQ S4
0336 B005           33         BCS S2
0338 38             34         SEC
0339 E940           35         SBC #$40
033B D004           36         BNE ECRAN
033D C9DB           37  S2     CMP #$DB
033F B0ED           38         BCS S3
0341 20F0FD         39  ECRAN  JSR OUT
0344 4C0303         40         JMP DEB
0347 60             41  FIN    RTS
0348 A424           42  GET    LDY H
034A B128           43         LDA (ADB),Y
034C 48             44         PHA
034D A9AD           45         LDA #$AD
034F 9128           46         STA (ADB),Y
0351 AD00C0         47         LDA $C000
0354 3005           48         BMI GET1
0356 68             49         PLA
0357 9128           50         STA (ADB),Y
0359 D0ED           51         BNE GET
035B 8D10C0         52  GET1   STA $C010
035E 8506           53         STA CAR
0360 68             54         PLA
0361 9128           55         STA (ADB),Y
0363 A506           56         LDA CAR
0365 60             57         RTS
```

Récapitulation : exemple 1

```
0300- 20 58 FC 20 48 03 C9 9B
0308- F0 3D C9 8D F0 33 C9 88
0310- F0 2F C9 95 D0 06 A4 24
0318- B1 28 D0 25 C9 A0 90 E3
0320- C9 B0 B0 05 38 E9 40 D0
0328- 18 C9 BA B0 05 38 E9 80
0330- D0 0F C9 C0 F0 F7 B0 05
0338- 38 E9 40 D0 04 C9 DB B0
0340- ED 20 F0 FD 4C 03 03 60
0348- A4 24 B1 28 48 A9 AD 91
0350- 28 AD 00 C0 30 05 68 91
0358- 28 D0 ED 8D 10 C0 85 06
0360- 68 91 28 A5 06 60
```

Examinons donc ce petit programme, en commençant par la fin, c'est-à-dire la routine de saisie de caractères située aux lignes 42 à 57.

$C000 est une adresse réservée au caractère reçu du clavier : lorsque vous appuyez sur une touche, le code "clavier" du caractère correspondant se trouve automatiquement stocké à cette adresse. En décimal, $C000 est égal à 49152, ou encore −16384, et cette valeur doit vous dire quelque chose car il y est fait référence dans le manuel de l'Applesoft, au chapitre concernant les PEEKs et POKEs. On vous explique ainsi que vous pouvez utiliser l'instruction X = PEEK (−16384) et tester ensuite la valeur de X : si X > 127, une touche a été pressée et son code est X, et inversement si X <= 127. En binaire, 127 = 01111111 et les nombres compris entre 127 et 255 sont donc compris entre 10000000 et 11111111 en binaire; en d'autres termes, si X est supérieur à 127, cela revient à dire que le contenu de l'adresse $C000 (ou −16384) est un nombre binaire négatif en complément à 2 (bit 7 à 1). En assembleur, on détectera donc l'enfoncement d'une touche en regardant si le contenu de $C000 est positif ou négatif (touche pressée dans ce second cas). Le manuel de l'Applesoft fait également référence à l'adresse −16368 ($C010) qu'il faut "toucher" d'une façon ou d'une autre (par un PEEK ou un POKE) après avoir lu le clavier par PEEK (−16384), et ce afin de mettre la machine en condition de recevoir un autre caractère éventuel. On agira de même en assembleur, en adressant, d'une façon ou d'une autre, la case-mémoire $C010. Ces quelques précisions nous aideront à comprendre le fonctionnement de la routine d'entrée GET.

Lignes 42 à 44 : on charge dans l'accumulateur le caractère situé à la colonne H dans la ligne courante dont l'adresse de base est donnée par ADB et on dépose celui-ci au sommet de la pile. Cette manipulation des adresses dans la page TEXT doit vous être maintenant familière après les deux premiers articles.

Lignes 45 et 46 : #$AD est le code-écran de "−" en mode normal, et on l'affiche à la place du caractère que l'on vient d'empiler.

Ligne 47 : on regarde ce qu'il y a à l'adresse $C000 par un LDA (on sait par ailleurs que LDA exerce une action sur le bit N du registre d'état).

Ligne 48 : BMI provoquera un branchement si le bit N est à 1, donc si la valeur lue en $C000 est négative, ce qui caractérise le fait qu'une touche a été pressée. Dans ce cas, on saute à GET1.

Ligne 49 à 51 : on dépile le caractère remplacé tout à l'heure par le "−" et on l'affiche de nouveau à sa place sur l'écran, avant de retourner à GET (BNE provoquera toujours le branchement car il n'existe pas de code-clavier égal à 0). On pourrait bien sûr ne remonter que jusqu'à LDA #$AD, mais l'exécution d'instructions supplémentaires ralentit le clignotement du curseur et le rend plus perceptible.

Lignes 52 à 57 : une touche a été enfoncée et son code se trouve dans l'accumulateur. STA $C010 n'a d'autre but que de "libérer" l'Apple pour une autre touche. On stocke ensuite le caractère en CAR, on remet à l'écran le caractère précédemment remplacé par le curseur, et on quitte enfin la routine après avoir récupéré le caractère saisi au clavier dans l'accumulateur.

Voyons maintenant le reste du programme.

Ligne 6 : équivalente à HOME.

Ligne 7 : saisie d'un caractère par notre routine personnelle.

Lignes 8 et 9 : #$9B est le code de ESC. Si l'utilisateur vient donc d'appuyer sur ESC, on saute à la fin de la routine.

Lignes 10 à 13 : si c'est un RETURN (#$8D) ou une flèche à gauche (#$88), on affiche.

Lignes 14 et 15 : si ce n'est pas une flèche à droite (#$95), on saute en S5.

Lignes 16 à 18 : on relit le caractère qui se trouve sous le curseur

pour le ré-afficher (il peut paraître plus simple d'augmenter simplement H pour déplacer le curseur, mais il faudrait alors vérifier par nous-mêmes que l'on ne change pas de ligne, ce que la routine OUT = $FDF0 fera très bien à notre place...).

Lignes 19 et 20 : si l'accumulateur contient une valeur inférieure à #$A0, on a tapé un caractère de contrôle qui se trouve donc rejeté par la demande d'une autre saisie (BCC DEB).

Lignes 21 et 22 : si le code est supérieur ou égal à #$B0, BCS branche en S0.

Lignes 23 à 25 : c'est autre chose qu'un chiffre ou une lettre : on retire #$40 au code pour l'afficher en FLASH. Il faut absolument faire SEC avant SBC puisque l'on sait justement que C=0 (BCS branche si C=1 et il n'y a pas de branchement).

Lignes 26 à 30 : pour un code >= #$BA, on passe en S1. Sinon, c'est un chiffre et on retire #$80 pour un affichage en inverse (même remarque que ci-dessus pour SEC).

Lignes 31 à 36 : si c'est #$C0, on remonte à S4 afin d'obtenir un affichage clignotant (on pourrait aussi passer directement à S3 puisque le bit C est mis à 1 lorsque l'accumulateur est supérieur ou égal à la valeur à laquelle on le compare). Si c'est supérieur à #$C0 (l'égalité est déjà traitée), on passe à S2. Sinon, c'est encore un caractère à afficher en mode clignotant et on retire donc #$40.

Lignes 37 à 40 : si c'est supérieur ou égal à #$DB, c'est un caractère situé au-dessus des lettres de l'alphabet et on saute à S3 pour un affichage en clignotant (on est sûr que C=1 puisque BCS provoque le branchement). Dans le cas contraire, c'est une lettre et on l'affiche sans modification du code (mode normal).

Après manipulation éventuelle du code, vous aurez constaté que l'on arrive toujours à la ligne 39, pour affichage par la routine standard du moniteur puis retour en début de routine.

Pour tester le fonctionnement de ce programme après en avoir rentré le code en mémoire ou l'avoir chargé à partir d'une disquette, il suffit de taper CALL 768 à partir du BASIC.

Exemple 2

Ce programme peut être un utilitaire, assez rudimentaire il est vrai, destiné à vous indiquer, après l'exécution d'un programme en Applesoft, si telle ou telle variable a bien été utilisée, ou pour le moins initialisée, au cours de cette exécution. Il ne suffit pas en effet qu'existe une instruction A=2 pour que la variable A soit effectivement créée en mémoire, il faut en outre que cette instruction soit réellement exécutée dans le déroulement du programme. L'intérêt "utilitaire" de notre routine pourrait donc consister à détecter des séquences d'instructions par lesquelles on ne passe pas lors d'un RUN et qui risquent fort d'être inutiles. Ce n'est là cependant qu'un aspect secondaire, le but essentiel restant toujours d'illustrer le mode d'emploi des instructions de l'assembleur que nous connaissons maintenant.

Il nous faut d'abord rappeler brièvement la façon dont sont stockées les variables en mémoire. Il existe deux grandes catégories de variables :
- les variables simples, stockées dans une zone dont l'adresse de début nous est donnée aux adresses $69 et $6A, dans l'ordre poids faible/poids fort ($69-$6A "pointe" vers le début de la zone des variables simples) et l'adresse de fin aux adresses $6B-$6C;
- les tableaux, dont le pointeur de début de zone est stocké en $6B-$6C (la zone des tableaux commence donc juste après la zone des variables simples) et le pointeur de fin en $6D-$6E.

Dans les zones de stockage des variables, les noms de variables sont toujours codés sur deux octets selon les règles suivantes, qu'il s'agisse de tableaux ou de variables simples :
- code ASCII standard (celui qui est retourné par la fonction ASC) si la variable est réelle : par exemple, sachant que le code ASCII de A est 65 en décimal, soit $41, A serait codé 41 00 et AA serait codé 41 41.
- code ASCII pour la première lettre et code ASCII augmenté de $80 (soit 128) pour la seconde si la variable est de type "chaîne de caractères". par exemple 41 80 pour A$ et 41 C1 pour AA$.
- code ASCII augmenté de $80 pour les deux lettres si la variable est de type "entier" : C1 80 pour A% et C1 C1 pour AA%.

Une variable simple occupe toujours 7 octets. Pour un tableau, le nombre d'octets occupés est stocké en mémoire juste après le nom du tableau.

Le principe d'utilisation de la routine est le suivant :
- On place aux adresses $6 et $7 les deux octets correspondant au nom de la variable examinée, codés selon les règles en vigueur dans les zones des variables, puis on appelle la routine par un CALL 768.
- Celle-ci va rechercher dans la zone des variables simples, puis dans celle des tableaux, la présence éventuelle d'une variable stockée sous le même nom. Pour cette recherche, on utilisera l'adressage indirect indexé par Y, avec pour adresse de base les différents pointeurs de zone. Si on trouve dans la première zone, on mettra 1 à l'adresse $8, et 1 à l'adresse $9 si on trouve dans la seconde.
- Pour afficher le résultat de la recherche, on utilise une routine du moniteur, commençant à l'adresse $F940, qui affiche à l'écran le contenu des registres Y et X sous forme de 4 chiffres hexadécimaux. On voudrait donc voir 0000 si la variable n'existe pas du tout, 0100 si elle existe uniquement dans les variables simples, 0001 si elle n'existe que dans les tableaux, et 0101 si elle existe dans les deux catégories de variables.

Exemple 2 - Lisa 1.5

```
0300                1
0300                2  C1
0300                3  C2
0300                4  VS
0300                5  TB
0300                6  STOP
0300                7  TVS
0300                8  TTB
0300                9  P
0300 A900          10
0302 8508          11
0304 8509          12
0306 A569          13
0308 8518          14
030A A56A          15
030C 8519          16
030E A519          17  S00
0310 C56C          18
0312 D006          19
0314 A518          20
0316 C56B          21
0318 F02A          22
031A A000          23  S0
031C B118          24
031E C506          25
0320 D014          26
0322 C8            27
0323 B118          28
0325 C507          29
0327 D00D          30
0329 E608          31
032B A56B          32
032D 8518          33
032F A56C          34
0331 8519          35
0333 4C4403        36
0336 A518          37  S1
0338 18            38

ORG $300
EPZ $6
EPZ $7
EPZ $69
EPZ $6B
EPZ $6D
EPZ $8
EPZ $9
EPZ $18
LDA #0
STA TVS
STA TTB
LDA VS
STA P
LDA VS+1
STA P+1
LDA P+1
CMP TB+1
BNE S0
LDA P
CMP TB
BEQ S20
LDY #0
LDA (P),Y
CMP C1
BNE S1
INY
LDA (P),Y
CMP C2
BNE S1
INC TVS
LDA TB
STA P
LDA TB+1
STA P+1
JMP S20
LDA P
CLC
```

```
0339 6907      39
033B 8518      40
033D 9002      41
033F E619      42
0341 4C0E03    43     S4
0344 A519      44     S20
0346 C56E      45
0348 D006      46
034A A518      47
034C C56D      48
034E F011      49
0350 A000      50     S2
0352 B118      51
0354 C506      52
0356 D011      53
0358 C8        54
0359 B118      55
035B C507      56
035D D00A      57
035F E609      58
0361 A408      59     FIN
0363 A609      60
0365 2040F9    61
0368 60        62
0369 A002      63     S3
036B A518      64
036D 18        65
036E 7118      66
0370 48        67
```

```
ADC #$7
STA P
BCC S4
INC P+1
JMP S00
LDA P+1
CMP STOP+1
BNE S2
LDA P
CMP STOP
BEQ FIN
LDY #0
LDA (P),Y
CMP C1
BNE S3
INY
LDA (P),Y
CMP C2
BNE S3
INC TTB
LDY TVS
LDX TTB
JSR $F940
RTS
LDY #2
LDA P
CLC
ADC (P),Y
PHA
```

```
0371 C8        68     INY
0372 A519      69     LDA P+1
0374 7118      70     ADC (P),Y
0376 8519      71     STA P+1
0378 68        72     PLA
0379 8518      73     STA P
037B 4C4403    74     JMP S20
```

Récapitulation : exemple 2

```
0300- A9 00 85 08 85 09 A5 69
0308- 85 18 A5 6A 85 19 A5 19
0310- C5 6C D0 06 A5 18 C5 6B
0318- F0 2A A0 00 B1 18 C5 06
0320- D0 14 C8 B1 18 C5 07 D0
0328- 0D E6 08 A5 6B 85 18 A5
0330- 6C 85 19 4C 44 03 A5 18
0338- 18 69 07 85 18 90 02 E6
0340- 19 4C 0E 03 A5 19 C5 6E
0348- D0 06 A5 18 C5 6D F0 11
0350- A0 00 B1 18 C5 06 D0 11
0358- C8 B1 18 C5 07 D0 0A E6
0360- 09 A4 08 A6 09 20 40 F9
0368- 60 A0 02 A5 18 18 71 18
0370- 48 C8 A5 19 71 18 85 19
0378- 68 85 18 4C 44
```

Reprenons donc tout cela plus en détail.

Lignes 10 à 16 : mise à 0 de nos deux drapeaux "existe / n'existe pas" et transfert dans une variable de travail P (adresses $18-$19) du pointeur de début de la zone des variables simples. Le poids faible (VS) viendra en P ($18) et le poids fort (VS+1) en P+1 ($19).

Lignes 17 à 22 : on veut savoir si on est arrivé à la fin de la zone des variables simples. On regarde d'abord si l'octet de poids fort, le plus significatif, P+1, est égal au poids fort du pointeur de début de la zone des tableaux TB+1 : si ce n'est pas le cas, on ne s'intéresse pas au poids faible. Sinon, on effectue le même contrôle pour les poids faibles P et TB; s'il y a égalité, on saute en S20.

Lignes 23 à 26 : on charge dans l'accumulateur le code de la première lettre du nom stocké dans la zone des variables. P/P+1 doit en effet toujours contenir l'adresse de début des informations pour une variable donnée (la première stockée en l'occurrence si on commence en début de zone). En prenant Y=0, on accède par LDA(P),Y à la première information pour la variable en question, soit la première lettre de son nom, que l'on compare ensuite à la première lettre du nom cherché (C1). Si elles sont différentes, on passe en S1.

Lignes 27 à 30 : même principe pour la seconde lettre, en prenant cette fois Y=1.

Lignes 31 à 36 : on a trouvé ! TVS passe de 0 à 1, suite au INC, et l'on met en P/P+1 le pointeur de début de la zone des tableaux avant de sauter à S20.

Lignes 37 à 43 : on n'a pas trouvé ! Il faudrait donc aller voir ce qu'il en est pour la variable stockée suivante, s'il y en a encore. Pour ce faire, nous devons mettre à jour notre pointeur de début d'informations pour une variable, P/P+1. On retrouve là un exemple typique d'utilisation de ADC : on ajoute 7 au poids faible P (la variable que nous venons de traiter occupe 7 octets et la suivante éventuelle commence donc 7 octets plus loin), si cette addition ne produit pas de retenue extérieure (BCC) on laisse le poids fort inchangé, sinon il doit être incrémenté. On retourne ensuite au début de la recherche, en vérifiant tout d'abord que l'on n'est pas encore en fin de zone (on passe à la recherche dans la zone des tableaux si le test est positif).

Lignes 44 à 49 : on va maintenant chercher dans les tableaux. On regarde de suite si la zone des tableaux n'est pas terminée (P+1 = STOP+1 et P = STOP). Si elle l'est, on passe à FIN.

Lignes 50 à 57 : même principe de recherche du nom que précédemment. A partir du pointeur sur le début des informations, Y=0 nous donne accès à la première lettre du nom stocké et Y=1 nous donne accès à la seconde.

Lignes 58 à 62 : on a trouvé ! TTB passe de 0 à 1, puis on charge Y et X respectivement avec TVS et TTB pour obtenir l'affichage voulu par $F940 avant de quitter la routine.

Lignes 63 à 74 : on n'a pas trouvé ! Là encore, il faut mettre à jour le pointeur de début des informations pour la variable suivante, mais c'est un peu plus complexe que pour les variables simples, car le nombre d'octets à ajouter n'est pas une constante. Nous savons seulement qu'il se trouve derrière les deux lettres du nom, dans l'ordre poids faible/poids fort, et nous devons utiliser la même technique d'adressage pour le lire. Pour Y−2, on récupère le poids faible du nombre d'octets occupés par le tableau, que l'on peut ajouter à P par ADC. Il ne faut pas toutefois ranger de suite le résultat en P car on modifierait ainsi l'adresse-base de notre adressage et nous ne pourrions plus aller lire le poids fort du nombre d'octets, qui est récupérable avec la valeur 3 pour Y mais sous réserve que l'adresse-base soit toujours correcte. C'est pourquoi on empile provisoirement le résultat du calcul après ADC. On peut alors lire le poids fort du nombre d'octets et l'ajouter à P+1 par ADC (sans CLC préalable cette fois puisqu'il peut y avoir une retenue provenant de l'addition précédente à prendre en compte). Le résultat de l'addition des poids forts est remis en P+1, celui de l'addition des poids faibles est ensuite retiré du sommet de la pile et remis en P : le pointeur est à jour et l'on retourne au début de la procédure de recherche.

Pour utiliser cette routine, la méthode la plus simple nous semble être la suivante :

- Après la fin d'exécution du programme Applesoft, passer en moniteur par CALL −151.
- Faire un BLOAD du code-objet de la routine s'il ne se trouve pas déjà en mémoire.
- Mettre aux adresses $6-$7 le code du nom de la variable cherchée (en tapant par exemple 6: 41 C1 "Return" si vous vous intéressez à la variable AA$).
- Lancer la routine par 300G suivi de "Return".

■

LIST

le journal des amateurs de programmation

Si programmer un ordinateur est devenu pour vous un loisir, un plaisir... une passion, sachez que **LIST** a été créé pour vous. **LIST** vous aide à tirer davantage de votre matériel, à vous perfectionner dans la conception des programmes qui "tourneront" sur votre machine. **LIST** vous initie aux langages informatiques et sélectionne les meilleurs livres pour progresser. **LIST** vous informe de l'actualité et vous fournit trucs, astuces et idées pour mieux programmer...
Pour être sûr de ne rater aucun numéro et pour recevoir **LIST** chez vous, **abonnez-vous** !

FAITES 40 F D'ECONOMIE !

LIST, LE PLAISIR DE PROGRAMMER

20 F chez votre marchand de journaux

BULLETIN D'ABONNEMENT

à retourner à **LIST**
(service abonnement)
5, place du Colonel-Fabien,
75491 Paris Cedex 10

Nom : _____
Adresse : _____
Ville : _____
Code postal : |_|_|_|_|_| Pays : _____

Veuillez m'abonner pour 10 numéros au prix avantageux de **160 F*** au lieu de 200 F. Je fais ainsi une économie de 40 F sur le prix de vente au numéro.
Je joins mon règlement indispensable libellé à l'ordre de **LIST**.

* Belgique : **1330 FB** ; Suisse : **50 FS** ; Canada : **30 $C** ; autres pays : **210 FF**

Le BASICIUM

Gérard Michel

Pom's vous propose sous ce nom une disquette regroupant un certain nombre de routines et d'utilitaires destinés à faciliter la gestion d'écran et la saisie de données réalisées par vos programmes en Applesoft.

Le BASICIUM reprend sous une forme plus élaborée et plus intégrée la gestion de masques et l'INPUT généralisé de tableaux déjà présentés dans ces pages, complétés par des instructions de gestion de messages.
Il permet notamment une préparation indépendante des opérations de saisie-clavier, d'affichage et d'impression, et une réduction de la taille des programmes Applesoft au moyen d'instructions synthétiques.

Ainsi, le petit programme listé ci-après à titre d'illustration suffit pour réaliser les traitements suivants :

- Affichage de l'écran de présentation ECRAN 1.
- Saisie de trois groupes de valeurs dans un tableau intermédiaire Z$, à l'intérieur du cadre donné par ECRAN 2, avec tabulation, contrôle sur la longueur, vérification d'un type alphanumérique pour les valeurs de la première colonne et numérique pour celles de la seconde, déplacement d'une zone à l'autre dans les deux sens...
- Demande de confirmation après chaque écran de saisie, avec contrôle d'une réponse O ou N, et modification dans la négative.
- Consultation à l'écran des trois tableaux, avec possibilité de hard-copy sur demande dans la mesure où l'imprimante est déclarée en service.
- Restitution des données sur papier sous la forme de l'état TABLEAU.

Dans la liste du programme, les instructions du BASICIUM sont repérées par le caractère "]". Ce système est conçu pour un Apple //e ou un Apple II+ 48K. Il est compatible avec le DOS et toutes les instructions "standard" de l'Applesoft.

Les différents utilitaires, masques et routines ont également été adaptés pour une utilisation avec un Apple //e équipé de la carte 80 colonnes Apple //e. On peut ainsi disposer d'un BASICIUM "80 colonnes" doté des mêmes possibilités et instructions que la version 40 colonnes (utilisation de masques, de tableaux de variables, hard-copy, impressions paramétrées, gestion de messages...). Cette seconde version est disponible sur la même disquette.

Pom's vous propose l'ensemble du système au prix de 150 francs, documentation comprise.

■

Ecran 1

```
****  **** **** *** **** *** *  * *    *
*  *  *  * *     *  *  *  *  *  * **  **
*  *  *  * *     *  *     *  *  * * ** *
*  *  *  * *     *  *     *  *  * *    *
***** **** ****  *  *     *  *  * *    *
*   * *  *    *  *  *     *  *  * *    *
*   * *  *    *  *  *  *  *  *  * *    *
***** *  * **** *** **** *** **** *    *


     ****************************
     *==========================*
     *+========================+*
     *++                      ++*
     *++ EXEMPLE D'UTILISATION ++*
     *++                      ++*
     *+========================+*
     *==========================*
     ****************************

----------------------------------------
!   -:-:-:   POUR P O M ' S   :-:-:-   !
----------------------------------------
```

Ecran 2

```
''''''''''''''''''''''''''''''''''''''''
'          TABLEAU DE DONNEES          '
''''''''''''''''''''''''''''''''''''''''


----------------------------------------
! I !      VALEUR      " I !   VALEUR  !
!===!==================="===!===========!
! 0 !                  " 1 !           !
!---!------------------"---!-----------!
! 2 !                  " 3 !           !
!---!------------------"---!-----------!
! 4 !                  " 5 !           !
!---!------------------"---!-----------!
! 6 !                  " 7 !           !
!---!------------------"---!-----------!
! 8 !                  " 9 !           !
!---!------------------"---!-----------!
!10 !                  "11 !           !
!---!------------------"---!-----------!
!12 !                  "13 !           !
----------------------------------------
```

Programme de démonstration BASICIUM

```
10  TEXT : HOME
20  HIMEM: 34680
25 BL$ = "                      "
30 D$ =  CHR$ (4):D1$ =  CHR$ (13) + D$:
        PRINT D$"BLOAD POBAS-VAR": POKE
       37191, PEEK (43634): POKE 37192,
       PEEK (43635): PRINT D$"BLOAD POBA
       S-B": POKE 37254, PEEK (43634): P
       OKE 37255, PEEK (43635)
40  DIM Z$(13),Y$(5),ZZ$(3,13)
50  FOR I = 0 TO 2: READ ME$(I): NEXT :
       GOTO 70
60  DATA  SAISIE CONFIRMEE,'RETURN' OU '
       ?' POUR HARD-COPY,IMPRIMANTE BRAN
       CHEE
65  VTAB 23: PRINT D1$"PR#1": RETURN
70  :]A1: FOR Z = 1 TO 5000: NEXT :NS = 1
80  :]A2:]IZ: IF  PEEK (6) THEN 70
90  :]GME0: IF  PEEK (6) THEN 110
100  :]MZ: IF  PEEK (6) THEN 70
105  GOTO 90
110  FOR I = 0 TO 13:ZZ$(NS,I) = Z$(I):]
       F: NEXT :NS = NS + 1: IF NS < 4 T
```

```
      HEN 80
120  FOR NN = 1 TO 3: FOR I = 0 TO 13:Z$
      (I) = ZZ$(NN,I): NEXT :JPZ:JRME1:
       IF  NOT  PEEK (6) THEN 140
130 :JQME2: IF  PEEK (6) THEN  POKE 34,2
      4: GOSUB 65:JH: PRINT D1$"PR#0":
      POKE 34,1:JA2
140  NEXT :JQME2: IF  NOT  PEEK (6) THEN
       TEXT : HOME : END
150 :JA3: GOSUB 65:JCA:JCE:JCB:JCE:JCF:J
      CD:JCG:JCD:JCF:JCD: PRINT D$"PR#0
      ":JA3
160  FOR I = 0 TO 12 STEP 2: FOR J = 0 T
      O 4 STEP 2:JJ = J + (J < 2) - (J
      > 2):Y$(J) = ZZ$(JJ,I):Y$(J + 1)
      = ZZ$(JJ,I + 1):Y$(J) =  RIGHT$ (
      BL$ + Y$(J),13):Y$(J + 1) =  RIGH
      I$ (BL$ + Y$(J + 1),10):JF: NEXT
      :JPY
170  GOSUB 65:JCC: PRINT D$"PR#0":JA3: N
      EXT
180  GOSUB 65:JCD:JCA: PRINT D1$"PR#0"
```

Tableau

SAISIE NO 1		SAISIE NO 2		SAISIE NO 3	
VALEUR 1	VALEUR 2	VALEUR 1	VALEUR 2	VALEUR 1	VALEUR 2
VALEUR 0	000	VALEUR 0 - T2	000.22	VALEUR 0 - T3	000.33
VALEUR 1	111	VALEUR 1 - T2	111.22	VALEUR 1 - T3	111.33
VALEUR 2	222	VALEUR 2 - T2	222.22	VALEUR 2 - T3	222.33
VALEUR 3	333	VALEUR 3 - T2	333.22	VALEUR 3 - T3	333.33
VALEUR 4	444	VALEUR 4 - T2	444.22	VALEUR 4 - T3	444.33
VALEUR 5	555	VALEUR 5 - T2	555.22	VALEUR 5 - T3	555.33
VALEUR 6	666	VALEUR 6 - T2	666.22	VALEUR 6 - T3	666.33

Micro-informations
Jean-Michel Gourévitch

***NDLR** : tous les prix en francs indiqués dans cet article sont TTC, sauf spécification contraire. Chaque fois que les coordonnées d'un fournisseur sont connues, nous les indiquons en fin d'article. Inutile par conséquent de nous appeler pour les demander au téléphone. Merci.*

En vedette : les logiciels intégrés. C'est vrai, dans la galaxie micro-informatique, on ne jure plus que par les fenêtres et les programmes intégrés. A preuve, le succès de Lotus 1-2-3, un logiciel pas spécialement bon marché, installé en tête dans la liste des best-sellers. Seulement, jusqu'à aujourd'hui, pour travailler avec un de ces programmes intégrés, il fallait disposer de 16 bits, et donc d'un IBM PC/XT ou compatible. Réjouissons-nous, voici que l'Apple // ouvre lui aussi les fenêtres.

Des fenêtres pour l'Apple //

Avec tout d'abord **Appleworks**, nécessitant un Apple //c ou //e avec 128K, et disponible aussi pour l'Apple /// sous le nom de "3 E-Z Pieces"... Avec Appleworks, on n'est pas tout à fait dans l'inconnu. C'est la réunion de trois programmes en un : un traitement de texte, une base de données et un tableur. Le tableur ne dépaysera guère les utilisateurs de Visicalc, dont les formules de calcul sont fort voisines. La base de données a un air de déjà vu. Et pour cause, car il s'agit de Quick File //. Son auteur, Rupert Lissner, est aussi celui d'Appleworks. Seul le traitement de texte est inédit. Une de ses nouveautés est, quant à elle, plus gênante : les options d'impression sont formulées en inches et non plus en colonnes. Il y a des calculatrices qui vont chauffer...

Premier avantage de ce programme exceptionnel : sa facilité d'utilisation. Les commandes sont en effet les mêmes pour toutes les opérations importantes sur les trois applications. La pression simultanée des touches "Pomme ouverte" et "H" permet une recopie "hard" de l'écran sur imprimante. Et des tableaux explicatifs sont accessibles dans chaque application avec "Pomme ouverte" et "?". Plus besoin de documentation !

Deuxième avantage, le plus important, c'est l'intégration. On peut par exemple commencer une lettre avec le traitement de texte, aller chercher un tableau créé avec le tableur, revenir à la lettre et y insérer le tableau. On peut encore interrompre la rédaction d'un rapport, presser "Pomme Ouverte-Q", et aller chercher une note déposée antérieurement sur le "desktop" (le plan de travail) et, une fois la note consultée, revenir au rapport précisément là où on l'avait quitté. On peut même utiliser des tableaux créés avec Visicalc ou des textes écrits avec Applewriter (à condition de les avoir convertis en ProDOS à l'aide de la disquette utilitaire ProDOS, car Appleworks est écrit pour ce système d'exploitation).

Particulièrement remarquable, la présence de nombreuses sécurités qui permettent d'éviter d'écraser un dossier par inadvertance (comme je

viens de le faire avec le début de cet article en tapant en Applewriter Ctrl-S au lieu de Ctrl-L !). Ainsi, quand on choisit l'option de quitter le programme, Appleworks affiche "Do you really want this ?" (Le voulez-vous vraiment ?) et n'accepte la commande qu'après un "Yes" explicite. Appleworks est à coup sûr l'un des premiers de ces programmes qui rendront la vie plus facile à tous les utilisateurs de micros. Il est vendu aux Etats-Unis 395 dollars et sera traduit et vendu par Apple.

Sortie dans sa version française à Apple Expo, **Jane** transforme votre Apple //c ou //e en mini-Macintosh, avec avant tout le souci d'être facilement utilisable par tous, à l'aide entre autres du célèbre "couper et coller" qui fait le succès du Macintosh. Jane offre les programmes JaneWrite, JaneCalc et JaneList en un programme intégré pouvant fonctionner avec la souris Apple, la souris Arktronics, un joystick ou le Koalapad. Bien entendu, non seulement les informations peuvent passer d'une application à l'autre, mais les trois logiciels-en-un utilisent la même syntaxe autant que possible, ce qui réduit bien évidemment le temps d'apprentissage. Les importateurs de Jane ont en outre bon goût, ce qui ne gâche rien : leurs hôtesses étaient manifestement parmi les plus mignonnes à Apple Expo ... Jane, ainsi d'ailleurs qu'Appleworks, fera prochainement l'objet d'un banc d'essai approfondi dans Pom's.

Jane est fournie dans un dossier rigide, avec un manuel détaillé et quatre disquettes : Demo, Systems, Data et Help. Ces disquettes sont de couleurs différentes, ce qui facilite leur repérage. Initialement prévue pour fonctionner seulement avec la souris Arktronics, Jane utilise à présent les souris Apple //c et //e, et se vend donc séparément de ces outils. Ceci a permis de baisser son prix à moins de 1.500 FF.

Selon le même principe, ARTSCI propose aux Etats Unis **"The Magic Office System"**. Il s'agit d'une combinaison de Magic Window, MagiCalc et Magic Words, permettant de "couper et coller" (l'expression fait aujourd'hui fureur dans le langage des micros) des informations entre les programmes : un tableur, un traitement de textes et une base de données classiques.

Même chose ou presque avec **"4 in 1"**, qui utilise un langage commun et des procédures identiques pour un seul programme et quatre applications. Une production de Softsmith, vendue 129 dollars.

En France, BIP importe **"The Bridge"**, qui permet de faire le pont entre un fichier PFS, un traitement de texte (Applewriter ou Magic Window) et un tableur (Magicalc ou Visicalc). Prix : 506 F HT.

Et en Grande Bretagne, Dark Star a conçu une carte s'insérant dans un des slots de l'Apple //e, la **Snapshot Shuttle**, qui permet d'interrompre un programme par une pression sur un interrupteur, de faire tourner un autre programme, puis de revenir en actionnant à nouveau l'interrupteur au premier programme, au point exact où on l'avait laissé. L'utilisation du Snapshot exige au moins un lecteur et 128K de mémoire. Prix : 115 livres.

Plus classique

Il y a encore de la place pour les logiciels non intégrés. A preuve **Epistole,** un traitement de texte français, qui calcule (utile pour factures, devis, ...), dispose d'un mailing intégré et permet l'intégration de tableaux créés avec Visicalc, Multiplan ou Magicalc. La dernière édition de ce logiciel proposé par Version Soft est réalisée sous ProDOS. Prix : 2372 F.

Quant à **Homeword,** c'est un traitement de textes utilisant cette autre facilité de la micro-informatique : les icônes. Le bas de l'écran est illustré de dessins représentant les applications. Pour une utilisation domestique, Homeword est particulièrement agréable. Comme pour tous les logiciels non encore francisés, attention aux accents ! Homeword est conçu par Sierra On Line et vendu outre-Atlantique pour 70 dollars.

Pour les amateurs d'exotisme, il existe aussi un traitement de texte en chinois doté de 400 caractères d'un vocabulaire de base, avec possibilités d'extension. Réalisé par Dune Associates; prix : 70 dollars.

Pour les applications "sérieuses", on peut relier à l'Apple II un disque dur Intec 505 (10 mégas de capacité) à 2700 livres, ce qui n'est pas donné. Coûteux également, "the Genius" de Micro Display Systems, un moniteur permettant d'afficher une page entière de texte : 57 lignes de caractères de 7x12 pixels, disponible en noir et blanc, vert ou ambre. Livré avec une carte d'interface (à placer dans le slot 3), ce génie permet une utilisation plus complète de l'Apple au bureau. Prix : 1250 dollars.

Des imprimantes

Décidément, Apple ne délaisse guère le terrain de l'impression. A la fin de l'été sortira la **Scribe**. Conçue comme adjonction à l'Apple //c, la Scribe ne coûtera que 299 dollars, et imprimera en couleurs. Eh oui, en couleurs, grâce à un ruban et selon le principe du transfert thermique. La Scribe imprimera sur n'importe quel papier. Un prix très attractif, mais des rubans onéreux qui la rendent peu adaptée à une utilisation intensive.

Autre nouveauté : la **Think Jet** de Hewlett Packard. Portable, à jet d'encre, imprimant des caractères en 12 langues, avec en outre des possibilités graphiques. Et surtout peu bruyante (la tête d'impression ne touche jamais le papier). Prix de vente aux USA : 500 dollars. Disponible en France à la fin de l'année.

Vu à Apple Expo : des imprimantes **Canon** présentées par ASAP, dans différents registres. Les matricielles avec la PW 1080A (80 colonnes, 160 cps) à 5.600 F HT et la PW 1156A (156 colonnes, 160 cps) à 8.000 F HT. Une imprimante à jet d'encre couleur, modèle PJ 1080A, 80 colonnes et 37 cps, à 7.500 F HT.

A remarquer encore une interface série, donc utilisable pour l'Imagewriter d'Apple, permettant en appuyant sur une touche spéciale toute recopie d'écran; c'est la **Print It** de Text-Print, vendue 199 dollars.

Enfin, un logiciel particulièrement intéressant, conçu notamment pour l'Imagewriter, permet d'imprimer n'importe quelle image écran de l'Apple, en couleurs, et en zoomant sur des détails. C'est le **Printographer** de Roger Wagner Publishing Inc., à 40 dollars.

Un Apple à tout faire

Jadis, il fallait acheter une carte Z80 pour faire tourner des programmes CP/M et, parfois, on vous offrait le logiciel. Désormais, c'est Micropro, l'éditeur de Wordstar, qui offre aux Etats Unis une carte CP/M à tout acheteur du plus célèbre des traitements de texte ! Avec l'Apple, c'est vrai, on peut tout faire : connaître l'heure grâce à la carte Time Kit de Glanmire (un tout petit circuit qui s'enfiche sur la prise de jeux), vendue 1150 francs par Micro-Périph (compatible ProDOS et Pascal), ou gérer un carnet de rendez-vous avec Time Trax, un système de "gestion du temps" livré avec un module horloge. Une création de Creative Peripherals Unlimited vendue 100 dollars.

On peut aussi évaluer avec l'Apple sa vie de couple. "Friends or Lovers" de Softsmith Corp doit permettre aux couples "d'explorer leur relation". Les impétrants peuvent déterminer s'ils partagent les mêmes intérêts. Tout ça pour 30 dollars, c'est vraiment donné, moins cher même qu'un divorce au Mexique ...

Macintosh

Vu le nombre de plus en plus grand des lecteurs de Pom's utilisant ce produit sympathique, nous vous offrons dès ce numéro une rubrique Macintosh spécialisée, dont l'importance croîtra avec le temps.

Apple //c

La revue A+, dans son numéro de juin, publie une liste des logiciels tournant sur le nouvel Apple. Car, attention, la compatibilité n'est pas aussi totale qu'on le souhaiterait avec les anciens programmes des autres Apple II. Voir à ce titre l'article de Guy Lapautre.

Epistole (voir plus haut) est déjà adapté et tourne avec la souris du nouvel Apple. A noter également une extension pour le //c, le "cricket" de Street Electronics, qui lui ajoute une voix de femme, une voix de robot, des effets sonores, des possibilités stéréo et en prime une horloge ! Pas mal pour un ordinateur qu'on disait "fermé". L'imagination des fabricants de périphériques n'est décidément pas en panne.

La radio communication avec l'Apple

Transmettre des messages en radiotélégraphie, en recevoir, ou décoder les depêches radio transmises par les agences de presse mondiales, c'est possible avec l'Apple. Et avec aussi quelques efforts, car les interfaces et logiciels ne sont pas importés actuellement. Avec la diffusion de nouveaux Apple et notamment la sortie du //c, et la vogue de la communication, tout pourrait changer...

Les messages radio sont en effet transmis soit en phonie (on les reçoit en clair sur un récepteur), soit en morse, soit encore en télégraphie. Il y a déjà quelque temps que les radio amateurs ont remisé au grenier les manipulateurs pour se doter du clavier d'un micro-ordinateur, sur lequel ils pianotent leurs messages. Cette communication-là, régie par des lois strictes, est réservée aux passionnés titulaires d'une licence (ils peuvent cependant utiliser les interfaces et logiciels que nous allons décrire).

Mais les progrès des microprocesseurs permettent surtout de construire à des prix raisonnables (dans les 3000 F) des décodeurs qui affichent sur un moniteur (ou impriment à la demande) et en clair les messages reçus. Voir s'afficher sur l'écran une depêche venue de Tirana ou un message émis par un bateau en mer a indiscutablement un côté magique. Avec un Apple, c'est encore mieux, car on peut stocker sur disquette les messages reçus.

Pour recevoir des émissions lointaines, il faut d'abord une bonne antenne particulièrement dégagée. C'est un "must". Sans antenne, pas de réception.
Vient ensuite un récepteur de trafic multibandes. Citons notamment le FRG7700 de Yaesu, ou mieux, l'ICR 71 d'Icom. Prix : entre 5.000 et 10.000 francs.
Pour relier le récepteur à une carte série RS232C, il faut encore une interface. Vous avez le choix entre le CP1 d'Advanced Electronic Applications Inc (200 dollars), l'InterfaceII de Kantronics (270 dollars) ou encore le MFJ 1224 de MFJ enterprise (100 dollars).

Les messages décodés entrent dans l'Apple par l'extension série. Reste encore à gérer ces communications. C'est l'affaire des logiciels de Kantronics : Hamsoft, 30 dollars pour gérer l'écran et contrôler la vitesse des transmissions; Hamtext (100 dollars) sert à sauver les messages sur disquette, transmettre depuis une disquette, indiquer l'heure des transmissions, etc. Attention, ce passe-temps devient très vite obsédant. Bonjour les nuits blanches...

Les tableurs

Courant mai, Pom's a vu **Practicalc II** (Apple '84, Londres), un tableur (nous en reparlerons) offrant du traitement de texte, de la gestion de fichiers, de la consolidation, ... et capable d'utiliser directement un disque dur et des fichiers Basic.

Nous avons aussi reçu récemment, bien qu'il soit sorti depuis quelque temps déjà, **Visicalc Advanced Version** de Visicorp : il s'agit là de la version avancée de Visicalc antérieurement disponible sur l'Apple /// seulement et longtemps attendue sur le //e. Une des particularités de VAV est d'autoriser, comme l'a fait plus récemment Lotus 1-2-3, la pré-programmation de séquences de touches. En outre, les possibilités de formatage et les fonctions sont nettement plus nombreuses.

Flashcalc enfin, lui aussi produit de Visicorp, a fait son apparition à Apple Expo. Un des principaux objectifs de Flashcalc, qui a beaucoup de points communs avec Magicalc, est d'assurer une rapidité de calcul à toute épreuve. Nous n'avons pas encore pu tester celle-ci en vraie grandeur, n'ayant eu entre les mains qu'une version bridée à 6K.

Disquettes de démonstration

De nombreux lecteurs, soit de par leur éloignement géographique, soit parce que leurs revendeurs sont mal approvisionnés ou peu disponibles, ont des difficultés à se faire une idée des produits nouveaux. C'est pourquoi Pom's a décidé d'offrir à ses lecteurs la possibilité d'acquérir à bas prix (celui des disquettes de Pom's, le meilleur rapport performance/prix du marché) les disquettes de démonstration en sa possession, sous réserve bien entendu que le fabricant ou l'importateur du produit donne son accord explicite. Dans le cadre de ce nouveau service, nous offrons dès à présent, comme vous pouvez le voir sur le bulletin d'abonnement, les disquettes de démonstration de CX Système et de Jane. Les fabricants ou importateurs de logiciels qui seraient intéressés par ce moyen très rapide de faire mieux connaître leurs produits peuvent nous proposer leurs disquettes.

Adresses

Advanced Electronic Applications - PO Box C2160 - Lynnwood, WA98036 - USA.
Artsci - 5547 Satsuma Av. - North Hollywood, CA 91601 - USA.
ASAP - 3 avenue des Trois Peuples - 78180 Montigny le Bretonneux - Tél (3) 043.82.33.
BIP - 13 rue Duc - 75018 Paris - Tél 255.44.63.
Creative Peripherals Unlimited - 1606 S.Clementine Anaheim, CA 92802 - USA.
Dark Star System - 78 Robin Hood Way - Greenford Middlesex - UB67QW - GB.
Dune Associates - PO Box 1631 - Kailua - HI96734 - GB.
Intec - 41/45 Knights Hill Westnorwood - London - SE 270HS - GB.
Kantronics - 1202 E 23rd Street - Lawrence, KS 66044 - USA.
MFJ - Box 494 - Mississippi State, MS 39762 - USA.
Micro-Périph - 62 rue Ducouédic - 75014 Paris - Tél 321.53.16.
Roger Wagner - 10761 Woodside Av.SuiteE - PO Box 582 - CA 92071 - USA.
Softsmith Corporation-1431 Doolitle Drive - San Leandro, CA 94577 - USA.
Sierra On Line - Coarsegold, CA93614 - USA.
Softsmith Corp (pour Friends and Lovers) - 2935 Whipple Road - Union City, CA 94587 - USA.
Street Electronics - 1140 Mark Av. - Carpinteria, CA93013 - USA.
TextPrint - 8 Blanchard Rd - Burlington, MA 01803 - USA.
Practicorp Ltd. - Goddard Road - Whitehouse Industrial Estate - Ipswich - Suffolk IP1 5NP - GB.
Visicorp - 1 place Gustave Eiffel - Silic 241 - 94568 Rungis Cedex - Tél 687.61.01.

■

Les nouvelles du Macintosh

Hervé Thiriez

On peut résumer le développement du Macintosh à l'aide d'un petit exemple comparatif : il a fallu deux ans et demi pour vendre les 50.000 premiers Apple II Plus, sept mois pour les 50.000 premiers IBM PC, deux mois un quart pour les 50.000 premiers Macs ... Il y en avait partout à Apple Expo sur les stands, des piles de cartons d'emballage de Macs étaient présentées, mais très peu de stands pouvaient effectivement en proposer à la vente immédiate. Les petits futés (parmi les revendeurs) qui s'en sont vite aperçus n'ont pas attendu pour vendre leurs Macs plus cher que le prix public conseillé, ce qui est peu fréquent dans le monde de la micro-informatique ! Idem pour les disquettes 3,5 pouces : il ne restait plus que des Memorex dont (surprise !) le prix a augmenté dès que fut constatée la rupture de stock des concurrents (Sony et Hewlett Packard). La rançon de la gloire ...

Pour ma part, je reste tout à fait enthousiasmé par le Mac, mais considère qu'il faut absolument avoir un second lecteur de disquettes pour pouvoir travailler avec confort. Si l'on s'en prive, on est condamné à jongler avec les disquettes à la moindre lecture ou sauvegarde de fichier.

Le logiciel

Les programmes commencent à se multiplier. La liste des programmes déjà disponibles ou pratiquement disponibles (visibles mais non encore vendus) augmente sans cesse. Au catalogue Apple du 20 juin, on trouvait MacWrite/MacPaint à 1.586 FF, Multiplan U.S. à 2.206 FF et Basic Microsoft à 1.696 FF.

Déjà disponibles ou sur le point de l'être, nous avons remarqué à Apple Expo :

- Multiplan en version française;
- Basic Microsoft en version française;
- CX MacBase, qui n'a plus grand chose à voir avec CX Système;
- Omnis;
- PFS, avec File et Report;
- La Pierre Mobile, jeu écrit par Bruno Rives, que les lecteurs de Pom's ont déjà vu à l'oeuvre; Bruno est le responsable en France de la famille Apple des produits 68000;
- Eleugram, une disquette de jeux mensuelle dont le numéro 1 était proposé à Apple Expo, vendue par Compusoft, une jeune entreprise française malgré son nom à consonnance anglaise (en effet, ce nom sonne mieux que Calcumou ...);
- et cette liste n'est pas limitative.

Attendus aussi à court terme :

- Think Tank;
- Chart de Microsoft;
- le Pascal déjà célèbre du Macintosh;
- Lotus 1-2-3 (pas encore de confirmation officielle);
- dB Master.

Particulièrement remarquable mais encore invendu en France, le carnet d'adresses d'Habadex. A l'écran de Mac, un superbe graphisme représentant un carnet alphabétique; il suffit de cliquer les lettres pour l'ouvrir. A mi-chemin entre l'agenda et la base de données. Etonnant. La revue A+ de juin donne une liste détaillée des logiciels à paraître ou déjà parus pour le Mac. Et notamment, dès juillet, les jeux d'Infocom (Zork, Infidel, etc.). Ceux de Penguin (Transylvania, The Quest) sont déjà disponibles aux Etats-Unis.

Périphériques

Le catalogue Apple du 20 juin propose seulement le disque supplémentaire (4.622 FF) et le clavier numérique (963 FF).

Tout le monde pouvait voir au stand de Symbiotic à Apple Expo leur disque dur, tout récemment commercialisé pour le Macintosh. Il y aura bientôt du choix en la matière avec le Davong et les autres, qui ne tarderont pas à se mettre de la partie.

MacPaint

Il arrive avec MacPaint, alors qu'il semble y avoir au moins 20K disponibles sur la disquette, que l'on obtienne le message "Le disque est saturé, veuillez effacer des documents ou changer de disque". Il faut alors sortir, avec toutefois, heureusement, la possibilité d'effectuer préalablement une sauvegarde.

Pourquoi le disque est-il "saturé" quand il semble encore y avoir de la place ? La raison en est la suivante : de façon automatique, MacPaint vous protège en permanence contre une extinction intempestive "des lumières" en conservant automatiquement l'état de votre travail dans des fichiers baptisés Paint1 et Paint2. Ces fichiers disparaissent dès que vous terminez normalement une séance de travail avec MacPaint.

Si par contre il y a une interruption anormale, ces deux fichiers se trouvent sur la disquette, grâce à quoi votre prochaine séance débutera automatiquement par le chargement de Paint1 et Paint2, le fichier étant alors baptisé "Récupéré" à l'écran.

Il faut 20 à 30K pour loger à tout moment Paint1 et Paint2, ce qui explique par conséquent le fameux message indiquant la saturation. Vous pouvez d'ailleurs constater ce que nous venons de vous expliquer en éteignant votre Mac alors qu'un fichier MacPaint sans importance est chargé : vous verrez en rallumant l'appareil (attendez quelques minutes) que les fichiers Paint1 et Paint2 se trouvent effectivement sur la disquette. Lors de la mise en oeuvre de MacPaint, vous obtiendrez ensuite le chargement du fichier; il s'appelera "Récupéré", et les fichiers Paint1 et Paint2 disparaîtront du catalogue si vous quittez ensuite MacPaint de façon normale.

Recopie d'écran

Si vous avez bien lu la documentation, vous savez probablement que l'on peut à tout moment obtenir une recopie de l'écran sur imprimante avec Commande-Shift-4. Par contre, en faisant Commande-Shift-3, vous envoyez la recopie d'écran sur disquette, où elle devient un fichier MacPaint sous le nom ScreenX (X=0, 1, 2, ... automatiquement).

Ce fichier MacPaint est par conséquent (comme tout fichier MacPaint) modifiable à loisir et intégrable en totalité ou en partie dans un fichier de texte MacWrite.

Copie de disquettes

Apple vient de sortir une disquette de copie rapide, permettant de copier une disquette entière en quatre passes avec le seul lecteur intégré. Pour aller plus vite, ce programme utilise la mémoire écran du Mac, qui se remplit alors de caractères farfelus. Ne vous en faites pas, tout est normal ...

Utilisation de programmes américains

Si vous utilisez des programmes ramenés des USA, ou non encore francisés, par exemple le Basic US et le Multiplan US de Microsoft, vous pouvez les modifier de façon à pouvoir utiliser pleinement le clavier français

et les caractères accentués. La procédure à suivre est la suivante :

– effectuer préalablement une copie de la disquette à modifier (facultatif, mais on ne sait jamais);
– redémarrer avec la disquette "Disque Système" française livrée avec le matériel;
– éjecter cette disquette;
– introduire la disquette du logiciel à modifier;
– copier le fichier "System" de la disquette "Disque Système" sur la disquette du logiciel à modifier (quelques échanges de disquettes peuvent être nécessaires).

Bug sur la disquette MacWrite/Paint

Avec la première série des disquettes MacWrite/Paint, il était impossible d'initialiser correctement une disquette vierge à partir du bureau, ou de travailler correctement avec plusieurs disquettes. Cela tient aux informations contenues dans les pistes de boot (celles qui se lisent à l'allumage de l'appareil) de la disquette Mac Write/Paint.

Vous pouvez remédier au problème avec l'une des solutions indiquées ci-dessous :

- soit copier le logiciel Macwrite/Paint sur une autre disquette;
- soit démarrer à partir d'une autre disquette, l'éjecter, puis insérer la disquette Macwrite/Paint.

Bibliographie

Nous avons remarqué deux ouvrages, dont l'analyse paraîtra dans le numéro de septembre :

– Mac, produit par Microsoft, avec une couverture type graffiti rouge.
– Macintosh, Multiplan, MacPaint, de Eddie Adamis chez Cedic/Nathan, à 89 FF.

■

Courrier des lecteurs

Alexandre Duback

Si je prends la plume pour vous écrire, c'est tout d'abord pour vous féliciter pour votre revue. En fait, je ne l'aurais pas fait il y a six mois. En effet, je suis possesseur d'Apple II Plus depuis un an et, ayant entendu parler de Pom's, j'ai acheté, peu après l'Apple, les numéros 4, 5 et 6. N'étant qu'un novice, je me demandais ce que voulait dire le charabia des programmes et, parfois, des articles. Mais, ayant progressé, j'ai relu récemment ces numéros et y ai trouvé une foule de renseignements et des programmes consistants, n'ayant rien à voir avec ceux d'autres revues. Je voudrais donc savoir comment m'y abonner.

D'autre part, j'ai relu l'article sur Haifa dans le numéro 5. Je voudrais aussi savoir à quel prix je pourrai le recevoir.

Dominique Gilles - 91610 Balancourt/Essonne

Merci pour votre lettre qui nous rassure sur le bien-fondé de notre politique éditoriale de "nivellement par le haut". Je joins à cette lettre un bulletin d'abonnement à Pom's, sur lequel vous trouverez tous les renseignements nécessaires.

En ce qui concerne le programme Haifa, vous le trouverez sur la disquette d'accompagnement du numéro 5. La disquette Haifa Source, quant à elle, reprend simplement les programmes source en assembleur Lisa 2.5, qui n'avaient plus la place de se loger sur la disquette du numéro 5. Cette disquette intéresse donc les lecteurs qui souhaitent analyser et/ou modifier le programme Haifa. Les disquettes du numéro 5 et Haifa Source sont vendues 55 francs pièce.

Sur les 11 disquettes de Pom's que j'ai reçues, j'en ai 3 qui ne bootent plus. Pourriez-vous m'expliquer ce qui se passe et me dire s'il y a moyen de réparer autrement qu'en faisant INIT HELLO et en recopiant tous les fichiers par FID ?

Monsieur Jean-Michel Vélin - 50450 Gavray

Ce qui pourrait expliquer qu'il y ait parfois des problèmes au niveau du boot, c'est que le DOS se trouve sur le bord extérieur de la disquette. Si celle-ci a été quelque peu écrasée par les PTT, c'est en effet le bord qui souffre en premier. Il est possible d'utiliser des logiciels tels que Locksmith, Super Copy, Copy II+ (4.1 à 4.3) ... pour recopier seulement les pistes 0, 1 et 2 (celles du DOS), sans pour autant avoir à initialiser et à recopier fichier par fichier.

Carte Eve Chat Mauve

L'ablation du buffer dont je vous ai parlé dans ma lettre (Pom's 12) n'est hélas pas sans conséquence, comme je le pensais initialement. Je me suis aperçu par la suite que cela empêche Visicalc Advanced Version de fonctionner, ce logiciel ne reconnaissant alors plus l'extension mémoire 64K ni les routines 80 colonnes. Par contre, l'ancien Visicalc les reconnaît sans problème, de même que Multiplan. Allez savoir pourquoi ...

Jean-Pierre Januel - 75009 Paris

La rubrique reste donc ouverte ...

Par votre intermédiaire, je voudrais m'adresser aux possesseurs de //e qui envisagent d'acheter la carte Chat Mauve, afin de leur éviter de faire une bêtise. C'est vrai qu'on peut avoir la couleur, mais on peut aussi l'avoir avec un poste PAL-SECAM et un câble à 100 francs. Il est vrai qu'avec la carte Chat Mauve et le logiciel Purplesoft, on peut véritablement faire du graphisme HGR couleur. C'est remarquable d'efficacité. Malheureusement, dès que l'on fait un petit programme Basic de plus de 2.5K, il écrase les routines du Purplesoft et plus rien ne marche. Vous avez donc une machine avec 128K dont 2.5 réellement utilisables. On n'arrête pas le progrès ! En outre, la carte ne permet plus d'imprimer avec le Pascal et une imprimante Epson ou Microline (NDLR : avec notre Oki92, cela marche); on peut enlever le circuit intégré au-dessus du T de CHAT MAUVE, mais alors Apple Writer ne marche plus ...

A. Delacourte - 59155 Faches Thumesnil

Et voilà ! Votre lettre se joint au dossier.

***NDLR** - Aux dernières nouvelles, il s'avère que la carte Eve Chat Mauve n'a rien à voir avec les problèmes rencontrés avec l'Epson. La carte graphique Epson n'admet pas que d'autres cartes utilisent le bus de données en même temps qu'elle : le problème rencontré avec la carte Chat Mauve se présente aussi avec des cartes horloge, ou d'autres cartes. Ce n'est pas le cas avec la*

carte 80 colonnes étendue, car celle-ci n'utilise que le connecteur auxiliaire. Merci, M. Januel, pour cette dernière information.

J'ai lu dans un numéro de Pom's que, si tous les slots sont occupés, cela ne peut pas marcher. Or, tous mes slots seront occupés, à part le slot numéro 7, avec les cartes que j'envisage d'acheter. Qu'en pensez-vous ?

Monsieur J-C Decret - Bordeaux

On peut utiliser tous les slots simultanément, mais il se peut alors que, soit l'appareil ne "boote" plus, soit il le fasse mais connaisse des pannes aléatoires dues à la surchauffe. Tout cela tient plus au type des cartes qu'à leur nombre : certaines cartes "tirent" beaucoup et fonctionnent comme des radiateurs. Si l'appareil ne boote plus, on peut récupérer la situation, dans certains cas, en remplaçant des composants standards de l'Apple par des composants de même nature, mais plus puissants (voir pour cela votre revendeur). S'il y a surchauffe, on peut utiliser l'un des nombreux modèles de ventilateurs spécialisés. De toute façon, vous pouvez toujours emmener votre Apple chez votre revendeur pour voir s'il boote toujours après l'adjonction des nouvelles cartes que vous envisagez d'acquérir.

Dans le POKEs à gogo, de Roland Jost (Pom's 11), je note le POKE 214,255. Pour en sortir, l'auteur indique qu'il faut passer en langage machine et faire "D6:00". Mais le POKE inhibe les CALLs. Comment passer en moniteur sans le CALL −151 ?

S. Querret - 59760 Grande Synthe

Votre remarque au sujet du POKE 214,255 est tout à fait pertinente. L'humour marquant la suggestion de faire un CALL pour rétablir la situation, alors que les CALLs ne sont plus possibles après le POKE, est en effet involontaire.
Pour revenir en assembleur sans passer par CALL −151, il suffit d'utiliser ce qu'indique Jacques Duma dans sa lettre publiée dans le Courrier des Lecteurs du Pom's 6 (page 66). Utilisez un des numéros d'instruction entre 437760 et 440319, ce qui a pour effet de vous placer à un endroit imprévu (et variable) en mémoire, par exemple avec "440000 A=1". A moins que vous n'ayez la malchance de tomber sur une boucle, vous finirez pas arriver sur un BRK (code 00), ce qui vous "plantera" et vous mettra donc en mode moniteur.
Il ne restera plus alors qu'à faire le D6:00 qui annihilera l'effet du POKE, non sans toutefois faire disparaître le programme Basic qui aurait résidé préalablement en mémoire.

Il semble que l'écriture sur l'écran ne marche pas dans MATGRAPH, parce que les tables ne figurent pas dans la liste parue dans le Pom's 10. Ne pourriez-vous pas les publier dans un prochain numéro ?

Pierre Arnaud - 92260 Fontenay aux Roses

Cela n'a pas marché car il y a un problème avec la partie de ALPHAGR publiée dans le Pom's 10. Comme la plupart des lecteurs sont abonnés aux disquettes, ce problème n'avait pas encore été soulevé. Il y a en effet eu une troncature abusive et une partie du code a été oubliée, de 9272 à 928D.
A titre correctif, nous listons ci-dessous le code hexa de ALPHAGR en entier, y compris la partie où les chiffres et les lettres ont été définis à l'aide du programme de création de polices de caractères (Pom's 8).

En réponse au courrier de M. Januel (Pom's 12), voici la modification du programme MENU des disquettes Pom's qui permet l'impression sur une Epson avec interface Apple :

```
70 POKE 34,24:PRINT D1$"PR#"S
71 PRINT CHR$(0)
72 PRINT CHR$(27)"A"CHR$(10)
73 POKE 1657,80
75 CALL 37989
76 PRINT CHR$(27);CHR$(50)
77 PRINT D$"PR#0"
78 TEXT
```

Alain Meizoz

Je vous informe de la création du club informatique "Bellegarde Info" à Bellegarde sur Valserine, dans l'Ain près de Genève. Nous possédons un Apple //e en configuration de base, deux ZX81 et bientôt un ordinateur intermédiaire.

Pierre-Jean Giraud - 3 allée Paul Claudel - 01200 Bellegarde

Avec Apple Writer 2.0 (et non //e) et une Epson MX-82, comment faut-il faire pour obtenir les caractères spéciaux ? J'utilise aussi le Magicalc français qui, par ailleurs, ne donne aucune indication à ce sujet. Comment faut-il faire ? Enfin, comment incorporer des tableaux Magicalc en Apple Writer 2.0 ?

Docteur Docteur Séror - 17, rue Georges Messier - 64400 Oloron Sainte Marie

Pour utiliser des caractères spéciaux, je vous conseille "Donnez du caractère à votre imprimante", du Pom's 9, et "Les codes ASCII épluchés" du Pom's 4. Pour Magicalc, il faut mettre les caractères comme pour Multiplan, mais non précédés de l'accent circonflexe.
Magicalc n'est pas tout à fait compatible Visicalc, comme vous pourrez le voir avec le fichier que nous publions dans ce numéro (Lève-toi et brille).
Enfin, pour incorporer des tableaux Visicalc ou Magicalc dans un traitement de texte qui utilise des fichiers TEXT, il suffit de les sauvegarder sous forme de fichiers d'impression sur disquette (PRINT in SAVE format).

Amélioration de GES-COMPTE

Michel Herlaut

Je me permets d'ajouter une petite modification au programme de gestion de compte bancaire de Dominique Compère (Pom's 10). Le sous-programme que je vous propose ici (10000 à 10520) calcule le solde du relevé du compte après la vérification des opérations. En plus de ce sous-programme, il convient d'ajouter au programme les lignes suivantes :

```
2297  IF TEMOIN = 1 THEN  RETURN
3175  IF A1$ / "R" THEN  GOSUB 10000
      : GOTO 3040
8667  IF TEMOIN = 1 THEN 8700
8742  IF TEMOIN = 1 THEN  RETURN
9622  IF TT = 0 AND B = 0 THEN  GOTO
       9630
9625  IF TEMOIN = 1 AND  MID$ (TB$(B
      ),24,1) = " " THEN  GOTO 9640
9626  IF TEMOIN = 1 AND B = 0 AND TT
       <  > 0 THEN 9640
9645  IF TEMOIN = 1 THEN  RETURN
```

La commande "R" permet le branchement vers le sous-programme. Les données nécessaires sont le mois et l'année de départ (MM,AA). Le sous-programme effectue alors la sauvegarde du mois en cours de traitement si nécessaire, puis le calcul du solde en ne prenant en compte que les opérations déjà vérifiées, jusqu'au dernier mois créé.

Je me permets d'ajouter encore deux lignes indispensables pour la bonne marche du programme original :

```
9512 A2$ =  STR$ (A2)
11045  VTAB 1: HTAB 32: INVERSE : PR
       INT  MID$ (TB$(0),32,9);: NOR
       MAL
```

```
10000  REM
10010  REM  *** CALCUL RELEVE ***
10020  REM
10025 TEMOIN = 1
10030 L2$ = "MOIS,ANNEE DE DEPART :"
      :L3$ = "RELEVE": GOSUB 5500
10040  INPUT M9,A9
10050  IF M9 = MM THEN 10120
10060 MM$ =  STR$ (M9):AA$ =  STR$ (
      A9)
10065 MM = M9:AA = A9
10070  REM
10080  GOSUB 8620
10090  GOSUB 2200
10100 CL = NB: GOSUB 11010
10110  REM
10120 A2 = NB - 1: GOSUB 9600
10130  ONERR  GOTO 10300
10140  IF MM = 12 THEN MM = 0:AA = A
      A + 1
10150 MM = MM + 1:MM$ =  STR$ (MM):A
      A$ =  STR$ (AA)
10160 MOIS$ = TM$(MM): GOSUB 1400: G
      OSUB 1210
10170 DSN$ = LIBELLE$ + MOIS$ + " "
      +  STR$ (AA)
10175  PRINT
10180  PRINT D$"VERIFY"DSN$
10190  POKE 216,0: POKE 222,0
10200  GOSUB 8650: GOSUB 2200:CL = N
      B: GOSUB 11010:A2 = NB - 1: G
      OSUB 9610
10210  GOTO 10130
10300  REM
10310  POKE 216,0: POKE 222,0
10320 MM = MM - 1: IF MM = 0 THEN MM
       = 12:AA = AA - 1
10325 MM$ =  STR$ (MM):AA$ =  STR$ (
      AA)
10330  GOSUB 1400:MM$ =  STR$ (MM):A
      A$ =  STR$ (AA)
10340 MOIS$ = TM$(MM)
10350 DSN$ = LIBELLE$ + MOIS$ + " "
      +  STR$ (AA)
10370  REM
10400 L3$ = "RELEVE":L2$ = "RELEVE D
      E COMPTE = " +  STR$ (TT): GO
      SUB 5500
10405  CALL  - 868
10450 TEMOIN = 0
10500  GET W$
10510  GOTO 3010
10520  REM
```

Le fichier ALPHAHGR de MATGRAPH était incomplet dans la liste du numéro 10. Voici à ce titre correctif la liste complète de ALPHAHGR.

```
*9088.95FF

9088- 00 00 00 00 00 00 00 04
9090- 04 04 04 00 00 04 0A 0A
9098- 0A 00 00 00 00 14 14 1F
90A0- 0A 1F 05 05 1F 05 05 1F
90A8- 14 14 1F 13 13 08 04 02
90B0- 19 19 1C 14 1C 06 15 09
90B8- 15 08 04 02 00 00 00 00
90C0- 08 04 02 02 02 04 08 02
90C8- 04 08 08 08 04 02 00 11
90D0- 0B 1F 0B 11 00 00 04 04
90D8- 0B 04 04 00 00 00 00 00
90E0- 08 04 02 00 00 00 1F 00
90E8- 00 00 00 00 00 00 00 04
90F0- 04 10 10 08 04 02 01 01
90F8- 1F 11 19 15 13 11 1F 08
9100- 0C 0A 08 08 08 08 1F 10
9108- 10 1F 01 01 1F 1F 10 10
9110- 1F 10 10 1F 09 09 09 09
9118- 1F 08 08 1F 01 01 1F 10
9120- 10 1F 01 01 01 1F 11 11
9128- 1F 1F 11 10 08 04 02 01
9130- 1F 11 11 1F 11 11 1F 1F
9138- 11 11 1F 10 10 10 00 00
9140- 04 00 00 04 00 00 00 04
9148- 00 04 04 02 08 04 02 01
9150- 02 04 08 00 00 1F 00 00
9158- 1F 00 02 04 08 10 08 04
9160- 02 1F 11 1C 04 04 00 04
9168- 0E 11 15 13 0D 01 1E 0E
9170- 11 11 11 1F 11 11 0F 11
9178- 11 0F 11 11 0F 1E 01 01
9180- 01 01 01 1E 0F 11 11 11
9188- 11 11 0F 1F 01 01 0F 01
9190- 01 1F 1E 01 01 0F 01 01
9198- 01 1E 01 01 0D 11 11 0E
91A0- 11 11 11 1F 11 11 11 04
91A8- 04 04 04 04 04 04 10 10
91B0- 10 10 10 11 0E 11 89 05
91B8- 03 05 09 11 01 01 01 01
91C0- 01 01 1F 11 1B 15 11 11
91C8- 11 11 11 11 13 15 19 11
91D0- 11 0E 11 11 11 11 11 0E
91D8- 0F 11 11 0F 01 01 01 0E
91E0- 11 11 11 19 15 0E 0F 11
91E8- 11 0F 05 09 11 1E 01 01
91F0- 0E 10 10 0F 1F 04 04 04
91F8- 04 04 04 11 11 11 11 11
9200- 11 0E 11 11 11 11 11 0A
9208- 04 15 15 15 15 15 15 0A
9210- 11 11 0A 04 0A 11 11 11
9218- 11 0A 04 02 01 01 1F 10
9220- 08 04 02 01 1F 0E 02 02
9228- 02 02 02 0E 01 01 02 04
9230- 08 10 10 0E 08 08 08 08
9238- 08 0E 00 00 04 0A 11 00
9240- 00 00 00 00 00 00 00 1F
9248- FF FF 00 00 FF FF 00 00
9250- FF FF 00 00 FF FF 00 00
9258- FF FF 00 00 FF FF 00 00
9260- FF FF 00 00 FF FF 00 00
9268- FF FF 00 00 FF FF 00 00
9270- FF FF 01 0A 09 88 90 21
9278- 01 21 11 32 A9 00 A8 B1
9280- 85 8D 72 92 C8 B1 85 85
9288- EE C8 B1 85 85 EF AD 73
9290- 92 4A 4A 4A 8D 7A 92 0A
9298- 0A 0A 85 ED AD 73 92 38
92A0- E5 ED 85 EC A9 20 8D 79
92A8- 92 A5 ED 0A 0A 18 65 ED
92B0- 8D 78 92 A5 EC 4A 85 EC
92B8- AD 78 92 90 02 69 7F 8D
92C0- 78 92 18 A5 EC 6D 79 92
92C8- 8D 79 92 18 AD 74 92 6D
92D0- 78 92 8D 7B 92 A9 00 6D
92D8- 79 92 8D 77 92 18 A9 00
92E0- 8D 78 92 A2 00 A9 00 85
92E8- FA AD 78 92 85 EC AD 77
92F0- 92 85 ED AC 78 92 B9 A8
92F8- 93 38 E9 20 8D 7A 92 85
9300- F9 06 F9 26 FA 06 F9 26
9308- FA 06 F9 26 FA 38 A5 F9
9310- ED 7A 92 85 F9 A5 FA E9
9318- 00 85 FA 18 AD 75 92 65
9320- F9 85 F9 AD 76 92 65 FA
9328- 85 FA A9 00 A8 B1 F9 8D
9330- 7A 92 A9 00 A8 AD 7A 92
9338- 91 EC E8 A9 04 18 65 ED
9340- 85 ED A5 F9 69 01 85 F9
9348- A5 FA 69 00 85 FA BA C9
9350- 07 D0 D7 18 AD 7B 92 69
9358- 01 8D 7B 92 AD 77 92 69
9360- 00 8D 77 92 EE 78 92 AD
9368- 78 92 CD 72 92 F0 03 4C
9370- E3 92 60 00 00 2D 2D 2D
9378- 2D 2D 2D 2D FF FF 00 00
9380- FF FF 00 00 FF FF 00 00
9388- FF FF 00 00 FF FF 00 00
9390- FF FF 00 00 FF FF 00 00
9398- FF FF 00 00 FF FF 00 00
93A0- FF FF 00 00 FF FF 00 00
93A8- 52 46 52 45 FF FF 00 00
93B0- FF FF 00 00 FF FF 00 00
93B8- FF FF 00 00 FF FF 00 00
93C0- FF FF 00 00 FF FF 00 00
93C8- FF FF 00 00 FF FF 00 00
93D0- FF FF 00 00 FF FF 00 00
93D8- FF FF 00 00 FF FF 00 00
93E0- FF FF 00 00 FF FF 00 00
93E8- FF FF 00 00 FF FF 00 00
93F0- FF FF 00 00 FF FF 00 00
93F8- FF FF 00 00 FF FF 00 00
9400- FF FF 00 00 FF FF 00 00
9408- 7B FF 00 00 FF FF 00 00
9410- FF FF 00 00 FF FF 00 00
9418- FF FF 00 00 FF FF 00 00
9420- FF FF 00 00 FF FF 00 00
9428- FF FF 00 00 FF FF 00 00
9430- FF FF 00 00 FF FF 00 00
9438- FF FF 00 00 FF FF 00 00
9440- FF FF 00 00 FF FF 00 00
9448- FB FF 00 00 FF FF 00 00
9450- FF FF 00 00 FF FF 00 00
9458- 7F FF 00 00 FF FF 00 00
9460- FF FF 00 00 FF FF 00 00
```

```
9468- FF FF 00 00 FF FF 00 00
9470- 01 00 04 00 1B 24 2D 2D
9478- 36 36 3F 3F 24 00 00 00
9480- FF FF FF FF FF FF FF FF
9488- FF FF FF FF FF FF FF FF
9490- FF FF FF FF FF FF FF FF
9498- FF FF FF FF FF FF FF FF
94A0- FF FF FF FF FF FF FF FF
94A8- FF FF FF FF FF FF FF FF
94B0- FF FF FF FF FF FF FF FF
94B8- FF FF FF FF FF FF FF FF
94C0- FF FF FF FF FF FF FF FF
94C8- FF FF FF FF FF FF FF FF
94D0- FF FF FF FF FF FF FF FF
94D8- FF FF FF FF FF FF FF FF
94E0- FF FF FF FF FF FF FF FF
94E8- FF FF FF FF FF FF FF FF
94F0- FF FF FF FF FF FF FF FF
94F8- FF FF FF FF FF FF FF FF
9500- 00 00 00 00 00 00 00 00
9508- 00 00 00 00 00 00 00 00
9510- 00 00 00 00 00 00 00 00
9518- 00 00 00 00 00 00 00 00
9520- 00 00 00 00 00 00 00 00
9528- 00 00 00 00 00 00 00 00
9530- 00 00 00 00 00 00 00 00
9538- 00 00 00 00 00 00 00 00
9540- 00 00 00 00 00 00 00 00
9548- 00 00 00 00 00 00 00 00
9550- 00 00 00 00 00 00 00 00
9558- 00 00 00 00 00 00 00 00
9560- 00 00 00 00 00 00 00 00
9568- 00 00 00 00 00 00 00 00
9570- 00 00 00 00 00 00 00 00
9578- 00 00 00 00 00 00 00 00
9580- FF FF FF FF FF FF FF FF
9588- FF FF FF FF FF FF FF FF
9590- FF FF FF FF FF FF FF FF
9598- FF FF FF FF FF FF FF FF
95A0- FF FF FF FF FF FF FF FF
95A8- FF FF FF FF FF FF FF FF
95B0- FF FF FF FF FF FF FF FF
95B8- FF FF FF FF FF FF FF FF
95C0- FF FF FF FF FF FF FF FF
95C8- FF FF FF FF FF FF 20 20
95D0- 20 20 20 20 20 20 20 20
95D8- 20 20 20 20 20 20 20 20
95E0- 20 27 49 4E 46 4F 2E 43
95E8- 41 44 52 45 53 27 20 20
95F0- 20 20 27 49 4E 46 4F 2E
95F8- 43 41 44 52 45 53 27 04
```

Lecteur assidu de votre revue, j'aimerais savoir si quelqu'un possède les notices d'emploi en français de Magic Window et Locksmith 5.0.

Philippe Le Guen 49, rue Paul Bert - 54520 Laxou

Comme vous le comprendrez certainement, il n'est pas possible à Pom's de vous transmettre des photocopies de docs de programmes commerciaux. Nous nous mettrions dans notre tort en recourant à de telles pratiques, et notre souhait de rendre service aux lecteurs ne peut aller jusque là. A notre connaissance, la notice en français de Magic Window existe (faite par Ordinateur Express - voir Pom's 11), mais pas celle de Locksmith 5.0.

Heureux possesseur d'un Apple //e et d'une imprimante Facit 4510, je suis un de vos fidèles lecteurs et vous appelle au secours. Dès que je veux faire de la recopie d'écran, je suis perdu.

Dans Pom's 9, vous avez eu l'excellente idée de publier le programme GrafText, qui marche très bien pour la Facit 4510 avec seulement quelques modifications. Je vous serais reconnaissant si vous pouviez, dans un prochain numéro, proposer un programme de recopie d'écran HGR pour mon imprimante.

Paul Genesio 80 rue de Rome - 13006 Marseille

Malheureusement, nous ne pouvons pas vous recommander ici de programme de copie d'écran HGR pour la Facit. Il n'existe pas de programme général de recopie d'écran HGR, et chaque programme de ce type demande un certain temps de mise au point et, surtout, que l'on ait l'imprimante (ou une bonne doc) sous la main. Nous ne connaissons personne ayant un programme adéquat pour votre imprimante.

Tout ce que je peux faire pour vous, c'est de publier votre lettre. Ainsi, si un lecteur possède déjà un tel programme pour la Facit, il pourra vous le faire savoir. Dont acte !

Sur les conseils d'un technicien d'Apple Seedrin, nous vous soumettons un problème concernant Apple Writer : il semble d'après cette personne que votre revue ait fait paraître un article répondant à notre question. Pour pouvoir employer l'imprimante qui équipe notre système, nous devons envoyer le code ASCII nul (0), ce qui à l'origine n'est pas possible avec Apple Writer. Comment faire ?

L. LECA - Chatelier Conseil - 75015 Paris

Nous avons publié dans le numéro 4 de Pom's (ou Recueil No 1) l'article "Les codes ASCII épluchés", qui permet de voir quels sont les caractères à envoyer pour obtenir tel ou tel code ASCII. L'article "Donnez du caractère à votre imprimante" du Pom's 9 montre comment vous pouvez utiliser des codes imprimante à partir d'Apple Writer. Nous avons aussi dans le numéro 12 un article plein de trucs pour Apple Writer.

Malheureusement, le code 0 est un code bien spécial indiquant la fin d'un fichier. Si on commence un fichier de TEXT par ce code, ce texte sera vide, et ce pas seulement avec Apple Writer... Dilemne ! La solution vous est offerte dans le Pom's 10, au Courrier des Lecteurs (page 71) : il faut modifier le programme Apple Writer//e.

Message aux lecteurs ...

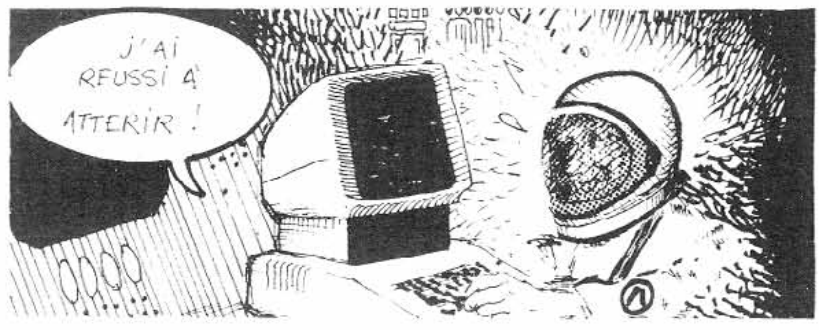
Nous recevons de plus en plus de lettres nous demandant comment il faut faire avec la configuration W, l'imprimante X, l'interface Y et le logiciel Z pour obtenir le résultat R ! Il y a tellement de combinaisons possibles que nous ne pouvons pas faire des recherches à chaque fois. Nous répondons donc à ces lettres quand nous pouvons y arriver rapidement, mais pas autrement : à vrai dire, c'est d'ailleurs le travail des fabricants de matériel et de logiciel. En outre, nous donnons la priorité aux lettres nous parvenant accompagnées d'une enveloppe timbrée pour la réponse.

Certains lecteurs nous ont posé des questions, suite à certaines informations qu'ils ont pu lire dans une autre revue francophone sur Apple, ou à certains bruits. Nous leur précisons, pour répondre à ces questions, que :

– le //c est incompatible avec le Duodisk comme avec les lecteurs du //e ou du II Plus;

– le réseau Apple-Bus fonctionnera initialement sur 68000 seulement pour tous types de fichiers, et par la suite (probablement) pour les fichers TEXT avec les familles 6502 et 68000;

– à court terme, seul le Lisa peut servir d'outil de développement pour le Macintosh; ultérieurement, ce sera aussi possible de développer avec deux "Macs" reliés ensemble. ■



LE MODEM QUI REND L'APPLE MINITELLIGENT

Apple-tell comprend :

• Une carte Modem incluant un décodeur Teletel.

• Un logiciel d'Emulation de Terminal Minitel enrichi de trois éblouissantes fonctions (celles qui faisaient le plus défaut jusqu'à présent sur votre Minitel).

IMPRESSION : l'imprimante de votre Apple est exploitée pour sortir les copies papier dont vous avez besoin lorsque vous consultez un serveur.

STOCKAGE : les disquettes de votre Apple sont utilisées pour enregistrer les pages dont la consultation vous est nécessaire :

• au format Teletel (c'est-à-dire telles que vous les avez reçues).

• en mode Texte pur (ASCII) pour exploitation locale ultérieure.

AUTOMATISME : l'intelligence de votre Apple est mobilisée pour accomplir l'interrogation automatique du serveur que vous lui avez désigné (appel téléphonique, orientation TRANSPAC, identification, choix successifs), enregistrer sur papier et / ou sur disque les données consultées, puis pour traiter celles-ci, en les incorporant dans votre application.
(Les procédures d'interrogation sont créées par l'utilisateur, sans aucun language de programmation, grâce au mode d'apprentissage Apple Tell.)

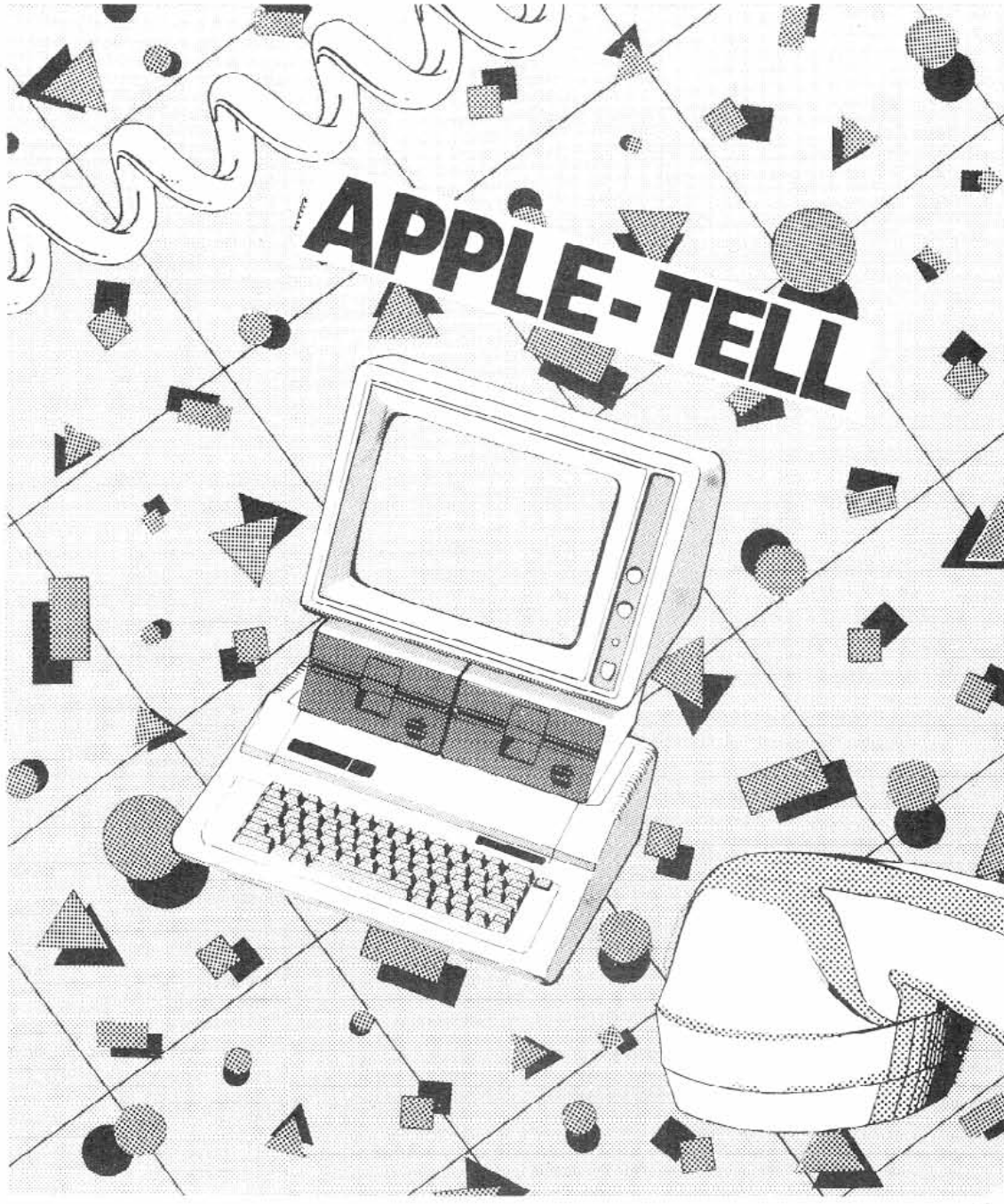
Evénement du dernier SICOB, salué par toute la presse, consacré Pomme d'Or 1983 par le jury Apple, le modem Apple-Tell marque une mutation décisive dans l'évolution des techniques vidéotex en environnement professionnel.

• point d'arrêt à la prolifération des matériels sur votre bureau (effet "mini-Sicob").

• Utilisation optimale des ressources dont vous disposez déjà (disques, imprimante, logiciels, etc...).

• Utilisation possible en mode Terminal autant qu'en mode Serveur (jusqu'à quatre portes).

• Enfin (et c'est sans doute le point le plus important) JONCTION entre le monde extérieur et les outils standards de votre Apple : l'incorporation des données dans Apple-writer, Visicalc, Multiplan, PFS, Quick File, etc..., et même dans vos applications personnelles (comptabilité, suivi de commandes, fichier...) devient possible.



GARRIC

CARACTERISTIQUES GENERALES :

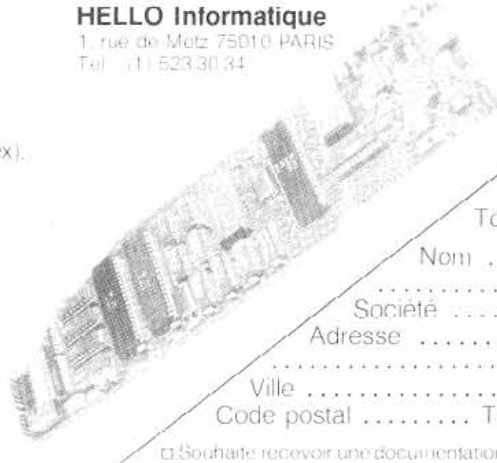
• Modem multimodes :
— 1 200 / 75 (full-duplex), 1 200 (half-duplex),
— 600 (half-duplex), 300 (full-duplex),
— standards CCITT et BELL (cette caractéristique unique rend accessibles les serveurs nord-américains, y compris par réseau téléphonique commuté).
Sorties vidéo composite (N & B) et Péritel couleurs.

• compatible Apple 2, Apple 2+, Apple 2e (48 K, une disquette).

• Enfichable dans n'importe quel slot libre de votre Apple 2.

• Transparence totale vis-à-vis du système.

HELLO Informatique
1, rue de Metz 75010 PARIS
Tél. : (1) 523.30.34



HELLO
Informatique
1, rue de Metz
75010 PARIS
Tél. : (1) 523.30.34
Telex FLASH 210500 F

Nom .....
.....
Société .....
Adresse .....
.....
Ville .....
Code postal ..... Tél. .....

☐ Souhaite recevoir une documentation sur le système Apple-Tell

☐ Commande ☐ système(s) Apple-Tell au prix de F 6 997 TTC (règlement ci-joint par : ☐ chèque bancaire ☐ CCP)

POM'S

Bibliographie

Le FORTH en douceur, de Michel Henric-Coll, Editions Eyrolles - 143 pages - 90 FF.

Cet ouvrage a été conçu comme une étude progressive du langage Forth pour débutants en informatique. Il a été présumé que le lecteur ne possédait pas encore de connaissance dans ce domaine et n'avait pas non plus appris un autre langage de programmation. C'est pourquoi il n'est pas structuré en chapitres homogènes et exhaustifs, que l'on pourrait directement consulter à la manière d'un dictionnaire. Ceci dit, un glossaire fourni en fin d'ouvrage se révèle très utile durant la lecture et plus tard, lors de l'utilisation du langage. A chaque chapitre est associé un ensemble d'exercices dont les corrigés se trouvent en annexe.

Jean-Luc Boyer

Débutez en FORTH, de Leo Brodie, Editions Eyrolles - 305 pages - 130 FF.

Ce livre est la traduction française de l'ouvrage "Starting FORTH", préfacé par l'auteur du langage, H.Moore. En premier lieu, cet ouvrage s'adresse à des débutants en informatique qui désirent s'initier au langage FORTH. Mais malgré cela, les informaticiens de métier y trouveront des informations très intéressantes dont l'accès est facilité par une table des matières complète et un résumé des mots Forth accompagnés d'une indication de la page où ils sont présentés. A la description du langage et de ses possibilités sont associés des dessins qui en facilitent la compréhension. A la fin de chaque chapitre, un résumé permet d'en mémoriser le contenu, et une liste de problèmes est fournie. En conclusion, cet ouvrage original par sa présentation est très complet, et devrait donc satisfaire l'ensemble de ses lecteurs.

Jean-Luc Boyer

Pangraphe par Jean-Pierre Petit (dessins en trois dimensions en Basic pour Apple II - Editions du P.S.I.).

Ouvrage curieux, mêlant le plus grand sérieux mathématique et la folie débridée de la bande dessinée d'anticipation.
Une culture mathématique de base est à l'évidence nécessaire pour se diriger sans trop de difficultés dans le labyrinthe des lignes trigonométriques. Une connaissance solide du Basic est également nécessaire pour suivre la "traduction" en programme de nombre d'options. Mais un grain d'humour est tout aussi indispensable pour ne pas s'irriter des facéties de BERNIE ou d'ARTHUR (de charmants robots)...

J'ai aimé !

Avec un air de ne pas y toucher, l'auteur emmène son lecteur assez loin, aussi bien dans les lois de la vision binoculaire que dans les éléments de géométrie descriptive permettant de comprendre en profondeur la formation et l'interprétation des images.

Cela étant, il ne faut pas croire qu'on va pouvoir créer avec Pangraphe un système de dessin révolutionnaire, ultra-rapide et ultra-précis. Pangraphe est essentiellement un outil didactique, même si des utilisations réelles peuvent s'appuyer sur les programmes. Mais l'écriture de quelques modules en assembleur ne serait pas un perfectionnisme douteux si on voulait vraiment en faire un outil opérationnel.

Un regret : le millier d'instructions (au moins) que représente Pangraphe peut bien évidemment être tapé au clavier, mais au prix de combien d'heures et de combien d'erreurs ? J'en ai tapé un certain nombre pour faire des tests et j'ai eu des problèmes avec l'un des tracés. Est-ce une erreur de programmation, de typographie ou de frappe ? Au bout d'une demi-heure, j'ai abandonné... Pourquoi diable, puisque c'est l'esprit de la collection "Utilisations de l'ordinateur", ne pas avoir accompagné ce livre d'une disquette ?

Une petite pointe bête et méchante, pour punir l'auteur de m'avoir fait tellement travailler sur mon clavier : le LOMEM, sur l'Apple, n'a jamais permis de loger le programme au-dessus de la valeur fixée, mais uniquement les variables. Soit dit tout à fait entre nous, si le programme ne tenait pas dans les 14K disponibles sous la page 2, comment pourrait-il tenir entre le haut de cette page et le DOS (nettement moins de 14K) !

Guy Mathieu

Le système MemDos par Pierre Clerc - EdiTests - 102 pages - 90 F.

Après un rapide historique du produit et de ses créateurs, cet ouvrage présente les différentes instructions du MemDos, tant en ce qui concerne la gestion des masques d'entrée/sortie que celle des fichiers.
On regrettera un certain manque de rigueur au niveau de l'introduction des concepts dans la partie descriptive "littéraire". C'est ainsi que le programme illustrant la restitution des informations à l'écran s'articule sur la lecture d'un fichier, alors que les instructions de lecture des fichiers ne sont traitées que dans les pages suivantes. De même, on voit apparaître des tests du type "IF WS > 0..." sans que le rôle particulier de cette variable WS dans le système MemDos ait été préalablement expliqué clairement.

Par ailleurs, il n'est pas fait explicitement mention de certaines précautions d'emploi, comme la nécessité de remettre les variables d'un masque à zéro avant une opération de saisie (sinon les anciennes valeurs sont conservées par défaut alors que l'utilisateur croit avoir saisi une donnée "vide" par un RETURN à vide), ou les risques que comporte une instruction LET"#R..." sur un fichier à accès par clé après une instruction LET"X..." si le fichier n'a pas été refermé puis ré-ouvert avant sa réorganisation.

A porter au crédit de cet ouvrage une présentation par ordre alphabétique des instructions, sous forme de fiches techniques reprenant la syntaxe et le champ d'application de chacune d'elles.
Mais le programmeur désirant s'initier à la pratique du MemDos tirera surtout profit de l'exemple d'application proposé. Il s'agit en effet d'un programme développé en détail, de la conception des masques de saisie jusqu'à l'écriture complète des différents modules de traitement, en passant par la définition des structures des fichiers nécessaires. On y trouve finalement une présentation condensée et concrète des possibilités du système.

En conclusion, ce livre peut apporter une aide complémentaire à la lecture de la documentation livrée avec le système (et toujours aussi touffue...), mais il aurait gagné, du point de vue de l'utilisateur débutant, à une démarche plus progressive et mieux structurée dans ses développements.

Alexandre Duback

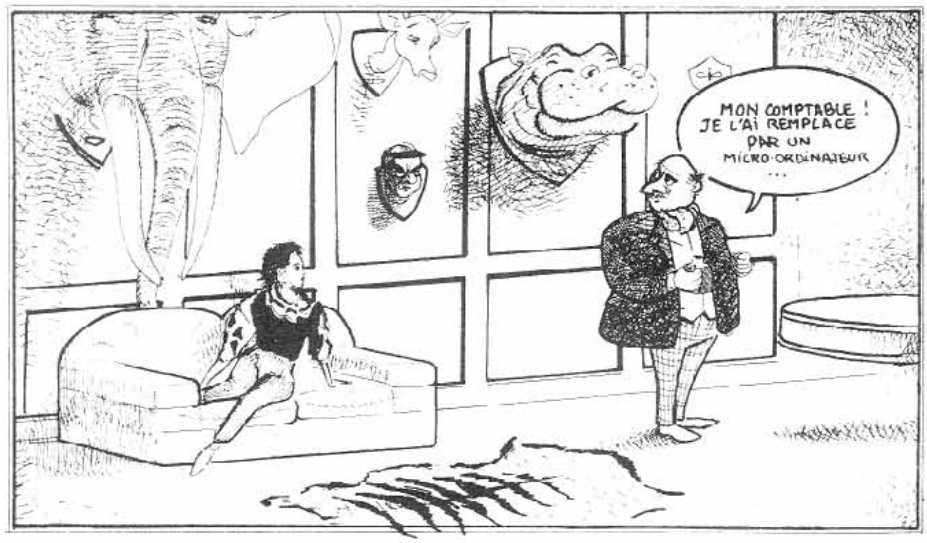
■

Les autres disquettes de Pom's

Pom's a lancé, à l'occasion de son précédent numéro, deux disquettes moins ludiques, proposées chacune à 450 FF TTC.

Disk Manager, de Dan Steerey, est un programme utilitaire qui permet de recréer les commandes du DOS et de programmer simplement, à partir du Basic, des utilitaires personnalisés de gestion du disque qui seraient beaucoup plus longs à programmer en assembleur. C'est à la fois un programme d'une grande utilité et un excellent outil pédagogique pour comprendre le DOS. Quatre utilitaires, écrits à l'aide du Disk Manager, sont fournis en prime sur la disquette : Utili-Disque (reconstruction de disquette, vérification, ...), Ultra-Copie (programme de backup très rapide), Edicat (éditeur de catalogue) et Multi-Disque (gère des fichiers de catalogues, opère des tris, ...). Documentation de 50 pages.

DBSTAG : disquette CP/M pour la création de statistiques et de graphiques à base d'histogrammes (impression non graphique) à partir de fichiers de données sur DBASE II. Ce logiciel peut exploiter, sans limitation ni transformation, toute base de données construite sous DBASE II. Il peut fonctionner seul, ou en sous-programme d'un programme écrit par l'utilisateur. Documentation détaillée, comportant un exemple traité, imprimable à partir de la disquette. ■



pom's

	Montant TTC
• **la disquette HAIFA Source** ☐ **au prix de 55 F la disquette (cf. Pom's n° 5)**	
• **le logiciel H-BASIC** ☐ **au prix de 150 F (cf. Pom's n° 8)**	
• **le logiciel MUSIC** ☐ **au prix de 80 F (cf. Pom's n° 10)**	
• **le Disk Manager** ☐ **au prix de 450 F (cf. Pom's n° 11)**	
• **DBSTAG (CP/M)** ☐ **au prix de 450 F (cf. Pom's n° 11)**	
• **Disquette de jeux A** ☐ **au prix de 80 F (cf. Pom's n° 12)**	
• **Disquette de jeux B** ☐ **au prix de 80 F (cf. Pom's n° 12)**	
• **le logiciel BASICIUM** ☐ **au prix de 150 F (cf. Pom's n° 13)**	
• **la disquette de démo** **au prix de 55 F la disquette**	
☐ CX Système ☐ Jane	
• **Recueil n° 1 de Pom's** (n°s 1 à 4)	
☐ avec ses 3 disquettes au prix de 280 F	
☐ sans disquette au prix de 130 F	
☐ les 3 disquettes seules au prix de 150 F	
• **Recueil n° 2 de Pom's** (n°s 5 à 8)	
☐ avec ses 4 disquettes au prix de 320 F	
☐ sans disquette au prix de 130 F	
☐ les 4 disquettes seules au prix de 190 F	

Je désire recevoir :

• **les numéros de la revue Pom's** ☐ 4 ☐ 7 ☐ 8	
au prix de 35 F le numéro	
• **les numéros de la revue Pom's** ☐ 9 ☐ 10 ☐ 11 ☐ 12 ☐ 13	
au prix de 40 F le numéro	
• **les disquettes d'accompagnement des numéros** ☐ 1 ☐ 2 ☐ 3 ☐ 4	
☐ 5 ☐ 6 ☐ 7 ☐ 8 ☐ 9 ☐ 10 ☐ 11 ☐ 12 ☐ 13	
au prix de 55 F par disquette	

Je désire m'abonner pour 6 numéros à partir du n° .....

☐ sans disquette au prix de 200 F	
☐ avec disquettes au prix de 480 F	
TOTAL :	

Envoyez ce bon de commande et votre règlement à :

Éditions MEV — 64-70, rue des Chantiers — 78000 Versailles

Nom _____

Adresse _____

Ces tarifs comprennent l'envoi postal en France Métropolitaine, CEE et Suisse (voie aérienne exceptée)
Supplément avion : 10 F par numéro et / ou disquette

VOTRE ORDINATEUR

LE MAGAZINE DE L'INFORMATIQUE A LA MAISON

ISSN 0752-2363

N° 6

EN VENTE DANS TOUS LES KIOSQUES

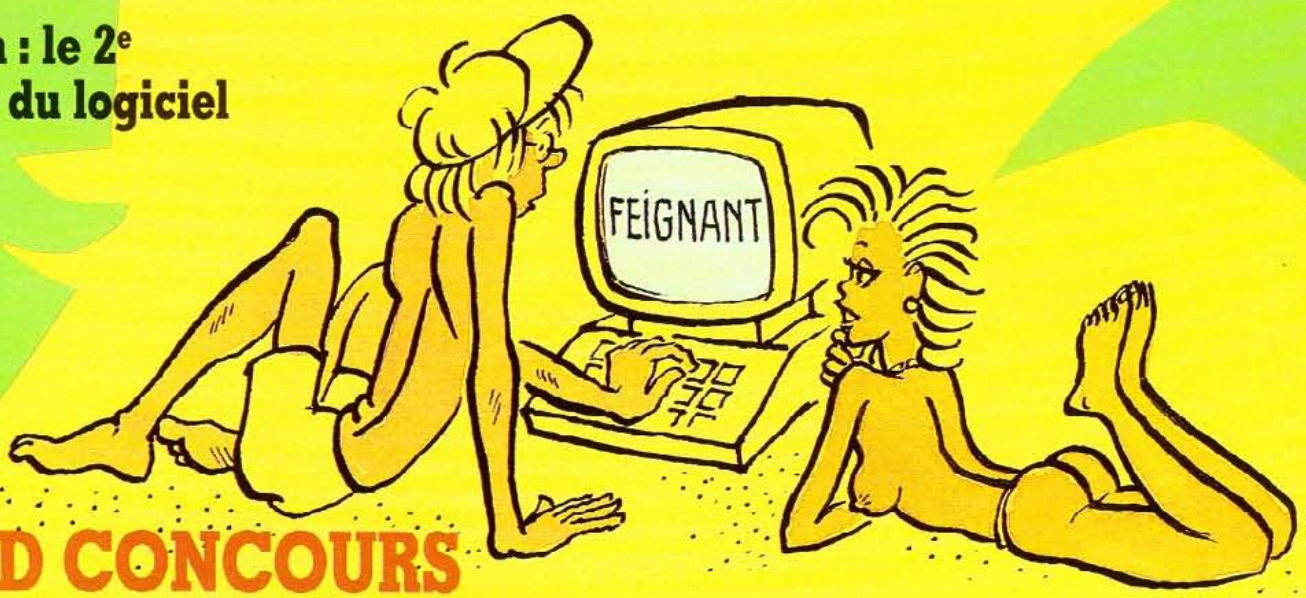
INITIEZ-VOUS EN JOUANT !

supplément de 64 pages cahier de vacances

à l'essai : Casio PB700, Electron, MO5 les ordinateurs autonomes

basic, logo les fiches programmes

Avignon : le 2e Festival du logiciel



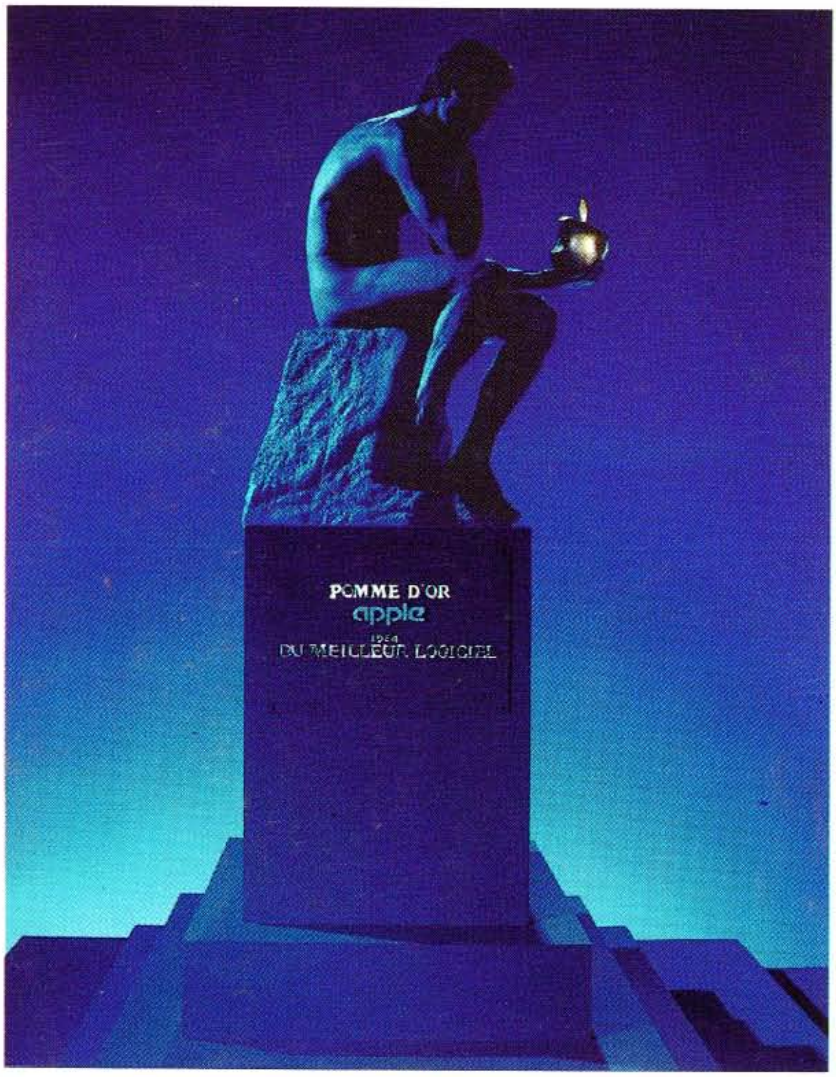
Gabu

GRAND CONCOURS

GAGNEZ DES ORDINATEURS AVEC *France inter* ET VOTRE ORDINATEUR

M 2867 - 6 SP - 25 F JUILLET-AOÛT 84 - 25 F BELGIQUE 207 FB - CANADA 3.65 $ C - SUISSE 7.5 FS

La création



Pomme d'Or 84 du meilleur logiciel

L'informatique appartient aux créateurs. Artistes de l'ordinateur : craquez et croquez la Pomme d'Or Apple 84 du meilleur logiciel. Jusqu'au 30 octobre 1984, pour votre création, Apple vous offre la consécration. Votre concessionnaire-parrain vous assistera dans la mise en forme et le pré-test de votre logiciel. Chaque lauréat recevra, outre le Trophée, un Macintosh et verra son programme édité et distribué. Pionniers de la nouvelle culture, par votre imagination devenez star de la programmation. Avec Apple, apprenez l'homme à la machine.

apple
le goût du savoir

Pour obtenir le règlement et votre dossier de participation, faites tamponner le coupon-réponse ci-joint par le Concessionnaire agréé Apple de votre choix et adressez-le pour inscription à Apple Éducation à l'attention de Jean-Luc Lebrun, avenue de l'Océanie, Z.I. de Courtabœuf, 91944 Les Ulis, BP 131. Tél. (6) 928.01.39. Veuillez m'inscrire à la Pomme d'Or Apple du meilleur logiciel 1984.

Nom _____ Prénom _____

Adresse _____

Profession _____ Téléphone _____

"Le nom Apple et le logo Apple sont des marques déposées par Apple Computer Inc."

Tampon du Concessionnaire

Nom du Concessionnaire _____ Tél. _____