tremplin micro

Compatibilité assurée... Ouf !

N° 11 - Bimestriel - Deuxième année
4 Novembre 1986 - 3 Janvier 1987
254 FB - 11 FS - 33 F

Apple et ProDOS sont des marques déposées par Apple Computer Inc.

Tremplin Micro 11

Un événement attendu, mais néanmoins imprévu, a quelque peu perturbé la pagination de ce numéro... mais la naissance de l'APPLE *IIGS* justifie bien un tel remue-ménage !

L'APPLE *IIGS*

Lisez ci-contre (page I) les premières impressions de notre ami NESTOR.

Les caractéristiques techniques de la bête se trouvent page 2.

A partir de la page 4, Marcel COTTINI vous invite à faire connaissance avec le 65C816, microprocesseur équipant l'Apple *IIGS*.

SOMMAIRE

TREMPLIN MICRO N°12
paraîtra le 3 janvier 1987

Vous avez dit "COMPATIBLE" ?

L'APPLE *IIGS* est né. Probablement avant terme, mais qui pourrait s'en plaindre ? C'est toujours un APPLE *II*, mais ce n'est plus le 65Ø2 (ou son successeur, le 65CØ2). La greffe du cerveau est réussie et nous allons pouvoir disposer, dans quelques semaines, d'une machine capable de rivaliser avec les plus récents des ordinateurs personnels. Ayant eu le plaisir d'essayer l'un des premiers APPLE *IIGS*, je n'ai pas voulu priver plus longtemps les lecteurs de *TREMPLIN MICRO* d'informations précises et objectives sur la bête.

Premier contact

Quand on est un inconditionnel de l'Apple *II*... ce qui est mon cas, cette objectivité reste bien entendu sujette à caution. Le préciser relève de la franchise la plus élémentaire. Voilà qui est fait !

Alors ce premier contact ? Sympa. Et même mieux que cela. Rien à monter. Ni vis ni écrou... seulement des emballages à ouvrir et à vider.

Il semble impossible de commettre une erreur lors des connections. Vous branchez où vous pouvez le faire... et c'est tout ! C'est facile sans doc et cela doit l'être davantage avec la documentation qui accompagnera bientôt la machine. Les détrompeurs existent et je les ai rencontrés !

Côté esthétique, excellente impression : le *GS* est simple, net, agréable à regarder. L'écran couleur ne se révèle pas trop envahissant. L'ensemble paraît plus petit que celui de l'ancien Apple... mais occupe pourtant une place aussi importante sur le bureau.

Le clavier est bien proportionné. Il rappelle celui de l'Apple *IIc*, mais il m'a semblé plus agréable. Il serait néanmoins surprenant qu'il fasse l'unanimité, notamment pour une utilisation intensive.

Heureusement, on apprécie d'emblée la présence de son pavé numérique. Les gauchers seront ravis en constatant qu'Apple ne les a pas oubliés (la souris se connecte indifféremment à droite ou à gauche). *(Suite page II)*

APPLE *IIGS*

Janus-Apple La firme de Cupertino aurait pu oublier ses quatre millions de clients et les abandonner à leur triste sort, avec leurs dizaines (ou centaines) de logiciels. C'est une démarche courante : on efface tout et on recommence. Nous lui saurons gré d'avoir adopté une autre politique, sans doute plus coûteuse, mais dont les possesseurs de *IIc* et *IIe* se féliciteront. L'APPLE *IIGS* est bistandard. C'est d'abord un Apple *II*... grâce à la présence — ô combien efficace ! — d'un seul et unique circuit (ou chip !) : le MEGA II, capable d'émuler la machine que vous et moi connaissons bien.

Le MEGA II sait tout faire : les fonctions de l'Apple *II* et aussi des tâches qui, dans votre Apple actuel, sont généralement confiées à diverses cartes.

JANUS-APPLE vous offre donc, sans utiliser aucun des slots d'extension, les 80 colonnes étendues, le contrôleur 3 pouces 1/2 mais aussi le contrôleur 5 pouces 1/4, le port souris, l'extension de mémoire, la sortie super série, un accélérateur 2,8 Mhz (mais on peut tourner à 1 Mhz).

Menu de configuration

Un menu de configuration (obtenu par un CONTRÔLE-POMME OUVERTE-ESCAPE) permet de modifier des tas de paramètres. L'utilisateur choisit successivement diverses options concernant le niveau sonore (deux réglages), l'affichage (couleur du fond et des lettres), vitesse du 65C816 (normale ou Turbo), vitesse de la souris, etc.
La configuration ainsi établie est mémorisée, ce qui permet de la retrouver lors d'une nouvelle utilistation. N'oublions pas que la carte possède aussi une horloge-calendrier.

Compatibilité On comprendra qu'il nous soit impossible, après un essai de deux semaines, d'affirmer que toutes les anciennes cartes sont vraiment compatibles... toutefois, les exceptions devraient se révéler rares... et il en ira de même pour les périphériques. Personnellement, j'ai connecté sans problème le lecteur 5 pouces 1/4 de mon *IIc* sur le second lecteur 3 pouces 1/2 de l'Apple *IIGS*. Non seulement cette configuration tourne normalement, mais elle démarre sous ProDOS ou DOS 3.3. suivant la disquette placée dans le lecteur 5 pouces 1/4. En fait, on peut affirmer sans risque d'erreur que, à quelques exceptions et détails près, tout ce qui fonctionne sur un Apple *IIe* ou *IIc* s'accommodera de la compatibilité de l'Apple *IIGS*... si les auteurs des programmes ont bien respecté les points d'entrées imposés par Apple aux développeurs.

Je me suis par exemple amusé à tester les disquettes de *TREMPLIN MICRO* sans rencontrer de difficultés insurmontables.

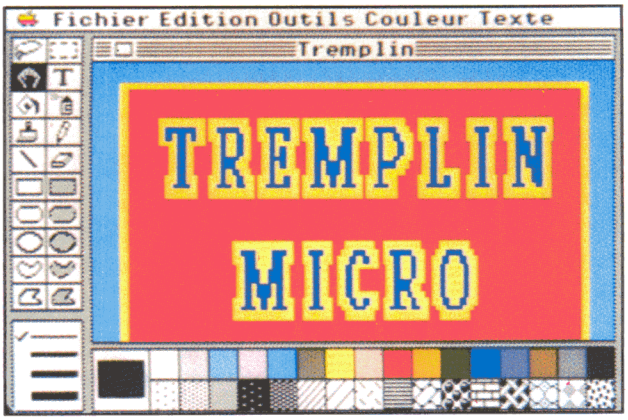
Restrictions

Il convient de les évoquer. Les programmes d'amateurs figureront probablement au nombre des routines acceptées par le *GS* (y compris sous DOS 3.3), il est évident que certains logiciels du commerce planteront royalement la machine, ou se révéleront visuellement mauvais. Ce sera notamment le cas pour les quelques superbes produits exigeant la présence active de la carte Féline, tournant en double haute résolution et en couleur. Bien souvent, il suffira de choisir la vitesse normale (lente) pour utiliser les programmes qui ne tolèreront pas un fréquence d'horloge de 2,8 Mhz. Souhaitons que tous les éditeurs de logiciels se tiennent à la disposition de leurs clients pour leur fournir (à un tarif intéressant) d'éventuelles nouvelles versions.

Janus-GS Mais, tel Janus et ses deux visages, l'APPLE *IIGS* est aussi une machine résolument tournée vers l'avenir. Il suffit, pour s'en convaincre, de lire la fiche technique que nous publions plus loin (page 2).

Si les sept slots d'extension nous laissent actuellement sur notre faim (mais libre à nous de les occuper avec nos "vieilles" cartes !), ils n'en sont pas moins prometteurs. Que de possibilités en vue ! Quant au slot spécial "16 bits", s'il paraît essentiellement réservé aux extensions de mémoire, on peut raisonnablement penser que d'aucuns lui trouveront d'autres usages.

La couleur C'est évidemment l'un des points forts de la machine. En fait, la ROM de 128 K contient non seulement le Basic Applesoft (celui-là, je l'aurais aimé plus musclé !), mais une Boîte à outils graphiques, que l'on dit inspirée du QuickDraw de Macintosh (moi, le MAC, je ne le connais que de nom !). Sera-t-il facile, à partir de ces outils essentiels, de créer et gérer les fenêtres et menus déroulants ? Je le souhaite et attend la documentation pour me faire une opinion. Je crois savoir qu'un bouquin de Nicole Bréaud-Pouliquen (sur le *GS*)



est en cours d'impression au PSI. Il répondra certainement à quelques-unes de mes préoccupations. Sans doute vous posez-vous la question de savoir si le *GS* est compatible avec le MAC... et s'il en lit les disquettes ? Dans l'état actuel de mes connaissances, je vous assure que la réponse est négative. D'ailleurs pourquoi un Apple *IIGS* irait-il s'acoquiner avec un MAC... incapable de nous offrir des images en couleur ? A moins d'être un inconditionnel de MAC (ça existe aussi), on est bien obligé de reconnaître que le *GSPaint* de Version Soft a tout de même une autre allure que le noir et blanc du 68000. Ah ! quand viendra OPEN MAC, couleurs... et ses slots d'extension, nous en reparlerons ! Pour l'heure, place au *GS*.

Le son

Je ne suis pas un dingue de la sono, mais j'aime bien la bonne musique et je suis même un peu musicien sur les bords. Il est probable que je ferai mes débuts de compositeur sur mon Apple *IIGS*. Ça promet pour l'entourage ! D'autres que moi y pensent déjà et, lors du récent SICOB, j'ai "espionné" un groupe de jeunes qui se voyaient déjà avec leur *GS*, au cœur d'une animation où, d'après ce que j'ai pu comprendre, l'aléatoire primerait sur la logique.

Les logiciels

Pas nombreux si l'on s'en tient au seul *GS*. Personnellement, je n'ai testé que *GSPaint*, mais *GSWrite* (également de Version Soft) sera livré avec la machine... et sa finition ne devrait donc point tarder. D'autres développeurs, en France et à l'étranger, travaillent actuellement sur des produits superbes, destinés au dernier-né de chez Apple. Nous en parlerons au fur et à mesure de leur apparition sur le marché... dans la mesure où nous les estimerons intéressants ! ORCA/M (voir notre dernier numéro) est déjà prévu pour travailler sur l'Apple *IIGS*, mais il est destiné à des programmeurs avertis, capables de comprendre une documentation en anglais. Il convient de noter que la plupart des grands classiques fonctionneront en émulation... à quelques rares exceptions près (gageons que les fabricants auront à cœur de corriger les éventuelles imperfections).

Un conseil : exigez dès maintenant une disquette 3 pouces 1/2 de tous les logiciels que vous déciderez de vous offrir, et cela dès que leur prix dépasse 500 F. C'est une assurance pour l'avenir.

Périphériques

On l'a vu, le *GS* accepte les nouveaux lecteurs 3 pouces 1/2, mais reconnaît les anciens. Vous n'êtes donc pas obligé, si vous possédez déjà l'Unidisk, de vous offrir une configuration complète. Je ne vous apprendrai rien en précisant que les 800 K sont chaînables... y compris avec un 5 pouces 1/4 (voir plus haut). Vous pouvez aussi glisser la carte contrôleur de votre *IIe* dans le *GS*... et y connecter vos lecteurs actuels. Ce ne sera pas pour autant le Salon du Bricolage ! Et puis, on aime ça... chez les applemaniaques, alors !...

Imprimante

Votre IMAGEWRITER (I ou II) fera l'affaire, avec une préférence pour la II... et ses caractères souris. Par la suite, vous éprouverez peut être l'envie de vous connecter — en passant par quelque réseau... ou directement si vous avez les moyens de vous l'offrir vous-même — à une Laser Writer. Cela supposera que vous disposerez alors d'un logiciel comme PageMaker (dont on distribue actuellement la version 1.2 française... mais pour MAC). Pourquoi pas ? Il est bien permis de rêver, non ?

Moi, je ne vous cacherai pas que, lorsqu'un fabricant me parle de son logiciel, je lui pose illico la question de confiance :

— Et il fonctionne aussi sur le GS ?

L'autre compatibilité

Vous me voyez venir ? Il va être question d'IBM. Mais d'abord, que signifie la compatibilité avec IBM ? Que l'on pourra utiliser, sur Apple *IIGS* des logiciels prévus pour l'IBM PC ? En connaissez-vous beaucoup qui n'ont pas leur équivalent sur Apple ? Ou qui ne l'auront pas, au cours des prochains mois sur APPLE *IIGS* ? D'ailleurs, le vrai compatible n'existe pas. C'est un leurre. Pourrez-vous lire, sur Apple *IIGS*, des fichiers écrits sur un IBM PC ? C'est déjà possible... tant il est vrai que tout est réalisable, quand il ne s'agit que de transmettre ou recevoir des données.

Evidemment, pour exécuter un programme, c'est une autre histoire !

On sait pourtant que, par cartes "coprocesseurs" interposées, l'Apple *IIGS* deviendra compatible avec CP/M (c'est la moindre des politesses) et avec MS/DOS (d'où sort-il, celui-là ?).

Moi, stoïque, fidèle à mon Apple *IIe*, mais tout disposé à le mettre dans un coin et à le remplacer par le *GS*, j'attends ma DOC, mon langage machine, mes adresses (et notamment celles du moniteur... car j'en ai semble-t-il égaré quelques-unes)... et une sœur de la belle bête que je viens de toucher... mais que je dois hélas restituer au service de Presse d'Apple !

NESTOR

Une révolution dans le domaine de la saisie des données :

la reconnaissance de l'écriture

PERSONAL WRITER

QUE sera l'informatique de 1990 ? Nous en avons aujourd'hui un avant-goût, sur MACINTOSH, avec *PERSONAL WRITER* — un nom anglais pour désigner un logiciel bien français, mais que ses promoteurs espèrent diffuser dans le monde entier. En fait, *PERSONAL WRITER* est le produit numéro un d'une société créée en 1985 : ANATEX S.A. Une dizaine de personnes — dont sept ingénieurs aussi jeunes qu'enthousiastes — y travaillent sous la direction du fondateur : Xavier Maury. Mais revenons à *PERSONAL WRITER !*

Objet Déchiffrer les caractères manuscrits et les transformer directement en texte dactylographié, sans nulle intervention du clavier.

Le système Il se compose d'une tablette à digitaliser, d'un stylo électronique et d'un logiciel de reconnaissance des caractères. Actuellement, *PERSONAL WRITER* n'est utilisable que sur Macintosh, mais il sera progressivement adapté à d'autres machines (IBM PC... et n'en doutons pas... l'Apple *IIGS*).

Apprentissage Pendant la phase dite d'apprentissage, la machine se contente de reconnaître l'écriture de l'usager, mais les variations d'écriture continueront ensuite d'être enregistrées lors de chaque utilisation. Plus on se sert de *PERSONAL WRITER* et plus il devient intelligent et performant !

Ecriture naturelle Avec ce système, on écrit normalement, sur la tablette, sans contrainte particulière. On peut rayer un mot, en ajouter un autre, intercaler une phrase ou changer la taille d'une lettre. L'écran restitue fidèlement toutes les modifications apportées au texte.

Et les fautes ? Mais oui, grâce à un dictionnaire intégré, *PERSONAL WRITER* est également capable de détecter et donc de corriger les fautes d'orthographe (on prévoit d'ailleurs des dictionnaires spécialisés). Evidemment, pour l'instant du moins, il n'est question que de corriger les mots... et non la syntaxe, mais tous les espoirs sont permis !

Prix Commercialisation prévue au début de l'année prochaine. Le prix public se situera entre 14 000 et 18 000 F (tablette à digitaliser, stylo électronique, logiciel et dictionnaire de 200 000 mots).

Clément RENARD.

tremplin micro

11

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

BIMESTRIEL

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

EDITIONS JIBENA

Direction-Rédaction :
Editions JIBENA
Guy-HACHETTE

La Petite Motte — Senillé
86100 CHÂTELLERAULT.

Téléphone :
49-93-66-66

PUBLICITÉ :
Raymond JULLIEN
(1) 45.75.41.81

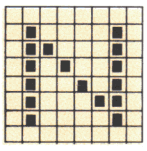
Commission paritaire :
Demande en cours.

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

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

L'APPLE *II GS*

Votre ordinateur personnel de demain ?



OUS l'attendions tous. On en avait beaucoup parlé, ici et là, mais surtout dans les revues d'informatique. Jusqu'au 15 septembre — un peu plus tôt pour quelques rares privilégiés — on ignorait quel serait exactement son nom de baptême. Il en va ainsi pour la plupart des nouveaux-nés.

Ce sera donc un *GS : G* pour le graphisme et *S* pour un son qui a laissé babas bien des journalistes (les *Amiga* et autres *Atari* sont de nouveau à la traîne). Un vrai synthétiseur 16 voix (ou instruments), comme sur les appareils réservés aux professionnels. Sans doute en déduisez-vous que, prochainement, on utilisera le *GS* pour commander des animations avec paroles et musique ? Vous avez sûrement raison. Reproduire la voix humaine ne pose pas de problème particulier à l'Apple *II GS*. On va bien s'amuser !

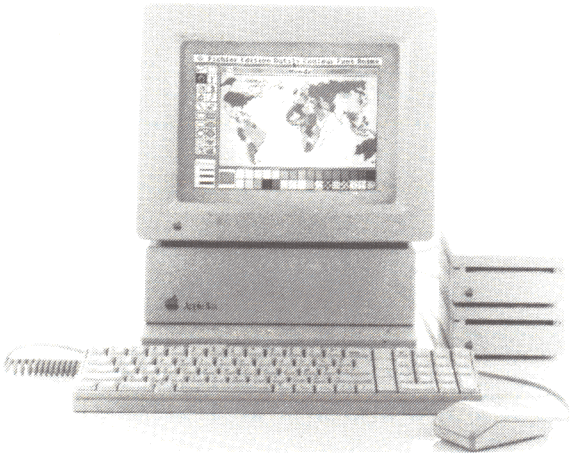
Je ne vous apprendrai rien en ajoutant que le *GS* doit beaucoup — et notamment sa rapidité d'exécution — au 65C816, un microprocesseur 16 bits dont vous découvrirez rapidement les nombreuses et immenses ressources.

J'avoue avoir été conquis, et d'emblée, par la démonstration que j'ai regardée et écoutée. Evidemment, le graphisme ne présente pas encore la qualité photo (telle qu'on la conçoit dans une revue comme CHASSEUR D'IMAGES), mais il s'en rapproche un peu plus, notamment quand il utilise la palette de couleurs avec ses 4Ø96 nuances.

Mais que deviennent votre vieil Apple *IIe* et sa montagne de logiciels ? Rassurez-vous, le *GS* saura exécuter la plupart des programmes qui lui étaient destinés, mais environ trois fois plus vite. Il est probable que vos routines personnelles figureront au nombre des merveilles réellement compatibles. Mais il y a mieux : dans quelques mois (sans doute au début de l'année 1987), vous aurez la possibilité, pour environ 5000 F, de remplacer votre carte actuelle par la nouvelle. Quel intérêt ? Répondre actuellement à cette question se révélerait fort présomptueux : attendons d'avoir réalisé un tel échange pour en parler sérieusement, mais félicitons tout de même Apple d'avoir pensé aux 4 millions d'utilisateurs fidèles à la Pomme.

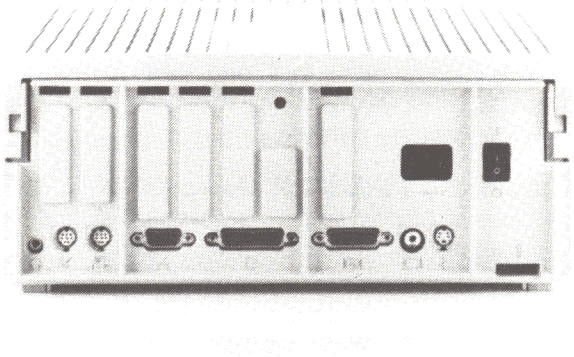
Au fait, il paraît que les premiers Apple *II GS* seront disponibles dès la fin de ce mois de novembre...

BONNE PROGRAMMATION ! **GUY-HACHETTE.**

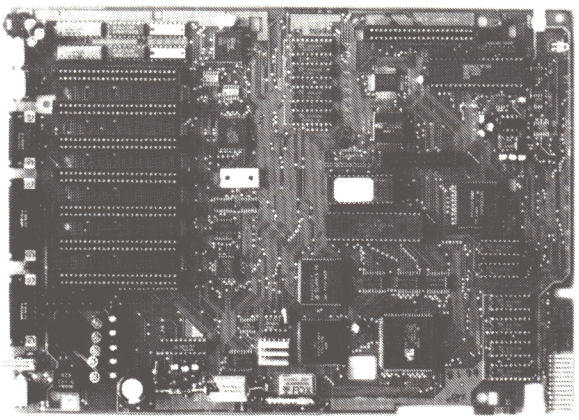


L'Apple IIGS

L'APPLE IIGS



Le panneau arrière de l'Apple IIGS



CARACTÉRISTIQUES TECHNIQUES

L'Apple IIGS comprend

- Un microprocesseur 16 bits 65C816, émulant le 65CØ2 ;
- Des processeurs spécialisés :
  - — MEGA II pour l'émulation Apple II ;
  - — VGC pour les nouveaux modes graphiques ;
  - — IWM pour les entrées-sorties disques 5"1/4 et 3"1/2 ;
  - — ENSONIQ, 32 oscillateurs pour le son.
- 256K de mémoire vive (RAM) extensible à 8 Mégaoctets ;
- 128K de mémoire morte (ROM) contenant les outils QuickDraw II et le BASIC Applesoft ;
- Un clavier détachable AZERTY avec bloc numérique ;
- Une souris ;
- Une horloge temps réel intégrée ;
- 8 ports intégrés :
  - — Un connecteur son pour haut-parleur ou casque d'écoute ;
  - — 2 ports série RS422C (compatibles RS232) dont l'un peut servir de port Apple Talk ;
  - — Un port pour manettes de jeu ou manche à balai (joystick) ;
  - — Un port lecteur de disquette 5"1/4 et 3"1/2 ;
  - — Un port video couleur analogique RVB ;
  - — Un connecteur video pour moniteur monochrome ;
  - — Un connecteur ADB (Apple Desktop Bus) pour clavier et souris.
- 8 connecteurs internes d'extension, dont un connecteur pour carte d'extension mémoire vive et mémoire morte ;
- Des graphiques super-haute résolution 64Ø × 2ØØ en 4 couleurs et 32Ø × 2ØØ en 16 couleurs parmi 4Ø96 couleurs possibles ;
- Sons 16 voix permettant de restituer la musique et la voix humaine.

L'Apple IIGS est compatible avec la gamme Apple II, c'est-à-dire avec la presque totalité des logiciels, périphériques et cartes d'extension Apple II. Par adjonction de cartes coprocesseurs, il sera également compatible CP/M et MS/DOS.

L'Apple IIGS sera commercialisé au prix de 15900 Frs. HT. Il s'agit de la version 512K, moniteur couleur, lecteur 800K 3"1/2, avec le logiciel de création graphique GS Paint* et le logiciel de traitement de texte GS Write*. La même configuration, mais avec un moniteur noir et blanc, sera vendue au prix de 12900 Frs. HT.
Notre "cher" Apple IIX n'est pas donné ! ■

** GS Paint et GS Write sont des marques déposées Version Soft.*

MARCEL COTTINI

L'AVENIR est à vous... avec l'APPLE *II* et son 65C816

- Le Microprocesseur...
- Ses instructions
- Ses registres
- Ses codes opérations
- Ses adressages sommaires

DANS CE NUMÉRO

DANS NOS PROCHAINS NUMÉROS

1 LE MICROPROCESSEUR W65C816PI-2

AVANT-PROPOS

Le microprocesseur 6502, comme le Z80, est un rescapé du début de l'informatique individuelle ayant équipé d'origine les premiers Apple *II*. L'évolution de la technologie aidant, un problème de taille s'est posé aux chercheurs de Cupertino : comment faire évoluer la série des Apple *II* sans perdre, dans l'immédiat, l'accès à l'imposante bibliothèque de logiciels ? Le premier essai tenté (et loupé) a été la mise en place du 65C02 dans l'Apple *IIc* et l'Apple *IIe+*. Ce microprocesseur, bien que plus complet que son prédécesseur, ne fut pas à la hauteur de la concurrence du marché actuel (processeur du type 8086/8088 ou 68000).

Le W65C816PI-2 de Western Design Center Inc est un successeur direct du 65C02, l'architecture interne des registres étant réalisée sur 16 bits de large avec un bus de données de 8 bits. Il peut adresser 16 Mo, possède un registre supplémentaire (Direct Register - D) et porte à 16 bits tous les registres existants. Néanmoins, son bus de données reste un 8 bits, ce qui le limite à travailler sur des mots d'une longueur de 8 bits. Cette situation de faux 16 bits n'est nullement un handicap, et ne le pénalise absolument pas par rapport à d'autres «chips». Il possède en effet le plus faible temps d'accès mémoire des microprocesseurs grand public (un cycle horloge). La version W65C816PI-8 autorise un fonctionnement de l'horloge huit fois plus rapide que celle de l'Apple II, c'est-à-dire 8 Mhz. C'est un concurrent potentiel du processeur 8088, cœur des IBM PC et compatibles IBM PC.

Il y a aussi le 65C802...

Le 65C816 n'est d'ailleurs pas le seul de sa famille, puisqu'il existe un autre microprocesseur possédant les mêmes instructions machine et de surcroît compatible broche à broche (pin to pin) avec le 6502, c'est le 65C802. Ces deux processeurs font partie de la famille des composants hybrides, capables de travailler sur 8 ou 16 bits en passant d'un mode à un autre.

Neuf registres de base

Le microprocesseur 65C802 peut être installé directement sur un Apple *IIe* sans la moindre modification. Par ailleurs, il pourra être programmé puisque certains assembleurs connaissent déjà son langage machine (S-C Macro Assembler). Dans les versions du 6502, les registres sont au nombre de six : les registres d'index X et Y, l'accumulateur, le registre d'état, le compteur ordinal et le registre pointeur de pile. Leur taille est de 8 bits de large, sauf pour le compteur ordinal, dont la taille est de 16 bits (adresses $0000 à $FFFF).

La structure des 65C816 et 65C802 est dotée de 9 registres de base, dont six sont communs aux 65C816, 65C802 et 6502. Les nouveaux arrivants sont : le registre d'adressage direct (D) et deux registres annexes qui se composent d'une part du registre page de données (DBR) et d'autre part du registre page programme (PBR). L'innovation principale réside dans le mode d'adressage pouvant être basculé soit en mode 8 bits, soit en mode 16 bits. Les registres de ce nouveau produit sont :

- le registre page de données. Data Bank Register -DBR
- le registre d'index X. Index Register X - X
- le registre d'index Y. Index Register Y - Y
- le registre pointeur de pile. Stack Pointer - S
- le registre accumulateur. Accumulator - C
- le registre page programme. Program Bank Register -PBR
- le compteur ordinal. Program Counter - PC
- le registre direct. Direct Register - D
- le registre d'état. Status Register - P

La sortie sur le marché de la série des Apple *II* suscita, parmi un grand nombre de ses adeptes, un enthousiasme tel que la courbe des ventes s'en ressentit. Mais, la limitation de la capacité mé-

moire et la lenteur relative de la vitesse d'exécution des instructions (processeurs 6502 et 65C02) furent très vite sujet à caution. Il faut par ailleurs rappeler que le marché de l'informatique subit périodiquement des bouleversements provoqués en grande partie par la sortie de nouveaux produits. C'est ainsi qu'Apple Computer a fait la douloureuse expérience de la montée en flèche du matériel IBM (PC, AT et XT). L'année 1986, témoin de la purge de bon nombre de modèles de micro-ordinateurs, reste tout de même une bonne année pour la micro-informatique grand public. Apple Computer a relevé le défi avec son nouveau modèle (Apple *II-GS*) équipé d'un microprocesseur hybride capable de travailler sur 8 ou 16 bits en passant respectivement du mode émulation au mode native. Le processeur est un 16 bits émulable avec le 6502 par l'intermédiaire d'un registre d'état (bit E associé aux bits M et X).

Un certain nombre de revues spécialisées dans le domaine de la micro-informatique grand public, en particulier Tremplin Micro, sont en droit de se poser des questions sur l'avenir immédiat des Apple *II* (processeurs 6502 et 65C02). Que tout ce monde se rassure, la sortie de l'Apple *II-GS* n'empiète en rien sur le marché existant de l'Apple *II*. L'Apple *II-GS* s'adresse à un nouveau public, désirant évoluer dans le domaine de la programmation tout en gardant la possibilité d'usage de la bibliothèque des logiciels du 6502. En quelque sorte un certain retour aux sources : compatibilité garantie, avec en prime la possibilité d'un nouveau mode de programmation (16 bits).

L'Apple *II-GS*

L'Apple *II-GS*, dernier né de la lignée des Apple *II*, est doté d'un microprocesseur 16 bits (65C816) dont le modèle retenu possède un temps d'horloge de 2 Mhz. Ce processeur, cœur de la «machine», est capable d'adresser 16 Méga-octets (16.384.000 octets) de mémoire vive (RAM). Par ailleurs, 4 Méga-octets seront directement accessibles dans la carrosserie du modèle fabriqué en série. L'originalité de cette opération commerciale réside dans le fait que ce processeur peut tout aussi bien fonctionner en 16 bits qu'en 8 bits. Dans ce dernier cas, on dit qu'il émule avec le 6502, pouvant utiliser toute la bibliothèque des logiciels existants. Certaines similitudes frappantes existent par rapport au Macintosh. La configuration de base se divise en trois maillons : unité centrale, clavier détachable et écran de visualisation.

Haute résolution extra

Un soin très particulier a été apporté à la haute résolution graphique. Une palette de 16 couleurs est disponible pour l'utilisateur, permettant un affichage de 200 x 320 points (ou pixels) sur l'écran. Le système d'exploitation ProDOS, dernier né des programmes système d'Apple Computer, a été mis à jour pour s'adapter aux exigences du 65C816 (ou 65C802). Il intègre un «Finder» style Macintosh et une horloge. Certaines routines ont été réécrites. Un certain nombre de dispositions ont été prises pour donner à ce nouveau-né un new look capable d'accrocher les «branchés» potentiels. Toute une panoplie d'extensions sont déjà disponibles, on pourrait citer une carte sonore* avec synthétiseur à 16 voies, la possibilité de raccordement au réseau AppleTalk, la compatibilité des lecteurs de disquettes 3 pouces et demi (800 Ko) et des lecteurs de 5 pouces 1/4 (140 Ko). Par ailleurs, tout est prévu pour permettre à votre machine préférée de dialoguer avec un IBM PC en lui raccordant une carte interface munie du fameux processeur 8088. C'est ainsi que la riche bibliothèque de logiciels MS-DOS sera à la portée de l'heureux détenteur du nouvel Apple. Une ombre pourtant au tableau de ce magnifique palmarès : le prix.

Cet article a été rédigé et composé avant la naissance officielle de l'Apple IIGS et aucune carte d'extension n'est nécessaire pour bénéficier d'un remarquable synthétiseur....

MICROCIRCUITS CMOS 8/16 BITS - Microprocesseur familial

Ce nouveau type de microprocesseur fait partie d'une famille de composants de la série 8 bits/16 bits CMOS. Le numéro de référence complet est W65C816PI-2. C'est un 16 bits, comportant un registre supplémentaire par rapport au 6502. Le registre de données est un 8 bits ne traitant que des mots de 8 bits de large. Cette restriction est imposée pour permettre l'émulation avec la série des processeurs 6502.

A la mise sous tension de la configuration, mode native, le bit E du registre d'état a la valeur zéro. En mode émulation, le bit E sera positionné à la valeur 1. Différents modes d'adressage sont possibles suivant la valeur du bit E. Si E = 1, la configuration se trouve en mode émulation, elle est alors compatible avec le 6502 (8 bits). Si E = 0, le mode est native avec la possibilité de traiter des registres ou la mémoire sur 16 bits.

COMPATIBILITÉ ENTRE LE 65C802 ET LE 6502

Le microprocesseur 65C802 (16 bits processor) est un produit similaire au 65C816, avec une compatibilité broche par broche totale (pin to pin) avec le 6502 de la série des Apple II. Il est donc permis de remplacer le 6502 par un 65C802, aux performances plus adaptées à notre temps.

Ce type de produit a l'opportunité de se substituer à un autre processeur sans modifier la carte mère. Il autorise dès sa mise en place la possibilité d'utiliser deux modes de programmation .

— en émulation, c'est toute la bibliothèque de l'Apple *II*(6502) qui sera offerte à l'utilisateur ;

— en mode native, des instructions supplémentaires et plus puissantes seront disponibles. L'avantage réside en la mise en place d'un seul composant mettant la puissance de l'adressage 16 bits à la portée du programmeur amateur.

En mode native, c'est-à-dire à la mise sous tension de la configuration, l'adressage 16 bits sera disponible, tandis qu'en mode émulation la configuration sera compatible avec le bon vieux 6502 (8 bits). Le microprocesseur 65C802 est donc un excellent produit de remplacement du 6502.

Le microprocesseur 65C816 ne peut pas remplacer directement le 6502, le brochage n'étant pas le même.

L'organisation interne des microprocesseurs 65C802 et 65C816 se divise en deux grandes parties :

— la partie réservée aux registres ;

— la partie réservée pour le contrôle des instructions (codes).

Architectures du processeur :

Architecture interne. Les signaux contribuant au transfert et à l'exécution des instructions relatives au processeur, via la section de contrôle des instructions, sont du type 16 bits pour les processeurs 65C802 et 65C816.

Architecture externe. Le bus de données (data bus) à l'encontre de la structure des registres internes (16 bits) est de 8 bits.

COMPARAISON DES MICROPROCESSEURS 65C802 ET 65C816

FONCTION	MODE ÉMULATION E = 1	MODE NATIVE E = 0
Pointeur de pile (S) Direct indexé	8 bits Page 1 Interne à une page	16 bits Dépassement limite page mémoire
Registre d'état (P)		
— Bit 4 (X)	Toujours à 1. Hormis si la pile est réactualisée suite à une interruption Hardware.	X = 0 Index sur 8 ou 16 bits.
— Bit 5 (M)	Toujours à 1.	M = 0 Accumulateur sur 8 ou 16 bits.
Branchement par dépassement de la limite d'une page mémoire.	4 cycles	3 cycles
Situation des vecteurs d'interruption		
ABORT	$00FFF8/$00FFF9	$00FFE8/$00FFE9
BRK	$00FFFE/$00FFFF	$00FFE6/$00FFE7
COP	$00FFF4/$00FFF5	$00FFE4/$00FFE5
IRQ	$00FFFE/$00FFFF	$00FFEE/$00FFEF
NMI	$00FFFA/$00FFFB	$00FFEA/$00FFEB
RESet	$00FFFC/$00FFFD	$00FFFC/$00FFFD (1-E)
Program Bank (PB) pendant une interruption — retour par RTI.	Non empilé/dépilé	Empilé et dépilé
Tension nulle sur la broche RDY du processeur - Write		
— 65C802	Ignoré avant Read	Processeur stop
— 65C816	Processeur stop	—
Sauvegarde (Write) pendant Read/Modify/Write	Durée 2 cycles	Durée 1 ou 2 cycles suivant le bit M du registre d'état.

DESCRIPTION DU MICROPROCESSEUR 65C816

— CMOS désigne une fabrication du produit en Metal Oxyde Semi-conducteur pour MOS, et où le C désigne que ce circuit est complémentaire. Les Jonctions semi-conducteurs du type P (Positif) et N (Négatif) sont utilisées de manière complémentaire. Ce procédé de fabrication réduit considérablement la dissipation d'énergie (chaleur) en autorisant une alimentation du circuit sous une faible intensité.

— émulation, d'où totale compatibilité des programmes écrits pour le processeur 6502 (8 bits).

— 16 bits pour les registres Accumulateur, pointeur de pile et d'index (X et Y).

— un registre 16 bits supplémentaire par rapport au 6502. Registre Direct (Direct Register) pour l'adressage de la page zéro (16 bits). Si le registre D est égal à 0, on se trouve en présence de l'adressage page zéro du 6502 (8 bits).

— deux registres annexes 8 bits, supplémentaires par rapport au 6502. Le registre page de données et le registre page programme.

— 24 modes d'adressage (le 6502 n'en possédait que 13).

— l'instruction WAI attend une interruption, tandis que l'instruction STP inhibe l'horloge du processeur. Cette commodité permet de réduire la consommation en énergie et diminue les interruptions latentes.

— un total de 91 instructions de base autorise 255 codes instructions combinés, suivant le mode d'adressage utilisé.

— une instruction spécifique (COP) pour l'utilisation d'un coprocesseur. Elle affecte l'adressage de la pile et force à 1 le bit I et à 0 le bit D du registre d'état.

— déplacement d'une page source (block bank) vers une page de destination (bank). Instruction très puissante pour le déplacement de données d'une page mémoire source vers une page mémoire destination.

— permet d'effectuer des opérations sur 16 bits. Le bus d'adresses est organisé pour adresser 16 Mo de mémoire vive.

— sélection en mode 8 bits pour émuler avec le processeur 6502.

— validation des broches de sortie du circuit de contrôle d'état du processeur (Status Control) : Valid Program Address (VPA) et Valid Data Address (VDA).

SIGNIFICATION DU NUMÉRO DE RÉFÉRENCE

W 65C816 P I -2

W type standard
WC usage général

65C816 identification du produit

65 — série
C — CMOS
816 — 8/16 bits

P enrobage du processeur

P = plastique
C = céramique
D = procédé d'enrobage
E = réservé à l'automobile

I température normale

sans indice = 00 à 70 degrés
I = 40/ à 85 degrés
M = 55 à 125 degrés

-2 horloge du processeur

-2 = 2 Mhz
-4 = 4 Mhz
-6 = 6 Mhz
-8 = 8 Mhz

COMPATIBILITÉ DES MICROPROCESSEURS DE LA SÉRIE 65C...

FONCTION	65C816 - ÉMULATION	65C02	6502
Mode décimal			
— Après interruption	Bits Ø-D	Bits Ø-D	Inutilisés
— Drapeaux N et Z	Validés	Validés	Indéterminés
— ADC et SBC	Pas de cycle en +	+ 1 cycle	Pas de cycle en +
Read/Modify/Write			
— Adressage absolu indexé. Pas de chevauchement de page mémoire	7 cycles	6 cycles	7 cycles
— Write	Dure 2 cycles	Dure 1 cycle	Dure 2 cycles
— Verrouillage de la mémoire	Dure 3 cycles	Dure 2 cycles	Non disponible
Saut Inconditionnel (JMP Indirect)			
— Durée en Cycles	5 cycles	6 cycles	5 cycles
— Saut à l'adresse opérante - forme XXFF	Correcte	Correcte	Non valide
Branchement ou passage d'une limite de la page mémoire	lecture d'un byte	Lit un byte	Read non valide
Tension nulle sur la broche RDY durant l'écriture -Write.			
— Processeur 65C802	Ignoré avant Read	Stop 65C802	Idem au 65C816
— Processeur 65C816	Processeur stop	—	—
Ecriture (Write) durant Reset	Non	Oui	Non
Déclaration de codes machine inexistants	Inopérant	Inopérant	Indéterminé
Signal valide sur la broche RDY du microprocesseur	Entrée/sortie	Entrée	Entrée

Pourrez-vous programmer l'APPLE IIGS ?

Certains de nos correspondants s'inquiètent et paraissent redouter leurs premiers contacts avec la bête...

Vous retrouverez, sur l'APPLE IIGS, le bon vieux Basic Applesoft que vous connaissez déjà. Vous le retrouverez avec ses inconvénients (nombre d'instructions limité, absence de PRINT USING, etc.) et ses avantages : une certaine rapidité et une grande tolérance lors de la saisie. Par conséquent, aucun problème de ce côté, sinon que vos programmes tourneront au moins deux fois plus vite.

Vous aurez par contre avantage à pratiquer le langage machine. Le 65C816, microprocesseur puissant et complet vous autorisera à créer des routines à la fois courtes et ultra-performantes. Pensez aussi au Langage C... dont on dit beaucoup de bien.

NESTOR.

2 LES INSTRUCTIONS DU MICROPROCESSEUR 65C816

GÉNÉRALITÉS

Le microprocesseur **65C816** possède la propriété de pouvoir traiter des données suivant deux modes d'adressage différents : le mode émulation qui traite des registres de 8 bits et le mode native qui traite des registres de 16 bits. Le passage d'un mode à l'autre se fait en positionnant la retenue (Carry Flag) et en transmettant au système l'instruction XCE (Exchange Carry an Emulation Bits). La commande XCE positionne le drapeau en E (bit d'émulation). Si E est à un, le microprocesseur est en mode émulation et adresse comme un 65Ø2 (8 bits). Toute la bibliothèque des logiciels existants reste alors disponible. Si E est à zéro, le mode native est actif. Dans ce mode, le microprocesseur utilise certains de ses registres sous un format 16 bits. En mode native, les bits 4 (Index Register Select) et 5 (Memory Select) du registre d'état changent de signification. Si le bit 4 est positionné à zéro, alors les registres d'index sont considérés comme des registres de 16 bits de large. Si le bit 5 est à zéro, l'accès à la mémoire se fait sur 16 bits.

Le microprocesseur **65C816** utilise 255 codes instructions, soit 91 de plus que le 65CØ2. Ces nouvelles instructions sont pour la plupart des améliorations apportées à l'adressage du 65Ø2, soit pour l'utilisation de la fonction 16 bits, soit pour exploiter les nouveaux modes d'adressage. Par ailleurs, de nouvelles instructions de transfert entre les registres ont fait leur apparition, sans oublier l'innovation des instructions de déplacement de bloc mémoire, ainsi que l'instruction COP, pour une utilisation d'un coprocesseur.

ADC Addition avec retenue — Add Memory to Accumulator with Carry. On ajoute à l'accumulateur le contenu de la case mémoire plus le bit de retenue. Cette opération s'effectue en mode binaire ou décimal. Agit sur les bits N, V, Z et C du registre d'état.

AND ET logique — AND. ET logique (AND) entre l'accumulateur et la mémoire. On effectue une opération bit à bit avec l'accumulateur et la mémoire. Permet de forcer des bits à zéro. Agit sur les bits N et Z du registre d'état.

ASL Décalage d'un bit vers la gauche — Shift One Bit Left (Memory or Accumulator). On décale à gauche d'un bit, opération effectuée sur l'accumulateur ou la mémoire. Un zéro entre à droite tandis que le bit sortant à gauche va dans la retenue. Agit sur les bits N, Z et C du registre d'état. Pour tous les modes d'adressage utilisant les instructions relatives à Read/Modify/Write, excepté l'adressage accumulateur, les règles suivantes sont à appliquer :

— Ajouter 2 cycles si E = 1 ou E = Ø et M = 1 (8 bits mode) ;

— Ajouter 3 cycles si E = Ø et M = Ø (16 bits mode).

BCC Branchement si pas de retenue — Branch on Carry Clear (C = Ø). Si le bit C = Ø on saute à l'adresse indiquée, sinon, on continue en séquence. Aucune action sur le registre d'état.

— Ajouter 1 cycle si le branchement est réalisé.

— En mode émulation (E = 1), ajouter 1 cycle si le branchement est réalisé et si une page mémoire est franchie.

BCS Branchement si la retenue est à 1 — Branch on Carry Set (C = 1). Si le bit C = 1 on saute à l'adresse indiquée, sinon, on continue en séquence. Aucune action sur le registre d'état.

— Ajouter 1 cycle si le branchement est réalisé.

— En mode émulation (E = 1), ajouter 1 cycle si le branchement est réalisé et si une page mémoire est franchie.

(suite page 10)

BEQ Branchement si zéro — Branch if Equal (Z = 1). Si le bit Z = 1 on saute à l'adresse indiquée, sinon, on continue en séquence (bit Z = 1 si le dernier résultat est nul, ou si la dernière comparaison a donné l'égalité). Aucune action sur le registre d'état.

— Ajouter 1 cycle si le branchement est réalisé.

— En mode émulation (E = 1), ajouter 1 cycle si le branchement est réalisé et si une page mémoire est franchie.

BIT Test de bits — Bit Test. On effectue le ET virtuel entre l'accumulateur et la mémoire spécifiée. Le résultat n'est pas remis dans l'accumulateur (AL ou A) qui reste inchangé. L'indicateur Z (Z flag) est positionné en conséquence. Agit sur les bits N, V et Z du registre d'état. L'instruction BIT n'affecte pas les bits N et V en adressage immédiat. Dans les autres modes d'adressage, N et V (flags) occupent respectivement les bits 7 et 6 ou 15 et 14 de la mémoire adressée, suivant le mode actuel (byte — 8 bits ou expression — 16 bits).

BMI Branchement si négatif — Branch if Result Minus (N = 1). Si le bit N = 1 on saute à l'adresse indiquée, sinon, on continue en séquence. Le résultat est négatif. Aucune action sur le registre d'état.

— Ajouter 1 cycle si le branchement est réalisé.

— En mode émulation (E = 1), ajouter 1 cycle si le branchement est réalisé et si une page mémoire est franchie.

BNE Branchement si non égal — Branch if Not Equal (Z = Ø). Si le bit Z = Ø on saute à l'adresse indiquée, sinon, on continue en séquence. Le bit Z = Ø (résultat nul) si le dernier résultat est différent de Ø, ou si la dernière comparaison n'a pas donné l'égalité. Aucune action sur le registre d'état.

— Ajouter 1 cycle si le branchement est réalisé.

— En mode émulation (E = 1), ajouter 1 cycle si le branchement est réalisé et si une page mémoire est franchie.

BPL Branchement si positif ou nul — Branch if Result Plus (N = Ø). Si le bit N = Ø on saute à l'adresse indiquée, sinon, on continue en séquence. N = Ø si le dernier résultat est supérieur ou égal à Ø. Aucune action sur le registre d'état.

— Ajouter 1 cycle si le branchement est réalisé.

— En mode émulation (E = 1), ajouter 1 cycle si le branchement est réalisé et si une page mémoire est franchie.

BRA Branchement relatif effectué sur 8 bits — Branch Always. On saute à l'adresse indiquée sans condition. Aucune action sur le registre d'état.

— Ajouter 1 cycle si le branchement est réalisé.

— En mode émulation (E = 1), ajouter 1 cycle si le branchement est réalisé et si une page mémoire est franchie.

BRK Interruption logicielle — Force Break. Met le bit I à 1 et D à Ø, puis simule une interruption. On empile les valeurs du compteur ordinal et du registre d'état puis on saute à l'adresse contenue dans le vecteur d'interruption. Force à Ø le bit D et à 1 le bit I du registre d'état. Deux octets sont occupés par BRK. Sept cycles sont nécessaires en mode 8 bits et huit cycles en mode 16 bits.

BRL Branchement relatif long (16 bits) — Branch Always Long. Effectue la recherche de l'adresse mémoire lors d'un branchement long effectué sur 16 bits ($ØØØØ-$FFFF). Aucune action sur le registre d'état.

BVC Branchement si non dépassement de capacité — Branch on Overflow Clear (V = Ø). Si le bit V = Ø on saute à l'adresse indiquée, sinon, on continue en séquence. Aucune action sur le registre d'état.

— Ajouter 1 cycle si le branchement est réalisé.

— En mode émulation (E = 1), ajouter 1 cycle si le branchement est réalisé et si une page mémoire est franchie.

BVS Branchement si dépassement de capacité — Branch on Overflow Set (V = 1). Si le bit V = 1 on saute à l'adresse indiquée, sinon, on continue en séquence. Aucune action sur le registre d'état.

— Ajouter 1 cycle si le branchement est réalisé.

— En mode émulation (E = 1), ajouter 1 cycle si le branchement est réalisé et si une page mémoire est franchie.

CLC Annulation de la retenue — Clear Carry Flag (C = Ø). On force à Ø le bit de retenue. Surtout employé avant une addition (ADC) pour l'effectuer sans retenue. Force à zéro le bit C du registre d'état.

CLD Annulation du mode décimal — Clear Decimal Mode (D = Ø). On force à Ø le bit D pour positionner l'unité arithmétique en mode binaire. Utilisé pour effectuer ADC et SBC. Force à zéro le bit D du registre d'état.

CLI Autorise les interruptions — Clear Interrupt Disable Bit (I = Ø). On force à Ø le bit I, inhibition des interruptions IRQ, ce qui autorise ces interruptions. Cette instruction est encore utilisée pour la fin des séquences critiques pendant lesquelles on inhibe les interruptions par SEI. Force à zéro le bit I du registre d'état.

CLV Annulation de l'indicateur de débordement — Clear Overflow Flag (V = Ø). Positionne l'indicateur de dépassement de capacité à la valeur Ø. On force à Ø le bit V du registre d'état.

CMP Compare la mémoire à l'accumulateur — Compare Memory and Accumulator. On effectue la soustraction virtuelle entre l'accumulateur et la mémoire. Le résultat n'est pas remis dans l'accumulateur qui reste inchangé. Les indicateurs N, Z et C sont positionnés d'après le résultat. Pour prévoir l'état de N, faire X + complément de la mémoire, où N sera correct s'il n'y a pas de débordement. V reste inchangé.

COP Initialisation d'une séquence d'interruption — Coprocesseur. Deux modes d'interruptions peuvent être gérés par le microprocesseur 65C816, en particulier les interruptions logicielles (Software) contrôlées par l'instruction COP. Les programmeurs sous système ProDOS auront vite fait le rapprochement avec ledit système d'exploitation. L'instruction COP affecte le contenu des registres PB (Program Bank Register) et DB (Data Bank Register). Force à Ø le bit D et à 1 le bit I du registre d'état.

MODE NATIVE (E = 0, mode 16 bits)

Dans ce mode, lors d'une interruption Hardware, le registre PB est purgé et initialisé par des zéros (ØØ). BRK ou COP est exécuté. Le contenu du registre PB sera automatiquement sauvegardé sur le sommet de la pile (8 bits).

MODE ÉMULATION (E = 1, mode 8 bits)

Dans ce mode, lors d'une interruption Hardware, les registres DB et PB sont purgés et des zéros écrits à ces emplacements. BRK ou COP est exécuté. Le contenu de l'adresse du registre PB ne sera pas automatiquement sauvegardé sur la pile.

COP Flag (Signature)

Les valeurs de $ØØ à $7F inclus sont utilisées par le système comme signature, tandis que les valeurs de $8Ø à $FF inclus sont réservées pour une utilisation future (extension à 32 bits avec le microprocesseur 65C832).

Hardware Interrupt – ABORT, IRQ, NMI ou RESET

REGISTRE	OPÉRATION EFFECTUÉE
PC	Opération interne.
PC	Opération interne.
S	Sauvegarde de PB sur la pile.
S	Sauvegarde de PCH sur la pile.
S	Sauvegarde de PCL sur la pile.
S	Sauvegarde de P sur la pile.
VL	Lecture de l'adresse pour la gestion de l'erreur — LByte.
VH	Lecture de l'adresse pour la gestion de l'erreur — HByte.

(suite page 12)

Software Interrupt – BRK, COP

REGISTRE	OPÉRATION EFFECTUÉE
PC-2	Opcode
PC-1	Signature
S	Sauvegarde de PB sur la pile.
S	Sauvegarde de PCH sur la pile.
S	Sauvegarde de PCL sur la pile.
S	Sauvegarde de P sur la pile.
VL	Lecture de l'adresse pour la gestion de l'erreur — LByte.
VH	Lecture de l'adresse pour la gestion de l'erreur — HByte.

Tableau des vecteurs d'interruption

INTERRUPTION	ORIGINE	ÉMULATION (E = 1)	MODE 65C816 (E = Ø)
ABORT	Matériel	$ØØFFF8 / $ØØFFF9	$ØØFFE8 / $ØØFFE9
BRK	Logiciel	$ØØFFFE / $ØØFFFF	$ØØFFE6 / $ØØFFE7
COP	Logiciel	$ØØFFF4 / $ØØFFF5	$ØØFFE4 / $ØØFFE5
IRQ	Matériel	$ØØFFFE / $ØØFFFF	$ØØFFEE / $ØØFFEF
NMI	Matériel	$ØØFFFA / $ØØFFFB	$ØØFFEA / $ØØFFEB
RESET	Matériel	$ØØFFFC / $ØØFFFD	$ØØFFFC / $ØØFFFD (1-E)

CPX Compare la mémoire au registre X — Compare Memory and Index X. Effectue la soustraction virtuelle entre le registre d'index X et la mémoire. Les indicateurs N, Z et C sont positionnés suivant le résultat. Pour prévoir l'état de N, faire X + complément de la mémoire, où N sera correct s'il n'y a pas de débordement. V reste inchangé.

CPY Compare la mémoire au registre Y — Compare Memory and Index Y. Effectue la soustraction virtuelle entre le registre d'index Y et la mémoire, et positionne les indicateurs N, Z et C suivant le résultat. Pour prévoir l'état de N, effectuer Y + complément de la mémoire, où N sera correct s'il n'y a pas de débordement. V reste inchangé.

DEC Décrémentation de la mémoire ou de l'accumulateur — Decrement Memory or Accumulator by One. On diminue de 1 le contenu de la mémoire indiquée. Agit sur les bits N et Z du registre d'état. Pour tous les modes d'adressage utilisant les instructions relatives à Read/Modify/Write, excepté l'adressage accumulateur, les règles suivantes sont à appliquer :

— Ajouter 2 cycles si E = 1 ou E = Ø et M = 1 (8 bits mode) ;

— Ajouter 3 cycles si E = Ø et M = Ø (16 bits mode).

DEX Décrémentation du registre d'index X — Decrement Index X by One. On diminue de 1 le contenu du registre d'index X. Agit sur les bits N et Z du registre d'état.

DEY Décrémentation du registre d'index Y — Decrement Index Y by One. On diminue de 1 le contenu du registre d'index Y. Agit sur les bits N et Z du registre d'état.

EOR OU exclusif — Exclusive OR Memory with Accumulator. On effectue le OU exclusif bit à bit entre l'accumulateur et la mémoire suivant la table de vérité. Agit sur les bits N et Z du registre d'état.

Table de vérité

Ø EOR Ø = Ø
Ø EOR 1 = 1
1 EOR Ø = 1
1 EOR 1 = Ø

INC Incrémentation de la mémoire ou de l'accumulateur — Increment Memory or Accumulator by One. On augmente de 1 le contenu de la mémoire indiquée. Agit sur les bits N et Z du registre d'état. Pour tous les modes d'adressage utilisant les instructions relatives à Read/Modify/Write, excepté l'adressage accumulateur, les règles suivantes sont à appliquer :

— Ajouter 2 cycles si E = 1 ou E = Ø et M = 1 (8 bits mode) ;
— Ajouter 3 cycles si E = Ø et M = Ø (16 bits mode).

INX Incrémentation du registre d'index X — Increment Index X by One. On augmente de 1 le contenu du registre d'index X. Agit sur les bits N et Z du registre d'état.

INY Incrémentation du registre d'index Y — Increment Index Y by One. On augmente de 1 le contenu du registre d'index Y. Agit sur les bits N et Z du registre d'état.

JML Saut inconditionnel long — Jump Long. Cette instruction est équivalente au JMP. JML réalise un branchement à un nouveau vecteur de la mémoire avec une adresse sur 24 bits. Adressage absolu indirect. Aucune action sur le registre d'état.

JMP Saut inconditionnel sur 16 bits — Jump to New Location. On saute à l'adresse indiquée. Effectue un branchement avec un saut inconditionnel suivant le mode d'adressage utilisé. Adresse de 16 bits de large. Aucune action sur le registre d'état.

JSL Saut long à un sous-programme (adresse 24 bits) — Jump to Long. Cette instruction est équivalente au JSR, appel d'un sous-programme. Adressage absolu. Aucune action sur le registre d'état.

JSR Appel d'un sous-programme — Jump to New Location Saving Return Address. On sauve PC (JSR + 2) sur la pile pour mémoriser l'adresse de retour, puis saut à l'adresse indiquée. Aucune action sur le registre d'état.

LDA Chargement de l'accumulateur — Load Accumulator with Memory. On charge l'accumulateur avec la valeur se trouvant à l'emplacement de la mémoire indiquée. La mémoire elle-même n'est pas modifée. Agit sur les bits N et Z du registre d'état.

LDX Charge le registre d'index X — Load Index X. On met dans le registre X le contenu de la mémoire indiquée. La mémoire n'est pas modifiée. Agit sur les bits N et Z du registre d'état.

LDY Charge le registre d'index Y — Load Index Y. On met dans le registre Y le contenu de la mémoire indiquée. La mémoire n'est pas modifiée. Agit sur les bits N et Z du registre d'état.

LSR Décalage d'un bit vers la droite — Shift One Bit Right Memory or Accumulator. On décale d'un bit vers la droite l'accumulateur ou la mémoire. Agit sur les bits Z, C et force à Ø le bit N du registre d'état. Pour tous les modes d'adressage utilisant les instructions relatives à Read/Modify/Write, excepté l'adressage accumulateur, les règles suivantes sont à appliquer :

— Ajouter 2 cycles si E = 1 ou E = Ø et M = 1 (8 bits mode) ;
— Ajouter 3 cycles si E = Ø et M = Ø (16 bits mode).

MVN Déplacement négatif de bloc — Block Move Negative. Cette instruction est utilisée en liaison avec le mode d'adressage Block pour déplacer un bloc de données d'un emplacement mémoire vers un autre. MVN incrémente les registres d'index X et Y utilisés comme compteur de bytes. Le premier byte de l'instruction contient le code de l'instruction. Le second byte contient les 8 bits de poids fort de l'adressage de destination. Le troisième byte contient les 8 bits de poids fort de l'adresse source. Le registre Y contient les 16 bits de poids faible de l'adresse de destination. Le registre X contient les 16 bits de poids faible de l'adresse source. L'accumulateur contient la valeur du nombre de bytes à déplacer, moins un. Le second byte relatif à l'instruction Block Move se trouve localisé dans le registre page de données (Data Bank Register).

MVP Déplacement positif de bloc — Block Move Position. Cette instruction est utilisée en liaison avec le mode d'adressage Block pour déplacer un bloc de données. MVP décrémente les registres d'index X et Y utilisés comme compteur de bytes. L'accumulateur est décrémenté si son contenu a une valeur positive, d'où PC ‹—— PC + 3. Les règles relatives aux registres restent les mêmes que pour l'instruction MVN. *(suite page 14)*

NOP Pas d'opération — No Operation. Instruction sans effet, qui ne fait aucune action mais utilise néanmoins 2 cycles et nécessite un byte. Utilisé après la phase de mise au point de programmes pour remplacer des instructions supprimées. Aucune action sur le registre d'état.

ORA OU logique entre la mémoire et l'accumulateur — OR Memory with Accumulator. On effectue le OU logique bit à bit entre l'accumulateur et la mémoire conformément à la table de vérité. Agit sur les bits N et Z du registre d'état.

PEA Empile l'adresse absolue ou une donnée en mode immédiat — Push Effective Absolute Address on Stack (or Push Immédiate Data on Stack). Empile l'adresse absolue ou une donnée lors de l'adressage immédiat. Se rapporte à toute instruction relative à l'adressage de la pile (empile). Aucune action sur le registre d'état.

PEI Empile l'adresse indirecte — Push Effective Indirect Address on Stack. Ajoute 1 cycle à l'opération si le contenu du registre DL (Direct Low Register) est égal à zéro (DL = Ø). Aucune action sur le registre d'état.

PER Empile l'adresse relative au PC — Push Effective Program Counter Relative Address on Stack. Empile l'adresse relative au compteur ordinal. Aucune action sur le registre d'état.

PHA Empile l'accumulateur (AL ou A) — Push Accumulator on Stack. On place le contenu de l'accumulateur au sommet de la pile et on met à jour le pointeur de pile. A reste intact. Aucune action sur le registre d'état.

PHB Empile le registre page de données — Push Data Bank Register on Stack. On place le contenu du registre page de données au sommet de la pile et on met à jour le pointeur de pile. Aucune action sur le registre d'état.

PHD Empile le registre direct (D) — Push Direct Register on Stack. On place le contenu du registre direct au sommet de la pile et on met à jour le pointeur de pile. Aucune action sur le registre d'état.

PHK Empile le registre page programme — Push Program Bank Register on Stack. On empile le registre page programme et on met à jour le pointeur de pile. Aucune action sur le registre d'état.

PHP Empile le registre d'état (P) — Push Processor Status on Stack. On place le contenu du registre d'état au sommet de la pile et on met à jour le pointeur de pile. P reste intact. Aucune action sur le registre d'état.

PHX Empile le registre d'index X — Push Index X on Stack. On copie le contenu du registre X au sommet de la pile et on met à jour le pointeur de pile. Aucune action sur le registre d'état.

PHY Empile le registre d'index Y — Push Index Y on Stack. On dépose le contenu du registre Y au sommet de la pile et on met à jour le pointeur de pile. Aucune action sur le registre d'état.

PLA Dépile l'accumulateur (AL ou A) — Pull Accumulator from Stack. On transfère vers l'accumulateur le contenu du sommet de la pile et on met à jour le pointeur de pile. Agit sur les bits N et Z du registre d'état.

PLB Dépile le registre page de données — Pull Data Bank Register from Stack. On transfère vers le registre DB le contenu du sommet de la pile et on met à jour le pointeur de pile. Agit sur les bits N et Z du registre d'état.

PLD Dépile le registre direct (D) — Pull Direct Register from Stack. On transfère vers le registre D le contenu du sommet de la pile et on met à jour le pointeur de pile. Agit sur les bits N et Z du registre d'état.

PLP Dépile le registre d'état (P) — Pull Processor Status from Stack. On transfère vers le registre P le contenu du sommet de la pile et on met à jour le pointeur de pile. Agit sur les bits N, V, M, X, D, I, Z et C du registre d'état.

PLX Dépile le registre d'index X — Pull Index X from Stack. On dépose le contenu du sommet de la pile dans le registre d'index X et on met à jour le pointeur de pile. Agit sur les bits N et Z du registre d'état.

PLY Dépile le registre d'index Y — Pull Index Y from Stack. On dépose le contenu du sommet de la pile dans le registre d'index Y et on met à jour le pointeur de pile. Agit sur les bits N et Z du registre d'état.

REP Met à zéro des bits du registre d'état (P) — Reset Status Bits. Permet de positionner à zéro la valeur des bits du registre d'état après un test de bits par exemple (BEQ, BNE, etc.). Agit sur les bits N, V, M, X, D, I, Z et C du registre d'état.

ROL Rotation d'un bit vers la gauche — Rotate One Bit Left — Memory or Accumulator. Décale d'un bit vers la gauche le contenu de l'accumulateur ou de la mémoire. L'ancienne valeur du bit de retenue entre par la droite, tandis que le bit qui sort par la gauche constitue la nouvelle valeur de C. Agit sur les bits N, Z et C du registre d'état. Pour tous les modes d'adressage utilisant les instructions relatives à Read/Modify/Write, excepté l'adressage accumulateur, les règles suivantes sont à appliquer :

— Ajouter 2 cycles si E = 1 ou E = Ø et M = 1 (8 bits mode) ;

— Ajouter 3 cycles si E = Ø et M = Ø (16 bits mode).

ROR Rotation d'un bit vers la droite — Rotate One Bit Right Memory or Accumulator. Décale d'un bit vers la droite le contenu de l'accumulateur ou de la mémoire. L'ancienne valeur entre par la gauche et le bit qui sort par la droite constitue la nouvelle valeur de C. Agit sur les bits N, Z et C du registre d'état. Pour tous les modes d'adressage utilisant les instructions relatives à Read/Modify/Write, excepté l'adressage accumulateur, les règles suivantes sont à appliquer :

— Ajouter 2 cycles si E = 1 ou E = Ø et M = 1 (8 bits mode) ;

— Ajouter 3 cycles si E = Ø et M = Ø (16 bits mode).

RTI Retour d'interruption — Return from Interrupt. Retour de sous-programme ou de routine d'interruption. On récupère sur la pile les contenus de PC (compteur ordinal) et P (registre d'état) qui y avaient été sauvegardés au préalable par le mécanisme d'interruption et on met à jour le pointeur de pile. On reprend l'exécution là où elle avait été laissée lors de l'interruption. Agit sur les bits N, V, M, X, D, I, Z et C du registre d'état.

RTL Retour d'un sous-programme appelé par JSL (24 bits) — Return from Subroutine Long. On récupère sur le sommet de la pile le PC et le contenu de PB lors d'un retour de sous-programme long (adresses de $ØØØØØØ à $FFFFFF). Aucune action sur le registre d'état.

RTS Retour d'un sous-programme appelé par JSR (16 bits) — Return from Subroutine. On récupère sur la pile le PC (compteur ordinal) qui avait été sauvé par le dernier JSR. On reprend l'exécution du programme principal directement derrière l'appel au sous-programme. Aucune action sur le registre d'état.

SBC Soustrait la mémoire avec la retenue à l'accumulateur — Subtract Memory from Accumulator with Borrow. On soustrait de l'accumulateur le contenu de la mémoire indiquée et l'opposé de la retenue (l'emprunt). On opère en mode décimal ou en binaire. Agit sur les bits N, V, Z et C du registre d'état.

SEC Met la retenue à 1 (C = 1) — Set Carry Flag. Est utilisée particulièrement avant SBC pour effectuer une soustraction sans retenue. Force à 1 le bit C du registre d'état.

SED Met en mode décimal (D = 1) — Set Decimal Mode. Met l'unité arithmétique en mode décimal. Force à 1 le bit D du registre d'état (par exemple avant ADC ou SEC).

SEI Met en mode non interruptible (I = 1) — Set Interrupt Disable Status. Inhibe les interruptions. Force à 1 le bit I d'inhibition des interruptions IRQ, ce qui a comme effet de masquer ces interruptions. Si la demande d'interruption est maintenue sur la borne IRQ du microprocesseur (pin IRQ = Ø) l'interruption sera prise en compte dès que le bit I sera remis à zéro. Très utile pour changer le vecteur d'interruption.

SEP Met à 1 des bits du registre d'état (P) — Set Processor Status Bits. Force à 1 les bits du registre d'état. Agit sur les bits N, V, M, X, D, I, Z et C du registre d'état.

STA Sauvegarde de l'accumulateur à un emplacement mémoire — Store Accumulator in Memory. On transfère le contenu de l'accumulateur (AL ou A) dans la mémoire indiquée tandis que l'accumulateur reste inchangé. Aucune action sur le registre d'état.

— Ajouter 1 cycle en mode émulation (E = 1) pour les modes d'adressage suivants :
- Direct indexé par Y ;
- Absolu indexé par X ;
- Absolu indexé par Y.

STP Arrête l'horloge — Stop the Clock. Cette instruction interrompt le circuit d'horloge du microprocesseur. A ce stade, le bus de données reste disponible pour le transfert de données, tandis que le banc d'adresses n'est plus en mode multiplexé (pour le bus de données). Le circuit d'horloge du processeur est relancé en effectuant un RESET du système. Aucune action sur le registre d'état.

STX Sauvegarde de X à un emplacement mémoire — Store Index X in Memory. On transfère le contenu du registre d'index X vers l'emplacement de la mémoire indiquée, tandis que X (XL ou X) reste inchangé. Aucune action sur le registre d'état. *(suite page 16)*

STY Sauvegarde de Y à un emplacement mémoire — Store Index Y in Memory. On transfère le contenu du registre d'index Y vers l'emplacement de la mémoire indiquée, tandis que Y (YL ou Y) reste inchangé. Aucune action sur le registre d'état.

STZ Met à zéro un emplacement mémoire — Store Zero in Memory. Place des zéros à l'emplacement de la mémoire indiquée. Aucune action sur le registre d'état.

— Ajouter 1 cycle en mode émulation (E = 1) pour les modes d'adressage suivants :
- Direct indexé par Y ;
- Absolu indexé par X ;
- Absolu indexé par Y.

TAX Transfert de l'accumulateur dans X — Transfer Accumulator to Index X. On met le contenu de l'accumulateur (AL ou A) dans le registre d'index X (XL ou X). Le contenu de l'accumulateur reste inchangé. Agit sur les bits N et Z du registre d'état.

TAY Transfert de l'accumulateur dans Y — Transfer Accumulator to Index Y. On met le contenu de l'accumulateur (AL ou A) dans le registre d'index Y (YL ou Y). Le contenu de l'accumulateur reste inchangé. Agit sur les bits N et Z du registre d'état.

TCD Transfert de l'accumulateur (C — 16 bits) dans le registre D — Transfert Accumulator to Direct Register. On copie le contenu de l'accumulateur (C) dans le registre direct (D). Le contenu de l'accumulateur reste inchangé. Agit sur les bits N et Z du registre d'état.

TCS Transfert de l'accumulateur (C — 16 bits) dans S — Transfert Accumulator to Stack Pointer Register. On copie le contenu de l'accumulateur (C) dans le registre pointeur de pile (S). A reste inchangé. Aucune action sur le registre d'état.

TDC Transfert de D dans C (accumulateur 16 bits) — Transfer Direct Register to Accumulator. On copie le contenu du registre d'adressage direct (D) dans l'accumulateur (C). D reste inchangé. Agit sur les bits N et Z du registre d'état.

TRB Test et met à zéro des bits — Test and Reset Bit. Teste et met à zéro des bits de la mémoire à l'aide de l'accumulateur.

MODE ÉMULATION (E = 1)

Effectue un AND logique entre le contenu de l'accumulateur (Low Accumulator — registre 8 bits) et l'adresse effective, moins ladite adresse. Agit sur le bit Z du registre d'état.

MODE NATIVE (E = 0)

Effectue un and logique entre le contenu de l'accumulateur (Accumulator — registre 16 bits) et l'expression représentant une adresse effective, moins la valeur arithmétique de ladite expression. Agit sur le bit Z du registre d'état. Pour tous les modes d'adressage de TRB, excepté l'accumulateur, si le bit Z du registre d'état est déterminé ou modifié lors du résultat des opérations A OR B ou A OR W et pour toute commande relative à Read/Modify/Write, la règle suivante est à appliquer :

— Ajouter 2 cycles si E = 1 ou E = Ø et le bit M = 1 (8 bits mode) ;
— Ajouter 3 cycles si E = Ø et le bit M = Ø (16 bits mode).

TSB Teste et met à 1 des bits — Test and Set Bit. Teste et met à 1 des bits de la mémoire à l'aide de l'accumulateur.

MODE ÉMULATION (E = 1)

Effectue un OR logique entre le contenu de l'accumulateur (Low Accumulator — registre 8 bits) et l'adresse effective, moins ladite adresse. Agit sur le bit Z du registre d'état.

MODE NATIVE (E = 0)

Effectue un OR logique entre le contenu de l'accumulateur (Accumulator — registre 16 bits)

et l'expression représentant une adresse effective, moins la valeur arithmétique de ladite expression. Agit sur le bit Z du registre d'état. Pour tous les modes d'adressage de TRB, excepté l'accumulateur, si le bit Z du registre d'état est déterminé ou modifié lors du résultat des opérations A OR B ou A OR W et pour toute commande relative à Read/Modify/Write, la règle suivante est à appliquer :

— Ajouter 2 cycles si E = 1 ou E = Ø et le bit M = 1 (8 bits mode) ;
— Ajouter 3 cycles si E = Ø et le bit M = Ø (16 bits mode).

TSC Transfert de S dans C (accumulateur 16 bits) — Transfer Stack Pointer Register to Accumulator. On copie le pointeur de pile (S) dans l'accumulateur (C). S reste inchangé. Agit sur les bits N et Z du registre d'état.

TSX Transfert de S dans X — Transfer Stack Pointer Register to Index X. On copie le pointeur de pile (S) dans le registre d'index X. S reste inchangé. Agit sur les bits N et Z du registre d'état.

TXA Transfert de X dans l'accumulateur — Transfer Index X to Accumulator. On copie le contenu du registre d'index X dans l'accumulateur, tandis que X reste inchangé. Agit sur les bits N et Z du registre d'état.

TXS Transfert de X dans S — Transfer Index X to Stack Pointer Register. On copie le contenu du registre d'index X dans le registre pointeur de pile (S), tandis que X reste inchangé. Aucune action sur le registre d'état.

TXY Transfert de X dans Y — Transfer Index X to Index Y. On copie le contenu du registre d'index X dans le registe d'index Y, tandis que X reste inchangé. Agit sur les bits N et Z du registre d'état.

TYA Transfert de Y dans l'accumulateur — Transfer Index Y to Accumulator. On copie le contenu du registre d'index Y dans l'accumulateur, tandis que Y reste inchangé. Agit sur les bits N et Z du registre d'état.

TYX Transfert de Y dans X — Transfer Index Y to Index X. On copie le contenu du registre d'index Y dans le registre d'index X, tandis que Y reste inchangé. Instruction complémentaire à TXY. Agit sur les bits N et Z du registre d'état.

WAI Attend une interruption — Wait for Interrupt. Cette instruction met en attente le système pour détecter une interruption sur la broche 2 (RDY — Ready) du microprocesseur 65C816. N'agit sur aucun des bits du registre d'état.

XBA Permute AH avec AL de C — Exchange B and A Accumulator. Permutation de l'octet de poids faible (AL ou A) avec l'octet de poids fort (AH ou B) du registre accumulateur de 16 bits de large (C). La position des bits du registre d'état conditionne l'utilisation du registre AL (Low Byte de l'accumulateur). Agit sur les bits N et Z du registre d'état.

XCE Permute les bits de retenue et d'émulation (C ‹ —— › E) — Exchange Carry and Emulation Bits. Permute la Carry (C) avec le bit d'émulation (E) pour n'importe quelle valeur de E (XH = Ø ; YH = Ø ; SH = 1 ; bit M = 1 ; bit X = 1). Le bit E est interne au microprocesseur et de ce fait non accessible directement par le programmeur. Cette instruction pemet de passer d'un mode d'adressage à un autre (émulation ou native ou inversement). Pour cela, il suffit de positionner la retenue (Carry Flag) et d'exécuter l'instruction XCE. En mode émulation, E = 1, le microprocesseur fonctionne comme le 6502, c'est-à-dire en 8 bits. En mode native, E = Ø, le mode 16 bits est actif. Si le bit 4 (bit X) du registre d'état est mis à zéro, alors les registres d'index sont utilisés en 16 bits. Si le bit 5 (bit M) est positionné à zéro, l'accès à la mémoire pourra se faire en 16 bits. Agit sur le bit C du registre d'état et permute celui-ci avec le bit d'émulation (bit E).

Marcel **COTTINI** continuera, dans les prochains numéros de **TREMPLIN MICRO**, à vous aider à vous familiariser avec le 65C816, mais aussi avec ProDOS, système d'exploitation avec lequel il faudra de plus en plus compter.

UTILITAIRE | DOS 3.3

Marcel COTTINI

DIRECTORY UTILITY

Pour APPLE *II*, *II+*, *IIE*, *IIe+* et *IIc*

Cet utilitaire vous rendra de nombreux services en autorisant la mise en place, dans votre catalogue (directory) de disquette, de certaines dispositions nouvelles. Plusieurs possibilités vous seront offertes, suivant que vous désirerez cacher un nom de programme, le restituer par la suite, ou encore mettre en place des titres pour repérer des sous-programmes éventuels. Placer des lignes d'espace sera un jeu d'enfant. Le nom de vos fichiers textes pourra être modifié pour adopter une présentation sans les paramètres caractéristiques (type de fichier et occupation des secteurs). Seul point noir au tableau : le nom du fichier ne devra pas dépasser la taille de 21 caractères maximum. Cette situation est imposée par la mise en place d'un artifice décalant de huit positions l'emplacement du nom du fichier dans son entrée de la table des matières.

À propos du programme

Lors de l'exécution du programme, un menu sera affiché :

```
(1) Cacher un programme.
(2) Récupérer le programme.
(3) Mise en place titre.
(4) Mise en place fichier/titre
(5) Récupérer fichier/titre
```

Détail de chaque option :

Cacher un programme

Cette option du menu permet la modification du nom d'un programme du type Basic Applesoft, Texte ou Binaire. Dans les deux premiers cas, le nom du programme aura totalement disparu après l'utilisation de l'utilitaire. Dans le dernier cas, utilisation d'un nom représentant un fichier Binaire, celui-ci restera apparent, mais ne pourra plus être exécuté. Cette pratique permet de protéger dans une certaine mesure certains de vos programmes, qui ne pourront être exécutés que par un artifice (consulter à cet effet les lignes 350 à 400).

Récupérer le programme

Option complémentaire de la précédente, permettant de récupérer le nom d'un fichier rendu inopérant par le menu (1). Les lignes 500 à 530 permettent à partir d'un autre programme, d'appeler et d'exécuter un fichier traité par l'option (1).

Mise en place titre

Option permettant de donner un certain cachet à votre table des matières de la disquette. Il vous sera donné à loisir de placer des titres ou sous-titres permettant de classer vos programmes et sous-programmes au niveau du directory. Des lignes d'espace, pour effectuer un travail fini, sont également autorisées. L'option 5 permet de récupérer le nom du titre et de le convertir en nom de fichier pour une éventuelle élimination du directory (DELETE).

Mise en place fichier/titre

Option spécialement étudiee pour traiter les noms des fichiers texte. Elle permet de re ıdre invisible au niveau du directory, tout nom de fich ıer texte. Elle est en somme similaire à l'option (1) qı ı traite les autres types de fichiers.

Récupérer fichier/texte

Option complémentaire à la précédente, elle autorise la récupération du nom du fichier texte traité avec l'option (4).

M. C.

A vos claviers pour effectuer le ménage au niveau de la table des matières de vos disquettes. Avec un peu de maîtrise, cet utilitaire vous permettra de réaliser des protections au niveau de l'accès au nom de vos fichiers, par le simple fait de déplacer l'entrée dans la table des matières. A cet effet, je vous conseille de lire du même auteur, l'ouvrage sur le DOS 3.3, GUIDE DU PROGRAMMEUR, édité par les éditions du SYBEX. Il y figure en bonne place, outre les vecteurs utilisés par le DOS, le détail d'une disquette, en particulier les entrées de la table des matières avec tous les paramètres disponibles. Un livre indispensable dans votre bibliothèque Apple.

NESTOR

```
10 REM ctMctM MENUctM ----ctJctJ
20 CALL 1002: REM REVECTORISE L
   E DOS                          014F
30 ONERR  GOTO 1110               EB13
40 D$ =  CHR$ (13) +  CHR$ (4)    8008
50 CA$ = "":A$ = "":B$ = "":C$
   = "":L$ = ""                   9996
60 TEXT : HOME : INVERSE          2132
70 VTAB 2: PRINT ":"; SPC( 12):
   HTAB 28: PRINT  SPC( 12);":"
   : VTAB 22: PRINT ":"; SPC( 3
   8);":";                        AB3E
80 FOR I = 3 TO 21: VTAB I: HTA
   B 1: PRINT " ";: HTAB 40: PR
   INT " ";: NEXT                 FD2D
90 VTAB 1: HTAB 15: PRINT ": ""
   """""""":": HTAB 15: PRINT "
   "TITREUR: "": HTAB 15: PRINT
   ": """""""""":": NORMAL        9045
100 VTAB 7: HTAB 5                D0DE
110 PRINT "(1) CACHER UN PROGRAM
   ME."                           AC2A
120 VTAB 8: HTAB 5                9FDF
130 PRINT "(2) RECUPERER LE PROG
   RAMME."                        5D20
140 VTAB 10: HTAB 5               0708
150 PRINT "(3) MISE EN PLACE TIT
   RE."                           1407
160 VTAB 12: HTAB 5               430A
170 PRINT "(4) MISE EN PLACE FIC
   HIER/TITRE."                   BE31
180 VTAB 13: HTAB 5               0D0B
190 PRINT "(5) RECUPERER FICHIER
   /TITRE."                       9279
200 VTAB 17: HTAB 5               450F
210 PRINT "VOTRE CHOIX :""        2BA3
220 VTAB 21: HTAB 3               0908
230 PRINT "( ESC ) Pour quitter
   ": VTAB 17: HTAB 20            8EEB
240 GET R$: IF R$ =  CHR$ (27) T
   HEN GOSUB 1120: GOSUB 1120:
   TEXT : HOME : VTAB 5: PRINT
   " Terminé...": END             3D73
250 R =  VAL (R$): IF R < 1 OR R
   > 5 THEN  GOSUB 1120: GOSUB
   1120: GOTO 240                 AC8E
260 ON R GOTO 270,420,580,580,82
   0                              2364
270 REM ctMctM CACHER UN PROGRAM
   MEctM ------------------ctJ
   ctJ
280 HOME : PRINT "NOM DU FICHIER
    A CACHER:": PRINT "---> ";    2545
290 GET A$: IF A$ =  CHR$ (27) T
   HEN 50                         BD09
300 IF CA$ <  > "" AND A$ =  CHR
   $ (13) THEN A$ = CA$: GOTO 3
   50                             43F5
310 IF A$ =  CHR$ (13) THEN  GOS
   UB 1120: GOSUB 1120: GOTO 27
   0                              32E2
320 PRINT A$;                     A75A
330 CA$ = CA$ + A$                174D
340 GOTO 290                      2946
350 PRINT D$;"UNLOCK";A$          C10D
360 B$ = A$                       209B
370 IF  LEN (A$) > 8 THEN A$ =
   MID$ (A$,1,8)                  0FA7
380 FOR I = 1 TO 20 +  LEN (A$):
   L$ = L$ +  CHR$ (8): NEXT I    0B76
390 PRINT D$;"RENAME";B$;",";A$;
   L$                             FCF0
400 PRINT D$;"LOCK";A$;L$         EE15
410 GOTO 50                       7410
420 REM ctMctM RECUPERER LE PROG
   RAMMEctM ------------------
   ---ctJctJ
430 HOME : PRINT "RECUP FICHIER:
   ": PRINT "**************": P
   RINT : PRINT "Nom du fichier
    à récupérer:": PRINT "--->
   ";                             463A
440 GET A$: IF A$ =  CHR$ (27) T
   HEN 50                         BD09
450 IF CA$ <  > "" AND A$ =  CHR
   $ (13) THEN A$ = CA$: GOTO 5
   00                             39F2
460 IF A$ =  CHR$ (13) THEN  GOS
   UB 1120: GOSUB 1120: GOTO 42
   0                              0CDF
470 PRINT A$;                     A75A
480 CA$ = CA$ + A$                174D
490 GOTO 440                      F643
500 B$ = A$                       209B
510 IF  LEN (A$) > 8 THEN A$ =
   MID$ (A$,1,8)                  0FA7
520 IF  LEN (A$) / 2 <  >  INT (
   LEN (A$) / 2) THEN A$ =  MID
   $ (A$,1, LEN (A$) - A)         4162
530 FOR I = 1 TO 20 +  LEN (A$):
   L$ = L$ +  CHR$ (8): NEXT I    0B76
540 PRINT D$;"UNLOCK";A$;L$       C6B8
550 PRINT D$;"RENAME";A$;L$;",";
   B$                             33F0
560 PRINT D$;"LOCK";B$            996B
570 GOTO 50                       7410
```

DIRECTORY UTILITY (suite)

```
580 REM ctMctM PLACER UN TITREct
    M ---------------ctJctJ
590 HOME : PRINT "TITRES et ESPA
    CES:": PRINT "**************
    ****": PRINT : PRINT " - < R
    ETURN > Pour une ligne d'esp
    ace.": PRINT " - < CARACTERE
    S > Pour un titre."                8FC2
600 PRINT                              75BA
610 GOSUB 730                          F94A
620 VTAB 9: HTAB 1: CALL  - 868        B411
630 PRINT "Tapez votre caractère
     de controle: ";: GET C$           3C37
640 IF C$ =  CHR$ (27) THEN 50         4BAE
650 IF C$ <  CHR$ (65) OR C$ >
    CHR$ (90) THEN  GOSUB 1120:
    GOSUB 1120: GOTO 620               1190
660 FOR I = 1 TO 8:L$ = L$ +  CH
    R$ (8): NEXT I                     EEEB
670 IF B$ = "" OR R = 3 THEN A$
    = "TITRE": PRINT D$"BSAVE";A
    $",A$2000,L1": GOTO 700            F67C
680 A$ = B$                            AB9B
690 PRINT D$;"UNLOCK";A$               C10D
700 PRINT D$;"RENAME";A$;",";C$;
    L$;B$                              E692
710 PRINT D$;"LOCK";C$;L$;B$           DBB8
720 GOTO 50                            7410
730 GET W$: IF W$ =  CHR$ (8) TH
    EN CALL  - 1008: PRINT " ";:
    GOTO 790                           EDD5
740 IF W$ =  CHR$ (27) THEN 50         1CC2
750 IF W$ =  CHR$ (13) THEN  CAL
    L - 868: PRINT W$: RETURN          A6AD
760 B$ = B$ + W$                       EBDF
770 PRINT W$;                          9470
780 GOTO 730                           2B45
790 IF B$ = "" THEN 730                0A85
800 PRINT W$;:B$ =  MID$ (B$,1,
    LEN (B$) - 1)                      219E
810 GOTO 730                           2B45
820 REM ctMctM RECUPERER LE TITR
    EctM ----------------ctJct
    J
830 HOME : PRINT "RECUP TITRE:":
    PRINT "************": PRINT
    : PRINT "Nom du fichier aprè
    s récupération:": PRINT "---
    > ";                               F6CB
840 GET A$: IF A$ =  CHR$ (27) T
    HEN 50                             BD09
850 IF CA$ <  > "" AND A$ =  CHR
    $ (13) THEN A$ = CA$: GOTO 9
    00                                 51F6
860 IF A$ =  CHR$ (13) THEN  GOS
    UB 1120: GOSUB 1120: GOSUB 1
    130: GOTO 820                      1E92
870 PRINT A$;                          A75A
880 CA$ = CA$ + A$                     174D
890 GOTO 840                           4E47
900 PRINT : PRINT "Tapez son anc
    ien nom: "                         B6B8
910 GOSUB 1010                         5F72
920 VTAB 9: HTAB 1: CALL  - 868        B411
930 PRINT "Tapez votre caractère
     de controle: ";: GET C$           3C37
940 IF C$ =  CHR$ (27) THEN 50         4BAE
950 IF C$ <  CHR$ (65) OR C$ >
    CHR$ (90) THEN  GOSUB 1120:
    GOSUB 1120: GOTO 920               3B93
960 FOR I = 1 TO 8:L$ = L$ +  CH
    R$ (8): NEXT I                     EEEB
970 PRINT D$;"UNLOCK";C$;L$;B$         AB5B
980 PRINT D$;"RENAME";C$;L$;B$;"
    ,";A$                              1792
990 PRINT D$;"LOCK";A$                 8F6A
1000 GOTO 50                           7410
1010 GET W$: IF W$ =  CHR$ (8) TH
    EN GOTO 1080                       1713
1020 IF W$ =  CHR$ (27) THEN 50        1CC2
1030 IF B$ = "" AND W$ =  CHR$ (1
    3) THEN  GOSUB 1120: GOSUB 1
    120: GOTO 1010                     B168
1040 IF W$ =  CHR$ (13) THEN  RET
    URN                                6509
1050 B$ = B$ + W$                      EBDF
1060 PRINT W$;                         9470
1070 GOTO 1010                         366D
1080 PRINT W$;:B$ =  MID$ (B$,1,
    LEN (B$) - 1)                      219E
1090 GOTO 1010                         366D
1100 REM ctMctM GESTION ERREURctM
     -------------ctJctJ
1110 GOSUB 1120: GOSUB 1120: GOTO
    50                                 7A6C
1120 FOR X = 1 TO 8:XX =  PEEK (4
    9200) -  PEEK (49200): NEXT
    X: RETURN                          CAB9
1130 VTAB 22: HTAB 1: PRINT " Err
    eur d'entrée !!!": PRINT " T
    apez une touche pour continu
    er. ";: GET R$                     E43B
1140 IF R$ =  CHR$ (27) THEN 50        3BBD
1150 RETURN                            63B1
```

Clément RENARD

ANAGRAMMES

RECHERCHE DES ANAGRAMMES DANS LES MOTS DE SIX LETTRES

ANA6

OBJET : Rechercher les anagrammes d'un mot de six lettres dans un fichier binaire.

FONCTIONNEMENT :

Le court programme ci-contre montre que tout est fait par la routine LM, à partir de la ligne 135. Seule la saisie du mot-clé passe par le Basic.

FICHIER : Vous lirez plus loin comment le constituer. La disquette TREMPLIN MICRO vous fournit un fichier de plus de 3000 mots (noms communs).

```
100 LOMEM: 28500: PRINT  CHR$ (4)"BLOAD M6"                    244C
105 TEXT : HOME : PRINT  CHR$ (4)"PR£3": PRINT                 E9EA
110 GOSUB 165                                                  294C
115 PRINT "RECHERCHE DES ANAGRAMMES D'UN MOT DE S
    IX LETTRES"                                                6DBD
120 VTAB 7: CALL  - 958                                        A30E
125 PRINT "MOT DE 6 LETTRES: ......";: POKE 1403,
    18: INPUT "";M$: IF M$ = "" THEN 140                       909C
130 IF  LEN (M$) <  > 6 THEN 120                               1E7F
135 PRINT : CALL 768,M$                                        4BC2
140 PRINT : VTAB 23: PRINT "<1> AUTRE MOT ¨<2> ME
    NU DE DISQUETTE ¨<3> TERMINEctG ";: GET R$: V
    TAB 22: PRINT                                              0A76
145 IF R$ = "1" THEN 120                                       7EBF
150 IF R$ = "2" THEN  PRINT  CHR$ (4)"RUN MENU,D1
    "                                                          5C82
155 IF R$ <  > "3" THEN 140                                    4393
160 HOME : END                                                 8E51
165 FOR I = 768 TO 862: READ R: POKE I,R: NEXT :
    RETURN                                                     9E14
170 DATA 32,190,222,32,227,223,160,2,177,131,153,
    24,0,136,16,248,198,24,169,250,133,6,169,31,1
    33,7,165,6,24,105,6,133,6,144,2,230,7,164,24,
    177,25,153,95,3,136,16,248,160                             744B
175 DATA 5,177,6,240,41,166,24,221,95,3,240,5,202
    ,16,248,48,217,169,0,157,95,3,136,16,232,160,
    0,177,6,9,128,32,237,253,200,192,6,208,244,16
    2,2,32,74,249,240,188,96                                   CE3C
```

LES LIGNES DE DATA

```
300 :  20 BE DE    JSR  $DEBE      ┐ CHKCOM teste la présence de la virgule et
303 :  20 E3 DF    JSR  $DFE3      ┘ PTRGET recherche la variable par son nom.
306 :  A0 02       LDY  £$02       ┐
308 :  B1 83       LDA  ($83),Y    │ On sait lire la longueur de la variable (et son adresse)
30A :  99 18 00    STA  $0018,Y    │ en $83-84. On sauvegarde ces renseingements en
30D :  88          DEY             │ $18-19-1A.
30E :  10 F8       BPL  $308       ┘
310 :  C6 18       DEC  $18          Longueur = longueur — 1.
312 :  A9 FA       LDA  £$FA       ┐
314 :  85 06       STA  $06        │ L'adresse du fichier binaire (— 6)
316 :  A9 1F       LDA  £$1F       │ est installée en $6-7.
318 :  85 07       STA  $07        ┘
```

(Suite page 22)

RECHERCHE DES ANAGRAMMES

```
31A : A5 06     LDA  $06
31C : 18        CLC
31D : 69 06     ADC  £$06      L'adresse du fichier est incrémentée de 6 pour lire les
31F : 85 06     STA  $06       caractères du mot suivant.
321 : 90 02     BCC  $0325
323 : E6 07     INC  $07
325 : A4 18     LDY  $18       Longueur du mot dans Y.
327 : B1 19     LDA  ($19),Y
329 : 99 5F 03  STA  $035F,Y   Le caractère de rang $19-1A + Y dans l'Accumulateur
32C : 88        DEY            est sauvegardé en $35F + Y... et ainsi de suite.
32D : 10 F8     BPL  $0327
32F : A0 05     LDY  £$05      Longueur des mots du fichier (0 à 5 = 6).
331 : B1 06     LDA  ($06),Y   On lit le caractère de rang $6-7 + Y.
333 : F0 29     BEQ  $035E     Si c'est un 0, le fichier est épuisé : sortie.
335 : A6 18     LDX  $18       Longueur du mot que l'on teste dans X.
337 : DD 5F 03  CMP  $035F,X   Comparaison de l'accumulateur avec caractère.
33A : F0 05     BEQ  $0341     Si égal, saut.
33C : CA        DEX            Sinon, X = X — 1 pour contrôler le caractère sui-
33D : 10 F8     BPL  $0337     vant... si la boucle n'est pas terminée.
33F : 30 D9     BMI  $031A     Le mot n'est pas une anagramme : retour au début.
341 : A9 00     LDA  £$00      Le caractère est mis à 0 pour qu'on ne puisse pas l'uti-
343 : 9D 5F 03  STA  $035F,X   liser plusieurs fois.
346 : 88        DEY            Y = Y — 1.
347 : 10 E8     BPL  $0331     Tant que Y ne vaudra pas $FF, on bouclera...
```

AFFICHAGE DU MOT

```
349 : A0 00     LDY  £$00
34B : B1 06     LDA  ($06),Y
34D : 09 80     ORA  £$80      Affichage du mot, caractère par caractère.
34F : 20 ED FD  JSR  $FDED     ORA met le bit 7 à 1 (le fichier binaire contient des
352 : C8        INY            mots enregistrés en ASCII).
353 : C0 06     CPY  £$06
355 : D0 F4     BNE  $034B
357 : A2 02     LDX  £$02      On envoie 2 espaces.
359 : 20 4A F9  JSR  $F94A
35C : F0 BC     BEQ  $031A     Vers un autre test.
35E : 60        RTS            Retour au Basic.
```

Recherche spécialisée

RECH6

OBJET : Rechercher, dans un fichier de mots de 6 lettres, ceux qui comportent tel ou tel caractère à tel rang (ligne 115).

FONCTIONNEMENT :

La saisie du "mot-clé", dans lequel les caractères "nuls" seront remplacés par un tiret, est faite à partir du Basic (ligne 125).

```
100 LOMEM: 28500: PRINT  CHR$ (4)"BLOAD M6"                    244C
105 TEXT : HOME : PRINT  CHR$ (4)"PR£3": PRINT                 E9EA
110 GOSUB 165                                                  294C
115 PRINT "RECHERCHE DES MOTS DE 6 LETTRES CONTEN
    ANT UNE PARTIE D'UN MOT DE 6 LETTRES """"(Sy
    ntaxe: -TOI ou ----UX ou encore -U--I)                     6317
120 VTAB 7: CALL  - 958                                        A30E
125 PRINT "MOT DE 6 LETTRES: ......";: POKE 1403,
    18: INPUT "";M$: IF M$ = "" THEN 140                       909C
130 IF  LEN (M$) > 6 THEN 120                                  E8AE
135 PRINT : CALL 768,M$                                        4BC2
140 PRINT : VTAB 23: PRINT "(1) AUTRE MOT "(2) ME
    NU DE DISQUETTE "(3) TERMINEctG ";: GET R$: V
    TAB 22: PRINT                                              0A76
```

Exemple : — J — — — E permettra d'afficher tous les mots de 6 lettres dont le 2e caractère est un J et le dernier un E : AJUSTE, AJOUTE... par exemple.

REMARQUE : Il est inutile de taper tout le mot quand on ne désire tester que n caractères.

Ex. : — AN fournira la liste des mots contenant AN en 2e et 3e positions

LIMITES : Si vous tapez A et que votre fichier est complet, l'écran va se remplir de tous les mots commençant pas A... C'est sans danger mais pas joli et inutile. Vous pouvez évidemment modifier la DÉMO en Basic pour l'éviter !

```
145 IF R$ = "1" THEN 120                                    7EBF
150 IF R$ = "2" THEN  PRINT  CHR$ (4)"RUN MENU,D1
    "                                                       5C82
155 IF R$ <  > "3" THEN 140                                 4393
160 HOME : END                                              8E51
165 FOR I = 768 TO 847: READ R: POKE I,R: NEXT :
    RETURN                                                  E517
170 DATA 32,190,222,32,227,223,160,2,177,131,153,
    24,0,136,16,248,198,24,169,250,133,6,169,31,1
    33,7,165,6,24,105,6,133,6,144,2,230,7,160,0,1
    77,6                                                    303C
175 DATA 240,36,164,24,177,25,201,45,240,4,209,6,
    208,227,136,16,243,160,0,177,6,9,128,32,237,2
    53,200,192,6,208,244,162,2,32,74,249,240,203,
    96                                                      C9EE
```

LES LIGNES DATA

```
300 :  20 BE DE   JSR   $DEBE
303 :  20 E3 DF   JSR   $DFE3
306 :  A0 02      LDY   £$02
308 :  B1 83      LDA   ($83),Y
30A :  99 18 00   STA   $0018,Y
30D :  88         DEY
30E :  10 F8      BPL   $0308
310 :  C6 18      DEC   $18
312 :  A9 FA      LDA   £$FA
314 :  85 06      STA   $06
316 :  A9 1F      LDA   £$1F
318 :  85 07      STA   $07
```

Voir programme précédent.

Longueur et adresse du "mot" de base sont sauvegardées en $18-1A.
La longueur est décrémentée de 1 car le 0 compte.

L'adresse du fichier binaire des mots de six lettres est installée en $6-7 (adresse — 6).

```
31A :  A5 06      LDA   $06
31C :  18         CLC
31D :  69 06      ADC   £$06
31F :  85 06      STA   $06
321 :  90 02      BCC   $0325
323 :  E6 07      INC   $07
```

Adresse = adresse + 6 pour lire le mot suivant.
Quand il y a retenue,
il faut incrémenter la partie haute ($7).

RECHERCHE

```
325 :  A0 00      LDY   £$00
327 :  B1 06      LDA   ($06),Y
329 :  F0 24      BEQ   $034F
32B :  A4 18      LDY   $18
32D :  B1 19      LDA   ($19),Y
32F :  C9 2D      CMP   £$2D
331 :  F0 04      BEQ   $0337
333 :  D1 06      CMP   ($06),Y
335 :  D0 E3      BNE   $031A
337 :  88         DEY
338 :  10 F3      BPL   $032D
```

Y = 0 (boucle).
Lecture du caractère de rang $6-7 + Y.
Si le caractère lu dans le fichier est 0, c'est fini !
Longueur du "mot" à tester dans Y.
Lecture du caractère de $19-1A + Y.
Est-ce un tiret ?
Si oui, saut.
Est-il égal à celui du mot ($6-7 + Y) ?
Non : il faut passer au mot suivant.

Boucle si Y plus grand ou égal à 0.

AFFICHAGE

```
33A :  A0 00      LDY   £$00
33C :  B1 06      LDA   ($06),Y
33E :  09 80      ORA   £$80
340 :  20 ED FD   JSR   $FDED
343 :  C8         INY
344 :  C0 06      CPY   £$06
346 :  D0 F4      BNE   $033C
348 :  A2 02      LDX   £$02
34A :  20 4A F9   JSR   $F94A
34D :  F0 CB      BEQ   $031A
34F :  60         RTS
```

Affichage des 6 caractères du mot puis envoi de 2 espaces.

Et on passe au mot suivant éventuel.
Retour au Basic.

Saisie (fichier binaire)

DOS 3.3 ou ProDOS
(ProDOS conseillé)

Vous pouvez vous amuser à saisir vos propres mots (à partir d'un dictionnaire des mots croisés... ou de Scrabble). Vous pouvez aussi vous offrir la disquette *TM* (plus de 3000 mots).

Adresse d'origine : Ici c'est $2000 (8192), mais c'est à vous de choisir. Même constation pour LOMEM. Dans notre DÉMO, LOMEM vaut 28500, mais c'est une limite arbitraire.

Fin de fichier : Le fichier se termine par deux mots nuls (douze octets à 0 dans le cas d'un fichier avec mots de 6 lettres).

SAISIE

```
100 TEXT : HOME : LOMEM: 28500                                    DA37
110 PRINT  CHR$ (21)                                              A455
120 PRINT "SAISIE DE MOTS POUR FICHIER BINAIRE"                   7F62
130 PRINT "--------------------------------"                      4B25
140 POKE 34,3:V = 6:L = 6                                         26A1
150 N = 0:AD = 8192 - L                                           F5C6
160 N = N + 1                                                     7F65
170 VTAB V: PRINT "MOT "N". ->  ....."";:H =  PEEK
    (36) + 1 - L: CALL  - 958                                     7A2B
180 VTAB V: HTAB H: INPUT "";M$                                   B7BE
190 IF M$ = "" THEN N = N - 1: GOTO 240                           C9D7
200 IF  LEN (M$) <  > L THEN 180                                  039B
210 AD = AD + L: FOR I = AD TO AD + L - 1:AS =  A
    SC ( MID$ (M$,I + 1 - AD,1)): IF AS < 65 OR A
    S > 90 THEN AD = AD - L: GOTO 180                             33B7
220 POKE I,AS: NEXT I                                             38C7
230 GOTO 160                                                      3942
240 VTAB 22: PRINT "<1> TERMINE <2> ENCORE ";: GE
    T R$: PRINT R$: IF R$ = "1" THEN 160                          27F5
250 IF R$ <  > "2" THEN 240                                       7893
260 AD = AD + L: FOR I = AD TO AD + L + L: POKE I
    ,0: NEXT :LF = AD + L + 5 - 8191                              50D7
270 PRINT  CHR$ (4)"BSAVE M."; STR$ (L);",A$2000,
    L"LF                                                          A50E
```

Cette petite routine est prévue pour saisir des mots de n'importe quelle longueur. Il suffit de modifier la valeur de L (ligne 140). Si vous changez l'adresse d'origine (ligne 150, AD) n'oubliez pas de remplacer le "A$2000" de la ligne 270 par la nouvelle valeur hexa.

ERREUR :

Aucune correction n'est prévue. Le droit à l'erreur n'existe pas ! Il est facile d'ajouter quelques instructions pour se donner le temps de relire chacun des mots rentrés au clavier. De plus, si vous constatez par la suite qu'un mot est erroné, il vous suffira de calculer son adresse, de taper un LOAD FICHIER, de poquer la correction... puis de vous offrir un BSAVE du FICHIER rectifié.

À propos de RECH6 et d'ANA6 :

Des modifications mineures vous permettront d'utiliser ces deux routines pour des recherches avec des mots de 7, 8, 9, 10 lettres et plus. Si vous ne trouvez pas vous-même la solution, adressez-moi une enveloppe timbrée et vous aurez rapidement la réponse. **NESTOR.**

Jean GOUTELLE

EN TOUTES LETTRES

OU LA BONNE ORTHOGRAPHE DES NOMBRES

Orthographier convenablement les grands nombres... lorsqu'il faut les libeller en toutes lettres, n'est pas forcément de tout repos. Les caissiers de nos établissements bancaires en savent quelque chose : que de **vingt** ou de **cent** gratifiés d'un **s**... ou privés de cette marque du pluriel, alors que celle-ci est indésirable... ou au contraire nécessaire.

LE PROGRAMME de Jean GOUTELLE saura rafraîchir votre mémoire défaillante ou pallier votre ignorance. Ou vous répondez et il analyse votre réponse, puis vous donne son verdict, ou vous ne répondez pas et il vous fournit alors la solution du problème posé. Attention ! quand vous écrirez en toutes lettres, n'oubliez pas la mention FRANC(S) !

LES CORRIGÉS proposés sont, en principe, en accord avec le *Dictionnaire des difficultés de la langue française* des Editions Larousse.

```
100 REM **********************
105 REM *LIBELLE D'UNE SOMME*
110 REM * EN TOUTES LETTRES *
115 REM **********************
120 :                                                    003A
125 REM Ce programme traite les entiers positifs de
     0 à 999 999 Billions
130 DIM N$(120),C$(50),CP$(50)                           49BE
135 FOR I = 0 TO 116: READ N$(I): NEXT : FOR I = 1
    TO 38:T$ = T$ + "_": NEXT                            8EFE
140 :                                                    003A
145 REM **********************
150 REM *ENTREE D'UNE SOMME *
155 REM *ET TEST DE VALIDITE*
160 REM **********************
165 :                                                    003A
170 TEXT : HOME : INVERSE : PRINT " CETTE SOMME...
    EN TOUTES LETTRES S.V.P.";: NORMAL : PRINT " "T
    $: POKE 34,2                                         3A61
```

170 Ecriture du nombre (18 chiffres maximum) à transposer.

EN TOUTES LETTRES

SUITE

```
175 FOR I = 1 TO 7:TN$(I) = "":TR$(I) = "": NEXT :S
    F$ = "":DE$ = "": VTAB 2: HTAB 5: PRINT "Ecrive
    z un nombre en chiffres": HTAB 10: PRINT ".....
    ..............."                                    53B0
180 VTAB  PEEK (37): HTAB 10: INPUT "";NC$: PRINT T
    $: IF NC$ = "" THEN 520                            ADD9
185 L =  LEN (NC$):NC =  VAL (NC$): IF NC <  > 0 TH
    EN FOR I = 1 TO L:LX$ =  LEFT$ (NC$,1): IF LX$
    = "0" THEN NC$ =  MID$ (NC$,2): NEXT               7BD2
190 L =  LEN (NC$): FOR I = 1 TO L:LX$ =  MID$ (NC$
    ,I,1): IF LX$ < "0" OR LX$ > "9" THEN 170          46DD
195 NEXT I                                             9DCB
200 :                                                  003A
205 REM ************************
210 REM *SEPARATION EN TRANCHES*
215 REM ************************
220 :                                                  003A
225 NT =  INT (L / 3): IF L < 3 THEN TN$(1) = NC$:T
    R = 1: GOTO 255                                    0D0A
230 FOR I = 1 TO NT:TN$(I) =  MID$ (NC$,L + 1 - I *
    3,3): NEXT :TR = I - 1                             8616
235 IF L / 3 <  >  INT (L / 3) THEN TN$(I) =  LEFT$
    (NC$,L - NT * 3):TR = I                            3CC0
240 IF TR > 6 THEN 170                                 B6B4
245 :                                                  003A
250 REM ***********************
255 REM *ANALYSE DES TRANCHES*
260 REM ***********************
265 :                                                  003A
270 FOR K = TR TO 1 STEP  - 1:TN$ = TN$(K): IF TN$(
    K) = "000" AND NC <  > 0 AND K <  > 5 THEN 290     D08E
275 CT =  VAL ( LEFT$ (TN$,1)):CD =  VAL ( RIGHT$ (
    TN$,2)): GOTO 535                                  88A0
280 SF$ = "": IF CT > 1 AND CD = 0 AND K <  > 2 AND
    K <  > 6 THEN SF$ = "S"                            5C63
285 TR$(K) = N$(100 + CT) + SF$ + " " + N$(CD) + "
    " + N$(110 + K): IF (K > 2 AND K < 6 AND  VAL (
    TN$) > 1) OR (K = 5 AND  VAL (TN$(6)) > 0) THEN
    TR$(K) = TR$(K) + "S"                              9CF1
290 NEXT K                                             A1CD
295 :                                                  003A
300 REM **********************
305 REM *FORMAT SCIENTIFIQUE*
310 REM **********************
315 :                                                  003A
320 CI$ = "": FOR I = 1 TO TR:CI$ =  MID$ (" "" " +
    NC$,L + 4 - 3 * I,3) + " " + CI$: NEXT             7E88
325 SF$ = "": IF NC > 1 THEN SF$ = "S"                 3731
330 VTAB 3: CALL  - 868: HTAB 14 - L / 2: PRINT CI$
    " ";: INVERSE : PRINT "FRANC" + SF$: NORMAL        756A
335 :                                                  003A
```

180 Arrêt du programme sur un simple RETURN.

185 Suppression des zéros en tête du nombre.

190 Si le nombre ne contient pas que des chiffres, rejet.

210 Tranches de 3 chiffres.

225 Cas d'un nombre plus petit que 1000.

235 La première tranche, à gauche, ne contient pas 3 chiffres.

240 S'il y a plus de 18 chiffres, rejet.

270 Analyse de gauche à droite.

280 Pluriel de cent.

285 Traduction littérale, par tranches, complétée par un **s**, sauf pour MILLE.

305 Pour plus de lisibilité du nombre en chiffres.

320 Séparation des tranches de 3 chiffres et formation d'une chaîne.

325 Pluriel de FRANC.

330 Affichage de la chaîne en chiffres à transposer.

360 Sur RETURN seul, pas de traduction proposée. Branchement direct au modèle.

370 Elimination des espaces après FRANC(S).

375 Chaque "mot" de la chaîne est mis dans un tableau CP$( ). NR = NM, nombre de mots de la chaîne.

405 Cas de nombre ⩾ 2 millions comportant six zéros ou plus.

415 Reconstitution d'une chaîne modèle unique, formée avec les éléments de la ligne 285.

420 Cas d'une réponse vide. Chaque mot de la chaîne modèle est transféré dans un tableau C$( ) par GOSUB 585.

450 Cadrage adapté à la longueur de la chaîne. Si le nombre de mots n'est pas le même, FG = 1, affichage sans analyse.

455 Evite les coupures en fin de ligne.

460 Affichage direct du modèle.

470 Non identité des "mots" en INVERSE.

480 Nombre de faute(s) et affichage si la chaîne réponse a le même nombre de mots que le modèle.

```
340 REM ******************
345 REM *CHAINE PROPOSEE*
350 REM ******************
355 :                                                          003A
360 VTAB 4: HTAB 6: PRINT "Ecrivez-le en toutes let
    tres": VTAB 10 - L / 4: CALL  - 958: INPUT "";C
    $:FG = 0: IF C$ = "" THEN FG = 1: VTAB 12: HTAB
    8: FLASH : PRINT "LA REPONSE CORRECTE EST:": NO
    RMAL : PRINT : GOTO 390                                    FFB3
365 VTAB 20: PRINT "D'accord "(O/N) ";: GET R$: PRI
    NT : IF R$ <  > "O" THEN 345                               8E5D
370 C =  LEN (C$): IF  RIGHT$ (C$,1) = " " THEN C$
    = LEFT$ (C$,C - 1): GOTO 370                               7A4A
375 VTAB 15: CALL  - 958: FOR X = 1 TO 50:CP$(X) =
    "": NEXT : HTAB 12: FLASH : PRINT "ANALYSE EN C
    OURS": NORMAL : GOSUB 585: FOR I = 1 TO NM:CP$(
    I) = C$(I): NEXT I:C$ = "":NR = NM                         F9B0
380 :                                                          003A
385 REM ***************
390 REM *CHAINE MODELE*
395 REM ***************
400 :                                                          003A
405 DE$ = "": IF  VAL ( RIGHT$ (NC$,6)) = 0 AND TR
    > 2 AND NC <  > 0 THEN DE$ = "DE "                         F8FF
415 FOR K = TR TO 1 STEP  - 1:C$ = C$ + TR$(K) + "
    ": NEXT :C$ = C$ + DE$ + "FRANC" + SF$:C =  LEN
    (C$)                                                       1613
420 GOSUB 585: IF FG THEN 455                                  EA28
425 :                                                          003A
430 REM ************************
435 REM *ANALYSE DE CONFORMITE*
440 REM *    ET AFFICHAGE     *
445 REM ************************
450 F = 0: VTAB 12: CALL  - 958: PRINT T$: VTAB  PE
    EK (37): HTAB 15: PRINT " Corrigé ": VTAB 19 -
    C / 60: IF NR <  > NM AND  NOT FG THEN  FLASH :
    PRINT "ERREUR D'ECRITURE! VERSION CORRECTE: ":
    NORMAL : CALL  - 198:FG = 1                                D659
455 FOR I = 1 TO NM:M$ =  LEFT$ (C$(I), LEN (C$(I))
    - 1): IF  PEEK (36) +  LEN (M$) > 39 THEN  PRIN
    T                                                          5BE0
460 IF FG THEN  PRINT C$(I);: GOTO 475                         0F79
465 P$ =  LEFT$ (CP$(I), LEN (CP$(I)) - 1): IF M$ =
    P$ THEN  PRINT C$(I);: GOTO 475                            1454
470 INVERSE : PRINT C$(I);: NORMAL :F = F + 1                  E034
475 NEXT I: PRINT :F$ = "": IF F > 1 THEN F$ = "s"             F339
480 IF  NOT FG THEN  POKE 35,24: VTAB 22: PRINT F"
    faute"F$                                                   8B34
485 :                                                          003A
490 REM ***************
495 REM *POUR LA SUITE*
500 REM ***************
505 :                                                          003A
```

(suite au verso)

EN TOUTES LETTRES

SUITE

```
510 POKE 35,24: VTAB 23: PRINT T$;: HTAB 6: VTAB 23
    : INVERSE : PRINT "ESP";: NORMAL : PRINT ": Aut
    re TEST; ou Fin: ";: INVERSE : PRINT "ESC";: NO
    RMAL : GET R$: IF R$ =  CHR$ (32) THEN 150      ED7F
515 IF R$ <  >  CHR$ (27) THEN 495                  0BCA
520 TEXT : HOME : END                               5514
525 :                                               003A
530 REM ******************
535 REM *CAS PARTICULIERS*
540 REM ******************
545 :                                               003A
550 IF NC = 0 THEN TR$(1) = "ZERO": GOTO 305        F51F
555 IF (K = 2 OR K = 6) AND CD = 1 THEN CT = 0:CD =
    0: GOTO 280: REM Cas des Mille et Mille Billion
    s                                               065A
560 IF K = TR AND L <  > 3 * TR THEN CT = 0: GOTO 2
    80: REM 1ère tranche incomplete                 84A4
565 IF NC = 100 THEN TR$(K) = "CENT": GOTO 305: REM
    Cent                                            8784
570 GOTO 280                                        2345
575 :                                               003A
580 REM ************************
585 REM *DECOUPAGE DES CHAINES*
590 REM ************************
595 :                                               003A
600 FOR X = 1 TO 50:C$(X) = "": NEXT X:NM = 1:DR =
    1: FOR I = 1 TO C                               E022
605 L$ =  MID$ (C$,I,1): IF L$ >  = "A" AND L$ <  =
    "Z" THEN DR = 0:C$(NM) = C$(NM) + L$: GOTO 625  066F
610 IF DR AND L$ = " " THEN 625                     F815
615 IF DR AND L$ = "-" THEN C$(NM - 1) =  LEFT$ (C$
    (NM - 1), LEN (C$(NM - 1)) - 1) + L$: GOTO 625  E389
620 IF L$ = " " OR L$ = "-" THEN DR = 1:C$(NM) = C$
    (NM) + L$:NM = NM + 1                           D04C
625 NEXT I: RETURN                                  F5B6
630 :                                               003A
635 REM **********************
640 REM *EN TOUTES LETTRES..*
645 REM **********************
650 :                                               003A
655 DATA "",UN,DEUX,TROIS,QUATRE,CINQ,SIX,SEPT,HUIT
    ,NEUF                                           F452
660 DATA DIX,ONZE,DOUZE,TREIZE,QUATORZE,QUINZE,SEIZ
    E,DIX-SEPT,DIX-HUIT,DIX-NEUF                    FE3B
665 DATA VINGT,VINGT ET UN,VINGT-DEUX,VINGT-TROIS,V
    INGT-QUATRE,VINGT-CINQ,VINGT-SIX,VINGT-SEPT,VIN
    GT-HUIT,VINGT-NEUF                              D39F
670 DATA TRENTE,TRENTE ET UN,TRENTE-DEUX,TRENTE-TRO
    IS,TRENTE-QUATRE,TRENTE-CINQ,TRENTE-SIX,TRENTE-
    SEPT,TRENTE-HUIT,TRENTE-NEUF                    3F83
675 DATA QUARANTE,QUARANTE ET UN, QUARANTE-DEUX,QUA
```

585 Notez que :

1 La chaîne proposée, puis la chaîne modèle sont découpées en éliminant les blancs et en terminant chaque "mot" par un blanc ou par un tiret.

2 L'absence d'un tiret et/ou un tiret illégal dans la chaîne proposée ne sont pas décomptés comme une faute.

NOTE DE NESTOR

Je vous conseille de créer une ligne :

126 TEXT : PRINT CHR$(21) : HOME

car le programme travaille en 40 colonnes. D'autre part, sous **ProDOS**, remplacez **FLASH** par **INVERSE** : c'est aussi joli et sans effet sur l'affichage !

ATTENTION !

Ces lignes sont évidemment très importantes.
Vous contrôlerez facilement la saisie de vos programmes (Basic et LM) en vous offrant, une fois pour toute, la disquette

SIGNATURE

(voir notre bulletin de commande).

```
      RANTE-TROIS,QUARANTE-QUATRE,QUARANTE-CINQ,QUARA
      NTE-SIX,QUARANTE-SEPT,QUARANTE-HUIT,QUARANTE-NE
      UF                                              8F39
680   DATA CINQUANTE,CINQUANTE ET UN,CINQUANTE-DEUX,C
      INQUANTE-TROIS,CINQUANTE-QUATRE,CINQUANTE-CINQ,
      CINQUANTE-SIX,CINQUANTE-SEPT,CINQUANTE-HUIT,CIN
      QUANTE-NEUF                                     8FDF
685   DATA SOIXANTE,SOIXANTE ET UN,SOIXANTE-DEUX,SOIX
      ANTE-TROIS,SOIXANTE-QUATRE,SOIXANTE-CINQ,SOIXAN
      TE-SIX,SOIXANTE-SEPT,SOIXANTE-HUIT,SOIXANTE-NEU
      F                                               547D
690   DATA SOIXANTE-DIX,SOIXANTE ET ONZE,SOIXANTE-DOU
      ZE,SOIXANTE-TREIZE,SOIXANTE-QUATORZE,SOIXANTE-Q
      UINZE,SOIXANTE-SEIZE,SOIXANTE-DIX-SEPT,SOIXANTE
      -DIX-HUIT,SOIXANTE-DIX-NEUF                     4BD7
695   DATA QUATRE-VINGTS,QUATRE-VINGT-UN,QUATRE-VINGT
      -DEUX,QUATRE-VINGT-TROIS,QUATRE-VINGT-QUATRE,QU
      ATRE-VINGT-CINQ,QUATRE-VINGT-SIX,QUATRE-VINGT-S
      EPT,QUATRE-VINGT-HUIT,QUATRE-VINGT-NEUF         9E3C
700   DATA QUATRE-VINGT-DIX,QUATRE-VINGT-ONZE,QUATRE-
      VINGT-DOUZE,QUATRE-VINGT-TREIZE,QUATRE-VINGT-QU
      ATORZE,QUATRE-VINGT-QUINZE,QUATRE-VINGT-SEIZE,Q
      UATRE-VINGT-DIX-SEPT,QUATRE-VINGT-DIX-HUIT,QUAT
      RE-VINGT-DIX-NEUF                               3143
705   DATA "",CENT,DEUX CENT,TROIS CENT,QUATRE CENT,C
      INQ CENT,SIX CENT,SEPT CENT,HUIT CENT,NEUF CENT 2129
710   DATA "",,MILLE,MILLION,MILLIARD,BILLION,MILLE   6F20
```

QUESTION :

J'ai conçu un programme graphique trop long pour se loger entre $800 et $3FFF, adresse après laquelle commence HGR2... aussi est-ce un S.O.S. que je vous lance.

Protection de HGR ou HGR2

Attention ! supprimer le LOMEM éventuel du programme concerné...

RÉPONSE : Il faut LOADer votre programme au-dessus de la page que vous désirez protéger. Pour cela, tapez :

- **HGR** POKE 103,1: POKE 104,64: POKE 16384,0
- **HGR2** POKE 103,1: POKE 104,96: POKE 24576,0

Ensuite, LOAD PROGRAMME ou RUN PROGRAMME (celui-ci est maintenant installé à sa nouvelle adresse).

Vous pouvez automatiser la chose par quelques lignes du type :

```
0 IF (PEEK (103) + PEEK (104) * 256) = 2049 THEN POKE 103,1:
  POKE 104,96: POKE 24576,0: PRINT CHR$(4) "RUN PRO"
10 TEXT : HOME
20 PRINT PEEK (105) + PEEK (106) * 256
```

N'oubliez pas qu'il est possible d'utiliser la pseudo page 3 (voir la collection de *Tremplin Micro*). ■

Votre bibliothèque INFORMATIQUE

par Clément **RENARD**

• MATHÉMATIQUE ET INFORMATIQUE (Cedic/Nathan)

Est-il maintenant l'heure de penser (r)évolution de notre enseignement qui donnerait la priorité aux processus algorithmiques et, à travers eux, à une mathématique plus active, plus vécue : **faire des mathématiques et non en contempler ?** C'est la question que pose l'introduction de ce livre. Une question à laquelle neuf chapitres apportent ensuite une multitude de réponses.

Si vous aimez les mathématiques et pouvez prétendre à un niveau «terminale» ou au-dessus, il est certain que cet ouvrage d'Arthur Engel, adapté par Daniel Reisz, fera votre bonheur.

De nombreux exposés, clairement développés, des exemples et des exercices judicieusement choisis... bref, une bonne méthode de formation, associant mathématiques et informatique !

• DICTIONNAIRE CEGOS (Cedic/Nathan)

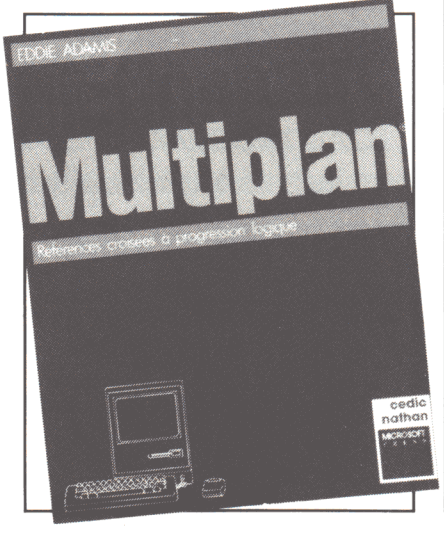
Titre de ce volume : **Définitions du vocabulaire micro-informatique et micro-électrique avec lexique Anglais-Français.** Auteur : Christiane Morvan. Destination : grand public.

Le nombre de mots est relativement limité, mais les notices claires et souvent concises. Le lexique rendra service à celles et à ceux qui ne pratiquent pas l'anglais.

• DICTIONNAIRE MULTIPLAN (Cedic/Nathan)

Ce dictionnaire appartient à une nouvelle collection **références croisées à progression logique**, créée aux Etats-Unis pour Microsoft Press par Eddie Adamis.

Normalement, avec de tels ouvrages, un utilisateur doit être en mesure de trouver la réponse à une question... et cela quelle que soit son expérience. Conception révolutionnaire ? Pas vraiment, mais néanmoins intéressante. En fait, chaque dictionnaire comprend six types d'entrées, toutes classées par ordre alphabétique : commandes, fonctions, opérateurs, valeurs d'erreur, cours d'initiation et tables. Trois répertoires et un index facilitent encore les recherches.



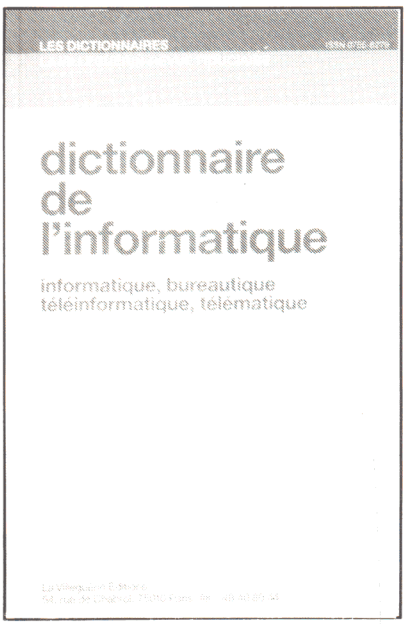
Si vous êtes l'heureux possesseur d'un Mac et si vous rencontrez des difficultés lors de l'utilisation de Multiplan, placez ce dictionnaire à proximité de votre machine. Il complétera avec bonheur votre actuelle documentation.

• DICTIONNAIRE DE L'INFORMATIQUE (Revue Fiduciaire)

Cet ouvrage, réalisé par toute une équipe, sous la direction de Yves-Robert de La Villeguérin, a été conçu pour être un ouvrage de «terrain», envisageant les problèmes techniques sous un angle concret. C'est en tout cas ce que prétendent les deux premières pages de présentation.

Et alors ? Vous vous attendez au pire ? Vous avez tort : le bouquin tient ses promesses, même si ses 185 pages ne sont pas données (185 F)... mais elles en valent bien 300 moins copieuses !

Pour chaque mot : le terme anglais correspondant, une définition (bien rédigée), un développement, des conseils et des références. La Revue fiduciaire, coéditrice de ce dictionnaire n'aura certes pas à en rougir.



DÉCALAGE

Montre bien la rapidité du langage machine.

Le premier élément d'une liste devient le dernier tandis que tous les autres éléments sont également décalés.

SYNTAXE : CALL 768,N,VAR$(Ø)... expression dans laquelle N est égal au nombre réel d'éléments MOINS UN (autrement dit, on ne tient pas compte de la valeur Ø, bien que celle-ci soit utilisée).
VAR$ doit être remplacé par le nom de la variable choisie.

REMARQUE : Il est évidemment possible, après avoir saisi 2Ø mots de donner une valeur inférieure à N (1Ø, par exemple). Dans ce cas, VAR$(Ø) est transféré dans VAR$(1Ø) et tous les contenus intermédiaires sont décalés. Par contre, de VAR$(11) à VAR$(19), la liste n'est pas modifiée.

DÉCALAGE

```
10 TEXT : HOME : PRINT  CHR$ (4)"PR£3": GOSUB 90                                F649
20 DIM M$(400)                                                                  AFDC
30 FOR I = 0 TO 25: FOR J = 1 TO 5:M$(I) = M$(I) +  CHR$ (65 + I): NEXT
   : NEXT                                                                       4A02
35 N = I:L = (N / 2) - 1: CALL  - 198:N = N - 1: GOTO 45                        9272
40 CALL 768,N,M$(0)                                                             86C9
45 HOME : FOR I = 0 TO L: POKE 1403,0: PRINT I" ":M$(I):: POKE 1403,15:
   PRINT 1 + I + L" ":M$(1 + L + I): NEXT                                       D8EB
50 CALL 768,N,M$(0)                                                             86C9
55 VTAB 1: FOR I = 0 TO L: POKE 1403,40: PRINT I" ":M$(I):: POKE 1403,5
   5: PRINT 1 + I + L" ":M$(1 + L + I): NEXT                                    515F
60 VTAB 22: CALL  - 198: PRINT "PRESSEZ UNE TOUCHE (T POUR TERMINER) ":
   : GET R$: PRINT : IF R$ <  > "T" THEN 40                                     93B6
65 VTAB 22: PRINT "<1> ENCORE <2> MENU DE DISQUETTE <3> TERMINE ":: GET
   R$: PRINT                                                                    4825
70 IF R$ = "1" THEN 40                                                          0390
75 IF R$ = "2" THEN  PRINT  CHR$ (4)"RUN MENU,D1"                               5C82
80 IF R$ <  > "3" THEN 65                                                       2969
85 HOME : END                                                                   8E51
90 FOR I = 768 TO 851: READ R: POKE I,R: NEXT : RETURN                          8F12
95 DATA 32,190,222,32,123,221,32,82,231,32,190,222,32,227,223,133,6,132
   ,7,160,2,177,6,153,249,0,136,16,248,198,80,165,80,201,255,208,6,198,
   81,201,255,240,30                                                            2CF0
96 DATA 165,131,24,105,3,133,131,144,2,230,132,160,2,177,131,145,6,136,
   16,249,165,131,133,6,165,132,133,7,208,212,160,2,185,249,0,145,6,136
   ,16,248,96                                                                   9393
```

TRANSMISSION DES PARAMÈTRES

```
300 : 20 BE DE    JSR  $DEBE     CHKCOM teste la présence de la virgule.
303 : 20 7B DD    JSR  $DD7B     FRMEVL évalue l'expression et la met dans FAC.
306 : 20 52 E7    JSR  $E752     GETADR convertit alors FAC en valeur entière
                                 (la valeur de N est dans $50-51).
309 : 20 BE DE    JSR  $DEBE     CHKCOM teste la deuxième virgule et
30C : 20 E3 DF    JSR  $DFE3     PTRGET recherche la variable par son nom.
30F : 85 06       STA  $06       A et Y contiennent la bonne adresse et nous la sauvons
311 : 84 07       STY  $07       dans $6-7 (par exemple).
```

ÉLÉMENT 0 EN ATTENTE

```
313 : A0 02       LDY  £$02      Registre Y mis à 2 pour lecture
315 : B1 06   ┌─  LDA  ($06),Y    et mémorisation des 3 octets contenant
317 : 99 F9 00 │  STA  $00F9,Y    les paramètres de l'élément 0 :
31A : 88       │  DEY             Longueur (1 octet) et adresse (2 octets).
31B : 10 F8   └─  BPL  $0315       La boucle fonctionne pour Y = 2, 1 et 0.
```

COMPTEUR DU NOMBRE D'ÉLÉMENTS À TRAITER

```
31D : C6 50       DEC  $50       Nous retrouvons le nombre d'éléments
31F : A5 50       LDA  $50       dans $50-51 (lire plus haut). Il convient
321 : C9 FF       CMP  £$FF      de décrémenter le compteur. Les décalages
323 : D0 06       BNE  $032B     se poursuivront aussi longtemps que $50-51
325 : C6 51       DEC  $51       n'auront pas pris la valeur $FF,
327 : C9 FF       CMP  £$FF      d'où les tests effectués ici.
329 : F0 1E    •  BEQ  $0349     C'est terminé ? En route vers la fin !
```

ADRESSE DE L'ÉLÉMENT À DÉCALER

```
32B : A5 83       LDA  $83       Chaque nouvelle adresse est égale à l'ancienne
32D : 18          CLC            PLUS 3 (au départ, à la sortie de PTRGET,
32E : 69 03       ADC  £$03      $83-84 contiennent aussi l'adresse 0...
330 : 85 83       STA  $83       et c'est pourquoi nous la retrouvons ici...
332 : 90 02       BCC  $0336     le hasard n'y étant pour rien !).
334 : E6 84       INC  $84       On incrémente quand il y a une retenue.
```

DÉCALAGE

```
336 : A0 02       LDY  £$02      Registre Y mis à 2 pour lire le contenu
338 : B1 83   ┌─  LDA  ($83),Y    des 3 adresses de l'élément à décaler
33A : 91 06   │   STA  ($06),Y    et les mettre à la place de celles
33C : 88      │   DEY            du précédent. Boucle classique en marche
33D : 10 F9   └─  BPL  $0338     arrière et avec BPL comme rétroviseur !
```

PRÉPARATION DU DÉCALAGE SUIVANT

```
33F : A5 83       LDA  $83
341 : 85 06       STA  $06       Sauvegarde, dans $6-7, des adresses
343 : A5 84       LDA  $84       contenues dans $83-84.
345 : 85 07       STA  $07
347 : D0 D4    •  BNE  $031D     Et on repart pour un nouveau décalage.
```

OÙ LE NUMÉRO 0 SE RETROUVE LE DERNIER (OU LE N)

```
349 : A0 02       LDY  £$02      Toujours le même processus
34B : B9 F9 00 ┌─ LDA  $00F9,Y   pour récupérer les paramètres de l'élément
34E : 91 06    │  STA  ($06),Y   zéro et les mettre à la place de ceux
350 : 88       │  DEY            de l'élément N (souvenez-vous : N va de 1 à N,
351 : 10 F8    └─ BPL  $034B     bien que le numéro 0 compte).
353 : 60          RTS            ET COMME IL FAUT BIEN UNE FIN...
```

CATALOGUE SPÉCIAL

QUESTION

Comment — en langage machine bien sûr ! — afficher une ligne verticale de texte sur toute la hauteur d'un écran, en 40 colonnes, en HTAB40, puis la protéger pendant que l'on affiche le catalogue... tout cela sans voir le curseur ?

CAT DÉMO

Essayez cette petite routine (qui ne demande évidemment qu'à être complétée ou modifiée). Elle peut être appelée à partir d'un programme Basic, comme dans notre exemple.

```
10 TEXT : HOME : PRINT  CHR$ (21): PRINT  CHR$ (4)"BLOAD CAT"      6BE7
15 ONERR  GOTO 25                                                  09B7
20 CALL 768: GOTO 35                                               D67E
25 TEXT : HOME : PRINT "METTRE UNE DISQUETTE DANS LE DRIVE 1 SV
   P"                                                              FF3E
30 CALL 797: POKE 216,0: GOTO 15                                   0B66
35 VTAB 24: PRINT "<E>NCORE <M>ENU DE DISQUETTE <T>ERMINEctG ";
   : GET R$: VTAB 22: PRINT                                        B778
40 IF R$ = "E" THEN  RUN 15                                        7452
45 IF R$ = "M" THEN  PRINT  CHR$ (4)"RUN MENU,D1"                  259D
50 IF R$ <  > "T" THEN 35                                          F187
55 HOME : END                                                      8E51
60 REM PROGRAMME LM EN DATA
65 FOR I = 768 TO 834: READ R: POKE I,R: NEXT                      8E28
70 DATA 32,88,252,160,39,162,23,134,37,138,32,193,251,166,37,18
   9,43,3,145,40,202,16,240,136,132,33,32,110,165,44,0,192,16,2
   51,44,16,192,32,57,251,76,88,252                                AADA
75 DATA 212,210,197,205,208,204,201,206,160,205,201,195,210,207
   ,160,206,213,205,197,210,207,160,49,49                          6F58
```

CAT

```
300 :  20 58 FC   JSR  $FC58      HOME pour effacer une première fois.
303 :  A0 27      LDY  £$27       Y ne sera pas détruit par BASCALC (longueur de ligne).
305 :  A2 17      LDX  £$17   ]   Numéro de ligne.
307 :  86 25      STX  $25    ]
309 :  8A         TXA             X passe dans A pour que BASCALC mette en $28-29 l'adresse
30A:  20 C1 FB   JSR  $FBC1   ]   de la ligne de numéro X.
30D:  A6 25      LDX  $25        On récupère X dans $25.
30F :  BD 2B 03   LDA  $032B,X ]  Lecture et affichage d'un caractère de rang $32B + X en $28-29 + Y.
312 :  91 28      STA  ($28),Y ]
314 :  CA         DEX             X = X — 1.
315 :  10 F0      BPL  $0307      Quand on aura X = $FF, on n'ira plus à $307.
317 :  88         DEY         ]   Y = Y — 1 ($26)
318 :  84 21      STY  $21    ]   et cela vaut un POKE 33,39.
31A:  20 6E A5   JSR  $A56E      ROUTINE CATALOG DU DOS 3.3.
31D:  2C 00 C0   BIT  $C000  ]
320 :  10 FB      BPL  $031D  ]   Attente d'une touche et remise à zéro du clavier.
322 :  2C 10 C0   BIT  $C010  ]
325 :  20 39 FB   JSR  $FB39      Mode TEXT.
328 :  4C 58 FC   JMP  $FC58      Effacement final.
```

32B : D4 D2 C5 CD D0 CC C9 CE A0 CD C9 C3 D2 CF A0 CE D5 CD C5 D2 CF A0 31 31 **TREMPLIN MICRO NUMÉRO 11** ☐ **Texte ASCII**

Votre bibliothèque INFORMATIQUE

par Clément **RENARD**

• *LA ROM DE L'APPLE IIc (Sybex - Marcel Cottini)*

L'objectif de cet ouvrage est de fournir au lecteur tous les moyens pour acquérir une maîtrise totale de son Apple IIc. Il traite les nombreux points qui le différencient des modèles précédents. Les principes de base et le Basic Applesoft sont supposés connus.

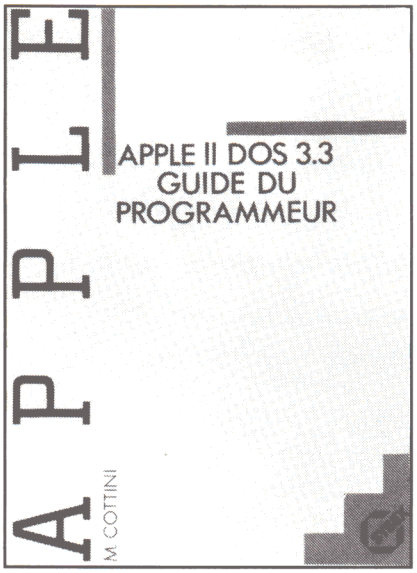
Les points suivants sont traités :

— la compatibilité des programmes,
— la place du IIc dans la gamme Apple *II*,
— les nouveaux moyens de programmation mis à la disposition de l'utilisateur,
— la place qu'occupe le nouveau système d'exploitation ProDOS dans l'environnement Apple,
— les différences fondamentales dans la ROM AppleSoft avec les modèles Apple antérieurs.

Ce livre est le prolongement de l'ouvrage LA ROM DE L'APPLE *II* écrit par le même auteur. Il s'adresse à toutes les personnes désireuses de persévérer dans le domaine de la programmation et qui, en plus du langage Basic qu'elles savent déjà pratiquer, voudraient se familiariser avec ce type de «machine».

Trois volets sont consacrés à l'Apple *IIc*

1. L'Apple *IIc*, dernier micro-ordinateur de la gamme Apple *II*, possède de nombreuses améliorations, tant dans le domaine technique, que dans la programmation. Toute sa structure est passée en revue avec, entre autres, le microprocesseur 65C02, pièce maîtresse, la RAM, la ROM, les nouveaux circuits spécialement conçus et adaptés pour ce type de micro-ordinateur, etc.
2. Le nouveau système d'exploitation ProDOS (Professional Disk Operating System), est largement traité et commenté. Conçu au départ pour la gestion d'un disque dur du type «profile», il fut adapté par la suite pour la série des Apple *II*. Il est fourni gratuitement depuis la commercialisation de l'Apple *IIc*.
3. La ROM AppleSoft, largement commentée dans une édition précédente, est listée au niveau des instructions différentes de la version antérieure (Apple *II* à *IIe*). Certaines routines sont nouvelles, d'autres ont été modifiées ou supprimées. Un chapitre à ne pas négliger, devenant une nécessité pour ceux qui programment en langage machine. Tous les détails sont fournis pour une meilleure compréhension de certains mécanismes de compatibilité.



• *DOS 3.3 GUIDE DU PROGRAMMEUR (Sybex -Marcel Cottini)*

Ce livre n'a pas pour objet de plagier le Manuel du DOS, document livré avec l'achat d'un ensemble Apple *II* et lecteur de disquettes, mais se veut d'une vocation beaucoup plus technique.
Il n'est pas question pour l'auteur de réécrire le Manuel du DOS édité par Apple Computer, d'ailleurs très bien conçu pour le démarrage d'une configuration standard. L'idée recherchée par cet ouvrage est d'apporter toute l'aide nécessaire au débutant qui voudrait en connaître un peu plus sur les mécanismes du fonctionnement interne des routines du DOS. Beaucoup de choses ont été dites et écrites sur les systèmes d'exploitation, en particulier le DOS 3.3, dernière version de cette série. Dans un passé plus récent, la sortie sur le marché de ProDOS a considérablement changé la part du marché offert aux systèmes d'exploitation. Ce nouveau SED permet entre autres, une rapidité accrue de la gestion des fichiers ainsi qu'un traitement de fichiers d'une taille plus importante (taille maximale : 16 mégabytes). Par ailleurs, il peut traiter des périphériques de sauvegarde d'une capacité de 32 mégabytes, disque dur du type profile par exemple.

Mais alors, pourquoi acheter un livre sur le DOS 3.3 ? Pour une raison bien simple : tout n'a certainement pas été dit sur ce système... et cet ouvrage le prouve ! Beaucoup de programmeurs chevronnés ont acquis une expérience et une maîtrise toute particulière dans la pratique de l'ordinateur, en particulier la série des Apple *II*. Mais bon nombre ont jalousement gardé le fruit de leurs découvertes et beaucoup d'astuces sont restées cachées à ce jour.

L'auteur, passionné inconditionnel de l'Apple *II*, possède une grande expérience dans la pratique et la technique de l'utilisation du DOS 3.3. Désireux d'en faire bénéficier le maximum d'amateurs de la **pomme**, le projet d'un ouvrage destiné aux lecteurs ayant des notions de base en programmation prit corps. Le résultat se concrétisa par la rédaction d'un manuscrit traitant la pratique du DOS 3.3 sur Apple *II*.

A qui s'adresse ce livre ? A tous ceux qui désirent aller de l'avant dans l'informatique grand public, car il ne faut pas oublier que les amateurs d'aujourd'hui sont les professionnels de demain. Rien ne se crée tout seul. Pour chaque découverte et mise au point, il faut cette impulsion qui fait émerger l'inconnu d'hier. **DOS 3.3, guide du programmeur** est l'ingrédient qui permettra à bon nombre de débutants en programmation, de faire surface et de découvrir le côté caché de son Apple.

Maurice CHAVELLI

SIMPLIFIEZ la saisie avec GLOSSAIRE

POURQUOI CE PROGRAMME ?

Combien de fois tapez-vous LIST, INPUT, HOME, CATALOG et autres CALL — 151 sur votre clavier ? Le présent programme va vous permettre d'économiser vos forces. Il permet de créer un glossaire pouvant contenir jusqu'à 64 définitions. Une définition est composée d'un caractère d'appel et d'une série limitée à 255 caractères. L'utilisation du glossaire est simple : l'action conjuguée de la "pomme ouverte" et du caractère d'appel entraîne la prise en compte de la série de caractères correspondante comme si elle était entrée au clavier.

QUELQUES PRÉCISIONS UTILES

Attention ! Chaque définition est validée par la touche ESC, et les caractères sont pris en compte JUSQU'AU CURSEUR seulement. Le programme accepte les caractères de contrôle, en particulier le célèbre CTRL-M (RETURN). Vous pouvez ainsi prévoir plusieurs commandes dans une définition, et notamment :

HOME CTRL-M
CATALOG CTRL-M

Le programme vous permet une gestion complète : le glossaire créé est sauvegardé sous forme de fichier binaire.

L'UTILISATION

Pour utiliser un glossaire il vous suffit d'un simple BRUN :

— le glossaire est alors chargé provisoirement en $2000 (adieu la page 1 HGR !) ;

— une interface est casée en $300 (donc attention à la page 3) ;

— DOS ou ProDOS est modifié ;

— le bloc de traitement et les définitions sont stockés en mémoire auxiliaire à l'adresse $4000 ; donc attention à la commande /RAM de ProDOS ! Je n'ai pas jugé utile d'interdire les *Blocks* correspondants : $19 et suivants. Si quelqu'un est tenté, la table d'occupation des *Blocks* (Volume Bit-Map pour les initiés) est en $C00 de la mémoire auxiliaire. Chaque bloc libre est représenté par un bit qui est à 1 (0 quand il est occupé).

LA ROUTINE EN LANGAGE MACHINE

Elle figure sous forme de DATA dans le programme BASIC. Une version source (ProCODE) commentée vous aidera à y voir plus clair...

GLOSSAIRE de Maurice Chavelli

```
 1 DATA "(C)harger","(S)auver","(N)ouveau","(A)jouter","(E)ffacer","(
   L)ister","(Q)uitter"                                                    949A
 2 FOR C = 1 TO 7: READ A$: NEXT                                           079F
 3 REM ------------------
 4 REM Routine Assembleur
 5 REM ------------------
 6 DATA 169,0,133,60,169,33,133,61,169,0,133,62,169,0,133,63,169,0,13
   3,66,169,64,133,67,56,32,17,195,162,33,189,68,32,157,0,3,202,16,24
   7,169,255,133,24,173,209,3,240,11                                       1C65
 7 DATA 169,0,141,85,170,169,3,141,86,170,96,141,201,154,169,3,141,20
   2,154,96,36,24,16,24,32,27,253,44,97,192,48,1,96,160,0,140,237,3,1
   60,64,140,238,3,56,184,76,20,195,160,45,208,239                         3E1C
 8 DATA 160,0,132,7,133,249,185,0,65,197,249,240,10,200,200,200,200,2
   08,243,165,249,208,37,169,0,133,24,200,185,0,65,133,250,200             397B
 9 DATA 185,0,65,133,8,200,185,0,65,133,9,164,7,177,8,200,132,7,196,2
   50,208,4,160,255,132,24,160,12,140,237,3,160,3,140,238,3,24,184,76
   ,20,195                                                                 AE40
10 FOR C = 8192 TO 8291: READ A%: POKE C,A%: NEXT : FOR C = 8448 TO 8
   522: READ A%: POKE C,A%: NEXT                                           298B
11 REM ---------------
12 REM Début Programme
13 REM ---------------
14 LOMEM: 24576:O = 8704: GOSUB 120                                        4E55
15 ONERR  GOTO 136                                                         7CEA
16 HOME : PRINT                                                            5A8B
17 HTAB 10: PRINT "Gestion de glossaire"                                   0CBA
18 HTAB 10: PRINT "--------------------"                                   FCB3
19 PRINT : PRINT                                                           15AE
20 RESTORE : FOR C = 1 TO 7: READ A$: HTAB 15: PRINT A$: PRINT : NEXT      A60A
21 PRINT : HTAB 10: PRINT "<ESP> => Catalogue"                             C2D5
22 WAIT  - 16384,128: GET A$                                               61A8
23 IF A$ = "N" THEN  PRINT : PRINT "Confirmation nouveau glossaire? "
   ;: GET A$: IF A$ = "O" THEN  GOSUB 120: GOTO 37                         FCBB
24 IF A$ = "A" THEN 36                                                     2E94
25 IF A$ = "E" THEN 66                                                     3D9B
26 IF A$ = "L" THEN 87                                                     2CA5
27 IF A$ = "C" THEN 103                                                    9DC1
28 IF A$ = "S" THEN 113                                                    31D2
29 IF A$ = " " THEN  PRINT : HOME : PRINT  CHR$ (4)"CAT": WAIT  - 163
   84,128: GET A$: GOTO 16                                                 F83F
30 IF A$ = "Q" THEN  PRINT : PRINT "Confirmation Quitter? ";: GET A$:
   IF A$ = "O" THEN  HOME : END                                            92BF
31 CALL  - 198: GOTO 16                                                    9843
32 REM ------------------
33 REM Traitement AJOUTER
34 REM ------------------
35 N = N + 1                                                               7F65
36 IF N > 64 THEN  HOME : FLASH : VTAB 10: HTAB 12: PRINT "PLUS DE PL
   ACE!": GOTO 136                                                         C722
37 HOME : PRINT "_____________________________________________";           43E2
```

```
38 INVERSE : PRINT " ";: HTAB 40: PRINT " ";                                  54F8
39 PRINT " ";: NORMAL : PRINT " Numéro: ";N;: HTAB 20: PRINT "Mémoir
   e: ";M - 0 + 256;: HTAB 40: INVERSE : PRINT " ";                           B6B9
40 PRINT " ";: NORMAL : PRINT "________________________________________
   ";: INVERSE : PRINT " ": NORMAL                                            5EC4
41 VTAB 10: PRINT "Touche utilisée? ";: GET A$: IF  ASC (A$) = 27 THE
   N 16                                                                       959B
42 FOR C = 0 TO 0 + 252 STEP 4                                               804F
43 IF  PEEK (C) =  ASC (A$) + 128 THEN  CALL  - 198: PRINT "Déjà util
   isée!": FOR C = 1 TO 1000: NEXT : GOTO 37                                  7095
44 IF  PEEK (C) = 0 THEN 46                                                  B351
45 NEXT                                                                      0582
46 IF  ASC (A$) < 27 THEN  INVERSE : PRINT  CHR$ ( ASC (A$) + 64);: N
   ORMAL : GOTO 48                                                           3D42
47 PRINT A$                                                                  D11F
48 POKE 0 + (N - 1) * 4 + 3, INT (M / 256) + 31: POKE 0 + (N - 1) * 4
   + 2,M - 256 * ( INT (M / 256))                                            7901
49 POKE 0 + (N - 1) * 4, ASC (A$) + 128: PRINT : PRINT : PRINT "Texte
    ? ";                                                                     4DF6
50 X = 0                                                                     F358
51 GET A$                                                                    9923
52 IF  ASC (A$) = 27 AND X = 0 THEN  GOTO 16                                 BE7D
53 IF  ASC (A$) = 27 THEN  POKE 0 + (N - 1) * 4 + 1,X:M = M + X: GOTO
   35                                                                        A73B
54 IF  ASC (A$) = 8 THEN 60                                                  5D7B
55 IF  ASC (A$) = 21 THEN 62                                                 F2A8
56 POKE M + X, ASC (A$) + 128: IF  ASC (A$) < 27 THEN  INVERSE : PRIN
   T CHR$ ( ASC (A$) + 64);: NORMAL : GOTO 58                                29CE
57 PRINT A$;                                                                 A75A
58 X = X + 1: IF X < 256 THEN 51                                             BF50
59 CALL  - 198                                                               DAF7
60 CALL  - 1008:X = X - 1: IF X >  = 0 THEN 51                               4E0A
61 CALL  - 198                                                               DAF7
62 CALL  - 1036: GOTO 58                                                     BF71
63 REM ------------------
64 REM Traitement EFFACER
65 REM ------------------
66 HOME : HTAB 8: PRINT "Effacement de définition"                           548D
67 HTAB 8: PRINT "------------------------"                                  243E
68 PRINT : PRINT : PRINT "Lettre clef? ";: GET A$                            30E7
69 IF  ASC (A$) = 27 THEN 16                                                 09AD
70 FOR C = 0 TO 0 + 252 STEP 4                                               804F
71 IF  PEEK (C) =  ASC (A$) + 128 THEN  PRINT : PRINT : GOTO 73              D4B3
72 NEXT : CALL  - 198: PRINT "Introuvable!": FOR C = 1 TO 1000: NEXT
   : GOTO 66                                                                 5DE5
73 GOSUB 124: PRINT : PRINT                                                  FB2F
74 PRINT "On efface? ";: GET A$: IF A$ <  > "O" THEN 66                      7AFB
75 VTAB 16: HTAB 11: FLASH : PRINT "UN PEU DE PATIENCE!": NORMAL             2A03
76 L =  PEEK (C + 1)                                                         6C8B
77 FOR D = AD TO M - L: POKE D, PEEK (D + L): NEXT                           D8E7
78 M = M - L:N = N - 1                                                       801F
79 I = (N - (C - 0) / 4 - 2) * 4: IF I =  - 4 THEN D = C: GOTO 83            1C8D
```

GLOSSAIRE de Maurice Chavelli

```
 80 FOR D = C TO C + I STEP 4                                                22E8
 81 AD =  PEEK (D + 6) +  PEEK (D + 7) * 256:AD = AD - L: POKE D + 3,
    INT (AD / 256): POKE D + 2,AD -  PEEK (D + 3) * 256: POKE D + 1, P
    EEK (D + 5): POKE D, PEEK (D + 4)                                        3531
 82 NEXT D                                                                   87C6
 83 POKE D,0: GOTO 66                                                        5AAA
 84 REM -----------------
 85 REM Traitement LISTER
 86 REM -----------------
 87 HOME :Z = 0                                                              EE2B
 88 FOR C = 0 TO 0 + 252 STEP 4                                              804F
 89 IF  PEEK (C) <  > 0 THEN 91                                              7921
 90 PRINT : PRINT "RETOUR MENU ( )";: HTAB 14: GET A$: GOTO 16               F5D2
 91 IF  PEEK ( - 16384) < 128 THEN 94                                        7F4C
 92 GET A$: IF  ASC (A$) = 27 THEN 16                                        F10A
 93 IF A$ = " " THEN Z = 1                                                   EC65
 94 IF Z = 0 THEN 98                                                         AF3C
 95 WAIT  - 16384,128                                                        EC4B
 96 GET A$: IF  ASC (A$) = 27 THEN 16                                        F10A
 97 IF  ASC (A$) = 13 THEN Z = 0                                             F69B
 98 GOSUB 124                                                                0F47
 99 NEXT C: GOTO 90                                                          ED13
100 REM ------------------
101 REM Traitement CHARGER
102 REM ------------------
103 PRINT : INPUT "CHARGER => Quel nom? ";A$                                 58B3
104 PRINT  CHR$ (4)"BLOAD";A$;",A$2000"                                      C33E
105 N = 1                                                                    3D4F
106 FOR C = 0 TO 0 + 255 STEP 4: IF  PEEK (C) = 0 THEN 108                   590C
107 N = N + 1: NEXT                                                          7B21
108 M =  PEEK (C - 2) + ( PEEK (C - 1) - 31) * 256 +  PEEK (C - 3)           15E5
109 GOTO 16                                                                  4812
110 REM -----------------
111 REM Traitement SAUVER
112 REM -----------------
113 POKE 8205, INT (M / 256): POKE 8201,M - 256 *  PEEK (8205)               03F6
114 PRINT : INPUT "SAUVER => Quel nom? ";A$                                  7E8D
115 PRINT  CHR$ (4)"BSAVE";A$;",A$2000,L";M - 0 + 512                        0BC5
116 GOTO 16                                                                  4812
117 REM ---------------
118 REM Initialisations
119 REM ---------------
120 FOR C = 0 TO 0 + 252 STEP 4: POKE C,0: NEXT :N = 1:M = 0 + 256: RE
    TURN                                                                     671C
121 REM ------------------
122 REM Lecture définition
123 REM ------------------
124 IF  PEEK (C) < 155 THEN  INVERSE : PRINT  CHR$ ( PEEK (C) + 64);:
    NORMAL : GOTO 126                                                        4355
125 PRINT  CHR$ ( PEEK (C));                                                 D6A3
```

```
126 PRINT " => ";                                                    5AF4
127 AD =  PEEK (C + 2) + ( PEEK (C + 3) - 31) * 256                  E6E3
128 FOR D = 0 TO  PEEK (C + 1) - 1                                   9FEF
129 IF  PEEK (AD + D) < 155 THEN  INVERSE : PRINT  CHR$ ( PEEK (AD + D
    ) + 64);: NORMAL                                                 F673
130 PRINT  CHR$ ( PEEK (AD + D));                                    D5F1
131 NEXT D: IF  PEEK (AD + D - 1) = 141 THEN  CALL  - 998            4A94
132 PRINT : RETURN                                                   FEA5
133 REM -----------------
134 REM Traitement erreur
135 REM -----------------
136 FOR C = 1 TO 80:S =  PEEK ( - 16336): NEXT : NORMAL : GOTO 16     6E29
```

Assemblage par ProCODE

```
1   IND       EQU  $7
2   ADD       EQU  $8
3   FLAG      EQU  $18
4   A1        EQU  $3C
5   A2        EQU  $3E
6   A4        EQU  $42
7   SVA       EQU  $F9
8   LONG      EQU  $FA
9   TABLE     EQU  $4100
10  PB1       EQU  $C061
11  AUXMOVE   EQU  $C311
12  XFER      EQU  $C314
13  KEYIN     EQU  $FD1B
14  MOVE      EQU  $FE2C
15            ORG  $2000
16  * Transfert table en mémoire auxiliaire
2000: A9 00   17            LDA  £0          ; paramètres pour AUXMOVE
2002: 85 3C   18            STA  A1          ; départ
2004: A9 21   19            LDA  £$21
2006: 85 3D   20            STA  A1+1
2008: A9 00   21            LDA  £0          ; fin ajustée par le BASIC
200A: 85 3E   22            STA  A2
200C: A9 00   23            LDA  £0
200E: 85 3F   24            STA  A2+1
2010: A9 00   25            LDA  £0          ; destination
2012: 85 42   26            STA  A4
2014: A9 40   27            LDA  £$40
2016: 85 43   28            STA  A4+1
2018: 38      29            SEC
2019: 20 11 C3 30           JSR  AUXMOVE     ; tranfert
              31  * Transfert interface en $300
201C: A2 21   32            LDX  £33
201E: BD 44 20 33 BCLX      LDA  DEBUT,X
2021: 9D 00 03 34           STA  $300,X
2024: CA      35            DEX
2025: 10 F7   36            BPL  BCLX
              37  * Initialisation FLAG
2027: A9 FF   38            LDA  £$FF
2029: 85 18   39            STA  FLAG
```

LA ROUTINE EN LANGAGE MACHINE

Ce programme source est donné sur la disquette *Tremplin Micro*.

(Suite page 40)

GLOSSAIRE de Maurice Chavelli

```
                        40  * Vectorisation SED en discriminant DOS et PRODOS
202B: AD D1 03          41          LDA  $3D1        ; DOS ou PRODOS?
202E: F0 0B             42          BEQ  PRODOS
2030: A9 00             43          LDA  £0          ; c'est le DOS!
2032: 8D 55 AA          44          STA  $AA55       ; petite magouille...
2035: A9 03             45          LDA  £3
2037: 8D 56 AA          46          STA  $AA56
203A: 60                47          RTS
203B: 8D C9 9A          48  PRODOS  STA  $9AC9       ; c'est PRODOS!
203E: A9 03             49          LDA  £3          ; une autre magouille...
2040: 8D CA 9A          50          STA  $9ACA
2043: 60                51          RTS
                        52  * Interface qui sera logée en $300
2044: 24 18             53  DEBUT   BIT  FLAG        ; définition en cours?
2046: 10 18             54          BPL  ENCORE      ; oui
2048: 20 1B FD          55          JSR  KEYIN       ; envoie le caractère
204B: 2C 61 C0          56          BIT  PB1         ; on appuie sur "pomme ouverte"?
204E: 30 01             57          BMI  POMME       ; oui
2050: 60                58  RETOUR  RTS              ; non, alors retour
2051: A0 00             59  POMME   LDY  £0
2053: 8C ED 03          60  POMME1  STY  $3ED       ; paramètres pour XFER
2056: A0 40             61          LDY  £$40
2058: 8C EE 03          62          STY  $3EE
205B: 38                63          SEC
205C: B8                64          CLV              ; donne la main au bloc de trai-
205D: 4C 14 C3          65          JMP  XFER        ; te;ent en $4000 de la mém. aux
2060: A0 2D             66  ENCORE  LDY  £$2D        ; ici c'est en $402D sur ENCORE1
2062: D0 EF             67          BNE  POMME1
                        68  * Bloc de traitement
                        69          ORG  $2100
2100: A0 00             70          LDY  £0
2102: 84 07             71          STY  IND
2104: 85 F9             72          STA  SVA
2106: B9 00 41          73  CHERCHE LDA  TABLE,Y     ; carac. d'appel dans la table ?
2109: C5 F9             74          CMP  SVA
210B: F0 0A             75          BEQ  TROUVE
210D: C8                76          INY
210E: C8                77          INY
210F: C8                78          INY
2110: C8                79          INY
2111: D0 F3             80          BNE  CHERCHE
2113: A5 F9             81          LDA  SVA
2115: D0 25             82          BNE  FIN         ; pas trouvé!
2117: A9 00             83  TROUVE  LDA  £0          ; trouvé...
2119: 85 18             84          STA  FLAG        ; marque FLAG
211B: C8                85          INY
211C: B9 00 41          86          LDA  TABLE,Y     ; récupère la long.de définition
211F: 85 FA             87          STA  LONG
2121: C8                88          INY
2122: B9 00 41          89          LDA  TABLE,Y     ; récupère adr. de la définition
2125: 85 08             90          STA  ADD
```

```
2127: C8          91            INY
2128: B9 00 41    92            LDA  TABLE,Y
212B: 85 09       93            STA  ADD+1
212D: A4 07       94  ENCORE1   LDY  IND
212F: B1 08       95            LDA  (ADD),Y     ; récupère un caractère
2131: C8          96            INY              ; avance d'un cran
2132: 84 07       97            STY  IND
2134: C4 FA       98            CPY  LONG        ; fini?
2136: D0 04       99            BNE  FIN         ; non
2138: A0 FF      100            LDY  £$FF        ; oui -> marque FLAG
213A: 84 18      101            STY  FLAG
213C: A0 0C      102  FIN       LDY  £$C         ; paramètres pour XFER
213E: 8C ED 03   103            STY  $3ED
2141: A0 03      104            LDY  £3
2143: 8C EE 03   105            STY  $3EE
2146: 18         106            CLC
2147: B8         107            CLV
2148: 4C 14 C3   108            JMP  XFER        ; donne la main à l'interface
```

Table des symboles ordre alphabétique

```
  -A1........$003C    -A2........$003E    -A4........$0042    -ADD.......$0008
  -AUXMOVE...$C311    -BCLX......$201E    -CHERCHE...$2106    -DEBUT.....$2044
  -ENCORE....$2060  ? -ENCORE1...$212D    -FIN.......$213C    -FLAG......$0018
  -IND.......$0007    -KEYIN.....$FD1B    -LONG......$00FA  ? -MOVE......$FE2C
  -PB1.......$C061    -POMME.....$2051    -POMME1....$2053    -PRODOS....$203B
? -RETOUR....$2050    -SVA.......$00F9    -TABLE.....$4100    -TROUVE....$2117
  -XFER......$C314
```

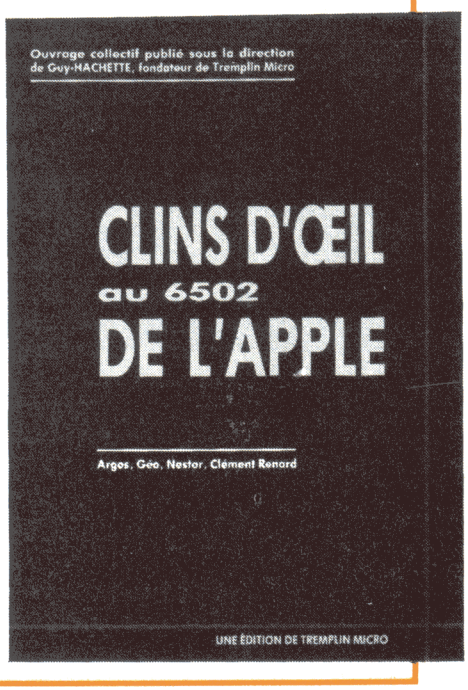

Une édition à ne pas manquer !

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

La rapidité du langage machine — celui que votre Apple comprend le mieux — est proprement stupéfiante.

CLINS D'ŒIL au 6502 DE L'APPLE

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



Votre bibliothèque INFORMATIQUE

par Clément **RENARD**

• *EXCEL Tome 1 : Le tableur (Cedic/Nathan)*

Les adeptes de Macintosh connaissent sans doute EXCEL, le nouveau tableur multi-fonctions intégrant graphiques, base de données et macros... un logiciel capable de rivaliser non sans succès avec LOTUS 1-2-3, tellement prisé par les possesseurs du PC d'IBM.

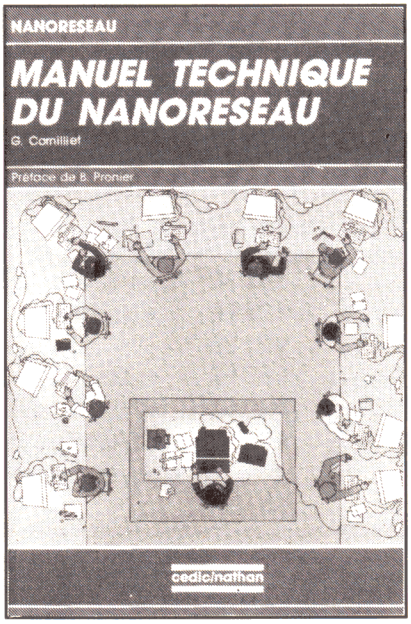
En plus d 'un tableur performant, EXCEL offre les meilleurs graphiques de tous les logiciels de gestion et permet donc de créer des histogrammes, courbes, etc. d'une rare précision.

Le livre de Douglas Cobb est le premier de deux tomes consacrés à ce nouveau logiciel et devrait permettre à ses utilisateurs de le maîtriser rapidement. Le second volume leur fera découvrir les possibilités graphiques d'EXCEL, la gestion de base de données et la programmation des macro-instructions. Tout un programme !

• *LE MANUEL TECHNIQUE DU NANORESEAU (Cedic/Nathan)*

Certains de nos lecteurs sont enseignants et parfois amenés à utiliser des MO5, TO7 et TO7/70. Ils connaissent donc le NANORESEAU, développé par la société Léanord. Celui-ci leur permet en effet de faire bénéficier leurs postes de travail de ressources qui leur font souvent défaut : mémoire de masse à disque souple ou dur, imprimante...

Ce manuel technique, rédigé par Gilbert Cornillet (qui travaille au Centre national de documentation pédagogique et a participé au développement du NANORESEAU) devrait leur permettre de mieux connaître ce système et d'en exploiter ainsi toutes les ressources.



• *ASTROLOGIE, NUMÉROLOGIE, BIORYTHMES SUR APPLE II (Sybex)*

Pierrick Bourgault est ingénieur informaticien, mais après une formation universitaire en biologie et en psychologie. Aussi est-il l'auteur de logiciels de calcul astrologique utilisés par bon nombre de professionnels.

Nous lui devons un ouvrage qui devrait satisfaire les amateurs d'astrologie occidentale ou chinoise et de numérologie. Les biorythmes — fort à la mode en informatique personnelle — n'y sont pas oubliés.

Chaque matière comprend un clair exposé sur le sujet considéré, puis un petit logiciel de calcul et d'interprétation analysé point par point.

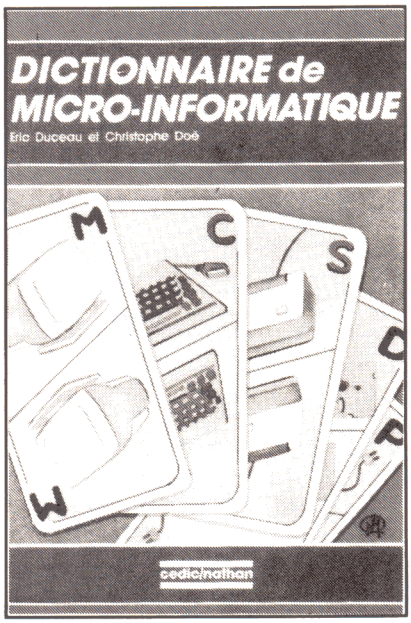
On peut regretter que Sybex ne propose pas une disquette des programmes... lesquels peuvent être considérés comme «consistants», mais c'est plutôt un bon point pour le livre de Pierrick Bourgault, non ?

• *DICTIONNAIRE DE MICRO-INFORMATIQUE (Cedic/Nathan)*

Une réédition d'un dictionnaire portant le même titre et paru dans la Collection Micro Monde. Ses auteurs : Eric Duceau et Christophe Doë. Son objectif : expliquer aussi simplement que possible, mais non sans rigueur, le vocabulaire de la micro-informatique familiale et professionnelle... à travers 200 articles.

Un peu court, mais un index de plus de 600 mots permet d'accéder directement à l'article où tel ou tel terme fait l'objet d'une explication. Deux lexiques — Français-Anglais et Anglais-Français — donnent les équivalents d'un certain nombre de mots-clés de la micro-informatique.

Ouvrage succinct, sans doute suffisant pour un certain nombre d'utilisateurs, mais qui laissera les autres sur leur faim.



UTILITAIRE | DOS 3.3 exclusivement

Marcel COTTINI

ENLÈVE DOS

ENLÈVE DOS est un utilitaire écrit en langage Basic AppleSoft. Il n'occupe que 8 secteurs sur la disquette. Son rôle essentiel est de récupérer 32 secteurs d'une disquette système, pour une utilisation future en disquette fichiers.

Les lecteurs de *T.M.* travaillant avec des fichiers texte connaissent très bien le problème. Une disquette système, comportant sur les trois premières pistes le DOS, pourra être facilement convertie en disquette fichiers, non bootable par la suite. Cette manipulation par le programme *ENLÈVE DOS*, permet de récupérer les 32 secteurs des pistes $Ø1 et $Ø2, ce qui accroît la capacité de la disquette d'environ 6%.

La récupération de la piste $ØØ n'est d'aucune utilité pour l'utilisateur, le DOS 3.3, de part sa conception d'origine, est incapable d'aller lire ou écrire sur cette piste. La VTOC (Volume Table Of Contents), table d'occupation des pistes et secteurs de la disquette, est automatiquement mise à jour. Le programme marque libres les secteurs $ØØ-$ØF des pistes $Ø1 et $Ø2. En fin de programme, à partir de la ligne 25Ø, un texte amorce avec un sous-programme de boot 1 qui est mis en place sur la piste $ØØ secteur $ØØ (92 bytes de long).

À PROPOS DU PROGRAMME

La table IOB, débutant à partir de l'adresse $B7E8 (47Ø8Ø) pour un système booté sur une configuration de 64 Ko, renferme tous les paramètres pour la lecture ou l'écriture d'un secteur (RWTS — Read Write Track Sector).

La ligne **16Ø** sélectionne la piste 17, secteur ØØ.

La ligne **17Ø** met le numéro du volume à ØØ pour autoriser RWTS à traiter n'importe quelle disquette. Le POKE 47Ø91,Ø positionne le compteur pour traiter 256 bytes d'un secteur.

La ligne **18Ø** fixe le pointeur de la zone du tampon de travail (buffer) utilisé ultérieurement par la routine RWTS.

La ligne **19Ø** met en place un micro-programme à partir de l'adresse $Ø3ØØ (768) :

```
Ø3ØØ : 2Ø E3 Ø3  JSR $Ø3E3  Recherche de
                            l'adresse de la table IOB des para-
                            mètres de RWTS.
Ø3Ø3 : 4C D9 Ø3  JMP $Ø3D9  Lecture ou
                            écriture d'un secteur (RWTS).
```

La ligne **21Ø** fixe le code de commande à la valeur Ø1, soit en mode lecture. CALL 768 appelle le sous-programme à l'adresse $Ø3ØØ. La piste 17 secteur ØØ est lue, puis placée dans le tampon se trouvant à l'adresse déclarée à la ligne 18Ø.

La ligne **22Ø** écrit 4 bytes (FF) dans le tampon du DOS réservé pour la routine RWTS. A cette adresse se trouve la réplique de la VTOC. Cette manipulation

(suite page 44).

équivaut à marquer comme libres, les pistes $01 et $02.

La ligne **230** réécrit, sur la piste 17 secteur 00, les 256 bytes se trouvant dans le tampon réservé au DOS (routine *RWTS*). Maintenant la VTOC marque entre autres, les pistes $01 et $02 comme libres.

La ligne **240** sélectionne la piste $00 secteur $00.

La ligne **250**, par l'intermédiaire d'une boucle FOR/NEXT, lit les DATAs des lignes 300-310 et les place dans le tampon fixé par la variable LOC (adresse $2710).

La ligne **260** appelle le sous-programme en $0300 et réécrit les données placées dans le tampon (adresses $2710-$276B), sur la disquette, à la piste $00 secteur $00 plus précisemment. Ce secteur très particulier pour le DOS, contient toujours le programme de boot 1, permettant au système de connaître la suite du processus de boot. Pour rappel, le programme du boot 0 se trouve implanté dans la carte contrôleur du lecteur de disquettes (adresses $Cs00-$CsFF). Boot 0 effectue une lecture du contenu de la piste $00 secteur $00, puis place les 256 octets à partir de l'adresse $0800. Ensuite, le micro-programme est exécuté à partir de l'adresse $0801. Le premier byte étant uniquement un paramètre de contrôle pour le DOS 3.3.

Les lignes **300-310** contiennent les DATAs qui, après lecture par l'instruction READ, seront implantées à partir de l'adresse $2710. Cette zone de la mémoire vive (RAM) n'est utilisée que provisoirement comme tampon interne de la routine RWTS.

Le listing suivant commente les instructions placées sur la piste $00 secteur $00 (boot 1 de la disquette). Chaque instruction machine est traitée, pour une meilleure compréhension par le lecteur novice.

2710 — 01	Paramètre utilisé par le sous-programme de la carte contrôleur. La valeur 01 indique que seul un secteur devra être lu, en l'occurrence le secteur $00.
2711 — JSR $FE89	SETKBD. Sélectionne le clavier comme périphérique d'entrée (IN£0).
2714 — JSR $FE93	SETVID. Sélectionne l'écran vidéo comme périphérique actuel, mode sortie (PR£0).
2717 — JMP $0827	Saut au sous-programme (boot 1), après implantation à partir de l'adresse $0800. En effet, après le boot de la disquette, le sous-programme se trouvant sur la piste $00 secteur $00 sera placé dans la zone mémoire débutant à partir de l'adresse $0800.
271A — PLA	Dépile le pointeur de poids faible,
271B — STA $40	et le sauvegarde dans la page 0 (A3L — LByte).
271D — PLA	Dépile le pointeur de poids fort,
271E — STA $41	et le sauvegarde dans la page 0 (A3H — HByte).
2720 — LDY £$00	Initialise le registre Y comme compteur, 256 positions.
2722 — INC $40	Incrémente le contenu de l'adresse $40 de 1. Positionne le compteur sur le prochain byte.
2724 — LDA ($40),Y	Charge le contenu de l'adresse $40 indexée par le registre Y. Adressage indirect,Y.
2726 — BEQ $272E	Est-ce le dernier byte à traiter ? Branchement si zéro. Si le dernier résultat est égal à zéro (bit Z = 1), ou si la dernière comparaison a donné l'égalité, on saute à l'adresse indiquée ; sinon, on continue en séquence.
2728 — JSR $FDED	COUT. Fait exécuter le sous-programme de sortie de caractère, par exemple COUT 1.

272B —	**JMP $0812**	Saut au sous-programme, après boot 1.
272E —	**LDA $41**	Charge le pointeur de poids fort,
2730 —	**PHA**	et l'empile au sommet de la pile.
2731 —	**LDA $40**	Charge le pointeur de poids faible,
2733 —	**PHA**	et l'empile au sommet de la pile.
2734 —	**JMP $FE84**	SETNORM. Commute le mode d'affichage en normal.
2737 —	**JSR $FC58**	HOME. Efface l'écran et positionne le curseur en haut et à gauche de l'écran.
273A —	**LDA £$0B**	Charge l'accumulateur avec la valeur 11,
273C —	**STA $25**	et la sauvegarde comme position verticale du curseur (CV).

(suite page 46)

ENLÈVE DOS

```
100 REM ENLEVE DOS
101 REM PAR MARCEL COTTINI
102 REM POUR TREMPLIN MICRO
103 REM JUILLET 86
104 REM ctJctJctJctJctJctJctJ
110 TEXT : HOME                                                    1C5A
120 PRINT "ENLEVE DOS": PRINT "----------": PRINT : PRINT "LIBERE 32
    SECTEURS,CE QUI AUGMENTE LA": PRINT "CAPACITE DE 6%.DISK NE SE BO
    OTE PLUS."                                                     D81A
130 PRINT : FLASH : PRINT "ATTENTION";: NORMAL : PRINT " NE CONTINUEZ
     PAS SI VOUS ¯ctMNE VOULEZ PAS DETRUIRE LE DOS DU DISK!": PRINT 8CC7
140 PRINT "INSEREZ LE DISK,PUIS ¯<RETURN>:";: GET A$: IF A$ <  > CHR
    $ (13) THEN 290                                                A1DA
150 HTAB 1: INPUT "EFFACER LE DOS DE CE DISK ? (O/N): ";A$: IF A$ <
    > "O" THEN 290                                                 397C
160 POKE 47084,17: POKE 47085,0: REM SELECTIONNE PISTE 17,SECTEUR 0 F0AB
170 POKE 47083,0: POKE 47091,0                                     8B6F
180 LOC = 10000: POKE 47088,LOC -  INT (LOC / 256) * 256: POKE 47089,
    INT (LOC / 256)                                                4FD6
190 POKE 768,32: POKE 769,227: POKE 770,3: POKE 771,76: POKE 772,217:
    POKE 773,3: REM ROUTINE POUR SAUT A RWTS                       84B6
210 POKE 47092,1: CALL 768: REM LIT LE SECTEUR                     E487
220 POKE LOC + 60,255: POKE LOC + 61,255: POKE LOC + 64,255: POKE LOC
    + 65,255: REM REECRIT 4 BYTES                                  3FEC
230 POKE 47092,2: CALL 768: REM ECRITURE DU SECTEUR                C788
240 POKE 47084,0: POKE 47085,0: REM SELECTIONNE PISTE 0,SECTEUR 0  1A73
250 FOR I = LOC TO LOC + 91: READ V: POKE I,V: NEXT                22DA
260 CALL 768: REM REECRIT NOUVEAU SECTEUR DE BOOT                  8331
270 VTAB 3: PRINT : CALL  - 958                                    24FE
280 PRINT : PRINT "TERMINE.": PRINT : PRINT "UN AUTRE DISK ? (O/N): "
    ;: GET A$: IF A$ = "O" THEN  RUN                               2A9C
290 VTAB 20: END                                                   B5BE
300 DATA 1,32,137,254,32,147,254,76,39,8,104,133,64,104,133,65,160,0,
    230,64,177,64,240,6,32,237,253,76,18,8,165,65,72,165,64,72,76,132
    ,254,32,88,252,169,11,133,37,169,10                            6770
310 DATA 133,36,32,34,252,32,128,254,32,10,8,213,206,194,207,207,212,
    193,194,204,197,160,196,193,212,193,160,196,201,211,203,0,166,43,
    189,136,192,169,22,133,37,76,0,224                             A129
```

```
273E — LDA £$ØA   Charge l'accumulateur avec la valeur 1Ø,
2740 — STA $24     et la sauvegarde comme position horizontale du curseur (CH).
2742 — JSR $FC22   VTAB déplace le curseur en CV
                   — contenu de l'adresse $25.
2745 — JSR $FE8Ø   SETINV. Commute le mode d'affichage en inverse.
2748 — JSR $Ø8ØA   Saut au sous-programme, après boot 1.
```

Table des codes ASCII du texte UNBOOTABLE DATA DISK

```
274B — D5 CE C2 CF UNBO
       CF D4 C1 C2 OTAB
       CC C5 AØ C4 LE D
       C1 D4 C1 AØ ATA
       C4 C9 D3 CB DISK

275F — ØØ          Fin de la table des codes ASCII.
2760 — LDX $2B     BAS2H. Charge le pointeur de poids fort de la ligne de destination
                   en déroulement d'écran.
2762 — LDA $CØ88,X Commutateur de lecture seulement en MEV.
2765 — LDA £$16    Charge l'accumulateur avec la valeur 22,
2767 — STA $25     et sauvegarde comme position verticale du curseur (CV).
2769 — JMP $EØØØ   Vecteur d'entrée du Basic AppleSoft.
```

EØØØ — JMP $F128 Démarrage à froid. Toutes les données sont mises à zéro, le pointeur de pile est réinitialisé. ■

À LIRE ET À RELIRE

Un ouvragre très complet de Jean-Pierre LAGRANGE

Connaissez-vous vraiment les systèmes d'exploitation et les systèmes de protection de votre Apple *II* ? Je parie que la réponse est négative... mais que grande est votre curiosité quand on évoque ces sujets.

Il est certain que le gros bouquin de Jean-Pierre LAGRANGE, un spécialiste incontestable, va faire non seulement le bonheur de certains développeurs, mais encore celui de bon nombre d'amateurs dits éclairés... ou désirant mériter ce qualificatif.

Quand vous aurez potassé ce nouveau volume de MICRO APPLICATION — éditeur dont la réputation n'est plus à faire —, vous saurez tout (ou presque) de la mémoire de votre Apple, de ses systèmes d'exploitation, de l'implantation en mémoire et sur disque, des appels système, de la structure d'une disquette...

Non, ce n'est pas terminé, et il y a encore : les protections de la mémoire et celles des disquettes, les codages et formatages spéciaux, la protection du chargement et les techniques des pirates...

Plus de 400 pages bourrées d'explications et de listages... pour moins de 180 F (MICRO APPLICATION vous fournira éventuellement, en échange de 120 F, la disquette correspondante... mais je ne l'ai pas testée).

Clément RENARD.

TRI RAPIDE*

Oubliez le tri en Basic !

Gagnez un temps précieux en utilisant, pour le tri de vos listes de mots, une routine en langage machine. Celle que nous vous proposons ici ne compte que 183 octets... à peine une longue ligne de Basic !

TRI.1.2 autorise le tri de tableaux comportant plus de 255 éléments (Ø à 254). Elle est implantée à partir de l'adresse $3ØØ (768) et utilise quelques octets de la page Ø (vérifiez que vous ne les employez pas déjà dans votre programme).

MODE D'EMPLOI N'oubliez pas de dimensionner le tableau, bien sûr et n'hésitez pas à forcer la dose. Finalement, ce sera le nombre exact des mots saisis qui sera pris en compte. Attention ! N sera toujours égal à N + 1. L'élément Ø peut être utilisé. **Syntaxe :** CALL 768, N, M$(Ø).

** Rapide... par rapport à la version en Basic... car il s'agit du tri à bulle, généralement considéré (avec de nombreux éléments) comme étant le plus lent.*

TRI.DÉMO

```
100 TEXT : HOME : PRINT CHR$ (4)"PR£3": PRINT :N = 1
110 DIM M$(400)
120 PRINT CHR$ (4)"BLOAD TRI.1.2"
122 REM Ligne à rétablir:GOTO130
123 PRINT "300 FAUX MOTS DE 5 LETTRES...PATIENCE!"
124 FOR I = 1 TO 5:M$(N) = M$(N) + CHR$ (65 + INT ( RND (1) * 2
    6)): NEXT :N = N + 1: IF N < = 300 THEN 124
125 CALL - 198: PRINT "LE TRI COMMENCE": GOTO 190
130 VTAB 10: CALL - 958: PRINT N
140 VTAB 12: PRINT M$(N);: HTAB 1:
150 INPUT "";M$(N): IF M$(N) = "*" THEN N = N - 1: GOTO 130
160 IF M$(N) = "" THEN 180
170 N = N + 1: IF N < = 400 THEN 130
180 PRINT : PRINT "OUI POUR TRIER ";: GET R$: PRINT : IF R$ < >
     "O" THEN 130
190 CALL 768,N,M$(0)
220 FOR I = 1 TO N: PRINT M$(I),: IF PEEK ( - 16384) > 127 THEN
     POKE - 16368,0: WAIT - 16384,128,127: POKE - 16368,0
230 NEXT
```

Toutes les versions de l'Apple n'acceptent pas la syntaxe de la ligne 220 (PRINT M$(I), :) Le cas échéant, travaillez sur 40 colonnes.

TRI À PARTIR DU CARACTÈRE D'UN RANG DÉTERMINÉ :

Inclure un POKE* 885, R — 1 (R = rang) avant la ligne Basic 190. Ainsi, POKE 885,1 triera la liste sans tenir compte du premier caractère.

** Valeurs normales : 885,0 — 887,252 — 889,0 — 905,176.*

CLASSER À PARTIR DE LA DONNÉE LA PLUS GRANDE (ordre alphabétique inverse) :

Par quelques pokes bien ajustés : POKE* 887,7: POKE* 889,252: POKE* 905,144. Ne pas oublier de rétablir les bonnes valeurs pour réaliser un tri normal.

EXPLOITATION DE N (NOMBRE D'ÉLÉMENTS)

```
300 : 20 BE DE   JSR $DEBE
303 : 20 7B DD   JSR $DD7B
306 : 20 52 E7   JSR $E752
309 : A6 51      LDX $51
30B : A5 50      LDA $50
30D : D0 05      BNE $0314
30F : E0 00      CPX £$00
311 : D0 01      BNE $0314
313 : 60         RTS
314 : 86 F9      STX $F9
316 : 85 FA      STA $FA
```

CHKCOM teste la virgule.
FRMEVL évalue l'expression (vers FAC).
GETADR convertit FAC en valeur entière que l'on peut lire en $50-51.
Si la valeur est égale à 0, pas de tri.
Pointeur $F9-FA initialisé avec le nombre d'éléments.

EXPLOITATION DE M$C0

```
318 : 20 BE DE   JSR $DEBE
31B : 20 E3 DF   JSR $DFE3
```

CHKCOM : teste la virgule dans TXTPTR.
PTRGET : recherche la variable par son nom.

(RÉ)INITIALISATION

```
31E : A5 83      LDA $83
320 : 85 85      STA $85
322 : A5 84      LDA $84
324 : 85 86      STA $86
326 : A9 00      LDA £$00
328 : 85 EB      STA $EB
32A : 85 EC      STA $EC
32C : 85 09      STA $09
```

L'adresse de base est contenue dans $83-84 et il est facile de la réinstaller dans $85-86.
On met dans $EB/EC le nombre d'éléments triés.
Si 9 = 0, pas d'échange, si 9 = 1, on a modifié la liste.

ÉLÉMENT 0

```
32E : A0 02      LDY £$02
330 : B1 85      LDA ($85),Y
332 : 99 06 00   STA $0006,Y
335 : 88         DEY
336 : 10 F8      BPL $0330
```

On écrit page 0 :
— 6 : longueur de l'élément.
— 7 :
— 8 : adresse où il est implanté dans la mémoire.

COMPARAISON COMPTEUR-POINTEUR

```
338 : E6 EC      INC $EC
33A : D0 02      BNE $033E
33C : E6 EB      INC $EB
33E : A5 EB      LDA $EB
340 : C5 F9      CMP $F9
342 : D0 06      BNE $034A
344 : A5 EC      LDA $EC
346 : C5 FA      CMP $FA
348 : F0 56      BEQ $03A0
```

Comparaison entre le pointeur
- $EB-EC : nombre d'éléments survolés... et
- $F9-FA : nombre d'éléments de la liste.

Quand il y a égalité, GOTO $3A0.

PRÉPARATION PERMUTATION

```
34A : A5 85      LDA $85
34C : 85 18      STA $18
34E : 69 03      ADC £$03
350 : 85 85      STA $85
352 : A5 86      LDA $86
```

L'adresse + 3 (élément suivant) est sauvegardée dans $18-19 (où elle sera utilisée plus loin pour la permutation éventuelle) et placée dans $85-86.

```
354 : 85 19      STA $19
356 : 69 00      ADC £$00
358 : 85 86      STA $86
```
(Voir explications des 5 lignes précédentes, page 48).

```
35A : A5 06      LDA $06
35C : F0 D0      BEQ $032E
```
Si la longueur de l'ÉLÉMENT 0 est nulle, GOTO $32E.

ÉLÉMENT 1

```
35E : A0 02      LDY £$02
360 : B1 85      LDA ($85),Y
362 : 99 FB 00   STA $00FB,Y
365 : 88         DEY
366 : 10 F8      BPL $0360
```
$FB = longueur de l'élément 1.

$FC, $FD : adresse où il est implanté.

```
368 : A5 FB      LDA $FB
36A : F0 1F      BEQ $038B
```
Si $FB = 0, permutation immédiate.

```
36C : C5 06      CMP $06
36E : 90 02      BCC $0372
370 : A5 06      LDA $06
372 : 85 FE      STA $FE
```
Si la longueur de l'ÉLÉMENT 1 est plus petite que celle de l'ÉLÉMENT 0, le contenu de A est bon. Sinon on prend la plus petite longueur que l'on stocke dans $FE.

ÉLÉMENT 1 COMPARÉ À ÉLÉMENT 0

```
374 : A0 00      LDY £$00
376 : B1 FC      LDA ($FC),Y
378 : D1 07      CMP ($07),Y
```
Caractère de l'ÉLÉMENT 1 (de rang Y) comparé à celui de l'ÉLÉMENT 0.

```
37A : F0 04      BEQ $380
```
Si égalité, la boucle continue.

```
37C : B0 2A      BCS $03A8
```
Si plus grand, pas de permutation (GOTO $3A8).

```
37E : D0 0B      BNE $038B
```
Si plus petit, permutation immédiate (GOTO $38B).

```
380 : C8         INY
381 : C4 FE      CPY $FE
383 : D0 F1      BNE $0376
```
Comparaison terminée si Y est égal à $FE.

```
385 : A5 FB      LDA $FB
387 : C5 06      CMP $06
389 : B0 1D      BCS $03A8
```
Si la longueur de l'ÉLÉMENT 1 est égale ou plus grande que celle de l'ÉLÉMENT 0, pas de PERMUTATION.

```
38B : A9 01      LDA £$01
38D : 85 09      STA $09
```
09 est mis à 1.

PERMUTATION

```
38F : A0 02      LDY £$02
391 : B1 85      LDA ($85),Y
393 : 48         PHA
394 : B1 18      LDA ($18),Y
396 : 91 85      STA ($85),Y
398 : 68         PLA
399 : 91 18      STA ($18),Y
39B : 88         DEY
39C : 10 F3      BPL $0391
```
L'adresse de l'ÉLÉMENT 1 prend la place de celle de l'ÉLÉMENT 0... et vice-versa.

```
39E : 30 08      BMI $03A8
```
GOTO $3A8.

TRI TERMINÉ ?

```
3A0 : A5 09      LDA $09
3A2 : F0 03      BEQ $03A7
```
S'il n'y a eu aucune permutation, tri terminé.

```
3A4 : 4C 1E 03   JMP $031E
```
Sinon on recommence.

```
3A7 : 60         RTS
```
RETOUR AU BASIC.

ULTIME CONTRÔLE

```
3A8 : A5 EB      LDA $EB
3AA : C5 F9      CMP $F9
3AC : D0 06      BNE $03B4
3AE : A5 EC      LDA $EC
3B0 : C5 FA      CMP $FA
3B2 : B0 EC      BCS $03A0
```
Compteur comparé avec $F9-FA.
Si son contenu est égal ou plus grand, voir $3A0 (permutation ou pas ?).

```
3B4 : 4C 2E 03   JMP $032E
```
Affaire à suivre... ■

TRI.COMPL

Voir **TREMPLIN MICRO** numéro 10, pages 26-27 et 28.

Le programme TRI.I.Ø ne fonctionne que dans la mesure où le nombre d'éléments à trier est identique à celui déclaré par DIM.

Avec le PATCH de TRI.COMPL (ligne 25Ø), vous pourrez utiliser TRI.I.Ø sans cette contrainte.

Notez la nouvelle formulation de l'appel à la routine ligne 19Ø.

```
100 TEXT : HOME : PRINT CHR$ (4)"BLOAD TRI.1": GOSUB
    240
105 PRINT CHR$ (4)"BSAVE TRI.1.1,A$300,L$9C": REM INU
    TILE DE CONSERVER CETTE LIGNE APRES UNE PREMIERE
    UTILISATION - LA NOUVELLE VERSION SERA SUR LA DIS
    QUETTE
110 DIM M$(254): REM PAS PLUS DE 255 ELEMENTS (0 A 25
    4, soit 255 ou $FF)
120 N = N + 1
130 VTAB 10: CALL - 958: PRINT N
140 VTAB 12: PRINT M$(N);: HTAB 1
150 INPUT "";M$(N): IF M$(N) = "*" THEN N = N - 1: GO
    TO 130
160 IF M$(N) = "" THEN 180
170 IF N < 254 THEN 120
180 PRINT : PRINT "OUI POUR TRIER ";: GET R$: PRINT :
     IF R$ < > "O" AND R$ < > "o" THEN 130
190 CALL 768,N,M$(0)
200 FOR I = 1 TO N: PRINT M$(I): NEXT
210 END
240 FOR I = 768 TO 794: READ R: POKE I,R: NEXT : RETU
    RN
250 DATA 32,190,222,32,123,221,32,82,231,165,80,201,2
    ,176,1,96,133,250,32,190,222,32,227,223,234,234,2
    34
```

LA LIGNE DATA

```
300 : 20 BE DE    JSR   $DEBE     CHKCOM teste la virgule.
303 : 20 7B DD    JSR   $DD7B     FRMEVL évalue l'expression (FAC).
306 : 20 52 E7    JSR   $E752  ]  GETADR convertit FAC en valeur entière et le résultat est
309 : A5 50       LDA   $50    ]  dans $50-51.
30B : C9 02       CMP   £$02   ]  Si moins de 2 éléments, pas de TRI.
30D : B0 01       BCS   $0310  ]
30F : 60          RTS             Retour.
310 : 85 FA       STA   $FA       Nombre d'éléments placé dans $FA.
312 : 20 BE DE    JSR   $DEBE     Re... teste la virgule.
315 : 20 E3 DF    JSR   $DFE3     PTRGET recherche la variable.
318 : EA          NOP           ]
319 : EA          NOP           ]  NOP... sans signification
31A : EA          NOP           ]
```

Après l'avoir utilisée une fois, vous pourrez sauver la nouvelle routine : **BSAVE TRI.1.1,A$300,L$9C.**
Rappelons qu'elle ne peut trier plus de 255 éléments DIM ×$(254). ■

Madeleine HODÉ, auteur de GRIBOUILLE, traitement de texte dont on ne vantera jamais assez les nombreuses qualités, prend des risques (calculés ?) : **elle commercialise son logiciel sans aucune protection... mais elle fait confiance aux...**

PIRATES AU CŒUR PUR

TREMPLIN MICRO : Que pensez-vous du piratage des logiciels ?

M. HODÉ : Je déteste les gens qui vendent les copies piratées, ou qui les donnent pour placer une machine. Ce sont des rampants.

Mais le pirate au cœur pur, je le comprends. Une disquette bien lockée, c'est une énigme terriblement excitante. On doit y perdre le boire et le manger !

Malheureusement, les copies circulent. Les logiciels ne se vendent plus. On finira par ne plus en écrire. J'aimerais connaître l'auteur de Locksmith. On ne sait rien de lui, son nom n'apparaît même pas à l'écran. Incroyable : il continue. Il y a toujours une dernière version, plus performante que la précédente. J'écris dans mon manuel qu'on devrait lancer une souscription pour lui. *Tremplin Micro* pourrait s'atteler à cette tâche ! Faites une enquête, trouvez-nous l'auteur de Locksmith, parlez-nous de lui. Existe-t-il ? Locksmith alias Bourbaki ? Dites-nous qui est l'auteur de Locksmith, dites-nous comment acheter Locksmith. Pas trop cher, quelques centaines de francs. Parmi vos lecteurs, beaucoup seraient heureux de lui envoyer une rémunération. Ils auraient un manuel (un manuel de Locksmith, quelle merveille !) et aussi un abonnement pour les mises à jour. Ça marcherait, et vos lecteurs seraient contents.

Même chose pour Disk Fixer, pour Bags of Tricks. Ce sont des logiciels merveilleux, indispensables, à peu près introuvables dans les boutiques, et inconnus du public. Décidément non, les pirates au cœur pur diffusent mal les logiciels.

TREMPLIN MICRO : Vous comptez cependant sur ces pirates au cœur pur pour vous aider à diffuser GRIBOUILLE, à en faire un traitement de texte populaire ?

M. HODÉ : Oui, pour mon GRIBOUILLE, vous avez vu que j'essaie précisément cette formule : pour quelques centaines de francs (610 francs exactement) on a un manuel et des dis-

(suite page 52)

quettes. Il est vrai que l'on peut avoir la même chose pour beaucoup moins cher : les disquettes ne sont pas protégées, et le manuel passe comme lettre à la poste sur les grosses photocopieuses. Je ne peux pas empêcher cette forme de piratage. Je tente de faire plus. J'ai le service après vente. On peut m'écrire (on ne s'en prive pas, heureusement). Interrogez vos lecteurs : depuis que je vends mon programme moi-même, vous ne trouverez pas de lettres en souffrance. Je réponds le jour même ou le lendemain. Je le ferai toujours. S'il y a trop de lettres, je me ferai aider. Mais je tiens par-dessus tout au dialogue, au contact. C'est indispensable pour moi et pour GRIBOUILLE. Les meilleures idées m'ont été données par des utilisateurs.

TREMPLIN MICRO : **Vous croyez donc au dialogue et mieux : vous le sollicitez...**

M. HODÉ : Bien sûr, et pour les utilisateurs de GRIBOUILLE aussi, c'est important. Un traitement de texte est un logiciel de base. C'est un outil. La passivité, l'anonymat sont une souffrance. Il faut pouvoir dire à l'auteur : "Tel truc est mal conçu, je veux faire telle chose, et je n'y arrive pas, le manuel est incompréhensible". Il y a un courant, une amitié qui passent. Vous avez vu dans mon manuel les trois "fiches" écrites par des utilisateurs ? Vous les avez lues ? Vous avez vu de quelle qualité elles sont ? Et ils ont accepté tous les trois que je donne leurs noms. Ces trois fiches, c'est vraiment le bonheur et la fierté !

TREMPLIN MICRO : **D'accord, mais revenons aux choses concrètes, voulez-vous ? Parlez-nous de votre système de mise à jour...**

M. HODÉ : Ce système de mise à jour : rien de compliqué. En achetant GRIBOUILLE, on s'abonne en quelque sorte. On reçoit de temps en temps "Gribouille and Co". On m'écrit et je réponds. On me dit ce qu'on veut trouver dans la prochaine mise à jour : mailing, fichier d'adresses, envoi sur modem, consultation de fichier avec possibilité de prélèvement, ou autre chose... (Les études de marché m'assomment, je préfère être guidée par les utilisateurs).

De temps en temps, on reçoit une proposition honnête : pour tel prix (par exemple 200 francs) je vous expédie une disquette et un complément de manuel qui font telle ou telle chose. On achète, ou on attend la mise à jour suivante.

TREMPLIN MICRO : **Intéressant et nouveau, mais où trouver GRIBOUILLE ? Jusqu'à maintenant, la commercialisation n'était pas très bien conduite...**

M. HODÉ : La version ProDOS de GRIBOUILLE est vendue uniquement par l'auteur et voici la bonne adresse :
Société GRIBOUILLE,
16, rue de Poules, 67000 STRASBOURG.
Chèque de 610 francs TTC, port compris, libellé à l'ordre de GRIBOUILLE.

Personne d'autre ne vend GRIBOUILLE ProDOS. La logique est rigoureuse : je vends bon marché pour décourager les copieurs ; je veux connaître avec certitude le nom des personnes qui ont acheté mon logiciel (à cause du service après vente). Ces deux impératifs m'interdisent de passer par des intermédiaires. D'ailleurs les concessionnaires Apple le comprennent très bien. Ils donnent mon adresse à leurs clients, alors qu'ils n'attendent aucune commission, ni aucune forme de rémunération. Cela aussi, je voudrais que vous le disiez.

TREMPLIN MICRO : **Parlez-nous un peu de votre bulletin de liaison...**

M. HODÉ : Vous ne connaissez pas "Gribouille and Co" ? C'est le journal des amis de GRIBOUILLE. Le mot le plus fréquent, c'est "Comment" : Comment écrire des programmes en Applesoft avec GRIBOUILLE ; comment imprimer l'adresse de l'expéditeur à gauche et celle du destinataire à droite, juste en face ; comment faire ses graphiques en Applesoft et les enregistrer (pour pouvoir les imprimer avec GRIBOUILLE, bien sûr).

Beaucoup de procédés et de "trucs". Un peu comme *Tremplin Micro*. Et puis aussi, probablement, des annonces et des échanges. J'ai un client qui veut échanger un alphabet grec contre un alphabet cyrillique. Une bourse d'échanges de caractères personnalisés, c'est une idée, non ?

Madeleine HODÉ aurait pu nous parler de son enfant pendant des heures. Elle en connaît le moindre octet et a dû surmonter bien des difficultés pour en faire l'excellent traitement de texte qu'il est devenu (les premières versions méritaient déjà ce qualificatif).

GRIBOUILLE

sait couper les mots en Français

La nouvelle version de Gribouille n'a rien à envier à l'ancienne, on s'en doute. Madeleine HODÉ s'est efforcée de tenir compte des suggestions, mais aussi des critiques que lui ont adressées les premiers utilisateurs de cet excellent traitement de texte français.

Il fonctionne sur Apple *IIc* et sur Apple *IIe* avec carte 8Ø colonnes (128 K). ProDOS lui confère une grande rapidité dans le chargement et la sauvegarde des fichiers.

L'un de ses premiers avantages est de se satisfaire de la présence d'un seul lecteur, mais il apprécie celle d'un second. Il semble être capable de s'adapter à toutes les imprimantes, pourvu qu'elles se révèlent compatibles avec l'Apple *II*... et son auteur est disposée à étudier les cas qui ne sont pas encore prévus (en existe-t-il ?).

L'apprentissage : une simple formalité !

Madeleine HODÉ vous propose de connaître "Gribouille en une heure" grâce à une disquette DEMO et elle est à peu près sûre de gagner son pari dans la plupart des cas. Même si vous n'avez jamais utilisé un logiciel de traitement de texte, vous maîtriserez rapidement celui-ci. D'ailleurs, si vous rencontrez des difficultés, le manuel, très complet, vous fournira tous les renseignements souhaités.

Les points forts

Frappe, remaniement de texte et corrections sont rapides et n'exigent ni acrobaties ni mémoire d'éléphant.

Le glossaire est créé par frappe directe et la longueur des clés d'appel est modulable. Il peut comporter des sauts de ligne et des caractères de contrôle destinés à la mise en page ou à des codes d'impression. Il sera mémorisé sur disquette et automatiquement chargé lors de chaque chargement du logiciel.

La capacité de Gribouille est de 45ØØØ caractères, ce qui n'est déjà pas mal, mais il est évidemment possible d'enchaîner plusieurs fichiers lors de l'impression.

La mise en page respecte parfaitement les règles de la typographie traditionnelle, notamment pour les coupures de mots, en bout de ligne. Elle peut être affichée et contrôlée sur l'écran. Cette facilité améliore considérablement la présentation de certains textes (rapports, thèses, etc.). Notons au passage que la justification reste assurée, pour une même ligne, avec différentes polices de caractères. *(suite page 54)*

Il est possible et facile de créer et d'utiliser des caractères personnalisés (quand l'imprimante le permet). C'est un plus appréciable quand on désire écrire des formules mathématiques... ou éditer un texte dans certaines langues étrangères. Et les graphiques ? Oui, ils viennent tout naturellement s'insérer au milieu du texte à condition de les avoir enregistrés au préalable.

Et encore... et encore...

Faut-il parler des lettres-types, des formulaires, de la tabulation des nombres (aucun problème !), des calculs arithmétiques faciles, en ligne et en colonne, de la facturation ? Inutile puisque tout cela a été prévu par une Madeleine HODÉ qui s'est véritablement surpassée !

Un détail : le manuel de Gribouille est fourni sans classeur et ce très beau logiciel a renoncé à la souris et à ses menus déroulants. Qui le lui reprochera ? Certainement pas ceux qui, à l'usage, en apprécieront les nombreuses qualités.

Guy-HACHETTE.

Du coq à l'âne

TRANSLIST (n°7)

Avez-vous corrigé l'erreur manifeste de la ligne 7Ø5... où E$(LI) doit être remplacé par ES$(LI) ? Annie Lecamp a bien gagné sa prime en nous signalant cette erreur dès la réception de son numéro. Bravo !

RUBANS SCRIBE

• *Pourriez-vous me dire si la production des rubans pour l'imprimante SCRIBE est arrêtée... et me préciser où il est possible de s'en procurer ?*

H.D. (69000 OULLINS)

TM

Vous n'êtes pas le seul à courir à la poursuite des rubans. Demandez à votre revendeur qu'il fasse son travail et ne se contente plus de vendre des machines. C'est cela qu'on appelle un SERVICE !

MINI-ASSEMBLEUR *IIc*

• *Je vous envoie cette note (traduction d'un extrait de la rubrique HELP, dans A+ de mai 1986).*

G. de J (69590, ST-SYPHORIEN-SUR-COISE)

La nouvelle ROM qui permet au *IIc* de fonctionner avec l'Unidisk 3.5 comporte un mini-assembleur. Pour savoir si votre *IIc* est doté de ce mini-assembleur, faites un CALL — 151 pour entrer dans le moniteur. Quand paraît le *, tapez ! suivi par ENTER ; si le symbole d'appel devient !, le mini-assembleur est présent et vous pouvez commencer à entrer un programme.

Vous pouvez aussi examiner certains bits d'identification dans la ROM pour savoir si vous avez la nouvelle ROM : pour le *IIc*, il faut avoir $ØØ dans $FBBF, et pour le *IIe* $EØ dans $FBCØ.

Si en revanche vous lisez $FF et $EA respectivement, vous avez l'ancienne ROM.

TEXT et RND

1. *Comment revenir en mode texte, à partir du moniteur, sans passer par le Basic ?*
2. *Comment être sûr que l'on n'obtiendra pas toujours le même nombre en allumant l'APPLE, avec la fonction RND ?*

Magali (38190 BRIGNOUD)

TM

1. Simplement en tapant *F399G (adresse de la routine TEXT).
2. Essayez ce truc en Basic (il existe sûrement mieux). C'est à charger en mettant le programme en route.

```
10 TEXT : HOME : PRINT CHR$ (4)
"BLOAD ALEA"
15 POKE 157, PEEK (768): CALL 61364
20 X = 1 + INT(RND(1) * 256)
25 PRINT X
30 POKE 768,X: PRINT CHR$ (4)
"BSAVE ALEA,A768,L1"
```

HISTO.TEXT EN COULEUR

Notre ami Pierre PRIVAT vous propose de modifier HISTO.TEXT (n°9) pour l'utiliser avec les options couleurs disponibles sur l'IMAGEWRITER II (listage ci-contre).

Lignes modifiées :

376, 377, 378, 379, 380, 381, 382, 383, 401, 402 et 403.

A vous de jouer et merci à Pierre PRIVAT (de CASTRES).

NOUVELLE INCURSION (°11)

Erreur dans la ligne 315. Il faut remplacer le GOTO 310 par un GOTO 305. En marge (255), il faut lire 205 au lieu de 105.
Merci à André Chagon (LYON) et aux autres lecteurs perspicaces !

CADRE (p33, n°6)

Remplacer le 84 de la ligne 280 par un 85 (nombre réel de données). Frédéric Blanc ne nous en voudra pas d'avoir oublié de publier ce rectificatif plus tôt. Nous recevons beaucoup de courrier et celui-ci — parole de Nestor ! — n'est pas informatisé.

DMP UTILITIES

• Je viens de relire mon Tremplin Micro n°9 et j'ai vu qu'un lecteur cherchait des "Fontes de caractères" à mettre sur son Imagewriter. Je crois savoir ce qui ferait son bonheur. Le programme s'appelle DMP Utilities. Il permet de mettre dans la mémoire de l'Imagewriter I ou II (ou encore EPSON FX/JX) 24 fontes de caractères. Il n'est pas distribué en France mais on peut le trouver directement à l'adresse suivante :

Vilberg Brothers Comp. Inc.
4201 Hegg Avenue
Madison, WI 53716
USA

Ce programme est sous DOS 3.3 livré avec un manuel de 67 pages. Le prix de vente est de 50 dollars (+ 4 dollars de transport).

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

```
355 REM ******* IMPRESSION ************
360 :                                                    003A
370 VTAB 24: HTAB 8: INVERSE : PRINT "
    PRESSER UNE TOUCHE POUR IMPRIMER CE
    TTE PAGE (CTRL-C = A REFAIRE) ";: N
    ORMAL : CALL  - 198: GET R$: VTAB 1
    : PRINT "": IF R$ =  CHR$ (03) THEN
    150                                                  283B
375 IF R$ =  CHR$ (27) THEN 780                          65F7
376 HOME                                                 2F97
377 VTAB 4: HTAB 23: PRINT "CHOISISSEZ
    LES COULEURS DES BARRES: ": PRINT :
    HTAB 35: PRINT "NOIR ¨= 0": HTAB 3
    5: PRINT "JAUNE ¨= 1": HTAB 35: PRI
    NT "ROUGE ¨= 2": HTAB 35: PRINT "BL
    EU ¨= 3": HTAB 35: PRINT "ORANGE =
     4"                                                  A672
378 HTAB 35: PRINT "VERT ¨= 5": HTAB 3
    5: PRINT "VIOLET = 6"                                C05F
379 VTAB 16: HTAB 29: PRINT "COULEUR BA
    RRE "A1" :¨";: INPUT "";C1: IF C1 >
    6 THEN  CALL  - 198: GOTO 379                        5006
380 VTAB 17: HTAB 29: PRINT "COULEUR BA
    RRE "A2" :¨";: INPUT "";C2: IF C2 >
    6 THEN  CALL  - 198: GOTO 380                        A002
381 VTAB 19: HTAB 31: INVERSE : PRINT "
     CORRECT ? (O/N) ";: NORMAL                          7450
382 GET R$: IF R$ = "" THEN 382                          CB06
383 IF R$ = "N" OR R$ = "n" THEN  GOTO
    376                                                  503A
384 HOME : PRINT  CHR$ (21): PRINT                       041A
385 PRINT  CHR$ (4)"PR£1": PRINT : PRIN
    T CHR$ (9)"80N": PRINT  CHR$ (27)"
    "; CHR$ (27)"T15"; CHR$ (27)"E";                     8F44
390 FOR I = 1 TO 23: IF I = 1 OR I = 23
    THEN  PRINT L$(I): GOTO 415                          A875
395 PRINT  LEFT$ (L$(I),6);                              FD9A
400 FOR J = 7 TO  LEN (L$(I)):L$ =  MID
    $ (L$(I),J,1)                                        1163
401 IF L$ = "W" THEN  PRINT  CHR$ (27)"
    K"C1;                                                27E5
402 IF L$ = "N" THEN  PRINT  CHR$ (27)"
    K"C2;                                                ECDD
403 IF L$ = "U" OR L$ = "_" OR L$ = "W"
    OR L$ = "L" OR L$ = "N" THEN  PRINT
    CHR$ (27); CHR$ (38);L$; CHR$ (27);
    CHR$ (36); CHR$ (27)"K0";: GOTO 410                  C7CE
405 PRINT L$;                                            1865
410 NEXT                                                 0582
415 PRINT : NEXT                                         6576
420 PRINT  CHR$ (4)"PR£0"                                815F
425 PRINT  CHR$ (4)"PR£3": PRINT : HOME
    : GOTO 180                                           FCA5
```

Les solutions d'Yvan KOENIG

Nous transmettons parfois certaines des lettres de nos correspondants à Yvan KOENIG, lequel réussit généralement à trouver une solution à leur problème. Voici des exemples récents...

CHANGER DE SLOT OU DE DRIVE ?

Oui, il est possible de changer de slot et de drive par un ou deux POKE(s).

DOS 3.3 POKE 43624,1 ——» lecteur 1
POKE 43624,2 ——» lecteur 2
POKE 43626,6 ——» port 6

ProDOS Dans ce cas un seul octet est concerné dans lequel sont stockées les deux informations à savoir $BF3Ø : %Dsss0000
POKE 48944, port * 16 + (lecteur — 1) * 128

LA FONCTION RESTORE

La fonction RESTORE numligne est tout à fait programmable. On peut la trouver dans *Clefs pour APPLE IIc,* dans *POM'S 5* etc.

PARAMÈTRE T SOUS ProDOS

• ***A quoi sert le T dans cette instruction ?***

10' PRINT CHR$(4) ; "OPEN" ; PR$;NF$;"TTXT"

Il est vrai que le paramètre T est peu utilisé par les amateurs. Cependant il est parfois indispensable. Pour créer un SOUS-CATALOGUE il faut faire CREATE/prefix/titre , TDIR.

On peut également employer ce paramètre pour affecter un titre à un fichier sans le garnir et ce peut être utile pour placer les divers titres dans un ordre choisi. On peut aussi définir des fichiers de type particulier ; ainsi, MERLIN PRO utilise des fichiers de type REL différents d'ailleurs des fichiers de même type générés par l'assembleur APPLE lorsque le pseudo-opcode REL est employé. Certains parmi vous connaissent déjà PRO-CMD de Glen BREDON qui utilise des fichiers de type CMD et des fichiers de type $F7 pour les images DHGR compressées. *Tremplin Micro* publie SLOADC (qui permet de sauver très facilement les écrans 8Ø colonnes sous forme normale ou sous forme compressée dans des fichiers de type $F1) dans la fiche de ce numéro.

Vous utiliserez encore le paramètre T pour obtenir un catalogue sélectif. Si vous tapez CATALOG,TBIN seuls les titres des fichiers binaires seront affichés. Vous pouvez aussi employer les autres types (TXT, BAS, VAR, REL, SYS) et il est même possible de créer vos propres types mais si vous en êtes là, vous n'avez plus besoin de mes services !

En tout état de cause, l'utilisation de TTXT ne s'imposait pas dans l'exemple ci-dessus car OPEN s'adresse par défaut à un fichier du type TXT.

LES IMPRIMANTES

• ***Peut-on revenir en arrière dans un programme quand on est entré dans une fonction d'impression sans avoir allumé l'imprimante ?***

A ma connaissance, c'est NON, mais avec certaines imprimantes ce n'est pas nécessaire : il suffit d'allumer l'imprimante et tout fonctionne normalement.

• ***Dans le cas de l'IMAGEWRITER + carte Super Série est-il possible de détecter l'état de l'imprimante (allumée ou pas) par un POKE ?***

C'est presque la même question et la réponse est encore "non", MAIS le problème se situe au niveau de l'interface car la carte ALPHABITS II de STREET ELECTRONICS envoie sur écran le message WAITING ON PRINTER si ladite imprimante n'est pas raccordée à la carte, si elle n'est pas alimentée ou allumée, si enfin elle est désactivée (voyant SELECT éteint). Je sais de quoi je parle car mon IMW 2 vient de me quitter 15 jours après sa livraison pour un problème de réglage mécanique, et que bien entendu, une fois sur deux je fais PR£1 alors que ma carte série alimente actuellement l'air chaud de VALLAURIS. ■

L'APPLE *IIc* et l'imprimante

Voici un élément de réponse au problème posé par les lecteurs utilisant une imprimante après le chargement d'un écran 80 colonnes (*T.M.* n°9, page 45 : Ecran 80 colonnes). Il s'agit encore des secrets des ports séries de l'Apple *IIc* !

Le programme de sauvegarde et celui de chargement d'écrans 80 colonnes touchent à toute la zone mémoire d'écran, d'adresse $400 à $7F7, aussi bien en mémoire principale qu'en mémoire auxiliaire. Mais ces zones contiennent aussi les fameux "trous d'écran", séries de huit octets placés entre chaque groupe de trois lignes et bien sûr non affichés, mais très utilisés

- En page principale, ces octets sont utilisés comme registres temporaires par les programmes des "slots" (ports séries, gestionnaires de disque, souris), aussi bien par l'Apple *IIe* que le *IIc*.
- En mémoire auxiliaire, l'Apple *IIc* utilise ces octets pour stocker la configuration par défaut des ports séries (tandis que la configuration est imposée par des interrupteurs mécaniques sur la carte série de l'Apple *IIe*).

Donc, quand on charge un écran 80 colonnes sur un Apple *IIc*, ces trous d'écrans sont remplacés par ceux qui existaient dans la machine d'où provient l'image : ça peut être n'importe quoi ! C'est pourquoi à l'utilisation suivante de l'imprimante, rien ne va plus...

ÉCRANS 80 COLONNES

- *Suite à votre article "créer des écrans sur 80 colonnes", paru dans le numéro 3 de T.M. et ayant utilisé récemment dans un logiciel des écrans créés grâce à votre utilitaire, nous avons constaté un petit problème concernant la sortie sur Imagewriter (I ou II) à partir d'un Apple IIc. Alors que tout fonctionne avant le chargement d'un écran 80 colonnes, notre imprimante refuse d'imprimer quoi que ce soit après ledit chargement. Vous pouvez d'ailleurs essayer de charger un écran à partir de votre utilitaire, puis interrompre ce dernier et faire l'habituel PR£1. Le résultat est surprenant. Nous avons d'ailleurs eu le même problème sur deux Apple IIc différents : cela ne provient donc pas d'une défaillance ou des connexions. Notez que cet ennui ne se pose pas avec un IIe (équipé d'un 6502). Cela proviendrait-il de la gestion mem.principale/mem.auxiliaire du 65C02, vu que l'écran 80 colonnes se décompose en deux parties ?* **Jean-Luc M et Lionel Z.**

Les remèdes :

Avant d'utiliser l'imprimante :

- Le plus élégant (et le plus logique) : ne sauver et ne charger que les lignes d'écran, et non pas toute la zone mémoire concernée : cela nécessite un programme qui déplace ces lignes vers une zone tampon puis sauve cette zone tampon ; et réciproquement au chargement : c'est ce que réalise la routine MOVER80 d'Yvan KOENIG (voir plus loin).
- Le plus immédiat : après l'ouverture du port ("PR£1" en BASIC), reconfigurer les ports par les commandes CTRL-I... (voir plus loin) : ça marche, mais ce n'est pas très pratique !
- Efficace aussi : utiliser le programme CONFIG modifié (BRUN CONFIG2,A$8000) qui replace les valeurs en mémoire auxiliaire (il faut le faire après le chargement de l'écran 80 colonnes, bien sûr). C.F. T.M. n°9 pages 53-54. Il faut cependant le modifier légèrement pour qu'il n'affiche rien en plus sur l'écran : il suffit de mettre $60 (RTS) en $8065 pour annuler la partie message : soit "BLOAD CONFIG, A$8000", puis POKE 32869,96, puis sauver après modification par "BSAVE CONFIG2, A$8000,L$FF".

Mais pourquoi ces voyages ?

Pour l'Apple *IIc*, les caractéristiques des ports sont mémorisés en plusieurs endroits, ce qui nécessite des éclaircissements :

A la mise sous tension :
Des caractéristiques prédéfinies (ce que j'appelle configuration par défaut) sont recopiées de la ROM vers les trous d'écran de la mémoire auxiliaire (ce qu'on peut faire par CALL 64734). *(suite page 58)*

En utilisant les utilitaires système (ou le programme CONFIG) :
Les caractéristiques des trous d'écran en mémoire auxiliaire sont remises à jour.

En faisant Pomme ouverte-Control-Reset :
Les trous d'écran en mémoire auxiliaire ne sont pas affectés (ce qui permet d'utiliser un programme d'application, ou de redémarrer sous DOS 3.3 ou Pascal avec les ports déjà réglés).

En ouvrant un port par "PR£1" ou "PR£2" ou "IN£1" ou "IN£2" :
Les caractéristiques contenues dans les trous d'écran en mémoire auxiliaire sont recopiés dans les trous d'écran en mémoire principale.

En utilisant les commandes Control-I etc... décrites ci-dessous :
Les caractéristiques contenues dans les trous d'écran en mémoire principale sont modifiées.

Pendant le fonctionnement du port :
Ce sont les caractéristiques contenues dans les trous d'écran en mémoire principale qui sont utilisées.

Ouf ! c'est long à lire, mais ce n'est pas si compliqué qu'il n'y paraît, et bien utile pour comprendre les difficultés !

Commandes de configuration de port : (traduit du manuel de référence)

Tout d'abord ouvrir le port : Contrôle-D (si le Dos ou ProDOS est en MEV), suivi de PR£1 (pour le port 1) ou PR£2 (pour le port 2). Puis chaque commande doit être précédée de CTRL-I (ou la touche de tabulation) (pour le port 1) ou CTRL-A (pour le port 2) : Le curseur devient un point d'interrogation clignotant.

Liste des commandes :

nnn Largeur de ligne : un retour chariot est généré automatiquement après nnn caractères. Cette commande doit être suivie de N ou d'un retour chariot.

nnB Met la vitesse en Baud correspondant à nn :

nn	Baud	nn	Baud	nn	Baud
1	50	6	300	11	3600
2	75	7	600	12	4800
3	110	8	1200	13	7200
4	135	9	1800	14	9600
5	150	10	2400	15	19200

nD Formate les données suivant la valeur de n :

n	Bits de données	Bits de stops	n	Bits de données	Bits de stops
0	8	1	4	8	2
1	7	1	5	7	2
2	6	1	6	6	2
3	5	1	7	5	2

I Echo : les caractères envoyés sont répétés à l'écran.

K Pas de saut de ligne "Line Feed" automatique après un retour chariot.

L Génère automatiquement un saut de ligne après un retour chariot.

nnnN Change la largeur de ligne (de 1 à 255) ; enlève l'écho éventuel. N.B. : ØN ne supprime pas la génération automatique du retour chariot : il suffit, pour cela, soit d'utiliser les utilitaires-système, soit de mettre Ø en mémoire $579.

nP Bit de parité suivant la valeur de n :
Ø, 2, 4 ou 6 : Pas de parité.
1 : Parité impaire "Odd".
3 : Parité paire "Even".
5 : Bit "mark" (1).
7 : Bit "space" (Ø).

R Réinitialise le port 1 (à partir des valeurs par défaut des trous d'écrans en mémoire auxiliaire) et le ferme.

S Envoie un BREAK de 233 ms.

Z Les caractères de contrôle qui suivent seront ignorés ; la sortie non formatée ; pas de génération automatique des retours chariot. Il faut fermer puis réouvrir le port pour sortir de cette commande.

CTRL-W (W ou n'importe quelle lettre) pour remplacer le caractère de commande (Control-I ou Control-A) en le faisant immédiatement suivre par le nouveau caractère de contrôle désiré.

Et voilà quelques mystères du IIc dévoilés : on regrette seulement que la documentation fournie avec l'appareil n'en parle pas. Il reste encore le fonctionnement des ports en entrée "IN£..", mais ça c'est pour une autre fois !

Et voici les routines d'Yvan KOENIG :

DOSMOVER.80 et PROMOVER.80

Le problème signalé par Jean-Luc M. (page 45 du n°9) au sujet des écrans 80 colonnes et du *IIc* est très classique. APPLE en fait état dans plusieurs documents.

Un écran 80 colonnes est en fait composé de deux écrans 40 colonnes imbriqués. Chacun de ces écrans a la structure classique, à savoir 8 groupes de 3 * 40 = 120 octets consécutifs ce qui laisse 8 groupes de 8 octets inutilisés... par l'affichage.

Ce gaspillage ne pouvant être accepté, APPLE a très rapidement trouvé divers emplois à ces précieux octets.

Sur les machines ouvertes, on y stocke des informations concernant les périphériques. Si on charge directement un écran créé dans un environnement différent, on peut avoir des surprises ou, tout au moins, des bruits au niveau du lecteur.

Sur le *IIc*, la situation peut devenir plus grave. En effet, certains des octets en cause concernent la configuration mémoire, ce qui risque de provoquer un plantage intégral (c.f. l'article de Luc MOREAU).

C'est pour cela que les auteurs de ProDOS ont pris soin de "protéger" l'écran par le biais de la BITMAP. Le poke 48984,0 (48984 = $BF58) que mentionne *Tremplin Micro* dans les versions ProDOS, supprime la protection BITMAP mais ne saurait pallier les inconvénients d'une perturbation des pointeurs.

La seule solution correcte est de charger le fichier écran en mémoire "normale", puis d'effectuer un transfert. C'est ce que j'ai fait pour DHGR. EPSON et c'est ce que font les programmes DOSMØVER8Ø et PROMØVER8Ø pour les écrans en mode texte.

Avec xxxMØVER8Ø, CALL 768 initialise un buffer de 4 pages sous les buffers du DOS sous DOS 3.3, entre les buffers et ProDOS sous ce système.

CALL 771 copie le contenu de l'écran PAGE2 dans le buffer d'où vous pouvez le sauver sur disque (ou sur /RAM), CALL 774 copie l'écran PAGE1 dans le buffer d'où vous pourrez aussi le sauver. ProDOS permet de n'utiliser qu'un fichier pour les deux moitiés d'écran (c.f. les programmes TST.xxx).

Pour rétablir l'écran, un BLOAD ramène l'écran page 2 dans le buffer et CALL 782 le transfère dans l'écran PAGE2. Ensuite, un BLOAD ramène l'écran page1 dans le buffer et CALL 785 le redépose dans PAGE1.

Dans TST.PROMØVER.8Ø je fais appel à PRO.FP qui, à l'instar de la commande FP du DOS 3.3, remet en place tous les pointeurs et, ce qui est nouveau, remet les buffers dans leur position normale ce qui évite que des initialisations successives de routines comme PROMØVER.8Ø ne réduisent la mémoire disponible à zéro.

TST.DOSMOVER.80

```
10  REM "TST.DOSMØVER.8Ø                                              B0E9
20  TEXT : HOME : PRINT CHR$ (21)                                      293D
30  PRINT CHR$ (4) "BLOAD DOSMØVER.8Ø" : CALL 768                      720F
35  BUF = 256 * (4 + PEEK (814)): D$ = CHR$ (4)                        305E
40  PRINT D$"PR£3"                                                     8CC2
50  FOR I = 1 TO 22 : HTAB I: PRINT "REMPLISSAGE ECRAN": NEXT          166E
60  CALL 771: PRINT D$" BSAVEECRAN2,L$3F8,A"BUF                        C970
65  CALL 774: PRINT D$"BSAVEECRAN1,L$3F8,A"BUF                         75B1
70  HOME : FOR I = 1 TO 5 : PRINT CHR$ (7) : NEXT                      3914
75  PRINT D$"BLOADECRAN2,A"BUF: CALL 782                               8116
80  PRINT D$"BLOADECRAN1,A"BUF: CALL 785
```

(Suite page 60).

DOSMOVER.80

```
300 : 4C 41 03 A0 55 2C A0 54 A2 28 A9 60 D0 09 A0 55   A446
310 : 2C A0 54 A2 60 A9 28 8D 36 03 8E 34 03 8C 21 03   102E
320 : 2C 55 C0 A2 17 8A 20 C1 FB 85 60 A5 29 69 8E 85   2F8F
330 : 61 A0 27 B1 60 91 28 88 10 F9 CA 10 E8 2C 54 C0   F585
340 : 60 A5 74 38 E9 04 85 51 A9 00 85 50 20 8E F2 EA   F17C
350 : 38 E9 04 8D 2E 03 60                              0643
```

BSAVE DOSMOVER.80,A768,L87

TST.PROMOVER.80

```
10  REM "TST.PROMOVER.80                                          
15  D$ = CHR$ (4) : IF PEEK (116) — 150 THEN PRINT D$"—PRO.FP" :  C009
    GOTO 30
20  TEXT : HOME : PRINT CHR$ (21)                                 B0E9
30  PRINT D$"BLOAD PROMOVER.80" : CALL 768                        F644
35  BUF = 256 * (4 + PEEK (814))                                  C431
40  PRINT D$"PR£3"                                                305E
50  FOR I = 1 TO 22 : HTAB I: PRINT "REMPLISSAGE ECRAN": NEXT     8CC2
60  CALL 771: PRINT D$" BSAVEECRAN,L$400,A"BUF                    C51F
65  CALL 774: PRINT D$"BSAVEECRAN,L$3F8,B$400,A"BUF               C565
70  HOME : FOR I = 1 TO 5 : PRINT CHR$ (7) : NEXT                 75B1
75  PRINT D$"BLOADECRAN,L$3F8,A"BUF: CALL 782                     262F
80  PRINT D$"BLOADECRAN,L$3F8,B$400,A"BUF: CALL 785               CA9E
```

PROMOVER.80

```
300 : 4C 41 03 A0 55 2C A0 54 A2 28 A9 60 D0 09 A0 55   A446
310 : 2C A0 54 A2 60 A9 28 8D 36 03 8E 34 03 8C 21 03   102E
320 : 2C 55 C0 A2 17 8A 20 C1 FB 85 60 A5 29 69 8E 85   2F8F
330 : 61 A0 27 B1 60 91 28 88 10 F9 CA 10 E8 2C 54 C0   F585
340 : 60 A9 04 20 F5 BE 90 08 A9 0C 20 0C BE 4C D0 03   6A36
350 : 38 E9 04 8D 2E 03 60                              0643
```

BSAVE PROMOVER.80,A768,L87

PRO.FP (cette routine est aussi utilisée par ailleurs ; voir fiche n°11).

```
300 : 20 F8 BE A9 9E 8D 07 BE A9 BE 8D 08 BE A9 03 8D   CF62
310 : F6 03 A9 BE 8D F7 03 A9 08 85 68 A2 01 86 67 CA   68DF
320 : 8E 00 08 8A A2 12 9D 58 BF CA D0 FA A9 3F 8D 6B   5BFC
330 : BF A9 CF 8D 58 BF A9 F0 8D 10 BE A9 FD 8D 11 BE   11D1
340 : 8D 21 BE A9 1B 8D 20 BE A9 00 8D 12 BE 8D 16 BE   5802
350 : 8D 22 BE 8D 26 BE A9 C1 8D 13 BE 8D 23 BE A9 C3   1980
360 : 8D 17 BE 8D 27 BE A9 95 20 ED FD 20 84 FE 20 2F   8B0D
370 : FB 20 58 FC A9 00 8D F2 03 A9 BE 8D F3 03 49 A5   E172
380 : 8D F4 03 A2 01 8E 9A 03 20 91 03 E8 E0 05 90 F5   5158
390 : 60 78 20 00 BF 41 99 03 60 01 01                  BSAVE PRO.FP,A768,L155
```

CHASSE AUX BOGUES

IW.TAB.HOR Vous avez eu beaucoup de chance si vous avez réussi à obtenir des tabulations avec les quelques lignes de programme parues page 58 de notre numéro 10 ! Un aimable farceur avait mal recopié la ligne 40... et oublié de tester la routine ainsi modifiée. Bref, voici ce que devrait être cette fameuse ligne 40 :

```
40  PRINT CHR$ (9)I;: NEXT
```

tremplin micro 11

Bulletin de commande et d'abonnement

Bulletin à retourner à : TREMPLIN MICRO — Senillé — 86100 CHATELLERAULT

Nom ..... Prénom .....

Adresse complète .....

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

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

QUANTITÉ	DÉSIGNATION	PRIX UNITAIRE	TOTAL
	Abonnement à 6 numéros (un an) : FRANCE à partir du numéro : _____	190 F	
	Abonnement à 6 disquettes (un an) : FRANCE à partir du numéro : _____	600 F	
	MINIE, LOGICIEL DE CLAUDE AUBRY	180 F	
	DISQUETTE UTILITY-DISK (Marcel COTTINI)	170 F	
	LIVRE : LE 65Ø2 PAS à PAS	120 F	
	LIVRE + DISQUETTE : CLINS D'ŒIL AU 65Ø2	160 F	
	RELIURE-ÉCRIN	40 F	
	DISQUETTE-INDEX (N°1 à 6)	30 F	
	SIGNATURE (CHECK-LIST) (voir page 64)	30 F	
	DISQUETTE DE TREMPLIN MICRO :* *(IIe et IIc)* N°1 ☐ N°4 ☐ N°7 ☐ N°10 ☐ N°2 ☐ N°5 ☐ N°8 ☐ N°11 ☐ N°3 ☐ N°6 ☐ N°9 ☐	105 F	
	NUMÉRO DE TREMPLIN MICRO :* N°1 ☐ N°2 ☐ N°3 ☐ N°4 ☐ N°5 ☐ N°6 ☐	30 F	
	NUMÉRO DE TREMPLIN MICRO :* N°7 ☐ N°8 ☐ N°9 ☐ N°10 ☐ N°11 ☐	33 F	

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

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

Envoi en paquet-poste recommandé à partir de 400 F

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

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

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

VISA

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

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

Montant à débiter _____ F

ÉDITIONS "TREMPLIN MICRO"

LE 6502 PAS à PAS

Deux copains se rencontrent. Le premier se met en tête d'initier le second au langage machine de l'Apple.

Le résultat : un bon petit bouquin qui vous prouvera à coup sûr que, réellement, l'assembleur du 65Ø2 (et du 65CØ2)... c'est facile !

par Bernard **GOURC** et Y. **HERBET**

En souscription : 120 F

CLINS D'ŒIL AU 6502

Vous trouverez dans ce recueil, outre des routines parues sous les signatures de Clément Renard, Nestor, Géo et Guy-Hachette, plusieurs programmes intéressants, notamment en matière de tri.

Collectif **Tremplin Micro**

En souscription : 160 F (avec la disquette)

SIGNATURE (CHECK-LIST)

Indispensable pour vérifier la saisie des programmes de *TREMPLIN MICRO* (aussi bien en BASIC qu'en LANGAGE MACHINE).

DISPONIBLE IMMÉDIATEMENT (DOS 3.3 et ProDOS) : 30 F ■

6502 - 65C02 - 68000

BASIC - ASSEMBLEUR - PASCAL

DOS - PRODOS - CP/M

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

POUR PROGRAMMER VOTRE APPLE :

Extrait du sommaire de la revue Pom's 27

- un programme d'enregistrement, de restitution et d'impression des écrans Minitel, pour diminuer les temps de connexion ;
- une alarme télématique qui alerte les correspondants par des écrans Minitel, correspondants qui peuvent 'télé-déclencher' une riposte ;
- un protocle de communication sous CP/M ;
- la réalisation d'un crayon optique et son programme de contrôle ;
- un programme d'apprentissage du Forth ;
- ...

Diversité ?

Pom's, c'est : Basic, Pascal, Assembleur 6502, 65C02, Z80, 68000, Turbo-Pascal, des feuilles de calcul pour tableurs, des astuces, des utilitaires de haut niveau utilisables par tous...

— Pom's, chez votre marchand de journaux —

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

Je désire recevoir :

- Revue Pom's 27 ..... à 45,00 F .....
- Disquette Pom's 27 ..... à 60,00 F .....

(Parution 28 novembre)

Total : .....

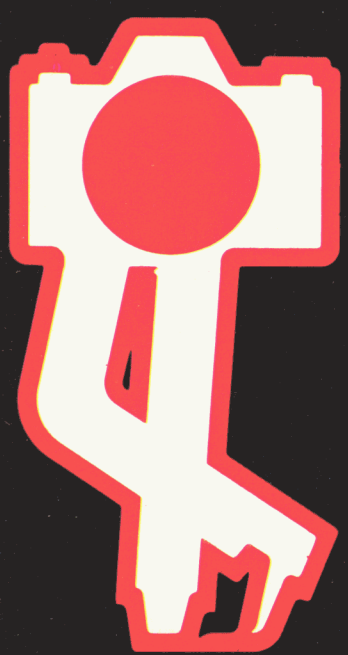
Nom, adresse :

_____

_____

Chasseur d'Images

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



Chez votre marchand de journaux !