Numero 3 - Printemps 86 - Prix 20 F - Belgique 160 FB - Suisse 6 FS.

ISSN 0753-2997

MICRO INFO

SPECIAL TRUCS ET ASTUCES

AMSTRAD, COMMODORE 64
Nouveau :
C 128, THOMSON MO5

DOSSIERS :

- un lecteur 5 pouces pour ATARI ST.
- le MEGA-JACK.

ENQUETE :

interview exclusive de Shiraz SHIFJI le père de l'ATARI ST.

A AFFICHER :

- les adresses mémoires de l'AMSTRAD CPC 6128.
- la page 0 du Commodore 128





Des performances en plus!

MICRO APPLICATION 13, rue Sainte Cécile 75009 PARIS tél. (1) 47-70-32-44

SOMMAIRE

EXCLUSIF :

une interview de Shiraz Shifji le père de l'ATARI ST ——— page 6

Le théatre des nouveautés, avec par ordre d'apparution :

MODULA-2ST, pour l'ATARI ST ——— 15
VIRGULE sur le COMMODORE 128
ML1, un assembleur pour MO5
Le basic de l'ATARI ST

Faites chauffer les fers à souder...

Un lecteur 5 pouces sur l'ATARI ST ——— 22
MEGA-JACK, ou comment doubler la capacité
mémoire de votre 520 ST

MICRO-ECHOS ——— 28

GRAPHISCOPE, une tablette pour vos micros

REALISATION :
Un convertisseur A/D et D/A pour l'AMSTRAD ——— 29

GRAPHISME 3D sur votre AMSTRAD ——— 34

La page 0 du COMMODORE 128 ——— 36

SUPER, le super compilateur 128 ——— 40

Lecteurs 1570 et 1571 : lequel choisir ? ——— 49

UN JEU, et quel jeu !
CENTIPEDE pour AMSTRAD ——— 50

25 PAGES DE TRUCS ET ASTUCES

pour AMSTRAD, COMMODORE 64 et 128, MO5 ...! ——— 60

En pages centrales : PEEK et POKE
Tableau des adresses d'implantation des CPC 664 et 6128

et au fil des pages : ALLO MA, petites annonces, ...

L'ABONNEMENT SE TROUVE A LA PAGE ——— 88

Directeur de la publication : PHILIPPE OLIVIER
REDACTEUR EN CHEF : PHILIPPE OLIVIER
REDACTEUR EN CHEF ADJOINT : FREDERIQUE BEAUDONNET

Ont participé à ce numéro : Gilles Colleu, Olivier Paulette, Alexandre Ungerer François Blancher, Pascal Séguy, Hervé Delort.

Publié par MICRO INFO - 13, Rue Sainte Cécile 75009 PARIS

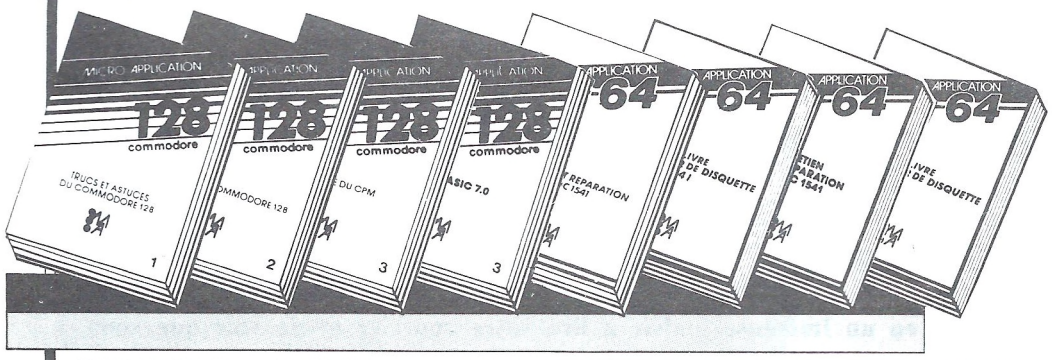
Copyright MICRO INFO. Ce numéro a été tiré à 5000 exemplaires.

MICRO

Toujours les Premiers

MICRO APPLICATION
Premier éditeur français
de livres et de logiciels
pour COMMODORE

Centre de démonstration permanente en nos locaux : 13, rue Sainte Cécile, 75009 Paris - Métro rue Montmartre Ouvert de 9 h à 13 h et de 14 h à 18 h du lundi au vendredi.



LIVRES

TRUCS ET ASTUCES DU COMMODORE 128
Plus de 300 pages.
Ce livre est un hit pour chaque utilisateur de Commodore 128 et contient des informations essentielles sur : le BANKSWITCHING et la configuration de la mémoire, la description des registres du contrôleur Vidéo et la programmation graphique en 640 x 200 sur l'écran 80 colonnes, les fenêtres, le fonctionnement en mode multi-tâches, la structure des instructions, et beaucoup de programmes exemples et utilitaires...
Disponible.
Réf. : ML 135
Prix : **149** FF

LA BIBLE DU COMMODORE 128
Plus de 600 pages.
Description complète du système, du hardware et des interfaces, explications des chips VIC, du système vidéo (640 x 200 en haute résolution avec écran de 80 colonnes sur 25 lignes), le SID, la description détaillée du MMU (Memory Management Unit) et comme toujours le listing commenté de la ROM et de nombreux programmes exemples et utilitaires à taper. Un super livre, comme toutes les BIBLES de Micro Application!
Disponible.
Réf. : ML 136
Prix : **249** FF

LE LIVRE DU CP/M
Si vous voulez entrer avec le Commodore 128 dans le monde du CP/M, voici l'ouvrage qu'il vous faut ! Ce livre en effet vous explique tout sur cet operating system en tenant compte des spécificités propres au Commodore 128, notamment le format Commodore et le format CP/M, la gestion et la programmation des données...
Indispensable pour profiter à fonds de CP/M sur le Commodore 128.

Réf. : ML137
Prix : **149** FF

BASIC 7.0
Tout, tout, tout, sur le nouveau Basic 7.0 (450 pages).
Réf : ML144
Prix : **149** FF

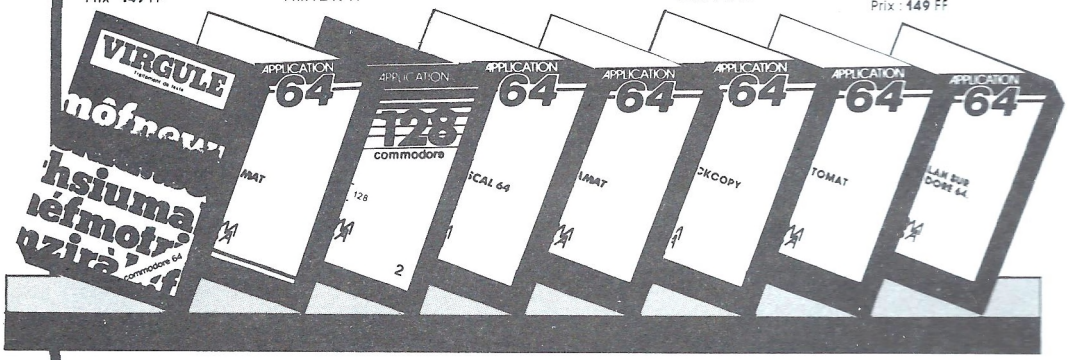
ENTRETIEN ET REPARATION DU VC 1541
Ce livre vous permettra de réparer et d'entretenir le lecteur de disquettes Commodore VC 1541. Vous y trouverez amplement décrits les principes mécaniques et électroniques de ce périphérique. Pour ceux qui ne connaissent pas l'électronique, cet ouvrage en est aussi un parfait apprentissage.
Réf : ML134
Prix : **149** FF

LIVRE DU LECTEUR DE DISQUETTE 1541
Un livre indispensable qui vous explique de façon exhaustive et claire L'utilisation du floppy Commodore 1541. Fichiers relatifs, manipulation des blocs et de la directory, gestion de fichiers complète, moniteur disque, listing du DOS commenté, spooler... Le must absolu!
Réf. : ML 101
Prix : **179** FF

LE LIVRE DU LANGAGE MACHINE Tome 1
Apprentissage et compréhension assures sur COMMODORE 64 et COMMODORE 128.
Réf : ML 105
Prix : **149** FF

LE LIVRE DU LECTEUR DE CASSETTE 1530
Un TURBO pour votre 1530.
Ce livre contient tout ce qu'il faut savoir sur le 1530 + un programme qui multiplie la vitesse d'accès par 10, rendant le 1530 aussi performant que le lecteur de disquette 1541.
INDISPENSABLE !
Prix : **99** F TTC
Réf. : ML 103

MICRO INFO N°3
SPECIAL trucs et astuces



LOGICIELS

PROFIMAT

PROFIMAT est un moniteur assembleur pour Commodore 64, compatible avec PASCAL 64. Ce logiciel comprend deux modules, un assembleur symbolique et un moniteur. Le moniteur ($C000 à $CBFF) permet les transferts de programme, de modifier l'espace mémoire, d'utiliser le mode TRACE et de placer des points d'interruptions dans un programme, pour sa mise au point.

Réf. : MD106
Prix : **350** FF

PASCAL 64

PASCAL 64 est un compilateur pour le COMMODORE 64 possédant les propriétés suivantes :
- un jeu d'instructions très complet
- permet la programmation des interruptions et est compatible avec le moniteur et l'assembleur
- crée des programmes très rapides en pur langage-machine
- ordres spécifiques pour la gestion des fichiers relatifs, du graphisme et du son
- permet l'interruption avant terme des procédures avec EXIT.

Réf. : MD108
Prix : **350** FF

DATAMAT

Gestion de fichiers facile à utiliser et extrêmement rapide. Commandé par un menu. Définition libre du masque de saisie, 50 champs par enregistrement. Jusqu'à 2000 enregistrements selon leur importance.
Recherche tout enregistrement en 2 ou 3 secondes. Sélection d'après n'importe quel champ d'impression de listes (format entièrement libre). Edition d'étiquettes.

Réf. : MD102
Prix : **350** FF

QUICKCOPY

Utilitaire de copie rapide pour COMMODORE 64
QUICKCOPY est un utilitaire de copie extrêmement rapide. Il permet de dupliquer entièrement une disquette et ceci au moyen d'un ou deux lecteurs 1541. Tous les types de fichiers sont copiés correctement (programmes, séquentiels, relatifs, user...).

Réf : MA346
Prix : **295** FF

TEXTOMAT

Entièrement commandé par un menu sur 2 lignes en bas de l'écran. Jeu de caractères français (même avec les imprimantes COMMODORE). Fonctions de calcul 24 000 caractères par texte en mémoire. Longueur de texte illimitée par concaténation.
Adaptable à presque tous les types d'imprimantes. manuel exhaustif avec cours et exercices.

Réf. MD107
Prix **350** FF

VIRGULE 128 ET 64
Premier logiciel de traitement de texte avec dictionnaire intégré, accents écran et imprimante, version COMMODORE 64 et COMMODORE 128 spécifique (80 colonnes) capacité mémoire accrue...
Réf : MA 343
Prix : **750** francs
hors taxes seulement

POWERPLAN SUR COMMODORE 64.
Premier Tableur et Logiciel Graphique INTEGRE sur Commodore 64. Très puissant et facile à utiliser.
Ref : MD103
Prix : **650** FF

BASIC 128
Compilateur pour Basic 7.0 extrêmement rapide, à posséder absolument.
Prix : **395** FF

BASIC 64 Une fusée !
Ref : MD107
Prix : **350** F TTC

Demandez notre catalogue gratuit spécial COMMODORE.
Pour alimenter votre COMMODORE 64 et COMMODORE 128.

MICRO APPLICATION
13, rue Sainte Cécile 75 009 PARIS
Tél. : (1) 47-70-32-44

Demandez notre catalogue gratuit ☐ Commodore

DESIGNATION	QUANTITE	PRIX
	TOTAL TTC	

CB date d'expiration: _____

☐ Mandat ☐ Cheque ☐ CCP.
Libellez vos chèques à l'ordre de Micro-Application.
Nom, Prenom _____
Adresse _____
Ville _____ C.P. _____
CB ☐☐☐☐ ☐☐☐ ☐☐☐ ☐☐☐
Date et signature

+ 20 F de frais d'envoi
ou 40 F pour envoi recommande.
Port gratuit pour toute commande supérieure à 250 F.

MI 86

EDITO

Micro Info, c'est votre magazine. Et, cher lecteur, qu'est-ce que vous nous réclamez avant tout ? Des trucs, des astuces, du technique et de la bidouille...

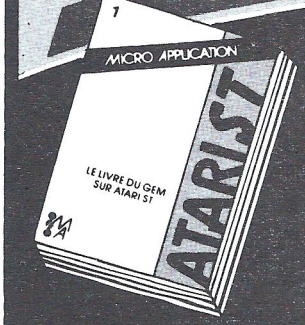
Nous avons eu un immense plaisir à lire votre courrier et de voir que vous êtes très, très, nombreux à utiliser votre matériel.
Alors nous vous avons préparé un spécial trucs et astuces, un spécial rusé ! Vous trouverez pêle-mêle des listings, des petits trucs de derrière les fagots, les premiers trucs pour le C128, les adresses mémoires du 6128 etc...

Ceux pour qui la garantie ne compte pas peuvent faire chauffer leur fer à souder et ouvrir le capot de leur ST, deux montages leurs sont proposés.

Enfin dans la série "les rusés parlent aux rusés" nous vous livrons une interview exclusive de Shiraz Shifji, le père de l'ATARI ST.
A ne pas manquer !

A bientôt !

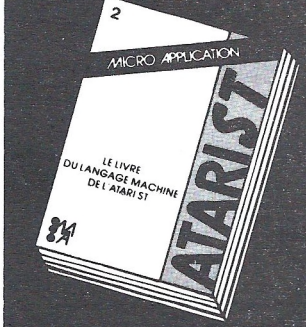
Toujours les Premiers



LE LIVRE DU GEM SUR ATARI ST
Plus de 350 pages.

Cet ouvrage contient ce qu'il est nécessaire de savoir pour utiliser GEM efficacement : Fonctionnement de la souris, Virtual Device Interface, Application Environment Services, Graphics Device Operating System. Description de routines utilisant GEM en BASIC, C et en Assembleur. Utilisation standard et spécifique de l'operating system.

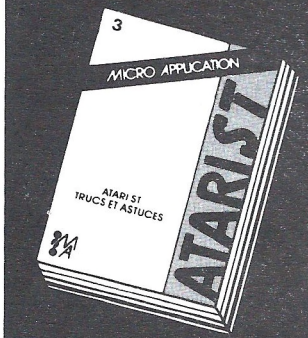
Réf. : ML139 Prix : 149 FF



LE LIVRE DU LANGAGE MACHINE DE l'ATARI ST
Plus de 250 pages.

Tout ce qu'il faut savoir pour tirer au mieux parti de votre ATARI ST : système de calcul et de bit manipulation du 68000 , utilisation des registres, structure des commandes, programmation structurée : récursion, piles, procédures et fonctions, listings sources de programmes assembleurs, routines systèmes... Un super livre !

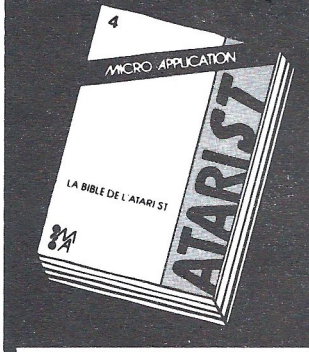
Réf. : ML141 Prix : 149 FF



ATARI ST TRUCS et ASTUCES
Plus de 250 pages.

Un recueil complet de trucs et d'astuces que votre tout nouveau ATARI ST va beaucoup apprécier ! Des graphismes fantastiques à partir de programmes en BASIC, des exemples et des conseils pour programmer en langage C et en Assembleur...

Réf. : MI 140 Prix : 149 FF



La BIBLE de l'ATARI ST
Plus de 500 pages.

Ce livre contient un ensemble complet d'informations sur l'ATARI ST, la description HARDWARE de la machine ainsi que des schémas détaillés et amplement expliqués notamment des interfaces V24, du port d'extension, de l'interface midi, la structure des graphiques, du BIOS, de GEM, les adresses systèmes importantes, le fonctionnement de la souris....

Réf. : ML142 Prix : 249 FF



LE NOUVEL ATARI ST

Ce livre décrit la superbe machine qu'est l'ATARI ST.
Architecture, interfaces, operating system, le bios, GEM, LOGO, le processeur 68000, sont quelques-uns des thèmes abordés. Ce livre doit être lu par tous ceux qui suivent de près le monde de la micro-informatique.

Réf. : ML125 Prix : 149 FF



LOGICIELS

MODULA-2 / ST
Compilateur **MODULA** pour ATARI ST.
MODULA-2 / ST est l'implantation complète du langage **MODULA** conçue par NIKLAUS WIRTH (auteur du PASCAL) sur 520 ST. Ce compilateur permet de développer des logiciels utilisant GEM et les capacités graphiques du ST. Il est livré avec un éditeur complet lié au compilateur pour une détection rapide des erreurs. Réf. : AT 001 Prix : **1450** F TTC

MICRO INFO
Pour en savoir plus ! Dans le numéro 3 :
.Interview exclusive de SHIRAZ SHIVJI le père de l'ATARI ST.
.De nombreux articles et documents.
20 Frs



MICRO-INFO n° 1 : 15 FF
MICRO-INFO n° 2 : 20 FF

MICRO APPLICATION
13, rue Sainte Cécile 75009 PARIS
tél. (1) 47-70-32-44

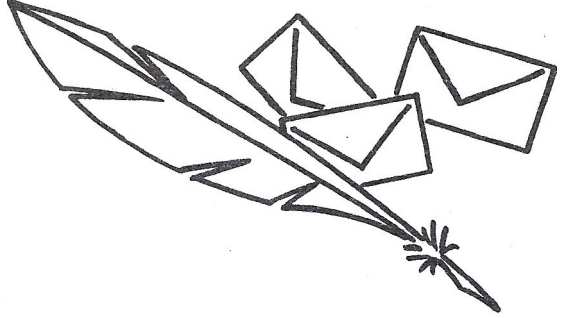
BON DE COMMANDE

DESIGNATION	QUANTITE	PRIX
	TOTAL TTC	

CB date d'expiration: _____

☐ Mandat ☐ Chèque ☐ CCP.
Libellez vos chèques à l'ordre de Micro-Application.
Nom, Prénom _____
Adresse _____
Ville _____ C.P. _____
CB □□□□ □□□ □□□ □□□
+ 20 F de frais d'envoi ou 40 F pour envoi recommandé.
Date et signature
Port gratuit pour toute commande superieure a 250 F.

ST 02 86



COURRIER

La routine trace pour Commodore 64 ne fonctionne pas.

"Un programme BASIC sans erreur lors de son premier essai cela n'existe pas !... (cf MICRO INFO N1 P. 68). La routine "Trace" qui vous était proposé n'échappe pas à la règle. En effet une erreur s'était glissée dans les dates. Il fallait lire en ligne 120 :

120 DATA 255,164,73,177,95,
201,34,208

En page 10 Paragraphe 7 (Le formattage de disquette) de ma documentation Virgule une lettre a été oubliée. Pourriez vous me la communiquer ?

Après avoir actionner la touche "F" pour passer en mode fonctionnel il faut appuyer sur la touche ">" et le caractère ">" s'affichera alors sur la ligne d'information continuelle.

Par ailleurs sur certaines versions de VIRGULE 128 pour charger VIRGULE en mode 64, il faut taper LOAD "VIRGULE SR",8 RETURN

Je n'arrive pas à charger ma disquette POWERPLAN avec un 1571. Pouvez vous m'aider ?

Certains programmes se chargeant normalement sur un lecteur de disquette 1541 ne peuvent plus être utilisés avec un lecteur de disquette 1570 ou 1571. Ceci concerne notamment les logiciels POWERPLAN, DATAMAT, TEXTOMAT, PASCAL 64, BASIC 64 et PROFIMAT.

Afin de palier à ce problème vous devez taper la commande suivante :

OPEN 15,8,15,"U0>M0" : CLOSE 15 RETURN

puis ensuite vous pouvez charger votre programme comme indiqué dans le manuel d'utilisation.

Je n'arrive pas à utiliser ma disquette SUPERPAINT proprement. Aidez moi!

Vérifiez que votre disquette SUPERPAINT n'est pas protégée contre l'écriture.

Après le chargement de SUPERPAINT, la fenêtre de travail représenté à l'écran est située au centre de la page virtuelle. Après une demande d'impression, et, si rien n'est dessiné dans le haut de la page virtuelle, il est normal que l'imprimante commence par faire des sauts de lignes représentant les parties cachées et vides de la page.

Le programme SUPERPAINT est compatible avec la DMP 2000 et les imprimantes EPSON et compatibles.

Je voudrais obtenir la qualité courrier avec mon TEXTOMAT CPC, que dois-je faire ?

Si tout le texte doit être imprimé en qualité courrier rajouter dans la table des caractères de contrôle, à la ligne initialisation, les codes 1B7801. Vous devez obtenir : Initialisation 1B401B7801

Si seulement une partie du texte est concernée, choisir deux caractères de contrôle, un pour la mise en route de la qualité courrier : code 1B7801 et un pour l'arrêt : code 1B7800

Pour arrêter les modes "indice" et "puissance", utiliser dans les deux cas CTRL S0, ce qui correspond aux codes 1B54

Ne pas utiliser TEXTOMAT avec un joystick branché sur l'ordinateur.

Je possède la cartouche "FORTH 64" de Handic. Ce langage marche bien sur le C64 mais la documentation est vraiment succincte.

Pouvez-vous m'indiquer une revue spécifique du FORTH sur C64.

Il n'y a pas de revue spécifique au FORTH sur C64, par contre de bons livres sont disponibles et en particulier chez les Editions Eyrolles.

Je t'écris pour te demander de me donner sincèrement la réponse à un problème qui m'assaille. Voici ce problème : On va m'offrir un ordinateur pour mon anniversaire mais j'hésite encore entre l'Amstrad CPC 6128 et le Commodore 128. Je voudrais que tu me conseilles lequel acheter car je n'arrive vraiment pas à me décider...

Ton anniversaire est passé et nous n'avons pas eu le temps de te conseiller. Toute l'équipe de Micro Info espère que tu es content de ton choix, écris nous si tu as d'autres questions.

Je voudrais utiliser ma cassette SPACE MOVING avec un CPC 664.

La version cassette de SPACE MOVING est utilisable seulement sur le CPC 464.
Et nous en profitons pour préciser que la version cassette de Dr WATSON est utilisable seulement sur le CPC 464 et que le programme AM COMPTA ne fonctionne que sur les CPC 664 et 6128.

EXCLUSIF, INTERVIEW DE SHIRAZ SHIFJI, LE PERE DE L'ATARI ST

Micro Info présente une interview exclusive de Shiraz Shifji le père de l'ATARI ST.

Shiraz est le père de la grande famille des ST. Nos collègues allemands de DATA WELT l'ont rencontré dernièrement et ont pu l'interviewer pendant plusieurs heures. Il se révélera être un homme sympathique et plein d'humour, ouvert à toutes les questions, mêmes celles qui touchent à certains aspects délicats. Cet article reprend les déclarations les plus intéressantes de cette rencontre exceptionnelle.

Participèrent à cette interview : Dr Achim Becker et Rainer Bartel de DATA WELT ainsi que Rolf Brueckmann et Lothar English, auteurs de livres sur l'Atari ST.

SHIRAZ SHIFJI...
SUR LUI MEME





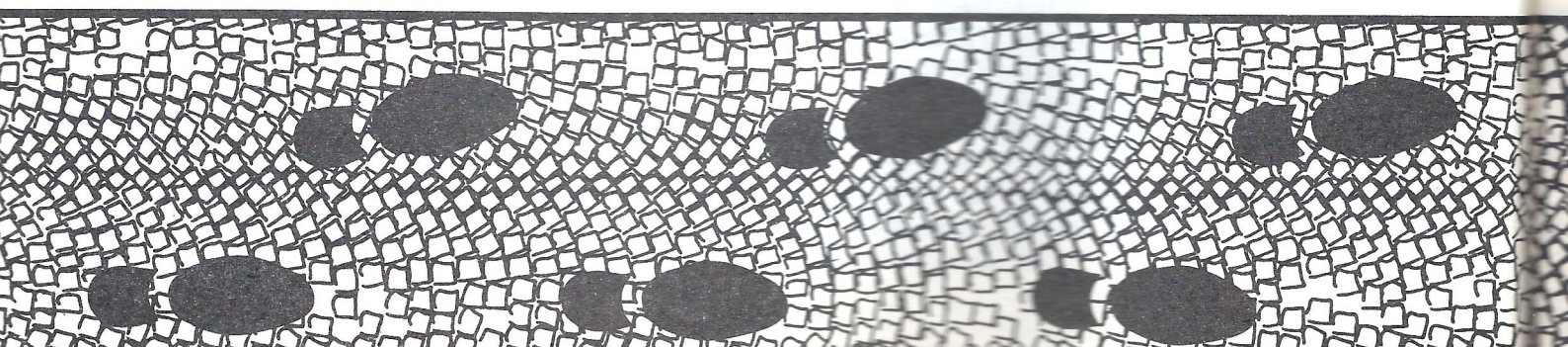
Q?: Nous avons voulu réaliser cette interview pour démontrer qu'un ordinateur n'est pas qu'une simple machine, mais que c'est avant tout l'aboutissement dû travail d'une équipe d'hommes. Nous voulions faire connaître le "Père du ST" à tous ceux qui possèdent un ST, ainsi qu'à ceux désirent qui en acquérir un.

Shiraz Shifji : Tout d'abord je voudrais dire que je ne suis pas le seul qui ait travaillé sur le ST, je parlerai donc également de mon équipe.
Je suis né au Tanganyika, dans une petite ville, du nom de Iringa. Mon père y est déjà né et mon grand-père arriva d'Inde à l'âge de 10 ans. A l'époque le Tanganyika était une colonie allemande. Ma famille fut la première à développer quelque chose qui ressemblât à une civilisation avec une ferme, des magasins, etc... Des pionniers en quelque sorte.
J'ai fait mes études en Grande Bretagne. Après le premier examen je suis allé à Standford (université de San Francisco) aux USA pour y passer un doctorat. Pour diverses raisons j'ai interrompu ces études et j'ai commencé à étudier l'électronique.

DW : Quand avez-vous arrêté vos études à Stanford ?

Shiraz Shivji : Environ en 1972/73.

DW : Donc à une époque à laquelle il n'existait pas encore de micro-ordinateurs.



Shiraz Shivji : Exact. A l'époque les premières grandes firmes commençaient à naître dans la Silicon Valley. Il y avait entre autres National Semiçonductor et Intel. AMI existait déjà et Hewlett Packard était même déjà assez important. En ce temps, la Silicon Valley était encore une région essentiellement fermière.

En 1974 je fis la connaissance de Jack Tramiel. A l'origine je fus engagé comme conseiller, Commodore avait justement quelques problèmes avec les micro-ports d'une puce qu'ils avaient développée eux-mêmes, une sorte de microprocesseur en technologie PMOS. Ce circuit travaillait de façon sérielle, il était donc plus lent qu'un circuit NMOS. Par contre la partie calcul en BCD était assez performante ce qui le désignait tout particulièrement pour les calculatrices de poche (ce que fabriquait Commodore à l'époque). Les calculatrices électroniques ont d'ailleurs eues une influence très importante sur le développement des microprocesseurs. Le premier microprocesseur fut le 4004, il avait été développé par la société japonaise Busicom, justement pour une calculatrice. L'ingénieur qui en a conçu l'architecture s'appelle Ted Hoff.

J'ai fait sa connaissance à Stanford où il donnait des cours avec le professeur Widrop, qui occupait lui-même la 12ième place chez Intel. Plus tard Ted Hoff travailla chez ATARI, aujourd'hui il enseigne à l'université. Le monde informatique est petit !

La même année je débutai chez COMMODORE, j'en suis reparti assez rapidement. Avec deux autres, nous avions pour mission de concevoir un micro-ordinateur construit autour du 6800 ; c'était longtemps avant le PET. Mais nous avons préféré le 6502 de chez MOS, beaucoup moins cher. De plus le bruit courait que c'était une copie du 6800. Tramiel a ensuite racheté MOS Technology, pour une bouchée de pain. Ce fut une excellente opération ! C'est d'ailleurs comme cela que Chuck Peddle nous a rejoint (a).

DW : Alwin Stumpf (b) a dit de vous que vous étiez "le Père du CBM 64".

Shiraz Shifji : Cela est très exagéré. En fait je n'étais qu'un concepteur parmi d'autres. J'étais effectivement responsable du système d'exploitation.

DW : Vous n'êtes donc pas exclusivement orienté Hard ?

Shiraz Shifji : Aujourd'hui je m'occupe plus du côté "hardware". De toute façon, quand on développe un ordinateur il faut toujours garder un oeil sur les deux aspects ; matériel et logiciel sont étroitement liés.

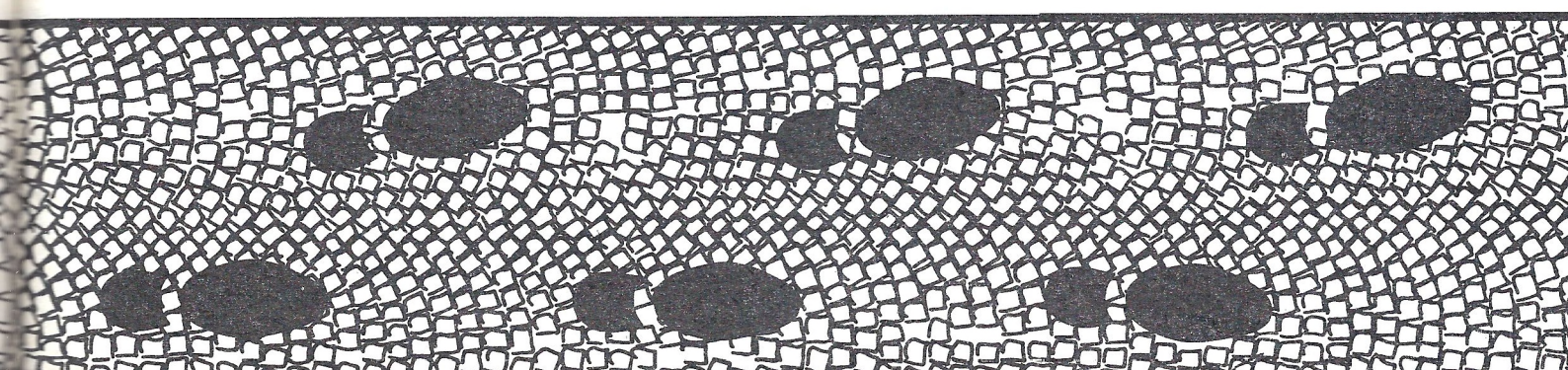
LES ORIGINES DU ST

Shiraz Shifji : Ok, Jack Tramiel quitta COMMODORE en janvier 1984 et moi en mai. A l'époque je parlai souvent avec Jack de projets futurs. Notre grande idée était de réaliser quelque chose autour du 32016 de chez National .

Tramiel avait fondé la société TTL (Tramiel Technology Ltd.) et j'y travaillai avec deux autres ingénieurs également partis de chez COMMODORE. Tramiel négociait avec la Warner Bros au sujet du rachat d'Atari, mais rien n'était encore réglé. En conséquence nous n'avions rien : pas de locaux, pas d'outils, rien. C'est chez moi que nous avons développé une machine basée autour du 32016. Il ne s'agissait pas seulement d'une préétude et de vagues croquis, non, le produit était complètement conçu. Le ST a d'ailleurs hérité du clavier que nous avions dessiné à l'époque. J'habitais alors en Pennsylvanie.

Le 2 ou 3 juillet, Jack Tramiel avait acheté Atari. Nous déménagions donc vers la côte ouest pour reprendre la boutique. Il y régnait un désordre complet : environ cinq équipes travaillaient chacune pour elle même, sans aucune coordination. Lorsqu'au bout de 6 mois ils commençaient à bien maîtriser le problème, le projet était en général abandonné.

Entre autres ils avaient deux projets en cours faisant appel aux circuits de l'Amiga. Pour ces projets, qui s'appelaient Mickey et Minnie, Atari avait donné de l'argent à Amiga (la société Amiga). Le contrat était pour le moins bizarre. La première année, Atari n'avait le droit de sortir qu'une console de jeu sur base d'Amiga, en outre la capacité mémoire ne devait pas dépasser 128K.



ENQUETE

Curieux, non ? De toute façon le design de Mickey et de Minnie ne nous plaisait pas.
A cela s'ajoutaient les querelles avec COMMODORE à propos des droits sur les circuits Amiga, nous avons donc annulé le contrat. Vous savez, Amiga nous a rendu l'argent, argent qu'ils ont d'ailleurs emprunté chez COMMODORE.

LES ENNUIS AVEC COMMODORE

Shiraz Shifji : Notre premier souci fut de faire le ménage chez Atari. Je suis retourné en Pennsylvanie pour organiser mon déménagement. C'était le 13 juillet, un vendredi. Pendant le week-end j'étais plongé dans mes affaires et ce n'est que le mardi que je découvris, au milieu du courrier, une lettre comme quoi COMMODORE m'assignait en justice, et avec moi mes trois collègues.
Imaginez un peu : La grande société multinationale COMMODORE se venge sur 4 ingénieurs parce qu'ils ont démissionné. Le choc fut particulièrement brutal pour mes jeunes collaborateurs ; de jeunes ingénieurs qui se retrouvent soudain devant un tribunal...
Vous savez qui est le vrai père de l'Amiga ? Il s'appelle Jay Minor. Il a longtemps travaillé pour Atari pour laquelle il a entre autres développé deux brevets, utilisés dans l'Atari 800XL. L'un deux a un rapport avec la gestion des sprites. Lorsque l'on examine les documents de l'Amiga, on s'aperçoit que le schéma synoptique comporte une partie absolument identique au brevet en question. C'est un fait et je peux le prouver. Le procès à propos des violations de brevets de COMMODORE est encore en cours. On se l'imagine un peu : on retrouve dans l'Amiga le même schéma que dans le 800XL.

L'EQUIPE

Shiraz Shifji : Les autres membres de mon équipe sont Doug Renn, John Hoenig et Art Morgan. Doug et Art ont environ 23 ans, John en a 28. Moi-même j'aurai 38 ans ce mois-ci. Doug est l'homme le plus important de l'équipe, mais aussi une des personnes les plus sympathiques que je connaisse. De plus c'est le meilleur ingénieur que j'aie jamais rencontré.

C'est lui qui était responsable du Video Shift Register. J'ai fait sa connaissance à l'université de Braxton où j'enseignais. A l'époque il travaillait pour son examen sur le sujet "traitement de l'image". Il m'a plu et je lui ai demandé de me rejoindre. Art Morgan a été engagé par un de mes collaborateurs.

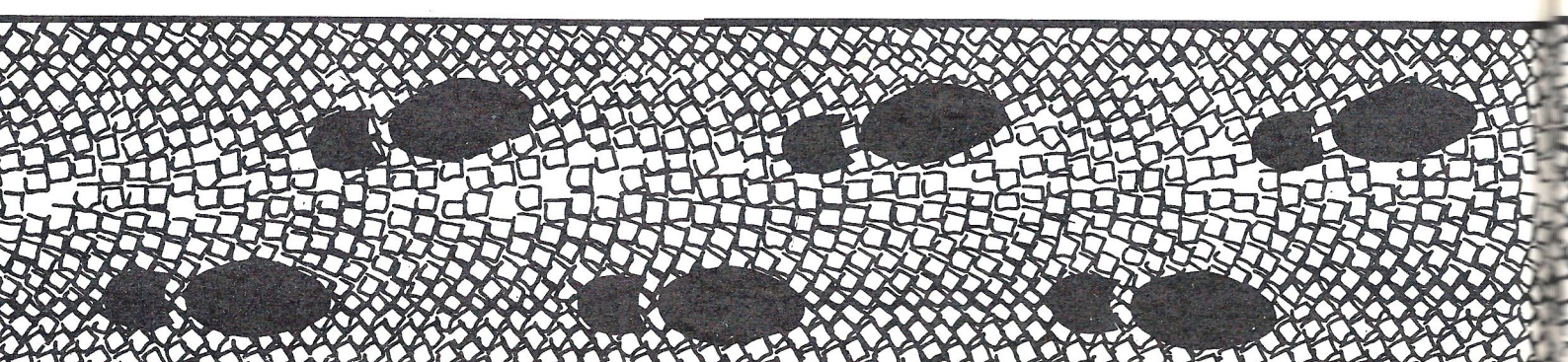
Dans l'équipe on trouve également John Tittsler qui travaillait déjà chez Atari, et John Horton de COMMODORE. Doug est en quelque sorte le chef d'équipe, en fait il est responsable de la coordination. A côté du Video Shifter il a également travaillé sur le Glue. John Hoenig a conçu le contrôleur DMA tandis que John Horton et Jim Tittsler s'occupaient des simulations et des différents autres éléments.

LE TRAVAIL D'EQUIPE

Shiraz Shifji : Nous avons commencé en juillet 84, ou plus exactement fin juillet. Les gars n'étaient pas encore installés chez Atari. Art et Doug faisaient régulièrement le voyage, John Hoenig, qui habitait la côte Est, venait de temps en temps pour une semaine ou deux. John Horton - il était auparavant chez COMMODORE Texas - habitait encore là-bas...

DW : Il n'était donc pas question d'un travail d'équipe ?

Shiraz Shifji : Oh, ce n'était pas si difficile. Vous savez, moi-même je ne me rend pas tous les jours au labo pour voir ce que chacun fait. Je ne dis pas non plus : aujourd'hui tu feras ceci ou cela. Chacun sait ce qu'il a à faire, c'est pourquoi il suffit que nous nous voyions de temps en temps.



LA CONCEPTION DU ST

Shiraz Shifji : Nous avons parlé avec National et avec Digital Research au sujet de la CPU. National n'avait pas assez de circuits pour nous soutenir, de plus ils étaient trop chers. C'est à ce moment que nous avons opté pour le MOTOROLA 68000.

J'ai encore la copie d'une note à Sam Tramiel, avec doubles pour Jack Tramiel et les autres personnes d'Atari ; elle date du 23 juillet 1984 et donne un aperçu de tous les composants disponibles, parce que Jack Tramiel m'avait chargé d'étudier le moyen de réaliser le plus simplement et le plus rapidement possible un ordinateur, sans tomber dans l'extrême qui consiste à piocher à droite et à gauche parmi des éléments qui existent déjà.
Vous savez, je suis un homme pratique. J'ai regardé autour de moi pour voir les circuits existants. J'ai examiné les circuits de l'Amiga, je me suis renseigné chez Texas Instruments et chez General Industries, etc... Il y avait par exemple une nouvelle société à Chicago qui venait de sortir de nouvelles puces, ou encore NCR avec de fantastiques circuits graphiques. Nous avons tout regardé à la loupe et à chaque fois il y avait un point qui ne nous plaisait pas. Nous nous sommes donc fixé un emploi du temps; le but à atteindre était de pouvoir présenter un produit fini au CES en janvier 1985. C'est à peine croyable mais nous avons réussi à respecter ce que nous nous étions fixé. La production n'a commencé que deux mois après la date prévue, mais la seule raison à cela était la formidable rapidité de la machine. Il y a eu de très gros problèmes dus à la haute fréquence et les circuits n'ont pu être terminés à temps.

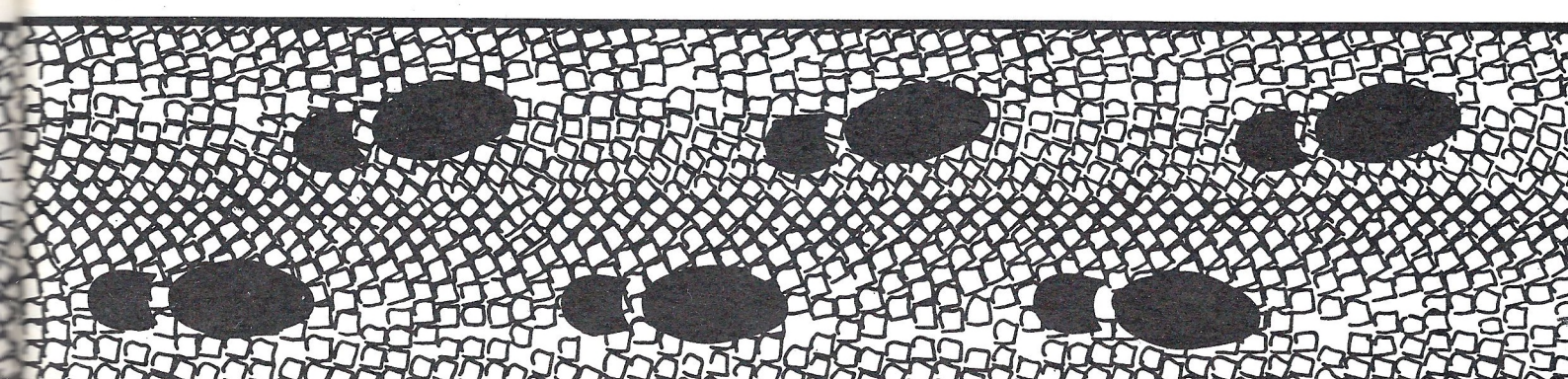
DW : Vous aviez vraiment espéré finir dans les temps ?

Shiraz Shifji : En tous cas nous avons travaillé dur pour y arriver. Un exemple : le 31 décembre 1984 nous étions au travail, jusqu'à minuit passé. Et il n'y avait pas que moi, tous mes collaborateurs y étaient.
Pour le contrôleur DMA, il est probable qu'aucun circuit n'ait jamais été conçu aussi rapidement. Au départ nous avions trois sociétés qui voulaient le réaliser pour nous. Deux l'ont conçu comme une cellule standard alors que la troisième le développa sous forme de Gate Array. Ce dernier n'a jamais fonctionné, et l'une des deux autres société abandonna également. Le troisième devait être terminé la semaine avant Noël mais ça n'a pas fonctionné non plus. Nous devions donc repartir à zéro. Nous avions six personnes qui ont travaillé 24 heures sur 24 sur le circuit. Résultat : en 14 jours la puce était terminée, en Gate Array. Pour être franc, le contrôleur ne fonctionnait pas encore à 100 % lorsque nous exposions au CES 85, mais ça allait à peu près.

DW : Vous avez donc commencé fin juillet 1984 ?

Shiraz Shifji : Pas exactement. Nous disposions déjà de certains éléments ; par exemple John Tittsler, qui travaillait déjà chez Atari, s'était construit lui-même un ordinateur portable autour du 68000, avec affichage à cristaux liquides et -en effet c'est un fan de FORTH- un langage FORTH complètement opérationnel. Nous-mêmes avons rapporté le clavier de Pennsylvanie. John a transplanté très tôt son FORTH sur le premier ST cablé, c'est pourquoi en l'espace d'un mois nous dispo-sions d'un matériel qui marchait.

De plus nous nous sommes beaucoup servi de CAO, sur un Cray avec des terminaux graphiques Mentor. Chaque membre de l'équipe pouvait disposer d'un tel terminal. A chaque fois qu'un élément ne fonctionnait pas correctement nous faisions une simulation. Comme tous les schémas étaient stockés dans l'outil de CAO, il nous était très facile de recabler les parties en cause. Ce même outil nous permettait également de commander une machine à cabler automatique.



ENQUETE

Avant qu'un circuit ne soit fabriqué en série, nous en avons réalisé deux versions : une sous forme de cellule standard et une autre en Gate Array. Théoriquement, et tous les fabricants vous le diront, une cellule standard a des dimensions inférieures au Gate Array ; nous avons constaté que dans tous les cas c'était le contraire, en parfaite contradiction avec la théorie.

Encore un exemple : les DMA, Glue et MCU étaient prévus pour être réalisés à l'aide de circuits CMOS 3 microns. Or 3 microns pour le video Shifter auraient été un peu trop lents, nous avons donc pris 2 microns. On nous avait prédit que nous n'arriverions pas à réaliser le Video Shifter tournant à 32 MHz de cette manière et que les CMOS ne travailleraient plus à cette fréquence, mais nous avons tout simulé.

Chez Atari on disposait de tous les outils nécessaires : nous avions plusieurs VAX 11/780 avec quantité de logiciels, c'est là-dessus que nous avons fait nos simulations. A ce propos : le Video Shifter fonctionna du premier coup.

LA PHILOSOPHIE DU ST

DW : quelles étaient vos motivations ? Etait-ce le fruit de longues années de pratique ? Quel rôle a joué le futur utilisateur dans vos choix ?

Shiraz Shifji : Vous savez, normalement on étudie un projet pendant six mois avant de commencer. Nous par contre savions exactement ce que nous voulions.
Tout repose sur l'expérience...

DW : Une expérience datant du PET et du CBM 64 ?

Shiraz Shifji : Exactement. Nous nous sommes posé la question : que souhaiterions nous en tant qu'utilisateur ? C'est également la raison pour laquelle nous nous ommes beaucoup préocupé de l'aspect iciel. Art Morgan et John Titster sont des spécialistes du logiciel, mais les autres sont également des programmeurs de très grande qualité, peut-être meilleurs que beaucoup de personnes qui ne font que ça ; ils programment en langage C, connaissent parfaitement UNIX, le VAX... Seulement en ce moment ils s'occupent plus du matériel.

On peut dire que nous avons eu une approche très moderne ; normalement il faut encore ajouter un an et demi pour arriver au produit fabriqué en série.
Par exemple pour le Commodore 64...

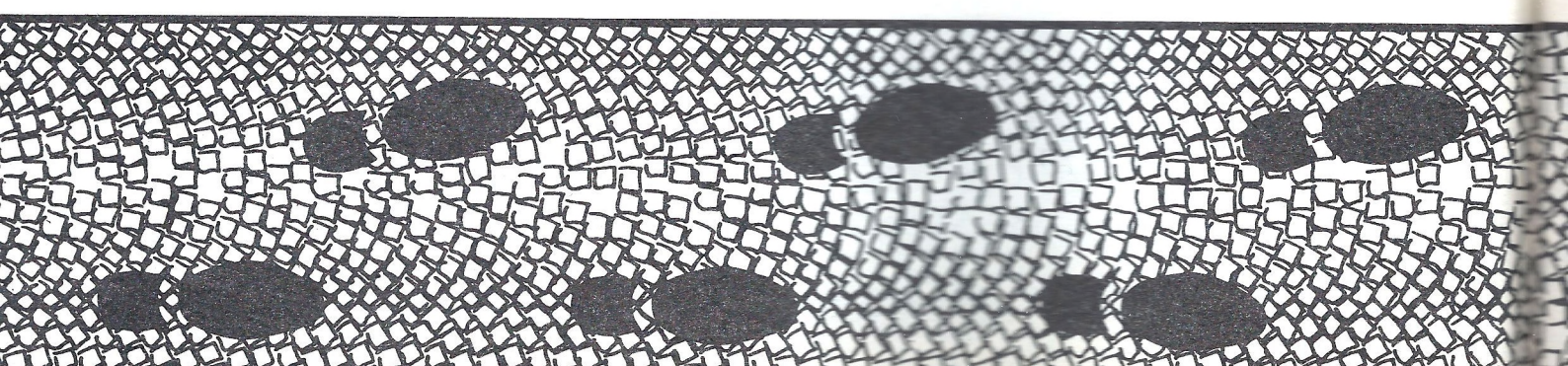
DW : Pour l'Amiga aussi, non ?

Shiraz Shifji : Là, cela a même duré 3 ans.
Notre rapidité va d'ailleurs profiter à l'utilisateur : le concept du ST autorise l'utilisation de coprocesseurs, par exemple pour le graphisme. A ce propos, cette année encore nous terminerons un coprocesseur graphique.

A l'origine ce circuit devait même être intégré, mais il était trop complexe.

DW : Si vous considérez votre approche comme très moderne, pourquoi trouve-t-on alors une telle collection de circuits démodés ?

Shiraz Shifji : Mais c'est justement ce qui fait la flexibilité du ST ! Le ST n'est pas une machine figée. Il est très simple d'adapter la version de base à l'évolution de la technique.
En fait, c'est beaucoup plus simple que par exemple sur un IBM PC où il faut retirer des vis, ouvrir la boîte et enficher des cartes. Sur le ST toutes les connexions se font au moyen de cables. Sur le PC vous achetez des slots d'extension, que vous en ayez besoin ou non. Une autre raison qui a fait que nous n'avons pas utilisé de circuits hautement intégrés était que des éléments séparés obtiennent beaucoup plus rapidement un visa d'exploitation. Par exemple pour les alimentations nous avons gagné environ 6 mois.



A PROPOS DE L'AMIGA

Shiraz Shifji : L'Amiga n'est pas un mauvais ordinateur. Ses possibilités graphiques sont fantastiques, je l'ai vu moi-même lors de la présentation à la presse. Mais la manière dont ce graphisme est produit ne plait pas à nos ingénieurs.

L'Amiga est une machine de jeux, en plus elle est de conception déjà un peu ancienne.

DW : Il paraît que dans les universités US on parle beaucoup d'un nouveau standard en matière de graphismes.

Shiraz Shifji : Oui, il est question de 1200 par 800 pixels. Le ST pourrait atteindre ce standard, il suffirait de remplacer certains circuits, alors qu'il faudrait reconcevoir toute une partie de l'Amiga. Le concept du ST est si flexible qu'il pourra pendant longtemps être adapté aux exigences du moment.
Par contre, l'Amiga n'est pas assez ouvert.

UN ST PORTABLE ?

Shiraz Shifji : Au départ nous avions deux projets parallèles, le premier nous l'avions baptisé RBP, ce qui veut dire Rock Bottom Price (prix au ras des pâquerettes), cela a donné le ST. L'autre projet s'appelait GHU pour God Help Us (Dieu soit avec nous) et là c'était un système intégré. Le nom laisse déjà deviner ce que nous en pensions à l'époque.

DW : Qu'en est-il d'un St portable ?

Shiraz Shifji : C'est au niveau de l'affichage que le bât blesse. Le LCD n'est pas la solution optimale. Il nous faudrait un affichage de qualité, consommant peu de courant.

DW : Est-ce qu'on travaille sur les écrans Plasma chez Atari ?

Shiraz shifji : En fait non. Vous savez, quand il existera quelque chose de bon et de pas cher, nous le prendrons. Mais pour l'instant il n'existe rien de valable dans ce domaine.

CHOIX DU SYSTEME D'EXPLOITATION

DW : Qu'en est-il d'UNIX ?

Shiraz Shifji : Nous avons fait des essais avec UNIX. Nous avons réalisé une extension qui fait du ST un terminal UNIX, cela fonctionne assez bien.

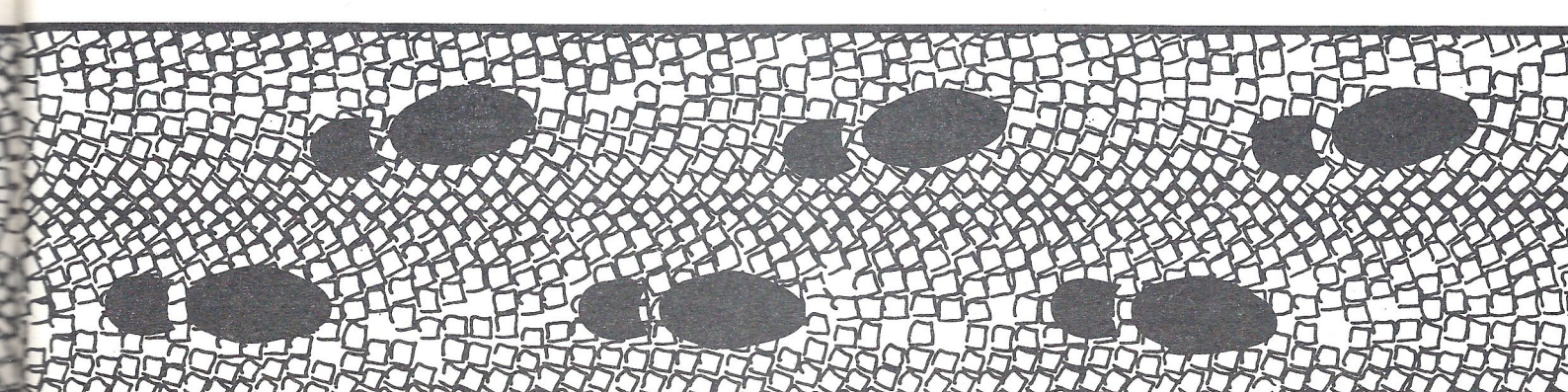
DW : Vous avez été si innovateurs en ce qui concerne le matériel et pourtant vous avez choisi GEMDOS comme système d'exploitation, un dérivé du CP/M 68K...

Shiraz Shifji : Non, non. Cela n'a rien à voir. GEMDOS est très différent. En janvier nous avons expérimenté CP/M 68K. Chez Digital Research il y avait un projet du nom de JasonDOS qui devint plus tard GEMDOS, et c'est celui que nous avons finalement choisi. GEMDOS travaille avec une structure de fichiers hiérarchique...

DW : Un soupçon d'UNIX, alors ?

Shiraz Shifji : Oui. Tout le contraire de CP/M 68K, très rigide puisqu'on ne peut travailler qu'avec des blocs de 128 octets. Ainsi, même si on pouvait physiquement travailler avec des blocs de 512 octets il faudrait à chaque fois en combiner 4 puis les dissocier à nouveau. CP/M 68K est excessivement lent. Sig Hartmann (directeur du département logiciel chez Atari et transfuge de chez Commodore) devait choisir entre CP/M et GEMDOS. Les techniciens optèrent pour GEMDOS. Certains jugeaient CP/M plus solide mais ce n'était qu'une supposition, il existait encore très peu d'ordinateurs sur lesquels était implanté le CP/M 68K.

En fait, au cours de son évolution GEMDOS s'est plutôt rapproché de MS-DOS. La lecture et l'écriture de fichiers s'effectue un peu à la manière d'UNIX.



ENQUETE

DW : A l'avenir vous pensez vous tourner vers UNIX ou alors allez vous conserver votre propre système ?

Shiraz Shifji : Nous travaillons sur le GEMDOS, réécrivons des parties entières et avons réussi à le rendre un peu plus compact et plus rapide.

DW : Pour le faire tenir plus facilement en ROM ?

Shiraz Shifji : Non, pas du tout. De toute façon la différence pour l'utilisateur est minime. De plus nous avons déjà GEMDOS en ROM, et il y tient très bien.

DW: Vous espérez donc que votre RBP va imposer son système d'exploitation comme standard ?

Shiraz Shifji : Nous avons l'ambition de faire du ST le standard dans la famille des 68000.

DES COMPATIBLES ST ?

Shiraz Shifji : Certaines sociétés en Europe et au Japon pourraient être intéressées à construire le ST sous licence. Elles pourraient soit fabriquer elles-mêmes les puces sous licence, soit les acheter chez nous et doter l'ordinateur de notre système d'exploitation.

DW : Par exemple si Thomson ou Olivetti désirait une telle licence, vous l'accorderiez ?

Shiraz Shifji : Bien entendu, cela ne pose aucun problème.

DW : Donc il se pourrait qu'un jour on ait un MSX-5 qui serait en réalité un ST ?

Shiraz Shifji : Oui, si les fabricants réagissent sufisamment rapidement. S'ils ne saisissent pas l'occasion d'ici un an nous dirons : "qu'ils aillent au diable" et nous fabriquerons seuls les ordinateurs. Ce n'est pas révéler un secret que de dire que nous n'arrivons pas à produire autant de machines que souhaitées. Par conséquent toute proposition pour construire des ST est la bienvenue. De toute façon c'est comme cela que se font les standards.

DW : Vous ne craignez donc pas les copies et compatibles ? Vous ne redoutez pas "l'effet PC", que les ordinateurs ne soient pas fabriqués sous licence mais "réinventés" ?

Shiraz Shifji : Aucun problème. Avec nos prix toute copie bon marché devient inutile. Nous ne comptons pas non plus sur les clones. Notre machine aura certainement une grande influence sur l'évolution des produits futurs. En Europe le phénomène se fait déjà sentir quand on voit des firmes comme Philips et Thomson s'associer pour certains projets. Les Japonais suivront également.

LES CHANCES COMMERCIALES DU ST

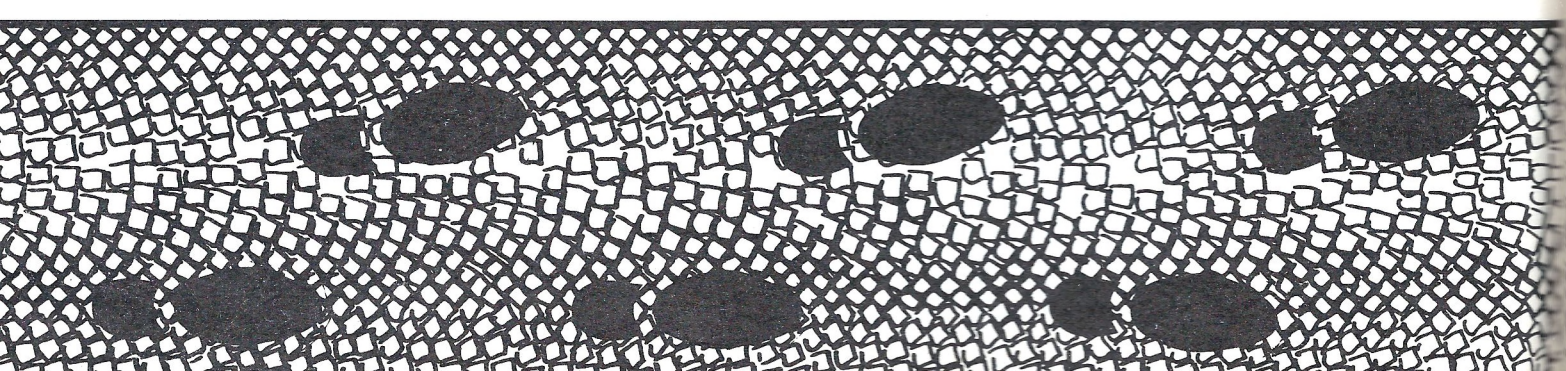
DW : A l'avenir, combien d'ordinateurs pourrez-vous produire ? Un million, deux millions par an ?

Shiraz Shifji : Nous sommes un peu revenu sur nos prévisions de départ en nous rendant compte de nos vraies possibilités. Mais je pense que nous arriverons quand même au chiffre de 1 million par an. En particulier quand on considère la révolution dans les prix et la nature même du ST. Depuis un moment les gens attendaient les ordinateurs à base de 68000.

Steward Olson a écrit dans la revue PC News que les utilisateurs étaient tellement sensibilisés par la campagne de publicité agressive menée parCOMMODOREpourl'Amiga,qu'ils se précipitent dans la prochaine boutique d'informatique. Mais en définitive ils achèteront le ST. En effet quand le client verra le ST à côté de l'Amiga il se demandera si pour 1000 dollars de moins il ne ferait pas mieux de prendre l'Atari.

A PROPOS DE JACK TRAMIEL

Shiraz Shifji : Vous savez, les temps ont changé et les personnes également. Lorsqu'en 1974 je fis la connaissance de Jack, c'était un type vraiment opiniâtre. Maintenant



qu'il a 100 millions sur son compte en banque il s'est beaucoup adouci. Je crois qu'avoir une opinion tranchée et un sens du bon produit, voilà ce qui est important. Jack n'est pas un homme très homogène, il n'a pas fréquenté une école de commerce et il ne suit pas un quelconque précepte quand il traite ses affaires. Il fait son travail comme il le ressent, avec beaucoup de passion et d'émotion. Si la terre n'était régie que par les théoriciens on s'ennuirait drôlement. A quoi serviraient les hommes ?

L'AVENIR DES ORDINATEURS

DW : Si vous regardez dix ans en avant, pour que vous n'ayez pas à parler de vos projets secrets, croyez-vous que l'ordinateur tel que nous le connaissons actuellement existera encore ?
A quoi ressemblera-t-il ?

Shiraz Shifji : Cela est quasiment impossible à prédire. Imaginez que vous ayez posé cette question il y a 10 ans. Même le VAX n'existait pas encore, à la pointe de la technologie on trouvait le PDP 11/70 avec 128K, un mini-ordinateur qui servait à l'industrie et à la recherche. Aujourd'hui on a des ordinateurs familiaux avec 256K.

Il y a quand même quelques éléments qui me paraissent prévisibles : les mémoires optiques vont faire leur apparition. On fera d'importants progrès dans le domaine de la technologie des mémoires. Les puces deviendront encore plus performantes. Lorsque l'arsenure de gallium détrônera le sicilium, on réalisera des entrées/sorties commandées par la parole. A l'heure actuelle ce n'est qu'un gadget, on ne dispose tout simplement pas de la vitesse d'exécution nécessaire. La reconnaissance de la parole ne deviendra intéressante que lorsqu'on obtiendra un taux d'erreur inférieur à un pour cent.

Par contre il va se passer des choses dans le domaine du traitement de l'image...

DW : Une collaboration entre la vidéo et l'ordinateur ?

Shiraz Shifji : Exact. Et pas seulement dans le domaine de l'informatique familiale. Le traitement de l'image apportera d'énormes progrès à la production. De nos jours les robots manquent un peu de besogne.

Le langage naturel et les systèmes experts sont deux sujets très en vogue en ce moment. Ce que l'on appelle aujourd'hui système expert n'est qu'un avant-goût de ce qui va bientôt arriver, dans les cinq prochaines années.

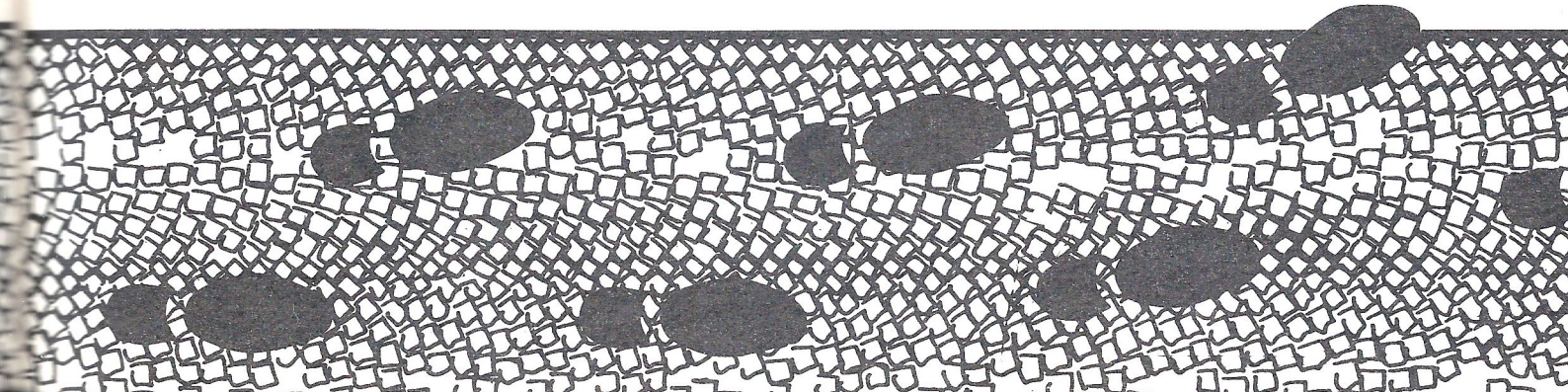
DW : Y aura-t'il demain une raison valable d'avoir un ordinateur à la maison ?

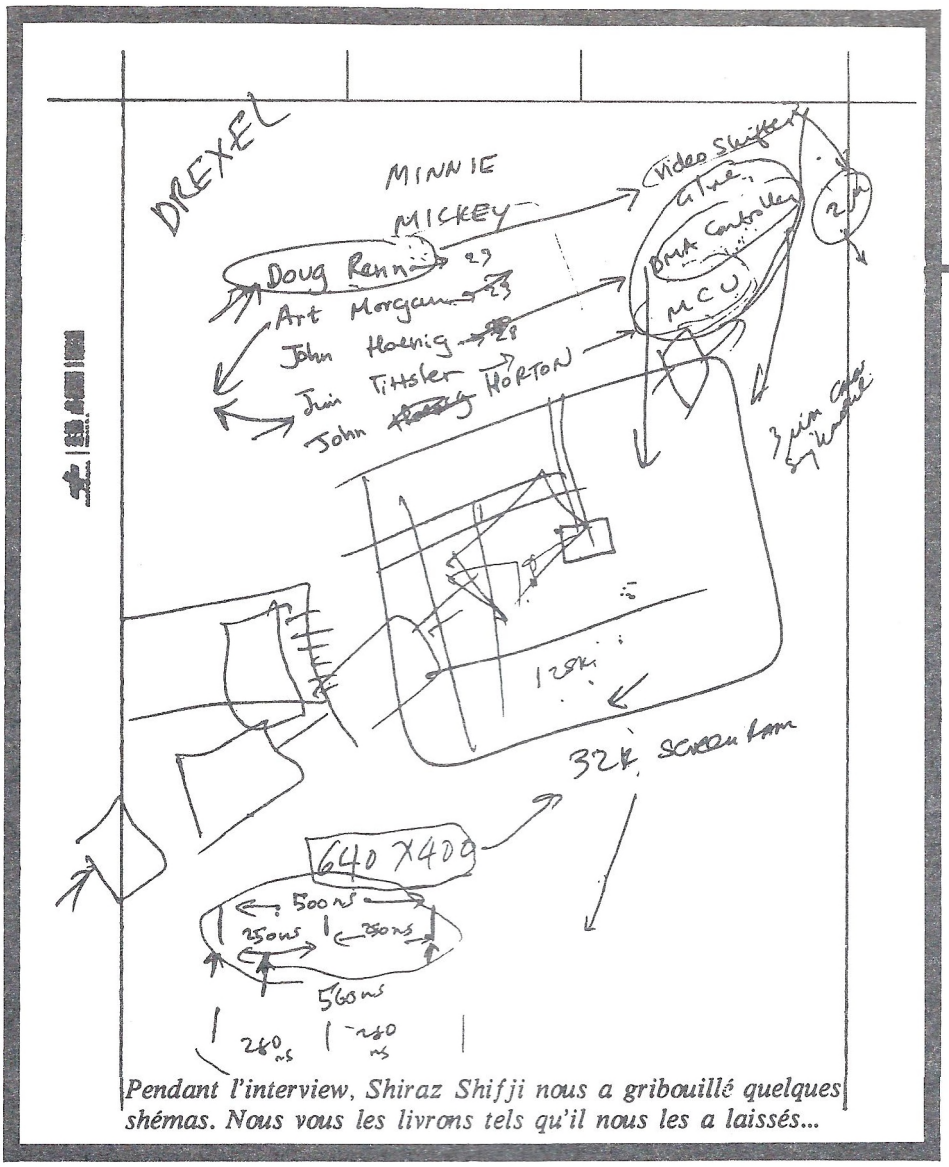
Shiraz Shifji : On le saura dans les deux prochaines années. La capacité mémoire est un facteur très important: l'avènement des disquettes a constitué un très grand bond en avant, on peut réaliser un grand nombre d'applications grâce à elles, mais ce n'est pas assez. Il nous faut des mémoires à lecture/écriture optique.

Je veux pouvoir utiliser mon ordinateur aussi simplement qu'un livre; c'est à dire n'avoir qu'à l'ouvrir et à lire. Par exemple pour ma déclaration d'impôts j'avais besoin de connaître la distance exacte qui sépare Philadelphie de San Francisco, mais je n'ai pas réussi à mettre la main sur le livre dans lequel j'aurai pu trouver l'information que je cherchais. J'aurai bien sûr également pu appeler The Source (une grande banque de données publique aux états-unis) avec un modem mais est-ce que j'aurai eu la réponse souhaitée...
Ce genre de questions apparaît toujours soudainement, sans prévenir, c'est pourquoi il me faut un ordinateur auquel il me suffit de poser la question et qui me délivre instantanément la réponse. Cela est tout à fait réalisable.

DW : L'ordinateur comme machine à connaissances ?

Shiraz Shifji : Oui, du moins en ce qui me concerne. Un musicien aura des exigences toutes autres. Ce dont





Pendant l'interview, Shiraz Shifji nous a gribouillé quelques shémas. Nous vous les livrons tels qu'il nous les a laissés...

j'ai besoin c'est d'une machine à connaissances autonome, c'est-à-dire qui n'ait pas besoin d'être connectée par modem interposé à un gros ordinateur central. De telles unités centrales pourraient devenir superflues puisqu'on aura tout chez soi.

DW : Vous utilisez un ordinateur chez vous ?

Shiraz Shifji : Au début j'avais un prototype avec lequel j'ai beaucoup travaillé. Depuis peu il y a un ST sur mon bureau. Le programme Paint a tout de suite fasciné mon fils de trois ans et il a immédiatement su jouer avec.
L'informatique doit devenir aussi simple que cela, c'est la seule solution pour que l'ordinateur devienne autre chose qu'un hobby ou qu'un luxe, pour qu'il se banalise comme le réfrigérateur ou l'aspirateur.

DW : L'ordinateur n'est donc pas le fruit de la mode comme le hula-hup, mais plutôt quelque chose comme la voiture. Si l'on va plus loin dans la comparaison ordinateur/automobile , à quel stade du développement de la voiture croyez-vous que nous nous situons actuellement ?
La Ford T ?

Shiraz Shifji : Non, nous n'en sommes pas encore là. Par exemple le CBM 64 en était encore très loin. Le ST pourrait devenir quelque chose comme une Ford T...

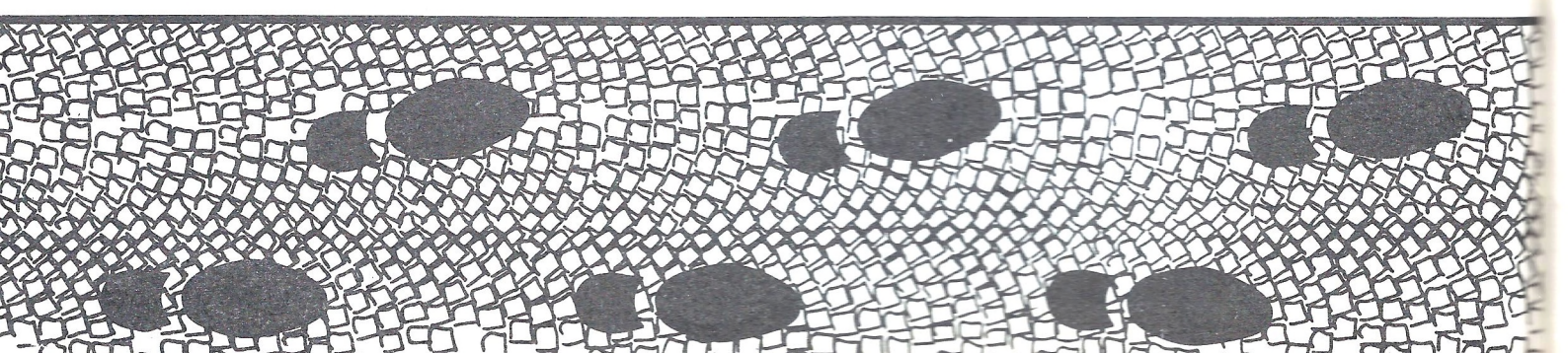
DW : On pourrait l'appeler Silicon T...

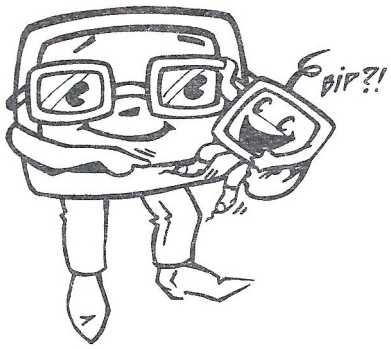
Shiraz Shifji : Exactement. Tout comme la Ford T on peut adapter le ST à l'évolution de la technique à l'aide de divers accessoires, il a la flexibilité requise.

DW : Merci beaucoup pour cette discussion très intéressante, M. Shifji.

(a) Chuck Peddle a longtemps travaillé chez Commodore et il a été le père des PET, CBM3000, CBM4000 et enfin 8000. Il a quitté Commodore pour fonder SIRIUS devenu ensuite VICTOR.

(b) Il s'agit de l'ancien Directeur de Commodore Allemagne devenu directeur d'ATARI Allemagne.





Nouveautés

MODULA-2 ST

MODULA-2/ST OU COMMENT DEVELOPPER SUR ATARI ST

L'ATARI 520/ST est sans contexte le micro qui fait le plus parler de lui en ce moment. Pourtant, les développeurs, enthousiamés à juste titre par les caractéristiques techniques, restent un peu sur leur faim. Mais tout doucement les logiciels arrivent, et parmi eux l'extraordinaire MODULA-2/ST.
Ce logiciel est un langage de programmation avancé de haut niveau, conçu par le professeur Niklaus WIRTH, déjà auteur du langage PASCAL. Il est assez simple pour permettre un démarrage facile aux novices de la programmation, mais suffisamment puissant pour que les programmeurs confirmés puissent écrire des logiciels complexes.

Un de ses points forts est sans aucun doute la structure très modulaire des programmes développés avec ce langage. Chaque programme doit être subdivisé en tâches plus petites, constituant chacune des modules. De plus chaque module se décompose en deux parties : Le module de définition et le module de mise en oeuvre. Cette technique de programmation permet d'écrire, de modifier, et de mettre au point des sous-programmes sans modifier le reste de l'application. En outre cela permet de développer de grosses applications à plusieurs programmeurs.

La synthèse de tous les modules constitue la dernière étape de la création. Enfin, l'utilisateur peut créer une bibliothèque de sous-programmes qu'il pourra utiliser au sein de nouvelles applications.

MODULA-2/ST comprend la compilation séparée, les types opaques, les co-routines et les routines en virgule flottante. De plus MODULA-2/ST est intégré à l'environnement graphique GEM de l'ATARI-2/ST. Toutes sortes d'interactions entre l'éditeur, le compilateur et l'éditeur de liens sont possibles grâce aux menus GEM et à la souris.

Les capacités graphiques de l'ATARI faciliteront vos premiers contacts avec MODULA-2/ST et si vous avez envie de développer des applications personnelles sur votre ATARI c'est, à notre humble avis, MODULA-2/ST qu'il vous faut !

MODULA-2/ST

MODULA-2/ST est un système de développement complet pour l'ATARI ST. Il est composé :

- D'un programme Editeur
- D'un programme MODULA-2
- D'un éditeur de liens (linker)
- De librairies pré-définies

MODULA-2/ST vous permet de créer des programmes utilisant :

- Plusieurs polices de caractères
- Plusieurs fenêtres
- Des menus déroulants
- Toutes les possibilités graphiques de l'ATARI ST
- Les co-routines

Caractéristiques techniques de MODULA-2/ST :

- Compilation séparée avec contrôle des types
- Support complet des routines GEM
- Execution directe des programmes à partir du bureau (desk)
- Création de codes compactés
- Documentation en français
- Programmes de démonstration (graphiques)

REF ST001
PRIX 1450 FF

VIRGULE



DEUX TRAITEMENTS DE TEXTE EN UN POUR LE COMMODORE 128

Dans le dernier numéro de MICRO INFO nous vous avions présenté VIRGULE, traitement de texte professionnel pour le C 64.

Si je vous en parle encore une fois c'est parce que VIRGULE est enfin disponible pour le COMMODORE 128, avec toutes les possibilités supplémentaires qu'offre cette machine (80 colonnes, dictionnaire 225.000 mots, 51760 caractères en mémoire)...

On peut tout faire avec VIRGULE, du publi-postage, des lettres types, votre thèse, ou votre nouveau roman que tout éditeur sensé attend avec impatience.
Vous n'avez pas l'âme d'un poète ?
Ne partez pas encore, VIRGULE peut aussi archiver tous vos documents, courriers, articles, etc.
C'est un traitement de texte professionnel, et en français qui plus est, avec tous les accents bien sûr, et même la possibilité, si ceux-ci ne vous suffisent pas de redéfinir certains caractères. Vous avez à votre disposition 43 fonctions d'édition pour créer et modifier vos textes : Recherche d'un mot par exemple, mais aussi remplacement d'une occurence par une autre.
Pour l'impression ?
Hé bien, vous avez à votre disposition 17 paramètres pour définir le format de votre document : nombre de lignes par page, nombre de lignes au pouce, interligne, saut de ligne, marges gauche et droite, justification, centrage, etc. Vous pouvez aussi sortir des fichiers chaînés, imprimer à partir d'une page précise, et ce, en autant d'exemplaires que vous le souhaitez.

Au fait, bien que vous ne faites jamais de fautes *d'ortaugrafe*, il vous ait déjà arrivé, je pense, de faire des fautes de frappe. Avec le dictionnaire intégré de VIRGULE elles ne seront plus qu'un mauvais souvenir, Il vous corrige vos fautes, et, luxe suprême, vous pouvez ignorer ses corrections...

CARACTERISTIQUES TECHNIQUES

Saisie :	80 colonnes
Format écran :	80,120 colonnes
Longueur total du texte en mémoire :	51760 caractères
Dictionnaire :	225.000 mots

Possibilité de rajouter des mots dans le dictionnaire jusqu'à concurence de 300.000 mots.

FONCTIONS D'EDITION

- Possibilité d'utiliser un texte principal et un texte annexe.
- Recherche avec ou sans remplacement d'une occurence dans un texte ou dans plusieurs textes chaînés.
- Transfert de lignes.
- Duplication de lignes.
- Utilisation de zones variables pour le publi-postage et les lettres types.
- Utilisation d'une banque de données.
- Tabulations simples ou numériques (avec possibilités de calculs).
- Définition d'attributs d'impression : soulignement, surimpression, indices, exposants ...
- Compteur de mots.

FONCTIONS D'IMPRESSION

- Définition d'un haut de page.
- Définition d'un bas de page.
- Numérotation automatique des pages.
- Définition du format d'une page d'impression : marge gauche, marge droite, nombre de lignes.
- Alignement à droite.
- Justification.
- Centrage du texte.
- Définition du nombre de lignes au pouce.
- Définition de caractères spéciaux. (symboles mathématiques par exemple).

REF MA349 (C128)
PRIX : 750 FF

L'ASSEMBLEUR SUR MO5, COUP D'OEIL SUR ML1

Cela faisait un moment que je me battait avec mon programme, je n'arrivait pas à rendre une séquence d'animation assez rapide. J'avais beau supprimer tous les espaces, retirer les REM, etc, mon bonhomme avancait toujours à une allure d'escargot. Il fallait passer à l'assembleur, seulement voilà, mon programme était en BASIC, et à par le bonhomme tout le reste fonctionnait parfaitement. Je n'avais pas envie de tout recommencer, et puis, il faut bien le dire, l'assembleur, je n'y connais pas grand chose.

C'est Costa qui ma sauvé, Il m'a ammené ML1, un macro-language assembleur pour mon MO5. Et alors me direz vous, cela reste un assembleur, avec toutes les difficultés que cela entraîne. Mais voila, ML1 à le gout d'un assembleur, ressemble à un assembleur, mais ce n'est pas seulement un assembleur. Tout d'abord il permet de mixer le BASIC et la source assembleur, et même de passer des paramètres à partir du BASIC. Grâce à plus de 100 macro instructions prédéfinies il devient facile de mettre en oeuvre des routines en langage machine complexes. En plus grâce à l'éditeur résident (amélioré avec l'instruction AUTO et la visualisation des lignes d'erreurs) il est tout aussi confortable de programmer en assembleur qu'en BASIC. Les nouvelles commandes que ML1 ajoute à l'éditeur sont :

@SYM : affichage de la table des symboles
@AUT : édition automatique du numéro de ligne suivant
@DES : désassembleur 6809
@DUMP : moniteur hexadécimal de type plein écran

La commande la plus importante est @ASS qui provoque l'assemblage. La première passe (PASS 1) traduit les labels et les directives ORG. A la fin de la seconde passe (PASS 2) on dispose du programme objet qu'on peut lancer tout simplement par EXEC, adresse de départ.

ML1, attentif à tout, vous indique en clair si vous avez fait une erreur, et vous indique le numéro de ligne où elle se trouve.
Toutes les sauvegardes, sorties sur imprimante, etc se font à partir des instructions classiques du BASIC : SAVE, LOAD, LIST, etc.
L'écriture du programme source est très simple malgré certaines petites originalités, par rapport à d'autres assembleurs 6809 :
- la directive "flêche vers le haut" pour forcer l'adressage indirect étendu
- l'utilisation des parenthèses pour l'adressage indirect indexé
- la directive "=" à la place de EQU

ML1 est vraiment très facile à comprendre et à utiliser, j'ai même trouvé quelque chose que je vous livre sous le sceau du secret :
Contrairement à ce qui est dit dans le manuel il est tout à fait possible de modifier la répartition mémoire source/code objet et cela grâce à l'instruction BASIC :
CLEAR, adresse.

En bref, ML1 est un produit superbe, mon bonhomme traverse l'écran à toute vitesse, je suis heureux...

CARACRERISIQUES TECHNIQUES

- Assembleur 6809
- Désassembleur 6809
- Fonctions DUMP AUTO ...
- Interaction BASIC/ langage machine

FONCTION D'EDITION

- Fonction AUTO...
- Visualisation des lignes d'erreurs
- Moniteur hexadécimal plein écran
- Affichage de la table des symboles
- Edition automatique du numéro de ligne suivant

INSTRUCTIONS

- 102 macro-instructions prédéfinies très puissantes
- Macro-instruction définissable
- Directives telles que : ORG, FCB, FDB, RMB, MACRO, ENDM, EQUATE, etc

REF : MO101
PRIX : 245 FF

les plus de Micro Application pour Thomson MO5

Allez plus loin avec votre THOMSON.



TRUCS ET ASTUCES du THOMSON MO5/TO7 70
Enfin un livre **MICRO APPLICATION** pour les **THOMSON !**
Avec trucs et astuces pour le MO5/TO7-70 nous voulons apporter le maximum à tous les possesseurs de ces ordinateurs. Tout au long des 400 pages de ce très bel ouvrage ils trouveront non seulement de nombreux conseils et explications pour profiter au plus de leur machine mais aussi des programmes utilitaires très puissants comme un traitement de texte, un générateur de sprites (lutins) qui font cruellement défaut à la machine française, un moniteur disque, une gestion de fichier complète, un programme graphique... Des astuces pour utiliser le DOS et pour programmer en langage machine, créer des fenêtres...
Réf. : ML 160 Prix : **149** F TTC

FANTASTIQUE ! Mélangez BASIC et ASSEMBLEUR.

ML 1 MACRO-LANGAGE ASSEMBLEUR POUR THOMSON MO5

***ML 1** est un macro-assembleur symbolique qui contrairement aux assembleurs conventionnels permet de mixer les langages **BASIC** et Assembleur à volonté. Vous pouvez donc utiliser les qualités propres à ces deux langages universels (souplesse et facilité de programmation pour le **BASIC**, puissance et rapidité pour l'assembleur) pour développer très efficacement des programmes sophistiqués (jeux, utilitaires, applications pro...).*

***ML 1** comprend :*
– un éditeur plein écran ;
– 102 macro-instructions prédéfinies qui rendent le langage machine accessible à tous ;
*– un Moniteur et un Désassembleur pour travailler directement sur la mémoire du **MO5** ;*
– définition des macro-instructions ;
*– chaînage et exécution Assembleur **BASIC** ;*
*– passages de paramètres à partir du **BASIC**...*

***ML 1** comprend une documentation complète.*
***ML 1** est l'outil de développement universel pour **THOMSON MO5** car il réunit les qualités des langages **BASIC** et **ASSEMBLEUR**.*
*L'apprentissage de **ML 1** est extrêmement simple et rapide car vous continuez à utiliser vos connaissances "BASIC".*
*Prix TTC livré sur K7 : **245 FF.***

Vous êtes auteurs, venez rejoindre notre équipe, votre nom pourrait déjà figurer sur ce logiciel ou un de nos livres.

Pour fêter son 1er utilitaire sur THOMSON, MICRO APPLICATION vous offre une cassette de jeu gratuite (100 % langage machine). Offre valable dans la limite des stocks disponibles.

MICRO APPLICATION
13, rue Sainte Cécile 75009 PARIS
tél. (1) 47-70-32-44

BON DE COMMANDE

DESIGNATION	QUANTITE	PRIX
	TOTAL TTC	

☐ Mandat ☐ Cheque ☐ CCP
Libellez vos cheques à l'ordre de Micro-Application
Nom, Prenom _____
Adresse _____
Ville _____ C.P. _____
CB ☐☐☐☐ ☐☐☐ ☐☐☐ ☐☐☐
+ 20 F de frais d'envoi ou 40 F pour envoi recommande
Date et signature

TO 1 86

DU NOUVEAU...

ENFIN UTILISABLE... LE BASIC SUR ST

L'effet boule de neige se met en place : Atari FRANCE s'est maintenant restructuré et va livrer des quantités plus importantes de ST, ce qui provoque des échos favorables de toutes parts. Il n'y a qu'une chose qui prête à l'amertume : sur le disque -fièrement appelé "Langage Disc"- il n'y a que du LOGO et ... beaucoup d'espace. De l'espace justement pour un programme de plus de 150 kbyte : le BASIC pour Atari ST. Peu avant le bouclage de la rédaction de ce numéro, nous en avons reçu une version préalable qui sera probablement rendue publique après avoir été examinée.

Tout de suite après le chargement des 150 kbyte de codes de programme désigné sur la disquette par BAS.PRG, l'observateur a l'impression de se trouver en face de quelque chose de bien spécial.
Tout d'abord, une fenêtre s'ouvre au milieu de l'écran pour être aussitôt remplacée par trois autres. Il n'en reste qu'une petite fente qui permet à l'utilisateur, selon la façon GEM, d'activer cette fenêtre, de l'"étaler" comme une feuille de papier en haut de l'écran.

Elles s'appellent EDIT, LIST, OUTPUT et COMMAND, ce qui indique déjà clairement la façon dont elles sont utilisées. En plus, elles sont numérotées de 0 à 3 de façon à communiquer avec le Soft correspondant. Sinon, comment effacerait-on le contenu des fenêtres ? Ainsi, on s'est efforcé de faire comprendre à Atari les commandes CLEARW 0 à CLEARW 3 ainsi que FULLW 0 à FULLW 3 pour que les fenêtres souhaitées puissent être agrandies au maximum ou bien être effacées.
Lorsque l'initialisation de l'écran est fini, BASIC apparaît avec un petit "ok" modeste. Il n'y a pas de message qui indiquerait combien de bits restent disponibles, puisqu'il ne servirait qu'au programmeur en langage machine... A côté de la fenêtre, à la toute première ligne de l'écran, le menu tel que nous le connaissons sur GEM, attire notre attention. Sous DESK se trouve l'indication : ABOUT ATARI BASIC et bien évidemment la notice indispensable du copyright. Sinon il n'a pas du tout changé et reste avec l'émulateur VT52, etc... Dans tous les cas il ne faut surtout pas sous estimer le fait que par exemple pour mettre les paramètres des lignes de programme en série on n'a pas besoin de retourner au système d'exploitation. FILE aussi est à la hauteur de ce que l'on attend : LOAD, SAVE as..., Delete File et Merge ouvrent au milieu de l'écran une petite fenêtre accessoire dans laquelle s'installe un extrait du directory. A l'aide de la souris ou des fonctions fenêtres on peut sélectionner une donnée qui sera traitée par la fonction souhaitée. QUITE enfin, est aussi une commande de ce menu intéractif, ce qui ferme toutes les fenêtres et quitte BASIC.

Sous RUN, les choses deviennent déjà plus intéressantes puisqu'on y trouve des commandes système telles que RUN BREAK STOP et CONTINUE. Qui a déjà programmé en BASIC devrait bien les connaître. Mais qu'en est-il de STEP et de BufGraphics ? Eh bien le Buffer-Graphics est connu au moins de ceux qui possèdent un Atari ST et qui en ont étudié les instructions Logo. Ainsi, une fenêtre ouverte couvre des parties d'une autre et les "drop down" menus eux, recouvre l'image représentée. Pour rétablir l'état initial de l'écran le système d'exploitation doit sauvegarder provisoirement les parties de l'image recouvertes - quand la mémoire du ST est presque remplie, on utilisera la disquette - ce qui implique une opération assez longue que l'on évitera, dans la mesure du possible. Et justement Buf-Graphics vous évite de redessiner (au cas où il n'y a vraiment pas de place dans la mémoire). Au niveau du menu EDIT le ST-BASIC d'ATARI montre à ceux qui croient qu'un éditeur plein écran, avec des touches de déplacement est bien confortable, qu'il existe des façons encore plus confortables et -après s'y être habitué- plus rapides. On y trouve des opérations telles que : effacer une ligne, ajouter une ligne, scroller dans le texte une page plus loin ou une page en arrière, sauvegarder un programme sous son ancien nom, le charger etc... Et ceci non seulement à l'aide de la souris mais également à l'aide de touches de fonctions (le ST en possède dix). D'ailleurs une fois que l'on s'est familiarisé avec le ST,

on arrive à écrire et éditer ses programmes dans un temps record !
De toute façon, rien que de programmer avec ce BASIC est un régal puisqu'ATARI a continué, depuis les 600 et 800 XL et depuis la série 130 XE, à contrôler chaque ligne de programme de façon à ce qu'il soit correct au niveau de la syntaxe.

Un aimable "SOMETHING IS WRONG" (au lieu d'un "SYNTAX ERROR" ou pire comme: "ERROR 13") refuse la ligne contenant l'erreur.

Mais les "vrais" problèmes commencent au premier essai du programme : Pourquoi les résultats ne sont pas affichés correctement sur l'écran ? Pourquoi ne coïncident-ils pas avec les calculs faits "à la main" ? La boucle est-elle parcourue une fois de trop ou l'inverse ? Pourquoi l'ordinateur se trouve-t-il dans telle partie du programme et non pas dans une autre ?

Ceux qui disposent d'un système assez sophistiqué apprécient souvent TRON (on ne parle pas du film des Disney's studios.., mais plutôt de l'abréviation de "TRACE ON"). Ce petit mot invite l'utilisateur à attribuer une adresse avant l'écriture de chaque ligne de programme, ce qui facilite beaucoup les choses quand on a recours à un listing. Seules les lignes très longues contenant beaucoup de mots-clé restent problématiques, et comme par hasard sont les plus nombreuses. Heureusement, les ingénieurs de chez DIGITAL RESEARCH (responsables du BASIC ATARI) ont aussi apprécié TRON. Sinon pourquoi auraient-ils attribué à leur BASIC les trois caractéristiques qui se cachent derrière les mots TRACE, FOLLOW et STEP ?

Après avoir appelé TRACE à l'aide de la souris ou du clavier l'évaluateur se montre très "éloquent" : chaque étape est annoncée et on peut suivre de près l'évolution du programme (quel résultat mène à quelle partie du programme etc.). S'il vous arrive tout de même que les choses se passent trop vite pour que vous puissiez suivre, l'option STEP vous avertit que désormais, pour l'exécution de toute instruction, il faut appuyer sur "RETURN".

Ne pensez désormais plus qu'à une chose : quelque part, il y a une routine (?) IMPRESSION (?) ; donc, au besoin, taper : GOSUB IMPRESSION.

Il faut maintenant mentionner les capacités graphiques extraordinaires du ST. N'y avait-il pas un ordinateur capable de tout au niveau hardware et dont le software causait de grands problèmes même aux professionnels ? Ceci ne peut jamais arriver à l'utilisateur d'un ST.
Il ne lui faut qu'une seule commande pour tracer une ligne :

LINE 1,1,100,100.

Les cercles ne sont pas plus difficiles : CERCLE 100, 100, 30. Et s'il s'agit de remplir une figure : FILL 100, 100, 1. Dans ce cas, cela aurait été le cercle si, dès le départ, on avait prévu un disque, ou choisi : PCIRCLE 100, 100, 30.
Toutes les commandes graphiques sont exécutées à une remarquable vitesse. Si on se demande: pourquoi si vite ? On retrouve un vieil ami : GEM.

GEM a largement fait parler de lui (aussi chez nous) et quelqu'un qui a suivi nos articles ou qui a eu affaire à GEM dans la pratique va peut-être se demander si un tel mariage entre BASIC et GEM portera d'autres fruits. La réponse : mais bien sûr ! ATARI et DR ont également pensé aux "fanas" qui veulent profiter au maximum de leur ordinateur. Ainsi, le professionnel et le spécialiste GEM connaissent bien le contrôle de dimensions des tableaux qui sert de point commun à de nombreuses routines GEM et qui leur fournit les paramètres nécessaires. L'aspect de la chose est le suivant : ces routines sont également utilisables en BASIC. BASIC connaît les adresses du contrôle de dimensions des tableaux. Dès le début quelques variables du système sont tout gentiment et correctement initialisées (comme l'illustrent, par exemple : PRINTDISYS, GEMDISYS).

Après que le BASIC Business d'ATARI ST connaisse aussi bien PEEK que POKE et URS, il n'y a plus d'obstacles à l'utilisation de cette routine.

Le dernier sujet que nous tenons à aborder est les "stations" de disquette.
On ne va pas s'étendre sur le traitement habituel avec les fonctions comme: PRINT£,GET£,IMPUT£,EOF etc. (en partant du principe que les designers n'ont oublié aucune des fonctions que vous connaissez déjà bien). Au contraire, on va aborder brièvement les choses un peu particulières et même tout à fait nouvelles. Ainsi il n'y a plus de raison de quitter le BASIC pour exécuter les commandes système.

Vous pouvez examiner le catalogue (directory : DIR), effacer des fichiers (KILL) ou bien changer leur nom (REPLACE). Les opérations de sauvegarde et de chargement des fichiers fonctionnent même dans les sous-directories. Il faut toujours examinerencommençantunprogramme en BASIC, l'espace encore disponible sur disquette. Car BASIC ouvre pour les opérations d'éditions ou de copie un fichier back-up (BASIC WRK).

Ce BACK-UP est conservé même après des opérations comme : changer le nom d'un programme, effacer ou transférer des lignes du programme il s'agit d'une caractèristique qui contribue considérablement à la sauvegarde des données, puisque les manipulations erronées peuvent être annulées.
En fin de compte, on ne peut que considérer le BUSINESS BASIC d'ATARI comme un langage sensationnel pour un ordinateur. Et si le fier propriétaire d'un ST jette un coup d'oeil sur des ordinateurs comparables, alors il ne peut que se dire : l'attente a valu la peine !

Si l'erreur est repérée et si on soupçonne la variable XYZ d'en être responsable, on laisse à l'évaluateur le soin de suivre l'évolution de cette variable : FOLLOW XYZ. Résultat de l'opération : dans la fenêtre de commande, on reçoit des messages comme :

XYZ ! is 500 in line 30
XYZ ! is 501 in line 50

En considérant qu'en même temps le listing reste visible (dans la fenêtre "LIST"), il ne nous reste plus grand chose à dire, sinon qu'avec une telle assistance on aurait presque souhaité écrire des programmes avec des erreurs. Mais pour revenir à la variable, depuis toujours , on conseille aux gens d'utiliser des noms de variables qui ont un sens au lieu d'abréviations comme X,Y,Z ou X1,X2,X3. Maintenant à vous de faire votre choix sur ATARI. Ce ne sont pas seulement les 2,3 ou même 8 premiers mots qui sont réservés mais les 31 premiers ! En plus ils se distinguent de par leur type :
entier, réel, chaîne de caractères ;
ou bien des indications comme DEFNG et DEFBL permettent de déclencher plusieurs variables à la fois.

D'autres données techniques : dix niveaux de parenthèses sont permis et jusqu'à 16 sous-programmes (GOSUB) peuvent être imbriqués les uns dans les autres. Et pour ceux à qui tout cela ne suffit pas encore, voici d'autres friandises :
aujourd'hui, la boucle WHILE/END fait partie de tout "bon BASIC". Les labels par contre ne sont pas si répandus, ce qui ne tardera certainement pas à prêter à des critiques de la part des défenseurs de la programmation structurée. Le BASIC ST vous permet de donner un nom à vos sous-programmes. L'avantage est que vous n'avez plus besoin de noter les numéros correspondants aux lignes qui, à cause d'un mauvais "planning" ou d'un appel RENUM (c'est aussi dans les capacités de l'ATARI !), peuvent être effacés.

PETITES ANNONCES

AMSTRAD CPC 464
+ Lecteur disquette

Cherche correspondants pour rompre solitude par échanges divers (idées, trucs etc...)

Monsieur AUDIFFREN Daniel
78 La Taillette
95180 MENUCOURT

Vend : MONITEUR couleur
FIDELITY CTM 14 s/s
garantie 2400 FF
MONITEUR monochrome AMBRE
ZENITH ZVM 122 : 700 FF
Nombreux programmes originaux pour CBM-64

Monsieur HARBONNIER Raoul
BP 64 92254 LA GARENNE
COLOMBES CEDEX

Tél. : 242-69-10 (après 18 h).

Vends : COMMODORE 64,
Floppy 1541, MPS 801, Moniteur 1701, etc...
Prix à débattre. Ecrire à :

Monsieur JUILLERAT
La Marchande 6
2534 ORUIN

Vends : AMSTRAD CPC 464 Couleur
+ Imprimante DMP-1 + Joystick JY-2
+ 30 Jeux
Le tout : 6000 FF

Monsieur MENELLA Stéphane
6 Square J. Racine
77440 LIEY S/ OURCQ

LECTEUR 5 POUCES SUR ST

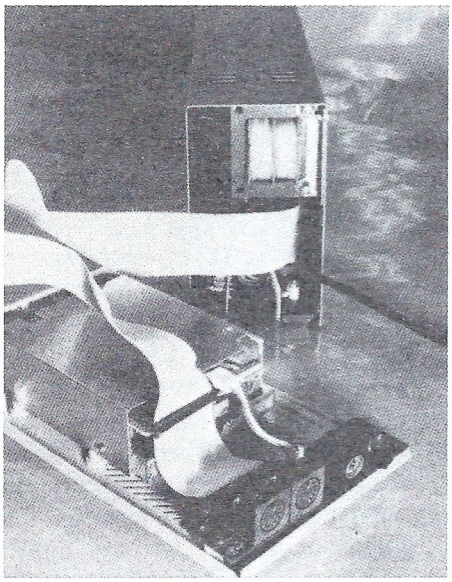
DOSSIER

COMMENT BRANCHER UN LECTEUR 5 POUCES SUR l'ATARI ST.

Un beau jour de l'été dernier j'entrais enfin en possession d'un Atari ST. Inutile de dire que pour moi il s'agissait surtout d'un outil de développement. Il est tout aussi inutile de préciser que j'ai dû faire face aux mêmes difficultés que les autres ST-istes de première heure.

Mais outre les "plantages" de la machine, toujours accompagnés de divers graphismes aléatoires et des "champignons" désormais bien connus, il y avait autre chose qui me tracassait au moins autant. En effet, dès le début de mon "ère Atari" je m'étais précipité sur le compilateur C et avais constaté avec stupéfaction qu'un simple programme affichant à l'écran le message "coucou, c'est moi" occupait déjà 12 Ko de mémoire.

En soi cela n'a rien de catastrophique, mais en regard des 20 Ko encore disponibles sur ma disquette "linker" je me rendis rapidement compte qu'il fallait faire quelque chose. Mon deuxième programme écrit en langage C ne pouvait déjà plus être compilé. Qu'en serait-il lorsque j'aurai à traiter des programmes source de 10 à 20 Ko ?



Il me fallait un deuxième lecteur de disquettes, si possible avec une capacité accrue, donc un double-face. Tant qu'à faire ce serait un 5 pouces 1/4 puisque ces disquettes coûtent moins de 8 francs contre 30 francs pour une 3 pouces et demi. La solution la plus économique serait assurément de pouvoir connecter un de mes "vieux" drives ayant déjà fait ses preuves. Mais comment trouver ce connecteur bizarre dont je ne connais même pas la désignation exacte ? Divers coups de fil m'apprirent que ce genre de connecteur était introuvable ; de plus la production de lecteurs de disquettes Atari doubles-faces était également repoussée à une date ultérieure à cette époque. Même le SF 354 était difficile à trouver à l'époque. Devais-je continuer à travailler comme précedemment ? Impossible.

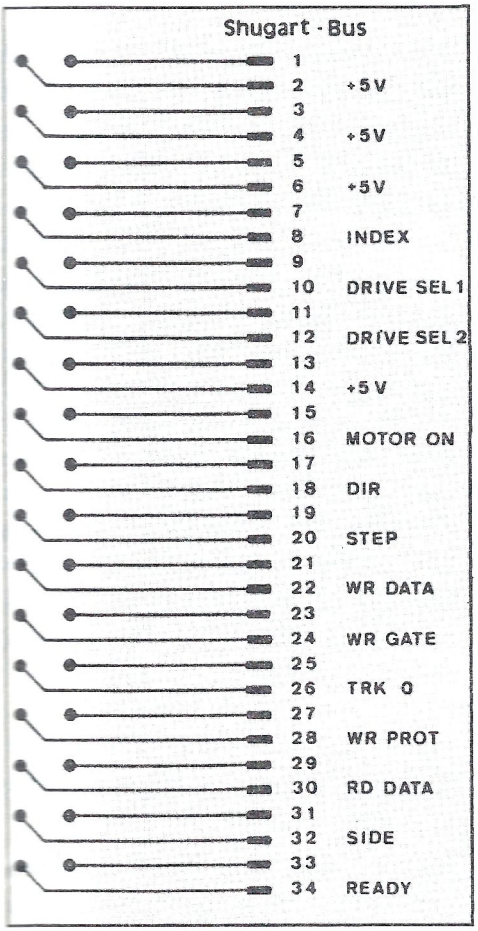
Il est bien connu que rien ne peut résister à l'esprit inventif d'un "hacker", pas même la perspective de perdre le bénéfice de la garantie en ouvrant le boîtier. En outre, il y avait une chose qui m'intriguait beaucoup : pourquoi donc le lecteur Atari est-il aussi volumineux (pour un lecteur 3,5") ?

Les quatre vis furent vite ôtées, le couvercle démonté encore plus rapidement, etc... L'effet de surprise fut parfait. En effet le câble Atari était juste enfiché dans une sorte d'adapteur dont l'autre extrêmité constituait un bus Shugart bien connu. Celui-ci se présenta à moi sous la forme d'un câble d'environ 10 cm de longueur, tout comme sur mon TRS 80. Sauf si les constructeurs de l'Atari avaient inversé quelques lignes, le problème était résolu.

La première chose fut donc de vérifier si toutes les connexions correspondaient bien ; heureusement Atari avait fourni la documentation donnant le brochage du connecteur du ST :

1 READ DATA	8 MOTOR ON
2 SIDE 0 SELECT	9 DIRECTION IN
3 GND	10 STEP
4 INDEX	11 WRITE DATA
5 DRIVE 0 SELECT	12 WRITE GATE
6 DRIVE 1 SELECT	13 TRACK 00
7 GND	14 WRITE PROTECT

Le brochage du connecteur 34 pattes du bus Shugart ne constitue un secret pour personne :

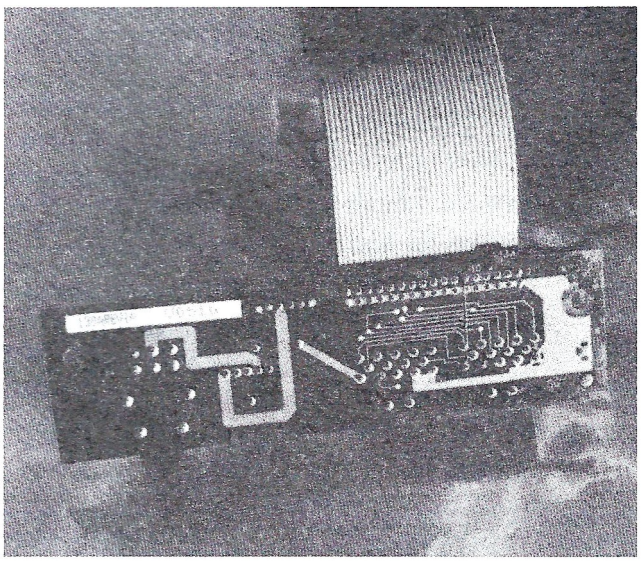


Une étude détaillée du parcours des liaisons sur le circuit imprimé adapteur révéla que tous les signaux étaient correctement amenés par le connecteur 14 broches Atari, à une exception près.

Dans la mesure où le lecteur ne s'est pas vu affecter un numéro au moyen d'une liaison électrique ("jumper"), la sélection se fait grâce au contrôleur.

Ceci est toujours le cas avec les lecteurs 3 pouces et demi, Atari a prévu les lignes 5 (drive 0 select) et 6 (drive 1 select) à cet effet. Le signal de la ligne 6 est directement relié à la prise de sortie, mais pas au connecteur 34 broches ; manifestement la manipulation que j'allais faire n'a pas été prévue par Atari. Normal, on veut vendre ses propres lecteurs.

Voici maintenant le détail de la modification. Pour commencer, le petit morceau de câble plat a été remplacé par une portion de câble de 1 m. Un connecteur à sertir 34 broches a été mis en place à 10 cm de chaque extrémité du câble. Comme vous pouvez le voir sur la photo suivante, le signal Drive 1 Select a été relié à la broche 12 du bon vieux bus Shugart au moyen d'un "strap". Pour terminer, le câble a été disposé de telle manière qu'aucune modification irréversible, du genre perçage d'un trou passe-fil, n'ait due être faite sur le drive ST.



Grande fut la surprise lorsque, après avoir connecté un drive Philips 5 1/4 double-tête, toute tentative de formatage d'une disquette resta sans effet, excepté un message dans le genre "lecteur B ne répond pas".

Pourtant la connexion était correctement réalisée, le lecteur et la disquette étaient également OK.
Finalement c'est l'oscilloscope qui permit de trouver la cause de ce mutisme, il révéla en effet que le signal Select du ST était très affaibli.

Le prix fabuleusement bas du ST doit bien s'expliquer d'une manière ou d'une autre, en l'occurence par l'abscence de drivers de bus.
Effectivement, le lecteur Philips utilisé était équipé d'une résistance de pull-up de 120 Ohm, une valeur trop faible pour le contrôleur de disque du ST.

La résistance fut donc purement et simplement déconnectée à la pince coupante et une nouvelle tentative de formatage apporta enfin les 726016 octets libres tant espérés.

En guise de conclusion je peux dire que jusqu'ici je n'ai rencontré aucun problème de lecture ou d'écriture. En principe n'importe quel drive relativement récent doit pouvoir se connecter de cette manière au ST, que ce soit un 5 1/4" ou 3,5". Le "récent" se rapporte au temps d'accès à une piste. Ce temps ne doit pas dépasser 5, ou mieux 3ms pour passer d'une piste à une autre. Les oldies avec des temps d'accès de 40 ms ou plus ne conviennent à aucun prix !

DOSSIER

MEGA JACK

Le MEGA-JACK
ou comment doubler la capacité mémoire de votre 520 ST !

Mon tout premier ordinateur (l'histoire se passe en 1978) était un système monocarte que j'avais construit moi-même autour d'un 8080 et qui disposait de 1024 octets de RAM. Et , chose qui me parut inconcevable au début, il arriva un jour où la RAM fut pleine. Je fis donc chauffer à nouveau le fer à souder pour ajouter une extension mémoire. Après un travail fructueux je me retrouvai donc avec 2048 octets.

Aujourd'hui nous sommes en 1986 et cela fait déjà un certain temps que ce "nano-ordinateur" a pris sa retraite. Ma dernière acquisition, un Atari ST, dispose en série d'une formidable mémoire de 512 Ko.
"Je tiendrai jusqu'en l'an 2000" m'étais-je dit au moment de l'achat. Mais au bout de quelques semaines je changeai déjà d'avis. La première pré-version de l'interpréteur BASIC ne laissait qu'un peu plus de 1000 octets libres pour un embryon de programme. Il fallait faire quelque chose. Et effectivement je trouvai la solution. Tout comme en 1978 je fis chauffer mon fer à souder et portais au double la capacité mémoire. Maintenant je dispose d'un ST avec 1 Méga Octet (affectueusement baptisé MEGA-JACK) et j'attends impatiemment le jour où il affichera le message "OUT OF MEMORY".

LA MMU GERE TOUTE LA MEMOIRE

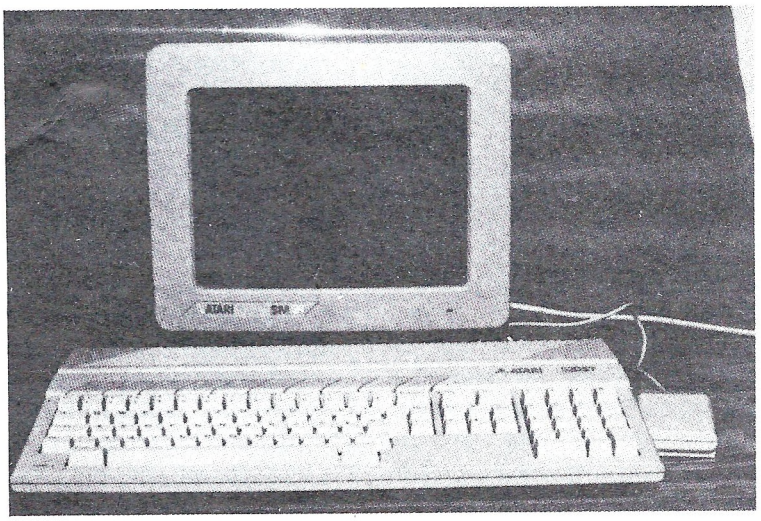
Les concepteurs du ST ont fait appel à une multitude d'astuces particulièrement bien pensées. Rien que la gestion de la mémoire, qui permet de choisir quantité de configurations mémoire différentes, est fascinante. Au coeur de la gestion mémoire on trouve un circuit développé par Atari et qui porte la désignation MMU. MMU veut dire MEMORY MANAGEMENT UNIT.

La conception particulièrement réussie de ce circuit fait qu'il est possible de brancher n'importe quel type de boîtier mémoire, que ce soit des puces de 64 Kbits, 256 Kbits ou 1 Mbits.
Examinons un peu la configuration mémoire du ST. Comme ce dernier est équipé du 68000, un microprocesseur 16 bits avec un vrai bus de données de 16 bits, on trouve 16 circuits intégrés du type 41256. Chacun de ces 16 circuits peut stocker 256 Kbits. Toutefois la MMU n'impose pas du tout l'emploi de ces circuits. On pourrait aussi bien utiliser des circuits de capacité inférieure, par exemple des 64 Kbits ; on obtiendrait alors le 128 ST (annoncé mais jamais fabriqué). A l'opposé on pourrait également faire appel aux nouvelles puces de 1 Megabits, ce qui ferait du ST un 2 MEGA ST.

Néanmoins le prix de 16 de ces circuits atteindrait encore une valeur astronomique. En tous cas la chose est réalisable grâce à la clairvoyance des concepteurs qui ont su adapter le matériel et le logiciel à cette éventualité.

DES CONNEXIONS INUTILISEES

Mais les inventeurs du ST ne se sont pas arrêtés en si bon chemin. Par chance un schéma du ST fut publié très tôt, et celui-ci révéla une information de tout premier ordre. En effet la MMU dispose de connexions supplémentaires, donc in-



utilisées par le ST, qui permettent d'ajouter jusqu'à 16 autres boîtiers mémoire ! Avec un peu de dextérité (et bien sûr la menue monnaie nécessaire) on peut étendre la mémoire du ST de 128 K, 512 K ou plus. La solution la moins onéreuse consiste en une extension de 128 K. Les circuits de 64 Kbits employés se trouvent chez certains revendeurs pour moins de 15 francs, de sorte que l'extension complète revient à moins de 250 francs.

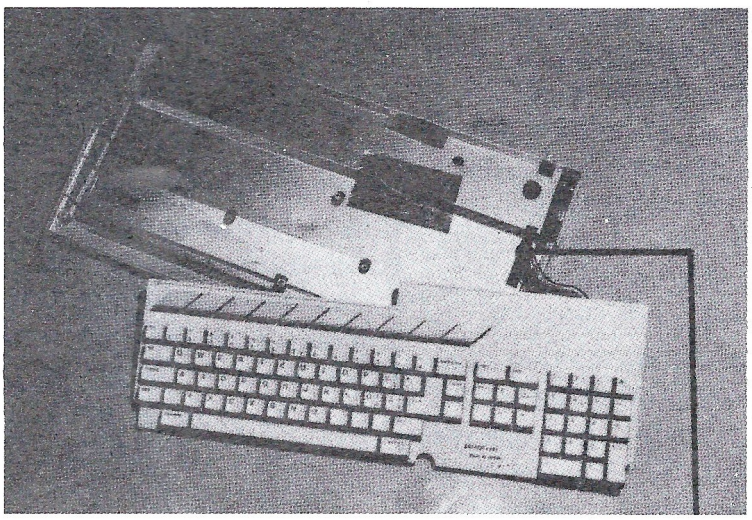
J'ai personnellement opté pour les puces de 256 Kbits, plus chères puisqu'elles coûtent aux alentours de 80 à 100 francs pièce.
Le prix de revient des composants pour l'extension de 512 Ko se situe alors aux environs de 1500 francs. Mis à part 50 cm de fil de câblage et un peu de soudure, les frais s'arrêtent là.

RESERVE AUX VIRTUOSES DU FER A SOUDER

La réalisation à proprement parler est à déconseiller à toute personne n'ayant pas une longue expérience du fer à souder. Si votre seule expérience en la matière se résume à la réparation d'un pot d'échappement ou du calfeutrage d'une gouttière au chalumeau, mieux vaut vous abstenir. En effet les circuits supplémentaires devront être montés à cheval sur les 16 boîtiers déjà existants. Si vous avez de la chance, ces derniers sont insérés dans des supports, ce qui vous facilitera grandement la tâche. Autrement il faudra opérer avec une extrême minutie. Par ailleurs il vous faut mesurer les conséquences d'une intervention dans la machine pendant la période de garantie. Une erreur de manipulation peut vous coûter cher si par votre intervention vous avez perdu le bénéfice de la garantie. Si tous ces arguments ne vous font pas reculer, vous pouvez à présent aller acheter les composants. Veillez à ce que les mémoires aient un temps d'accès de 150 ou 120 nanosecondes. Les circuits plus lents ne fonctionnent pas.

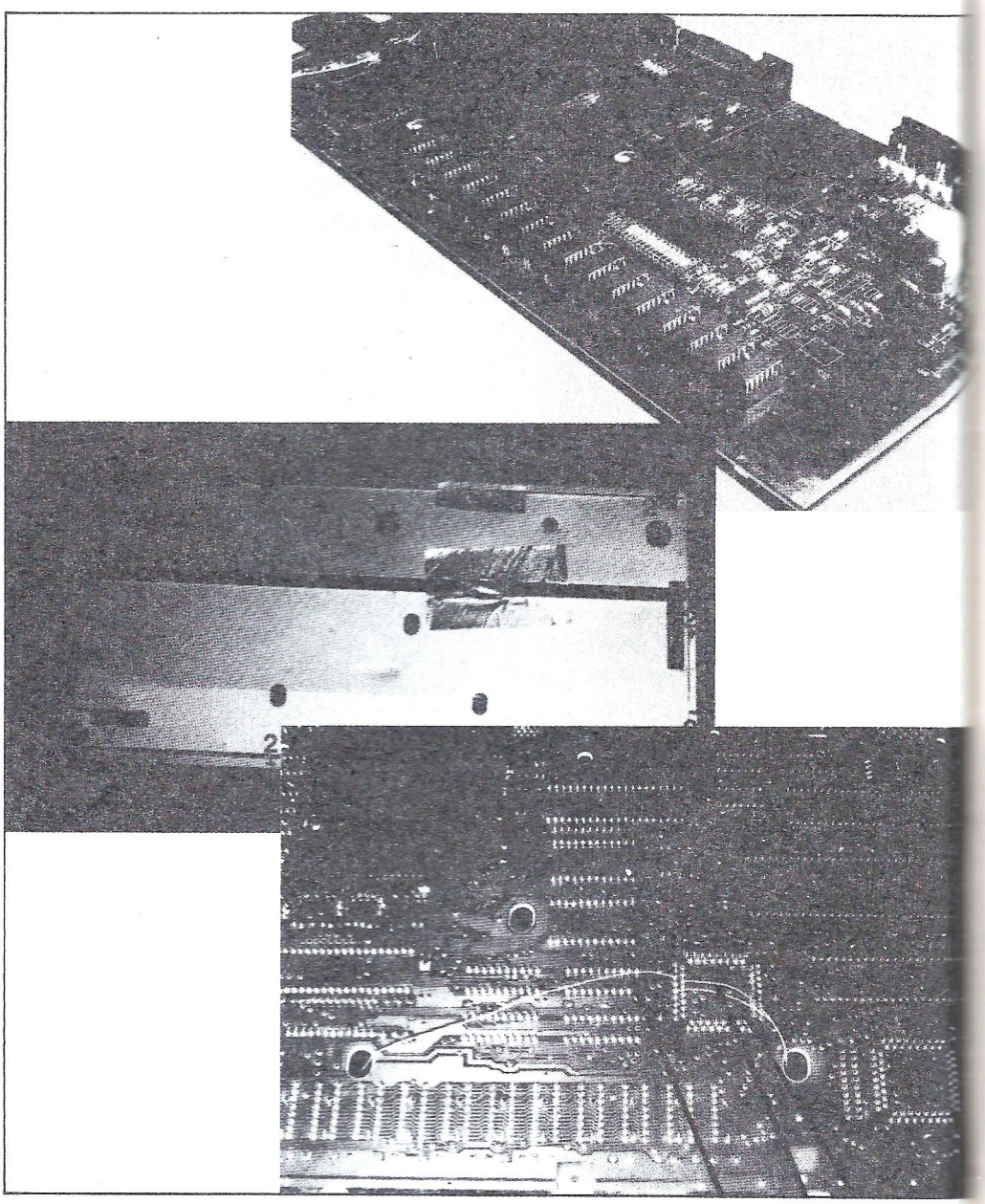
Voici comment s'effectue le montage. Commencez par enlever toutes les connexions allant au ST. Retournez ensuite l'ordinateur, le clavier vers le bas. Un chiffon entre la table et la machine ne peut pas faire de mal.

Le fond du boîtier est fixé au moyen de 6 vis cruciformes que vous allez ôter à l'aide d'un tournevis approprié. Retournez à nouveau la machine, maintenant les deux moitiés se séparent aisément. Le résultat est visible en figure 1. Sur le côté droit le clavier est relié à l'unité centrale au moyen d'un connecteur.
Retirez doucement le connecteur de sa prise.
Le clavier ainsi dégagé révèle un blindage métallique qui entoure complètement l'unité centrale.
Ce blindage est fixé à l'aide de languettes de fixation et de vis cruciformes (figure 2). Rabattez délicatement les languettes avec une pince plate. Retirez la plaque de blindage et reprenez le tournevis cruciforme. Il est maintenant possible d'extraire le circuit imprimé. Si tout s'est bien passé jusqu'ici, je vous suggère une pause café.



Le prochain passage s'adresse à ceux qui ont dépensé les 1500 francs pour les circuits 256 Kbits. Si les CIs de votre ST sont montés sur supports, je vous conseille de commencer par tester vos nouveaux circuits en les insérant à la place des composants d'origine. Si les mémoires ne fonctionnent pas, soit vous n'avez pas inséré correctement toutes les pattes dans le support, soit on vous a vendu de la "camelote". Si le test s'est avéré positif vous pouvez passer à la phase suivante. Pour cela il vous faut une pince plate et un fer à souder (30 W max., régulé en température, avec une panne très fine). Mais avant de faire entrer ce dernier en action, recourbez à angle droit les pattes 4 et 15 des 16 boîtiers de RAM. Elles seront traitées à part. Si par contre vous avez choisi la solution économique (128 Ko) il faudra également replier les pattes 1 des 4164, celles-ci resteront en l'air.

Par contre les broches 1 des 41256 doivent être reliées aux pattes correspondantes du CI inférieur. Si les RAM de votre ST sont montées sur support, retirez les. Soudez ensuite les circuits préparés précédemment sur les boîtiers d'origine. Cette opération nécessite une attention particulière et beaucoup de calme. Les CIs doivent être superposés de manière à ce que les encoches pointent dans la même direction. Soudez toutes les pattes à l'exception de celles qui ont été recourbées. Au risque de me répéter : apportez le plus grand soin à cette manipulation, la vie des RAM en dépend.



Si vous avez pu extraire les CIs de la platine vous pouvez maintenant remettre dans leurs supports les paquets ainsi formés. Vous n'avez pas noté le sens ? Les encoches pointent en direction des connecteurs.

LIAISON AVEC LA MMU

La deuxième phase va consister à relier à la MMU les connexions laissées en l'air. Pour commencer il faut relier ensemble toutes les pattes 4 des 16 boîtiers. La meilleure solution consiste à utiliser du fil de câblage fin, si possible argenté. La broche 4 (c'est l'entrée RAS de la mémoire) doit ensuite être soudée

à la patte 18 de la MMU. Pour cela vous pouvez faire passer le fil dans un des trous pratiqués au niveau des entretoises sur lesquelles vient se fixer la deuxième moitié du ST ; voir aussi figure 3. C'est encore une fois un travail de précision car les soudures au niveau du support de la MMU sont très rapprochées.

Restent les pattes 15 des mémoires (entrée CAS).

Les broches des huit circuits de droite et celles des huit circuits de gauche doivent respectivement être reliées ensemble.
Les pattes de la rangée de gauche (correspondant aux positions U16 à U30) seront reliées à la broche 21 de la MMU tandis que les pattes 15 des CIs U32 à U45 iront sur la patte 22 de la MMU. Les numéros de broches de la MMU sont d'ailleurs inscrits sur le circuit imprimé, côté composants. En cas de doute, vous pouvez consulter la figure 4 où les différentes connexions sont facilement repérables.

Avant de poursuivre, passez en revue toutes les soudures, vérifiez que vous n'en avez oublié aucune, qu'il n'y a pas de "soudure sèche" ou de court-circuit.
A présent plus rien ne s'oppose à un test de fonctionnement.
Reposez le circuit imprimé dans la partie inférieure du boîtier, sans les plaques de blindage.
Enfichez le connecteur du clavier ainsi que les câbles de l'alimentation, du moniteur, du lecteur de disquettes et de la souris. Lorsque vous mettez en marche le ST, il doit "booter" comme d'habitude, c'est-à-dire qu'il charge et lance le système d'exploitation. Ne vous attendez pas à un message sensationnel à l'écran, l'ordinateur doit (espérons-le !) fonctionner comme avant.
Par contre si vous chargez le BASIC, l'instruction PRINT FREE(0) doit vous fournir le nombre 535904 (cette valeur peut varier légèrement d'une version du BASIC à une autre). Cette quantité de mémoire disponible devrait largement suffire, même pour de gros programmes BASIC. Il faudrait réviser votre style de programmation si vos programmes BASIC venaient à dépasser cette taille !

ON REFERME LA BOITE

Si tout s'est bien passé vous pouvez remonter le ST.
Il faudra laisser de côté la plaque de blindage supérieure car la hauteur de notre empilement empêche sa mise en place. Ne la jetez pas, vous pourriez peut-être en avoir besoin un jour. Lors de l'assemblage des deux moitiés du boîtier il faudra faire attention aux vis, en effet celles-ci n'ont pas toutes la même longueur. Les plus courtes viennent se loger dans les trous de la partie avant.

QUE FAIRE DANS LE CAS OU CELA NE FONCTIONNE PAS

Premièrement (très important) : pas de panique !
Deuxièment (très important) : marquer une pause et se calmer.
Troisièment (très important) : vérifier à nouveau et très minutieusement toutes les soudures.

Si on vous a vendu des mémoires en bon état de marche, la panne vient nécessairement de votre travail.
Si la vérification susdite n'a rien apporté, redémontez les mémoires et vérifiez si votre ST fonctionne sans l'extension. Si c'est le cas, vous pouvez soit en rester là, soit recommencer toute l'opération par le début. Si par contre l'ordinateur ne fonctionne plus, il ne vous reste plus qu'à prendre le ST sous le bras et à vous rendre chez votre revendeur. Comme vous avez ouvert votre appareil pendant la période de garantie, l'addition risque d'être salée. C'est un risque que vous aviez pris sciemment.....

MICRO-ECHOS

POUR AMSTRAD CPC : TOUJOURS PLUS !

Bien sûr vous connaissiez déjà le tome I des TRUCS ET ASTUCES POUR LE CPC, comment faire autrement d'ailleurs ?
Notre standard a sauté le jour de sa sortie, et vous nous avez vraiment mis du baume au coeur avec vos innombrables courriers de félicitations.
Assez pour la pommade... Nous ne nous sommes pas endormis sur nos lauriers, et avons concocté le tome II des TRUCS ET ASTUCES.
Pourquoi ?
Avec le premier volume vous avez appris à rentrer au coeur de votre ordinateur, à vous y loger bien au chaud pour occuper vos longues soirées d'hiver. Mais, inévitablement, des questions se sont posées, se sont accumulées, bref le tome I ne suffisait plus.
Vous nous avez questionné, nous vous avons répondus, et avons décidé de faire de toutes vos interrogations ce nouveau livre.
En vrac, vous y trouverez :
programmes de tri, graphisme 3D, générateur de menus, impression de l'écran graphique, création de lignes BASIC à partir du BASIC, programmes en langage machine, générateur de masques de saisie, scrolling en douceur etc ...
Une foule de conseils pratiques et d'astuces pour la programmation en BASIC ou en langage machine vous éviteront bien des désespoirs. Vos programmes pourront être paufinés, optimisés, pour reculer encore les limites de la programmation.
Si vous ne voulez pas laisser dormir votre CPC, si vous avez peur qu'il devienne routinier, ce livre vous permettra d'en tirer le maximum et de ne plus jamais lui laisser un instant de repos.

LE LIVRE DU GEM SUR ATARI ST

Si vous êtes l'heureux possesseur d'un ATARI ST, ou si vous avez l'intention d'acheter cette merveilleuse petite machine, vous avez sans doute eu à un moment ou à un autre des interrogations sur ce petit mot de trois lettres : GEM.
Tout d'abord GEM cela veut dire : Graphic Environnement Manager, ce qui vous renseigne sur l'appelation, vous donne une vague idée de ce que cela peut être...
GEM est un système d'exploitation aussi étendu qu'extensible qui a été développé par DIGITAL RESEARCH pour le modèle ST d'ATARI.
Mais voilà, les manuels fournis par ATARI sur le GEM sont issus des systèmes implantés sur les calculateurs fonctionnant sous MS-DOS et je vous assure que le désespoir vous gagnera vite si vous voulez vous servir de GEM avec ces seuls manuels. Quand l'équipe de MICRO-INFO à voulu commencer à faire du développement sur l'ATARI, les poignées de cheveux arrachées étaient aussi nombreuses que les multiples problèmes qui se bousculaient au portillon de l'ATARI. Puis, après une cure de sommeil forcé, nous avons commencé à disséquer la gigantesque bibliothèque de GEM.
Vous trouverez dans **LE LIVRE DU GEM SUR ATARI ST**, le fruit de nos longues pratiques et reflexions autour du VDI (Virtual Device Interface) et de l'AES (Application Environnement System).
Nous avons essayé de vous décrire de la façon la plus complète possible, les mécanismes de programmation sous GEM, avec une introduction au langage C et à l'assembleur, et des exemples pratiques d'utilisation.
Nous n'avons pas oublié la description de l'éditeur, du compilateur, de l'assembleur, du linker, etc.
Puisse cet ouvrage vous éviter les heures de galère que nous avons passées, C'est vraiment tout le mal que je vous souhaite...

REPARATION SUR 1541

C'est un Commodoriste qui vous parle, un vrai, un pur, un dur. Oh bien sûr, j'ai l'occasion d'utiliser d'autres machines, mais je reste fidèle à ce bon COMMODORE 64.
Seulement voilà, la fidélité cela ne doit pas être à sens unique. Il n'y a rien qui m'horripile autant qu'un lecteur de disquettes qui trahit ma confiance. Il m'est déjà arrivé, au bout d'une longue nuit de labeur, d'être dans l'impossibilité de sauver un programme. Après avoir consulté des spécialistes en tout genre, la conclusion était simple : mauvais alignement de la tête.
Naïvement, je m'enquis de la marche à suivre, les réponses ont été unanimes, "ne touchez à rien, amenez votre unité chez un réparateur". La mort dans l'âme j'allai me décider à emmener l'enfant dans ma boutique préférée, quand un titre m'a accroché :
ENTRETIEN ET REPARATION DU LECTEUR DE DISQUETTE 1541, édité par **MICRO APPLICATION**
Je me ruai dessus et appris que, en fait, ce n'était pas si difficile de réajuster la tête de lecture ; tout y est expliqué, analysé, détaillé.
J'appris ensuite à le lubrifier correctement, à régler la partie électronique, et même à fabriquer une disquette d'alignement. Je n'ai pas eu besoin de faire appel à mes connaissances restreintes en électronique, tout y est clair, détaillé et agrémenté de nombreux shémas.
Ce livre décrit parfaitement la panoplie du petit docteur es 1541, un peu d'huile, un zeste de fer à souder, etc...
Je peux enfin m'occuper de la santé de mon lecteur, je lui facilite un peu la vie et, maintenant, il me le rend bien le bougre !

CONVERTISSEURS A/D ET D/A POUR L'AMSTRAD

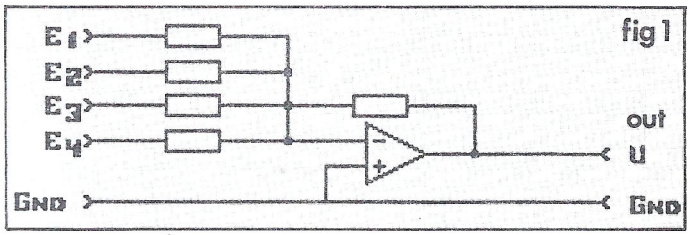


fig 1

Il fut un temps (il n'y a pas si longtemps de cela) où le mot TECHNIQUE DIGITALE n'existait pas encore. On évaluait l'heure grâce à la position que prenaient les aiguilles sur le cadran d'une montre. Sans parler de la très respectable règle à calcul, aujourd'hui reléguée au rang d'objet de musée. Autrefois toutes les grandeurs, que ce soit la température, même la parole et la musique, étaient mesurées de manière analogique (par exemple sous la forme de la dilatation du mercure dans le thermomètre).

Puis vint la technique digitale (ou numérique), s'installant peu à peu dans notre vie quotidienne : l'affichage digital de notre montre ou les calculatrices de poche n'en sont qu'un exemple.
La mesure de la température et même l'enregistrement de la musique ne résistèrent pas longtemps à la poussée de cette nouvelle technique. Toutefois il convient de distinguer plusieurs cas d'espèces. Alors que la montre digitale et la calculatrice font appel à un processus purement numérique, nos deux derniers exemples nécessitent d'abord une conversion entre un phénomène analogique, la température ou l'onde acoustique, et une représentation numérique. En clair cela veut dire que l'on a besoin d'un dispositif assurant cette conversion. On l'appellera convertisseur analogique/digital ou convertisseur digital/analogique, suivant que l'on passe d'une grandeur analogique à une grandeur numérique ou inversement.

Nous allons aujourd'hui nous intéresser à la possibilité de réaliser un tel convertisseur pouvant se connecter à notre ordinateur et dont la mise en oeuvre serait la plus simple possible.

L'ordinateur en question sera un AMSTRAD qui, par suite de l'absence d'un "user-port", n'est pourtant pas parfaitement adapté à la commande de circuits externes.
Mais même sans un tel "user-port" il est néanmoins possible de réaliser un certain nombre d'applications tout à fait intéressantes. Cet article mettra un accent particulier sur la théorie des différents convertisseurs. Pour cette raison ne considérez pas les schémas des réalisations qui vous sont proposées comme quelque chose d'imperfectible. Bien que tous les montages aient été testés sur leur fonctionnement et leur réalisabilité, j'espère qu'ils vous inciteront à faire d'autres expériences.

Commençons par la production de signaux analogiques. Ces signaux sont constitués de tensions de différentes valeurs. Avant d'entrer dans le vif du sujet, intéressons nous à un convertisseur d'une toute autre espèce : la résistance variable, également connue sous le nom de potentiomètre. Un tel potentiomètre peut être considéré comme un convertisseur d'angle de rotation/tension. En effet suivant l'angle de rotation de l'axe du potentiomètre, on trouvera à ses bornes une tension comprise entre 0 et une valeur maximale. L'exactitude de la correspondance dépend de l'homogénéité de la couche de carbone du potentiomètre, elle n'est en tous cas jamais parfaite.

On rencontre ce même type de problème avec les convertisseurs D/A. Les variations de tension entre les valeurs minimales et maximales ne se font pas de manière continue mais par paliers, leur nombre étant dicté par la technologie du dispositif.

Pour commencer, contentons nous pour notre première étude d'un convertisseur D/A ayant une résolution de 4 bits ; la figure 1 en présente un modèle simplifié. Ce montage (que vous ne devez pas obligatoirement réaliser, il n'a rien de formidable) est essentiellement constitué d'un amplificateur opérationnel et de 5 résistances. Associés de telle manière, ils forment un amplificateur additionneur.
En sortie on trouve un signal égal à la somme de toutes les tensions présentes sur les différentes résistances d'entrée.

En supposant que notre ordinateur soit capable de fournir sur 4 lignes une tension de 0 ou 1 volt, ce montage nous permettrait d'obtenir 5 tensions différentes en sortie. Comment , seulement 5 ?
L'informaticien, qui est en vous, sait pourtant qu'avec 4 bits on peut former 16 combinaisons distinctes. La différence vient du fait qu'il existe des combinaisons redondantes, c'est-à-dire que certaines combinaisons produisent la même tension de sortie. Pour y remédier il existe deux solutions. Soit on fournit aux entrées des tensions différentes, soit on donne des valeurs différentes aux résistances. Voyons tout d'abord la première possibilité. L'entrée E1 recevra une tension de 100 millivolts, E2 200 mV, E3 400 mV et E4 800 mV. Ce qui nous donne les combinaisons suivantes :

E1	E2	E3	E4	SORTIE
0V	0V	0V	0V	0V
0.1V	0V	0V	0V	0.1V
0V	0.2V	0V	0V	0.2V
0.1V	0.2V	0V	0V	0.3V
.				.
.				.
.				.
0V	0.2V	0.4V	0.8V	1.4V
0.1V	0.2V	0.4V	0.8V	1.5V

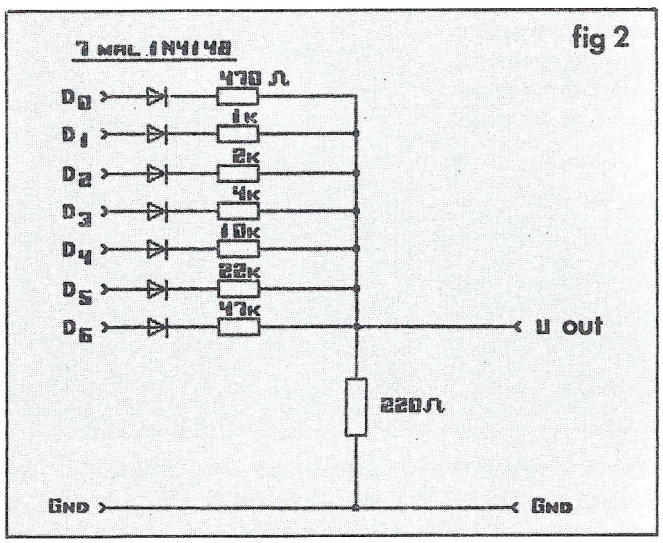
On obtient un effet similaire en variant non pas les tensions mais les résistances d'entrée. Comme pour les tensions, chaque nouvelle résistance sera doublée par rapport à la précédente. La figure 2 montre une réalisation pratique d'un tel dispositif.

On a même supprimé l'amplificateur opérationnel et porté le nombre des entrées à 7. On prélèvera la tension de sortie aux bornes de résistance R8. Avec les 7 entrées nous avons maintenant 128 niveaux de sortie différents. L'avantage principal de la suppression de l'ampli OP réside dans le fait qu'on ne nécessite plus de tension d'alimentation séparée. En contrepartie l'impédance de sortie est assez élevée, mais on pourra prélever le signal à l'aide d'un petit amplificateur et le rendre ainsi audible.

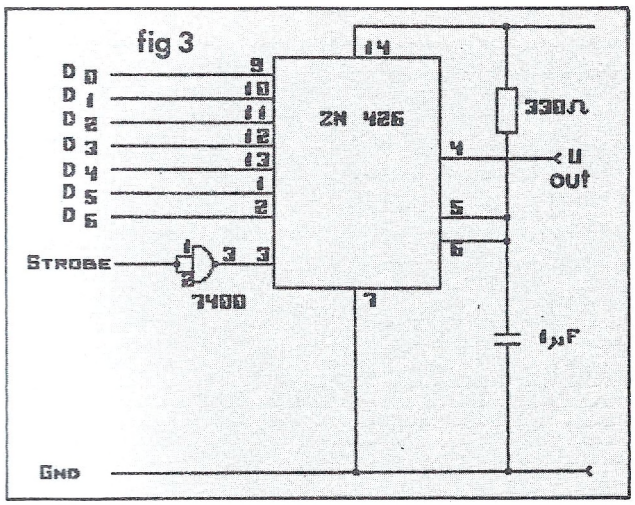
Les diodes sont là pour que les entrées à 0 n'affectent pas la sortie. S'il n'y avait pas ces diodes, chaque entrée à 0 mettrait la résistance qui lui correspond à la masse et ferait chuter la tension de sortie. Le port centronics, auquel nous connectons notre montage, est accessible par l'adresse &EF00. L'instruction BASIC...

OUT &EF00,0

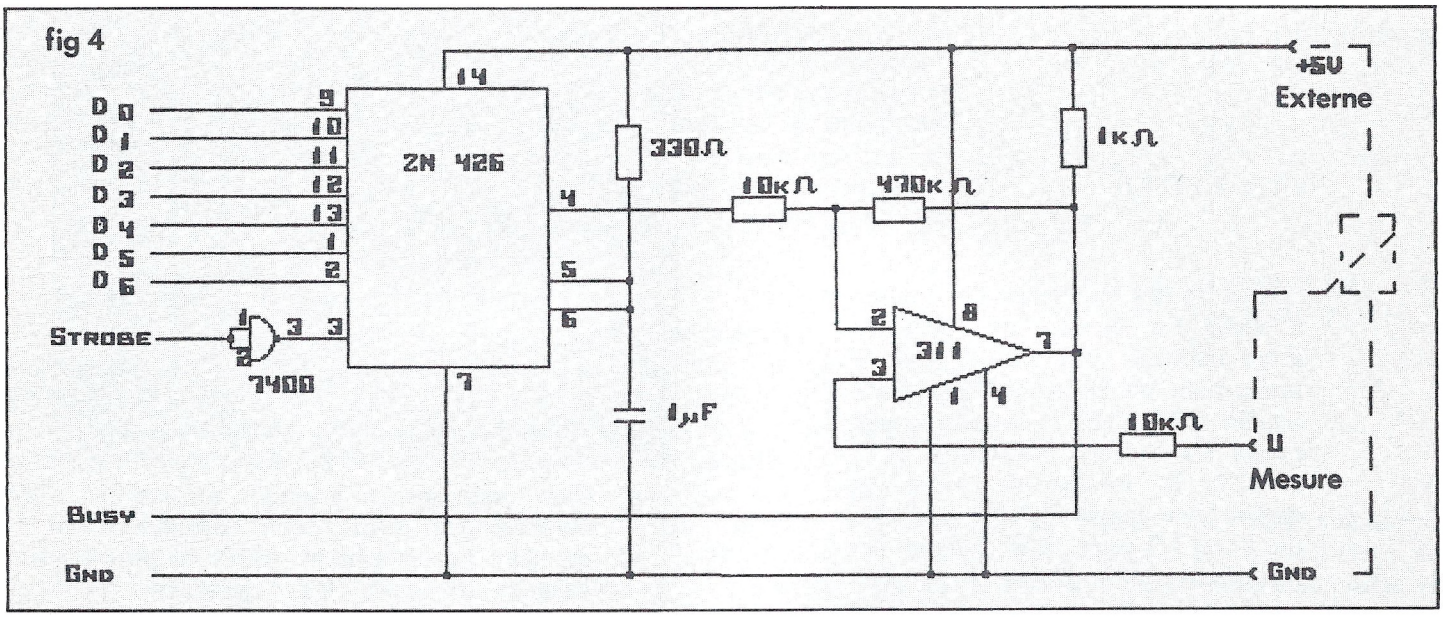
... met toutes les lignes de données à zéro. La tension de sortie de notre montage vaut alors 0 volt. Si maintenant on envoie à l'aide d'une boucle FOR NEXT successivement toutes les valeurs croissantes de 0 à 127, la tension de sortie va croître par bonds de 10 millivolts de 0 à 1.2 volt. Les valeurs des résistances de la figure 2 réalisent un compromis entre une facilité d'approvisionnement et une bonne exactitude de la tension de sortie.



Si on voulait améliorer la résolution tout en gardant ce même principe on atteindrait rapidement des impédances d'entrée inexploitables. Par exemple si on augmentait le nombre d'entrées de la figure 2 à 12, la plus grande résistance vaudrait déjà 2.2 MOhm. De telles valeurs de résistances sont très difficiles à intégrer.



C'est pourquoi on a plutôt recours au procédé dit R/2R. Cette technique permet de n'utiliser que deux valeurs distinctes de résistances. Le ZN 426 de chez FERRANTI constitue un convertisseur intégré fonctionnant sur ce principe. Le ZN 426 est particulièrement simple à connecter à un micro-ordinateur. Il se contente d'une alimentation +5 volt, d'une résistance et d'un condensateur. La tension de sortie maximale vaut 2.5 volt ce qui nous donne, avec 8 bits d'entrée, à nouveau une résolution de 10 millivolts. La plus petite tension de sortie que l'on puisse obtenir est 3 mV. La figure 3 montre comment relier ce convertisseur au CPC. Pour exploiter la résolution maximale de 8 bits on se sert du signal STROBE comme huitième bit. Ce bit étant inversé à l'intérieur du CPC, il a fallu ajouter un inverseur dans notre montage.
Bien entendu la programmation se fait toujours par l'adresse &EF00, mais cette fois on peut envoyer des valeurs comprises entre 0 et 255.
Avec ce même CI il est également possible de réaliser un convertisseur A/D. Il suffit pour cela d'adjoindre au convertisseur D/A que nous venons de voir, un amplificateur opérationnel, quelques résistances et un peu de programme. La figure 4 montre un convertisseur A/D complet réalisé avec un ZN 426.

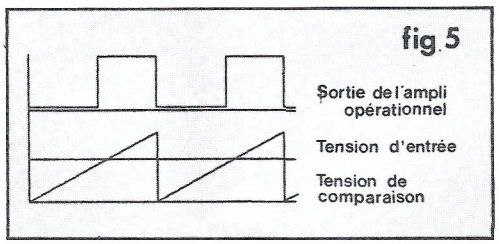


Comment est-ce que cela fonctionne ?

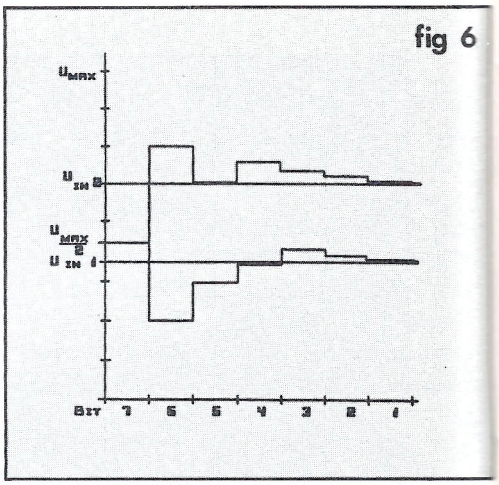
Le principe en est assez simple. Nous allons étudier son fonctionnement à l'aide d'un exemple. Appliquons une tension de 1 Volt à l'entrée du convertisseur. Par programme on commence par faire en sorte que la sortie du ZN 426 (c.à.d. un convertiseur D/A) soit à 0. L'amplificateur opérationnel, monté en comparateur, voit donc sa sortie passer à 0 Volt.

```
100 '*****************************************************************
110 '
120 port%   = &EF00
130 status% = &F500
140 busy%   = &X10000000
150 '
160 '*****************************************************************
170 '
180 FOR mesure% = 0 TO 255
190 OUT port%,mesure%
200 IF (INP(status%) AND busy%) THEN resultat% = mesure%: mesure% = 255
210 NEXT mesure%
220 PRINT resultat%
230 GOTO 180
```

Cette sortie est reliée à l'entrée BUSY du Port centronics du CPC, il est par conséquent possible de lire l'état de sortie du comparateur. Notre programme va donc incrémenter le mot de sortie aussi longtemps que la ligne BUSY est au niveau bas. Il arrivera un moment où la tension de sortie du ZN 426 dépassera la tension à mesurer. Le comparateur basculera alors et sa sortie passera au niveau haut (+5), transmis à la ligne BUSY. Le dernier mot fourni au convertisseur D/A constituera la valeur digitale correspondant à la tension mesurée. Un exemple de programme qui réalise ce que nous venons de voir vous est proposé en listing 1.



Cette façon de faire présente néanmoins un inconvénient : la durée de la conversion est assez importante et dépend de la tension à mesurer. Une tension élevée nécessitera un temps de mesure beaucoup plus important qu'une tension faible. Comme la tension à mesurer ne doit pas varier pendant le cycle de mesure, il faudra prévoir pour toutes les mesures un délai égal au temps le plus long. De plus si on veut faire un grand nombre de mesures on a intérêt à ce qu'elles soient les plus courtes possibles.



Pour toutes ces raisons on préférera une autre méthode, la technique dite des approximations successives. Dans ce cas on ne fait pas varier linéairement la tension de comparaison, mais plutôt suivant un schéma spécial qui rappelle certains algorithmes de recherche ou de tri. On commence par envoyer une tension de comparaison égale à la moitié de la tension maximale (en écrivant dans le Port Centronics la valeur 128 ou &80 ou %10000000). Puis on lit la ligne BUSY qui indique si on a dépassé ou non la tension à mesurer.

```
100 '*****************************************************************
110 '
120 FOR indice%=7 TO 0 STEP -1: bit%(7-indice%)=2^indice%: NEXT
130 '
140 '*****************************************************************
150 '
160 port%   = &EF00
170 status% = &F500
175 busy%   = &X10000000
180 mesure% = 0
190 '
200 '*****************************************************************
210 '
220 FOR indice%=0 TO 7
230 mesure% = mesure% + bit%(indice%)
240 OUT port%,mesure%
250 IF (INP(status%) AND busy%) THEN mesure% = mesure% - bit%(indice%)
260 OUT port%,0
270 NEXT indice%
280 '
290 PRINT mesure%
300 mesure% = 0
310 GOTO 220
```

Si c'est le cas on revient à 0, sinon on reste à &80. Nous savons maintenant dans quelle moitié se trouve la tension. On ajoute alors 64 (ou &40 ou %01000000) à la valeur précédente et on regarde à nouveau si on a dépassé la tension d'entrée. Ce test est répété jusqu'à ce que l'on arrive au bit 0. On voit qu'avec cette méthode il suffit de 8 mesures, quelle que soit la valeur de la tension d'entrée ! Les figures 5 et 6 montrent les deux procédés. Elles sont très explicites et devraient éclairer ceux pour qui les explications semblaient trop confuses.

```
8000 (8000)            org &8000
8000 (00EF)  port      equ &ef      ;&ef00 port Centronics
8000 (00F5)  status    equ &f5      ;port B 8255, BUSY
8000 (0080)  hibit     equ &80      ;1ere valeur
8000 (0040)  busy      equ &40      ;bit 6 port B

8000 AF      start     xor a        ;val.mesure det. ds accu
8001 4F                ld  c,a      ;octet inf. port a 0
8002 16 80             ld  d,hibit  ;charge valeur a ecrire
8004 06 EF             ld  b,port   ;octet sup. Centronics
8006 82      suiv      or  a,d      ;res*(res OR bit suivant)
8007 ED 79             out (c),a    ;envoie valeur sur port
8009 5F                ld  e,a      ;stockage intermediaire
800A 06 F5             ld  b,status;teste bit BUSY
800C ED 78             in  a,(c)
800E E6 40             and a,busy   ;valeur trop grande?
8010 28 04             jr  z,suite  ;non
8012 7A                ld  a,d      ;si oui, effacer le
8013 2F                cpl         ;dernier bit
8014 A3                and a,e
8015 5F                ld  e,a
8016 06 EF   suite     ld  b,port
8018 ED 49             out (c),c
801A CB 3A             srl d        ;decale bit a droite
801C 7B                ld  a,e      ;fini?
801D 30 E7             jr  nc,suiv  ;sinon on recommence
801F 32 23 80          ld  (aux),a  ;stocker resultat
8022 C9                ret          ;c'est tout
8023 (0001)  aux       defs 1
```

Alors qu'avec la première méthode il suffisait de programmer une boucle qui compte à partir de 0, la méthode des approximations successives requiert un peu plus de réflexion. Vous trouverez en listing 2 un exemple de programme BASIC relativement rapide. Si l'exécution est encore trop lente pour l'application que vous désirez en faire, il ne vous reste plus qu'à faire appel au langage machine. Le listing 3 présente un tel programme, la fréquence des mesures est plus qu'honorable.



AMSTRAD; GRAPHISME 3D SUR MICRO

La représentation parfaite des choses et objets qui nous entourent, suivent les lois rigoureuses de la perspective et commence avec l'histoire de la peinture occidentale. D'abord l'art Grec et l'époque du style 'géométrique', mais aussi et surtout l'art romain avec des fresques remarquables, tel ce mur peint en trompe-l'oeil (vers 60 après JC) découvert à Pompéi, témoin d'une maitrise parfaite de la perspective.

Aujourd'hui, le graphisme en général est devenu une des fonctions majeures des ordinateurs, pour de multiples applications technologiques (C.A.O.), statistiques (courbes, camemberts...), mais aussi artistiques (images de synthèse), publicitaires ou ludiques...

D'abord réservées aux 'gros systèmes', les images numériques sont apparues sur mini ordinateur.
Maintenant, la conception des 'micros' ayant en quelques années fait un extraordinaire bond en avant, de nombreux logiciels graphiques complexes sont disponibles sur des machines familiales telles que l'Amstrad CPC.
SPACE MOVING, récemment sorti chez MICRO APPLICATION, est un exemple de logiciel de création et d'animation de figures en 3D, en technique 'fil de fer'. En effet, après avoir défini une figure dans l'espace point par point, on peut la faire évoluer, par rotation ou translation, de manière à la voir sous différents angles, et même enregistrer ces mouvements.

D'une manière générale, on distingue deux types d'images en trois dimensions:

- les images 'en fil de fer' utilisées en CAO, où seules les arêtes sont représentées, ce qui réduit considérablement les calculs.
- les images de synthèse, de plus en plus utilisées par les chaines télévisées ; leur réalisme est très poussé et les techniques de réalisation complexes.

Nous essayons ici de vous présenter, de façon superficielle, quelques outils mathématiques pour la construction de volumes en 3D et la transformation en 2D, c'est à dire la projection sur un plan (ici, le plan de l'écran). Nous nous contenterons de vous communiquer ces outils, car, dans le cadre de cet article, il nous serait impossible de procéder à un cours d'algèbre et de géométrie dans l'espace. Pour une approche plus complète du problème, nous vous conseillons de vous rapporter aux nombreux ouvrages publiés sur le sujet, dont nous vous donnons une liste non exhaustive en fin d'article.

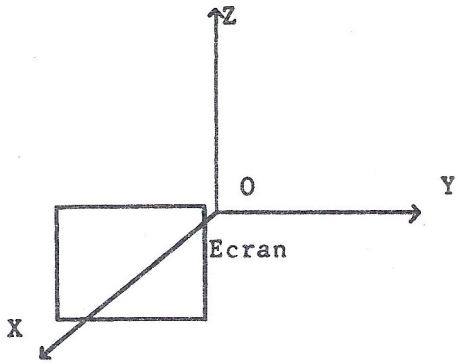
Il existe deux possibilités de variation de l'angle de vue :

- l'objet est mobile dans son repère, la position de l'oeil de l'observateur restant fixe.
- inversement, l'objet est immobile et c'est la position de l'oeil de l'observateur qui se déplace.

Nous avons arbitrairement choisi de vous présenter la première option. Toutefois, nous espèrons pouvoir vous présenter dans un prochain numéro la deuxième, ainsi que d'autres étapes de la CAO telles que le découpage du champ de vision, les cercles et arcs de cercles, le traitement des faces cachées, etc...

DEFINITION DE L'OBJET

Tout d'abord, il faut situer chaque point de l'objet par trois coordonnées, dans un repère à trois dimensions (trièdre), que nous orienterons de façon directe, l'axe Ox étant perpendiculaire à l'écran :



Trièdre direct

Les trois coordonnées, suivant les trois axes Ox, Oy et Oz, s'appellent respectivement l'abscisse, l'ordonnée et la côte.

La première opération sera donc la saisie des 3 composantes de chaque point, dans un tableau dimensionné. La deuxième sera la définition des arêtes, c'est à dire des couples de points qui devront être reliés, dans un second tableau. Dans le cas d'un cube de 100 de côté et posé sur l'origine, nous aurions :

- 8 points ;
  Point 1: (0,0,0)
  Point 2: (100,0,0)
  Point 3: (100,100,0)
  ...(0,100,0),(0,0,100),
  (100,0,100),(100,100,100),
  (0,100,100)

- 12 arêtes ;
  Arête 1: (1,2)
  (relier les points 1 et 2)
  Arête 2: (2,3)
  ...(3,4),(4,1),(5,6),(6,7),
  (7,8),(8,5),(1,5),(2,6),
  (3,7),(4,8)

Dans le petit programme basic suivant, ces opérations sont effectuées dans les lignes 50 à 200.

Vous pourrez les faire varier pour apprécier leurs effets.
Les commandes sont :

- (1) (2) (3) pour les translations Ox, Oy et Oz ;
- (4) (5) (6) pour les rotations autour des axes Ox Oy et Oz ;
- (7) pour le zoom ;
- (8) pour l'inversion des commandes.

LES MOUVEMENTS

- les TRANSLATIONS : soit un point A (x,y,z) translaté de tx,ty et tz, le long des axes principaux. Ses nouvelles coordonnées seront :

A (x+tx, y+ty, z+tz)

- les ROTATIONS : soit A (x,y,z) et teta l'angle de rotation.

posons st=sin(teta) et ct=cos(teta)

- rotation autour de l'axe des abscisses
A (x, y*ct+z*st, -y*st+z*ct)
- rotation autour de l'axe des ordonnées
A (x*ct-z*st, y, x*st+z*ct)
- rotation autour de l'axe des côtes
A (x*ct+y*st, -x*st+y*cp, z)

- le ZOOM soit A (x,y,z) et 'zoom' le coefficient du zoom.

A (x*zoom, y*zoom, z*zoom)

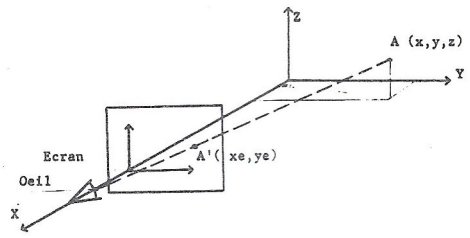
- l'INVERSION des déplacements :

- translations : tx=-tx, ty=-ty, tz=-tz
- rotations : st=-st ct=ct
- zoom : zoom=1/zoom

Dans le petit programme basic suivant, ces opérations sont effectuées dans les lignes 260 à 550.

LA PROJECTION

Nous postulerons que l'oeil de l'observateur se trouve sur l'axe Ox à une abscisse 'Oeil' et le plan de l'écran à une autre abscisse 'Ecran' :



Projection sur le plan de l'écran

On a les relations suivantes :

ye=y*(Oeil-Ecran)/(Oeil-x)
ye=z*(Oeil-Ecran)/(Oeil-x)

Relations que l'on peut simplifier en fixant l'abscisse du plan de projection (l'écran) à l'origine du repère, soit Ecran=O. On a alors :

xe=y/(1-x/Oeil)
ye=z/(1-x/Oeil)

Dans le petit programme basic suivant, nous avons fixé la valeur de 'Oeil' à 500. Vous pouvez faire varier cette valeur pour modifier la déformation de l'objet, due à la perspective. Toutefois, cette valeur ne doit pas être inférieure ou égale à l'abscisse de tout point de l'objet, car la déformation serait fausse ou infinie (division par O).

PROGRAMME BASIC

Il s'agit vraiment du minimum utile pour une première visualisation. A vous de l'améliorer par des menus, la possibilité de sauvegarder les figures créées, etc...

Les valeurs fixées par défaut en ligne 30 sont :

- trans : incrément pour les translations à 20 ;
- teta : incrément pour les rotations à 15 degrés ;
- zoom : coefficient du zoom à 1.2 (et donc 0.833) ;
- oeil : distance de l'observateur à 500.

BIBLIOGRAPHIE

- 'Graphisme 3D'
de M. ROUSSELET,
collection MICRO-SYSTEMES.

- 'Graphisme scientifique'
de R.DONY,
aux éditions MASSON.

-'Graphisme 3D SUR VOTRE ORDINATEUR'
de JL VULDY,
aux éditions EYROLLES.

- 'PRINCIPALES OF INTERACTIVE COMPUTER GRAPHICS'
de NEWMAN et SPROULL,
Mc GRAW HILL (deuxième édition).


```
10 '-------------PERSPECTIVES
20 MODE 2:DEG:ORIGIN 320,200
30 trans=20:teta=15:zoom=1.2:oeil=500
40 st=SIN(teta):ct=COS(teta)
50 '-------------SAISIE DES COORDONNEES
60 INPUT".Nombre de points:",nbpoint%
70 DIM point(nbpoint%,3),point%(nbpoint%,2)
80 FOR i=1 TO nbpoint%
90 PRINT:PRINT"* Point numero:";i
100 INPUT"abscisse:",point(i,1)
110 INPUT"ordonnee:",point(i,2)
120 INPUT"cote     :",point(i,3):NEXT i
130 '-------------SAISIE DES ARETES
140 CLS:INPUT".Nombre d'aretes:",nbarete%
150 DIM arete%(nbarete%,2)
160 FOR i=1 TO nbarete%
170 PRINT:PRINT"* Arete numero:";i
180 INPUT"point origine:",arete%(i,1)
190 INPUT"point arrivee:",arete%(i,2)
200 NEXT i:GOSUB 570
210 '-------------COMMANDES
220 a$="":WHILE a$="":a$=INKEY$:WEND:a=VAL(a$)
230 ON a GOSUB 260,270,280,330,390,450,510,550
240 GOTO 220
250 '-------------TRANSLATIONS
260 i=1:GOTO 290
270 i=2:GOTO 290
280 i=3
290 FOR j=1 TO nbpoint%
300 point(j,i)=point(j,i)+trans
310 NEXT:GOTO 570
320 '-------------ROTATIONS XX'
330 FOR j=1 TO nbpoint%
340 a=point(j,2)*ct+point(j,3)*st
350 b=-point(j,2)*st+point(j,3)*ct
360 point(j,2)=a:point(j,3)=b
370 NEXT:GOTO 570
380 '-------------ROTATIONS YY'
390 FOR j=1 TO nbpoint%
400 a=point(j,1)*ct-point(j,3)*st
410 b=point(j,1)*st+point(j,3)*ct
420 point(j,1)=a:point(j,3)=b
430 NEXT:GOTO 570
440 '-------------ROTATIONS ZZ'
450 FOR j=1 TO nbpoint%
460 a=point(j,1)*ct+point(j,2)*st
470 b=-point(j,1)*st+point(j,2)*ct
480 point(j,1)=a:point(j,2)=b
490 NEXT:GOTO 570
500 '-------------ZOOM
510 FOR j=1 TO nbpoint%:FOR i=1 TO 3
520 point(j,i)=point(j,i)*zoom
530 NEXT i:NEXT j:GOTO 570
540 '-------------INVERSION
550 trans=-trans:st=-st:zoom=1/zoom:RETURN
560 '-------------PROJECTION
570 FOR j=1 TO nbpoint%
580 i=1-point(j,1)/oeil
590 point%(j,1)=point(j,2)/i
600 point%(j,2)=point(j,3)/i:NEXT
610 '-------------TRACE
620 CLG:FOR j=1 TO nbarete%
630 a=arete%(j,1):b=arete%(j,2)
640 MOVE point%(a,1),point%(a,2)
650 DRAW point%(b,1),point%(b,2)
660 NEXT:RETURN
```

LA PAGE Ø DU COMMODORE 128

0000:	0000	Sens des données Port Processeur 6510
0001:	0001	Registre des données Port Processeur 6510
0002:	0002	Octet de définitiondu Banc mémoire
0003:	0003	Ad.mémoirecompteurProgramme(Octetfort)
0004:	0004	Ad.mémoire compteur Programme (Octet faible)
0005:	0005	Octet de définition pour registre 'Status' du CPU
0006:	0006	Octet de définition de l'Accumulateur
0007:	0007	Octet de définition du Registre 'X'
0008:	0008	Octet de définition du Registre 'Y'
0009:	0009	Octet de définition du Pointeur de Pile
000A:	0010	Recherche guillemet à la fin d'une chaîne
000B:	0011	Colonne écran lors du dernier 'TAB'
000C:	0012	Drapeau Floppy: 0=LOAD, 1=VERIFY
000D:	0013	Nombre d'éléments, Pointeur tampon de saisie
000E:	0014	Dimensionpardéfautpourlacommande DIM
000F:	0015	Drapeau données 1: $00=numérique, $FF=chaîne
0010:	0016	Drapeaudonnées2:$00=flottant,$80=entier
0011:	0017	Drapeau : LIST, lecture DATA, GARBAGE COLLECTION
0012:	0018	<DORES, SUBFLG SUBSCRIPT REF. /USER FUNCTION CALL>
0013:	0019	Drapeau:$00=INPUT,$40=GET,$98=READ
0014:	0020	Drapeau : signe pour TAN/résult. d'1 comparaison
0015:	0021	<CHANNEL , POKER>
0016:	0022 - 0023	Numéro de ligne, valeur entière (Lo/High)
0018:	0024	Pointeur temporaire Pile de chaînes
0019:	0025 - 0026	Adresse de la dernière chaîne de caractères
001B:	0027 - 0029	3 Octets: pointeur temporaire de chaîne
001E:	0030 - 0032	3 Octets: pointeur temporaire de chaîne
0021:	0033 - 0035	3 Octets: pointeur temporaire de chaîne
0024:	0036 - 0037	2 Octets: pointeur provisoire Index 1
0026:	0038 - 0039	2 Octets: pointeur provisoire Index 2
0028:	0040 - 0044	Résultat d'une Multiplication de Réel
002D:	0045 - 0046	Pointeur: Début de texte BASIC Lo/Hi
002F:	0047 - 0048	Pointeur: Début des variables BASIC Lo/Hi
0031:	0049 - 0050	Pointeur: Début des tableaux BASIC Lo/Hi
0033:	0051 - 0052	Pointeur: Fin des tableaux +1 Lo/Hi
0035:	0053 - 0054	Pointeur: Début des Chaînes de caractères Lo/Hi
0037:	0055 - 0056	Pointeur: Aide mise en mémoire d'1 Chaîne Lo/Hi
0039:	0057 - 0058	Pointeur:Finmémoiredechaînes/Var.Banc1 Lo/Hi
003B:	0059 - 0060	Numéro du moment ligne BASIC Lo/Hi
003D:	0061 - 0062	Pointeur sur texte BASIC de CHRGET/ CHRGOT Lo/Hi

003F:	0063 - 0064	Pointeur PRINT USING ou de recherche caract.Lo/Hi
0041:	0065 - 0066	Numéro de ligne de DATA en cours Lo/Hi
0043:	0067 - 0068	Pointeur adresse DATA en cours Lo/Hi
0045:	0069 - 0070	Vecteur d'INPUT Lo/Hi
0047:	0071 - 0072	Nom de la Variable BASIC en cours Lo/Hi
0049:	0073 - 0074	Pointeur adresse de la Variable en cours Lo/Hi
004B:	0075 - 0076	Pour pointeur AND, LIST et FOR NEXT
004D:	0077 - 0078	<VARTXT , OPPTR>
004F:	0079	<OPMASK>
0050:	0080 - 0081	<GRBPNT , TEMPF3 , DEFPNT>
0052:	0082 - 0084	<DSCPNT>
0055:	0085	Drapeau 'HELP': $xx=HELP, $xx=LIST
0056:	0086 - 0087	<JMPER>
0058:	0088	<OLDOV>
0059:	0089	<TEMPF1,MULTPLYDEFINEDFORINSTR>
005A:	0090 - 0091	<ARYPNT , HIGHDS>
005C:	0092 - 0093	<HIGHTR>
005E:	0094	<TEMPF2>
005F:	0095 - 0096	<DECCNT>
0061:	0097	<GRBTOP , DPTFLG>
0062:	0098	<EXPSGN >
0063:	0099	Accumulateur flottant 1: Exposant
0064:	0100 - 0103	Accumulateur flottant 1: Mantisse
0068:	0104	Accumulateur flottant 1: Signe
0069:	0105	Pointeur vers table de polynôme
006A:	0106	Accumulateur flottant 2: Exposant
006B:	0107 - 0110	Accumulateur flottant 2: Mantisse
006F:	0111	Accumulateur flottant 2: Signe
0070:	0112	Produit des signes de l'Accu 1 à l'Accu 2
0071:	0113	Accu.1 virg.flottante (arrondissement)
0072:	0114 - 0115	Pointeur: Tampon cassette
0074:	0116- 0117	ValeurdedécalagepourAUT,00=horsservice
0076:	0118	Drapeau haute résolution: Lors d'un départ BASIC est placé 10 Koctets plus haut
0077:	0122	<SPRNUM , NOZE , KEYNUM>
0078:	0120	<HULP COUNTER , KEYSIZ>
0079:	0121	<SYNTMP>
007A:	0122 - 0124	Description de la variable DS$
007D:	0125 - 0126	Fin de Pile lors du fonctionnement du Prog.
007F:	0127	Drapeau mode RUN ou Mode DIRECT
0080:	0128	Pointeur USING pour décimaux ???
0081:	0129	<PARSTX>
0082:	0130	<OLDSTX>
0083:	0131	Couleur du moment en mode graphique
0084:	0132	Mode MULTICOLOR: couleur 1
0085:	0133	Mode MULTICOLOR: couleur 2
0086:	0134	Couleur premier-plan
0087:	0135 - 0136	Facteur de l'échelle de représentation suivant l'axe des 'X'
0089:	0137 - 0138	Facteur de l'échelle de représentation suivant l'axe des 'Y'
008B:	0139	<STOPNB STOP PAINT IF NOT BACKGROUND/ NOT SAME COL>
008C:	0140 - 0141	<GRAPNT>
008E:	0142	<VTEMP1>
008F:	0143	<VTEMP2>
0090:	0144	Mot STATUS pour opérations du KERNAL Mode Entrées/Sorties
0091:	0145	Drapeau STOP: touche STOP, touche RVS
0092:	0146	Constante de temps pour Opérations DATASSETTE
0093:	0147	Drapeau LOAD: $00=LOAD, $01=VERIFY
0094:	0148	Drapeau BUS série: caractère dans le tampon
0095:	0149	Caractère dans le tampon pour le BUS série
0096:	0150	Numéro Synchr. pour DATASSETTE, réception du signal EOT
0097:	0151	Adresse temporaire de données
0098:	0152	Index pour tableau de données, Nombre de Fichiers ouverts
0099:	0153	Numéro par défaut pour Périphérique d'entrée (0 pour le Clavier)

009A:	0154	Numéro par défaut pour Périphérique de Sortie (3 pour l'Ecran)
009B:	0155	Octet de Parité pour DATASSETTE
009C:	0156	Drapeau Bande: Octet reçu
009D:	0157	Drapeau d'information pour KERNAL 009E:
	0158	Erreur DATASSETTE 1ère passe: Erreur caractère
009F:	0159	Erreur DATASSETTE 2ème passe: Correction
00A0:	0160 - 0162	Horloge 24 heures temps réel: 1/60 Seconde
00A3:	0163 - 0164	Zone de stockage temporaire pour BUS série
00A5:	0165	Compteur décrémental pour SAVE sur DATASSETTE Pointeur intermédiaire pour BUS série
00A6:	0166	Pointeur du tampon cassette
00A7:	0167	Compteur de BITs lus sur RS-232
00A8:	0168	STATUS lecture cassette Compteur entrée des BITs RS-232
00A9:	0169	Drapeau lecture Bande '0' Drapeau STARBIT RS-232
00AA:	0170	Mode 'READ' bande Tampon Octet d'Entrée sur RS-232
00AB:	0171	Compteur lecture de la bande Parité de l'Octet lu sur RS-232
00AC:	0172 - 0173	Pointeur: Déroulement de l'écran Lo:Hi
		Tampon cassette Lo/Hi
00AE:	0174 - 0175	Pointeur: Fin de Programme Lo/Hi
		Fin de cassette Lo/Hi
00B0:	0176 - 0177	Constante de temps pour DATASSETTE
00B2:	0178 - 0179	Pointeur: Début du tampon cassette Lo/Hi
00B4:	0180	Pointeur intermédiaire DATASSETTE Prochain BIT RS-232 pour déroulement
00B5:	0181	Bande signal EOT Prochain BIT RS-232 pour transmission
00B6:	0182	Pointeur intermédiaire pour DATASSETTE Tampon d'Octets RS-232
00B7:	0183	Longueur du Nom du Fichier du moment
00B8:	0184	Numéro logique du Fichier actuel (LFN)
00B9:	0185	Adresse secondaire du moment (SA)
00BA:	0186	Numéro de Périphérique du moment (GA)
00BB:	0187 - 0188	Pointeur: Adresse du nom du Fichier actuel Lo/Hi
00BD:	0189	Pointeur intermédédiaire DATASSETTE Tampon rotation parité RS-232
00BE:	0190	Nombre de blocs restants à lire/écrire
00BF:	0191	Tampon série
00C0:	0192	Tampon moteur DATASSETTE
00C1:	0193	Adresse standard Entrée/Sortie (Lo), TRACK NR.
00C2:	0194	Adresse standard Entrée/Sortie (Hi), SEKTOR NR.
00C3:	0195 - 0196	LOAD temporaire DATASSETTE Vecteur adresse du KERNAL
00C5:	0197	Espace d'Ecriture/Lecture sur bande
00C6:	0198	Numéro de Banc pour appel de LOAD, SAVE ou VERIFY
00C7:	0199	Banc où se trouve le nom du Fichier ($BB,$BC)
00C8:	0200 - 0201	Pointeur: Tampon d'Entrée RS-232
00CA:	0202 - 0203	Pointeur: Tampon de Sortie RS-232
00CC:	0204 - 0205	Pointeur: Tableau de décodage clavier
00CE:	0206 - 0207	Pointeur pour position dans la chaîne lors de la routine 'PRIMM' du KERNAL
00D0:	0208	Index du tampon Clavier
00D1:	0209	Drapeau pour Touche de Fonction lors de l'appel
00D2:	0210	Index de la Touche de Fonction lors de l'appel
00D3:	0211	Drapeau SHIFT: SHIFT=$01,C<=$02,CTRL=$04,ALT=$08
00D4:	0212	Drapeau pour une touche enfoncée
00D5:	0213	Drapeau pour la touche enfoncée du moment CHR$(0)=aucune

00D6:	0214	Drapeau pour INPUT ou GET (touche de saisie)
00D7:	0215	Drapeau mode 40 / 80 caractères
00D8:	0216	drapeau mode Texte/Graphique
00D9:	0217	<CHARENRAM-ROMVICCHARACTER FETCH FLAG BIT- 2>
00DA:	0218	Pointeur pour MOVLIN (Lo), <KEYSIZ , BITMASK >
00DB:	0219	Pointeur pour MOVLIN (Hi), <KEYLEN , SAVER>
00DC:	0220	<SEDEAL , KEYNUM>
00DD:	0221	<KEYNXT>
00DE:	0222	Banc lors de l'appel de la routine <SEDT1>
00DF:	0223	<SEDT2>
00E0:	0224 - 0225	Pointeur de la ligne-écran en cours: TEXT RAM
00E2:	0226 - 0227	Pointeur de la ligne-écran en cours: ATTRIBUT RAM
00E4:	0228	Limite basse de la Fenêtre
00E5:	0229	Limite haute de la Fenêtre
00E6:	0230	Limite gauche de la Fenêtre
00E7:	0231	Limite droite de la Fenêtre
00E8:	0232	Index de la ligne de la saisie en cours
00E9:	0233	Index de la colonne de la saisie en cours
00EA:	0234	Fin de la ligne de saisie en cours
00EB:	0235	Position du moment du curseur: ligne
00EC:	0236	Position du moment du curseur: colonne
00ED:	0237	Nombre maximun de lignes à l'écran
00EE:	0238	Nombre maximum de colonnes à l'écran
00EF:	0239	Tampon intermédiaire pour caractère à éditer
00F0:	0240	Tampon:Caractère saisie précédemment (test ESC)
00F1:	0241	Code couleur du moment pour édition caractère
00F2:	0242	Répétition du code couleur pour les routines INSERT et DELETE
00F3:	0243	Drapeau: Mode 'RVS' activé
00F4:	0244	Drapeau: Mode guillemets activé (QUOTE)
00F5:	0245	Drapeau: Mode d'insertion activé (INSERT)
00F6:	0246	Drapeau: Mode d'insertion automatique activé (AUTO-INSERT)
00F7:	0247	Verrouillage:C-SHIFT($80)etCTRL-S($40)
00F8:	0248	Verrouillage du défilement de l'écran
00F9:	0249	Verrouillage de CTRL G (émission d'un BEEP)
00FA:	0250 - 254	???
00FF:	0255	<LOFBUF>

SUPER, LE SUPER COMPILATEUR 128

S'inspirant de la conception du compilateur **BASIC 64** qui a fait ses preuves maintenant, Thomas Helbig présente aujourd'hui **BASIC 128** un super compilateur pour le COMMODORE 128 (C 128). Ce compilateur est adapté à la nouvelle version BASIC 7.0 du C 128 et réalise de nombreuses optimisations : BASIC 128 permet d'une part d'effectuer des optimisations au cours de la compilation, mais aussi au cours de l'exécution du programme compilé. A l'image du logiciel BASIC 64, BASIC 128 exploite tout l'espace mémoire disponible du C128, y compris les bancs externes. La vitesse de compilation a été accrue, et l'est encore davantage en utilisant le lecteur de disquette 1570 en mode FAST. Le générateur de codes, complètement remanié, produit maintenant des pseudo-codes encore plus courts et des codes machine encore plus condensés. En outre, les concepteurs ont ajouté de nouvelles routines arithmétiques, et le calcul de fonctions transcendantes s'en voit accéléré.
D'une grande flexibilité et d'une utilisation simple, BASIC 128 étonne tout de suite ses utilisateurs. La commande de la compilation en est un très bon exemple. Nous avons testé BASIC 128 sur des configurations très variées du C 128 : C 128, Floppy 1541, Floppy 1570, Floppy 1571, version 40 et 80 colonnes. Le compilateur est "compatible" avec toutes les configurations imaginables.

BASIC 128 permet de réaliser des applications que l'interpréteur du C 128 interdit par sa limitation au niveau de la programmation. A la différence de l'interpréteur, une meilleure exploitation de l'espace mémoire de BASIC 128 permet d'utiliser plus de données et d'augmenter la taille des programmes. La compilation des programmes accélère nettement leurs temps d'exécution, mais nous y reviendrons plus tard.

BASIC 128 compile les programmes soit en pseudo-codes soit directement en langage machine. Il est même possible de mélanger ces deux formes de compilation. En plus, il est facile d'emploi : après le chargement du programme, un menu avec quatre options s'affiche sur l'écran. En appuyant sur la touche "1" ou sur la touche "RETURN" on exécute la routine de compilation qui de-

mande d'abord le nom du programme à compiler. En fin de compilation, le programme objet est enregistré sur disquette et est prêt à être exécuté. Voilà, c'est tout ce que vous avez à faire si vous voulez compiler un programme sans vous préoccuper des paramètres.

Les autres points du menu permettent de sélectionner de nombreuses options ainsi que de choisir différents paramètres utilisés lors de la compilation.

POINT 4 = overlay : il est nécessaire lors de la compilation d'un programme utilisant la technique de l'OVERLAY. Cette technique consiste à charger un autre programme en cours d'exécution d'un programme en mémoire. Le code du programme appelé est chargé en mémoire à la place du précédent, ce qui évite l'utilisation de place mémoire supplémentaire. Plusieurs programmes se partagent ainsi la même place mémoire, et chaque programme a accès aux variables des autres.

POINT 3 = Advanced Development Package : sous ce terme se cache un menu important lors de la compilation. En effet c'est par ce menu que vous pourrez avoir accès aux différents paramètres concernant la compilation. Une fois modifiés, ces paramètres pourront être stockés sur disquette.

POINT 2 = Load Environment : il sert à l'appel des paramètres enregistrés.

Mais regardons maintenant le POINT 3 plus en détail :

- **A :** choix de la génération de pseudo-codes ou de codes machine.

- **B, C :** ils permettent la sauvegarde et le chargement de tables de symboles (liste des variables et de leurs adresses), importantes lors de la compilation de programme utilisant la technique de l'OVERLAY ou la combinaison de programmes en BASIC et en assembleur.

- **D :** la mise au point de programmes compilés nécessite l'usage de listes d'adresses.

- **E :** permet la commande du partage de la mémoire.
L'interprèteur enregistre dans le banc 0 le texte du programme et dans le banc 1 les variables. Le BASIC 128 permet d'enregistrer des variables dans le banc 0 et donc une meilleure utilisation de la place mémoire disponible. Ainsi, vous pouvez même utiliser le dernier octet de la mémoire.

- **F :** Si vous avez un écran 40 colonnes, le compilateur travaille en mode SLOW afin de permettre l'affichage à l'écran. En mode FAST, l'affichage est supprimé et le BASIC 128 compile plus rapidement les programmes. Le compilateur fait d'ailleurs tout seul la différence entre un écran 40 colonnes et un 80 colonnes.

- **G** : permet la suppression de l'adjonction automatique du module runtime lors de l'utilisation de programmes utilisant la technique de l'OVERLAY.

- **H** : extensions BASIC. Nous nous référons ici au manuel.

- **I** : permet l'enregistrement et le chargement d'instructions de compilations.

- **J** : le compilateur a besoin d'informations pour enchaîner les commandes et les lignes. Ce point permet leurs saisies et choix.

- **K** : permet la mise en place de la commande TRAP à l'intérieur d'un programme.

- **L** : Lors de la compilation d'un programme utilisant la technique de l'OVERLAY, le compilateur fait une deuxième passe de compilation. Ici, vous pouvez lui dire s'il s'agit d'un tel programme ou non.

- **M, N** : servent à l'échange avec l'unité de disquettes sans quitter le compilateur. Ceci est très pratique si vous voulez par exemple consulter le répertoire pour vérifier l'écriture du nom d'un programme juste avant la compilation.

- **O** : choisit le degré d'optimisation sous lequel le programme sera compilé. Les opérations arithmétiques sont beaucoup plus rapides avec des entiers qu'avec des nombres à virgule flottante. Le **degré I** d'optimisation interprète les variables telles qu'elles ont été définies dans le programme. **Le degré II** traite toutes les variables en tant que variables entières. Nous attirons ici l'attention du lecteur sur la possibilité d'imbriquer des instructions de compilation à l'intérieur d'un programme afin que celles-ci ne soient valables que pour une partie du logiciel.

IMBRICATION D'INSTRUCTIONS DE COMPILATION

Les instructions de compilation sont introduites sous formes de lignes REM de sorte que l'interpréteur les ignore et qu'il n'affiche pas d'erreurs de syntaxe. Elles sont par exemple indispensables dès que certaines parties du programme doivent être compilées en langage machine, d'autres en pseudo-codes.

Quelques possibilités supplémentaires du compilateur, moins évidentes à première vue peut-être, méritent une petite explication, car l'utilisateur s'étonnera certainement de la vitesse d'exécution des programmes compilés.

En premier lieu, il faut noter la nette accélération du calcul des fonctions transcendantes, comme SQR, TAN, ATN, EXP, LOG, SIN et COS. En plus, les échanges entre les différents bancs de la mémoire ne ralentissent que très peu le temps d'exécution d'un programme compilé.

OPTIMISATION DU TEMPS D'EXECUTION

Le compilateur ne peut pas faire certaines optimisations au niveau du temps lors de la compilation. L'optimisation du temps d'exécution a été réalisée au niveau des opérations arithmétiques et le choix de cette option permet de diviser leur temps d'exécution par deux.

Pour rendre la vitesse d'exécution du compilateur BASIC 128 plus parlante, nous avons compilé le programme de tri publié dans Micro Info No2 et avons chronométré les temps.

L'interpréteur du C 128 met pour le tri de 1 000 données 6 minutes et 6 secondes en mode SLOW, 2 minutes et 59 secondes en mode FAST. En utilisant toutes les possibilités du compilateur BASIC 128, nous avons mesuré 4 secondes, nous répétons QUATRE secondes pour la même application. Le temps d'exécution est donc divisé par 90 par rapport au mode SLOW et par 45 par rapport au mode FAST. Le compilateur BASIC 128 met deux fois moins de temps que le compilateur BASIC 64.

BASIC 128 : UN ARTISTE PEINTRE

Arrivés à la fin de ce test, il nous fallait encore vérifier la compatibilité du compilateur avec les instructions graphiques du BASIC 7.0. Il a parfaitement compilé plusieurs programmes extraits du livre Trucs et Astuces du C128 , encore une fois avec des configurations variées du C 128.

Par exemple le plotteur de fonctions devenait très rapide sans que nous ayons utilisé toutes les possibilités dont dispose le compilateur BASIC 128.

BASIC 128 enrichit donc véritablement la gamme de logiciels pour le C 128. Developpé à partir du BASIC 64, ce compilateur a hérité de tous les avantages de ce dernier et les dépasse largement. D'une conception très flexible, les deux compilateurs pourront donc prendre en compte des développements futurs. La structure modulaire des compilateurs a permis d'écrire en peu de temps un super programme pour le C 128. Ainsi, nous ne pouvons apprécier qu'aujourd'hui la prévoyance intervenue lors du développement du BASIC 64.
Néanmoins, le seuil "magique" des 64K n'a pas été franchi par une gymnastique fastidieuse et donc dangereuse, mais par une prise en compte de toutes les possibilités qu'offre le C 128.

MAIS POURQUOI UTILISER UN COMPILATEUR ?

Un ordinateur ne comprend que des commandes en langage machine, le langage de son microprocesseur. Une commande est enregistrée dans l'ordinateur sous forme de chaines de bits, une suite de 0 à 1. Mais le langage machine est peu orienté vers les besoins de l'homme, et, en général, l'utilisateur développe ses programmes dans un langage évolué. Par ailleurs, un microprocesseur ne peut exécuter que des instructions élémentaires. Ainsi, une petite ins-

PEEKS & POKES

TABLEAU DES ADRESSES D'IMPLANTATION DES CPC 664 ET 6128

Ce tableau reproduit les fonctions de toutes les cases mémoires. Naturellement certaines fonctions peuvent avoir des effets secondaires imprévus, c'est pourquoi les PEEK et les POKE dans certaines zones mémoires sont à employer avec circonspection. Que cela ne vous empêche pas d'expérimenter, mais n'oubliez pas de sauvegarder auparavant d'éventuels programmes.

AE15-AE16	adresse pour READ
AE17-AE18	adresse pour DATA
AE19-AE1A	adresse du pointeur de la pile-BASIC
AE1B-AE1C	adresse de l'instruction en cours
AE1D-AE1E	adresse de la ligne programme en cours
AE1F	flag pour TRACE
AE26-AE27	adresse de départ pour LOAD
AE28	flag pour CHAIN MERGE
AE29	type du fichier
AE2A-AE2B	longueur du fichier
AE2C	flag de protection de programme
AE2D-AE5B	zone de travail pour conversion en ASCII
AE55-AE56	adresse de CALL
AE57	mode bankswitching pour CALL
AE58-AE59	registre HL pour CALL
AE5A-AE5B	registre SP pour CALL
AE5C	grandeur de la tabulation
AE5E-AE5F	pointeur HIMEM
AE60-AE61	pointeur sur fin de la RAM disponible
AE62-AE63	pointeur sur début de la RAM disponible

0000-3FFF	ROM contenant le système d'exploitation
0044-0590	noyau du système. Kernel
0591-08BC	machine Pack
08BD-0ABE	restore les vecteurs
0ABF-1073	routines d'écran
1074-15A7	gestion textes et fenêtres
15A8-1B5B	gestion fenêtre graphique
1B5C-1FE8	gestion du clavier
1FE9-24BB	gestion des sons
2C02-2F72	éditeur d'écran
2F73-37FF	routines d'arithmétique a virgule flottante
0000-003F	copie de la ROM pour le bankswitching
0040-013F	tampon clavier et zone de travail
0170-A67B	Mémoire pour programme
3800-3FFF	ROM contenant la police de caractères
AC01	flag pour AUTO
AC02-AC03	numéro de ligne AUTO
AC04-AC05	largeur du pas pour AUTO
AC06	dernier numéro de 'stream'
AC07	canal d'entrée

AE66-AE67	pointeur sur fin du programme BASIC
AE68-AE69	pointeur sur début des variables
AE6A-AE6B	pointeur sur début des tableaux
AE6C-AE6D	pointeur sur fin des tableaux
AE70-B06E	pile pour BASIC (FOR, GOSUB etc.)
B06F-B070	pointeur de pile BASIC
B071-B072	pointeur sur début des strings
B073-B074	pointeur sur fin des strings
B09F	type de la variable
B100-B103	nombre aléatoire
B113	flag pour DEG/RAD
B115	flag pour l'insertion
B118-B1E9	divers registres pour la disquette
B11F	nom du fichier INPUT
B164	nom du fichier OUTPUT
B1ED-B396	divers registres pour le son
B628-B692	divers registres pour la lecture du clavier
B693-B6A7	divers registres pour le graphisme
B693-B694	ORIGIN-X
B695-B696	ORIGIN-Y
B6B5	fenêtre actuelle
B6B6-B725	paramètres pour les fenêtres
B726-B727	position du curseur (ligne,colonne)
B728-B72C	divers registres pour les fenêtres
B72D-B7C7	divers registres pour l'écran
B7C3	mode de l'écran actuel
B7C6	adresse de la mémoire écran
B7C7-B7D1	utilisations diverses
B7D2-B7D3	vitesse de clignotement
B7D4-B7F9	divers registres pour les couleurs changeantes
B82D-B8D9	zone de travail du système d'exploitation
C000-FFFF	RAM vidéo
C000-FFFF	ROM interpréteur
C000-FFFF	ROM d'extension

AC09	WIDTH
AC0A	dernière position sur disquette
AC0C	flag pour FOR-NEXT
AC0D-AC0E	mémoire auxiliaire pour FOR-NEXT
AC12-AC13	adresse pour NEXT
AC14-AC15	adresse pour WEND
AC1A-AC1B	adresse pour ON-BREAK
AC1E-AC29	séquence de son 0
AC2A-AC35	séquence de son 1
AC36-AC41	séquence de son 2
AC42-AC53	domaine pour interruption 0
AC54-AC65	domaine pour interruption 1
AC66-AC77	domaine pour interruption 2
AC78-AC89	domaine pour interruption 3
AC8A-AD89	mémoire tampon
AD8C-AD8D	adresse des ERRORs
AD8E-AD8F	pointeur BASIC après ERROR
AD90	numéro de l'erreur
AD92-AD93	pointeur BASIC après une interruption
AD94-AD95	adresse de ligne interrompue
AD96-AD97	adresse pour ON-ERROR
AD98	flag : ON-ERROR est actif
AD99	état du canal
AD9A	ENT
AD9B	ENV
AD9C-AD9D	Période
AD9E	Période du bruit blanc
AD9F	volume
ADA0-ADA1	longueur
ADA2-ADA3	ENV et ENT
ADEB-ADEC	tableau FN
ADED-ADF2	tableau 'array'
ADF3-AE0C	type des variables prédéfinies A-Z
AE14	caractère de séparation pour INPUT

MICRO APPLICATION

MICRO APPLICATION 13, rue Sainte Cécile 75 009 PARIS Tél.: (1) 47-70-32-44

les plus de Micro Application

DONNEZ UN LOOK DE 32 BITS A VOTRE CPC

POUR AMSTRAD

SUPERPAINT
Vous connaissez certainement les programmes appelés "PAINT" qui étaient jusqu'à maintenant réservés aux micros 32 bits? Rapides, spectaculaires, ils permettent de réaliser de fantastiques graphismes très facilement et en un temps record. Aujourd'hui avec **SUPERPAINT** vous pouvez vous aussi sur votre CPC accéder et profiter des fantastiques possibilités d'un véritable logiciel **"PAINT".**

Les Fonctions Principales de SUPERPAINT
Tracé de rectangles, lignes, cercles, ellipses...
Arrondissage des angles (très important).
Toutes les formes peuvent être coloriées ou tramées.
Quatre grosseurs de traits disponibles.
Sélection par menu déroulants et icônes (type 32 bits).

Le Dessin à main levée.
Différentes formes de pinceaux disponibles.
Constitution de brosses.
Tracé au crayon.
Peinture à l'aérographe (ou bombe) en couleur ou en trame.
Fonctions ciseaux et colle pour réaliser des montages et découpages.

Fonctions supplémentaires
24 trames présélectionnées.
Création de vos propres trames et d'une palette personnalisée.
Sauvegarde sur disque de ces éléments.
Travail en format A4 (21 x 29,7) grandeur réelle.
Scrolling pour visualisation totale.
Repérage d'une aire par fonction LASSO pour entourage, copie, déplacement, remplissage, effet miroir, rotation horizontale et verticale et manipulations diverses.

Fonctions de Dessin Assisté
Fonction ZOOM intégral d'une partie d'un dessin.
Correction points par points en mode ZOOM.
Modification des couleurs instantanée.
Vitesse du Joystick réglable pour plus de confort.
Catalogue disquette accessible en permanence.
Mémorisation et rappel de la dernière fonction (très pratique).
Travail sur plusieurs écrans avec transferts d'images.
Impression complète du document sur imprimantes AMSTRAD et EPSON.

EDITEUR DE TEXTE INTEGRE
Intégration de texte dans le graphique (explications, messages...).
Plusieurs jeux de caractères disponibles.
3 tailles de caractères.
Ecriture en gras, italique, souligné...
Mixage des caractères.

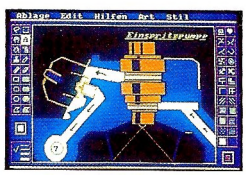
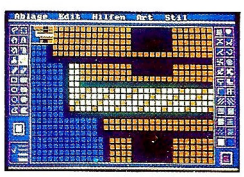
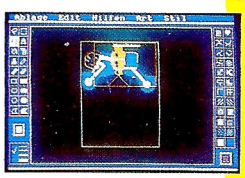
Commandes
Utilisation du joystick comme d'une souris.
Déplacement des fenêtres ultra rapide sans altération de l'écran de travail.



SUPERPAINT est livré sur disquette 3 pouces accompagné d'une documentation en français.
SUPERPAINT fonctionne sur CPC 464, 664 et 6128.

Prix sur disquette 395 TTC
Réf. AM 309



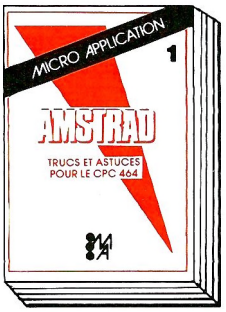
395 F

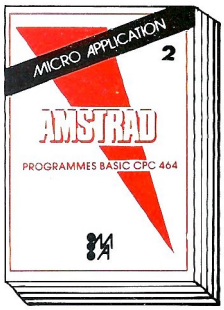
- **Les livres**
- **Les logiciels de programmation**
- **Trois super plus pour CPC**

les livres plus de Micro Application

TRUCS ET ASTUCES POUR L'AMSTRAD CPC (Tome 1)

C'est le livre que tout utilisateur d'un CPC doit posséder. De nombreux domaines sont couverts (graphismes, fenêtres, langage machine) et des super programmes sont inclus dans ce best-seller (gestion de fichiers, éditeur de textes et de sons...).
Réf. : ML 112
Prix : **149** FF





PROGRAMMES BASIC POUR LES CPC 464

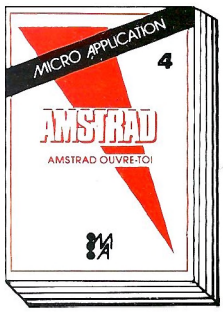
ALIMENTEZ VOTRE CPC 464.
Ce livre contient de supers programmes, notamment un désassembleur, un éditeur graphique, un éditeur de texte... Tous les programmes sont prêts à être tapés et abondamment commentés.
Réf. : ML 118
Prix : **129** FF

LE BASIC AU BOUT DES DOIGTS CPC 464

Ce livre est une introduction complète et didactique au BASIC du micro-ordinateur AMSTRAD CPC 464. Il permet d'apprendre rapidement et facilement la programmation (instructions BASIC, analyses des problèmes, algorithmes complexes...).
Principaux thèmes abordés :
- Les bases de la programmation
- Bit, Octet, ASCII
- Instructions du BASIC
- Organigrammes
- Les fenêtres
- Programmes BASIC plus poussés
- Le programme et menus.



Comprenant de nombreux exemples, ce livre vous assure un apprentissage simple et efficace du BASIC CPC 464.
Réf. : ML 119
Prix : **149** FF



AMSTRAD OUVRE-TOI

Le bon départ avec le CPC 464 ! Ce livre vous apporte les principales informations sur l'utilisation, les possibilités de connexions du CPC 464 et les rudiments nécessaires pour développer vos propres programmes. C'est le livre idéal de tous ceux qui veulent pénétrer dans l'univers des micro-ordinateurs avec le CPC 464.
Réf. : ML 120
Prix : **99** FF

JEUX D'AVENTURES. COMMENT LES PROGRAMMER

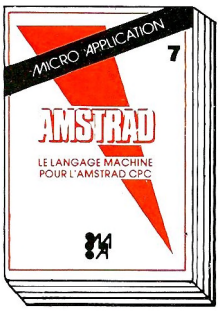
Voici la clé du monde de l'aventure. Ce livre fournit un système d'aventures complet, avec éditeur, interpréteur, routines utilitaires et fichiers de jeux. Ainsi qu'un générateur d'aventures pour programmer vous-mêmes facilement vos jeux d'aventures. Avec, bien sûr, des programmes tout prêts à être tapés.
Réf. : ML 121
Prix : **129** FF



LA BIBLE DU PROGRAMMEUR DE L'AMSTRAD CPC 464 (Tome 6)

Tout, absolument tout sur le CPC 464. Ce livre est l'ouvrage de référence pour tous ceux qui veulent programmer en pro leur CPC. Organisation de la mémoire, le contrôleur vidéo, les interfaces, l'interpréteur et toute la ROM DESASSEMBLEE et COMMENTEE sont quelques-uns des thèmes de cet ouvrage de 700 pages.
Réf. : ML 122
Prix : **249** FF



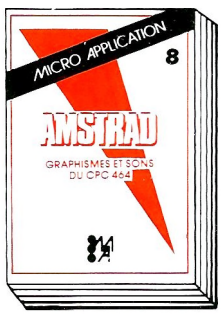


LE LANGAGE MACHINE DE L'AMSTRAD CPC (Tome 7)

Ce livre est destiné à tous ceux qui désirent aller plus loin que le BASIC. Des bases de la programmation en assembleur à l'utilisation des routines système, tout est expliqué avec de nombreux exemples. Contient un programme assembleur, moniteur et désassembleur.
Réf. : ML 123
Prix : **129** FF

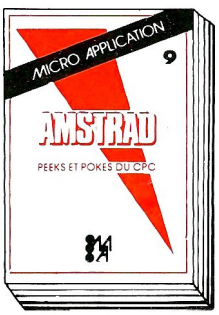
GRAPHISMES ET SONS DU CPC

L'AMSTRAD CPC dispose de capacités graphiques et sonores exceptionnelles. Ce livre en montre l'utilisation à l'aide de nombreux programmes utilitaires.
Contenu :
- base de programmation graphique
- éditeur de police de caractères
- "sprites", "shapes" et chaînes
- représentations multicouleurs
- calcul des coordonnées
- rotations, mouvements
- représentations graphiques de fonctions en 3D
- D.A.O. (dessin assisté par ordinateur)



- synthétiseur
- mini-orgue
- enveloppes de son, et beaucoup d'autres choses...

Réf. : ML 124
Prix : **129** FF

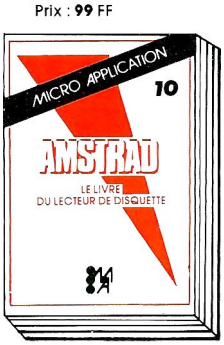


PEEKS ET POKES DU CPC (Tome 9)

Comment exploiter à fond son CPC à partir du BASIC? C'est ce que vous révèle ce livre avec tout ce qu'il faut savoir sur les peeks, pokes et autres call... Vous saurez aussi comment protéger la mémoire, calculer en binaire... et tout cela très facilement. Un passage assuré et sans douleur du BASIC au puissant LANGAGE MACHINE.
Réf. : ML 126
Prix : **99** FF

LIVRE DU LECTEUR DE DISQUETTE AMSTRAD CPC (Tome 10)

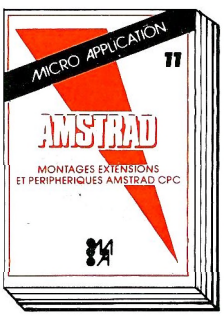
Tout sur la programmation et la gestion des données avec le floppy DDI-1 et le 664! Utile au débutant comme au programmeur en langage machine. Contient le listing du DOS commenté, un utilitaire qui ajoute les fichiers RELATIFS à l'AMDOS avec de nouvelles commandes BASIC, un MONITEUR disque et beaucoup d'autres programmes et astuces... Ce livre est indispensable à tous ceux qui utilisent un

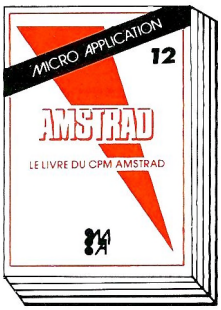


floppy ou un 664 AMSTRAD.
Réf. : ML 127
Prix : **149** FF

MONTAGES, EXTENSIONS ET PERIPHERIQUES AMSTRAD CPC (Tome 11)

Pour tous les amateurs d'électronique, ce livre montre ce que l'on peut réaliser avec un CPC. De nombreux schémas et exemples illustrent les thèmes et applications abordés comme les interfaces, programmateur d'EPROM... Un très beau livre de 450 pages.
Réf. : ML 131
Prix : **199** FF



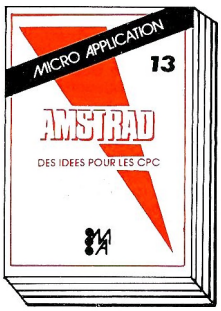


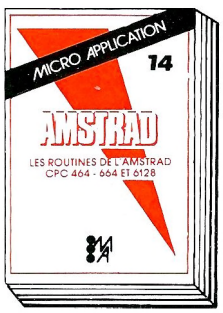
LE LIVRE DU CP/M AMSTRAD (Tome 12)

Ce livre vous permettra d'utiliser CP/M sur les CPC 464, 664 et 6128 sans aucune difficulté. Vous y trouverez de nombreuses explications et les différents exemples vous assureront une maîtrise parfaite de ce très puissant système d'exploitation qu'est CP/M.
Réf. : ML 128
Prix : **149** FF

DES IDÉES POUR LES CPC (Tome 13)

Vous n'avez pas d'idées pour utiliser votre CPC (464, 664, 6128)? Ce livre va vous en donner! Vous trouverez de très nombreux programmes BASIC couvrant des sujets très variés qui transformeront votre CPC en un bon petit génie. De plus les programmes vous permettront d'approfondir vos connaissances en programmation. (250 pages).
Réf. : ML 132
Prix : **129** FF





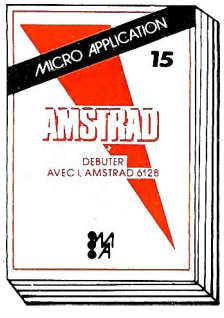
Les routines utiles de l'AMSTRAD CPC (Tome 14)

Pour bien connaître et utiliser les routines utiles de l'AMSTRAD 6128, 664, 464. A la portée de tous. Nombreux programmes utilitaires, exemples, désassembleur, etc.

Réf. : ML 143
Prix : **149** FF

Débuter avec le CPC 6128 (tome 15)

Ce livre s'adresse à ceux qui débutent avec le CPC 6128. Tout leur est clairement expliqué aussi bien pour le matériel que pour le logiciel. Une fois leur machine bien en main, ils pourront s'attaquer au Basic et utiliser l'utilitaire de gestion d'adresses que contient le livre.
Réf. : ML 145
Prix : **99** F TTC



Disquettes des livres AMSTRAD

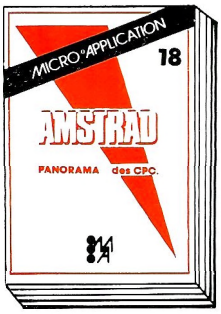
Si vous n'avez pas le temps de taper les programmes des livres, sachez que M.A. les a déjà tapés pour vous :

AMSTRAD	Disquette	
Livre n° 1 :	120 FF	MICRO-INFO n° 1 : 15 FF
Livre n° 7 :	120 FF	MICRO-INFO n° 2 : 20 FF
Livre n° 10 :	120 FF	MICRO INFO n° 3 : 20 FF

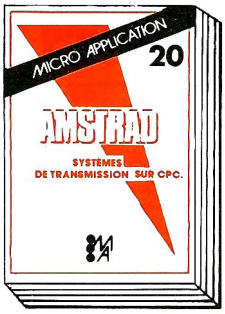
Des plus pour votre CPC



La bible des CPC 664/6128 (tome 16)
Un régal pour tous ceux qui veulent tout connaître sur les CPC 6128 et 664. Analyse du système d'exploitation, du processeur, le GATE ARRAY, le contrôleur vidéo, le 8255, le chip sonore, les interfaces... Comprend un désassembleur, les points d'entrée des routines commentés de l'interpréteur et du système d'exploitation. Un super livre comme toutes les Bibles !
Réf. : ML 146 Prix : **199** F

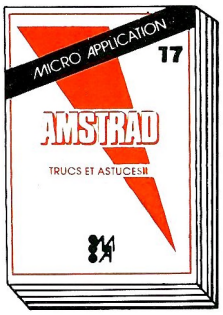


TOME 18 : PANORAMA des CPC.
Vous trouverez dans ce livre tout ce qu'il faut savoir sur les CPC. Accessible à tous cet ouvrage est un véritable guide de référence des AMSTRAD CPC et se révèlera très pratique à l'utilisation.
Réf : ml 148
Prix : **129** FF



TOME 20 : SYSTÈMES DE TRANSMISSION SUR CPC.
Encore une exclusivité Micro Application. Grâce à ce livre les communications et transmissions n'auront plus de secrets pour vous et vous pourrez profiter au maximum des possibilités offertes aujourd'hui dans ce domaine. Complet avec beaucoup d'applications pratiques, un ouvrage pratique et original.
Réf : ml 151
Prix : **199** F

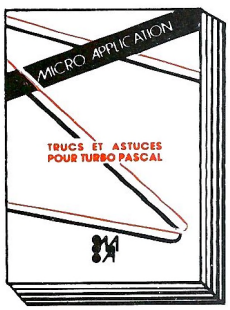
TRUCS ET ASTUCES II POUR CPC (tome 17)
Ce livre concerne tous les possesseurs de CPC 464, 664 et bien sûr 6128 !). Vous y trouverez un générateur de menus, un générateur de masques, des aides à la programmation comme un DUMP, l'utilisation des routines systèmes et plein d'astuces de programmation. Pour tous ceux qui veulent tirer le maximum de leur CPC !
Réf. : ML 147
Prix : **129** F TTC



TOME 19 : PROGRAMMES et APPLICATIONS ÉDUCATIFS sur CPC.
Ce livre est un recueil complet de programmes complets et d'applications prêts à fonctionner sur CPC. Chaque programme est très bien commenté et l'ouvrage couvre de nombreux sujets (mathématiques, chimie...). Ce livre est tout particulièrement destiné aux lycéens.
Réf : ml 150
Prix : **179** FF



TRUCS ET ASTUCES POUR TURBO PASCAL
Plus de 250 pages.
Ce livre est un recueil de trucs et d'astuces pour utiliser au mieux Turbo Pascal. Vous trouverez en outre les nombreux conseils et méthodes des programmes exemples et utilitaires comme des procédures de tri, de gestion d'écran... Un ouvrage très complet indispensable à tout utilisateur de Turbo Pascal.
Réf. : ML 133
Prix : **149** FF



LES LOGICIELS MICRO APPLICATION POUR AMSTRAD CPC : PLUS DE PERFORMANCE POUR MOINS D'ARGENT.
Les Logiciels Micro Application pour les AMSTRAD CPC ont tous été créés spécialement non seulement pour les AMSTRAD CPC mais aussi pour les utilisateurs d'AMSTRAD CPC ! Ce sont des programmes modernes et fiables qui utilisent les techniques de programmation les plus avancées et qui profitent au maximum des fantastiques capacités des CPC. En choisissant un logiciel Micro Application vous êtes certain d'acquérir un produit de haute qualité à un juste prix.

D.A.M.S. POUR AMSTRAD CPC 464 & 664

- Editeur pleine page avec scrowling/haut et bas
- Assembleur Turbo ultra-rapide
- Moniteur de langage machine complet
- Désassembleur avec génération de listing-source (Label et DEFB).

D.A.M.S. est un logiciel intégrant un assembleur, un moniteur et un désassembleur symbolique pour développer et mettre au point facilement des programmes en langage machine sur les micro-ordinateurs AMSTRAD. Les trois modules sont co-résidents en mémoire ce qui assure une grande souplesse d'utilisation. Vous pouvez notamment utiliser un éditeur plein écran, un assembleur immédiat, un désassembleur symbolique, une trace et beaucoup d'autres fonctions très puissantes. D.A.M.S. est entièrement relogeable et est bien évidemment écrit en langage machine.
Ref. : AM 208
Prix : sur cassette : **295** FF TTC pour CPC 464
Ref. : AM 308
Prix sur disquette : **395** F TTC pour CPC 664 & CPC 464.

WATSON AMSTRAD AUTOFORMATION A L'ASSEMBLEUR EN FRANÇAIS

Contient un livre et un logiciel.
LE LIVRE :
Cet ouvrage introduit le débutant à la programmation du Z80 grâce à la méthode du Dr WATSON qui selon les critiques vaut son pesant d'or ! Aucune connaissance préalable n'est requise et le but du livre est d'assurer au novice un succès total. A la fin du livre les instructions du Z80 sont expliquées en détail. De nombreux exemples illustrent les différentes étapes du cours alors que des exercices (les solutions sont fournies) testent la compréhension.
LE LOGICIEL : Un assembleur Z80 complet est livré sur cassette et comprend :
- Etiquettes Symboliques
- Directives d'Assemblage
- Chargement / Sauvegarde
- Copie Ecran
- INSERT / DELET.

L'assembleur permet d'écrire des programmes facilement en langage d'assemblage puis les transforme en code machine (langage machine). Pour vous aider à comprendre les rotations mathématiques utilisées, une démonstration de l'utilisation des nombres binaires et hexadécimaux est fournie. Un programme utilisant les commandes graphiques additionnelles décrites dans le livre est également fourni.
Réf. : ML 126
Prix : **195** FF K7 - **295** FF disquette.

SPACE MOVING

SPACE MOVING est un utilitaire sophistiqué pour la création et l'animation de figures dans l'espace. Il est composé de deux modules :

- **SPACE :** un éditeur professionnel, qui autorise de nombreuses manipulations comme, la création de trois types de fichiers, la recopie graphique d'écran sur imprimante matricielle, le listing des points constituant une figure, toutes sortes de commandes d'animation de cette figure, l'enregistrement et la conservation de ces mouvements...
- **SPACEB :** programme "condensé" qui permet l'exploitation des figures créées, pouvant être exploitées à partir d'un programme BASIC.

Chacun de ces modules est écrit en langage machine, ce qui leur confère efficacité et rapidité. Par ailleurs, vous disposez d'un programme de démonstration et de plusieurs fichiers déjà créés. Documentation et logiciel en français.
Prix : version K7 : **295** F TTC Réf : AM 210
version DISK : **395** F TTC Réf : AM 310

Avez-vous lu Micro-Info le magazine de MICRO APPLICATION ?



Trois Super plus pour CPC

Gestion de fichiers 1

Datamat

DATAMAT est une gestion de fichier efficace et performante. DATAMAT vous permettra de tenir à jour et d'exploiter tous vos fichiers (clients, club, collections, adresses, articles...) et vous serez étonné des possibilités de recherche de ce logiciel. De plus relié à CALCUMAT vous pourrez reprendre les données de vos fichiers pour établir des calculs et des graphes (par exemple : répartition géographique de vos clients, histogramme des ventes...). Relié à TEXTOMAT vous pourrez intégrer vos données pour réaliser des mailings, courrier personnalisé etc...

- Emploi extrêmement simple grâce à l'utilisation de menus
- Traite tout type de données
- Définition d'un masque de saisie personnalisé
- 40 ou 80 caractères par ligne
- Fonction de recopie d'écran sur imprimante
- 50 champs par enregistrement
- 512 caractères par enregistrement
- Jusqu'à 4000 enregistrements par fichier
- Définition des couleurs écran-bordure-caractère
- Utilisation des fichiers avec TEXTOMAT (mailing, relances...)
- Gestion complète des disquettes
- Fonctionne avec un ou deux lecteurs de disquettes
- Entièrement écrit en langage machine : extrêmement rapide
- Adaptable à tout type d'imprimante
- Jeu de caractères français accentué complet (ou anglais)
- Programme principal en mémoire : pas d'attente de chargement
- Manuel d'utilisation complet en français
- Commercialisé à plusieurs milliers d'exemplaires dans le monde entier
- Garantie et Service Micro Application

Ce que DATAMAT vous permet notamment de faire :

- Rechercher les données en quelques secondes
- Extraire les données selon un plusieurs critères
- Trier les données selon différents critères
- Imprimer les données sous un format définissable
- Impression d'étiquettes
- DATAMAT fonctionne sur CPC 464,664 et 6128 et coûte 450 Francs ttc.

Traitement de textes 2

Textomat

Un Traitement de Texte puissant et simple qui tire partie de toutes les capacités des CPC. TEXTOMAT vous permettra d'écrire, archiver et modifier vos courriers, rapports, thèses, études... Vous pourrez intégrer dans vos documents des données extraites des fichiers DATAMAT et des calculs réalisés par CALCUMAT.

- Utilisation aisée à partir de menus
- Jeu de caractères français complet accentué
- Fonction de calcul en mode texte
- Jusqu'à 16640 caractères
- Possibilité de chainage de textes sur disquette
- Fonctionne en mode 80 caractères avec accents
- Travaille avec un ou deux lecteurs de disquettes
- Choix des couleurs écran-caractères-bordure
- Mode Insertion-Gomme....
- Tabulation
- Numérotation des pages
- Impression proportionelle avec table d'espacements redéfinissable
- Caractères de contrôle librement définissable (soulignage,double épaisseur..)
- Lettre type avec insertion automatique (adresses par exemple)
- Formatage des textes à l'écran
- Adaptation à tout type d'imprimante
- Manuel détaillé et didactique
- Fonctionne sur 464, 664 et 6128. 450 Francs TTC.

Tableur 3

Calcumat

CALCUMAT : Le logiciel plus plus !! 450 Francs TTC seulement

TABLEUR + GRAPHISMES + OUTILS DE BUREAU + INTEGRATEUR

CALCUMAT est un logiciel intégrant un tableur, un logiciel de représentation graphique ,des outils de bureau (calculatrice, bloc note, ciseaux et colle) et un intégrateur. Grâce au tableur vous réaliserez tous vos calculs sous forme de tableaux. Simultanément vous disposerez d'outils de bureaux indispensables comme la calculatrice ou les ciseaux pour déplacer une partie du tableau. Vous pourrez ensuite avec le logiciel graphique représenter le résultat de vos calculs sous forme de diagrammes ! Enfin l'intégrateur vous permettra de transmettre à CALCUMAT des données de DATAMAT pour les "travailler" et ensuite de communiquer à TEXTOMAT les résultats obtenus pour éditer un rapport etc.... CALCUMAT c'est vraiment PLUS !

TABLEUR :

- Feuille de Calcul Complète de 1024 lignes sur 255 colonnes
- Possibilté de nommer les cellules
- Fonctions Trigonométriques
- Formattage des données
- Sélection des commandes par menu déroulant type MAC
- Recalcul automatique des cellules
- Largeur des colonnes définissable à tout moment
- Tri Alphanumérique des cellules
- LOGICIEL GRAPHIQUE INTEGRE
- Quatre Fenêtre à l'écran
- Graphismes en 2 et 3 Dimensions
- Histogrammes
- Courbes
- Camemberts ou Fromages
- Editeur de Texte pour commenter les diagrammes
- OUTILS DE BUREAU disponible à tout moment
- Calculatrice
- Bloc Note
- Ciseaux et colle pour découper, modifier et coller la feuille de calcul
- INTEGRATEUR
- Liaison de DATAMAT à CALCUMAT pour récupererdesdonnées selectionnées
- Liaison de CALCUMAT à TEXTOMAT pour intégrer les calculs dans des rapports
- CALCUMAT fonctionne sur CPC 464,664 et 6128 et coûte 450 Francs ttc.

Pour CPC 464,664, 6128

Les livres M.A. sont egalement distribues par les editions RADIO et presents chez votre revendeur habituel.
REVENDEURS CONTACTEZ-NOUS DES MAINTENANT.

MICRO APPLICATION
13, rue Sainte Cécile 75009 PARIS

truction comme SIN (sinus), une ligne dans un programme BASIC, nécessite une centaine d'instructions élémentaires effectuées par le microprocesseur. La division de deux nombres est un jeu d'enfants en langage évolué, la même opération avec virgule flottante exprimée en langage machine demande 15 jours de programmation. L'ordinateur a donc besoin d'un "traducteur" qui lui permet d'interpréter les programmes en langages évolués.

On différencie plusieurs formes de "traducteur" : les interpréteurs, les compilateurs et les formes mélangées.

L'INTERPRETEUR

L'interpréteur lit le programme à exécuter caractère par caractère et analyse ceux-ci. Dès qu'il trouve une commande, il exécute une routine en langage machine correspondant à la commande dans laquelle le mode d'exécution de la commande est décrit. Il avance ainsi dans le programme. Si le logiciel comporte par exemple une boucle, l'interpréteur doit analyser autant de fois les instructions que la boucle doit être exécutée, dans la mesure où il ne garde jamais les analyses précédentes. Ce processus est long et fait perdre beaucoup de temps.

LE COMPILATEUR

Un compilateur analyse d'abord le programme BASIC appelé programme "source" et le traduit en langage machine, c'est-à-dire en programme objet. Ce programme objet est donc directement exécutable par la machine et ne nécessite plus d'analyses supplémentaires lors de son exécution. Cette compilation prend évidemment du temps mais elle n'est réalisée qu'une seule fois. De plus le programme ainsi compilé garantit une plus grande rapidité d'exécution et une meilleure gestion de la place mémoire.

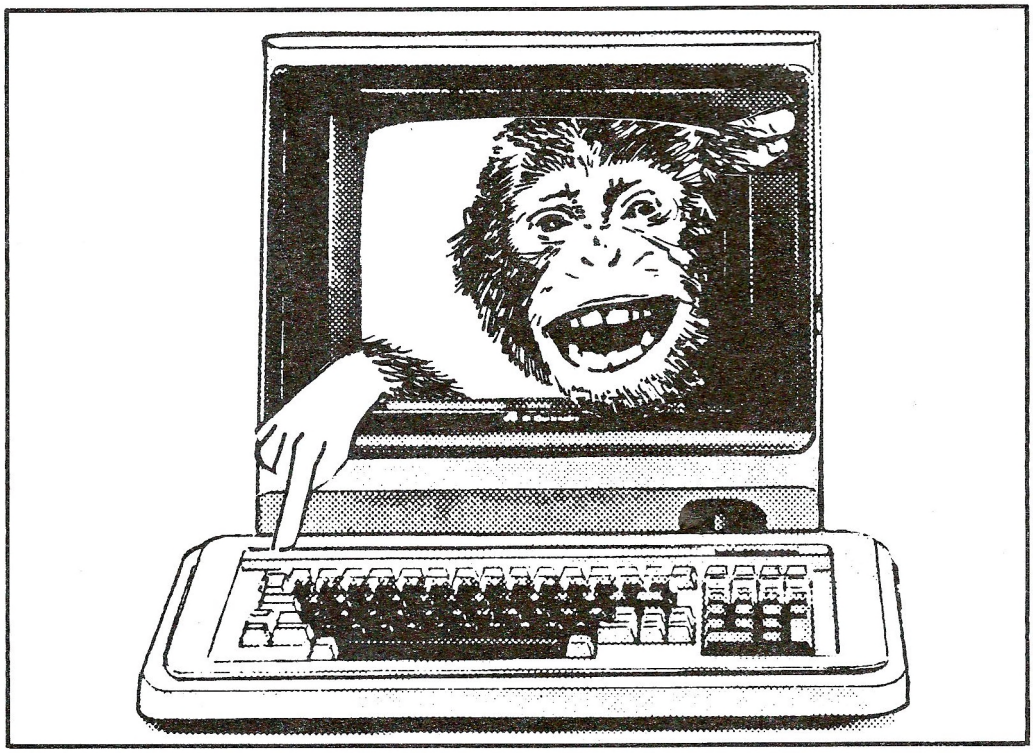
Malgré cette distinction théorique entre compilateur et interprèteur, il n'existe pas de produits purs.

Lors de la saisie, l'interprèteur BASIC analyse le programme ligne par ligne et le transforme en codes reconnus que par lui seul.

Sauvegardé sous cette forme, le programme sera, plus rapidement interprèté lors d'une exécution future. Les compilateurs, de leur côté, disposent d'un module runtime comprenant les routines les plus usuelles. Ce module est automatiquement joint à tout programme compilé. Il s'y référera dès qu'une de ces routines sera appelée.

Les programmes BASIC sont donc encodés ce qui réduit l'espace mémoire occupé. Il en va autrement des programmes compilés dont le programme source est, en général, plus court. Ceci vient d'une part de l'adjonction du module runtime, d'autre part de la non-optimisation de la génération des codes machines par le compilateur. Et la grande rapidité des programmes compilés par rapport aux programmes interprètés ne compense pas forcément ce gaspillage de place mémoire.

BASIC 128 est commercialisé par Micro Application et coûte 450 Francs; BASIC 64 coûte lui 350 Francs. Ces deux produits fonctionnent respectivement sur C64 et C128.



PC et Compatibles

CRÉEZ, gérez vos écrans, vos fenêtres, vos fichiers multiclés et vos états, simplement ! Quelques instructions suffisent.

MASTER PC Basic et Turbo Pascal

• *MASTER PC*

Grâce à MASTER PC, vous pouvez tout écrire et mettre au point dans votre langage naturel simplement, rapidement et sans aucune limite. La gestion des écrans et des fenêtres, les fichiers multiclés, la préparation des états et les impressions, MASTER s'en charge pour vous. Et vous n'avez pas à faire l'apprentissage d'un nouveau langage puisque vous parlez déjà le même que lui. Vous pouvez maintenant vous consacrer à l'analyse de votre application, étant assuré d'en obtenir des performances et une présentation optimums (fenêtre, touche de fonctions, page d'aide, pop-up menu, accès disque ultra-rapide, etc.).

• *SPÉCIFICATIONS TECHNIQUES :*

MASTER PC se présente sous la forme de 50 nouvelles instructions BASIC totalement intégrées à l'interpréteur. Un programme écrit en BASIC avec MASTER PC peut bien sûr être compilé.

• *GESTION D'ÉCRAN*

Permet des présentations et des saisies irréprochables. Création de pages-écran de manière interactive. Définition de zones de saisie. Formatage et contrôle automatiques. Définitions des touches de fonction, des attributs vidéo. Sauvegarde des pages sur disque. Création de pages de toutes dimensions.

Une seule instruction permet la saisie complète d'une page-écran.

GESTION MULTIFENÊTRES pour visualiser plusieurs pages à l'écran.

• *GESTION DE FICHIER*

De type séquentiel indexé multiclés. Elle permet l'accès ultra-rapide à tout enregistrement à partir d'une clé d'accès. Permet aussi la lecture ordonnée sur une clé dans l'ordre croissant ou décroissant. Recherche multicritères au sein du fichier. Création d'une nouvelle clé à tout moment. Jusqu'à dix clés par fichier. Aucune réorganisation nécessaire. Intégration totale à MS/DOS et PC/DOS. Gestion automatique des transferts fichier-variables ou fichier-écran.

• *GÉNÉRATEUR D'ÉTATS*

Permet très rapidement une mise en œuvre d'éditions complexes.

• *LOGICIEL DE GESTION DE FICHIER*

Exemple d'application de MASTER, ce logiciel très performant est fourni sous forme de listing source commenté pour une totale compréhension du produit.

• *DOCUMENTATION*

Complète et en français, elle vous permettra de vous familiariser progressivement avec PC MASTER et d'en tirer très rapidement profit.



MICRO APPLICATION se réserve le droit d'interrompre cette offre à tout moment.

OFFRE SPÉCIALE

Ce bas prix ne nuisant en aucune façon aux possibilités quasi-illimitées de cet utilitaire, nous nous engageons à rembourser tout acheteur qui trouverait que ce logiciel n'est pas à la hauteur de ce qu'il annonce... Dans un délai de 8 jours après la date de réception du logiciel et sous réserve que le produit ne soit pas altéré.

MICRO APPLICATION
13, rue Sainte Cécile 75009 PARIS
tél. (1) 47-70-32-44

BON DE COMMANDE

VERSION MASTER PC
BASIC INTERPRETE : ☐
TURBO PASCAL : ☐

PRIX HORS TAXES	=	801,02 FF
PRIX TTC	=	950,00 FF
frais d'envoi en Recommandé	=	40,00 FF
TOTAL	=	990,00 FF

☐ Mandat ☐ Cheque ☐ CCP

Libellez vos cheques a l'ordre de Micro-Application.

Nom, Prenom _____

Adresse _____

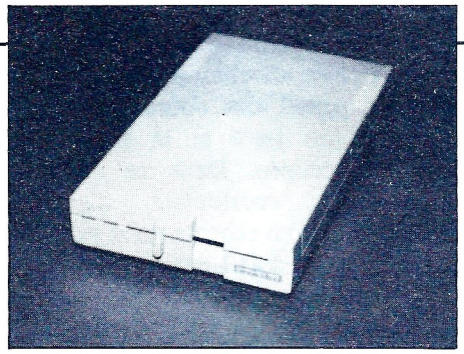
Ville _____ C.P. _____

CB ☐☐☐ ☐☐☐☐☐☐☐☐

Date et signature

01 02 86

LECTEURS 1570 / 1571





LECTEURS 1570/1571

L'enregistrement des informations est certainement l'opération la plus importante, et la plus délicate, en informatique. Les "Commodoristes" d'aujourd'hui préfèrent de plus en plus, nous les encourageons d'ailleurs dans ce sens, la disquette 5"1/4 comme support d'informations. La cassette devrait retourner petit à petit à son utilisation originelle et ne servir qu'à enregistrer vos morceaux de musique préférés.

Pour ceux d'entre vous qui se sont décidés se pose alors le problème du choix. Dernièrement il est apparu sur le marché deux nouveaux lecteurs de disquettes : le 1570 et le 1571. Inutile de s'emballer. La somme à engager mérite toute votre réflexion. Dans l'ordre chronologique des choses les détenteurs de 64 (ou plus 4) doivent savoir que ces deux nouveaux lecteurs de disquettes ne leurs seront d'aucune utilité. En effet les nouvelles performances techniques de ces floppys ne peuvent être exploités qu'à partir d'un Commodore 128. Le bon vieux 1541 leur est donc tout indiqué.
Cependant, si d'aventure ils envisageaient d'acquérir ultérieurement un C128, l'investissement ne serait que momentanément inutile.

Les possesseurs de C128 D, eux n'ont pas à se creuser la tête. Le 1571 est intégré à l'ordinateur.

Reste enfin pour qui le choix leur est réellement proposé : 1541, 1570 ou 1571 ?

La solution économique est évidemment le lecteur 1541. Ils s'habitueront à boire une tasse de café pendant le chargement d'un programme. Pour quelques centaines de francs en plus il pourront disposer d'un 1570 carrossé "1541" mais beaucoup plus rapide (au moins 7 fois, la différence est évidente pour ceux qui sont passés par le 1541). Enfin, si la facture ne les effrayent pas, le 1571 est alors la solution la plus performante : pour une vitesse de transmission égale à celle du 1570, ils doubleront la capacité de stockage de leurs disquettes.

Pour vous conseiller dans votre choix nous vous proposons un tableau comparatif des deux lecteurs 1570 et 1571. Maintenant reste à vous de faire le bon choix !

	1570	1571
FORMAT GCR		
Stockage	1 face	1 ou 2 faces
Capacité totale formatée	174 848 octets	349 696 octets
Taille max. d'un fichier séquentiel	168 656 octets	337 312 octets
Taille max. d'un fichier relatif	167 132 octets	167 132 octets
Articles pour fichiers	65535	65535
Fichiers par disquettes	144	144
Secteurs par disquette	683 total 664 disponibles	1366 total 1328 disponibles
octets par secteurs	256	256
FORMAT MFM		
Capacité d'enregistrement		
128 octets/secteurs	133 120	266 240
256 octets/secteurs	163 840	327 680
512 octets/secteurs	184 320	368 640
1024 octets/secteurs	204 800	409 600
Nombre de piste MAX	40	80
Secteurs par piste		
128 octets/secteurs	26	26
256	16	16
512	9	9
1024	5	5
microprocesseur	6502	6502 A
RAM	2KO	2KO
ROM	32KO	32KO

JEU Un super jeu pour votre

LA CHENILLE

1) Présentation :

La "chenille" est un descendant du célèbre jeu de café "centipède".

Le but du jeu est simple : vous devez guider la chenille pour qu'elle mange les fruits. Dès qu'elle en mange un elle grossit. Elle doit éviter les champignons, la bordure du champ et aussi sa queue. Dès que tous les fruits d'un tableau sont mangés on passe au tableau suivant qui bien sûr contient davantage de champignons et de fruits.
Pour vous diriger, pas besoin de joystick, seulement 2 touches du clavier : une pour aller à droite, l'autre à gauche par rapport à la direction actuelle.
Au début, ce n'est pas évident, mais ensuite vous pourrez vous diriger facilement et rapidement car tout se fait avec uniquement 2 touches.

2) Le principe de programmation :

Le coeur du jeu gravite autour de 2 routines, une qui affiche les tableaux de jeu, l'autre qui s'occupe du déplacement de la chenille.
Ces deux routines n'ont pas de représentation interne de l'état du jeu : elles travaillent directement avec la mémoire écran. Par exemple, pour savoir si la chenille est entrée en collision avec un champignon on ne teste pas si les coordonnées de la chenille correspondent à celles d'un champignon mais on lit tout simplement le caractère se trouvant à l'écran.

Ceci est possible grâce à la routine de la ROM #DB60 qui renvoie le code ASCII du caractère affiché à l'écran à la position du curseur.
Elle est aussi utilisée par la routine "AFTAB" qui affiche le tableau de jeux :

Pour afficher un champignon ou un fruit, on prend des coordonnées au hasard et on teste si la place est libre ou occupée. Dans ce dernier cas, on recalcule d'autres coordonnées et on recommence jusqu'à ce que la place soit libre.

Ainsi on est sûr d'afficher le nombre correct de champignons et de fruits.
Pour la gestion de la chenille, au fur et à mesure qu'elle grossit, on stocke les coordonnées ligne et colonne de tous les caractères qui la composent dans un "buffer" et on tient à jour 2 pointeurs, un pour la tête et un pour la queue.
Ainsi, quand on affiche la tête, on stocke aussitôt ses coordonnées à l'endroit pointé par "tête" et pour effacer la queue on retire les coordonnées pointées par "queue" et on affiche un espace.
Quand la chenille rencontre un fruit, on oublie volontairement d'effacer la queue un certain nombre de fois.

3) Programmation :

Voyons un peu comment ça se passe au niveau programmation.
L'assembleur est sûrement le langage de programmation le moins lisible mais les commentaires présents à chaque ligne vous aideront à comprendre ce qui se passe.
D'abord, on commence par définir l'origine du programme (ORG) mais ceci n'est pas obligatoire, ensuite on indique le point d'entrée.

3.1 Les vecteurs :

Ensuite on octroie un nom à toutes les routines de la ROM qui sont utilisées, de manière à les appeler non pas par leur adresse mais par leur nom, ce qui améliore la lisibilité du programme. Par exemple pour fixer le mode écran, on écrit CALL SET MODE au lieu de CALL #BC0E. Ces routines sont appelées en RAM par les vecteurs, ce qui garantit la comptabilité du jeu avec les 3 différents modèles d'AMSTRAD.

3.2. Les constantes

Ensuite on définit les constantes utilisées par le programme.

"GROS" est la valeur que l'on ajoute à la chenille lorsqu'elle mange un fruit.
Elle est initialisé à 3 mais vous pouvez essayer d'autres valeurs pour obtenir plus ou moins de grossissement.

"LONGUEUR" est la taille du "buffer" de la chenille. Ce buffer contient les coordonnées ligne, colonne de chaque point de la chenille. Ici, LONGUEUR est fixée à 256 ce qui permet d'avoir une chenille de 128 caractères de long, valeur largement suffisante. Si on réduit la taille de ce buffer (par exemple : LONGUEUR EQU 24) on constate que dès que la chenille dépasse 12 caractères de longueur, elle se coupe en 2 et la tête repart toute seule.
INK0, INK1, INK2 et INK3 sont les constantes utilisées pour la couleur des encres. Si vous utilisez un moniteur vert vous pouvez changer ces trois dernières pour leur donner une couleur plus claire.

3.3 Les variables

Les variables en langage machine, correspondent aux variables utilisées en BASIC, à la différence près que l'on travaille directement avec leurs adresses par l'intermédiaire de leurs noms. Par exemple, "niveau" est l'adresse de l'octet qui contient le niveau du jeu.

3.4 Les routines

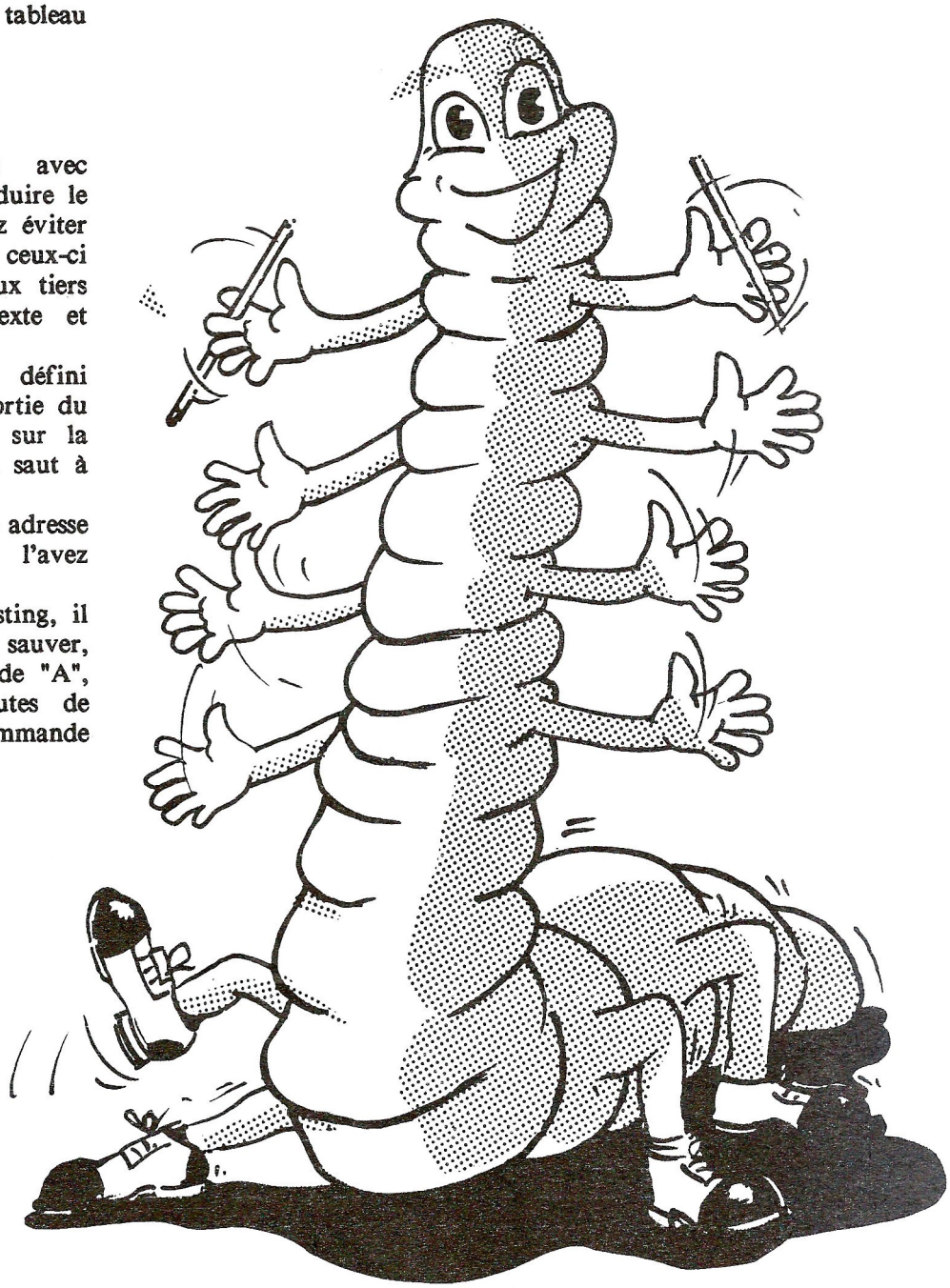
La première routine qui est appelée lorsqu'on lance le jeu est la routine d'initialisation INITP. Elle fixe le mode d'écran 1, redéfinit les caractères, crée les fenêtres, définit les encres utilisées et crée une enveloppe pour l'amplitude du son.

Les trois "CALL" suivants affichent la présentation et la suite se fait par la routine JEUL0 qui est le point par lequel on redemarre le jeu lorsque la chenille est morte.
JEUL1 est le point d'entrée lorsqu'on passe à un tableau suivant.

4) Utilisation avec DAMS

Le listing a été écrit avec l'assembleur DAMS. Pour réduire le travail de saisie vous pouvez éviter de taper les commentaires : ceux-ci représentent en effet les deux tiers de la taille du fichier texte et sont ignorés à l'assemblage.
Le premier label qui est défini (BREAK) est l'adresse de sortie du jeu. Lorsque vous appuyez sur la touche [ESC] le jeu fait un saut à l'adresse [BREAK].
Dans le cas de "DAMS" cette adresse doit être celle où vous l'avez chargé.
Quand vous avez saisi le listing, il ne vous reste plus qu'à le sauver, l'assembler avec la commande "A", corriger les éventuelles fautes de frappe, et le lancer par la commande "J".

Bon jeu !



```
.COPYRIGHT 1985 MICRO-APPLICATION.
.DAMS.

;
;          CHENILLE.
;
;  (C) 1985 MICRO APPLICATION -PS-.
;
;
        ORG  38000
;
        ENT  DEP
BREAK   EQU  5000                   ;Adresse a laquelle est chargee DAMS
;                                       ou adresse d'un moniteur.
;
; Definition des vecteurs utilises.
;
SETMODE  EQU  #BC0E                 ;Fixe mode.
WINDEN   EQU  #BB66                 ;Cree fenetre.
WINDOW   EQU  #BBB4                 ;Selectionne fenetre.
CLW      EQU  #BB6C                 ;Efface fenetre.
PRINT    EQU  #BB5A                 ;Affiche caractere.
RDCHAR   EQU  #BB60                 ;Lis caractere a l'ecran.
SETROW   EQU  #BB72                 ;Fixe numero de ligne.
SETCOL   EQU  #BB6F                 ;Fixe numero de colonne.
SETPEN   EQU  #BB90                 ;Fixe stylot.
SETINK   EQU  #BC32                 ;Fixe encre.
KMRESET  EQU  #BB03                 ;Reset clavier.
WAITCHAR EQU  #BB06                 ;Attend un caractere.
READCHAR EQU  #BB09                 ;Lis clavier.
TESTKEY  EQU  #BB1E                 ;Teste touche.
SDRESET  EQU  #BCA7                 ;Reset son.
SDAMPLI  EQU  #BCBC                 ;Cree une enveloPPe.
SDQUEUE  EQU  #BCAA                 ;Ajoute un son.
SDRELEAS EQU  #BCB3                 ;Produit un son.
;
; Definition des constantes.
;
GROS     EQU  3                     ;Grossissement de dirgestion.
LONGUEUR EQU  256                   ;Taille de la chenille.
;
INK0     EQU  0                     ;Encre 0: noire.
INK1     EQU  23                    ;Encre 1: turquoise Pastel.
INK2     EQU  6                     ;Encre 2: Rouge vif.
INK3     EQU  18                    ;Encre 3: vert vif.
;
; Definition des variables.
;
AX
NIVEAU   DEFB                       ;Niveau de jeux.
TETE     DEFW                       ;Adresse de la tete.
QUEUE    DEFW                       ;Adresse de la queue.
DIGERE   DEFB                       ;Duree de la digestion.
DIRECT   DEFB                       ;Direction.
SEED     DEFW                       ;Nombre aleatoire.
SCORE    DEFW
HISCORE  DEFS 2
VITESSE  DEFB 64
NBFRUIT  DEFB                       ;Nombre de fruits.
COORD    DEFW                       ;Coordonnees de la tete.
;
;
DEP
        CALL INITP
        CALL INITJEU                ;Score = 0 , Niveau = 1.
        CALL AFWIND1                ;Affiche la fenetre 1.
        CALL HELP                   ;Affiche les instructions.
;
;
JEUL0
        CALL INITJEU                ;Score = 0 , Niveau = 1.
JEUL1
        CALL AFWIND1                ;Affiche la fenetre 1.
        CALL AFTAB                  ;Affiche le tableau de jeux.
        CALL READY                  ;Joueur Pret ?
        LD   A,2                    ;Active la fenetre 2.
        CALL WINDOW
        LD   HL,BUFFER              ;Adresse de debut  de la chenille.
        LD   (QUEUE),HL             ;Fixe l'adresse de la queue.
        LD   BC,10*256+18           ;B = ligne 10, C = colonne 18.
        LD   (HL),B                 ;Fixe Point de
        INC  HL                     ;  dePart de
        LD   (HL),C                 ;  la chenille.
        LD   (COORD),BC
        INC  HL
        LD   (TETE),HL              ;Fixe l'adresse de la tete.
        CALL LOCATE                 ;Positionne le curseur.
        LD   A,32
        CALL PRINT                  ;Efface ce Point.
        LD   A,255                  ;Signal: reste immobile.
        LD   (DIRECT),A
        LD   (IX+DIGERE-AX),0       ;Initialise la digestion.
;
        CALL JELO                   ;Lance le jeux.
        JR   NC,CRASH               ;Si la chenille est morte, alors fin.
        INC  (IX+NIVEAU-AX)         ;Sinon Passe au niveau suPerieur.
        JR   JEUL1                  ;Continue le jeux.
;
CRASH
        LD   A,MORT                 ;Caractere de mort.
        LD   DE,20                  ;Frequence du son.
        CALL EFCHEN                 ;Efface la Chenille.
        LD   A," "                  ;Caractere d'effacement.
        LD   DE,100                 ;Frequence du son.
        CALL EFCHEN                 ;Efface la chenille.
        LD   DE,(SCORE)
        LD   HL,(HISCORE)
        AND  A
        SBC  HL,DE                  ;Score > hiscore ?
        JP   NC,JEUL0               ;Non alors debut de jeux.
        LD   (HISCORE),DE           ;Oui alors stoke le nouveau.
        JP   JEUL0                  ;Debut de jeux.
;
;
JELO
;
; Boucle PrinciPale du Jeux.
;
; Entree: Pas de registres.
; Sortie: Carry a 1 => le tableau est fini.
;       : Carry a 0 => la chenille est morte.
;
        CALL FTDIR                  ;Teste clavier, modifie la direction.
        CALL CALPOS                 ;Calcule nouvelle Position de la tete
        LD   (COORD),BC             ;  et stocke la.
        CALL TESTPOS                ;Teste si la Place est occuPee.
        PUSH AF                     ;Sauvegarde le flag.
```

```
        CALL Z,MANGE                    ;Si c'est un fruit alors mange le.
        CALL LOCATE                     ;Positionne le curseur
        LD   A,240                      ;  et affiche la tete.
        CALL PRINT
        CALL CHAINE                     ;Enchaine la tete et efface la queue.
        POP  AF                         ;Retire le flag de collision.
        RET  NC                         ;Si il y a eu collision alors fin.
        LD   A,(NBFRUIT)
        AND  A                          ;Reste t-il des fruits ?
        JR   NZ,JELO                    ;Si oui alors continue.
        SCF                             ;Signal: Tableau fini.
        RET
;
INITJEU
        LD   A,1
        LD   (NIVEAU),A
        LD   HL,0
        LD   (SCORE),HL
        RET
;
;
EFCHEN
;
; Routine d'effacement de la chenille.
;
; Entree: A  = caractere d'effacement.
;         DE = hauteur initiale du son.
;
; Sortie: Pas de registres.
;
        LD   HL,(TETE)                  ;HL = adresse de la tete.
EFCHENL1
        PUSH AF                         ;Sauvegarde le caractere d'effacement.
        LD   BC,BUFFER+1                ;Bas du buffer de la chenille.
        AND  A                          ;Met carry a 0.
        SBC  HL,BC                      ;La tete est au debut du buffer ?
        ADD  HL,BC                      ;Restaure HL sans perturber carry.
        JR   NC,EFCHENP1                ;Si non alors Passe.
        LD   HL,ENDBUFF                 ;Si oui alors met la tete en haut.
EFCHENP1 DEC  HL
        LD   C,(HL)                     ;Met coord. colonne de la tete dans C.
        DEC  HL
        LD   B,(HL)                     ;Met coord. ligne de la tete dans B.
        PUSH AF                         ;Sauvegarde caractere d'effacement.
        CALL LOCATE                     ;Posisionne le curseur.
        POP  AF                         ;Retire le caractere d'effacement
        CALL PRINT                      ;  et efface.
        PUSH HL                         ;Sauvegarde adresse de la tete.
        PUSH DE                         ;Sauvegarde frequence du son.
        LD   HL,0                       ;Signal: duree est definie par enveloP.
        LD   A,1                        ;Signal: enveloPPe numero 1.
        CALL SON                        ;Produit le son.
        POP  DE                         ;Retire frequence du son.
        LD   HL,30
        ADD  HL,DE                      ;Baisse la frequece.
        EX   DE,HL                      ;Remet le resultat dans DE.
        LD   A,20                       ;Attend 20 unitees.
        CALL TEMPO
        POP  HL                         ;Retire adresse de la tete.
        POP  AF                         ;Retire caractere d'effacement.
        AND  A                          ;Met carry a 0.
        LD   BC,(QUEUE)                 ;BC Prends l'adresse de la queue.
        PUSH HL
        SBC  HL,BC                      ;Toute la chenille a ete effacee ?
        POP  HL
        JR   NZ,EFCHENL1                ;Si non alors continue.
        RET                             ;Si oui alors fin.
;
;
TEMPO
;
; Produit une temporisation dont la duree depend de A.
;
        LD   B,A
TEMPOL2
        LD   C,0
TEMPOL1
        DEC  C
        JR   NZ,TEMPOL1
        DJNZ TEMPOL2
        RET
;
;
;
FTDIR
;
; Teste le clavier et calcule nouvelle direction.
;
; Entree: Pas de registres.
;
; Sortie: A  = code de direction.
;
        LD   D,(IX+DIRECT-AX)           ;D Prends l'ancienne direction.
        CALL LICLAV                     ;Lit le clavier.
        LD   A,D                        ;Passe le code de direction a A.
        RET  NC                         ;Revient si direction n'a Pas change.
        AND  3                          ;Modulo 4  (Pour les 4 directions).
        LD   (DIRECT),A                 ;Stocke nouvelle direction.
        RET
;
LICLAV
        LD   A,66                       ;Teste la touche EscaPe.
        CALL TESTKEY
        JP   NZ,BREAK                   ;Si elle est enfoncee alors exit.
        CALL READCHAR                   ;Retire catactere du buffer clavier.
        RET  NC                         ;Revient si il n'y en avait Pas.
        CP   "/"                        ;Touche "a gauche" ?
        JR   NZ,LICLAVP3                ;Si non alors Passe.
        DEC  D                          ;Calcule nouvelle direction a gauche.
        SCF                             ;Signal: la direction a change.
        RET
LICLAVP3
        CP   "\"                        ;Touche "a droite" ?
        JR   NZ,LICLAVP4                ;Si non alors Passe.
        INC  D                          ;Calcule nouvelle direction a droite.
        SCF                             ;Signal: la direction a change.
        RET
LICLAVP4
        OR   A                          ;Signal: la direction n'a Pas change.
        RET
;
CALPOS
;
; Calcule la nouvelle Position de la tete d'aPres la direction.
;
; Entree: A  = code de direction.
;
; Sortie: BC = coordonnees de la tete.
```

```
;
        LD   BC,(COORD)             ;Ancienne coordonnees de la tete.
        AND  A                      ;A = 0 ?
        JR   Z,DIRDROI              ;Si oui alors direction a droite.
        DEC  A                      ;A = 1 ?
        JR   Z,DIRBAS               ;Si oui alors direction vers le bas.
        DEC  A                      ;A = 2 ?
        JR   Z,DIRGAU               ;Si oui alors direction a gauche.
        DEC  A                      ;A = 3 ?
        RET  NZ                     ;Retour si direc. pas vers le haut.
;
        DEC  B                      ;Passe a la ligne superieure.
        RET
DIRBAS
        INC  B                      ;Passe a la ligne de dessous.
        RET
DIRDROI
        INC  C                      ;Passe a la colonne de droite.
        RET
DIRGAU
        DEC  C                      ;Passe a la colonne de gauche.
        RET
;
;
TESTPOS
;
; Teste si la position de la tete est un espace.
;
; Entree: BC = coordonnees ligne et colonne de la tete.
;
; Sortie: carry = 0  => la chenille a heurte un obstacle.
;         carry = 1  => OK.
;             Z = 1  => la chenille a mange un fruit.
;             Z = 0  => OK.
;
        CALL LOCATE                 ;Positionne le curseur.
        CALL RDCHAR                 ;Lis le caractere a l'ecran.
        CCF
        RET  C                      ;Retour avec signal OK si illisible.
        CP   FRUIT                  ;Est-ce un fruit ?
        SCF                         ;Signal: OK.
        RET  Z                      ;Retour si oui.
        CP   33                     ;Si ce n'est pas un espace
        RET                         ;  alors retour avec carry annule.
;
MANGE
;
; Produit un son, calcule et affiche le score.
;
        PUSH BC                     ;Sauvegarde les coordonnees de la tete.
        LD   A,(NIVEAU)
        LD   E,A
        ADD  A,A                    ;Multiplie par 2.
        ADD  A,A                    ;Par 4.
        ADD  A,E                    ;Par 5.
        ADD  A,A                    ;Par 10.
        LD   E,A                    ;Passe le resultat a E.
        LD   A,(VITESSE)
        RRCA                        ;Divise par 2.
        RRCA                        ;Par 4.
        RRCA                        ;Par 8.
        LD   D,A                    ;Passe le resultat a D.
        LD   A,31
        SUB  D                      ;Inverse l'intervalle 0-31.
        RLCA                        ;Multipli par 2.
        ADD  A,E                    ;Ajoute le premier total.
        LD   E,A                    ;Passe le resultat a E.
        LD   D,0                    ;Reset le high-byte.
        LD   HL,(SCORE)
        ADD  HL,DE                  ;Met le score a jour
        LD   (SCORE),HL             ;  et sauvegarde le.
        EX   DE,HL
        LD   A,1                    ;Active la fenetre 1.
        CALL WINDOW
        LD   BC,256+4               ;Ligne 1, Colone 4.
        CALL LOCATE                 ;Positionne le curseur.
        CALL OUTNB                  ;Affiche le score.
        LD   HL,0                   ;Signal: utilise l'enveloppe.
        LD   DE,700                 ;Frequence du son.
        LD   A,1                    ;Signal: Enveloppe numero 1.
        CALL SON                    ;Produit le son.
        LD   A,2                    ;Active la fenetre 2.
        CALL WINDOW
        LD   A,(DIGERE)             ;A = etat de digestion.
        ADD  A,GROS                 ;Ajoute la digestion actuelle
        LD   (DIGERE),A             ;  et stocke la.
        DEC  (IX+NBFRUIT-AX)        ;Decremente le nombre de fruits.
        POP  BC                     ;Retire coordonnees de la tete.
        RET
;
;
CHAINE
;
; Ajoute la derniere position de la tete dans la chaine.
;
; Entree: BC = Coordonnees ligne et colonne de la tete.
;
        LD   HL,(TETE)              ;Adresse de la tete.
        LD   (HL),B                 ;Ajoute nouvelle ligne.
        INC  HL
        LD   (HL),C                 ;Ajoute nouvelle colonne.
        INC  HL
        CALL TESTENDB               ;Teste si la fin du buffer est atteinte
        LD   (TETE),HL              ;Stocke la nouvelle adresse de la tete.
        LD   A,(DIGERE)
        AND  A                      ;Cycle de digestion ?
        JR   Z,EFFQUEUE             ;Si non passe.
        CP   GROS                   ;Est-ce le premier cycle ?
        LD   A,(VITESSE)
        CALL NZ,TEMPO               ;Si non ajoute temporisation.
        DEC  (IX+DIGERE-AX)         ;Un cycle de digestion en moins.
        RET
EFFQUEUE
        LD   A,(VITESSE)
        CALL TEMPO                  ;Attend.
        LD   HL,(QUEUE)             ;Adresse de la queue.
        LD   B,(HL)                 ;Coord. ligne de la queue.
        INC  HL
        LD   C,(HL)                 ;Coord. colonne de la queue.
        INC  HL
        CALL TESTENDB               ;Teste si fin du buffer est atteinte.
        LD   (QUEUE),HL             ;Stocke nouvelle adresse de la queue.
        CALL LOCATE                 ;Positionne le curseur,
        LD   A,32                   ;  et efface la queue.
        JP   PRINT
;
TESTENDB
;
```

```
; Teste si fin du buffer est atteinte.
;
        LD   DE,ENDBUFF
        AND  A
        SBC  HL,DE
        ADD  HL,DE
        RET  C
        LD   HL,BUFFER
        RET
;
;
AFWIND1
;
; Affiche la Premiere fenetre.
;
        LD   A,1                    ;Active la fenetre 1.
        CALL WINDOW
        LD   BC,256+1               ;Ligne 1, colonne 1.
        CALL LOCATE                 ;Positionne le curseur.
        CALL PRTL                   ;Affiche chaine de caracteres.
        DEFM SC:
        DEFB -1                     ;Signal: fin de chaine.
        LD   DE,(SCORE)
        CALL OUTNB                  ;Affiche le score.
        LD   BC,256+10              ;Ligne 1, colonne 10.
        CALL LOCATE                 ;Positionne le curseur.
        CALL PRTL                   ;Affiche chaine de caracteres qui suit.
        DEFM NIVEAU:
        DEFB -1                     ;Signal: fin de chaine.
        LD   A,(NIVEAU)
        ADD  A,48                   ;Transforme le niveau en code ASCII,
        CALL PRINT                  ;  et affiche le.
        LD   BC,256+20              ;Ligne 1, colonne 20.
        CALL LOCATE                 ;Positionne le curseur.
        CALL PRTL                   ;Affiche chaine de caracteres qui suit.
        DEFM SPEED:
        DEFB -1                     ;Signal: fin de chaine.
        LD   A,(VITESSE)
        SRL  A                      ;Divise par 2.
        SRL  A                      ;Par 4.
        SRL  A                      ;Par 8.
        DEC  A
        AND  31                     ;Modulo 32.
        LD   E,A                    ;Passe le resultat a E.
        LD   D,0                    ;Reset hi-byte.
        CALL OUTNB2                 ;Affiche le resultat.
        LD   BC,256+30              ;Ligne 1, colonne 1.
        CALL LOCATE                 ;Positionne le curseur.
        CALL PRTL                   ;Affiche chaine de caractere qui suit.
        DEFM HISC:
        DEFB -1                     ;Signal: fin de chaine.
        LD   DE,(HISCORE)
        JP   OUTNB                  ;Affiche le hi-score.
;
;
AFTAB
;
; Affiche le tableau de jeux.
;
        LD   A,2                    ;Active la fenetre 2.
        CALL WINDOW
        CALL CLW                    ;Efface la fenetre.
        CALL CADRE                  ;Affiche le cadre.
;
        LD   A,(NIVEAU)
        LD   B,A                    ;B sert de compteur.
        PUSH BC                     ;Sauvegarde le niveau.
CHAMPL1
        PUSH BC                     ;Sauvegarde le compteur.
        CALL AFCHP                  ;Affiche 2 champignons
        CALL AFCHP                  ;  par niveau.
        POP  BC                     ;Retire le compteur.
        DJNZ CHAMPL1
;
        LD   (IX+NBFRUIT-AX),0
        LD   A,3                    ;Pen 3.
        CALL SETPEN
        POP  BC                     ;Retire le niveau de jeux.
        SLA  B                      ;Multiplie B par 2.
        SLA  B                      ;Par 4.
FRUITL1
        PUSH BC                     ;Sauvegarde le compteur.
        CALL AFRUIT                 ;Affiche un fruit.
        POP  BC                     ;Retire le compteur.
        DJNZ FRUITL1
        CALL KMRESET                ;Vide le buffer d'entree du clavier.
        LD   A,1                    ;Pen 1.
        CALL SETPEN
        LD   BC,10*256+18           ;Ligne 10, colone 18.
        CALL LOCATE                 ;Positionne le curseur.
        LD   A,CHENI                ;Affiche la tete de la chenille.
        JP   PRINT
;
CADRE
;
; Affiche le cadre de la fenetre 2.
;
        LD   A,2                    ;Pen 2.
        CALL SETPEN
        LD   BC,256+1               ;Ligne 1, colonne 1.
        CALL LOCATE                 ;Positionne le curseur.
        LD   A,BORD                 ;Caractere de bordure.
        LD   B,40
CADREL1 CALL PRINT
        DJNZ CADREL1                ;Affiche 40 fois.
;
        LD   BC,22*256+1            ;Ligne 22, colonne 1.
        CALL LOCATE
        LD   A,BORD
        LD   B,40
CADREL2 CALL PRINT
        DJNZ CADREL2                ;Affiche 40 fois.
;
        LD   BC,2*256+1             ;Ligne 2, colonne 1.
        CALL LOCATE
        LD   B,20
CADREL3 LD   A,BORD
        CALL PRINT
        LD   A,10                   ;Ligne suivante.
        CALL PRINT
        LD   A,8                    ;recule d'une case.
        CALL PRINT
        DJNZ CADREL3                ;Affiche 20 fois.
;
        LD   BC,2*256+40            ;Ligne 2, colonne 40.
        CALL LOCATE
        LD   B,20
CADREL4 LD   A,BORD
```

```
        CALL PRINT
        LD   A,8                        ;Recule d'une case.
        CALL PRINT
        LD   A,10                       ;Ligne suivante.
        CALL PRINT
        DJNZ CADREL4                    ;Affiche 20 fois.
        RET
;
AFCHP
;
; Calcule une Position et affiche un champignon.
;
AFCHPL1
        CALL RND                        ;Produit un nombre aleatoire dans A.
        AND  15                         ;Modulo 16.
        CP   10                         ;Modulo 9.
        JR   C,AFCHPP1
        SUB  7
AFCHPP1
        ADD  A,A                        ;Multiplie par 2.
        INC  A                          ;Produit un nombre Pair
        INC  A                          ;  entre 2 et 18 inclus.
        LD   B,A                        ;Resultat dans B  (ligne).
        CALL RND                        ;Produit un nombre aleatoire dans A.
        AND  31                         ;Modulo 32.
        CP   20                         ;Modulo 20.
        JR   C,AFCHPP2
        SUB  14
AFCHPP2
        ADD  A,A                        ;Multiplie Par 2.
        INC  A                          ;Produit un nombre Pair
        INC  A                          ;  entre 2 et 38 inclus.
        LD   C,A                        ;Resultat dans C  (colonne).
        CALL TESTPOS                    ;La Position est occupee ?
        JR   NC,AFCHPL1                 ;Si oui, calcule une autre Position.
        CALL TESTMID                    ;Est-ce le milieu de la fenetre ?.
        JR   Z,AFCHPL1                  ;Si oui, recalcule une autre Position.

        LD   A,CHAMPI
        CALL PRINT                      ;Affiche un quart du champignon.
        INC  A                          ;Affiche un autre quart.
        CALL PRINT
        INC  B                          ;Ligne suivante.
        PUSH AF
        CALL LOCATE                     ;Positionne curseur.
        POP  AF
        INC  A                          ;Affiche le troisieme quart.
        CALL PRINT
        INC  A                          ;Affiche le dernier quart.
        JP   PRINT
;
;
AFRUIT
;
; Calcule une Position et affiche un fruit.
;
        CALL RND                        ;Produit un nombre aleatoire dans A.
        AND  31                         ;Modulo 32.
        CP   20                         ;Modulo 20.
        JR   C,AFRUITP1
        SUB  12
AFRUITP1
        INC  A
        INC  A                          ;Intervalle  2 a 21.
        LD   B,A                        ;Numero de ligne dans B.
        CALL RND                        ;Produit nombre aleatoire dans A.
        AND  63                         ;Modulo 64.
        CP   38                         ;Modulo 38.
        JR   C,AFRUITP2
        SUB  26
AFRUITP2
        INC  A
        INC  A                          ;Intervalle 2 a 39.
        LD   C,A                        ;Numero de colonne dans C.
        CALL TESTPOS                    ;Cette Position est occupee ?
        JR   NC,AFRUIT                  ;Si il y a un obstacle, re-calcule.
        JR   Z,AFRUIT                   ;Si il y a un fruit aussi.
        CALL TESTMID                    ;Est-ce le milieu de l'ecran ?
        JR   Z,AFRUIT                   ;Si oui, re-calcule.
        INC  (IX+NBFRUIT-AX)            ;Un fruit de Plus.
        LD   A,FRUIT                    ;Affiche-le.
        JP   PRINT
;
READY
;
; Attend une touche et agit en consequence.
;
        LD   A,3                        ;Active la fenetre 3.
        CALL WINDOW
        CALL CLW                        ;Efface la fenetre.
        CALL PRTL                       ;Affiche chaine de caracteres qui suit.
        DEFM OK ? (o/n), v =
        DEFM vitesse, h = help
        DEFB -1                         ;Signal: fin de chaine.
        CALL WAITCHAR                   ;Attend un caractere du clavier.
        PUSH AF
        CALL CLW                        ;Avant de le traiter Efface la fenetre.
        POP  AF
        RES  5,A                        ;Converti les minuscules en majuscules.
        CALL READYT                     ;Traite le caractere.
        JR   NC,READY                   ;Si c'etait une commande alors recom.
        RET
READYT
        CP   "N"
        JR   Z,NOREADY                  ;Re-affiche un tableau.
        CP   "V"
        JR   Z,SPEED                    ;Change la vitesse de jeux.
        CP   "H"
        JR   Z,AFHELP                   ;Affiche help.
        SCF                             ;Signal: Pas de commande.
        RET
;
NOREADY
        CALL AFTAB                      ;Re-affiche un tableau.
        OR   A                          ;Signal: commande.
        RET
;
SPEED
        CALL PRTL                       ;Affiche chaine suivante.
        DEFM / ET \ POUR MOINS ET
        DEFM PLUS.
        DEFB -1                         ;Signal: fin de chaine.
SPEEDL1
        CALL WAITCHAR                   ;Attend un caractere.
        LD   C,(IX+VITESSE-AX)
        CP   "/"                        ;Moins vite ?
        JR   Z,SPEEDP1
        CP   "\"                        ;Plus vite ?
```

```
        JR   Z,SPEEDP2
        OR   A                          ;Signal: commande.
        RET
SPEEDP1
        LD   A,C
        SUB  8                          ;Reduit la vitesse.
        JR   SPEEDP3
SPEEDP2
        LD   A,C
        ADD  A,8                        ;Augmente la vitesse.
SPEEDP3 LD   (VITESSE),A
        CALL AFWIND1                    ;Affiche la nouvelle vitesse.
        JR   SPEEDL1                    ;Recommence.
;
AFHELP
        CALL HELP                       ;Affiche help.
        CALL AFTAB                      ;Affiche le tableau de jeux.
        OR   A                          ;Signal commande.
        RET
;
HELP
        LD   A,2                        ;Active la fenetre 2.
        CALL WINDOW
        CALL CLW                        ;Efface la fenetre.
        CALL CADRE                      ;Affiche le cadre.
        LD   A,1                        ;Pen 1.
        CALL SETPEN
        LD   BC,4*256+6                 ;Ligne 4, colone 6.
        CALL LOCATE
        CALL PRTL
        DEFM Vous etes aux commandes d
        DEFM 'une
        DEFB 31,3,5
        DEFM chenille folle qui se nou
        DEFM rrit
        DEFB 31,3,6
        DEFM de fruits.
        DEFB 31,6,8
        DEFM Pour la guider vo
        DEFM us disposez des
        DEFB 31,3,9
        DEFM touches / et \ pour droi
        DEFM te et gauche.
        DEFB -1                         ;Signal: fin de chaine.
        JP   WAITCHAR                   ;Attend un caractere du clavier.
;
;
TESTMID
;
; Teste si les coordonees BC sont celle du millieu de l'ecran.
;
        LD   A,B
        CP   10
        RET  NZ
        LD   A,C
        CP   18
        RET
;
;
RND
;
; Produit un nombre pseudo-aleatoire dans A.
;
        LD   HL,(SEED)                  ;Ancien nombre aleatoire.
        PUSH HL
        ADD  HL,HL                      ;Multiplie par 2.
        ADD  HL,HL                      ;Par 4.
        POP  DE
        ADD  HL,DE                      ;Par 5.
        ADD  HL,DE                      ;Par 6.
        ADD  HL,DE                      ;Par 7.
        LD   DE,41
        ADD  HL,DE
        LD   (SEED),HL                  ;Stocke nouveau nombre.
        LD   A,L
        XOR  H                          ;Reduit  de 16 bits en 8 bits.
        RET
;
RANDOM
;
; Cree le Hasard.
;
        LD   A,R
        LD   (SEED),A
        RET
;
;
LOCATE
;
; Positionne le curseur.
;
; Entree: BC contient la ligne et la colonne.
;
        PUSH HL
        LD   A,B
        CALL SETROW                     ;Fixe ligne.
        LD   A,C
        CALL SETCOL                     ;Fixe colonne.
        POP  HL
        RET
;
;
INITP
;
; Initialisation
;
        LD   A,1                        ;Mode 1.
        CALL SETMODE
        CALL REDEFC                     ;Redefinit les caracteres.
        LD   A,1                        ;Fenetre 1.
        CALL WINDOW
        LD   HL,0                       ;Coin haut gauche.
        LD   DE,39*256+1                ;Coin bas droite.
        CALL WINDEN                     ;Cree la fenetre.
        LD   A,2                        ;Fenetre 2.
        CALL WINDOW
        LD   HL,2                       ;Coin haut gauche.
        LD   DE,39*256+23               ;Coin bas droite.
        CALL WINDEN                     ;Cree la fenetre.
        LD   A,3                        ;fenetre 3.
        CALL WINDOW
        LD   HL,24                      ;Coin haut gauche.
        LD   DE,39*256+24               ;Coin bas droite.
        CALL WINDEN                     ;Cree la fenetre.
        LD   A,0                        ;Ink 0.
        LD   BC,INK0*256+INK0           ;Couleurs dans B et C.
        CALL SETINK
        LD   A,1                        ;Ink 1.
```

```
        LD   BC,INK1*256+INK1       ;Couleurs dans B et C.
        CALL SETINK
        LD   A,2                    ;Ink 2.
        LD   BC,INK2*256+INK2       ;Couleurs dans B et C.
        CALL SETINK
        LD   A,3                    ;Ink 3.
        LD   BC,INK3*256+INK3       ;Couleurs dans B et C.
        CALL SETINK
        LD   HL,AMPLENV             ;Adresse des donnees d'enveloppe.
        LD   A,1                    ;Enveloppe numero 1.
        CALL SDAMPLI                ;Cree l'enveloppe.
        CALL RANDOM                 ;Cree le hasard.
        LD   IX,AX                  ;IX Pointe sur les variables.
        RET
;
;
REDEFC
;
; Redefini les caracteres.
;
        LD   C,240                  ;Premier caractere redefini.
        LD   HL,TABCC               ;Table des caracteres modeles.
        LD   B,8                    ;8 caracteres a redefinir.
REDEFCL1
        PUSH BC
        LD   A,25                   ;Signal: Redefinition.
        CALL PRINT
        LD   A,C                    ;Code du caractere a redefinir.
        CALL PRINT
        LD   B,8                    ;Compteur de Parametres.
REDEFCL2
        LD   A,(HL)                 ;Prends un Parametre.
        INC  HL
        CALL PRINT                  ;Envoie-le.
        DJNZ REDEFCL2               ;Recommence huit fois.
        POP  BC
        INC  C                      ;Caractere suivant.
        DJNZ REDEFCL1               ;Recommence 8 fois.
        RET
;
PRTL
;
; Affiche la chaine de caractere adressee Par PC.
;
; Entree: PC = adresse de la chaine de caracteres.
;
        POP  DE                     ;Retire l'adresse.
        LD   A,(DE)                 ;Prend le caractere.
        INC  DE                     ;Pointe sur le caractere suivant
        PUSH DE                     ;  et stocke son adresse.
        CP   255                    ;Est-ce le caractere de fin ?
        RET  Z                      ;Si oui alors fin.
        CALL PRINT                  ;Si non affiche-le
        JR   PRTL                   ;  et continue.
;
;
OUTNB
;
; Affiche un nombre sur 5 chiffres.
;
; Entree: DE = nombre a afficher.
;
        LD   BC,-10000              ;Chiffre des dizaines de milliers.
        CALL PRCH                   ;Sort le chiffre.
        LD   BC,-1000               ;Chiffre des milliers.
        CALL PRCH
        LD   BC,-100                ;Chiffre des centaines.
        CALL PRCH
OUTNB2  LD   BC,-10                 ;Chiffre des dizaines.
        CALL PRCH
        LD   A,E
        ADD  A,48                   ;Chiffre des unites.
        JP   PRINT
;
PRCH
;
; Affiche un chiffre
;
; Entree: BC contient l'oPPose du multiPlicateur du chiffre.
;       : DE contient le nombre duQuel on extrait le chiffre.
;
        EX   DE,HL                  ;Passe le nombre a HL Pour la division.
        XOR  A                      ;Compteur a 0.
PRCHL1  ADD  HL,BC                  ;Soustrais le diviseur.
        INC  A                      ;Compte chaque passage.
        JR   C,PRCHL1               ;Tant que c'est possible.
        SBC  HL,BC                  ;Restaure la soustraction de troP.
        EX   DE,HL                  ;Le reste de la division dans DE.
        ADD  A,47                   ;Resultat en code ASCII.
        JP   PRINT                  ;  et affiche le chiffre.
;
;
SON
;
; Produit un son.
;
; Entree: HL = duree du son.
;       : DE = Freauence du son.
;         A = Numero d'enveloPPe a utiliser.
;
        PUSH IX
        PUSH DE
        LD   IY,SDPARAM             ;IY Pointe sur table de Parametres.
        LD   (IY+1),A
        LD   (IY+7),L
        LD   (IY+8),H
        PUSH DE
        CALL SDRESET                ;StoPPe les sons.
        LD   (IY+0),%01000001       ;Signal: Canal 1.
        POP  DE
        CALL SETTONE                ;Inclue la freauence.
        CALL SOUND
        LD   (IY+0),%01000010       ;Signal: Canal 2.
        CALL EFFET                  ;Desacorde Pour faire un son Puissant.
        CALL SOUND
        LD   (IY+0),%01000100       ;Signal: Canal 3.
        CALL EFFET
        CALL SOUND
        LD   A,7                    ;Signal: Tous les canaux.
        CALL SDRELEAS               ;Produit le son.
        POP  DE
        POP  IX
        RET
;
SOUND
        PUSH DE
        LD   HL,SDPARAM
        CALL SDQUEUE                ;Met le son dans la queue d'attente.
```

```
        POP  DE
        RET
;
EFFET
        LD   A,R
        AND  7                      ;Modulo8.
        INC  A
        EX   DE,HL
        LD   E,A
        LD   D,0
        ADD  HL,DE
        EX   DE,HL                  ;Nouvelle fre9uence dans DE.
SETTONE
        LD   (IY+3),E
        LD   (IY+4),D
        RET
;
SDPARAM
        DEFB 0                      ;Canal.
        DEFB 0                      ;Numero d'enveloPPe.
        DEFB 0                      ;EnveloPPe de fre9uence.
        DEFW 0                      ;Fre9uence du son.
        DEFB 0                      ;Periode de bruit.
        DEFB 0                      ;AmPlitude initiale.
        DEFW 0                      ;Duree
;
AMPLENV
        DEFB 2                      ;Nombre de sections d'env. utilises.
;
        DEFB 1                      ;Nombre de Pas.
        DEFB 15                     ;Taille du Pas.
        DEFB 1                      ;TemPs de Pause.
;
        DEFB 15                     ;Nombre de Pas.
        DEFB -1                     ;Taille du Pas.
        DEFB 3                      ;TemPs de Pause.
;
        DEFS 9,0                    ;Parametres inutilises.
;
;
BUFFER
        DEFS LONGUEUR,0             ;Place Pour la chenille.
ENDBUFF
;
TABCC
;
CHENI   EQU  240
;
        DEFB %00111100
        DEFB %01011110
        DEFB %10011111
        DEFB %11111111
        DEFB %11111111
        DEFB %11111111
        DEFB %01111110
        DEFB %00111100
;
MORT    EQU  241
;
        DEFB %00111100
        DEFB %01100110
        DEFB %11000011
        DEFB %10000001
        DEFB %10000001
        DEFB %11000011
        DEFB %01100110
        DEFB %00111100
;
FRUIT   EQU  242
;
        DEFB %00000010
        DEFB %00000100
        DEFB %00001000
        DEFB %01010110
        DEFB %11111111
        DEFB %11111110
        DEFB %01111110
        DEFB %00101100
;
BORD    EQU  243
;
        DEFB %00011000
        DEFB %00011000
        DEFB %11111111
        DEFB %11111111
        DEFB %00011000
        DEFB %11111111
        DEFB %11111111
        DEFB %00011000
;
CHAMPI  EQU  244
;
        DEFB %00000000
        DEFB %00011111
        DEFB %01111111
        DEFB %11111111
        DEFB %11111111
        DEFB %11111100
        DEFB %01100011
        DEFB %00000111
;
        DEFB %00000000
        DEFB %11110000
        DEFB %11111100
        DEFB %11111110
        DEFB %11111111
        DEFB %01111110
        DEFB %10000000
        DEFB %11100000
;
        DEFB %00011111
        DEFB %00100111
        DEFB %00000111
        DEFB %00001111
        DEFB %00011111
        DEFB %00011111
        DEFB %00001111
        DEFB %00000011
;
        DEFB %11111000
        DEFB %11100100
        DEFB %11100000
        DEFB %11110000
        DEFB %11110000
        DEFB %11110000
        DEFB %11100000
        DEFB %11000000
;
        FIN

Text:12457          End:30856          18399 Bytes
Hmem:36000
```

MICRO INFO

TRUCS ET ASTUCES

TRUCS ET ASTUCES DU CPC 464

Getscreen

lecture des caractères à l'écran.

La gestion d'écran spécifique du CPC rend la lecture d'un caractère directement à l'écran très malaisée.

La routine CHECK de JAN-MIRKO MACZEWSKI simplifie beaucoup la tâche. Le programme est auto-documenté.

```
10 '\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
20 '  Chargeur basic pour  !check,x,y . CPC 464
30 '/////////////////////////////////////////////////////////////
40 MODE 2
50 MEMORY &7FFF
60 FOR i=1 TO 45
70 READ valeur$:valeur=VAL("&"+valeur$):POKE &7FFF+i,valeur
80 s=s+valeur
90 NEXT i
100 IF s <> 4164 THEN PRINT "Erreur dans le Premier bloc de data":END
110 s=0
120 POKE &802D,&C9
130 '  Lecture de la routine
140 FOR i=1 TO 28
150 READ valeur$:valeur=VAL("&"+valeur$):POKE 39994+i,valeur
160 s=s+valeur
170 NEXT i
180 IF s <> 3519 THEN PRINT "Erreur dans le deuxieme bloc de data":END
190 CALL &8000
200 PRINT "Nouvel ordre : !CHECK,x,y"
210 PRINT:PRINT"C'est le contraire de l'ordre 'LOCATE' qui Positionne un caract
re en x,y ."
220 PRINT"En effet, il lit le caractere situe a cette Place et sauve le caracte
e ASCII    dans la  Place memoire 41007.
230 PRINT "On Peut  obtenir le caractere Par 'PRINT CHR$(PEEK(41007))' ."
240 PRINT"On obtient le code ASCII en taPant 'PRINT PEEK(41007)'.":PRINT "examP
e : "
250 PRINT:PRINT CHR$(24)"!check,12,1",CHR$(24)"    lit le caractere en "CHR$(24
"Li9ne 1"CHR$(24);
260 PRINT ","CHR$(24)" colonne 12"CHR$(24);" et inscrit le":PRINT
270 PRINT"                code ASCII correspondant en 41007"
280 '
290 ' Datas Pour liasons RSX, table, tests de Parametres, etc...
300 '
310 DATA 01,09,80,21,16,80,C3,D1,BC
320 DATA 0E,80,C3,1A,80,43,48,45,43
330 DATA CB,00,00,00,00,00,00,00,FE
340 DATA 02,C0,ED,53,28,A0,DD,66,03
350 DATA DD,6E,02,22,29,A0,CD,3B,9C
360 '
370 ' Datas Pour routine !check
380 '
390 DATA CD,78,BB,22,64,A1,2A
400 DATA 28,A0,01,2A,A0,CD,75
410 DATA BB,CD,60,BB,32,2F,A0
420 DATA 2A,64,A1,CD,75,BB,C9
```

MICRO INFO

LES DETRITUS DU TAMPON CLAVIER

Quand votre programme effectue de longs calculs, l'utilisateur peut le croire bloqué et presser n'importe quelle touche. Cela reste sans effet sur le programme mais les touches pressées sont chargées dans le tampon clavier. Si un INPUT est alors exécuté, les caractères des touches pressées apparaissent au début de la ligne à entrer. Le petit programme qui suit évite cela :

```
60000 P$ = INKEY $ :
WHILE P$ <> "":
P$ = INKEY$ :
WEND:RETURN
```

AMELIORATION DE L'ASSEMBLEUR DU CPC

De l'avis de nombreux programmeurs en langage machine, l'assembleur Z-80 qui est listé dans "le langage machine pour l'AMSTRAD CPC" dispose de nombreuses facilités.
Mais rien n'est si bien que ce qui se laisse transformer. Aussi, ROBERT HAIRGROVE de ST GALLEN en Suisse nous donne un truc pour mettre le code-objet à une adresse plus faible que &A000 :

```
1 MEMORY & 5FFF : GOTO 10 000
.
14180 VAPT = ZEI $
.
14310 BPC = 384 : mpc = HIMEM + 1 :
mpstrart = mpc
```

TOUJOURS DES ENNUIS AVEC LA DATE

Comment vos programmes se comportent-ils lorsque, devant préciser la date, un utilisateur particulièrement malveillant tape le 33.08.85 ou bien le 01.15.85 ?

Avec le programme qui suit, l'utilisateur entrera des dates correctes lentement mais sûrement. Même s'il tente de taper le 29.02.85, une date correcte sera codée dans la variable entière DATE et pourra donc plus facilement être comparée aux autres dates. Le sous-programme situé à partir de la ligne 62 000 remet une date codée dans le format d'entrée. Le format d'entrée est : JJ.MM.AA

```
5 ' ***  Date   CPC 464  ***
7 '
10 DIM table(12):FOR mois=1 TO 12:READ table(mois):NEXT mois
20 DATA 31,29,31,30,31,30,31,31,30,31,30,31
25 '
30 REM *** aPPel du Programme PrinciPal ***
35 '
40 INPUT "Date :";date$:GOSUB 60000:d(1)=date
50 PRINT "Code de la date entree :";d(1)
60 PRINT "transformation en format normal :";:date=d(1):GOSUB 62000:PRINT date$
70 GOTO 30
80 '
60000 REM *** Test de validitee ***
60010 '
60020 p(1)=INSTR(date$,"."):p(2)=INSTR(p(1)+1,date$,"."):l=LEN(date$)
60030 IF l<6 OR l>10 OR p(1) < 2 OR p(1) > 3 OR p(2) < 4 OR p(2) > 6 THEN 61000
60040 d(1)=VAL(LEFT$(date$,p(1)))
60050 d(2)=VAL(MID$(date$,p(1)+1,p(2)-p(1)))
60060 d(3)=VAL(RIGHT$(date$,l-p(2))):IF d(3) >=1900 THEN d(3)=d(3)-1900
60070 '
60080 IF d(2)<0 OR d(2) > 12 OR d(3) < 0 OR d(3) > 99 THEN 61000
60090 IF d(1) < 1 OR d(1) > table (d(2)) OR d(3) MOD 4 <> 0 AND d(1)=29 AND d(2)
=2 THEN 61000
60100 '
60110 REM *** Entree correcte ***
60120 REM *** Transformation chaine de caracteres en chiffres ***
60130 '
60140 date=d(3) * 366+d(1)
60150 FOR mois = 1 TO d(2)-1:date = date + table (mois):NEXT mois
60160 RETURN
60170 '
61000 REM *** Mauvaise entree ***
61010 '
61020 INPUT "Entree incorrecte, recommencez :";date$:GOTO 60000
61030 '
62000 REM *** Transformation chiffres en chaine de caracteres ***
62010 '
62020 d(3)=INT((date-1)/366):date = date-d(3)*366
62030 FOR mois = 1 TO 12:d(2) = mois: IF date > table (mois) THEN date = date -
table (mois):NEXT
62040 d(1)=date
62050 '
62060 date$ = DEC$ ((d(1),"##")+"."+DEC$((d(2),"##")+"."+DEC$((d(3),"##")
62070 FOR l= 1 TO 7 STEP 3
62080 IF MID$(date$,l,1)=" " THEN MID$(date$,l,1)="0"
62090 NEXT l
62100 RETURN
```

LES MODES 0,1 ET 2 EN MEME TEMPS

Le CPC 464 offre le choix d'afficher 20, 40 ou 80 caractères par ligne. C'est très pratique, mais il est possible de faire encore mieux. Une simple modification du système d'exploitation vous permettra de souligner des textes comme avec l'imprimante et d'afficher simultanément deux tailles de caractère différentes. Vous pourrez alors présenter votre écran comme vous le voulez.

Pour souligner des caractères ou en changer la taille il faut exécuter les séquences d'ordres suivantes :

PRINT#8,CHR$(27);CHR$(45);

CHR$(N) et

PRINT#8,CHR$(27);CHR$(87);

CHR$(N)

Si N est impair, la fonction est en marche, si N est pair, elle est arrêtée.
Après avoir tapé le chargeur BASIC et l'avoir mis en route, vous pourrez user de ces codes à l'écran comme sur l'imprimante. Le programme de démonstration qui suit les lignes de DATA montre l'utilisation et l'effet de cette routine.

Pour des raisons d'ordre pratique, le programme utilise les caractères de codes 254 et 255 pour y inscrire la matrice modifiée. Vous devez malheureusement vous passer des deux caractères préprogrammés qui se trouvent habituellement à cette place. S'il avait fallu éviter cela, le programme aurait été beaucoup plus long.

Le CPC 464 ne tient pas compte de l'instruction SYMBOL AFTER lorsque l'adresse haute de la mémoire a été changée par l'instruction MEMORY. (Message : "IMPROPER ARGUMENT").

Si vous désirez utiliser le programme avec des caractères redéfinis, il faudra utiliser l'instruction SYMBOL AFTER avant de mettre en route le chargeur BASIC.

Le programme en langage machine n'est pas relogeable mais il calcule bien entendu les adresses absolues indépendamment de HIMEM et il conviendra à n'importe quelle configuration de mémoire. Il faudra bien entendu le transformer et le mettre en route comme indiqué sur le listing.

```
10 REM *** CHARGEUR BASIC CPC 464 ***
20 '
30 MEMORY HIMEM-140
40 FOR adr=HIMEM+1 TO HIMEM+140:READ byte:POKE adr,byte:s=s+byte:NEXT
50 IF s<>15422 THEN MEMORY HIMEM+140:PRINT "erreur en data !":END
60 CALL HIMEM+1
70 DATA 33,28,0,25,34,21,179,62
80 DATA 2,50,20,179,17,16,0,25
90 DATA 34,218,189,62,255,205,165,187
100 DATA 34,191,178,201,95,35,126,33
110 DATA 193,178,254,45,40,4,254,87
120 DATA 192,35,115,201,254,32,218,12
130 DATA 20,95,58,184,178,183,123,194
140 DATA 12,20,58,193,178,31,48,17
150 DATA 123,205,165,187,237,91,191,178
160 DATA 1,7,0,237,176,62,255,18
170 DATA 95,58,194,178,31,123,210,12
180 DATA 20,205,165,187,221,42,191,178
190 DATA 6,8,197,78,6,4,203,9
200 DATA 31,203,47,16,249,221,119,0
210 DATA 6,4,203,9,31,203,47,16
220 DATA 249,221,119,248,221,35,35,193
230 DATA 16,224,62,254,205,12,20,62
240 DATA 255,195,12,20
250 '
260 REM ********* DEMO *********
270 '
280 a9on$=CHR$(27)+CHR$(87)+CHR$(1):REM Mode a9randissement.
290 a9of$=CHR$(27)+CHR$(87)+CHR$(0):REM Arret du mode a9randissement
300 slon$=CHR$(27)+CHR$(45)+CHR$(1):REM Mode souli9nement
310 slof$=CHR$(27)+CHR$(45)+CHR$(0):REM Arret du mode souli9nement
320 MODE 2:INK 1,0:INK 0,12
330 LOCATE 2,4:PRINT "normal"
340 LOCATE 10,8:PRINT slon$;"Souli9ne";slof$
350 LOCATE 18,12:PRINT a9on$;"A9randissement";
360 LOCATE 26,16:PRINT slon$;"Et les deux essembles";slof$;a9of$
370 LOCATE 1,20
```

HARDCOPY GRAPHIQUE POUR VOTRE CPC 464, 664 OU 6128 ET LA DMP1

Voici le listing assembleur et le chargeur basic d'une routine de hardcopy graphique.
Ce listing a été écrit avec le logiciel "DAMS".
Si vous ne possédez pas d'assembleur vous pouvez taper le programme BASIC correspondant. Dans ce cas votre routine de hardcopy est fixée à l'adresse & A000.
Si vous possédez un assembleur, vous pouvez changer le "ORG" et assembler la routine à l'adresse que vous désirez.
Pour utiliser cette routine à partir du BASIC, il vous suffira de taper : "CALL &A000".

FONCTIONNEMENT :

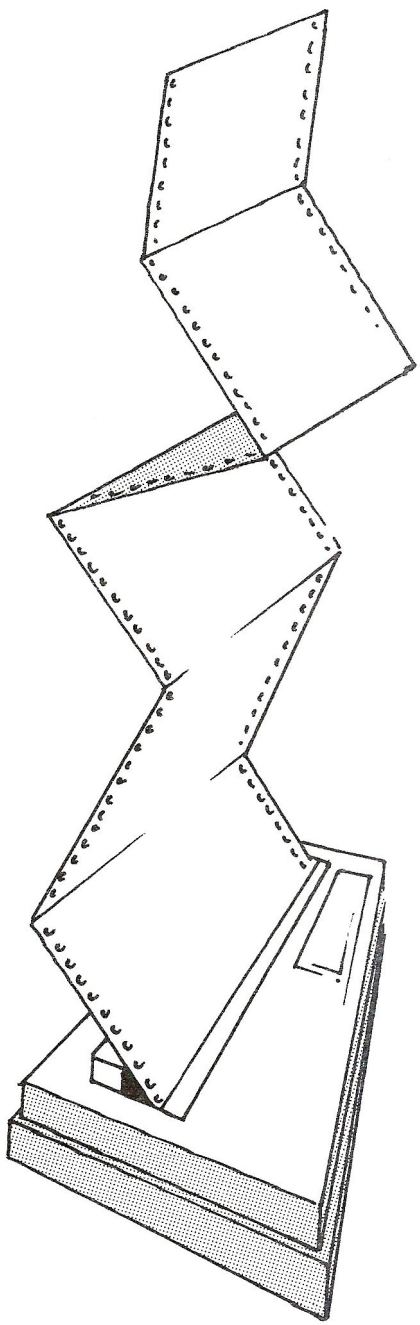
Cette routine est faite pour fonctionner sur l'imprimante DMP1 (AMSTRAD). Le mode graphique est activé dans la routine "INITLP" par la séquence ESC,K,H,L où H et L représentent l'octet fort et l'octet faible du nombre de caractères graphiques qui suivent.

Vous vous demandez peut-être comment on peut imprimer les 640 points de l'écran AMSTRAD sur les 480 que permet l'imprimante.
Pour détourner ce problème on imprime les 640 points sur la longueur du papier, et les 200 points de résolution verticale, sur la largeur. Pour éviter d'avoir une image trop aplatie, on double chaque point imprimé horizontalement (sur le papier) et le tour est joué.
On obtient sur une page une image de 640 points sur 400 qui a à peu près les mêmes proportions que l'écran.

Quant au problème de la sortie parallèle sur sept bits, il est résolu d'une part, en envoyant les données graphiques par groupe de cinquante au lieu d'en envoyer 400 d'un coup, d'autre part en communiquant les données graphiques sur sept bits.

Ceci implique que l'imprimante doit faire automatiquement la DMP1 dès qu'elle est en mode graphique.
Le fait de ne disposer que de sept bits pose encore un problème : 640 n'étant pas un multiple de sept, la dernière ligne imprimée aura donc 3 pixels de hauteur.
Ce cas est traité de façon particulière par la routine HORIZL1.

(programme)



```
        DEC  A
        JR   NZ,BLOCL1              ;Recommence 50 fois.
        RET
;
RDBYT
        PUSH BC                     ;Sauve le nombre de Pixels Par tranche.
        LD   A,8
        SUB  B                      ;Calcule le decala9e.
        PUSH DE                     ;Sauve9arde coord. debut de colonne.
        PUSH AF                     ;Sauve9arde la correction de decala9e.
RDBYTL1
        PUSH DE                     ;Sauve9arde coord. x, colonne.
        PUSH HL                     ;Sauve9arde coord. y, li9ne.
        PUSH BC                     ;Sauve9arde le nombre de Pixel a lire.
        CALL TSTPOINT               ;Teste le Pixel a l'ecran.
        POP  BC
        LD   HL,PAPER
        CP   (HL)                   ;Pixel allume ?
        SCF
        POP  HL
        POP  DE
        JR   NZ,RDBYTP1
        OR   A
RDBYTP1 RR   C                      ;Inclue le Pixel dans C.
        INC  DE                     ;Passe a la colonne de Pixel suivante.
        DJNZ RDBYTL1                ;Lit tous les Points.
;
        POP  AF                     ;Retire la correction a aPPli9uér.
        LD   B,A
RDBYTL2 SRL  C                      ;Corri9e le decala9e,
        DJNZ RDBYTL2
        LD   A,C                    ;  et Passe le resultat a A.
        POP  DE                     ;Restaure coord. de debut de colonne.
        POP  BC                     ;Restaure le Nbr. de Pixels / colonne.
        INC  HL                     ;Et Passe a la Li9ne
        INC  HL                     ;  de Pixels au dessus.
        RET
;
INITLP
        LD   A,27                   ;Code escaPe.
        CALL PRINT
        LD   A,75                   ;Si9nal: mode 9raPhi9ue (Pour DMP1).
        CALL PRINT
        XOR  A
        CALL PRINT
        LD   A,100                  ;100 caracteres 9raPh. / blocs.
;
PRINT
        PUSH HL
PRINTL1
        CALL TSTLPR                 ;ImPrimante busy ?
        JR   C,PRINTL1              ;Si oui, attend.
        CALL PRINTOUT               ;ImPrime un caractere.
        POP  HL
        RET
;
TSBREAK
        LD   A,66                   ;Numero de la touche escaPe.
        PUSH DE
        CALL TSTKEY                 ;Teste la touche.
        POP  DE
        RET
;
PAPER   DEFB
```

```
.COPYRIGHT 1985 MICRO-APPLICATION.
.DAMS.

;
; HardcoPy 9raPhi9ue  640 x 200
;
;
        ORG  #A000
;
GRINIT  EQU  #BBBA
GETPAP  EQU  #BBE7
TSTPOINT EQU #BBF0
PRINTOUT EQU #BD2B
TSTLPR  EQU  #BD2E
TSTKEY  EQU  #BB1E

        CALL GRINIT                 ;Active le mode 9raPhi9ue.
        CALL GETPAP                 ;Determine la couleur du PaPier
        LD   (PAPER),A              ;  et stocke la.
        LD   A,10                   ;Vide le buffer de l'imPrimante.
        CALL PRINT

        LD   DE,0                   ;Colonne de Pixels de 9auche.
HORIZL1
        LD   B,7                    ;Si9nal: 7 Pixels de lar9eur.
        CALL VERTIC                 ;HardcoPy d'une colonne.
        CALL TSBREAK                ;EscaPe aPPuyee ?
        RET  NZ                     ;Si oui, retour au basic.
        LD   HL,7                   ;7 Pixels Par colonne.
        ADD  HL,DE                  ;Coordonnees de la colonne suivante
        EX   DE,HL                  ;  dans DE.
        LD   HL,636
        AND  A
        SBC  HL,DE                  ;Est-ce la derniere colonne ?
        JR   NC,HORIZL1             ;Non, alors recommence.

        LD   B,3                    ;Si9nal: 3 Pixels de lar9eur Pour
        CALL VERTIC                 ;  le dernier hardcoPy de colonne.
        RET                         ;Fini.
;
VERTIC
        LD   HL,1                   ;Coord. x, Li9ne de Pixels du bas.
        LD   A,4                    ;4 blocs / colonne.
VERTICL1
        PUSH AF
        CALL BLOC                   ;HardcoPy d'un bloc.
        POP  AF
        DEC  A
        JR   NZ,VERTICL1            ;Recommence 4 fois.
        LD   A,10                   ;ImPrime une colonne d'ecran sur  1
        JR   PRINT                  ;  li9ne d'imPrimante.
;
BLOC
        CALL INITLP                 ;PrePare l'imPrimante  9ra9hi9ue.
        LD   A,50                   ;50 Pixels / blocs. (50*4=200).
BLOCL1
        PUSH AF
        CALL RDBYT                  ;Lis 1 octet sur l'ecran.
        PUSH AF
        CALL PRINT                  ;Envoie-le a l'imPrimante
        POP  AF                     ;  2 fois, Pour corri9er
        CALL PRINT                  ;  le raPPort vertic. / horiz.
        POP  AF
```

```
50 '  HardcoPy 9raPhi9ue Pour CPC et DMP1
60 '   Le hardcoPy doit etre aPPele Par
70 '          CALL &A000
80 '
100 DATA CD,BA,BB,CD,E7,BB,32,9F,A0,3E,0A,CD,8C,A0,11,00
110 DATA 00,06,07,CD,2D,A0,CD,97,A0,C0,21,07,00,19,EB,21
120 DATA 7C,02,A7,ED,52,30,EA,06,03,CD,2D,A0,C9,21,01,00
130 DATA 3E,04,F5,CD,3E,A0,F1,3D,20,F8,3E,0A,18,4E,CD,7C
140 DATA A0,3E,32,F5,CD,54,A0,F5,CD,8C,A0,F1,CD,8C,A0,F1
150 DATA 3D,20,F0,C9,C5,3E,08,90,D5,F5,D5,E5,C5,CD,F0,BB
160 DATA C1,21,9F,A0,BE,37,E1,D1,20,01,B7,CB,19,13,10,EA
170 DATA F1,47,CB,39,10,FC,79,D1,C1,23,23,C9,3E,1B,CD,8C
180 DATA A0,3E,4B,CD,8C,A0,AF,CD,8C,A0,3E,64,E5,CD,2E,BD
190 DATA 38,FB,CD,2B,BD,E1,C9,3E,42,D5,CD,1E,BB,D1,C9,00
200 '
210 MEMORY  40959
220 s=0
230 FOR i= 40960 TO 41119
240 READ b$:b=VAL("&"+b$):POKE i,b:s=s+b
250 NEXT
260 IF s<> 21037 THEN PRINT "erreur en DATA":END
```

TOUJOURS PLUS DE TOUCHES DE FONCTION

Le pavé de touches numériques ne peut pas être redéfini et continuer parallèlement à fonctionner normalement. C'est dommage, mais en fait, pas impossible !

Nous vous offrons une solution.

Les fonctions définies par l'ordre KEY sur les touches chiffrées sont appelées par SHIFT + chiffre et CTRL + chiffre. Si ces 20 fonctions supplémentaires ne vous suffisent pas, la plupart des autres touches peuvent être redéfinies de la même manière et utilisées avec la touche CTRL. Le truc réside dans le fait que le système d'exploitation efface le bit 7 de l'argument qui suit l'ordre KEY.

```
10 '-----------------------------------
20 ' Fonctions
30 '-----------------------------------
40 FOR k=128 TO 159 : KEY k,"" :NEXT
50 ' ***  Definition dans 3 secteurs  ***
60 KEY DEF 15,0,48,128,138  :KEY DEF 7,0,46,148,149
70 KEY DEF 6,0,13,150,151   :KEY DEF 13,0,49,129,139
80 KEY DEF 14,0,50,130,140  :KEY DEF 5,0,51,131,141
90 KEY DEF 20,0,52,132,142  :KEY DEF 12,0,53,133,143
100 KEY DEF 4,0,54,134,144   :KEY DEF 10,0,55,135,145
110 KEY DEF 11,0,56,136,146  :KEY DEF 3,0,57,137,147
120 ' ***  Touches de fonction  (modulo 128)  ***
130 KEY 0,"cls"+CHR$(13)     :KEY  10,""
140 KEY 1,"run"+CHR$(13)     :KEY  11,""
150 KEY 2,"list "            :KEY  12,""
160 KEY 3,"renum "           :KEY  13,""
170 KEY 4,"&"                :KEY  14,""
180 KEY 5,"&x"               :KEY  15,""
190 KEY 6,"mode 2"+CHR$(13) :KEY  16,""
200 KEY 7,"run"+CHR$(34)     :KEY  17,"load "+CHR$(34)
210 KEY 8,"goto "            :KEY  18,"cat"+CHR$(13)
220 KEY 9,""                 :KEY  19,"save "+CHR$(34)
230 ' ***  Touche virgule  ***
240 KEY 20,"border 0:ink 0,0:ink 1,26"+CHR$(13):REM  SHIFT
250 KEY 21,"" :REM CTRL
260 ' ***  Petite touche enter  ***
270 KEY 22,"border 26:ink 0,26:ink 1,0"+CHR$(13):REM SHIFT
280 KEY 23,"run "+CHR$(34)+CHR$(13) :REM CTRL
```

DECODEUR D'ENREGISTREMENT

Avec ce programme, il vous suffira d'introduire une cassette dans le magnétophone et de l'enrouler à n'importe quel endroit pour obtenir toutes les informations importantes sur ce qu'elle contient, comme : le nom de fichier, le type de fichier, l'adresse de départ d'un programme en langage machine, etc...

Et tout cela apparaîtra à l'écran.

```
10 ' ***  DECODEUR D'ENTETE DE CASSETTE ***
20 '
30 MODE 2
40 MEMORY &9FFF
50 FOR i=&A000 TO &A00B
60 READ w:POKE i,w
70 NEXT
80 DATA &21,&10,&aa,&11,&40,&00,&3e,&2c,&cd,&a1,&bc,&c9
90 PRINT "Press PLAY then any key:":CALL &BB06
100 CALL &A000
110 CLS
120 PRINT "nom du fichier     : ";
130 FOR i=&AA10 TO &AA1F:PRINT CHR$(PEEK(i));:NEXT
140 LOCATE 1,3:PRINT "type de fichier  : ";
150 typ$ = BIN$(PEEK(&AA22))
160 IF RIGHT$(typ$,1)="1" THEN ls=1 ELSE ls=0
170 typ$="&x"+typ$:typ=VAL(typ$):IF ls THEN typ=typ-1
180 IF typ=0 THEN ftyp$="Basic" ELSE IF typ=2 THEN ftyp$="Binaire" ELSE ftyp$="A
SCII"
190 PRINT ftyp$
200 LOCATE 1,5:PRINT "Protection    : ";
210 IF ls =1 THEN PRINT "oui" ELSE PRINT "non"
220 LOCATE 1,7:PRINT "Premier bloc    : ";
230 IF PEEK(&AA27)=0 THEN PRINT "non" ELSE PRINT "oui"
240 LOCATE 1,9:PRINT "Bloc numero   : ";
250 PRINT PEEK(&AA20)
260 LOCATE 1,11:PRINT "Dernier bloc   : ";
270 IF PEEK(&AA21)=0 THEN PRINT "non" ELSE PRINT "oui"
280 LOCATE 1,13:PRINT "Longueur du bloc   : ";
290 bl$=HEX$(PEEK(&AA24),2)+HEX$(PEEK(&AA23)):PRINT bl$
300 LOCATE 1,15:PRINT "Longueur du fichier   : ";
310 dl$=HEX$(PEEK(&AA29),2)+HEX$(PEEK(&AA28)):PRINT dl$
320 LOCATE 1,17:PRINT "Adresse de chargement  : ";
330 ra$=HEX$(PEEK(&AA26),2)+HEX$(PEEK(&AA25)):PRINT ra$
340 LOCATE 1,19:PRINT "Adresse d'execution  : ";
350 jp$=HEX$(PEEK(&AA2B),2)+HEX$(PEEK(&AA2A),2):PRINT jp$
360 '
370 ***  Listing assembleur  ***
380 '
390 ' A000  21 10 AA   LD HL,#AA10
400 ' A003  11 40 00   LD DE,#0040
410 ' A006  3E 2C      LD A,#2C
420 ' A008  CD A1 BC   CALL #BCA1
430 ' A00B  C9         RET
```

MICRO

TUEUR DES ESPACES DE REM

Nous pensons que les espaces dans un programme qui tourne sont une perte de place. Cela est vrai également pour certains commentaires. Cependant, l'explication des parties de programme pouvant être utiles même dans des programmes qui fonctionnent, le "récupérateur de place en BASIC" différencie deux types de commentaires :

REM - Tous les commentaires restent.

' - Tous les commentaires précédés par ' sont effacés.

L'adresse de départ en ligne 1030 peut être changée. Si vous utilisez une disquette, la valeur devra être inférieure à &AB00.
Pour utiliser le tueur de REM :
CALL &AB00 ou CALL l'adresse donnée en ligne 1030.

```
1000 REM ********************************************
1010 REM **     Tueur de REM   CPC 464               **
1020 REM ********************************************
1030 address=&AB00' inscrire ici l'adresse voulue
1040 control=0' Variable de controle des datas
1050 MEMORY address-1
1060 FOR i=0 TO 78:READ t$:control=control+VAL("&"+t$):NEXT
1070 IF control<>-12439 THEN PRINT"Erreur en data !":END
1080 RESTORE:a=address
1090 FOR i=0 TO 70
1100 READ t$:content=VAL("&"+t$)
1110 IF LOWER$(t$)<>"a"THEN POKE a,content:GOTO 1160
1120 READ t$:content=address+VAL("&"+t$)
1130 IF content<0 THEN content=content+65536
1140 POKE a,content-256*FIX(content/256)
1150 a=a+1:POKE a,FIX(content/256)
1160 a=a+1
1170 NEXT i
1180 READ t$:a=VAL("&"+t$):READ t$:POKE a,VAL("&"+t$)
1190 READ t$:content=address+VAL("&"+t$)
1200 IF content<0 THEN content=content+65536
1210 a=a+1:POKE a,content-256*FIX(content/256)
1220 a=a+1:POKE a,FIX(content/256)
1230 DATA 3E,01,32,00,AC,2A,81,AE,23,18,14,E5,DF,a,43,DF,a,46
1240 DATA 21,A4,AC,DF,a,49,E1,CD,a,31,28,06,09,CD,a,31,20,E7,AF
1250 DATA 32,00,AC,2A,75,AE,77,23,77,23,77,C9,4E,23,46,2B,78,B1
1260 DATA C9,3A,00,AC,B7,C8,7E,D6,27,C0,77,C9,63,E1,FD,BC,E6,FD
1270 DATA CB,DD,FD,AC10,C3,38
```

DES GO TO ET GO SUB AVEC DES EXPRESSIONS

KARSTEN BUCKING devait penser "Les CPC ne doivent pas être en reste" et il a créé une routine de GOTO/GOSUB avec expressions.
La routine prend le message SYNTAX ERROR à l'adresse &AC04, teste si l'erreur vient d'une ligne GOTO ou GOSUB puis elle retrouve le numéro de ligne et utilise l'ordre CALL pour calculer l'expression.
Puis l'adresse cible est recherchée avec la routine de l'ordre GOTO et on revient à l'interpréteur.
Si la ligne calculée n'existe pas, le message d'erreur habituel apparaît.
Encore une remarque : l'adresse de chargement doit être donnée en décimal.

```
100 '-----------------------------------------
110 ' ***  GOTO avec des expressions  ***
120 ' ***     Pour CPC 464           ***
130 '-----------------------------------------
140 '
150 INPUT "Adresse de chargement :";adr
160 hi=INT(adr/256)
170 MEMORY adr-1
180 FOR x=0 TO 43:READ a$:v=VAL("&"+a$):s=s+v:POKE adr+x,v:NEXT
190 IF s <> 6602 THEN PRINT "Erreur en data !":END
200 POKE &AC04,&C3
210 POKE &AC05,adr-256*hi:POKE &AC06,hi
220 DATA C1,C1,E5,2B,2B,2B,2B,2B
230 DATA 7E,FE,9F,28,10,FE,A0,C2
240 DATA 97,CA,23,23,CD,91,CE,CD
250 DATA 9A,E7,C1,2B,C9,23,23,CD
260 DATA 91,CE,CD,9A,E7,EB,E1,CD
270 DATA F0,C6,2B,C9
```

DES FONCTIONS D'EDITION INCONNUES

On a découvert sur le CPC quelques fonctions d'édition très utiles qui ne se trouvent pas dans le manuel.

CTRL + <curseur haut> :
curseur au début de la ligne entrée

CTRL + <curseur bas> :
curseur à la fin de la ligne entrée

CTRL + TAB :
mise en route ou arrêt du mode insertion de l'éditeur.

Comme indiqué dans le manuel, les combinaisons de touches CTRL + <curseur gauche> et CTRL + <curseur droit> mettent le curseur au début ou à la fin de la ligne d'écran. Bien évidemment, la différence avec les fonctions que nous vous proposons n'apparait que pour des lignes dont la taille excède celle de l'écran.

SCROLL POUR COMMODORE 128

Bien que le BASIC 7.0 du COMMODORE 128 soit musclé, il ne possède pas d'instruction de défilement partiel d'écran (scroll en anglais, mais essayons de rester français), qui pourtant peut être fort utile.
Nous allons donc les créer.

Prenez une instruction "window", fort bien faite, ajoutez-y des touches programmées par COMMODORE et vous obtiendrez ce que vous cherchiez. Voici la marche exacte à suivre : vous définissez une fenêtre, avec l'instruction window, qui recouvrira toute la zone que vous désirez faire défiler.

Ceci fait vous n'avez plus qu'à y inscrire vos variables, qui vont défiler sur la partie de l'écran que vous venez de définir. L'instruction PRINT se charge de faire défiler l'écran.
Souvenez-vous maintenant qu'en mode direct, en pressant la touche "esc" et "w" ou "esc" et "v", vous pouvez faire défiler l'écran de haut en bas et de bas en haut. En mode programme vous obtiendrez le même résultat avec PRINT CHR$ (27)+ "w" ou PRINT CHR$ (27)+ "v". Ecrivez la nouvelle variable dans la place libérée.

Tout cela peut paraître fort compliqué, alors essayons d'examiner un programme qui le fait.
Pour l'exemple nous allons travailler avec 20 variables A$(. Il faut donc dimensionner à 21.

programme ligne 1120-1300 :

Puisqu'il y a 2 écrans possibles (40 et 80 colonnes) nous allons demander sur quel écran il faut travailler.

programme ligne 1310-1370 :

Voilà 20 variables qui ont été créées et définies :
A$ (1) = "A A A A A A A A A A"

A$ (2) = "B B B B B B B B B B"
etc...

programme ligne 1390-1620 :

Pour rendre la démonstration plus spectaculaire, entourez la zone qui devra défiler d'un rectangle.

Remarquez que ce que nous avons programmé jusqu'à maintenant, à part la ligne 1100, n'est pas nécessaire dans vos programmes, mais sert ici pour la démonstration.
Passons aux choses sérieuses.

programme ligne 1630-1700 :

Une fenêtre a été définie et dans laquelle sont affichées les variables. Une variable REP donnera toujours l'indice de la variable se trouvant en dernière ligne.

programme ligne 1710-1760 :

On attend la pression sur une touche avec la boucle DO... LOOP et GET A$

programme ligne 1770-1870 :

Si la touche pressée est "curseur haut", la variable REP va permettre d'effectuer ou non le scroll.
Si elle vaut 10 la variable A$(1) a été affichée ; donc on n'effectue rien.

Dans les autres cas, on va définir la fenêtre, décrémenter REP et récrire la "formule magique" qui fera descendre l'écran. Il ne reste plus qu'à écrire la nouvelle variable dans la place laissée libre et à sortir de la fenêtre.

programme ligne 1880-1970 :

Si la touche pressée est "curseur bas" il faut vérifier que la variable affichée sur la dernière ligne de la fenêtre n'est pas la dernière variable. Dans ce cas REP est incrémentée.

programme ligne 1980-2010 :

Si la touche pressée est "F" alors on sort de la boucle.

programme ligne 2020-2030 :

Dans les autres cas, ou si aucune touche n'a été pressée, on revient au départ de la boucle.

programme ligne 2040-2140 :

Et voilà, tous les secrets viennent de vous être révélés, à vous d'utiliser ce truc à bon escient.

```
1000 REM ------------------------------------------------
1010 REM                    SCROLL 128
1020 REM
1030 REM        FONCTION SCROLL POUR COMMODORE 128
1040 REM
1050 REM             PAR PROG 64  OCTOBRE 85
1060 REM ------------------------------------------------
1070 :
1080 REM --- INITIALISATION ET CHOIX ECRAN--------------
1090 :
1100 CLR:DIMA$(21)
1110 :
1120 GRAPHIC0,1
1130 CHAR,8,10,"VOULEZ-VOUS L´EXEMPLE SUR"
1140 CHAR,10,12,"1- ECRAN 40 COLONNES"
1150 CHAR,10,14,"2- ECRAN 80 COLONNES"
1160 :
1170 DO UNTIL A$="1" OR A$="2"
1180 : GETA$
1190 LOOP
1200 :
1210 :IF A$="1" THEN BEGIN
1220 :  GRAPHIC0,1
1230 :  CHAR,4,2,"EXEMPLE DE SCROLL SUR 40 COLONNES",1
1240 :BEND
1250 :
1260 :IF A$="2" THEN BEGIN
1270 :  GRAPHIC5,1
1280 :  CHAR,24,2,"EXEMPLE DE SCROLL SUR 80 COLONNES",1
1290 :BEND
1300 :
1310 REM --- CREATION DE CHAINES DE CARACTERES ---------
1320 :
1330 FOR A=1 TO 20
1340 : FOR B=1 TO 10
1350 :   A$(A)=A$(A)+CHR$(64+A)
1360 : NEXT B
1370 NEXT A
1380 :
1390 REM --- DESSINE UN CARRE OU L´ON VA SCROLLER ------
1400 :
1410 CHAR,9,4,CHR$(176)
1420 :A=0
1430 : DO UNTIL A=11
1440 :   PRINT CHR$(192);
1450 :   A=A+1
1460 : LOOP
1470 PRINT CHR$(174)
1480 :
1490 :A=0
1500 : DO UNTIL A=11
1510 :   CHAR,9,5+A,CHR$(221)+"           "+CHR$(221)
1520 :   A=A+1
1530 : LOOP
1540 :
1550 CHAR,9,16,CHR$(173)
1560 :A=0
1570 : DO UNTIL A=11
1580 :   PRINTCHR$(192);
1590 :   A=A+1
1600 : LOOP
1610 PRINTCHR$(189)
1620 :
1630 REM --- AFFICHE LES CHAINES AVEC SCROLL -----------
1640 :
1650 WINDOW 10,5,20,15
1660 : FOR A=1 TO 20
1670 :  PRINTA$(A)
1680 : NEXT
1690 :REP=20
1700 PRINT CHR$(19);CHR$(19)
1710 :
1720 REM --- SCROLL SELON TOUCHE CURSEUR ---------------
1730 :
1740 DO
1750 : GETA$
1760 :
1770 REM --- SCROLL  BAS -------------------------------
1780 :
1790 :  IF A$=CHR$(145)THEN BEGIN
1800 :   IF REP<11 THEN 1860
1810 :     WINDOW 10,5,20,15
1820 :     REP=REP-1
1830 :     PRINTCHR$(27)+"W"
1840 :     CHAR,0,0,A$(REP-9)
1850 :     PRINTCHR$(19);CHR$(19)
1860 :  BEND
1870 :
1880 REM --- SCROLL  HAUT ------------------------------
1890 :
1900 :  IF A$=CHR$(17) THEN BEGIN
1910 :   IF REP>19 THEN 1970
1920 :     WINDOW 10,5,20,15
1930 :     REP=REP+1
1940 :     PRINTCHR$(27)+"V"
1950 :     CHAR,0,10,A$(REP+1)
1960 :    PRINTCHR$(19);CHR$(19)
1970 :  BEND
1980 :
1990 REM --- POUR SORTIR DE LA BOUCLE PRESSEZ F --------
2000 :
2010 : IF A$="F" THEN EXIT
2020 LOOP
2030 :
2040 SCNCLR
2050 :
2060 END
2070 :
2080 REM --- SAUVEGARDE -------------------------------
2090 :
2100 SCRATCH"SCROLL 128 (1-0)"
2110 DSAVE"SCROLL 128 (1-0)"
2120 DVERIFY"SCROLL 128 (1-0)"
2130 :
2140 REM --- C´EST FINI -------------------------------

READY.
```

LES INSTRUCTIONS TRAP ET RESUME

INTRODUCTION

Le nouveau COMMODORE 128 possède deux instructions basic qui peuvent sembler, au premier abord, d'une utilité restreinte. Mais avant tout, regardons-les d'un peu plus près. Par la même occasion après avoir étudié leur synthèse, nous jetterons un coup d'oeil sur quelques variables réservées qui peuvent nous aider lors de l'établissement d'un programme.

TRAP

Cette instruction se place généralement en tête de programme. Sa syntaxe est la suivante :

TRAP nl : en cas d'interruption le programme continue en ligne nl.

RESUME

Retour sur une ligne précise du programme, après le traitement d'une interruption. Cette instruction est liée à TRAP. Sa syntaxe est la suivante :

RESUME : le programme reprendra à la ligne où il a été interrompu.

RESUME NEXT : le programme reprendra à la ligne suivante.

RESUME nl : le programme reprend au numéro de ligne nl.

Et voici maintenant les variables réservées :

el : numéro de ligne où l'interruption s'est produite
er : numéro d'erreur qui a produit l'interruption.
ERR$(): associée à er, donne l'énnoncé de l'erreur.

Pour finir ce tour d'horizon, tapez la ligne suivante :

```
FOR I=1 TO 41 : PRINT I;" -
";ERR$(I) :NEXT
```

Et vous obtiendrez la liste de tous les messages d'erreur du BASIC 7.0.

Une restriction est à signaler : les interruptions de programme se produisant dans la partie, où le programme a été dirigé par TRAP, (dans notre exemple les lignes 1830 à 2360) ne sont pas traitées, et le programme s'arrête.

UTILISATION

Elles peuvent, par exemple, vous aider dans les situations suivantes :
- corriger les erreurs dans un programme.

- indiquer qu'un fichier est plein.

- empêcher l'arrêt accidentel d'un programme.

- contrôler la présence d'un périphérique.

- vérifier la bonne sauvegarde d'un programme.

Prenons ces cas les uns après les autres :

Mais d'abord, n'oublions pas de placer TRAP en début de programme, pour indiquer à l'ordinateur quelle est la ligne où il doit se rendre :

```
1130 TRAP 1830
```

CORRECTION

La première façon la plus logique de l'employer est la suivante :
l'utiliser pour la correction de nos propres erreurs, ou celles des lecteurs qui recopient le programme à partir d'une revue. Il est conseillé de la mettre en fin de programme, comme dans l'exemple.

Un deuxième conseil : avant de commencer à taper vos programmes, écrivez ces lignes avec comme numéro de ligne 63000, par exemple.

Le listing d'une telle opération pourrait se présenter ainsi :

programme ligne 2250-2360 :

2250 : si vous travaillez sur l'écran 40 caractères, prenez la solution 1. Pour l'écran 80 caractères, choisissez la solution qui se trouve après le REM.

Si vous travaillez sur l'un des écrans graphiques, employez toujours GRAPHIC0,1, car en cas d'arrêt du programme dû à une erreur le message s'inscrira sur l'écran 40, mais vous verrez toujours l'écran graphique sur lequel vous travaillez.

2270 : affiche le numéro de l'erreur (ER) et le texte de l'erreur ERR$(ER) qui lui est associé.

2280 : a le même effet que de presser la touche HELP.

2300 : début d'une boucle qui ne s'arrêtera pas avant qu'une touche ne soit pressée.

2310 : produit un signal sonore.

2320 : regarde si une touche vient d'être pressée. Si aucune touche ne l'a été, la boucle continue.

2330 : l'ordinateur fait une pause d'une seconde.

2340 : suite de la boucle.

2360 : fin du programme pour vous permettre de corriger votre erreur.

FICHIER PLEIN

Voici une utilisation pratique de RESUME. Regardons le listing : Voyons d'abord la partie de la saisie des noms :

programme ligne 1140-1430 :

Passons au traitement de l'erreur en lui-même :

programme ligne 1850-1940 :

1140 : dimensionnons notre variable à 2, pour pouvoir voir le résultat au plus tôt.

1380 : efface l'écran.

1390 : indique que nous sommes en saisie.

1400 : comme il y aura une variable de plus dans le tableau, incrémentons N de 1. Créons une variable A$(et donnons lui comme valeur un espace. Ceci est la partie la plus importante de la saisie. Car si l'on a dépassé le dimensionnement, (ici N>2) le programme va s'arrêter et produire l'erreur 18. Il sera alors aiguillé sur le traitement des erreurs, que nous verrons dans quelques lignes.

1410-1420 : si la valeur n'a pas été dépassée, la saisie peut se faire normalement.

1430 ... et l'on retourne au menu.

1850 : si le numéro de l'erreur est 18, ce qui est le cas, commençons le traitement de l'erreur.

1860 : on efface l'écran.

1870-1900 : le texte de votre choix est affiché.

1910 : il ne reste plus qu'à attendre que l'utilisateur du programme se manifeste en pressant une touche.

1920 : comme en ligne 1400, la variable N avait été incrémentée de un, enlevons-lui la même valeur, pour retrouver le nombre réel de variables A$(.

1930... et le programme reprend au menu.

1940 : il ne faut surtout pas oublier cette ligne, qui indique que c'est la fin de la condition commencée à la ligne 1850.

ARRET ACCIDENTEL

Un coup d'oeil au listing :

programme ligne 2150-2230 :

2150 : si le numéro de l'erreur est 30, ce qui arrive en pressant la touche STOP, le traitement commence.

2160 : on efface l'écran.

2170-2220 : il suffit d'afficher le texte de votre choix.

2210 : il ne reste plus qu'à attendre que l'utilisateur se manifeste...

2220... pour retourner au menu.

2230 : encore une fois n'oubliez pas d'indiquer à l'ordinateur que la condition commencée en ligne 2150 est terminée.

C'est simple, non ?

CONTROLE D'UN PERIPHERIQUE

C'est à mon avis, l'utilisation la plus intéressante. Il y a deux périphériques dont il serait intéressant de connaître la présence : le lecteur de disquette et l'imprimante.

Pour le lecteur de disquette l'opération est assez simple : lorsqu'il n'est pas enclenché le programme s'arrête, il suffit donc de traiter l'erreur.

programme 1960-2030 :

1960 : si l'interruption est due à l'erreur 5, on commence.

1970 : comme on utilise le lecteur de disquette en ligne 1470 à 1560, il suffit de contrôler que l'interruption s'est produite à une ligne comprise entre ces valeurs.

1980 : on ferme le canal que l'on vient d'ouvrir.

1990-2200 : mettez le texte de votre choix...

2210... et attendez la pression d'une touche...

2220... pour revenir au menu.

2230 : ne pas oublier cette ligne.

Pour l'imprimante, c'est par accident que j'ai trouvé la solution. Regardons la partie consacrée à l'impression :

programme ligne 1600-1690 :

1600 : l'écran est effacé.

1610 : indique que l'on est en impression.

1620 : c'est là que réside l'astuce : il faut ouvrir un fichier pour l'imprimante en indiquant une adresse secondaire et un nom de fichier, c'est dans ce cas-là que le programme s'arrête, si l'imprimante n'est pas connectée.

Voyons comment traiter l'erreur :

programme ligne 2040-2110 :

2040 : l'imprimante est utilisée à partir des lignes 1600 à 1690.

2060-2070 : à vous de choisir le texte.

2080 : une touche doit être pressée...

2090... pour commencer en ligne 1600.

2100 : fin de la condition commencée en 2040.

2110 : fin de la condition commencée en ligne 1960.

CONTROLE D'UN PROGRAMME

Il est très pratique d'avoir le programme et les données d'un fichier sur une seule disquette, d'où il est utile que le programme prépare une disquette et se sauve lui-même et enfin vérifie s'il est sauvé correctement. Ce qui donne :

programme ligne 1730-1780 :

1730 : l'écran est effacé.

1740 : nous sommes en préparation de disquette.

1750 : nous pouvons formater la disquette.

1760 : le programme est effacé s'il se trouve encore sur la disquette...

1770... sauvé ...

1780 et vérifié.

L'erreur est traitée de la manière suivante :

```
2130 IF ER=28 THEN
RESUME 1730
```

2130 : s'il y a une "VERIFY ERROR" le programme interrompu reprendra à la ligne 1730.

CONCLUSION

Et voilà quelques possibilités d'emploi de TRAP et RESUME. Je suis certain que vous ferez d'autres découvertes plus intéressantes encore. Soyez sympa, envoyez-les à Micro-Info pour que nous puissions tous en profiter.

Vous habitez en BELGIQUE, en SUISSE ou au CANADA et vous désirez acheter certains de nos produits ou vous abonner à MICRO INFO, n'hésitez pas à contacter nos importateurs dans chacun de ces pays :

EN BELGIQUE :

MICRO BELGIUM APPLICATION
Avenue de Tervuren, 327
BRUXELLES 1150

TEL : 02-762 72 02
770 65 50
TELEX : 20271 MBA

EN SUISSE :

HILCU INTERNATIONAL
CH-3063 ITTIGEN
BERN

TEL : 031-58 66 56
TELEX : 911356 HILCU

AU CANADA :

VISIONTRONIC SA
BUREAU SALLE DE MONTRE
218 ALEXANDRE
SHERBROOKE, Qué.

TEL : 819-569 7415

Pour recevoir plus de renseignements sur nos produits, remplissez le coupon ci-dessous et retournez le à l'adresse du pays qui vous concerne.

NOM :

PRENOM :

ADRESSE :

VILLE : CODE POSTAL :

MICRO INFO

```
1000 REM                    ***************************
1010 REM                    *  EXEMPLE POUR ILLUSTRER *
1020 REM                    *                         *
1030 REM                    *     TRAP ET RESUME      *
1040 REM                    *                         *
1050 REM                    *       PAR PROG 64       *
1060 REM                    *                         *
1070 REM                    *      DECEMBRE 1985      *
1080 REM                    *                         *
1090 REM                    ***************************
1100 :
1110 REM ------------------------ INITIALISATION --------------------------
1120 :
1130 TRAP 1830
1140 DIMA$(2)
1150 :
1160 REM ------------------------ MENU ------------------------------------
1170 :
1180 DO
1190 : GRAPHIC0,1
1200 : CHAR,5,1," EXEMPLE POUR TRAP ET RESUME ",1
1210 : CHAR,7,7,"1-SAISIE DE DONNEES"
1220 : CHAR,7,9,"2-SAUVEGARDE"
1230 : CHAR,7,11,"3-IMPRESSION"
1240 : CHAR,7,13,"4-PREPARE UNE DISQUETTE"
1250 : CHAR,7,15,"5-FIN"
1260 : GETKEY R$
1270 : R=VAL(R$)
1280 : IF R<1 OR R>5 THEN 1260
1290 : ON R GOSUB 1380,1470,1600,1730,1340
1300 LOOP
1310 :
1320 REM ------------------------ FIN -------------------------------------
1330 :
1340 END
1350 :
1360 REM ------------------------ SAISIE ----------------------------------
1370 :
1380 SCNCLR
1390 CHAR,12,0,"      SAISIE      ",1
1400 N=N+1:A$(N)=" "
1410 CHAR,5,5,"DONNEZ LE NOM "+STR$(N)
1420 CHAR,5,7:INPUTA$(N)
1430 RETURN
1440 :
1450 REM ------------------------ SAUVEGARDE ------------------------------
1460 :
1470 SCNCLR
1480 CHAR,12,0,"   SAUVEGARDE     ",1
1490 DOPEN#1,"@ESSAI FICH",W
1500 :
1510 FOR A=1 TO N
1520 :   PRINT#1,A$(A)
1530 NEXT
1540 :
1550 CLOSE1
1560 RETURN
1570 :
1580 REM ------------------------ IMPRESSION ------------------------------
1590 :
1600 SCNCLR
1610 CHAR,12,0,"   IMPRESSION     ",1
1620 OPEN 4,4,0,"ESSAI"
1630 :
1640 FOR A=1 TO N
1650 :   PRINT#4,A$(A)
1660 NEXT
1670 :
1680 CLOSE4
1690 RETURN
1700 :
1710 REM ------------------------ PREPARATION -----------------------------
1720 :
1730 SCNCLR
1740 CHAR,12,0,"   PREPARATION    ",1
1750 REM FORMATE LA DISQUETTE
1760 SCRATCH"TRAP RESUME"
1770 DSAVE"TRAP RESUME"
1780 DVERIFY"TRAP RESUME"
1790 RETURN
1800 :
1810 REM ------------------------ TRAITEMENT DES ERREURS ------------------
1820 :
1830 PRINTCHR$(7);
1840 :
1850 IF ER=18 THEN BEGIN
1860 :     SCNCLR
1870 :     CHAR,8,5,"LE FICHIER EST PLEIN"
1880 :     CHAR,9,7,"VEUILLEZ EN OUVRIR"
1890 :     CHAR,14,9,"UN AUTRE"
1900 :     CHAR,9,13,"PRESSEZ UNE TOUCHE",1
1910 :     GETKEY R$
1920 :     N=N-1
1930 :     RESUME 1160
1940 BEND
1950 :
1960 IF ER=5 THEN BEGIN
1970 :     IF EL < 1560 THEN BEGIN
1980 :               CLOSE1
1990 :               CHAR,5,10,"ENCLENCHEZ LE LECTEUR DE DISQUETTE"
2000 :               CHAR,11,14,"ET PRESSEZ UNE TOUCHE",1
2010 :               GETKEY R$
2020 :               RESUME1470
2030 :     BEND
2040 :     IF EL < 1690 THEN BEGIN
2050 :               CLOSE4
2060 :               CHAR,10,10,"ENCLENCHEZ L'IMPRIMANTE"
2070 :               CHAR,11,14,"ET PRESSEZ UNE TOUCHE",1
2080 :               GETKEY R$
2090 :               RESUME1600
2100 :     BEND
2110 BEND
2120 :
2130 IF ER=28 THEN RESUME 1730
2140 :
2150 IF ER=30 THEN BEGIN
2160 :     SCNCLR
2170 :     CHAR,5,10,"VOUS NE POUVEZ PAS ARRETER"
2180 :     CHAR,6,12,"VOTRE PROGRAMME ICI SOUS"
2190 :     CHAR,2,14,"PEINE DE PERDRE TOUTES VOS DONNEES"
2200 :     CHAR,5,16,"PASSEZ PAR L'OPTION 5 DU MENU"
2210 :     GETKEY R$
2220 :     RESUME1180
2230 BEND
2240 :
2250 GRAPHIC0,1:REM  GRAPHIC5,1
2260 PRINT"IL RESTE UNE ERREUR DE PROGRAMMATION"
2270 PRINTER,ERR$(ER)
2280 HELP
2290 :
2300 DO UNTIL R$<>""
2310 : PRINTCHR$(7);
2320 : GET R$
2330 : SLEEP1
2340 LOOP
2350 :
2360 END
2370 REM ------------------------ C'EST FINI ------------------------------

READY.
```

POINTEUR DE REM POUR C64

Grâce à l'instruction REM, le programmeur peut avoir une vue globale d'un programme. Cependant, il lui faudra parfois faire preuve d'attention pour retrouver toutes les lignes REM d'un programme. Avec notre POINTEUR DE REM, vous surmonterez cette difficulté de manière élégante : il affichera en vidéo inversée tous les commentaires situés à la suite d'une instruction REM, ceci pour vous permettre de bien distinguer les autres lignes.

Ce programme inspecte la mémoire BASIC à la recherche du token de l'instruction REM (143) et il inscrit, juste après, le code ASCII de l'affichage inversé. C'est ainsi que seront créées les lignes de REM en vidéo inversée, après la mise en route par RUN 63000. Mais si vous mettez le programme en route par RUN 63010, il effacera le code CHR$ (18).

```
63000 ZE=18:REM *POINTEUR DE REM**
63001 FOR I=PEEK(43)+256*PEEK(44) TO PEEK(45)+256*PEEK(46)
63002 IF PEEK(I)=143 THEN 63004
63003 KO=0:NEXT:END
63004 IF PEEK(I+1)=8 THEN 63003
63005 IF KO<>1 THEN POKEI+1,ZE:KO=1
63006 GOTO 63003
63010 ZE=32:GOTO 63001

READY.

READY.
```

UNE BONNE CONCATENATION

Il est d'usage d'utiliser la méthode suivante pour concaténer deux programmes en BASIC (routine MERGE) : on repousse le début de la mémoire BASIC puis on charge le second programme. Mais s'il faut transformer une partie du programme avec cette méthode, le problème est tellement compliqué qu'il vaut mieux taper le programme une nouvelle fois.

La routine MERGE que voici fonctionne avec un seul lecteur de disquette et rend cette tâche plus simple. Si vous exécutez l'ordre SYS 49152, "Nom du programme n°2", le programme n°2 sera chargé puis concaténé avec celui en mémoire. Les lignes dont le numéro est le même ne sont pas prises en compte par le nouveau programme. Les autres sont placées correctement.

```
1000 REM ********************
1010 REM *      MERGE         *
1020 REM *--------------------*
1030 REM *1985 PAR M.STECHER*
1040 REM ********************
1050 REM
1060 REM FONCTIONNEMENT :
1070 REM
1080 REM     SYS 49152,"NOM"
1090 REM
1100 REM CONCATENE LE PROGRAMME "NOM"
1110 REM
1120 PRINT CHR$(147)
1130 FOR I=0 TO 227
1140 READ P
1150 S=S+P
1160 POKE 49152+I,P
1170 NEXT I
1180 IF S<>28746 THEN PRINT "ERREUR DE DATAS: DIFFERENCE=";ABS(S-28746):STOP
1190 PRINT "LA ROUTINE MERGE EST INITIALISEE !!"
1200 END
1999 REM ROUTINE-MERGE
2000 DATA 32,253,174,32,158,173,32,163,182,166,34,164,35,32,189,255,169,2,162
2001 DATA 8,160,2,32,186,255,32,192,255,144,3,76,164,192,76,186,192,162,2,32
2002 DATA 198,255,32,207,255,32,207,255,32,207,255,32,207,255,240,90,32,207
2003 DATA 255,133,20,32,207,255,133,21,162,255,232,32,207,255,157,0,2,208,247
2004 DATA 232,232,232,232,232,134,11,32,19,166,32,51,165,24,165,45,133,90,101
2005 DATA 11,133,88,164,46,132,91,144,1,200,132,89,32,184,163,165,20,164,21
2006 DATA 141,254,1,140,255,1,165,49,164,50,133,45,132,46,164,11,136,185,252
2007 DATA 1,145,95,136,16,248,32,51,165,76,47,192,32,204,255,169,2,32,195,255
2008 DATA 169,1,32,195,255,32,89,166,76,116,164,169,174,160,192,32,30,171,76
2009 DATA 145,192,13,63,73,47,79,32,69,82,82,79,82,0,169,0,32,189,255,169,1
2010 DATA 162,8,160,15,32,186,255,32,192,255,162,1,32,198,255,32,207,255,201
2011 DATA 48,208,205,32,207,255,201,48,208,198,32,204,255,76,36,192

READY.
```

SIMULATION DE SOURIS

Les nouveaux ordinateurs 16 - bits disposent d'instructions spécifiques permettant d'utiliser la souris et l'écran graphique. Mais le C 64 peut le faire aussi. Afin de vous éviter des dépenses importantes, le joystick remplacera la souris.

Pour le reste, la souris du C 64 se comportera comme ses semblables. Avec la manette de jeu, vous pouvez déplacer un lutin en forme de flèche dans les quatre directions et vous choisissez la fonction désirée par une pression sur le bouton de tir.

En ce qui concerne le programme, vous pouvez rattacher la routine INITIALISATION à votre routine d'initialisation normale. Dans tous les cas, il faudra l'appeler avant d'utiliser la routine TEST DU JOYSTICK. Les autres routines du programme peuvent être placées n'importe où.

Mais il faudra adapter les GOTO en conséquence. Chaque fois que vous utiliserez la souris, il faudra que le menu apparaisse à l'écran. La routine TEST DU JOYSTICK permet de déplacer la flèche en fonction de la manette de jeu et saute à la ligne voulue, inscrite en ligne 260, lorsque le bouton de tir est pressé.

Il faudra inscrire la table d'adresse à cette ligne. Cette table doit être construite de la manière suivante :

IF H > (plus petite abscisse -1)
AND H < (plus grande abscisse +1)
AND G > (plus petite ordonnée -1)
AND G < (plus grande ordonnée +1)
THEN (numéro de ligne)

```
10 REM ********************************
20 REM *   SIMULATION DE SOURIS       *
30 REM ********************************
40 REM
50 REM ********************************
60 REM * DEVELOPPE : 1/85 PAR         *
70 REM *          DIRK NEERFELD       *
80 REM ********************************
90 REM
100 :
110 REM *** INITIALISATION ***
120 :
130 A=53248:D=24:E=50
140 POKE A+21,1:POKE 2040,13:POKE A+39,0
150 FOR X=0 TO 21:READ B:POKE 832+X,B:NEXT
160 :
170 REM *** TEST DU JOYSTICK ***
180 :
190 C=PEEK (56320)
200 IF (C AND 1)=0 THEN E=E-4:IF E<=50 THEN E=50
210 IF (C AND 2)=0 THEN E=E+4:IF E>=240 THEN E=240
220 IF F=0 THEN IF (C AND 4)=0 THEN D=D-4:IF D<=24 THEN D=24
230 IF F=1 THEN IF (C AND 4)=0 THEN D=D-4:IF D<0 THEN D=254:POKE A+16,0:F=0
240 IF F=0 THEN IF (C AND 8)=0 THEN D=D+4:IF D>254 THEN D=0:POKE A+16,1:F=1
250 IF F=1 THEN IF (C AND 8)=0 THEN D=D+4:IF D>=84 THEN D=84
260 IF (C AND 16)=0 THEN 370
270 POKE A,D:POKE A+1,E
280 GOTO 190
290 :
300 REM *** DONNEES DU LUTIN ***
310 :
320 DATA 128,0,0,192,0,0,224,0,0,240,0
330 DATA 0,248,0,0,224,0,0,176,0,0,24,0,0
340 :
350 REM *** TABLE D´ADRESSSES ***
360 :
370 G=INT(E/8)-5
380 IF F=0 THEN H=INT(D/8)-2
390 IF F=1 THEN H=INT((D+255)/8)-2
400 :
410 REM *** IL FAUDRA ECRIRE LES ADRESSES A PARTIR D´ICI ***
420 REM *** COMME INDIQUE DANS LA DESCRIPTION DU PROGRAMME ***

READY.
```

Par exemple, si votre écran affiche la ligne "MICRO INFO" aux coordonnées indiquées et qu'il faut sauter à la ligne 5000.
La ligne d'adresse devra se présenter ainsi :

IF H > (2-1) AND H < (10+1) AND G>(2-1) AND G<(2+1)THEN 5000.

```
130-150     INITIALISATION
190-280     TEST DU JOYSTICK
320-330     DONNEES DU LUTIN
370-450     TABLE D´ADRESSES

VARIABLES UTILISEES
A = ADRESSE DE BASE DU VIC
B = DONNEES DU LUTIN
C = ADRESSE POUR TEST DE JOYSTICK
D = ABSCISSE DU LUTIN
E = ORDONNEE DU LUTIN
F = SECTEUR DU LUTIN
G = ABSCISSE DU SECTEUR DE L´ECRAN
H = ORDONNEE DU SECTEUR DE L´ECRAN
X = DIVERS

READY.
```

ADAPTATION DE L'IMPRIMANTE

Si vous désirez utiliser des feuilles volantes pour alimenter votre imprimante, vous risquez de rencontrer des difficultés à imprimer plusieurs pages. Soit l'arrêt automatique intervient trop tôt et il reste une grande marge blanche en bas de la feuille, soit il faut rester "planté" devant l'imprimante pour l'arrêter à temps lorsque la feuille arrive à sa fin.

Mettons fin à cette angoisse. Le programme que nous vous proposons va résoudre ce problème. Il se rattache à la routine d'impression du système d'exploitation et compte le nombre de RETURN envoyés à l'imprimante. Lorsque 63 retours-chariot se sont produits, le programme arrête l'édition pour vous laisser changer de feuille et il attend que vous pressiez la touche [F7] pour continuer.

Si vous voulez que le programme compte les LINE-FEED et non les [RETURN], changez la ligne 10 ainsi :

```
10 W = 10
```

Si vous voulez imprimer plus ou moins 63 lignes par page, il vous suffit d'inscrire la valeur correspondante en ligne 20. Le numéro de périphérique de l'imprimante est fixé à la ligne 30. Vous appelerez la routine par SYS 828.
Si vous adaptez les lignes 20 et 30 ainsi :

```
20  z = 1
30  g = 3
```

alors l'édition à l'écran ne se produit que lorsque vous pressez la touche F7. Cela s'avère utile pour les listings.
L'affichage est réinitialisé par RUN/STOP + RESTORE

MICRO INFO

```
10 W=13
20 Z=63
30 G=4
100 FOR A=828 TO 905
110 READ X:POKE A,X:S=S+X
120 NEXT
130 IF S<>8837 THEN PRINT "ERREUR DANS LES DATAS !":END
140 PRINT "OK!"
150 POKE 839,Z:POKE 890,Z
160 POKE 855,W
170 POKE 849,G
1000 DATA 169,77,141,38,3,169,3,141,39,3,169,63,141,76,3,96,0
1010 DATA 72,165,154,201,4,208,50,104,72,201,13,208,44,206,76,3,208
1020 DATA 39,32,202,241,36,148,16,5,32,64,237,70,148,120,169,254,141
1030 DATA 0,220,173,1,220,201,247,208,249,88,169,63,141,76,3,169,127
1040 DATA 141,0,220,104,24,96,104,76,202,241

READY.
```

ENFIN UN CURSEUR CLIGNOTANT AVEC GET

"A l'impossible, nul n'est tenu" serait-on tenté de dire face aux difficultés rencontrées pour avoir le curseur clignotant avec l'ordre GET. Voici une astuce simple qui maintient le curseur clignotant comme le fait l'ordre INPUT :

POKE 204,0 active le clignotement du curseur

POKE 204,1 désactive le clignotement du curseur

Ex :

```
10 POKE 204,0
20 GET A$ : IF A$ = "" then 20
30 POKE 204,1
40 GET A$ : IF A$ = "" then 40.
```

Comme pour le GET normal, plusieurs variables chaîne de caractères peuvent être précisées, en les séparant par une virgule.

ARRET DE LISTING

Pour retrouver une erreur commise dans un programme, vous utilisez sans doute la touche CTRL afin de ralentir l'édition de la liste. Mais malgré cela, il est parfois difficile de suivre le listing.
En fait, un seul ordre suffit à résoudre ce problème :

POKE 56325,1

Après cet ordre, presque toutes les fonctions de l'ordinateur en BASIC sont ralenties. En effet, l'interruption-système intervient alors 60 fois par seconde.

Les fonctions d'édition et le clignotement du curseur sont alors fortement perturbés.
L'affichage est ralenti par une pression sur n'importe quelle touche et il s'arrête à chaque ligne si la touche CTRL est pressée.

Pour revenir en mode normal, il suffit d'exécuter l'ordre :

POKE 56325,50

DU MULTITACHE EN BASIC

Le programme MULTITACHE sert à faire fonctionner en même temps un programme en BASIC et un programme en langage machine.

Le programme en langage machine doit se situer à l'adresse 49199 et ne pas faire plus de 300 octets. Il faudra le terminer par l'instruction RTS.

Le fonctionnement en sera assuré par l'ordre : SYS 49152. A partir de là, le programme en langage machine fonctionnera tant que le programme en BASIC ne sera pas arrêté. De plus, il s'arrêtera en même temps que le programme en BASIC.

Le programme en langage machine peut de nouveau être exécuté en BASIC par SYS 49168.



```
100 FOR I=49152 TO 49198
110 READ X:POKE I,X:S=S+X:NEXT
120 DATA 169,0,141,80,195,169,27,141,8,3,169,192,141,9,3,96,169
130 DATA 228,141,8,3,169,167,141,9,3,96,32,115,0,201,92,141,32
140 DATA 203,32,47,192,173,32,203,32,121,0,76,231,167
150 IF S<>5054-255 THEN PRINT "ERREUR DANS LES DATAS !"

READY.
```

DE NOUVEAUX CARACTERES POUR LES MPS 801/803

Il est possible de créer de nouveaux caractères et de les imprimer en programmant la matrice d'aiguilles des imprimantes MPS 801/803. Mais le calcul des données est fastidieux et il vaut mieux laisser ce travail à un programme.

Après avoir démarré le logiciel, vous dessinerez le caractère souhaité en mode direct avec des étoiles "*" pour chaque point à imprimer, dans le coin gauche en haut de l'écran. La hauteur ne devra pas dépasser 7 points et la largeur ira de 1 à 16 points.

Ainsi, le coin gauche de l'écran pourrait se présenter comme suit :

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

Quand vous avez créé votre nouveau caractère, tapez les instructions POKE 915, x : SYS 828.
X indique la largeur du caractère. Dans l'exemple précédent, X = 8.

Une ligne de données s'affichera et vous pourrez inscrire un numéro de ligne ou bien un DATA ou bien encore utiliser un autre moyen pour préserver ces données. Si vous décidez de créer une ligne de DATA, la virgule à la fin de la ligne devra être effacée.

```
Ò REM ***********************************
1 REM *                                 *
2 REM *  +++ NOUVEAUX CARACTERES+++     *
3 REM *  ++  POUR MPS 801/803    ++     *
4 REM *                                 *
5 REM *  COPYRIGHT (C) WOLFGANG KUNZ    *
6 REM *                ROTTALSTR. 11    *
7 REM *                7951 EICHENBERG* *
8 REM *                                 *
9 REM ***********************************
10 DIM H(75):FOR I=0 TO 9
20 H(48+I)=I:H(65+I)=I+10:NEXT
30 FOR I=828 TO 918: READ A$
40 H=ASC(LEFT$(A$,1)):L=ASC(RIGHT$(A$,1))
50 D=H(H)*16+H(L):S=S+D:POKEI,D
60 A=A+1:IF A<20 THEN NEXT:A=-1
65 PRINT "LIGNE : ";1000+Z;
70 READ V:Z=Z+1:IF V=S THEN 85
80 PRINT "ERREUR DANS LES DATAS !";999+Z:STOP
85 IF A<0 THEN END
90 S=0:A=0:PRINT:NEXT:END
95 :
1000 DATA A9,01,8D,97,03,A9,00,8D,98,03,AA,8D,99,03,AE,99,03,BD,00,04,  1920
1001 DATA C9,2A,D0,0A,AD,98,03,18,6D,97,03,8D,98,03,8A,18,69,28,AA,AD,  2022
1002 DATA 97,03,0A,8D,97,03,D0,E1,AD,98,03,18,69,80,8D,98,03,A9,00,AE,  2116
1003 DATA 98,03,20,CD,BD,A9,2C,20,D2,FF,A9,00,8D,98,03,A9,01,8D,97,03,  2221
1004 DATA EE,99,03,AD,99,03,C9,06,D0,B4,60,  1414

READY.
1 REM ***********************************
2 REM *         PROGRAMME TEST          *
3 REM ***********************************
10 OPEN 4,4
20 PRINT#4,CHR$(8);
30 FOR I=1 TO 8
40 READ A
50 PRINT#4,CHR$(A);
60 NEXT
70 PRINT#4
80 CLOSE 4
100 DATA193,193,193,201,221,247,227,193

READY.
```

ENCORE ET TOUJOURS DE LA MEMOIRE

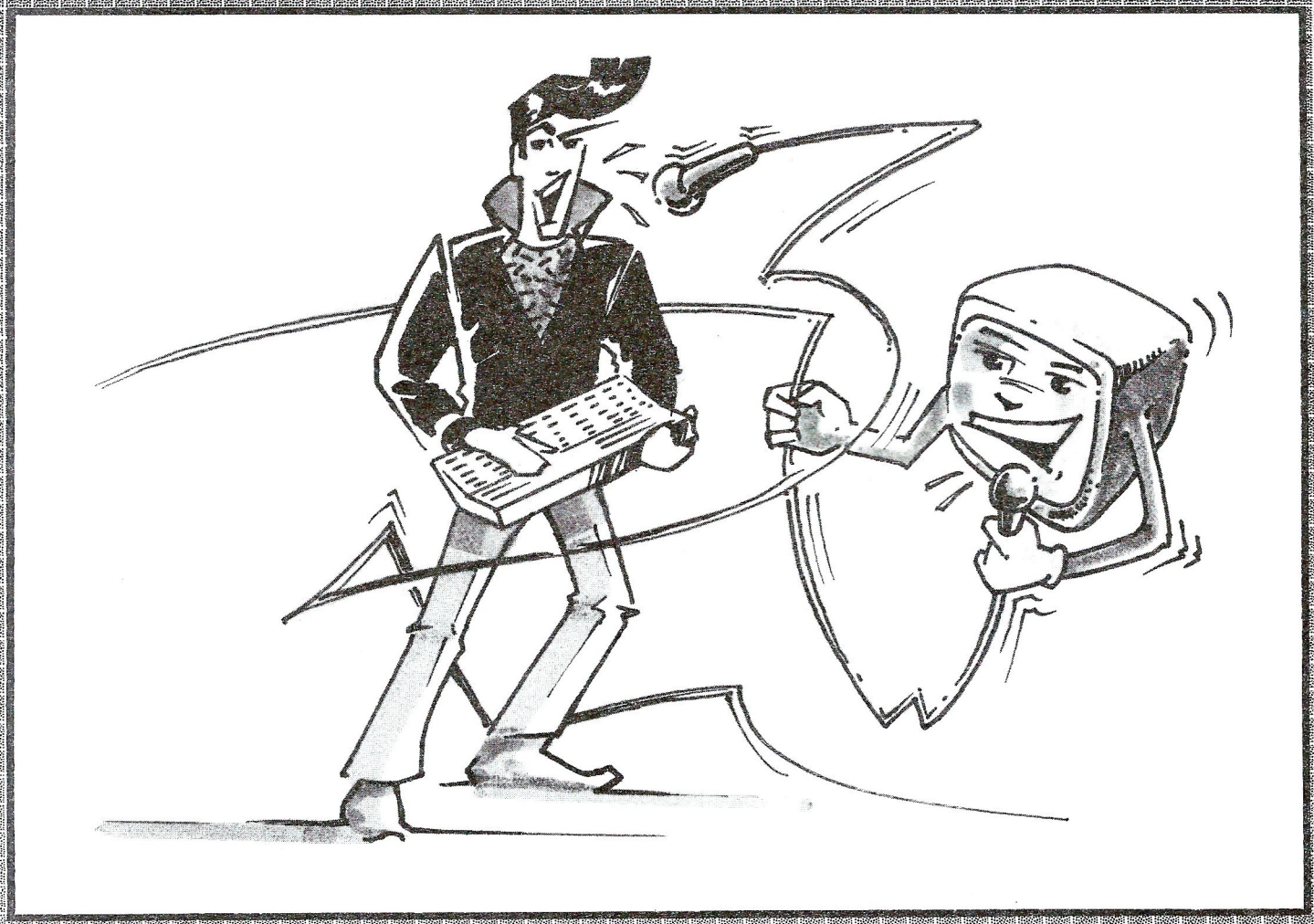
lorsque l'on se sent à l'étroit...
...le C 64 dispose encore d'une réserve de mémoire de 1228 octects pour le BASIC

Le petit programme qui suit vous permettra d'utiliser cette zone mémoire. Sauvegardez-le avant de le mettre en route car il s'effacera de lui-même pour réinitialiser les pointeurs du BASIC.

PRINT FRE (0) + 65535 donne maintenant comme résultat : 40136 octets de libre. Pour obtenir cela, il a fallu déplacer la mémoire d'écran en 50176 ($ C400) et le générateur de caractères en 57 344 ($ E000). Vous devez vous en souvenir si vous désirez utiliser les POKE pour la mémoire d'écran. D'autre part, vous ne pourrez pas utiliser le magnétophone à cassette car le tampon cassette est utilisé pour la mémoire BASIC.

Cependant, puisque le générateur de caractères se situe à partir de $E000 en RAM, les caractères pourront être aisément redéfinis.

```
10 POKE 56334,0:POKE 1,51:FOR I=0 TO 4095:POKE 57344+I,PEEK(53248+I):NEXT
20 POKE 1,55:POKE 56334,1
30 POKE 53272,25:POKE 56576,196:POKE 648,196
40 POKE 43,53:POKE 44,3:POKE 820,0:NEW

READY.
```



MO5, TO7/70

COMPACTEUR POUR MO5, TO7/70

Cet utilitaire supprime tous les espaces inutiles dans un programme écrit en BASIC. Bien entendu, les espaces seront respectés à l'intérieur des guillemets et dans les lignes de REM.
Le programme a été implanté à partir de l'adresse $9F90 Mais vous pouvez le reloger où vous le voulez.
Pour compacter un programme tapez :
EXEC $H9F90

Vous pouvez sauvegarder le code objet par :
SAVEM"DEL",$H9F90,$H9FF3,0

La liste ci-dessous est donnée pour le TO7/70, pour que ce programme fonctionne sur MO5 il faut remplacer la première ligne :

```
001C    PRG    EQU    $1C
```

par

```
001C    PRG    EQU    $13
```

```
6809 Language Module 1.0
                                    Page 1


9F90                          ORG     $9F90


                    * COMPACTEUR BASIC T07
/70

            001C    PRG       EQU     $1C

9F90 9E     1C      START     LDX     <PRG
9F92 34     10      STO       PSHS    X
9F94 AE     84                LDX     ,X

                    *TEST FIN PRG

9F96 26     02                BNE     SUI
9F98 35     90                PULS    PC,X

9F9A 35     10      SUI       PULS    X
9F9C 34     10                PSHS    X
9F9E 30     04                LEAX    4,X

                    *TEST POUR REM OU ""

9FA0 EC     84                LDD     ,X
9FA2 1083   3A8D              CMPD    #$3A8D
9FA6 27     3C                BEQ     LINE
9FA8 81     8C                CMPA    #$8C
9FAA 27     38                BEQ     LINE
9FAC A6     80      LOAD      LDA     ,X+
9FAE 27     34                BEQ     LINE
9FB0 81     22                CMPA    #'"
9FB2 27     36                BEQ     UND
9FB4 81     20                CMPA    #$20
9FB6 26     F4                BNE     LOAD
9FB8 34     10                PSHS    X
9FBA 8D     06                BSR     DEL
9FBC 35     10                PULS    X
9FBE 30     1F                LEAX    -1,X
9FC0 20     EA                BRA     LOAD

                    * ENLEVE ESPACES

9FC2 A6     84      DEL       LDA     ,X
9FC4 A7     82                STA     ,-X
9FC6 30     02                LEAX    2,X
9FC8 10AE   84                LDY     ,X
9FCB 26     F5                BNE     DEL
9FCD E6     02                LDB     2,X
9FCF 26     F1                BNE     DEL
9FD1 6F     82                CLR     ,-X
9FD3 10AE   64                LDY     4,S
9FD6 EC     A4      DDD       LDD     ,Y
9FD8 83     0001              SUBD    #1
9FDB ED     A4                STD     ,Y
9FDD 1F     02                TFR     D,Y
9FDF EC     A4                LDD     ,Y
9FE1 26     F3                BNE     DDD
9FE3 39                       RTS

                    * NLLE LIGNE
```

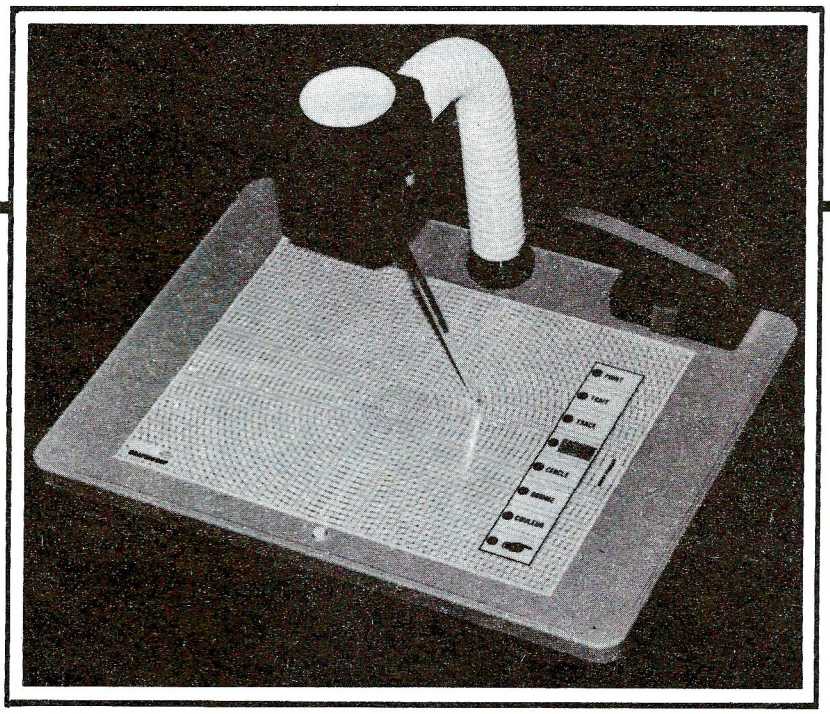
```
6809 Language Module 1.0
                                    Page 2


9FE4 35     10      LINE      PULS    X
9FE6 AE     84                LDX     ,X
9FE8 20     A8                BRA     STO
9FEA A6     80      UND       LDA     ,X+
9FEC 27     F6                BEQ     LINE
9FEE 81     22                CMPA    #'"
9FF0 27     BA                BEQ     LOAD
9FF2 20     F6                BRA     UND
            0000              END

00000 Total Errors


6809 Language Module 1.0
                                    Page 3


DDD       9FD6
DEL       9FC2
LINE      9FE4
LOAD      9FAC
PRG       001C
START     9F90
STO       9F92
SUI       9F9A
UND       9FEA
```

MICRO

mmc
INTERNATIONAL

GRAPHISCOP II

ÉTUDE, RECHERCHE, CRÉATION
DE PROTOTYPES EN
ROBOTIQUE, AUTOMATISME
ET INFORMATIQUE
BREVETS - INDUSTRIALISATION



DISPONIBLE SUR

- APPLE II +, II E, II C
- ORIC-1 / ATMOS
- COMMODORE 64, SX-64 ,128
- AMSTRAD 464, 664, 6128

Concue et fabriquee en France, la tablette graphique "GRAPHISCOP II" (brevet et Nom deposes) vous permettra de representer vos dessins a l'ecran en utilisant la resolution graphique maximale de votre micro-ordinateur pour un maximum de couleurs.

"GRAPHISCOP II" est livree avec un boitier interface connectable a votre micro,un manuel d'utilisation,un logiciel sur cassette ou sur disquette.

◆ SPECIFICATIONS TECHNIQUES :

Dimensions tablette graphique en plastique moule	465 x 340 mm
Hauteur potence montee	300 mm
Surface tracable	220 x 220 mm
Precision	1 pixel
Poids	800 g
Alimentation :	depend des interfaces utilisees.

◆ BOITIERS INTERFACE :

Connecteur d'alimentation de la tablette graphique
4 potentiometres de reglage X, Y
Led temoin s'allumant chaque fois que l'on appuie sur le bouton poussoir
Cable d'interface a relier a votre micro-ordinateur.

S.A.R.L. Capital de 20 000F - R.C. PARIS B 329 311 187
R. Merzoug 1, Rue Lincoln 75008 - Paris Tel (1) 42.56.12.82

◆ TYPES DE CONNECTIONS :

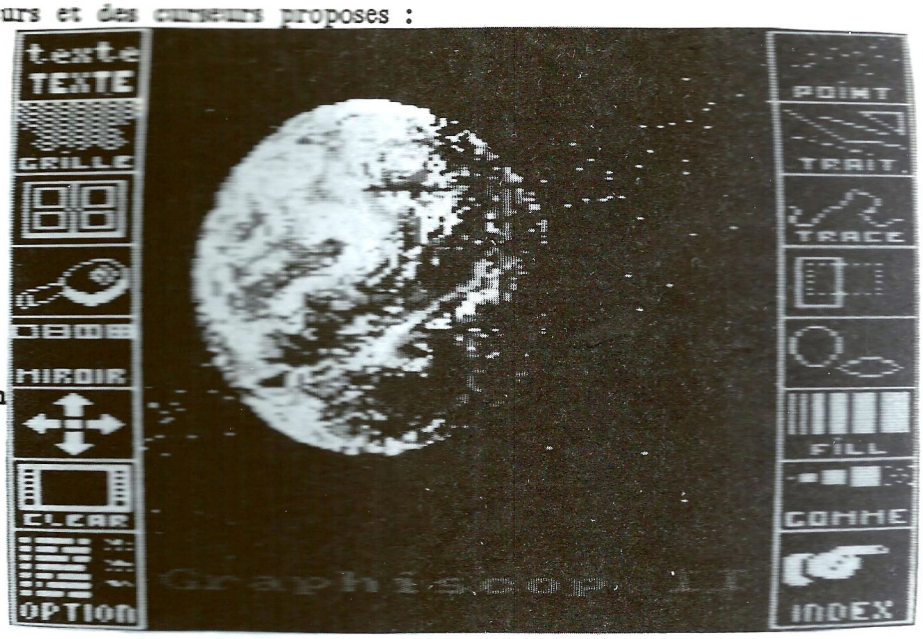
APPLE II +	Sur la prise joystick 16 broches
APPLE IIe, IIc	Sur la prise joystick 9 broches
ORIC-1/ATMOS	Un connecteur 20 broches a l'entree imprimante et le fil d'alimentation au +5v du bus d'extension.
COMMODORE 64	Sur la prise joystick 9 broches port 2
AMSTRAD	Sur la prise joystick 9 broches.

◆ FONCTIONS LOGICIELLES :

"GRAPHISCOP II" execute toutes fonctions graphiques a partir d'un menu de commandes (ex Commodore et Amstrad).

Vous pourrez avec le choix des couleurs et des curseurs proposes :

- Faire des points, des traits, des traces de segments continus
- Faire des boites, des cercles
- Remplir une zone fermee, gommer
- Ecrire du texte sur votre dessin
- Travailler en mode miroir
- Copier, deplacer, translater verticalement ou horizontalement une zone
- travailler a la loupe
- Annuler l'action de la derniere fonction
- Sauvegarder, Charger un dessin etc....



◆ T A R I F S :

	Prix Public :		
GRAPHISCOP II complet	TTC	1490.00	Frs
Interface et logiciel	TTC	415.10	Frs

Revendeurs : nous consulter

◆ GARANTIE :

Les produits M.M.C. International beneficient d'une garantie de 1 an constructeur

◆ BON DE COMMANDE :

Je desire recevoir un GRAPHISCOP II au prix de _____TTC + 35 F (port et emballage) comprenant l'appareil, le logiciel, le manuel utilisateur et la garantie constructeur pour :

Apple II + ☐ Apple II e/c ☐ Oric-1/Atmos ☐ Commodore 64 ☐ Amstrad ☐

Nom _____ Prenom _____

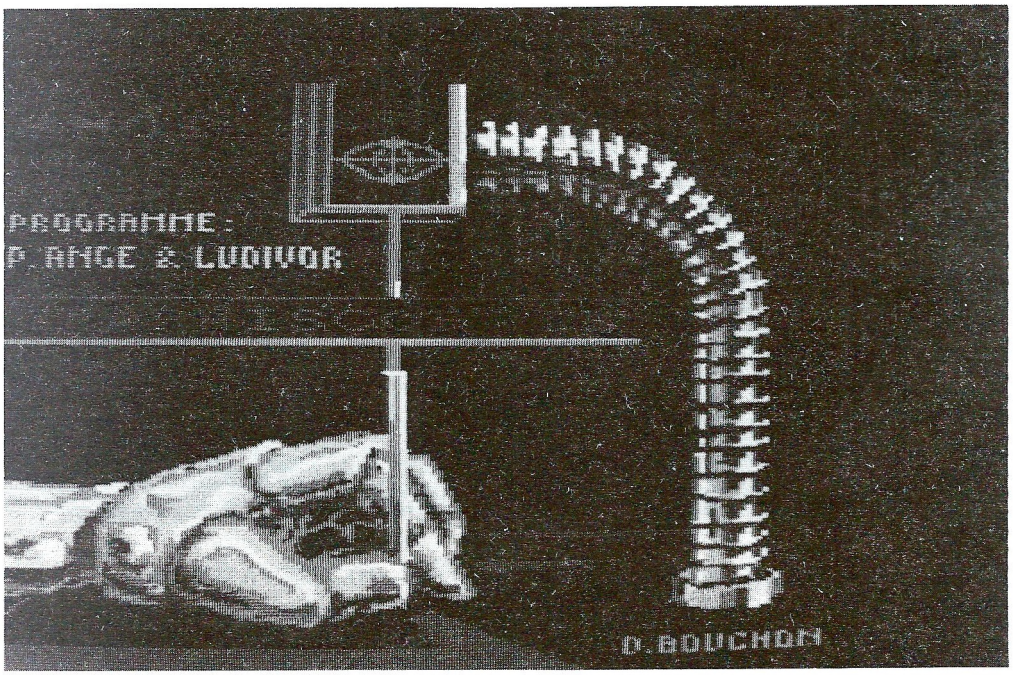
Adresse _____

_____ Code postal _____ Ville _____

MI 3 86

Ci-joint : ☐ Cheque postal ou bancaire ☐ Mandat poste de _____Frs.

"GRAPHISCOP II",
la tablette graphique





Les tablettes graphiques étaient, jusqu'à présent, réservées aux systèmes de D.A.O professionnels, principalement en raison de leur technologie assez onéreuse à mettre en oeuvre. GRAPHISCOP II, créée et industrialisée par M.M.C International est une tablette graphique française. C'est un périphérique de saisie de données accessible à tous les possesseurs de micro-ordinateurs du type :

APPLE II+,IIe,IIe
ORIC1/ATMOS
COMMODORE64,SX64,128 (mode 64)
AMSTRAD 464,664,6128
MO5 et IBM PC logiciel en réalisation

GRAPHISCOP II transmet à l'ordinateur des coordonnées X et Y pointées sur une surface plane.

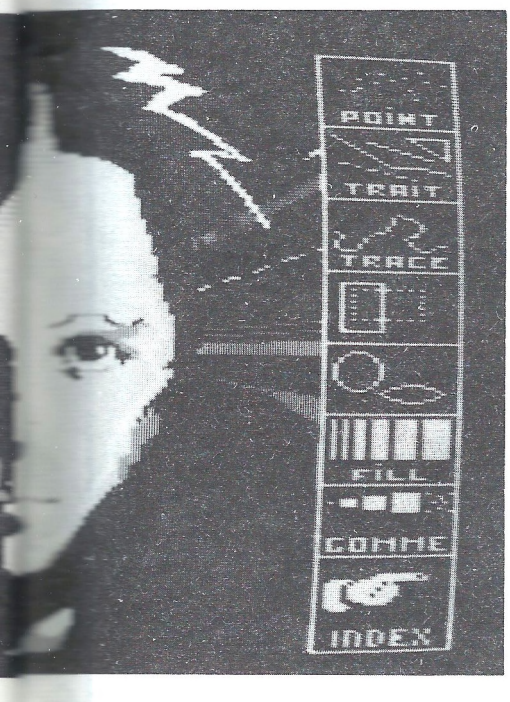
Le système se présente sous la forme d'une tablette moulée de 45 X 35 cm surmontée d'une potence constituée d'un tube canelé formant angle droit, dont la base est fixée sur le bord de la tablette. Un boitier rectangulaire est situé à l'autre extrémité de ce tube, à l'aplomb du centre de la table à une hauteur de 25 cm. Un tube métallique monté sur rotule émerge de sa face inférieure et la pointe d'écriture coulissant dans ce tube déplace deux curseurs capteurs potentiométriques placés à 90°. Un bouton poussoir est situé sur l'avant de la tablette dont la fonction est définie par le logiciel d'exploitation.

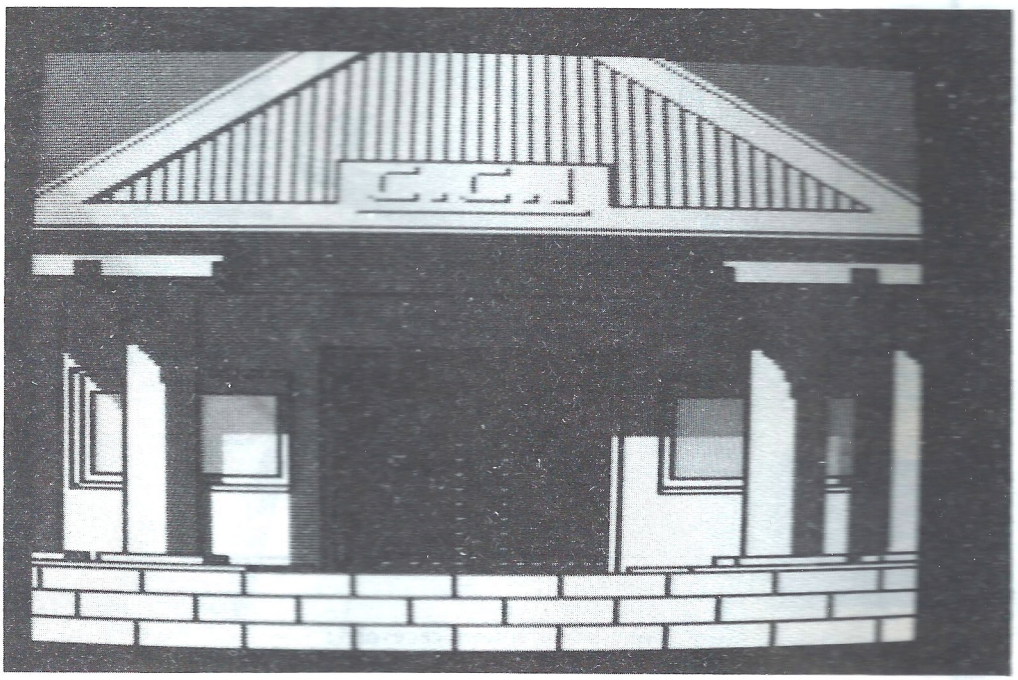
GRAPHSICOP II est livrée avec un boitier interface connectable avec votre micro, un manuel d'utilisation, un logiciel sur cassette ou sur disquette.

Les avantages de cette tablette réside dans la multiplicité des micro ordinateurs sur lesquels elle peut se brancher. Seul l'interface est à changer, et cela pour un cout peu élevé.

FONCTIONS LOGICIELS

Un menu de commande disposé sur le côté droit permet de choisir les options en les pointant. Il est facile de l'utiliser grâce au choix des couleurs (16 parmi 27 pour AMSTRAD, 16 pour COMMODORE etc...) et des curseurs ou brosses proposées (8 différentes) de faire des points, des traits (3 sortes), des tracés de segments continus, de faire des boites, des cercles ou ellipses vides ou pleines, remplir une zone, gommer, écrire du texte sur votre dessin, effacer ce texte





sans toucher au dessin, travailler en mode miroir (4), copier, déplacer, translater verticalement ou horizontalement une zone, déplacer dans les quatre directions l'écran, travailler au pixel près grâce à la loupe, annuler l'action de la dernière fonction, sauvegarder, charger un dessin etc.

La puissance du logiciel et la maniabilité de la tablette font de GRAPHISCOP II "la petite pro", elle est d'ailleurs utilisée par les graphistes des éditeurs de logiciels (Michel RHO de ERE Informatique...).

PRIX

GRAPHSICOP II complet :
TTC 1490,00 FF
Interface et logiciel supplémentaire :
TTC 415,10 FF

Les produits MMC International bénéficient d'une garantie de 1 an constructeur.
Grâce au bon rapport qualité/Prix, le constructeur met à la portée de l'amateur un accessoire normalement réservé aux "pros".

ABONNEMENT

BULLETIN D'ABONNEMENT

MICRO INFO N° 3 VOUS A PLU : ABONNEZ VOUS !

NOM :
PRENOM :

ADRESSE :

CODE POSTAL : VILLE :

Je règle la somme de 70 FF plus 10 FF de frais d'envoi par :

[] chèque banquaire [] CCP [] carte bleue : N°

DATE : SIGNATURE

Je désire uniquement recevoir :

[] Le prochain numéro [] le numéro 1 Mai 85 [] le numéro 2 Sept. 85

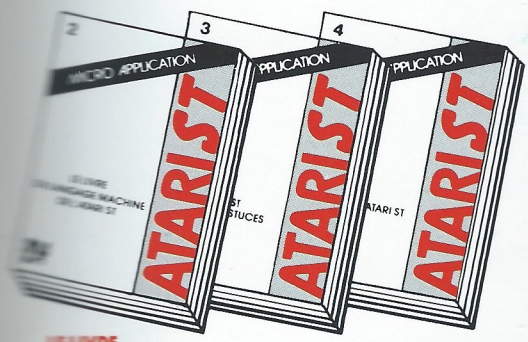
[] Un autre numéro 3

Je règle la somme de 20 FF plus 5 FF de frais d'envoi par :

[] chèque bancaire [] CCP

DATE : SIGNATURE :

Application MA



LE LIVRE DU LANGAGE MACHINE DE L'ATARI ST

Tout ce qu'il faut savoir pour tirer au mieux parti de votre ATARI ST : système de calcul et de bit manipulation du 68000, utilisation des registres, structure des commandes, programmation structurée : récursion, piles, procédures et fonctions, listings sources et programmes assembleurs, routines systèmes... Un super livre!

Réf. : ML 141
Prix : **149** FF

ATARI ST TRUCS et ASTUCES

Un recueil complet de trucs et d'astuces que votre tout nouveau ATARI ST va beaucoup apprécier! Des graphismes fantastiques à partir de programmes en BASIC, des exemples et des conseils pour programmer en langage C et en Assembleur...

Réf. : ML 140
Prix : **149** FF

La BIBLE de l'ATARI ST

Ce livre contient un ensemble complet d'informations sur l'ATARI ST, la description HARDWARE de la machine ainsi que des schémas détaillés et amplement expliqués notamment des interfaces V24, du port d'extension, de l'interface midi, la structure des graphiques, du BIOS, de GEM, les adresses systèmes importantes, le fonctionnement de la souris...
Réf. : ML142 Prix : **249 FF**



TRUCS ET ASTUCES DU COMMODORE 128

Ce livre est un hit pour chaque utilisateur de Commodore 128 et contient des informations essentielles sur : le BANKSWITCHING et la configuration de la mémoire, la description des registres du contrôleur vidéo et la programmation graphique en 640 x 200 sur l'écran 80 colonnes, les fenêtres, le fonctionnement en mode multi-tâches, la structure des instructions, et beaucoup de programmes exemples et utilitaires...

Réf. : ML 135
Prix : **149** FF

LA BIBLE DU COMMODORE 128

Description complète du système, du hardware et des interfaces, explications des chips VIC, du système vidéo (640 x 200 en haute résolution avec écran de 80 colonnes sur 25 lignes), le SID, la description détaillée du MMU (Memory Management Unit) et comme toujours le listing commenté de la ROM et de nombreux programmes exemples et utilitaires à taper. Un super livre, comme toutes les BIBLES de Micro Application!

Réf. : ML 136
Prix : **249** FF

LE LIVRE DU CP/M

Si vous voulez entrer avec le Commodore 128 dans le monde du CP/M, voici l'ouvrage qu'il vous faut! Ce livre en effet vous explique tout sur cet operating system en tenant compte des spécificités propres au Commodore 128, notamment le format Commodore et le format CP/M, la gestion et la programmation des données... Indispensable pour profiter à fonds de CP/M sur le Commodore 128.

Réf. : ML 137
Prix : **149** FF

LE LIVRE DU LECTEUR DE DISQUETTE 1571

Ce livre concerne débutants et confirmés. Il explique tout sur les fichiers relatifs et séquentiels à partir du BASIC 7.0. Les mordus apprendront comment utiliser l'accès direct, les routines internes du DOS et la programmation de celui-ci et bien sûr le listing source du DOS commenté. Tout pour utiliser efficacement le nouveau 1571!

Réf. : ML 138 Prix : **179** FF

LE LIVRE DU BASIC C 7.0

Cet ouvrage est une introduction complète et didactique à l'univers du BASIC 7.0. Il permet d'apprendre facilement et rapidement la programmation structurée, de l'élaboration de l'algorithme à la réalisation finale du programme. Des exemples corrigés guideront le programmeur tout au long de son apprentissage.

Sujets traités :
Modes de programmation et organigrammes, techniques avancées de programmation, Réalisation de menus, Programmation graphique, Tableaux multi-dimensionnels, Routines de sortie, Gestion des données, et de nombreuses applications pratiques!

Réf. : ML 144 Prix : **149** FF

NOUVEAU



EXCLUSIF MICRO APPLICATION !

LE LIVRE DU GEM SUR ATARI ST

Cet ouvrage contient ce qu'il est nécessaire de savoir pour utiliser GEM efficacement : Fonctionnement de la souris, Virtual Device Interface, Application Environment Services, Graphics Device Operating System. Description de routines utilisant GEM en C et en Assembleur. Utilisation standard et spécifique de l'operating system.

Réf. : ML 139 Prix : **149** FF

LOGICIEL

MODULA-2 / ST

Compilateur **MODULA** pour ATARI ST. **MODULA-2** / ST est l'implantation complète du langage **MODULA** conçue par NIKLAUS WIRTH (auteur du PASCAL) sur 520 ST. Ce compilateur permet de développer des logiciels utilisant GEM et les capacités graphiques du ST. Il est livré avec un éditeur complet lié au compilateur pour une détection rapide des erreurs. Réf. : AT 001 Prix : **1450** F TTC

NOUVEAU



Centre de démonstration permanente en nos locaux : 13, rue Sainte Cécile, 75009 Paris - Métro rue Montmartre Ouvert de 9 h à 13 h et de 14 h à 18 h du lundi au vendredi.

TRUCS ET ASTUCES du THOMSON MO5 T07 70

Enfin un livre **MICRO APPLICATION** pour les **THOMSON!** Avec trucs et astuces pour le MO5/TO7-70 nous voulons apporter le maximum à tous les possesseurs de ces ordinateurs. Tout au long des 400 pages de ce très bel ouvrage ils trouveront non seulement de nombreux conseils et explications pour profiter au plus de leur machine mais aussi des programmes utilitaires très puissants comme un traitement de texte, un générateur de sprites (lutins) qui font cruellement défaut à la machine française, un moniteur disque, une gestion de fichier complète, un programme graphique... Des astuces pour utiliser le DOS et pour programmer en langage machine, créer des fenêtres...

Réf. : ML 1160 Prix : **149** F TTC

Avez-vous lu MICROINFO n° 3 Spécial Trucs et Astuces ?

BON DE COMMANDE

Demandez notre catalogue gratuit ☐ Atari ☐ Amstrad ☐ Commodore

DESIGNATION	QUANTITE	PRIX
	TOTAL TTC	

CB date d'expiration : _____

☐ Mandat ☐ Chèque ☐ CCP.

Libellez vos chèques à l'ordre de Micro-Application.

Nom, Prénom _____

Adresse _____

Ville _____ C.P. _____

CB ☐☐☐☐ ☐☐☐ ☐☐☐☐ ☐☐☐

Date et signature

+ 20 F de frais d'envoi
ou 40 F pour envoi recommandé.
Port gratuit pour toute commande supérieure à 250 F.

MICRO APPLICATION
13, rue Sainte Cécile 75 009 PARIS
Tél. : (1) 47-70-32-44



Des performances en plus !

MICRO APPLICATION 13, rue Sainte Cécile 75009 PARIS tél. (1) 47-70-32-44

HILCU INTERNATIONAL P.O Box 221 CH-3063 ITTIGEN BERN