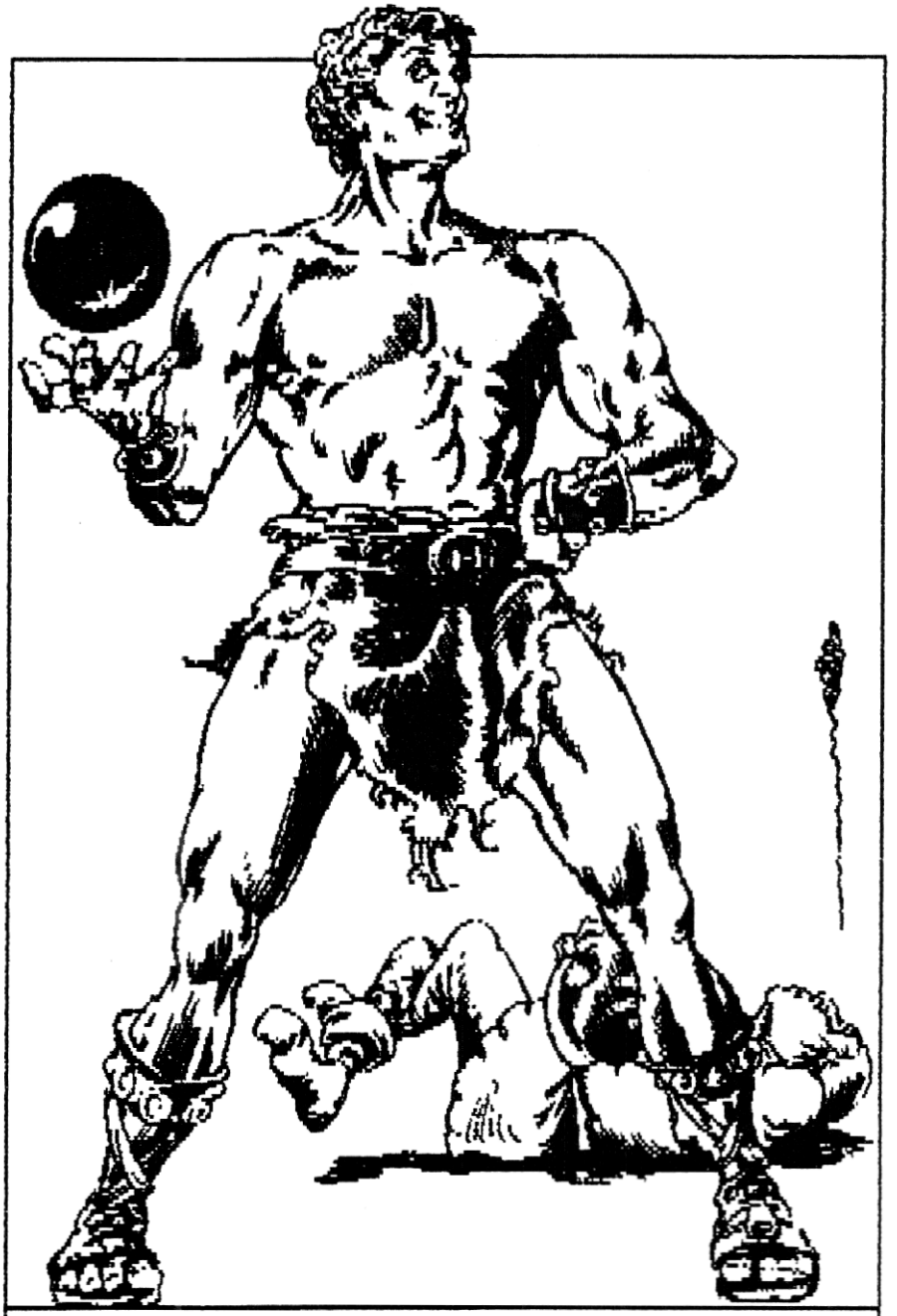
NUMERO 4

PROGRAMMEZ VOTRE CPC OU CPC+ AVEC

# Quasar CPC

DERNIER TRIMESTRE 1993

CPC TOUT SIMPLEMENT



Qui veut jouer au boulet de canon avec moi ?

CPC TOUT SIMPLEMENT

... NE MANQUEZ SURTOUT PAS LE PROCHAIN NUMERO DE QUASAR CPC... ... NOUS VOUS PREPARONS DEUX NOUVEAUX COURS... L'UN SUR LE CPC+ (COMMENT L'EXPLOITER A SA JUSTE VALEUR) ET L'AUTRE SUR L'ELECTRONIQUE APPLIQUEE AU CPC (COMMENT CONCEVOIR DES INTERFACES). VOUS Y RETROUVEREZ BIEN SUR TOUS LES AUTRES COURS AINSI QUE LES TESTS, LES BIDOUILLES, LES POKES, LES PETITES ANNONCES, LA RUBRIQUE X ET LES IMMUABLES HISTOIRES PERPENDICULAIRES... A BIENTOT !

4ème Numéro

# QUASAR CPC

## Sommaire — Pages



**SEKILAOUBLIÉDEDIREDANLÉDITO :**

Dans cette nouvelle formule de notre fanzine il faut que vous sachiez que chaque redacteur a etudie lui-meme la presentation de sa rubrique ; la consequence en est donc une assez grande diversite... J'espere que ce choix qui apporte une touche d'originalite a Quasar saura vous satisfaire...

Vous remarquerez tout de meme que nous avons conserve une assez grande homogeneite quant a la presentation des cours... C'est plus clean...

## EDITORIAL

Salut tout le monde ! Voici enfin le 4ème numéro de Quasar CPC ! Nous espérons que ce nouveau numéro sur papier vous comblera... Comme vous pourrez le constater nous avons conservé toutes les rubriques du numéro 3 qui, semblerait-il, vous a beaucoup plus. Vous aurez ainsi le loisir de retrouver les rubriques d'initiation et de perfectionnement à la programmation en Basic, Turbo Pascal 3 et Assembleur. Toutefois, vous constaterez la disparition de la rubrique intitulée "Basic Au Berceau" puisque personne ne paraissait s'y intéresser. Enfin, je vais vous laisser parcourir les 22 pages de ce fanzine qui, je vous le garantis, n'en est pas encore à son dernier numéro !

En effet, les temps sont durs et beaucoup de fanzines disparaissent (meme les meilleurs nous quittent...). Mais nous, nous resterons jusqu'au bout pour que nous puissions partager ensemble notre passion du CPC. Mais c'est dur, très dur... Amstrad Cent Pour Cent a fermé boutique, Titus a édité son dernier jeu, Amstrad a fabriqué ses dernières disquettes 3"... Et la liste pourrait encore etre longue ; tout ce qui faisait la force du CPC est en train de disparaitre petit à petit mais il ne faut pas oublier les freewares, les démos, c'est vers eux qu'il faut à présent se tourner ! Vous allez me dire que nombre de programmeurs ont déjà déserté le CPC mais vous, nous, on est encore là !

Et puis il ne faut pas négliger les nouveaux groupes qui se forment ci et là, notamment en Allemagne. Le CPC est encore là, et il n'est pas pret de disparaitre ! Les quelques millions de CPCs vendus en Europe ne vont pas s'évanouir comme ça, d'un coup de baguette magique...

Néanmoins vous ne retrouverez pas les Actus dans ce numéro car nous ne savions quoi vous y mettre ; nous aurions certes pu vous parler des dernières démos venues d'Allemagne mais je suis sur que vous les connaissez déjà alors...

Mais si nous parlions ici d'autre chose ; en effet, les fetes approchent... Et quelles fetes ! Savez-vous qu'en 1994 la gamme des CPCs aura tout juste 10 ans ? Ca se fete non ?

Bon, eh bien je vais vous laisser explorer Quasar CPC. Et je vous donne rendez-vous dans le numéro 5 qui, je l'espère, sera enrichi de nouvelles rubriques et peut-etre meme de nouveaux rédacteurs...

La Rédac.

ATTENTION !!! SI VOUS VOULEZ ENTRER EN CONTACT AVEC NOUS FAITES-LE IMPERATIVEMENT EN NOUS ECRIVANT AUX ADRESSES SPECIFIEES DANS LA RUBRIQUE X... NE NOUS CONTACTEZ SURTOUT PAS PAR MINITEL CAR NOUS N'Y RELEVONS PAS LES MESSAGES QUE VOUS AURIEZ PU NOUS Y ADRESSER POUR LA SIMPLE ET BONNE RAISON QUE NOUS N'AVONS PAS DE MINITEL !!! ALORS ECRIVEZ-NOUS... EN PLUS, CA COUTE MOINS CHER...

Concernant les digits que vous rencontrerez dans ce numero de Quasar elles sont toutes faites maison... Si, si ! — Sources : Ed. Folio, Generation 4.

# La Rubrique X

**La voici la voilà cette petite Rubrique X... Je n'ai rien de particulier à mettre dans cette intro et pourtant il va bien falloir que je les remplisse ces quelques lignes. Bon, eh bien tout comme dans Quasar 3 cette rubrique a pour but de vous donner les adresses de tous les participants à l'élaboration de votre fanzine préféré afin que vous puissiez nous contacter sans problème. Alors, vous êtes prêts, c'est parti !!!**

Tout d'abord, voici l'adresse de la rédac. proprement dite :

QUASAR CPC
RIMAURO Gilles et Philippe
8, chemin des Maillos
09200 SAINT-GIRONS

A cette adresse vous pouvez tout demander, et la réponse est assurée. Le matériel disponible à cette adresse est le plus important puisque nous y disposons d'un CPC6128 muni de multiples extensions, d'un PC 2286 et d'une imprimante...
Ensuite il y a l'adresse où je poursuis mes études (je n'y suis donc que pendant la période scolaire) :

RIMAURO Philippe
Chez M. et Mme FOURCADE René
75, avenue des sports
65690 BARBAZAN-DEBAT

Ici, vous pourrez m'écrire pour me demander des renseignements techniques concernant soit la programmation soit l'installation de périphériques sur le CPC. La réponse est là aussi assurée mais je ne peux pas vous garantir le délai de la réponse car j'ai pas mal de boulot... alors ne me surchargez pas de courrier S.V.P.... En outre, je ne dispose à cette adresse que d'un CPC6128+ sans aucune extension... c'est triste...
Et puis il y a l'adresse de notre cher Tony, non, non, il n'habite pas dans une caverne...

RENAUT Antoine
43/45, avenue Paul-Laffont
09200 SAINT-GIRONS

Vous pouvez lui demander tout ce que vous voulez, mais je ne garantis pas la typographie de ses réponses. Toutefois, il est préposé à la réception des lettres d'insultes en tout genre... Son matériel (dont je ne garantis pas la qualité...) : un CPC6128 et une Multiface Two pour CPC+ (non, non, vous lisez bien !).
(MOZik : C'est a cause des prises...)

**Ciao tout le monde,**

**OffseT**

# LES HISTOIRES PERPENDICULAIRES

Salut tout le monde ! Pour ce nouveau numéro papier de Quasar CPC nous allons revenir à une formule plus simple et plus intéressante des Histoires Perpendiculaires car j'ai cru remarquer que celle-ci dérivait sérieusement...

Bref, pour commencer, une petite histoire (perpendiculaire) qui, je l'espère, vous divertira avant le casse-tête qui va suivre...

## LA SENTINELLE

De Fredric Brown

Il était trempé jusqu'aux os, couvert de boue, il avait faim et froid et il était à cinquante mille années-lumière de chez lui. Un soleil étranger répendait une glaciale lumière bleutée, et la pesanteur, double de celle à laquelle il était habitué, rendait tout mouvement infiniment pénible.

Mais après des dizaines de milliers d'années, ce coin de terre n'avait pas changé. Il était commode pour ceux de l'aviation, avec leurs astronefs bien astiqués et leurs superarmes ; pourtant quand on en venait au moment décisif, c'était au fantassin, à l'armée de terre de prendre la position et de la tenir ; au prix de son sang, pied à pied. Il n'avait jamais entendu parler de cette maudite planète jusqu'à ce qu'on l'y ait débarqué. Et maintenant, c'était un sol sacré, parce que l'ennemi y était aussi venu. L'ennemi, la seule autre race intelligente de la Galaxie, ces monstres cruels, hideux, répugnants.

Le premier contact avait eu lieu vers le centre de la Galaxie, après une lente et difficile colonisation de quelques milliers de planètes : et ç'avait été la guerre, tout de suite. Ils avaient commencé à tirer sans même tenter un accord, une solution pacifique.

Et à présent, planète par planète, il fallait combattre avec acharnement.

Il était trempé jusqu'aux os, couvert de boue, il avait faim et froid, le jour était limpide et balayé par un vent violent, qui lui faisait mal aux yeux. Mais les ennemis essayaient de s'infiltrer et chaque avant-poste était vital.

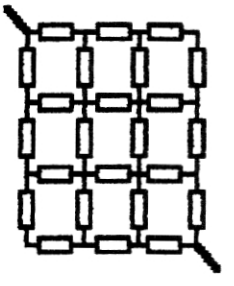
Il était sur ses gardes, le fusil prêt. A cinquante années-lumière de son pays, en train de se battre sur un sol étranger et de se demander s'il y arriverait, à ramener sa peau à la maison. Il vit alors l'un d'eux ramper vers lui. Il visa et prit feu. L'ennemi poussa ce hurlement étrange, terrifiant, qu'ils poussaient tous, et ne bougea plus.

Le vent et la vue du cadavre le firent frissonner. Beaucoup, avec le temps, avaient fini par s'habituer et n'y prêtaient plus attention ; mais lui, non.

C'étaient des créatures repoussantes, avec seulement deux bras et deux jambes, avec la peau d'un blanc répugnant et sans écailles.

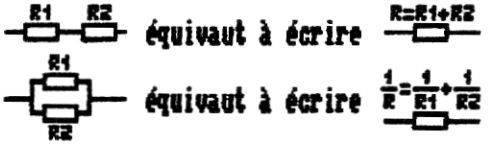
Extrait de
"LE MERAVIGLIE DEL POSSIBILE"
Einaudi, Turin 1959
Ref : Le Grand Livre de l'Espace
p 273

Voilà, ça, c'était pour détendre les nerfs avant le petit jeu sous forme de casse-tête... Mais avant de vous l'exposer je dois tout de même vous signaler qu'il faut absolument maitriser la loi des noeuds des intensités électriques pour pouvoir espérer arriver à faire quelque chose... et encore...
Bon, maintenant, allons-y ! Considérons le circuit électrique suivant :



Chacune des résistances vaut 10 et le courant entre par le coin haut gauche avant de ressortir par l'autre avec bien évidemment la même intensité.
Le but du jeu est de trouver la résistance équivalente à ces associations parallèlo-séries.

Je vous rappelle quand même quelques notions élémentaires sans lesquelles vous ne parviendriez à rien.
Vous devez en effet savoir que :




Voilà, vous voici fin prêts pour affronter ce casse-tête électrique ! Vous aurez la solution dans le prochain numéro où je vous proposerais un autre petit jeu moins vicieux que celui-ci...

Que la force soit avec vous,

Allez et la solution vous trouverez,

A la prochaine,

OffseT

# BIENVENUE dans la rubrique TESTS

ZAP'T'BALLS

A FUTUR'S PRODUCTION — RENDEZ VOUS AVEC LA M.O.R.T.

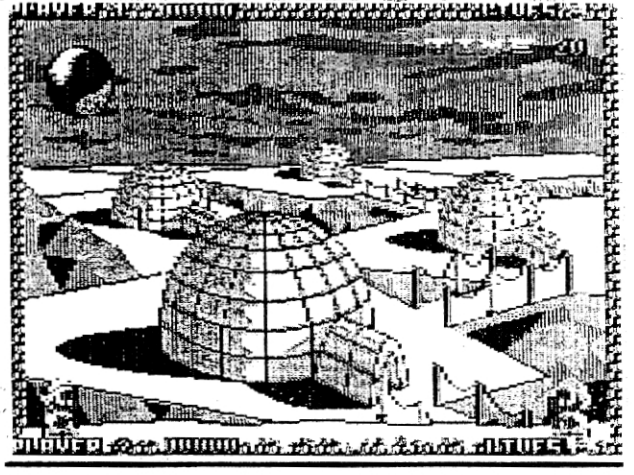
Salut à tous ... Je suis heureux de vous retrouver une nouvelle fois dans cette rub' tant appréciée des joueurs de tam-tam.Mais j'ai quand même un doute sur l'authenticité de votre joie,allez savoir pourquoi. Voici le con-tout-nu de ce numéro:tout d'abord vous aurez droit au test du sublissime ZAP'T'BALL codé par Elmar Krieger, alias Elmsoft (il a codé depuis Super Cauldron et Préhistorik 2). En un deuxième temps, nous critiquerons RENDEZ VOUS AVEC LA MORT programmé par Offset. Pourquoi ?Je vous le dirai là-bas, mais sachez tout de même que ce jeu de rôle n'a rien à voir avec la surprise 93 de l'équipe de Futur's...

Ceci est un cadre dédié à Factice...

FACTICE

Désormais vous n'aurez plus le droit de ne pas connaitre Factice ... De toute façon, vous connaissiez avant que fasse de la pub , n'est-ce-pas ? (hum... Le petit blond à lunettes fait de gros yeux...). C'est très simple (Nagui),il s'agit d'une association qui distribue des démos et des freewares : c'est conforme à cette loi de loi. Leur liste est disponible à leur adresse (ci-dessous), à l'adresse Quasar ou à la mienne. DAMIEN DELURIER 82 Chemin du Charbonnier 69200 VENISSIEUX.

## ZAP'T'BALLS 92%



Vous connaissez tous Pang , le fameux jeu édité par Océan , sur arcade , CPC + et autres machines ? Et bien en voici la réplique (à quelques détails près): ZAP'T'BALLS . La plus grande différence entre Pang et Zap'T'Balls : Zap'T'Balls est à mon goût , mieux.

.....

Ne me demandez pas la notice , je l'ai perdue lors de mon dernier séjour à Tombouctou... Je crois quand même me souvenir d'une partie du scénario ; 4 mondes sont envahis par des boules meurtrières , le seul moyen de les exterminer est d'envoyer 2 petits mecs tirer dessus avec leur harpon .

.....

Pour les ceusses qui ne connaitraient pas (NDMoi:inventée par moi( c'est pas marrant : )" Et , t'es là pour eux !!!") , le but est d'éliminer avec l'aide d'un harpon* toutes les boules qui ne cessent de rebondir ... (* ce harpon peut changer de taille , ce qui accroit la difficulté et la variété du jeu ... )

➡ Deux joueurs peuvent jouer simultanément, cela donne des résultats surprenants :

Engueulements , rigolades ou suicides ...

➡ Je signale pour les ceusses qui n'auraient pas de dentifrice (???) que les grosses bouboules se divisent en 2 boubouboules qui se divisent en 2 boubouboubouboules . Cela nous fait pour éliminer la grosse bouboule de départ 7 boules à toucher. Evidemment (bin voyons ...), le temps est là pour vous empêcher de mener à bien vos bagarres contre les boules envahissantes ... Celui-ci varie d'ailleurs beaucoup .

.....

Présentation du jeu : 4 merveilleux mondes s'offrent à vous . Le deuxième est obtenu à l ' aide du code final du premier , les troisièmes et quatrièmes sont obtenus à l'aide des codes finals des 2 précédents .

Dans chaque monde plusieurs niveaux sont accessibles (de 15 à 40 selon les mondes) eux aussi à l'aide de codes donnés tout les 2 niveaux . L'Intro est très surprenante (agrémentée de 3D !), le Turn-Disk (parce que Zap'T'Balls prend 2 faces !!!) est le plus esthétique et "malin en progamnation" que je n'ai jamais rencontré, le menu des différents mondes accompagné de zolis rasters en Overscan propose un bon nombre d'options , ce qui n'est point désagréable .

## FICHE TECHNIQUE :

L'action se déroule sur tout un écran standard sans le border sauf quand le jeu donne une impression d'Overscan entre 2 niveaux . Les graphismes sont soignés , les musiques, différentes à chaque monde , excellentes , les bruitages sont bien choisis , et l'animation , point fort de ce petit bijoux est heureusement →excellente← : même si une vingtaine de balles rebondissent , tout réagit toujours aussi vite !

Et je peux vous dire que c'est rapide ...Pour parler de la jouabilité , 2 défauts vraiment minimes : à 1 joueur , le Joystick est indispensable , et lorsqu'on est débutant , il est difficile de monter aux échelles , sinon tout est toqué ... Heu ... OK ...

.....

Avant le jugement dernier (T2 ! , T2 ! , T2 !) , je tiens à fournir quelques renseignements sur l'auteur

ELMAR KRIEGER

Ce programmeur est né en Allemagne , c'est donc un Allemand (Non , tu crois ?). Son surnom ? Le voici : ELMSOFT . Il a tout , je dis bien tout , fait : les graphs , le code (il a codé , quoi ...) , et tout ce qui fait d'un jeu un jeu , ainsi que la musique,aidé quand même du Soundtracker et d'un copain pour cette dernière . Mais voilà le + choquant : il a arreté ses études à 16 ans , et compte continuer à travailler sur PC et CPC 128 Ko ! (Çà , je l'ai lu dans la End-Part ...) . Depuis , il a produit 2 Méga-jeux : Super-Cauldron et Préhistorik II .

Que pensez-vous de la bordure ? elle provient de mes prouesses en dessin !!!..

| | | | |
|---|---|---|---|
| GRAPHISMES | : 18/20 | ANIMATION | :19,5/20 |
| MUSIQUE | : 18/20 | BRUITAGES | : 18/20 |
| RICHESSE | : 18/20 | JOUABILITE | : 18/20 |
| DIFFICULTE | : 19/20 | | |

NOTE GLOBALE : 18,5/20 [92%]

# RENDEZ-VOUS AVEC LA M.O.R.T.

Copyright Futur'S 91

Comment ça : " Qu'est-ce-que c'est ?" ? Vous ne connaissez pas ?!? Mais c'est inadmissible !!! Dans le genre des jeux de rôle , il va falloir vous apprendre que "RENDEZ VOUS AVEC LA M.O.R.T" est un jeu FUTUR'S ( nous , quoi ! OFFSET et ZIK mais pas TONY ( moi , quoi ): eux , en fait (j'existais pas. Enfin si mais je ne faisais pas partie du groupe qui a ensuite décidé de fonder QUASAR ). Je ne vais tout de même pas vous raconter ma vie ! ). Bon maintenant que je vous ai appris ce qu'était FUTUR'S je vais vous apprendre que "RENDEZ-VOUS AVEC LA MORT" est un jeu FUTUR'S . (Comment ? Je l'ai déjà dit ?! Bon écoute ZIK ,je le redis , ça fait de la pub' !)

.....

Scénario : Nous sommes au 41ème siècle.Toutes les planètes du système solaire sont colonisées par les Terriens . La technologie avancée leur permet maintenant de partir à la découverte d'autres mondes . Et l'on découvre des planètes habitées : ZORB et KRAPES. Les Zorbiens ont une technologie très peu avancée , mais il s'agit d'un peuple de guerriers . Les Krapus eux sont souvent plus intelligents que l'homme. Mais il leur manque les vaisseaux spatiaux équipés du système de l'hyper-espace . Trés vite Zorb et Krapès demandent leur indépendance vis à vis de l'alliance. La terre comprend, mais trop tard , après avoir accepté avec réticence , que ceux-ci ne voulaient que prendre la place de l'homme ! Le F.A.S.T. est créé , pour se protéger en cas de guerre , et du côté des ennemis , c'est la M.O.R.T. qui est crée. (F.A.S.T.: Forces Armées du Système Solaire Terrien . M.O.R.T.: Milice de l'Ordre pour la Revanche contre la Terre.) Nous sommes en 4033 et le conflit éclate . La Terre perd toutes les batailles car le F.A.S.T. n'ose utiliser la bombe atomique . La Terre perd toutes ses colonies. La M.O.R.T. lance maintenant un ultimatum: la F.A.S.T. doit rendre les armes sous 24H, sinon c'est le massacre assuré... Aujourd'hui 24 Avril 4035 à 19H , il ne reste que 5H pour découvrir les plans d'attaque de la M.O.R.T. ...

J'abrège, j'ai plus de place... Vous devez vous infiltrer dans une centrale spatiale de la M.O.R.T. après une heure de voyage. G:11 A:/ M:18 B:15 R:14

Pour tuer un être, un curseur se déplace et il faut appuyer sur espace quand il est sur un carré... J:15 NG:14,5

TONY

# PERFECTIONNEMENT AU BASIC

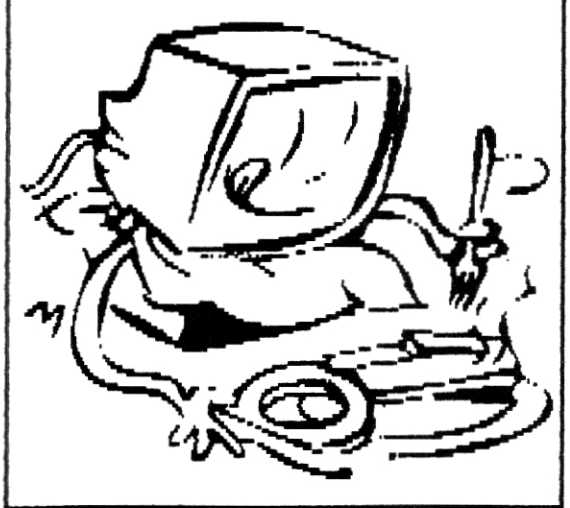
Suite à l'article du numéro précédent vous trouverez dans ce cours un deuxième programme d'ondulation d'écran en Basic. Celui-ci utilise le registre 3 du CRTC. Ce registre permet un réglage fin de la position en abscisse de l'écran ; le prog. envoie des valeurs de 0 à 7, au dela de 7 ce registre fonctionne en positions relatives. Le programme proposé a exactement la même structure que celui du numéro 2 de Quasar (bien sûr !) et c'est pourquoi je ne prendrai pas la peine de le commenter une seconde fois (non mais !). Je précise tout de même que, pour avoir une ondulation idéale, il faudrait combiner les registres 2 et 3 (peut-être que je le ferai pour vous si je suis en forme (ou si vous me le demandez !)).

Mais il y a un autre programme !!! Quelle surprise !! Et oui, encore plus fort que l'ondulation, voici en exclusivité pour Quasar des rasters en Basic (je pense que j'y crois un peu trop) !!! La technique est très simple (façon de parler...). Vous avez sûrement remarqué (mais si, puisque je vous le dis !) que les lignes 80 et 90 du programme sont assez chargées (je veux dire par là que j'ai écrit un maximum de commande sur la même ligne). Et cela pour la seule raison que des commandes placées sur une même ligne sont enchainées plus rapidement que placées sur des lignes différentes. On gagne donc ainsi du temps machine lors de l'exécution (ce qui est important pour pouvoir obtenir de beaux rasters).

Voyons maintenant le programme lui-même. On trouve d'abord un Speed Ink 1,1 qui installe une "remise à jour" des couleurs à chaque balayage écran, comme ça on n'est pas géné par des changements de couleur inattendus. Le Border 1 est là pour l'esthétique et voila ensuite les OUT.

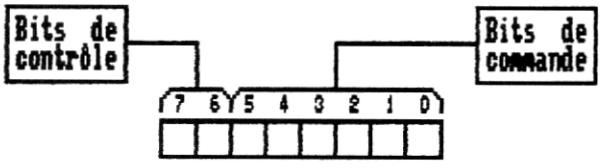
"Gate Array" est le nom d'un des composant de votre ordinateur, ce cher boitier de silicium permet de gérer les interruptions, les connexions de Rom et de Bank, le mode écran et les couleurs !!! Le port d'adressage de ce circuit se situe en &7F00, on lui envoie des ordres à l'aide de deux bits de contrôle et donc six bits de commande. Les bits de contrôle permettent de sélectionner quatre modes, nous allons nous intéresser aux modes 0 et 1. Je vais vous faire un petit schéma pour que vous compreniez au moins quelque chose dans mes articles. Voila tout d'abord un schéma représentant la "disposition" des divers bits qui composent l'octet à envoyer au Gate Array.

Comme je n'ai plus de place ici pour tracer mon pseudo-schéma on va faire une petite translation (je sais vraiment pas quoi dire pour boucher moi !).

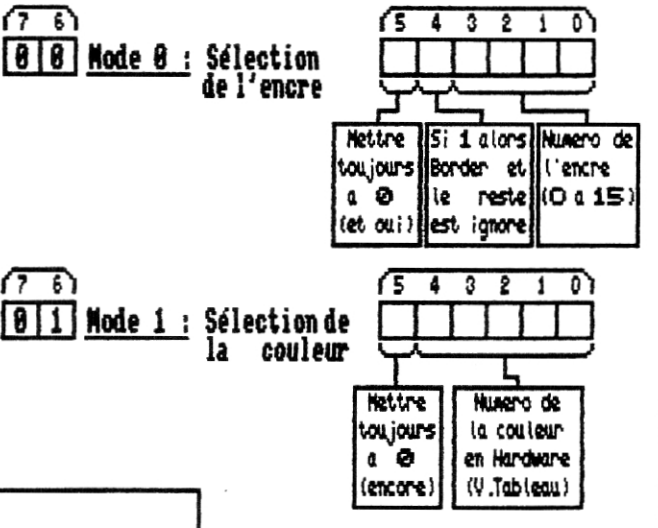


En utilisant les techniques de l'assembleur en Basic : On se régale !!!

Adressage sur le port &7F00 (Gate Array) :




Tant que j'y suis, je vous fais aussi le schéma pour les modes 0 et 1...




Les cases numérotées de 0 à 7 représentent les huit bits qui composent un octet (c'est le codage binaire : 0 ou 1). Si les Bits 6 et 7 sont mis à 0, on sélectionne le mode 0 ; si le Bit 6 est à 1 alors que le 7 est à 0 on valide le mode 1.

Pour faire nos rasters, nous devons indiquer l'encre dont on veut modifier la couleur (mode 0) et quelle couleur on veut afficher. On va d'ailleurs changer cette couleur plusieurs fois pendant le balayage vidéo pour que ce qu'on va faire soit des rasters et pas un bête changement de couleur.

Dans notre exemple, c'est sur la bordure écran qu'agit notre raster et c'est pourquoi on envoie le nombre 16 en &7F00 (16 = 00010000 en binaire) ; il n'est pas nécessaire de renouveller cette "initialisation" sur le mode 0. Les autres commandes OUT envoient les couleurs, on met donc le mode 1 et c'est pourquoi on additionne 64 au numéro de la couleur (les couleurs sont, je le répète, codées en Hardware : il y a des tableaux dans ces pages).

Vous savez donc maintenant faire des rasters (en théorie...). Il reste toutefois un détail (très important) dont je ne vous ai pas parlé : le Call &BD19. C'est un vecteur qui sert à se synchroniser par rapport au balayage écran (ça se passe tout les 50ème de seconde). Après avoir été lancé, ce vecteur rend la main au Basic quand notre cher canon à électrons commence un nouveau rafraichissement de l'écran. Ce vecteur est donc très pratique pour nos rasters. Je tiens à indiquer (et oui !) que ce Call &BD19 est équivalent à la commande FRAME du Basic qui n'est d'ailleurs pas présente sur toutes les générations de Basic.

Ce Call étant placé dans la boucle parmi les Out, on obtient des rasters fixes. C'est à dire que les mêmes couleurs se retrouvent toujours aux mêmes emplacements sur l'écran mais, si l'on veut améliorer l'esthétique de nos rasters, nous avons la possibilité de les faire se déplacer. Il suffit pour cela de supprimer le Call de la ligne 60 ; ainsi, à la fin de la séquence de couleurs, le programme revient tout de suite en ligne 60. De cette manière, nos rasters se meuvent gaiement. D'après les essais que j'ai tenté, c'est en ajoutant deux couleurs (deux Out) dans la liste que l'on obtient le plus beau mouvement. A vous d'essayer pour trouver mieux !

Une autre possibilité nous est offerte puisque, si vous regardez de nouveau les schémas de la page de gauche, vous verrez qu'en modifiant la valeur envoyée au Gate Array par le premier Out, on peut facilement changer l'encre sur laquelle agirons les couleurs des rasters et obtenir des effets intéressants.

Comme il me reste beaucoup de place je vais vous parler brièvement des connections de Banks puisque je sais que certains sont très intéressés. Je ne m'en sors plus de mes tableaux et de mes schémas mais bon, au point où j'en suis ! Vous devez savoir qu'une Bank se connecte uniquement sur la page mémoire qui se situe de &4000 à &7FFF. Une page mémoire fait 16Ko donc vous possédez 4 pages de Banks et 4 en mémoire centrale (128Ko en tout). Les 4 banks sont numérotées de 0 à 3. Voici des exemples pour aider à comprendre...

Connection de la Bank no1 (en &4000):

OUT &7F00,&x11000101

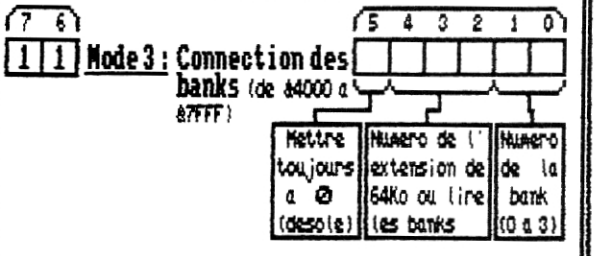
Remise en place de la Ram centrale :

OUT &7F00,&x11000000

Quand on met les bits 2, 3 et 4 à 0, on sélectionne la Ram centrale. Et quand on choisi l'extension 1 (on compte toujours en partant de 0) on connecte les Banks supplémentaires du CPC 6128. Sur un CPC 664 ou 464 il faut ajouter une extension mémoire et si vous ne l'avez pas il vous est alors inutile de comprendre ceci !

(si vous avez tout compris, vous etes tres forts !)

Voici donc le schema du Mode 3...




Bonnes fêtes à tout le monde,

Zik

```
10 '
20 ' Ondulations en Basic
30 '   par Futur's 1991
40 '
50 OUT &BC00,3:OUT &BD00,4
60 OUT &BC00,3:OUT &BD00,5
70 OUT &BC00,3:OUT &BD00,5
80 OUT &BC00,3:OUT &BD00,6
90 OUT &BC00,3:OUT &BD00,6
100 OUT &BC00,3:OUT &BD00,7
110 OUT &BC00,3:OUT &BD00,7
120 OUT &BC00,3:OUT &BD00,7
130 OUT &BC00,3:OUT &BD00,7
140 OUT &BC00,3:OUT &BD00,7
150 OUT &BC00,3:OUT &BD00,7
160 OUT &BC00,3:OUT &BD00,6
170 OUT &BC00,3:OUT &BD00,6
180 OUT &BC00,3:OUT &BD00,5
190 OUT &BC00,3:OUT &BD00,5
192 OUT &BC00,3:OUT &BD00,4
200 GOTO 50
```

```
10 '
20 ' Rasters en Basic
30 '
40 SPEED INK 1,1
50 BORDER 1
60 CALL &BD19:OUT &7F00,16:OUT &7F00,68:OUT &7F00,85:OUT &7F00,93
:OUT &7F00,87:OUT &7F00,95:OUT &7F00,83:OUT &7F00,75:OUT &7F00,67
:OUT &7F00,74:OUT &7F00,78:OUT &7F00,76:OUT &7F00,92:OUT &7F00,86
:OUT &7F00,94:OUT &7F00,82:OUT &7F00,74:OUT &7F00,82
70 OUT &7F00,94:OUT &7F00,86:OUT &7F00,88:OUT &7F00,77:OUT &7F00,
79:OUT &7F00,83:OUT &7F00,64:OUT &7F00,70:GOTO 60
```

Correspondance numéros de couleurs Software et Hardware

| Couleurs | Software | Hardware |
|---|---|---|
| Noir | 0 | 20 |
| Bleu | 1 | 4 |
| Bleu Vif | 2 | 21 |
| Rouge | 3 | 28 |
| Magenta | 4 | 24 |
| Mauve | 5 | 29 |
| Rouge Vif | 6 | 12 |
| Pourpre | 7 | 5 |
| Magenta Vif | 8 | 13 |
| Vert | 9 | 22 |
| Turquoise | 10 | 6 |
| Bleu Ciel | 11 | 23 |
| Jaune | 12 | 30 |
| Blanc | 13 | 0 |
| Bleu Pastel | 14 | 31 |
| Orange | 15 | 14 |
| Rose | 16 | 7 |
| Magenta Pastel | 17 | 15 |
| Vert Vif | 18 | 18 |
| Vert Marin | 19 | 2 |
| Turquoise Vif | 20 | 19 |
| Vert Citron | 21 | 26 |
| Vert Pastel | 22 | 25 |
| Turquoise Pastel | 23 | 27 |
| Jaune Vif | 24 | 10 |
| Jaune Pastel | 25 | 3 |
| Blanc Brillant | 26 | 11 |

Correspondance numéros de couleurs Hardware et Software

| Couleurs | Hardware | Software |
|---|---|---|
| Blanc | 0 | 13 |
| Vert Marin | 2 | 19 |
| Jaune Pastel | 3 | 25 |
| Bleu | 4 | 1 |
| Pourpre | 5 | 7 |
| Turquoise | 6 | 10 |
| Rose | 7 | 16 |
| Jaune Vif | 10 | 24 |
| Blanc Brillant | 11 | 26 |
| Rouge Vif | 12 | 6 |
| Magenta Vif | 13 | 8 |
| Orange | 14 | 15 |
| Magenta Pastel | 15 | 17 |
| Vert Vif | 18 | 18 |
| Turquoise Vif | 19 | 20 |
| Noir | 20 | 0 |
| Bleu Vif | 21 | 2 |
| Vert | 22 | 9 |
| Bleu Ciel | 23 | 11 |
| Magenta | 24 | 4 |
| Vert Pastel | 25 | 22 |
| Vert Citron | 26 | 21 |
| Turquoise Pastel | 27 | 23 |
| Rouge | 28 | 3 |
| Mauve | 29 | 5 |
| Jaune | 30 | 12 |
| Bleu Pastel | 31 | 14 |


# INITIATION AU TURBO PASCAL 3


Nous voilà repartis pour un nouveau cours d'initiation au Turbo Pascal 3 sur CPCs. Si vous êtes des fidèles de cette rubrique et que vous la suivez depuis le numéro 1 de Quasar CPC, en principe, vous devriez être capables de gérer : des boucles, des affichages, et des saisies de données au clavier. Donc, comme promis dans le dernier numéro nous allons cette fois-ci commenter un prog. complet ce qui va nous permettre de revoir toutes les structures que vous connaissez déjà mais aussi de découvrir de nouvelles fonctions et procédures (ne vous inquiétez pas je vais détailler ce que représentent les fonctions et les procédures ; attention à ne pas les confondre...). Bon, on va y aller, vous devriez en principe voir le listing que je vais vous commenter sur la page ci-contre.

Je tiens tout d'abord à vous prévenir que si certaines parties du prog. vous paraissent inutiles voire sans intérêt pratique cela est tout à fait possible car ce listing n'a pas été écrit dans cette optique là mais plutôt pour faire apparaitre un maximum de structures.

C'est parti ! Tout d'abord, je vais vous expliquer ce que fait le programme dans son ensemble. En voici donc l'algorithme simplifié :

-Saisir une phrase de 160 caractères
-La faire précéder de 40 Blancs
 Et la faire suivre d'un blanc.
 (en définitive la phrase fait donc 160+40+1=201 caractère)
-L'afficher en décalage lettre à l'écran autant de fois que le désire l'utilisateur.

Et si on se faisait un petit Base Ball ?

En bref, c'est un scrolling ! Dans la première ligne du programme on donne un nom logique au programme puis on passe aux déclarations...
La première déclaration, avec "Const", configure des des constantes qui seront ensuite utilisées dans le programme comme des variables à ceci près qu'on ne pourra pas modifier leur valeur. On s'en sert principalement lorsque l'on doit utiliser des valeurs (ou des chaines de caractères) de manière répétitive car, puisqu'on ne passe alors pas par des variables on gagne à la fois en vitesse et en place en mémoire.
Dans notre exemple, on déclare ici "titre" qui est le titre qui sera affiché en haut de l'écran pour l'enjoliver mais aussi "blancs" (une succession de 40 espaces) afin de ne pas surcharger l'écriture du prog. et de gagner un tout petit peu en vitesse de calcul.

Ensuite, on passe à la déclaration des variables que vous devez commencer à bien connaitre. Ici, on déclare "phrase" comme étant une chaine de 201 caractères (voir l'algorithme pour la justification de cette taille) et puis x,y et "passage" comme des variables entières.

C'est maintenant qu'arrive notre programme proprement dit. On commence tout d'abord par effacer l'écran avec un petit "Clrscr" ; ça fait plus propre. Puis on passe à la mise en place du titre. On positionne le curseur avec un "GotoXY", commande que vous connaissez bien. En revanche, vous devez moins comprendre l'expression dont je me sers pour calculer la position et je vais donc la détailler.
Le but du jeu est de centrer le titre en haut de l'écran qui fait 80 caractères de large. En réfléchissant un petit peu on trouve la formule mathématique suivante :

Posit. en X = (80-longueur phrase)/2
ou encore :
Posit. en X = 40-(longueur phrase/2)

Nous allons maintenant traduire ça en Turbo Pascal. La fonction permettant de calculer la longueur d'une phrase, c'est "Length(var)". D'où la formule en Turbo :

Posit. en X = (80-Length(Titre))/2
ou encore
Posit. en X = 40-(Length(Titre)/2)

Mais vous constaterez qu'il y a encore un petit problème car si vous effectuez ce calcul il se peut que le résultat ne soit pas entier (si on a Length(Titre) égal à un nombre impair). Dans ce cas là il faudra arrondir le résultat du calcul afin d'obtenir un nombre entier (dans le cas contraire vous hériteriez d'un magnifique Runtime Error en exécutant votre petit programme). Heureusement il y a "Round", Rahound !
Car lorsque l'on fait Round(nombre) cette fonction nous renvoie en récompense un nombre entier arrondi... En définitive on retrouve la formule mathématique de notre programme :

Posit. en X = Round(40-Length(Titre)/2)

C'est'y pas beau tout ça ? Je pense que la suite de l'affichage ne vous pose aucun problème. Je vous rappelerai juste que "Writeln;" seul est utilisé pour sauter une ligne.

Nous voici enfin arrivés à la saisie de la phrase, je pense qu'il n'y a là aucune difficulté puisque "Readln" est justement le sujet du cours du numéro 3. Toutefois,

je préfère vous rappeler que "Read(var)" équivaut à "INPUT"";var" comme ça j'aurais la conscience tranquille.

Si on poursuit dans le programme on a d'autres affichages qui ne posent aucun problème de compréhension et une autre saisie tout aussi simple que la précédente.

Ensuite, on arrive à l'adaptation de la phrase saisie en vue du scrolling. Là aussi, vous devriez ne suivre sans problème car les constantes s'utilisent comme des variables. Dès lors je vous ai ajouté un petit test clavier histoire de revoir l'utilisation du Repeat ...Until et pour vous faire découvrir la fonction booléenne Keypressed. Pourquoi booléenne ? Eh bien tout simplement car elle renvoie 0 (false) si aucune touche n'est pressée lors de son émission et 1 (true) si une touche quelconque du clavier est enfoncée. Vous voyez donc que le Repeat Until Keypressed va faire patienter votre CPC jusqu'à ce que vous appuyiez sur une touche.

Dès qu'une touche est enfoncée on efface de nouveau l'écran, on replace le titre en haut puis on passe aux choses sérieuses... Pour plus de clarté, je vais d'abord vous expliquer par quel procédé ce programme fait son scrolling. En fait, c'est très simple et très logique ; si on veut faire un scrolling de 40 caractères de large, au premier affichage, on affiche les 40 premières lettres de la phrase. Ensuite, pour simuler un décalage vers la gauche, au second affichage, on va toujours écrire 40 lettres de la phrase, mais, cette fois-ci, en commençant par la seconde lettre de la phrase et ainsi de suite jusqu'à la dernière qui, vous le comprenez, devra être un espace afin qu'elle ne laisse pas de trace à l'écran lors de son évacuation par la gauche. De plus, pour que le scrolling arrive par la droite on aura préalablement ajouté 40 espaces au début... Voilà, le concept est relativement simple. Sa traduction en Turbo ne pose quant à elle aucun réel problème.

On a tout d'abord une boucle "For To" qui contrôle le nombre de passages du scrolling en fonction du choix de l'utilisateur lors de la saisie précedente et puis on a une autre boucle "For To" qui va contrôler le scrolling en décalant à chaque passage la position de départ dans la lecture de la phrase. Cette boucle contient l'affichage qui gère ce décalage à l'aide de la fonction "Copy" dont je vais vous détailler l'utilisation.

En fait, cette fonction se comporte exactement le MID$ de notre cher Basic. Sa syntaxe est la suivante :

Copy(var,position,longueur).

Dans le programme vous voyez donc bien que l'on affiche toujours 40 caractères mais en décalant à chaque boucle le point de départ de l'affichage d'un caractère vers la droite ce qui provoque donc un décalage de la phrase vers la gauche à l'écran.

Ça y est, j'espère avoir été suffisamment clair. Mais en fait, dans ce prog., vous connaissiez déjà pratiquement toutes les procédures ; les trois seules nouveautés sont les fonctions : Keypressed, Length et Copy.

Comme promis au début de l'article je vais donc maintenant vous expliquer la différence essentielle qui existe entre une procédure et une fonction. En Basic, vous connaissez certainement les GOSUB...RETURN qui sont communément appelés des sous-programmes alors qu'il s'agit en fait bel et bien de procédures. Une procédure est donc tout simplement un sous-programme. Par exemple Clrscr, GotoXY, Write, etc... sont des procédures. Une fonction, c'est différent. Il s'agit d'un programme qui va effectuer des calculs à partir d'une variable. Il sera donc nécéssaire de la gérer comme une variable. C'est à dire par l'intermédiaire d'une procédure ou bien en attribuant le résultat du calcul à une variable.

Si c'est encore un peu confus, ne vous acharnez pas dessus car nous verrons cela plus en détail car on peut également créer ses propres fonctions et procédures comme en Basic avec le GOSUB...RETURN d'une part et les DEF FN... d'autre part. Nous reprendrons donc ce prog. en y utilisant un maximum de procédures et fonctions.

Essayez donc de bien maitriser ce prog. pour la prochaine fois car avec les procédures et les fonctions ça risque d'être un peu compliqué...

OffseT

```
Program Animation;

Const Titre='A N I M A T I O N  D'' U N E  P H R A S E';
      Blancs='                                        ';
Var   phrase:String[201];
      x,y,passage:Integer;

Begin
 Clrscr;
 GotoXY(Round(40-Length(Titre)/2),1);Write(Titre);
 GotoXY(32,3);Writeln('SAISIE DE LA PHRASE');
 Writeln;
 Writeln;
 Writeln('Entrez votre phrase (2 lignes maxi) puis tapez RETURN');
 Writeln;
 Readln(phrase);
 Writeln;
 Writeln;
 Write('Combien de passages desirez vous ? ');Readln(passage);
 Phrase:=Blancs+phrase+' ';
 Writeln;
 Write('C''est parfait, on va pouvoir y aller... Pressez une touche S.V.P....');
 Repeat Until Keypressed;
 Clrscr;
 GotoXY(Round(40-Length(Titre)/2),1);Write(Titre);
 For y:=1 To passage Do
  For x:=1 To Length(phrase) Do
  Begin
   GotoXY(20,12);Write(Copy(phrase,x,40));
  End;
End.
```


# PERFECTIONNEMENT AU TURBO PASCAL 3


Avec le retour de Quasar CPC sur papier je vais pouvoir me permettre de développer cette rubrique...
En effet, les programmes ASCII du Turbo Pascal prennent beaucoup de place sur les disquettes et j'avais donc du les écourter. Bref, cette fois-ci, nous allons aborder un gros programme, un pur et dur !

En fait, dans les numéros 2 et 3, je vous avais fait des cours un peu théoriques, eh bien maintenant, on passe à la pratique ! Je vous propose donc de construire ensemble un programme de gestion de logithèques qui va nous permettre de revoir l'ensemble des structures dont je vous ai déjà parler mais aussi de les approfondir et d'en découvrir quelques autres... En bref, on a du pain sur la planche !

Le "petit" programme que je vous ai préparé s'appelle LogiBase et nous allons tout d'abord voir quelles y sont les options disponibles avant de nous attaquer à sa programmation. Ce prog. permettra donc :

- Le listage des enregistrements (c'est la moindre des choses)
- L'ajout d'enregistrements (tout de même)
- La suppression d'enregistrements (Au cas où...)
- La sauvegarde des enregistrements (ça ne parait plus prudent...)
- Le chargement des enregistrements (forcément)
- L'impression des enregistrements (c'est plus classe !)

Comme vous le voyez, ça risque d'être long, je dirais même plus, c'est très long !!! Le programme approche les 500 lignes et vous comprendrez donc qu'il m'est impossible de vous le donner dans son intégralité ; c'est pourquoi je vais vous le commenter petit à petit et ainsi, au fil des numéros de Quasar vous pourrez le reconstituer. Toutefois, si vous voulez absolument l'avoir sous les yeux dans son intégralité n'hésitez pas à écrire à la rédac. et nous vous en enverrons le listing illico-presto contre un tout petit timbre à 2F80...

Bon, nous avons assez bavassé comme ça, passons aux choses sérieuses : le programme ! Nous allons donc commencer (ce qui me parait logique) par la déclaration des types, des constantes et des variables avant d'aborder les fonctions et les procédures de base.

Comme la dernière fois nous avions vu le type Record, je me suis dit, pourquoi ne pas faire un programme qui utilise les records... Mais, comme je vous en avais déjà un peu parlé, il faut alors utiliser des tableaux de records ce qui peut paraitre rebuttant au départ mais qui se révèle en fait très simple d'utilisation (Attention, j'ai dit simple, pas souple !). Plutôt que de vous commenter le programme ligne à ligne (vous êtes bien assez grands) je vais vous expliquer les différentes déclarations indépendamment les unes des autres.

## DECLARATION DU TABLEAU DE D'ENREGISTREMENTS (Records)

Ce tableau servira, comme vous vous en doutez certainement, à stocker les données. Pour éviter une surcharge d'écrire j'ai préféré le déclarer en deux fois.
Tout d'abord, on configure un type record, ici Champs, pour lequel j'ai prévu 4 champs : le titre du soft, son type (j'ai ici utilisé style car type est un mot réservé), sa taille et ses référence (le numéro du disc).

Pour éviter les conflits de type, j'ai préféré déclarer tous ces champs sous forme de chaines de caractères plus ou moins longues. On obtient donc une déclaration qui est l'application directe du dernier cours :

```
Type Champs=Record Titre:String[25];
                   Style:String[15];
                   Ko:String[3];
                   Ref:String[5];
     End;
```

Ensuite, on va stocker tout un tas d'enregistrements de ce type dans un tableau de "maxi" cases. Maxi est une constante que vous pourrez modifier à votre guise (dans la limite de la mémoire disponible). La valeur 50 est une moyenne ; suivant que vous compiliez en Ram ou sur Disc elle pourra varier jusqu'à 600.

La déclaration du tableau de records se fait le plus simplement du monde puisque le type record que l'on vient de créer (Champs) s'utilise comme n'importe quel autre type. La différence se fera sentir lors de l'utilisation de ce tableau. La déclaration est donc tout simplement :

```
Var fiche:Array[1..maxi] Of Champs;
```

Je pense que vous m'avez tous suivi jusque là ! On déclare ensuite une variable annexe avec le type Champs ; elle nous servira à faire des transferts de données plus directs puisqu'elle nous évitera de repasser pas les champs à chaque attribution... On en reparlera.

Les autres déclarations ne posant aucun problème, je vais me contenter de les énumérer en expliquant brièvement leur rôle :

- Les types Chaine et Chaine25 serviront pour les fonctions pour lesquelles, vous le savez sûrement, il est impossible d'utiliser directement des types qui sont des composées d'autres types (String est en effet la composée de plusieurs Char).
- La variable fin servira elle à localiser la position du dernier enregistrement.

## PASSONS MAINTENANT AUX FONCTIONS

Les deux fonctions ci-dessous serviront dans la suite du programme aussi, n'y cherchez pas de logique pour l'instant. Pour commencer, on crée une fonction Space ; celle-ci nous renvoie une chaine de caractère de "a" espaces ; il y a donc un passage par valeur avec la variable "a" et la fonction proprement dite est déclarée comme étant une chaine de 255 caractères maxi. En bref, c'est l'équivalent du Space$(x) du Basic, d'ailleurs l'utilisation sera la même.

```
Program LogiBase;

Type Champs=Record Titre:String[25];          (* Declaration des types *)
                   Style:String[15];
                   Ko:String[3];
                   Ref:String[5];
                   End;
     Chaine=String[255];
     Chaine25=String[25];

Const maxi=50;                                (* Declaration de maxi *)

Var fiche:Array[1..maxi] Of Champs;           (* Declaration des variables *)
    courant:champs;
    fin:integer;

Function Space(a:Integer):chaine;             (* Fonction Space *)
Var x:Integer;
    y:chaine;
Begin
y:='';
For x:=1 To a Do
 y:=y+' ';
Space:=y;
End;

Function Maj(chain:chaine25):chaine25;        (* Fontion Maj *)
Var x:Integer;
    y:chaine25;

Begin
y:='';
For x:=1 To Length(chain) Do
 y:=Upcase(Copy(chain,x,1));
Maj:=y;
End;

Procedure Init;                               (* Procedure Init *)
Var x:Integer;

Begin
For x:=1 To maxi Do
Begin
 fiche[x].titre:='                         ';
 fiche[x].style:='               ';
 fiche[x].ko:='   ';
 fiche[x].ref:='     ';
End;
End;

Procedure Entete;                             (* Procedure Entete *)
Begin
Clrscr;
GotoXY(27,1);Write('- - - L O G I B A S E - - -');
GotoXY(69,24);Write('Quasar CPC');
End;

... A Suivre...
```

Ensuite, vous avez la fonction Maj qui est elle une chaine de 25 caractères et qui renvoie la chaine de caractères qui lui à été transmise grâce à un passage par valeur après l'avoir mise en majuscule. En effet, la fonction intégrée UpCase n'agit que sur les Char ce qui était génant. Cette fonction est donc l'équivalent du Upper$(a$) du Basic.

Pour information, si vous avez des doutes sur l'utilisation de Length et de Copy jetez un coup d'oeil dans la rubrique d'initiation au Turbo Pascal.

## VOYONS VOIR CE QUI CE PASSE DU COTE DES PROCEDURES

Là, rien de bien méchant, il n'y a en effet ni passages par valeur, ni passages par variables.

La procédure Init sert à initialiser le tableau d'enregistrements ; vous pouvez y voir les modes d'attribution de ce tableau.

La procédure Entete affiche tout simplement le titre du soft (LogiBase) en haut de l'écran et Quasar CPC en bas et à droite. J'ai créé cette procédure car cet affichage est répété pour chaque option du logiciel.

Bon, eh bien je crois que c'est fini pour cette fois-ci. J'espère que vous avez bien compris le principe des différentes déclarations. Nous prendrons la suite de ce prog la prochaine fois.

J'attaquerai le menu principal et l'option lister, vous verrez ainsi concrètement l'utilisation des ces fonctions et procédures.

Je vous rappelle que le programme est disponible dans sa totalité en écrivant à la rédac. ; je vous conseille de vous le procurer car je pense que vous comprendriez mieux tout ce qui précède.

OffseT

# INITIATION A L'ASSEMBLEUR

Je vous préviens tout d'abord que vous pouvez oublier ce que je vous ai dit à la fin de l'article d'initiation du numéro précédent puisque je vais essayer de vous apprendre (enfin j'espère...) comment afficher un sprite (ou lutin pour les anglophobes). Je pense d'ailleurs que c'est plus utile et intéressant que les flags P/V, P et autres M (vive les maths !).

Commençons tout d'abord par la commande LDIR qui va nous servir. Cette instruction fait parti des quelques mnémoniques intégrées, c'est à dire que LDIR est équivalente à une succession de commandes élémentaires (qui prendraient beaucoup plus de temps machine utilisées directement). Je signale que LDIR signifie block LoaD & IncRement. Mais à quoi sert-ce ? me direz-vous d'un air hum, je m'égare... Cela sert tout simplement (bof !) à copier un bloc d'une adresse mémoire vers une autre (c'est étonnant ! N'est-il pas ?) ; on précisera également la longueur de ce bloc. Pour utiliser cette commande (enfin !) il faut préciser l'adresse mémoire du bloc à copier dans HL, l'adresse de destination de notre copie dans DE et la longueur du bloc (en octet) dans BC.

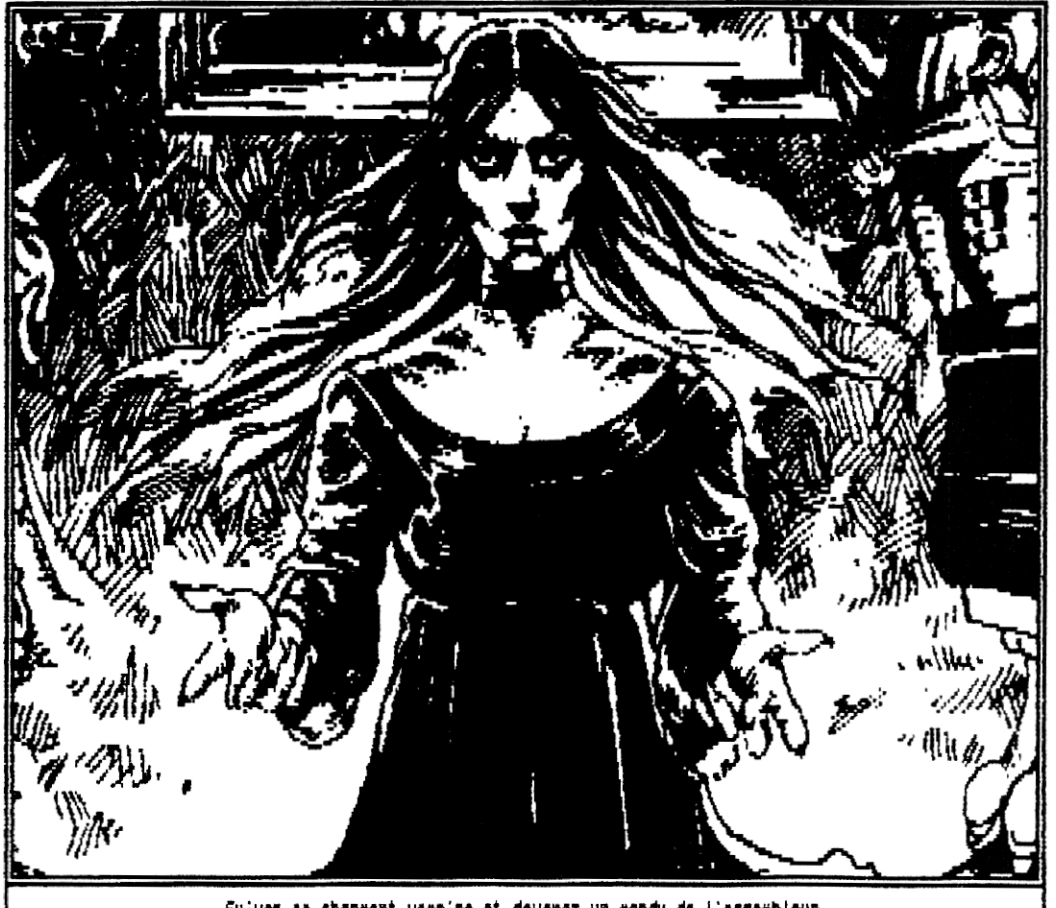
Comme je veille consciencieusement à ce que vous compreniez tout ce que je raconte voici un exemple :

```
LD HL,&4000
LD DE,&C000
LD BC,&1000
LDIR
RET
```

Ce fabuleux exemple copie un bloc de &1000 octets qui se situe en &4000 à l'écran en commençant par le bord haut-gauche car, comme chacun sait l'écran commence en &C000 et finit à l'adresse &FFFF (ce qui peut nous aider quelque peu pour afficher notre sprite).

Pour afficher un sprite il faut qu'il soit stocké en mémoire et que l'on connaisse ses dimensions (qui sont ici en octets). Je vous donne tout de suite le programme avec LDIR qui affiche un sprite stocké en &4000 en haut à gauche de l'écran (donc en &C000) :

```
        LD   DE,&C000
        LD   HL,&4000
        LD   B,50
BOUCLE  PUSH BC
        PUSH DE
        LD   BC,10
        LDIR
        POP  DE
        EX   DE,HL
        CALL &BC26
        EX   DE,HL
        POP  BC
        DJNZ BOUCLE
        RET
```



Suivez ce charmant vampire et devenez un mordu de l'assembleur

Vous avez sûrement remarqué le CALL &BC26. Vous connaissez le CALL mais pas le vecteur &BC26 et c'est là votre tort ! Ce vecteur sert à calculer l'adresse de la ligne au dessous sur l'écran de l'adresse qu'HL contient. Je m'explique : vous mettez dans HL l'adresse écran et après le &BC26, HL contient l'adresse de l'octet qui, sur l'écran est en dessous du précédent. Pour information, le vecteur &BC29 calcule l'adresse au dessus.

Mais &BC26 agit sur HL or on utilise DE pour pointer sur l'écran, donc on utilise EX DE,HL qui échange les contenus de DE et de HL : le contenu de DE va dans HL et vice versa.

Une autre chose qui puisse vous paraitre curieuse est que je sauve DE et pas HL (Push et Pop). En effet, LDIR modifie des registres. Ainsi après un LDIR :

DE=DE+BC
HL=HL+BC } Dans notre exemple BC=10
BC=0

Donc, toujours dans notre exemple, après le LDIR, le registre HL pointe juste à la bonne adresse ; mais par contre, pour afficher la ligne en dessous de la précédente il faut que DE pointe sur la gauche de celle-ci au moment ou l'on veut faire le calcul de la ligne suivante, c'est à dire avant le CALL &BC26 (précédé du EX). Et c'est pourquoi on sauve DE quand il contient la bonne adresse, c'est à dire avant le LDIR.

Ce programme affiche donc 50 fois une ligne de 10 octets de large en descendant à chaque fois d'une ligne grâce au fameux &BC26.

Voyons maintenant le programme qui se trouve à droite de cette page. Ce programme affiche un sprite qui se déplace de gauche à droite puis de droite à gauche et ainsi de suite. J'espère que les commentaires que j'ai rajouté vous aideront efficacement pour comprendre ce programme qui est assez complexe puisqu'il s'automodifie.

L'automodification consiste à faire modifier les commandes d'un programme par lui-même lors de son exécution. Pour remplacer des commandes par d'autres il faut savoir en combien d'octets sont codées les instructions à écrire, la manière la plus simple étant d'observer à l'assemblage. La structure du programme est la suivante :

- Le programme principal affiche le sprite et calcule la prochaine adresse où l'on doit l'afficher, si c'est la droite de l'écran (ou bien la gauche selon la dernière automodif) on va aux sous-programmes d'automodif pour la prochaine exécution de la boucle principale (qui contient un test clavier sur la touche espace).

- On trouve ensuite les sous-progs d'automodif et une routine d'affichage en HL du contenu de DE avec son BC26 (merci ACPC).

Les sous-programmes d'automodif écrivent soit un DEC HL soit un INC HL (ces commandes font un seul octet), ils changent également le test qu'il faut faire sur HL (CP exprimé avec une valeur prend 2 octets : le second est la valeur de comparaison) et enfin ils modifient l'adresse du sous-prog d'automodif où il doit sauter (un CALL est codé par 3 octets : les deux derniers codent l'adresse où aller (les adresses sont toujours stockées à l'envers, ainsi &C000 sera codé &00 puis &C0)). Au risque de me répéter, je vais vous parler des deux mnémoniques ORG et EQU que possèdent tous les assembleurs. ORG sert à indiquer à partir de quelle adresse mémoire on veut que son programme soit assemblé. EQU doit être précédé d'un nom quelconque et doit être suivi d'un nombre qui peut-être une adresse ou n'importe quoi d'autre (même un autre nom). Cela permet d'utiliser un nom à la place d'un nombre, il est ainsi plus facile de modifier un nombre dans un programme.

Rendez-vous au prochain numero... Zik

```
SPR     EQU     &0000       ; Adresse de stockage du sprite
HAUT    EQU     16          ; Hauteur du sprite en octets
LARGE   EQU     20          ; Largeur du sprite en octets
        ORG     &4000       ; Programme à assembler en &4000
        LD      A,2         ;- A contient le numéro du mode d'
        CALL    &BC0E       ;- écran désiré et &BC0E le met.
        LD      HL,&C000
DEBUT   LD      B,&F5       ;- Cette petite boucle correspond
ATTEND  IN      A,(C)       ;- au vecteur &BD19 qui attend la
        RRA                 ;- fin du balayage de l'écran pour
        JR      NC,ATTEND   ;- rendre la main...
        CALL    AFFICHE     ; Copie du sprite SPR à l'adresse HL
MODIF1  INC     HL
        LD      A,L         ; On teste si on est à droite ou à
MODIF2  CP      80-LARGE    ; gauche de l'écran
MODIF3  CALL    Z,RECTIF1
        XOR     A           ;- &BB1B teste tout le clavier et ren-
        CALL    &BB1B       ;- voie le code ASCII d'une touche
        CP      32          ;- éventuellement enfoncée dans l'Accu
        JR      NZ,DEBUT    ;- 32 = Espace. Si A=32 on passe au RET
        Ret
RECTIF1 PUSH    HL
        LD      A,(DATA1)   ; On écrit un DEC HL à la place du
        LD      (MODIF1),A  ; INC HL en MODIF1 (Voir texte)
        XOR     A           ; On écrit un 0 après le CP de
        LD      (MODIF2+1),A ; MODIF2 (Voir texte)
        LD      HL,RECTIF2  ; On "poke" l'adresse de RECTIF2
        LD      (MODIF3+1),HL; après le CALL de MODIF3 (V.T.)
        POP     HL
        Ret
RECTIF2 PUSH    HL
        LD      A,(DATA2)   ; On écrit un INC HL à la place du
        LD      (MODIF1),A  ; DEC HL en MODIF1 (Voir texte)
        LD      A,80-LARGE  ; On écrit 80-LARGE après le CP de
        LD      (MODIF2+1),A ; MODIF2 (Voir texte)
        LD      HL,RECTIF1  ; On écrit l'adresse de RECTIF1
        LD      (MODIF3+1),HL; après le CALL de MODIF3 (V.T.)
        POP     HL
        Ret
AFFICHE PUSH    HL          ; Revoici à peu près la routine d'
        LD      DE,SPR      ; affichage que l'on a vu dans la
        LD      B,HAUT      ; première partie de ce cours.
BOUCLE  PUSH    BC          ; La seule différence est que l'on
        PUSH    HL          ; considère que LDIR écrit en HL
        LD      BC,LARGE    ; et lit en DE jusqu'à ce que l'on
        EX      DE,HL       ; ait à utiliser cette commande que
        LDIR                ; l'on encadre alors de deux jolis
        EX      DE,HL       ; EX DE,HL (qui inversent les con-
        POP     HL          ; tenus de DE et de HL).
        CALL    BC26        ; -> On appelle notre BC26
        POP     BC          ; Et on continue tant qu'il y a
        DJNZ    BOUCLE      ; des lignes à afficher...
        POP     HL
        Ret
BC26    LD      A,H         ; Notre fameux (et adoré) BC26...
        ADD     A,8         ; Encore et toujours
        LD      H,A         ; Ça continue
        RET     NC          ; On revient ou pas ?
        LD      BC,&C050    ; Ici on est resté...
        ADD     HL,BC       ; et puis...
        Ret                 ; on repart
DATA1   DEC     HL          ; On connait pas la valeur Hexa des
DATA2   INC     HL          ; commandes -> on écrit pour lire après
```

Voici le programme d'exemple
Si vous voulez plus d'explications sur
l'automodification (ou sur autre chose) ecrivez-le nous


# PERFECTIONNEMENT A L'ASSEMBLEUR


Nous revoici réunis pour un nouveau cours d'assembleur sur la gestion de l'overscan. La dernière fois nous avions vu à quels registres du CRTC il fallait s'adresser afin d'obtenir un écran reformaté en overscan.

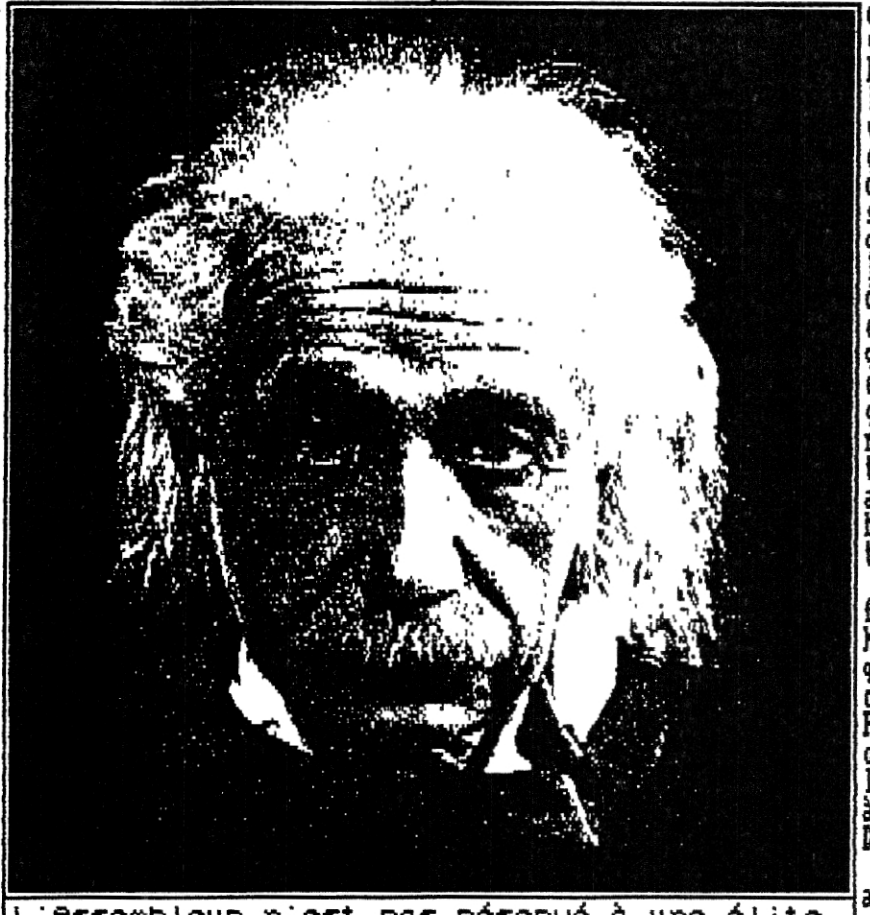
Ça, c'était la partie hardware de l'overscan mais une fois que l'écran est installé, il faut le gérer... Et c'est donc la gestion des écrans overscan que nous allons voir cette fois-ci.

Comme vous vous en êtes sûrement aperçu, la mise en place d'un écran overscan passe obligatoirement par un écrasement total ou partiel du système. Dès lors, lorsque l'on veut faire des programmes avec un overscan il faut soit reconfigurer les interruptions (passer en mode IM 2), soit déplacer la pile et, dans tous les cas, il faut se reprogrammer entièrement tous les vecteurs de gestion de l'affichage (BC26, BC29, etc...). Donc, il y a du boulot !

Je vais traiter ici 2 gestions possibles de l'overscan et, pour chacune d'entre-elles, vous expliquer comment doit se faire la reconfiguration du système.

Tout d'abord, envisageons le cas d'un écran de 32Ko logé de &C000 à &FFFF et de &0 à &4000. Dans ce cas là, les avantages sont assez nombreux, surtout si on débute ; en effet, on écrase pas les vecteurs (la plupart d'entre-eux, notamment les BB1B, BB06 et BB18, restent utilisables) ce qui facilite un peu la programmation, la pile reste intacte, l'amsdos est toujours accessible, etc... Mais, car il y a toujours des mais, du fait de la partie de l'écran logée de &0 à &4000, les pauvres petites interrutions IM 1 en &38 se retrouvent au beau milieu de l'écran... gênant non ? Il y a donc deux solutions pour remédier à ce problème (j'en utilise parfois d'autres mais elle sont scabreuses) : soit on fait un DI et on en parle plus (mais là on devra carrément se passer des interrutions ce qui est souvent très embêtant !), soit on se met franchement en mode vectorisé (IM 2, c'est le mode d'interruption où l'on peut choisir l'adresse de l'interruption). Mais dans ce dernier cas il y a problème de compatibilité entre CPC et CPC+ (qui n'est pas insurmontable du tout) c'est pourquoi le prochain article sur l'assembleur y sera entièrement consacré... patience...

Ensuite, si on a un écran logé de &8000 à &FFFF (c'est la configuration que je préfère même si elle pose quelques problèmes pendant la programmation). Là, plus de système du tout ! Adieu les vecteurs, ciao l'amsdos bye bye la pile ! Vous le voyez, avant de pouvoir utiliser cette configuration, il faut déjà assez bien maitriser l'assembleur puisqu'on doit se refaire la gestion du clavier, du FDC (pas toujours), etc... Bref, encore de quoi faire un article entier consacré à tout ça ! En effet, il faut, reloger la pile, se retaper les accès au PPI, au FDC, au Gate Array, bref, ce sera pour plus tard !



L'Assembleur n'est pas réservé à une élite.

Maintenant qu'on a fait un rapide tour d'horizon des problèmes que pose l'assembleur, on va pouvoir y aller ! Pour plus de simplicité commencez par travailler avec un écran logé à partir de &C000 et un DI...

Nous allons à présent afficher un sprite sur un écran overscan. La plupart d'entre-vous ont déjà essayé et se sont écriés... Mer.. Ça foire complètement ! Et pour cause, il faut, dès que l'on a reformaté l'écran en largeur se refaire des BC26 et des BC29 adaptés car ceux du système sont prévus pour un écran de 80 octets de larges... si on les utilise malgré tout on obtient un affichage avec des lignes en décalé... Donc, nous allons voir comment il faut s'y prendre pour refaire les routines de calcul des lignes inférieures et supérieures.

Mais comme je ne sais pas si vous connaissez bien la structure d'un écran (overscan ou standard) nous allons commencer par voir ça ensemble ! Etudions tout d'abord la configuration d'un écran standard (80 octets par 200) et voyons comment ça marche. Allumez votre CPC et faites

L'assembleur, au début, c'est pas évident, mais on fini par y arriver...

Seul l'assembleur permet d'avoir un réel aperçu des capacités du CPC...

un petit POKE &C000,&FF... Résultat des courses, un magnifique petit octet s'est affiché dans le coin en haut à gauche de votre écran ! Imaginons maintenant que nous voulions décaler cet octet vers la droite (qui a dit on fait RRA ?), eh bien il suffit d'incrémenter son adresse. D'où le fameux POKE &C001,&FF qui nous affiche un autre octet juste à droite du précédent... Bon, jusque là c'est simple... Maintenant, si on continue à incrémenter l'adresse (allez, disons 78 fois de plus (au hasard)) il va bien arriver un moment où on va se retrouver contre le bord droite l'écran (dans le cas contraire, dites-vous qu'il doit peut-être y avoir un léger problème... si, si !). Dès lors, si on ajoute alors encore un... Faites un POKE &C050,255 (&C050=&C000+80) et observez... L'octet se retrouve 8 lignes en dessous du premier que nous avions affiché. Dès lors, on peut renouveller l'opération et notre petit octet va continuer à divaguer de gauche à droite et à sauter de 8 lignes en 8 lignes et ce jusqu'en bas de l'écran. Si on continue alors à incrémenter l'adresse écran le curseur réapparait en haut à gauche mais cette fois-ci juste une ligne en dessous du &C000... etc...

Si vous n'avez pas bien compris tout ça tapez le programme suivant et vous verrez exactement ce qui se passe :

```
10 ' L'écran et ses octets...
20 FOR x=49152 TO 65535
30 ' 49152=&C000 et 65535=&FFFF
40 POKE x,255:NEXT
50 END ' Ca fait plus propre !
```

Bon essayons maintenant de voir ce qu'il faut faire pour passer d'un seul coup d'un seul d'une ligne à celle du dessous. Il se trouve que l'écran fait &4000 octets de long (16384 si vous préférez la barbarie), or, lorsque l'on incrémente notre petite adresse (comme nous l'avons vu dans le paragraphe précédent) il descend de 8 lignes en 8 lignes d'où le calcul savant du nombre qu'il faut ajouter à l'adresse pour descendre d'une ligne et d'une seule : x=&4000/8=&800. Donc si vous faites un poke en &C800 vous vous retrouverez à l'adresse située juste en dessous de &C000. Mais attention !!! Ce calcul ne fonctionne que dans le cas où vous n'êtes déjà pas à une adresse correspondant à un ligne multiple de 8 !!!

En effet, dans le cas où vous êtes sur une ligne qui est multiple de huit, si vous ajoutez &800 vous sortez de l'écran et vous vous attaquez aux restarts... Pour obtenir une adresse correcte il vous faudra ajouter &C000 (vous vous retrouvez ainsi de nouveau sur l'écran mais huit lignes au dessus de l'adresse escomptée) puis enfin 80 (pour redescendre du 8 lignes). Oui, je sais, je ne suis pas très clair, mais c'est franchement pas évident à expliquer ! Donc, pour plus de clarté, voici la routine du BC26 (pour un écran de 80 octets de large) :

```
;
; Calcul de la ligne inférieure
;

BC26    LD A,H
        ADD A,8
        LD H,A
        RET NC
        LD BC,&C000+80
        ADD HL,BC
        RET
```

Oups, j'ai oublié de vous dire qu'il s'agit là d'un BC26 sur HL et qui modifie BC, de plus il ne tourne que pour un écran logé en &C000 puisqu'il utilise la carry pour détecter le débordement. Libre à vous de le modifier selon vos besoins. Ensuite, pour l'adapter à n'importe quelle largeur d'écran, si vous avez suivi, cela ne devrait pas vous poser de problème ; il suffit de remplacer le 80 par la nouvelle largeur d'écran en octet (Reg1*2). Bon, eh bien je vois qu'il ne me reste plus beaucoup de place et je vais donc vite passer à la gestion de l'écran overscan proprement dit.

Ce qu'il faut savoir, lorsque l'on passe en overscan, c'est qu'on ne gère pas un écran de 32Ko mais plutôt 2 écrans de 16Ko (généralement un en haut et un en bas, non, non, je ne blague pas !). Vous voyez donc logiquement qu'il vous faudra deux BC26 différents suivant que vous vouliez afficher un sprite sur une partie de l'écran ou sur l'autre. J'espère que c'est bien clair...

Je sais que ce cours est un peu abstrait mais, si vous connaissez déjà un petit peu l'assembleur si vous maitrisez la gestion des sprites avec les vecteurs système vous devriez être en mesure d'afficher de superbes pages overscan... je vous assure que ce cours contient tous les élements nécessaires... certes, ils sont peut-être un petit peu éparpillés mais, boh, vous commencez à avoir l'habitude de me lire...

Bon, eh bien la prochaine fois vous aurez droit à un vrai listing assembleur en exemple (un pur et dur !) et nous verrons en détail la gestion "professionnelle" d'un écran overscan avec l'utilisation du mode vectorisé et la reconfiguration de la pile. Ensuite (pas avant le numéro 6) nous verrons comment utiliser les banks supplémentaires des CPC6128/CPC6128+ (ou DK'Tronics) lorsque l'on est en overscan car on a dès lors trois fois plus de RAM disponible ce qui n'est pas négligeable, on pourra ainsi conserver le système ce qui permettra par exemple de retourner au basic pour utiliser l'Amsdos après avoir lancé un programme assembleur gérant un écran overscan et utilisant une petite centaine de Ko...

OffseT

FAITES LA CHASSE AUX SECTEURS ENDOMMAGES !

NE LES LAISSEZ PAS EMPOISONNER VOS PAUVRES PETITS FICHIERS !

# BIDOUILLES


Eh oui, ça devait arriver, la rubrique Bidouilles est de retour dans Quasar CPC... Mais cette fois-ci, c'est la bonne ! Elle y est et elle y restera !

Le menu du jour : comment se débarasser des secteurs endommagés des disquettes payées à bas prix. En effet, qui ne s'est jamais énervé contre une disquette sur laquelle on avait sauvé un fichier qui se soit fait aggressé par un secteur défectueux... Qui n'a jamais fait voltiger une disquette sur laquelle deux misérables petits secteurs déteriorés empêchaient toute copie fiable, ou encore... la liste des méfaits de ces secteurs est longue... très longue...

## MAIS POURQUOI CES PETITS SECTEURS NOUS EMBETENT-ILS ?

Eh bien, c'est tout simple, c'est parce que l'Amsdos ne s'en occupe pas ! Les utilisateurs de PC auront sans doute noté le fait que, lors du formatage des discs, le MS/DOS repère les secteurs défaillant et les précise sur le disc. Ainsi, lorsque vous sauvegardez un fichier le système consulte d'abord la liste des secteurs interdits puis effectue la sauvegarde en les évitant.

Mais, sur CPC, avec l'Amsdos et le CP/M, les ingénieurs d'Amstrad on préféré faire l'économie de telles options... Résultat : les secteurs endommagés ne sont pas (et ne peuvent pas !) être différenciés des autres et ils en profitent biensûr pour vous aggresser.

## COMMENT LES NEUTRALISER...

Comme vous vous en doutez, si je suis là aujourd'hui, c'est parce qu'il y a une manière simple de les neutraliser... Car il faut les neutraliser ; vous vous doutez bien qu'on ne peut pas les éliminer, comme ça, d'un coup de baguette magique (n'est-ce pas B.L. ?).

Mais avant de pouvoir tirer sur une cible, il faut la localiser... Biensûr, il est possible de faire un mapping total sous disco mais je ne pense pas que ce soit la méthode la plus rapide ; en effet, même si Discology vous indique précisément la position des erreurs sur un disc, il ne vous attend pas et fonce direct à la piste suivante ! Vous allez me dire que l'on peut mettre la pause oui mais que de temps perdu !

Non, je pense que la meilleure méthode consiste à reformater les discs incriminés sous disco (c'est le plus rapide) et à les vérifier sous CP/M Plus (eh bien oui, il sert à quelque chose !).

Tout d'abord, le formatage. Il va nous falloir différencier deux types de formatage suivant les symptômes de vos discs. Ceux pour lesquels les erreurs ne se manifestent que lors du chargement de fichiers et ceux où le CAT lui même vous renvoie un Disc Read Fail. Pour les premiers, pas de problème, formatez-les en DATA pistes 0 à 41. Pour les autres, c'est plus compliqué car c'est le directory qui est atteint... Or, on en a besoin pour neutraliser nos fichiers... Heureusement VENDOR est là, en effet, en format Vendor, on a certes plus que 169Ko de disponibles sur le disc mais le directory est transféré en piste 2 ce qui va nous permettre d'éviter le ou les secteurs défectueux de la piste 0 puisque les piste 0 et 1 sont alors réservées pour des accès CP/M (avec la RSX CPM) dont nous ne nous servirons pas...

Voilà, vos discs sont formatés, maintenant, on va les passer au Scanner avec Disckit3 (ou Disckit sur CPC+). Chargez donc votre CP/M Plus (si, si, cherchez bien, vous devez l'avoir quelque part au fond d'un vieux tiroir fermé à clef depuis des mois.) et au prompt A) tapez Disckit3 (ou Disckit sur CPC+). Alors, ça ne vous rapelle pas de vieux souvenir que de retrouver ce menu élégant et ergonomique... Choisissez Verify, et vérifiez vos discs un à un (forcément !). A chaque fois qu'une erreur sera détectée, le CP/M vous affichera un message du genre : Disc Error : Track 23, Sector #4. Puis, quelques ligne plus bas : Ignore, Retry or Cancel. Dès lors, il ne vous reste plus qu'à relever le secteur et la piste où se trouve l'erreur et à taper sur I pour que la vérification se poursuive. Voilà, simple non ?

Et paf ! J'en était sur, le petit blond à lunettes est revenu pour me critiquer paskeu le CP/M ne vérifie que les pistes allant de 0 à 39 et que les secteurs défectueux des piste 40 et 41 passent donc au travers de notre vérif... Certes, j'avoue, je suis coupable, mais il est en général inutile de s'occuper de ces pistes car l'Amsdos les ignore royalement. Le débat est donc clos. Juste quelques petites précisions, si vous utilisez le logiciel Copyluck pour faire vos copies fichier sur un format 187Ko il vous faudra là vérifier les pistes 40 et 41 car celui s'en sert (utiliser alors un mapping sous Discology). Et puis avant de prendre une autre page je tiens à vous préciser que dans le cas de discs au format Vendor il est inutile de relever les erreurs des pistes

0 et 1 puisque celles-ci ne seront pas utilisées.

ET APRES ?

Maintenant que vous avez relevé toutes ces erreurs vous vous demandez certainement ce qu'on va en faire ? Rappellons tout d'abord ce pourquoi on est là : on veut empêcher l'Amsdos d'écrire sur les secteurs défectueux. Et quelle est la condition pour que l'Amsdos évite une piste ? Il n'y en a qu'une, il faut qu'un fichier en occupe déjà la place ! Vous en déduisez donc logiquement qu'il va nous falloir créer des fichiers factices qui vont neutraliser ces vilains secteurs. En fait, ça n'est pas très compliqué à faire à condition de bien connaitre la structure du directory... ce que nous allons donc voir maintenant !

NEUTRALISONS LES VILAINS SECTEURS

Je vous averti qu'il va vous falloir être très rigoureux.. Chargez donc Disco, choisissez l'éditeur et éditez la piste 0 de vos discs (ou 2 en Vendor). Dès lors, voici ce qu'il vous faut mettre à chaque adresse :

0000 : Numéro de User (mettez n'importe quoi sauf &E5)

0001 à 0008 : Nom de fichier (je vous propose " B U G !")

0009 à 000B : Extension du fichier (BUG par exemple...)

000C à 000E : On y met des zéros !

000F : Nombre d'enregistrements (nombre de blocks*8)

0010 à 001F : Liste des blocks du fichier sachant que :
Block=INT([(Piste*9)+Secteur]/2)

UN EXEMPLE S'IL VOUS PLAIT !

Du calme, ça arrive. Prenons un cas complexe. Imaginons une disquette où on a relevé ces défaillances :

- Piste 2 : secteur #1 et #2
- Piste 5 : secteur #0
- Piste 9 : secteur #5

Déjà, essayons de voir le nombre de blocks dont nous auront besoin... Eh non, pas quatre ! Trois suffisent ! Car en fait, un block ne défini pas un secteur mais deux secteurs consécutifs. Ici, les secteurs #1 et #2 de de la piste 2 pourrons donc être déclarés dans le même block. Mais attention, ceci ne marche que si les deux secteurs se suivent car, à partir du numéro de block la formule dont se sert l'Amsdos pour repérer ses fichiers est la suivante :

Piste=INT(2*BLOCK)/9
Secti=[(2*Block MOD 9)+1] OR &C0
et Sect2=Secti+1.

Votre esprit subtil n'a-t-il rien remarqué ? Il y a 9 secteurs par piste ; imaginons que la formule de calcul du secteur i nous rende &C9... Sect2 vaudrait alors &CA ce qui n'est pas possible puisqu'un tel secteur n'existe pas en format standard... L'Amsdos fait alors automatiquement la conversion et Sect2 se trouve remanieé pour devenir le premier secteur de la piste suivante. Tout ça pour vous faire remarquer que deux secteurs consécutifs ne sont pas forcément sur la même piste.

Bref, revenons-en à notre exemple et calculons nos 3 blocks :

- Block1=INT([2*9+1]/2)=9=&9
- Block2=INT([5*9+0]/2)=22=&16
- Block3=INT([9*9+5]/2)=43=&2B

Maintenant, il nous faut calculer le nombre d'enregistrements :

- Enreg.=NbrBlocks*8=3*8=24=&18

Voilà, on choisi à présent le nom que l'on va donner à son fichier : " B U G !" pour le nom et "BUG" pour l'extension par exemple ! On choisira de préférence un nom de fichier avec des espaces comme ça on sera sûr que personne ne pourra le charger.
Pour finir on va se choisir un user quelconque. Vous éviterez tout de même la valeur &E5 (229) qui est réservée aux fichiers effacés. Je ne vais pas m'étendre sur ce sujet aujourd'hui puisque nous verrons ça en détail la prochaine lorsque j'expliquerai comment on récupère des fichiers effacés. Prenons 00 comme user.

Voici ce que vous devriez finalement obtenir après introduction des données sous l'éditeur de Disco :

```
&0000 : 00 20 42 20 55 20 47 20 21 42 55 47 00 00 00 18
(Ascii   .  B  U  G  !  B  U  G  .  .  .  .)
&0010 : 09 16 2B 00 00 00 00 00 00 00 00 00 00 00 00 00
(Ascii     +                             )
&0020 : E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5 E5
(Ascii                                   )
&0030 : Etc..
```

Attention, ne tenez pas compte des codes ascii que je vous ai ajouté entre parenthèses ; en standard, ils apparaissent dans la colonne de droite mais elle ne contenait pas sur cette page.

ATTENTION ! N'OUBLIEZ PAS DE SAUVER VOS MODIFICATIONS !

Ensuite, par mesure de sécurité, je vous conseille de charger l'exploreur et de prendre Exploration Fichier afin de vérifier si votre fichier factice est bien là où vous le désiriez.

Bon, eh bien j'espère avoir été clair. J'attends vos lettres pour connaitre votre avis sur cette nouvelle formule des bidouilles. Comme je vous l'ai déjà dit, la prochaine fois nous reverrons vite fait comment on peut récupérer simplement les fichiers effacés (j'en avais déjà parlé dans le numéro 2 mais celui-ci ne tournait pas sur tous les CPCs). Mais nous étudierons surtout la structure détaillée d'un fichier Amsdos (Basic, Binaire ou Ascii).

Tiens, pour une fois il me reste de la place en bas de la page... Comme je n'ai pas grand chose à dire de plus... Heu... Eh bien...

Ciao tout le monde !

OffseT (qui remercie Sined)


# LE CONCOURS...


Il semblerait, enfin c'est l'impression que j'ai, que personne (mais alors absolument personne !) ne se soit intéressé à notre concours... Tant pis, comme je suis beau joueur je vous donne quand même les réponses.

Mais avant tout, je précise que je parle ici du concours du numéro 1 que Quasar... Celui du numéro 3, c'était une plaisanterie ; j'adresse d'ailleurs toute ma sympathie à Cossart J.M. qui est amicalement rentré dans notre jeu... De plus, je suis tout à fait d'accord avec lui... Je me demande sérieusement si certains CPCistes ne trouveraient pas le moyen d'y répondre de travers... Mais non Tony, je ne pensais pas à toi, à ton sujet, je ne me demande plus rien... j'en suis sûr... Aïe, non, ne me frappe pas ! Apprends à rester calme...

Bref, passons et revenons à notre concours. A défaut d'y avoir répondu vous apprendrez certainement des choses en en lisant les réponses ; alors, c'est parti !

**1/** Quel est format des disquettes des Amstrad CPC ?

- J'avoue, cette question est un petit peu délicate car il fallait répondre qu'il s'agit des disquettes 3 pouces ce qui est extrêmement vicieux... si, si !

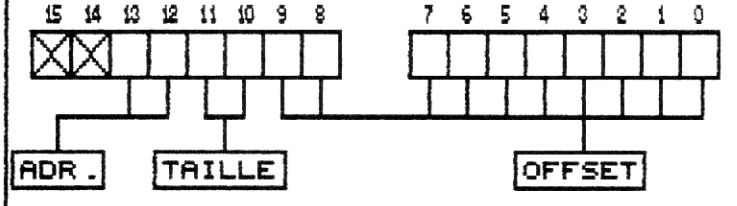
**2/** Citez quatre commandes Basic qui ne fonctionnment pas sur CPC464.

- Les voici, les voilà, il y a :
  .CLEAR INPUT (vidage du buffer)
  .FILL x (remplissage d'une zone)
  .ON BREAK CONT (No comment.)
  .GRAPHICS PEN (et PAPER) (R.A.S.)

**3/** A quoi servent les registres 12 et 13 du CRTC ?

- Le registre 13 contient les 8 premiers bits de l'offset. Et le 12 est un peu plus complexe puisque ses deux premiers bits sont les bits 8 et 9 de l'offset alors que les quatre suivants définissent l'adresse de l'écran et sa taille (les bit 6 et 7 ne sont pas cablés). Pour plus de claté voici un schéma :

Registres 12 et 13 du CRTC6845 :




Voici maintenant le détail des bits 10, 11, 12 et 13 :

| 13 | 12 | Adresse |
|---|---|---|
| 0 | 0 | &0000 |
| 0 | 1 | &4000 |
| 1 | 0 | &8000 |
| 1 | 1 | &C000 |

| 11 | 10 | Taille |
|---|---|---|
| 0 | 0 | 16Ko |
| 0 | 1 | 16Ko |
| 1 | 0 | 16Ko |
| 1 | 1 | 32Ko |

De plus, sur les CPC plus, le bit 10, sert également de huitième bit à la sortie imprimante.

Si vous n'avez pas très bien compris, allez jeter un coup d'oeil du côté de la rubrique de perfectionnement à l'assembleur du dernier numéro.


Alors, il vous plait pas mon concours ?!?

**4/** Comment commande-t-on au Gate Array de changer la couleur du border en cours de balayage ?

- Là, je vous dis : cf. cours assembleur du premier numéro.

**5/** A quoi sert le FDC ?

- Le FDC sert à gérer l'accès au(x) lecteur(s) de disquettes de votre brave petit CPC. En fait, le FDC est au lecteur de discs ce que le CRTC est à l'écran.

**6/** Quel mode émule de CRTC sur CPC ?

- Comme je l'ai écrit plus haut, le CPC est équipé d'un CRTC6845 qui émule donc logiquement le mode 6845...
On notera par ailleurs que certains PC (VGA !) utilisent eux-aussi cette petite pupuce.

**7/** Que signifient les appelations CRTC, FDC et PSG ?

- FDC ça veut dire Floppy Disc Controller.
PSG signifient Program Sound Générator.
CRTC... je me souviens plus... Ah, si, ça veut dire Cathode Ray Tube Controller. Soit en français, controleur du canon à électron... eh oui, sans lui l'écran de votre petit CPC serait tout noir...
En outre, la plupart des problèmes d'incompatibilité entre les différents CPCs viennent de lui car les ingénieurs d'Amstrad ont cru bien faire d'utiliser différentes versions de ce composant...

**8/** Question subsidiaire : c'est quoi une rupture d'écran ?

- Il s'agit d'une technique programmation qui consiste à changer l'adresse de l'écran et l'offset pendant le balayage de l'écran. Ainsi, on peut obtenir différents effets, comme la distorsion de scrollings ou l'effet de miroir... Mais on peut encore faire plein de choses avec les ruptures ; je dirais même plus, sans les ruptures 99% des démos que vous voyez tourner sur CPC n'existeraient pas. Enfin, nous aurons l'occasion dans reparler dans un cours de perfectionnement à l'assembleur.
Je vous ferais tout de même remarquer que, si sur les CPCs classiques l'utilisation de la rupture n'était pas très aisée (c'est le moins qu'on puisse dire.) la gestion est grandement facilitée avec l'Asic qui génère tout seul cette fonction (classe non ?).

**B**on, eh bien on l'a enfin fini ce concours... Ça a été dur quand même ; je dois vous avouer que j'ai hésité fichtrement longtemps avant de me décider à vous donner les réponses alors que PERSONNE n'y a répondu...
**E**nfin, maintenant, c'est fini, vous ne verrez plus jamais de concours dans les pages de Quasar CPC car j'ai comme l'impression que de telles rubriques ne font pas l'unanimité...

**D**onc, la disparition de la rubrique Concours entrainera l'apparition d'une nouvelle rubrique... C'est ainsi que dès le numéro 5 vous pourrez découvrir une rubrique consacrée à l'électronique. Mais nous allons tâcher de faire quelque chose d'un peu original... En effet, vous n'y découvrirez pas de montages qui visent à faire clignoter des lampes ou à faire bipper un haut-parleur mais vous y verrez plutôt des montages qui peuvent avoir une utilité directe avec le CPC... Patience...

**M**ais ce n'est pas sans beaucoup d'émotion que je clos cette rubrique... Snif ! Snif ! Passons...

**J**e vous dit donc au revoir car c'est Zik qui s'occupera de la rubrique d'électronique.

**C**iao,

OffseT qui va aller dormir...

Je m'excuse pour cette intrusion mais comme il y avait un grand blanc en bas de la page on m'a demandé de venir boucher le trou...

# POKES par TONY

Salut tous, salut toutes . Tony au clavier , afin d' énumérer pas mal de trucs pour éviter de se retrouver comme ci-dessous ... Je ne suis pas un rechercheur de pokes ou quoi que ce soit , mais j'ai eu un jour l'envie , et vous devinez pourquoi , de trouver bien des astuces pour terminer les jeux les plus cruels...

C'est la raison pour laquelle les nombreux pokes que vous aurez la joie de découvrir ou de redécouvrir avec moi sont tirés principalement de : AMSTRAD 100 % , (snif, on est triste...) la bible des pokes volumes I et II, et aussi de fanzines que je remercie au passage (on est là pour s' entraider !!! )... Allez, joyeux Noël à tous...

Bon, bah on va commencer par ce qu'on appelle les "totales" bien qu'il n'y ait pas tout ("faut pas pousser même dans les orties...").

## A.M.C (astro marine corps):

CODE : Niveau 2⇒DAGOBAH

CHAINES : Vies Infinies :

Rechercher la chaine hexadécimale 96,35,20,27,CD,5D et remplacer 35 par 00

Bombes Infinies :

Rechercher la chaine hexadécimale 3A,4B,42, 32,9E,96 et remplacer 32,9E,96 par 3E, FF,00

POKES : Vies Infinies :

Poke &876C,&00

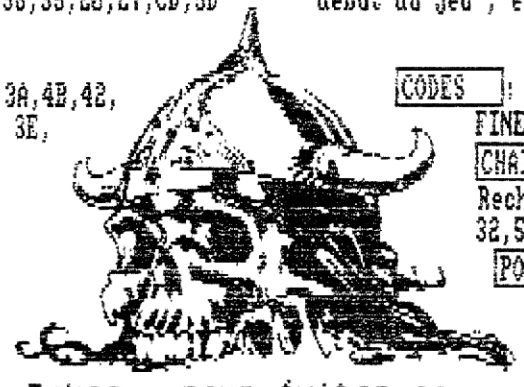
Je sais, il pourrait en avoir le temps ...

Bombes Infinies :

Poke &4EE3,&3E
Poke &4EE4,&FF
Poke &4EE5,&00

CHEAT MODE : Invulnérabilité :

Entrer dans le tableau des scores le mot : CREEP ...Une crêpe à quiconque me trouve le temps Inf.



Pokes : pour éviter ça...

## LIVINGSTONE II :

CODE : Niveau 2 15215

CHEAT MODE : Pour obtenir des vies infinies , dès le début du jeu , entrer D,F,B,N (sans les virgules).

## PIPEMANIA :

CODES : Dans l'ordre de difficulté : FINE,NEWS,FAILS,SAIL,ERIC,TAPE,SLOW,ACHE.

CHAINE : Temps Infini :

Rechercher la chaine hexadécimale 25,3D, 32,55,25,18 et remplacer le 3D par 00.

POKES : Temps Infini :

Poke &125F,&00

Reprise du jeu au dernier tableau visité avant le GAME OVER fatal :

Poker les valeurs 00 aux adresses &2227,&2228,&2229.

Maintenant on y va en vrac ... Et ne vous inquiétez pas , il m'en reste encore au moins 5 fois ça ...

Pour bien se comprendre voici mon langage : -V.Inf.: Vies Infinies
-Eneg.Inf:Energie Infinie
-Ips.inf.:Temps Infini
-33,CD,00()00 par 3D : rechercher la chaine hexadécimale 33,CD,00 et remplacer 00 par 3D.
-p3,s02,a03CE()35 par B6 : aller en piste 3, secteur 2, adresse &03CE et remplacer 35 par B6.

Pour les autres abréviations que j'oublie, y'aura qu'à deviner !!!

1942:Réduction des avions 44,4B,0A,6F,83,0A()4D par 00 .

Beyond The Ice Palace:

Eneg.Inf:p10,s46,a0059 () 32 par 30

Bomb Jack I:V.Inf.

3E,03,32,59,2F()03 par FF

Bomb Jack II:V.Inf.

Poke &917A,&A7

Cauldron I:comp.La Collec.

V.Inf.3D,32,84,BA()3D par 00

Cauldron II:

V.Inf.Poke &1C90,&00 ou C2,1F,84,3A,1C,BE,3D()3D par 00

Cauldron III(S.Cauldron):

Eneg.Inf.CD,2D,78()B7,00, 00 . Ce ne sont pas les vies infinies !!! Je lance d'ailleurs un petit défi-concours : le premier qui trouve les Vies Inf. dans ce Méga-jeu programmé par ELM-SOFT. Allez, au boulot...

Druid :Au revoir, monstres

Entrer dans le tableau des scores le nom :

THE COVENANT

Mange-Cailloux :

Qui ne connait pas ce merveilleux jeu d'OCEAN ? Voici le listing paru voila un bon bout de temps dans ACPC :

```
10 'M.-C. 100 % K7   POUR K7
20 'par J-M TALENTON(merci)
30 OPENOUT "x":MEMORY &DBFF
-1:CLOSEOUT
40 LOAD "MANGE.BIX",&6000
50 LOAD "MANGE.BIY",&4000
60 OPENOUT "x":MEMORY &2BFF-1
:CLOSEOUT
70 LOAD "MANGE.BIZ"
80 POKE &7381,255
90 CALL &5000
```

Mazemania : V.Inf.

Poke &2576,&00

Moving Target : V.Inf.

Poke &6B36,&00

Puzznic: Essais Inf.

Poke &3F5F,&00
Poke &593D,&00

Predatron Mega-Demo :

Exclusif !!!Pour la première fois vous allez avoir sous les yeux les réponses aux questions posées dans une des parties (!?) :

B,C,A,C,A,B,C,B,C,A,B,A.

(... par Tony ...)

ADOLFENFACTICE

Rick Dangerous :

Il a vraiment insisté pour se crever la bouille afin de se trouver des pokes qui ont déjà été trouvés par pas mal de personnes qui vivaient dans leur époque ( hé hé ...) non pas comme ce il : OFFSET

Enfin mettons les quand même :

V.Inf.: Poke &88C3,&00
Tir Inf.: Poke &9F35,&00
Bombes Inf.: Poke &9F90,&00

Bravo tout de même à OFFSET.

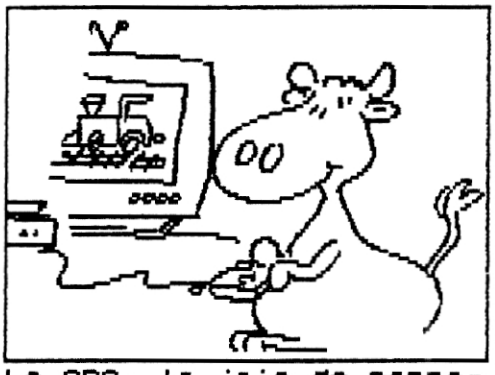
Wonderboy : V.Inf.:

21,31,81,35,3A()35 par 00

Et voilà c'en est terminé de la rubrique pokes pour ce numéro de Noël .

page entièrement réalisée par TONY

# P . A . PAR TONY



Le CPC: la joie de reproduire de bonnes conditions chez soi.

Salut à tous et à toutes, Je suis heureux de vous rédiger cette rubrique (pour la première fois d'ailleurs...) et je compte sur vous pour vous décarcasser pour nous écrire et passer des petites annonces. On est là pour ça et il est bien normal que nous vous ussions aidés !
D'autre part je tiens à signaler que cette page est écrite pour la deuxième fois et il n'est jamais marrant de se retaper une page. Mais devinez QUI a bousillé le premier fichier sur lequel j'avais passé de longues heures ? OFF-cette maniaque ! Du coup je suis obligé de tout recommencer quelques minutes avant le bouclage ! (NDZik : C'est pas sa faute !)
En tout cas je vous autorise à passer n'importe quelle sorte d'annonce sauf pour la vente de votre CPC...
Cela dit, voici les petites annonces du trimestre. Bonne pêche !

-Cherche Scanner Vidi.
S'adresser à la rédac.

-Cherche Imprimante pour CPC.
S'adresser à Tony(ou à la rédac).

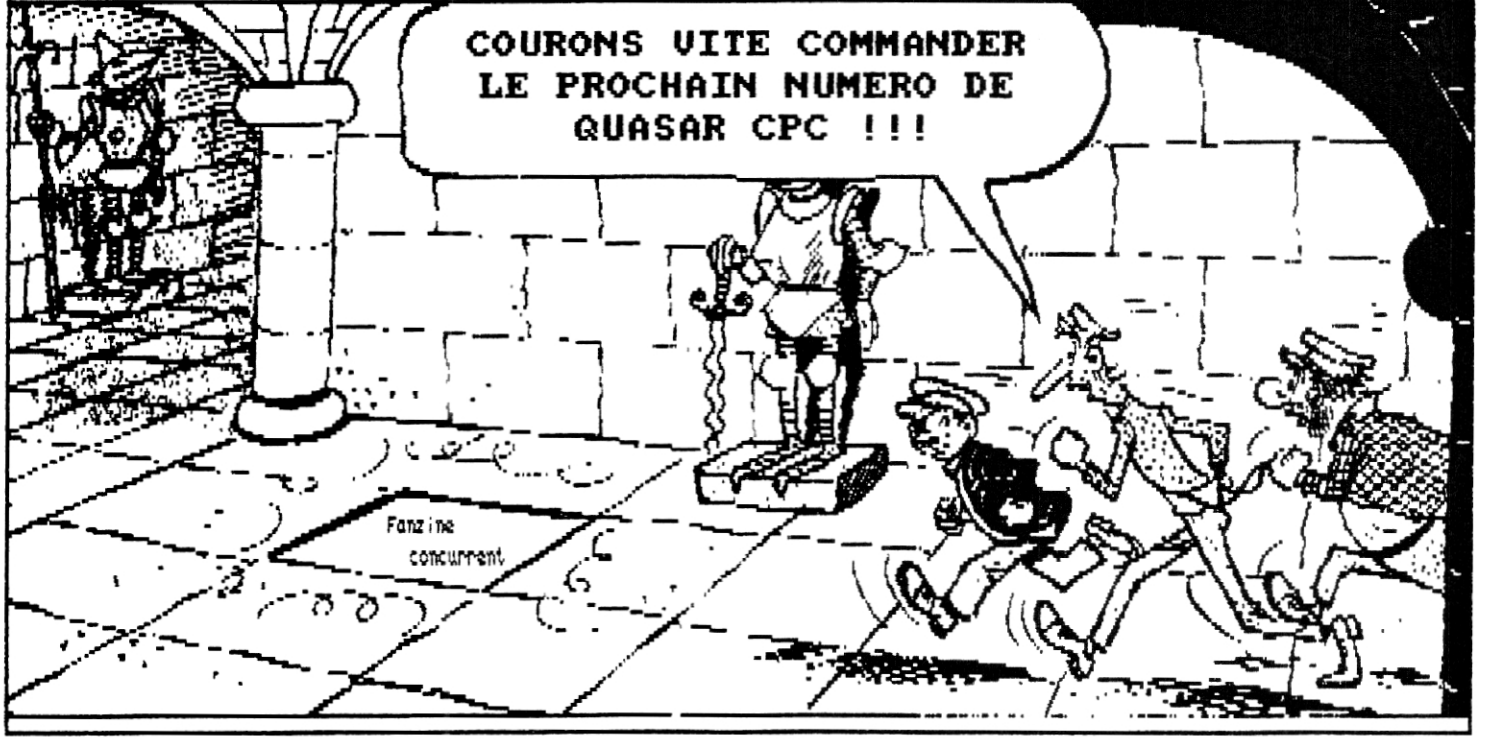
-Cherche le jeu DIG-DUG.
S'adresser à Tony(ou à la rédac).

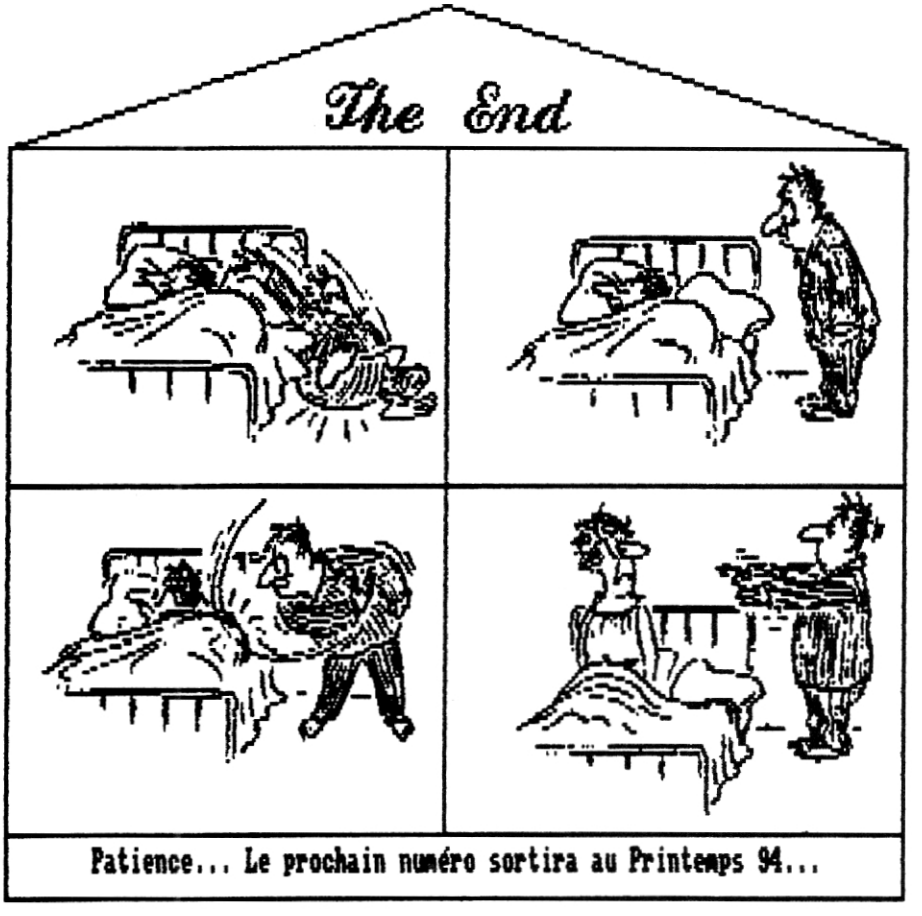
-CHERCHE : Graphiste en décors et visages en gros plan. C'est pour la "surprise 94"! Débrouillez-vous il nous faut un graphiste spécialisé dans les décors, mais surtout dans les visages en gros plan...

- F A C T I C E vous propose une longue série de démos, mais aussi des freewares, tout ça gratuitement et évidemment rapidement à condition de ne pas oublier:
A noter : -la DK
Cybersoft faisait -l'enveloppe auto-adressée
DEAD PLAYER -le timbre !
Ce principe d'échanges est super, nous devons nous retourner vers cette méthode d'échanges de l'avenir.
(NDMOI:Au fait Damien, tu me la renvois quand, ma DK ?)

Voici une grande petite annonce: le l'ai recopiée mot à mot majuscule à minuscule:
SUPER-MOMO CHERCHE TUNER TV BON ETAT POUR 6128 plus:
C'est pour regarder la TELE TRANQUILLEMENT dans ma CHAMBRE !
TRANQUILLEMENT !!! AVEC MES 4 FRERES qui gigotent de PARTOUT !!!!!
SI VOUS AVEZ PAS DE TUNER alors donnez !!!, non, vendez moi un CABLE PERITEL pour que je puisse jouer AVEC LES CONSOLES DE MES POTES !! (SUPER N.E.S., MES, MEGADRIVE)
(NDTONY:N'est-ce-pas Offset(avec un O comme zéro...))
Ne vous trompez-pas, 6128 + et non pour 6128, sauf si c'est le MEME CABLE.
Vous pouvez M'ENVOYER UNE LETTRE EN ME DISANT LE PRIX OU ME TELEPHONER OU ECRIRE à la REDAC.
Mon adresse: Hôpital de JEAN-IBANèS
B.P.111
09200 ST-GIRONS
Num.de tél: 61-96-20-09 (entre 18H et 21H)
Demander MADJAR MOHAMED.

Ne tenez surtout pas compte de ma dernière note !
FACTICE Damien Delurier, 82 chemin du Charbonnier 69200 Venissieux






Patience... Le prochain numéro sortira au Printemps 94...