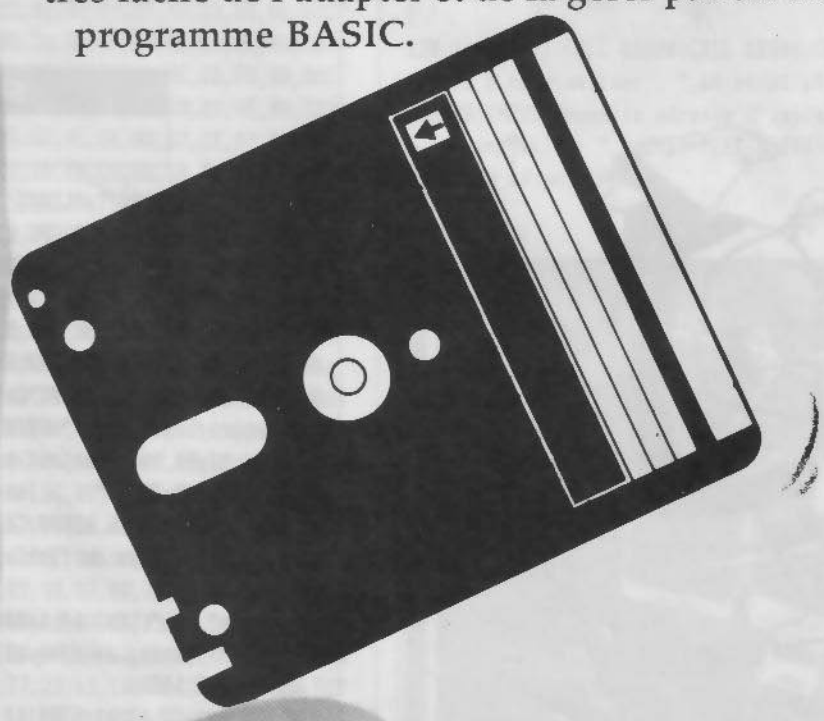
CPC utilitaire

MODIFICATION DU PROGRAMME CATEDIT

Serge BREUZIN

Valable pour CPC 464 – 664 – 6128

J'ai tout d'abord éprouvé le besoin d'apporter une petite amélioration au très utile programme CATEDIT de F. TACHET, S. ST-MARTIN et C. VEYRE (AMSTAR & CPC N° 46). Ceci fait, j'ai pensé qu'il serait intéressant de vous montrer comment j'ai procédé. Comment j'ai détourné le programme pour lui faire exécuter une routine de mon cru. Et comment cette routine fonctionne. Il vous sera très facile de l'adapter et de la gérer par un simple programme BASIC.



Les lecteurs peu intéressés par les explications se contenteront de saisir et lancer **LOADER1** ou **LOADER2**. Ces loaders ajoutent un peu plus de 110 octets juste sous le programme, afin de différencier les disquettes formatées VENDOR de celles contenant effectivement le SYSTEME proprement dit. **LOADER1** conserve le programme dans son intégralité, tandis que **LOADER2** vous soulage de 11 K en supprimant la (luxueuse) présentation.
Le source est celui de la routine contenue dans **LOADER2**. **LOADER1** est identique, seules les adresses sont changées.
Ceux qui n'auraient pas encore saisi cet excellent programme du fait de sa longueur peuvent ne taper que la partie qui gère le disque. Il suffit de saisir les programmes **CAT2** et **CAT3** qui, une fois lancés, seront réunis par :

```
10 MEMORY &6FFF
20 LOAD"catedit.bi2",&7000
30 LOAD"catedit.bi3",&8000
40 PRINT "APPUYEZ SUR UNE TOUCHE..."
50 CALL &BB06
60 SAVE"cated",B,&7000,&24A0,&7000
```

Il faudra alors utiliser LOADER2 en modifiant comme suit la ligne 10 :

```
10 MEMORY &6FFF:LOAD"cated",&7000
```

Détourner un programme n'est pas toujours facile. Il faut repérer un **JUMP (&C3)** ou un **CALL (&CD)** et changer la valeur des deux octets suivants, dans l'ordre : octet de poids faible → octet de poids fort. Par exemple si le programme donne **CD,DE,A9** (CALL (&A9DE) et que votre routine se trouve en &6F90, vous devrez remplacer les octetsDE, A9 par 90, 6F. Votre routine devra alors obligatoirement se terminer par un **JUMP &A9DE** (C3, DE, A9). Pourquoi un JUMP ? Tout simplement parce que la machine, en effectuant un CALL (qui fonctionne comme un GOSUB en BASIC), va mettre dans une mémoire particulière (la pile) l'adresse de l'instruction suivante, à laquelle le programme reviendra en rencontrant un RET (RETURN). Si vous programmiez un CALL, le programme reviendrait à la fin de votre propre routine. A ce propos, laissez la pile dans l'état où vous l'avez trouvée en entrant ! Le nombre de PUSHs (sauvegardes) doit être le même que le nombre de POPs (récupérations), si vous utilisez ces instructions. Sauf bidouilles d'équilibriste.
Il est souvent nécessaire de sauvegarder certains registres avant de faire exécuter le vôtre, car le programme que vous aurez détourné peut avoir besoin de ces valeurs pour continuer à fonctionner correctement. Il est possible d'opérer ainsi :

```
PUSH AF
PUSH BC
PUSH DE
PUSH HL
Votre programme
POP HL
POP DE
POP BC
POP AF
```

Vous remarquerez que les octets sont récupérés dans l'ordre : derniers entrés → premiers sortis.

La routine elle-même n'est pas sans intérêts, elle est commentée pas à pas dans le source. A vous de l'utiliser, quitte à la modifier pour vos propres besoins. Vous pouvez facilement créer un programme BASIC qui pourrait vous afficher le contenu de n'importe quel secteur de n'importe quelle piste, voire toute la disquette. Il est indispensable de connaître à l'avance le format de la disquette car il faut choisir le bon nom de secteur afin d'éviter un plantage. Heureusement l'adresse magique &A89F vous fournira ce renseignement. Il suffit de faire une lecture ou une écriture pour que le nom du premier secteur s'inscrive à &A89F. Pour ne rien déranger faites simplement OPENOUT"bidon":**CLOSEOUT**.

Une précaution nécessaire : il n'est pas possible de charger directement les registres du Z80 à partir du BASIC. Mais vous pouvez POKEr lesvaleurs utiles à des adresses choisies par vous et notées avec soin. Attention de faire ces POKEs au-dessus du HIMEM et en dehors des cases mémoires déjà utilisées. Par exemple, si vous voulez récupérer dans E une valeur POKEe en &A650 :

PUSH HL ; Sauvegarde éventuelle de HL, code E5.
LD HL, &A650 ; adresse de la valeur, code 21, 50, A6.
LD E, (HL) ; met dans E cette valeur, code 5E.
POP HL ; Récupération éventuelle de HL, code E1.

Pour afficher le contenu du secteur, une simple boucle FOR NEXT suffit. Si le secteur lu (de longueur 2) est en &6000 :

FOR tune=&6000 TO &61FF:print HEXS(PEEK(tune));" ";:NEXT

Pour une présentation plus "pro" :

```
10 MODE 2:tune=&6000
20 IF tune=&6200 THEN END
30 PRINT HEXS(tune);"  ";
40 FOR tiche=1 TO 16
50 x=PEEK(tune):IF x<&10 THEN PRINT"0";
60 PRINT HEXS(x);" ";:tune=tune+1:NEXT
70 tune=tune-16:PRINT"  ";:FOR tran=1 TO 16:y=PEEK(tu
ne):IF y<32 THEN PRINT".";ELSE PRINT CHRS(y);
80 tune=tune+1:NEXT:PRINT
90 ligne=ligne+1:IF ligne=25 THEN ligne
=0:CALL &BB06:CLS
100 GOTO 20
```

Pour une sortie sur imprimante (PRINT#8,---) il faudra diminuer la variable Y de 127 si elle dépasse ce nombre et ceci avant le test Y<32. J'ai précisé secteur de longueur 2, car c'est le cas le plus fréquemment rencontré mais il existe des secteurs de longueur 1 qui font 256 octets

(&100) ; long.2 de 512 (&200) ; long.3 de 1024 (&400) ; long.4 de 2048 (&800) ; long.5 de 4096 (&1000). Il existe aussi des secteurs de long.0, mais ils sont réservés au bouche à oreille. Pensez tout de même à avoir suffisamment de place en mémoire car le secteur est la plus petite unité lisible sur disquette.
Tous les autres détails sont commentés dans le source. Mais avant de terminer je voudrais attirer l'attention de ceux qui désireraient modifier un programme qui refuse obstinément de tourner sur le lecteur B. Deux manières sont le plus souvent employées pour lire un secteur, soit d'utiliser la routine &C666 comme je l'ai fait, soit d'utiliser le vecteur &BCD4 ce qui revient au même. Recherchez donc si le programme ne contient pas un CALL de ces adresses (CD, 66, C6 ou CD, D4, BC). En ce qui concerne cette dernière, HL contient le numéro de la commande augmenté de &80. Pour une lecture de secteur (commande N° 4) HL doit contenir &84. Dans tous les cas le registre E contient le numéro du lecteur. &00=A et &01=B. A vous de modifier cette valeur. Par exemple LD E,&00 (1E, 20, 00) par LD E,&01 (1E, 20, 01). Si vous avez la place nécessaire vous pouvez opérer un détournement comme expliqué plus haut et vous offrir le choix du lecteur. Pour une routine en langage machine la réponse sera dans A après un CALL &BB18, faites simplement LD E, A (5F) après avoir sauvegardé AF si nécessaire. Pour une modification directement dans le programme qui utilise moins d'octets que l'original, complétez avec des NOPs (&00). En BASIC chargez HL avec l'adresse à laquelle vous aurez POKé la réponse et faites LD E, (HL) (5E), entre PUSH HL (E5) et POP HL (E1). Le PUSH/ instructions:POP est probablement obligatoire car HL devra contenir l'adresse du tanpom.

Le plus souvent on se contente de faire un loader en BASIC qui effectue les modifications désirées avant le CALL de lancement du programme. Cela évite de toucher au programme sur la disquette, même si l'exécution s'en trouve ralentie.

Notez bien aussi l'adresse &A700. Car celle-ci contient le numéro du lecteur courant. Il vous est possible de le consulter (comme &A89F) par un PEEK, mais aussi de changer de lecteur en POKant à cette adresse la valeur désirée. POKE &A700, 1 vous fera passer en B que vous ayez ou non un deuxième lecteur, qu'il soit allumé ou non et qu'il s'y trouve ou non un disque ! (&A702 contient la même valeur mais après une lecture/écriture, cette différence peut-être utile).

Encore deux adresses de la ROM disque : &CDDA pour ùA et &CDDD pour ùB. Un exemple pour passer en B :

```
LD C,&07     (0E,07)    ;N° ROM disque.
CALL &BB0F   (CD,0F,BB);Sélection ROM disque.
CALL &CDDD   (CD,DD,CD);Nous passons en B.
CALL &B903   (CD,03,B9);Resélectionne la RAM.
RET          (C9)       ;Pour revenir.
```

Vous pouvez également rechercher, toujours dans la ROM disque, &C2F2 et &C4F0. Deux routines particulièrement fumeuses. Cette fois-ci le numéro du lecteur semble devoir être en A en entrée, pour passer en C puis en E !

En recherchant ces adresses dans les programmes récalcitrants vous devriez, dans la plupart des cas, arriver à vos fins. Sinon vous pouvez rechercher dans vos archives la bidouille superbe d'Heddy MENTALECHETA (CPC N° 19 de février 1987) qui arrive, avec un simple inverseur, à tromper l'ordinateur et les programmes en passant allègrement de A en B et de B en A. Malheureusement cela n'est valable que pour les disquettes en formats AMSDOS. Savez-vous qu'un lecteur 3,5 pouces/ double tête contient plus de 800 k, avec un directory de 256 (!) programmes ? Hélas les gestionnaires de ces disques refusent de les reconnaître en A. Pour un lecteur extérieur 3 pouces la bidouille ci-dessus est parfaitement valable.

LOADER 1

```
1Ø MEMORY &4995:LOAD"catedit",&4996
2Ø FOR t=&492Ø TO &4995
3Ø READ z$:z=VAL("&"+z$):POKE t,z:NEXT
4Ø POKE &7ØBE,&2Ø:POKE &7ØBF,&49
5Ø CLS:PRINT"APPUYEZ SUR UNE TOUCHE...":CALL &BBØ6
6Ø SAVE"catedit2",b,&492Ø,&51B5,&95ØØ
7Ø DATA 21,53,8E,5E,16,ØØ,3E,41,21,ØØ
8Ø DATA 6Ø,F5,ØE,Ø7,CD,ØF,B9,F1,C5,4F
9Ø DATA CD,66,C6,C1,CD,18,B9,7E,FE,11
1ØØ DATA C2,7Ø,49,23,7E,FE,ØØ,C2,7Ø,49
11Ø DATA 23,7E,FE,Ø3,C2,7Ø,49,21,B2,8D
12Ø DATA 36,28,23,36,53,23,36,29,23,36
13Ø DATA 79,23,36,73,23,36,74,23,36,65
14Ø DATA 23,36,6D,C3,DE,A9,ØØ,ØØ,ØØ,ØØ
15Ø DATA 21,B2,8D,36,28,23,36,56,23,36
16Ø DATA 29,23,36,65,23,36,6E,23,36,64
17Ø DATA 23,36,6F,23,36,72,C3,DE,A9,ØØ
18Ø DATA ØØ,ØØ,ØØ,ØØ,ØØ,ØØ,ØØ,ØØ,ØØ,ØØ
```

LOADER 2

```
1Ø MEMORY &4995:LOAD"catedit",&4996
2Ø FOR t=&6F9Ø TO &6FFF
3Ø READ z$:z=VAL("&"+z$):POKE t,z:NEXT
4Ø POKE &7ØBE,&9Ø:POKE &7ØBF,&6F
5Ø CLS:PRINT"APPUYEZ SUR UNE TOUCHE...":CALL &BBØ6
6Ø SAVE"cat2",b,&6F9Ø,&25ØF,&7ØØØ
7Ø DATA 21,53,8E,5E,16,ØØ,3E,41,21,ØØ
8Ø DATA 6Ø,F5,ØE,Ø7,CD,ØF,B9,F1,C5,4F
9Ø DATA CD,66,C6,C1,CD,18,B9,7E,FE,11
1ØØ DATA C2,EØ,6F,23,7E,FE,ØØ,C2,EØ,6F
11Ø DATA 23,7E,FE,Ø3,C2,EØ,6F,21,B2,8D
12Ø DATA 36,28,23,36,53,23,36,29,23,36
13Ø DATA 79,23,36,73,23,36,74,23,36,65
14Ø DATA 23,36,6D,C3,DE,A9,ØØ,ØØ,ØØ,ØØ
15Ø DATA 21,B2,8D,36,28,23,36,56,23,36
16Ø DATA 29,23,36,65,23,36,6E,23,36,64
17Ø DATA 23,36,6F,23,36,72,C3,DE,A9,ØØ
18Ø DATA ØØ,ØØ
```

LISTING SOURCE

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

;
        LD   HL,#8E53
;Charge  en HL l'adresse à laquelle se trouve le numéro du drive (0=A,1=B).
        LD   E,(HL)
;Ce numéro est placé dans E.
        LD   D,#00
;Charge  en D le numéro de piste (ici, #00) pour le call de lecture (#C666).
        LD   A,#41
;Charge  en A le nom du premier secteur (en format Système/Vendor).
        LD   HL,#6000
;Charge  en HL l'adresse à laquelle sera stokée le contenu du secteur lu
;(adresse du tampon).
        PUSH AF
;Sauvegarde de A car AF sera modifié par le prochain call.
        LD   C,#07
;place en C le numéro de la ROM chargée de la gestion du disque.
        CALL #B90F
;Connection de la ROM No7.
        POP  AF
;Récupération de A (Nom du premier secteur).
        PUSH BC
;Sauvegarde de C car C doit contenir le nom du premier secteur pour
;le prochain call
        LD   C,A
;Charge  en C le nom du premier secteur.
        CALL #C666
;Lecture du secteur C (#41) de la piste D (#00).
        POP  BC
;Récupération de C.
        CALL #B918
;Il ne suffit pas de lire un secteur encore faut-il le placer quelque part!
;Cette commande, (similaire à LDIR) transfère le contenu du secteur pointé par
;DE à l'adresse contenu dans HL. Toute cette procédure est nécessaire et peut
;etre utilisée pour charger n'importe quel secteur de n'importe quelle piste à
;l'endroit que vous aurez choisi. Attention au format de la disquette! Pour une
;disquette formatée DATA le nom du premier secteur est #C1. Sinon: Plantage!
;Il est utile de préciser que l'utilisation provisoire du tampon n'affecte
;en rien la capacité de stockage du programme pour des copies de fichiers
;car ce tampon est placé sous le himem et n'est donc pas protégé.
        LD   A,(HL)
;Charge en A la valeur du premier octet (HL contient toujours l'adresse du
;tampon, &6000.
;Va suivre maintenant la vérification des trois premiers octets qui devront
;etre #11,#00,#03. Si oui, le Système est chargé.
        CP   #11
;Comparaison du premier octet.
        JP   NZ,#6FE0
;Si la comparaison est fausse (le premier octet n'est pas égal à #11), on saute
;en #6FE0 afin d'écrire "(V)endor".
        INC  HL
;On augmente HL d'une unité pour lire le deuxième octet.
                LD   A,(HL)
;Chargement du deuxième octet.
                CP   #00
;Comparaison comme pour le premier octet.
                JP   NZ,#6FE0
;Saut en &6FE0 si le résultat est faux etc...
                INC  HL
        LD   A,(HL)
        CP   #03
        JP   NZ,#6FE0
        LD   HL,#8DB2
;Les trois octets sont exacts. Chargement en HL de l'adresse à laquelle se
;trouve le texte qui sera lu par le programme ("(V)endor" ou "(S)ystem").
        LD   (HL),#28
;Ecrit le caractère "(" (#28 est le code ASCII de "(").
        INC  HL
;Augmentation d'une unité afin d'écrire le deuxième caractère.
        LD   (HL),#53
;Ecriture de "S" etc...
        INC  HL
        LD   (HL),#29                ;)
        INC  HL
        LD   (HL),#79                ;y
        INC  HL
        LD   (HL),#73                ;s
        INC  HL
        LD   (HL),#74                ;t
        INC  HL
        LD   (HL),#65                ;e
        INC  HL
        LD   (HL),#6D                ;m
        JP   #A9DE
;Le mot "(S)ystem" ayant été placé, on retourne dans le programme principal.
;Attention pour ce genre de modifications de ne pas écrire des mots trop
;longs qui écraseraient le mot suivant. Si votre mot est plus court,
;complétez-le par des espaces (&20).
        NOP
        NOP
        NOP
        NOP
        LD   HL,#8DB2
;Nous voici à l'adresse &6FE0.
;Les trois premiers octets n'ayant pas été reconnus, nous allons procéder
;comme ci-dessus, mais nous allons écrire "(V)endor".
        LD   (HL),#28                ;(
        INC  HL
        LD   (HL),#56                ;V
        INC  HL
        LD   (HL),#29                ;)
        INC  HL
        LD   (HL),#65                ;e
        INC  HL
        LD   (HL),#6E                ;n
        INC  HL
        LD   (HL),#64                ;d
        INC  HL
        LD   (HL),#6F                ;o
        INC  HL
        LD   (HL),#72                ;r
        JP   #A9DE
;Le mot "(V)endor" est maintenant écrit, nous retournons dans le programme
;principal en sautant au CALL que nous avons détourné.
        NOP
        NOP
;Nous arrivons en &7000 qui est le point d'entrée du programme principal.
;

Text:26869      End:30285       3336 Bytes
Hmem:36153
```