JPC

DECEMBRE 1989

NUMERO 70

Le numéro 40 F

A PROPOS DU CLUB

HP-28

HP-41

HP-71

EDITORIAL

Nous voilà donc en 1990. Cette année commence pour nous par notre Assemblée Générale. Nous espérons que vous y participerez nombreux.
Comme vous pu le constater, nous avons changé d'imprimeur. Le numéro que vous evez entre les mains a été réalisé en photocopie. PPC ne peut plus supporter les coûts pratiqués auparavant.
Nous avons hésité entre deux solutions : éditer un numéro tous les deux mois, augmenter la cotisation ou changer le mode d'impression. Cette dernière solution nous semble la meilleure. Ce sera à l'Assemblée Générale de trancher.

Xavier Bille et Jacques Belin ont présenté leur démission du bureau. Nous ne pouvons que regretter leur décision et les raisons invoquées.
Vous trouverez un droit de réponse de Pierre David, Jean-Jacques Dhénin et Jannick Taillandier. Le travail qu'ils ont fourni pendant de nombreuses années a permis au club de tourner. C'est un très, très gros travail... La réalisation, les mises au point successives de JPCRom ou de EDIT ne sont pas non plus de petits morceaux. Qu'ils en soient ici remerciés.

Je dois ici réitérer mon appel à une plus large contribution de votre part à la réalisation de JPC. Il serait dommage que PPC mette les clés sous la porte, faute d'articles. C'est là le principal problème de JPC. Tous vos textes sont les bienvenus. Pourquoi ne publierait-on pas des programmes pour d'autres machines que les ordinateurs de poche?

Jean REIBEL

Pierre David
33 Bd St Martin
75003 Paris

Vend :
Une imprimante ThinkJet HP-IL : 1500 FF.

Jean REIBEL
9 square Victor Fleming
92350 LE PLESSIS ROBINSON
Tél (1) 46 31 46 11

Cherche:
Carte HPIL pour HP150

Vend:
Interface Vidéo (HPIL) 200FF

R. VERTENTE
8 rue des MARNAUDES
93250 VILLECOMBE

Recherche: Livre HP-28 Insights
de William Wickes ed. EDUCALC
en prêt ou occasion (ou autre)
c'est très urgent pour moi.
Merci d'avance.

MEMORY LOST

Rom, unique objet de mon ressentiment...

Le Bureau actuel du Club mérite notre reconnaissance pour le travail accompli au cours de cette année. Pour l'avoir vécu pendant 5 ans, nous sommes bien placés pour savoir les difficultés et la charge inhérentes à sa gestion. Nous comprenons donc que certains puissent souhaiter passer la main.

Tout au long de l'année, nous n'avons pas senti de difficulté importante et pourtant il semble qu'il y a avait des ressentiments qui maintenant font matière à pré-texte.

Pour ceux qui ont raté le début du film, rappelons que lorsque nous avions commencé JPC Rom, nous nous sommes confrontés à un ensemble de Lex disparates. Il a fallu en assurer la cohérence et l'homogénéité pour en faire *un* Lex : ceci représente un travail considérable auquel s'ajoutent les difficultés de la communication indispensable avec HP (attributions de numéros de Lex, etc.). Tout cela n'a été réalisable qu'après l'écriture de l'assembleur en C.

Il nous était facile de continuer ce travail précis puisque nous avions commencé depuis le début. Foin des anacoluthes, nous ne porterons pas le chapeau.

Nous ne nous attribuons pas la propriété de ce code. Les modifications essentielles ont toutes été publiées et nous n'avons pas fait de rétention d'information. Loin de tout profit personnel, nous n'avons retiré de JPC Rom que des soucis, dont cette lettre de notre ex-secrétaire empreinte de naïveté n'est que le dernier en date...

Nous ne saurions chagriner ne fusse qu'une seule personne, et même s'il nous semble désagréable d'être interpelés de cette manière, c'est volontiers que nous donnons les sources du travail tel qu'il est aujourd'hui.

Rappelons enfin que depuis plus d'un an, JPC Rom est vendu aux Etats-Unis, rapportant ainsi quelques milliers de francs chaque année sans aucun effort pour nos successeurs. Nous espérons que le Club saura continuer à faire fructifier ce travail.

Signé : certaines personnes très bien placées dans l'ancien Bureau...

L'ex-président, Pierre David
L'ex-trésorier, Janick Taillandier
L'ex-secrétaire, Jean-Jacques Dhénin

MISE AU POINT

Il était inévitable que des erreurs se tapissent dans les 13000 lignes de source composant l'éditeur en assembleur dont la parution a commencé dans *JPC 68*.

Ce qui devait donc arriver arriva donc. Les remarques des utilisateurs que nous sollicitions dans l'article ne se sont pas fait attendre... C'est Jean Reibel qui a ouvert le feu en signalant deux comportements pour le moins anormaux.

Si le premier est dû à une réelle erreur dans GENRPLC$, l'autre est le résultat d'un Lex erroné, mais fonctionnant sans histoire depuis plus de 4 années !

GENRPLC$

Le premier problème est mis en évidence par le segment de code suivant :

```
10 DIM A$
20 FOR I=1 TO INF
30     A$=GENRPLC$("","x","")
40     DISP I;MEM
50 NEXT I
```

L'exécution nous montre une mémoire perdant 145 octets à chaque itération, et une vitesse allant en s'amenuisant. Ce sont les signes caractéristiques des buffers non désalloués :
- n'étant justement pas désalloués, la place n'est pas récupérée pour être remise dans la mémoire disponible,
- le temps de recherche pour trouver un buffer libre est d'autant plus grand qu'il y a de buffers présents dans le système.

Une simple reconfiguration (éteindre et rallumer le HP-71 par exemple) suffit à récupérer la mémoire perdue.

Pour ceux que la technique intéresse, la correction se limite au simple déplacement d'une étiquette. Aux environs des lignes 310-320 dans le fichier source xaux.as, vous trouvez l'étiquette RPL900 située après la ligne GOSUBL =ENDPOS. Il suffit de déplacer RPL900 avant le GOSUBL, et nos ennuis sont terminés !

Pour ceux que la technique n'intéresse pas et pour les autres, voici comment faire pour mettre à jour votre éditeur. Faites les manipulations suivantes, en faisant attention à ne pas vous tromper !

```
A=HTD(ADDR$("EDIT"))
POKE DTH$(A+364),"8"
POKE DTH$(A+18558),"2"
```

Et voila ! Vous disposez maintenant de la version A8 de l'éditeur. Notez que la version distribuée par le Club est elle aussi corrigée.

CURLEX

Mais le plus étonnant est sans aucun doute l'erreur tapie dans CURLEX depuis plus de quatre ans !

L'effet était radical. Lorsque vous éditiez un fichier avec TEDIT ou XEDIT, et que vous vouliez connaître la position du curseur avec la séquence de touches [f] [VIEW] en mode USER, tout ce que vous obteniez était un Memory Lost après quelques manipulations.

Le plus surprenant était que les possesseurs de JPC Rom étaient à l'abri de ce comportement pour le moins étrange, alors que le code exécuté était le même !

Après quelques cachets d'aspirine, la vérité se fit jour. Pour comprendre, il faut savoir que CURLEX utilise une zone de 22 octets pour stocker une image de l'affichage du HP-71. Cette zone s'étend sur toute la *function scratch* et sur deux quartets de la *statement scratch*. Fort logiquement, le buffer commence à FUNCRO-2.

Le seul et unique problème de CURLEX venait de ce que FUNCRO était mal défini ! Deux chiffres étaient inversés, ce qui fait que CURLEX affichait la position du curseur en écrasant soigneusement, tel un éléphant dans un magasin de porcelaine, quelques zones cruciales pour XEDIT !

Mais alors, pourquoi le CURLEX inclus dans JPC Rom ne posait-il pas ce problème, bien qu'utilisant le même code ? La raison est que JPC Rom est assemblé avec le système de développement AREUH. Avec ce système, les définitions des étiquettes internes (telles que FUNCRO par exemple) sont déjà connues de l'assembleur, il n'y a donc pas de risque de se tromper...

Cette erreur est restée durant plus de quatre ans, car aucun autre Lex n'utilisait les zones utilisées par XEDIT et TEDIT... Il faut savoir que l'éditeur utilise le moindre recoin de Ram disponible.

Etonnant, non ?

Pierre David (37)
Janick Taillandier (246)

```
        LEX     'CURLEX'
* (c) Copyright PPC Paris 1986, 1987, 1988, 1989
        ID      #E1
        MSG     0
        POLL    POLHND
        ENDTXT

*
* Définition des valeurs système
*
=SFLAG? EQU     #1364C  Teste le flag système
=VIEWD1 EQU     #15147  Affiche la zone ^ D1
=HXDCW  EQU     #0EC84  Conversion Hex-Dec
=WIPOUT EQU     #1B0AF  Remplissage avec 0
```

```
=CURSOR EQU     #2F47E  Posit. du curseur [0..95]
*FUNCRO EQU     #2F8B9  arghhhh !
=FUNCRO EQU     #2F89B  Zone scratch

=pKYDF  EQU     #1B     KeY ReDefinition Poll
=flUSER EQU     #F7     User flag ( <0 )
=kcVIEW EQU     #0000B  Code de la touche VIEW

*
* Définition des valeurs locales
*
 BUF    EQU     (FUNCRO)-2

 POLHND LC(2)   =pKYDF  Poll KeY Definition
        ?B=C    B
        GOYES   POL20   Interception éventuelle

 POL10  RTNSXM          Retour à Basic sans action

 POL20  A=R0            R0(A)=Key Code
        LC(2)   =kcVIEW
        ?A#C    B
        GOYES   POL10   Retour si touche # VIEW

        C=D     A       SFLAG? abime D(A)
        B=C     A       B=D A
        LC(2)   =flUSER
        GOSBVL  =SFLAG? Tester le flag "User".
        C=B     A
        D=C     A
        GONC    POL10   Si désarmé, retour

        D1=(5)  BUF     Zone de 22 caractères
        C=0     A       C(A)=Longueur de la zone
        LC(2)   22*2
        GOSBVL  =WIPOUT à remplir avec le code 0

        D1=(5)  =CURSOR
        C=0     W       On ne gardera que C(B)
        C=DAT1  B       C(B)=position du curseur.
        GOSBVL  =HXDCW
*
* Après l'appel: Mode DEC et résultat dans A:
* A(W)=0...095 si le curseur est en 96ème position
*
        A=A+1   B       Curseur devient [1..96]
        SETHEX          Mode HEX pour conversion

        D1=(5)  BUF
        LCASC   '0'
        ASRC            A: 60...09
        ?A=0    P       Pas d'affichage poids fort
        GOYES   POL40

        A=A+C   B       Conversion DEC-ASCII.
        DAT1=A  B
        D1=D1+  2       Poids faible en 2ème
        A=0     B

 POL40  ASLC            A(W)= 00.....06
        A=A+C   B       Conversion
        DAT1=A  B

        D1=(5)  BUF     D1 := ^ zone
        GOSBVL  =VIEWD1 Affichage du buffer pointé

        XM=0            Poll intercepté
        ST=0    0       Action (et non redéf.)
        RTN

        END
```

FAUSSES NOTES

Quelques modifications (no. 67 page 13) :
1 Dans la liste GAMME :
Insérer "E" entre "D#" et "F";
2- Dans le corps du programme ORGUE, 14e ligne :
Au lieu de : "OCTAVE" "OCTAVE" lire : "OCTAVE" OCTAVE
(le 2e OCTAVE est une variable).

Roger LE BRIS (535)

Erratum

Dans une fonction LINE par Michel POLSKI dans le JPC n°68 page 6, Le programme ECHLI est erroné voici la bonne version, fournie par l'auteur.

```
<< 'PPAR' RCL DUP 'NPAR' STO LIST->
4 DROPN - C->R
31 SWAP / ABS 'PY' STO PY *H
136 SWAP / ABS 'PX' STO PX *W
>>
```

HP-28

Un défileur de texte pour HP 28C

Ce petit programme a pour but de faire défiler sur une ligne une chane de caractères à la vitesse voulue. Il a été conçu sur HP 28C et peut fonctionner sur HP 28S, mais il est certain que sur HP 28S cette routine est susceptible d'être améliorée avec les fonctions LCD-> et ->LCD par exemple...

Cette routine peut être intéressante pour les débutant car elle n'est pas compliquée à comprendre et fournit une application amusante -et O combien utile dans des programmes de jeux, des formulaires etc...- de la fonction SUB et du principe des boucles.

Mode d'emploi:

Entrez la chane de caractères entre guillemets. Entrez le délai d'attente entre l'impression de deux caractères consécutifs (de 0 à 1 seconde par exemple). Faites scrol ... Et amusez-vous !

N.B. : On peut changer la ligne de défilement en modifiant le numéro de ligne -ici 1- avant chaque instruction DISP.

```
SCROLL
<< SWAP DUP SIZE
-> V CH CS
   << CLLCD
   1 23 FOR I
      "                      "
      (il y a 22 blancs dans la chane vide)
      1 23 I - SUB CH 1 I SUB + 1 DISP V WAIT
   NEXT
   IF CS > 23 THEN
      24 CS FOR I
         CH I 22 - I SUB 1 DISP V WAIT
      NEXT
   END
   1 22 FOR I
      CH CS 21 - I + CS SUB 1 DISP V WAIT
   NEXT
   CLLCD V WAIT CLMF
   >>
>>
```

Michel POLSKI (531)

HP-41

HP-41 DDT ?

Non, il ne s'agit pas d'une série spéciale... Ce programme est simplement la simulation d'un "bug" du HP-41 : lorsque l'on fait un reset sur le troisième ton d'un BEEP, la machine émet une tonalité étrange (venue d'ailleurs) à chaque appui d'une touche, ou lors de l'exécution d'un programme.

Si vous vous demandez à quoi ça peut servir, voici quelques suggestions :
- XEQ MPR, XEQ CLOCK : vous avez maintenant une horloge qui égrène les secondes...
- Il est possible (à vérifier) que la fréquence émise fasse fuir les moustiques (d'où le titre). Avec un oscilloscope et un micro, ça pourrait se vérifier...
- Ce mode peut vous servir à vérifier le mode du microprocesseur : si la machine émet ce son désagréable, le microprocesseur est en mode running. Dans les autres cas, la machine est, soit arrêtée, soit en sommeil léger : elle attend l'appui d'une touche...

Remarques :

- Il est possible que ce programme bouffe vos piles en deux temps, trois mouvements...
-Pour supprimer l'effet de MPR, effectuez simplement un BEEP, ou un TONE non grésilleur.

Le programme :

```
B4C3 092 "R"
B4C4 010 "P"
B4C5 00D "M"
B4C6 05E C=0 MS
B4C7 006 A=0 S&X
B4C8 3C4 ST=0
B4C9 388 SETF 0
B4CA 019 ?NC GO
B4CB 05E ->1706
B4CC 3E0 RTN
```

H.P. à tous !!

Bertil Wiklund (508)

HP-71

TRI RAPIDE DE REPERTOIRES

Plusieurs programmes, relatifs aux répertoires de mémoires de masse, sont apparus dans JPC. Chacun ayant des objectifs différents et évoluant avec les facilités offertes par les nouvelles fonctions ou ordres de JCP Rom. L'auteur du plus récent de ces programmes, Alexandre Boldireff, nous proposait un tri de répertoires (SORTFILE : JPC 61 et 62) en utilisant d'une part un fichier-catalogue généré par l'ordre DDIR de JPC Rom et d'autre part un programme de tri alphanumérique de Laurent Istria (JPC 57).

Ce programme d'Alexandre facilite grandement le tri suivant différentes options mais présente deux inconvénients :

1-pour les disquettes riches de nombreux fichiers les temps de tri sont longs.
2-le programme duplique une partie, qui peut être très importante, du fichier-catalogue (un fichier de 252 entrées occupe 10586 octets sans compter la partie dupliquée), ce qui peut provoquer un encombrement mémoire génant.
Aussi j'ai essayé d'améliorer les choses grâce à un un programme en assembleur, et je vous propose donc un lex TRICATLX et deux nouvelles fonctions :
TRICAT fait deux choses : trie un fichier généré par DDIR ou PDIR et donne le nombre d'enregistrements triés. C'est à la fois un ordre pour le tri et une fonction qui renvoie un nombre.
OCTCAT donne la somme des octets des fichiers d'un répertoire ou des fichiers d'un type choisi.
Ces deux fonctions opèrent très rapidement avec beaucoup de possibilités et de souplesse.
Pour la suite tous mes exemples seront basés sur l'exploitatation du répertoire de la première disquette de JPC : CAT1 avec 252 fichiers.
Parlons déjà du gain de rapididité : un tri de l'ensemble de CAT1 avec le programme de Laurent prend, avec mon HP71, 852 secondes soit 14 minutes et 12 secondes. Avec TRICAT cela se fait en 5,5 secondes soit plus de 150 fois plus rapidement !

Mode d'emploi de TRICATLX

Pour les exemples on suppose que le répertoire de CAT1 s'appelle aussi CAT1.

TRICAT
syntaxe :
TRICAT ("fichier"[,"option"]) \Fichier : nom de fichier généré par DDIR ou PDIR

sans option

TRICAT ("CAT1") -> 252

le répertoire CAT1 est trié alphabétiquement tous types de fichiers confondus et la disquette CAT1 est composé de 252 fichiers.

les options :

première série : tri alpha pour chaque type.

ALL BASIC BIN DATA D-LEX FORTH KEY LEX SDATA TEXT ROM 41 75

TRICAT ("CAT1","ALL") ou TRICAT ("CAT1","AL")

-> 252

CAT1 est trié par type et chaque série d'un type est triée alphabétiquement (ce que ne fait pas le programme d'Alexandre) donc pour les 252 fichiers.
TRICAT ("CAT1","BASIC") ou TRICAT ("CAT1","BA")
-> 107
les 107 premiers enregistrements de CAT1 sont ceux de type BASIC et triés alphabétiquement.
On peut donc se contenter d'écrire les 2 premiers caractères de l'option.

deuxième série :
tri seulement par type mais non alpha.
il suffit d'ajouter le préfixe "N" à l'option.
NALL NBASIC NBIN N75
TRICAT ("CAT1","NALL") ou TRICAT ("CAT1","NAL)
-> 252
/la différence avec TRICAT ("CAT1","ALL") est que ici le répertoire n'est trié que par type et donc beaucoup plus rapidement : 0,48 secondes au lieu de 5,5 secondes.\
TRICAT ("CAT1","NLEX") -> 50
les 50 premiers enregistrements sont de type LEX mais non triés alphabétiquement.

on pourra donc lister CAT1 trié par type et alpha en faisant tout simplement :
TRICAT ("CAT1","AL") @ PLIST CAT1
pour lister les fichiers LEX seuls, par exemple, on pourra faire :
N=TRICAT ("CAT1",LE") @ PLIST CAT1,1,N

OCTCAT
syntaxe :
OCTCAT ("fichier"[,"option"])
même données, mêmes options que pour TRICAT
le préfixe "N" devant les options est inopérant.

OCTCAT ("CAT1") ou OCTCAT ("CAT1","ALL")
-> 553079
l'ensemble des fichiers de CAT1 représente 553709 octets.
OCTCAT ("CAT1","BA")+OCTCAT ("CAT1","LE")
-> 215530

Deuxième syntaxe :
TRICAT (no de canal,["option"]) \OCTCAT (no de canal["option"])
exemple pour imprimer la liste des fichiers basic et des fichiers lex de CAT1 sans les dates :

```
ASSIGN #1 TO CAT1 @ DIM A$[40]
N=TRICAT (1,"BA") @ FOR I=0 TO N-10
READ #1;A$ @ PRINT A$[1,24] @ NEXT I
N=TRICAT (1,"LE") @ FOR I=0 TO N-1
READ #1,I;A$ @ PRINT A$[1,24] @ NEXT I
ASSIGN #1 TO *
```

Quelques idées d'utilisation de TRICATLX :
-recherche de l'existence ou non d'un type de fichier sur un support de masse sans avoir à afficher tout le répertoire.
-extraire de disquettes-archives, telles celles de JPC, les fichiers BASIC et LEX par exemple pour les implanter sur une disquette d'usage courant.
-inventorier une "disquette-brouillon" pour y faire le "ménage". Supprimer les anciennes versions de programmes.
/-faire un unique répertoire ordonné de plusieurs supports de masse.\

Je ne propose pas de programmes BASIC pour l'utilisation de TRICATLX, laissant à d'autres le soin de le faire (en particulier je m'adresse à Alexandre), et j'espère voir fleurir des versions toutes plus performantes les unes que les autres.
En ce qui me concerne TRICATLX m'a été très utile pour "éplucher" le contenu des SWAP DISCS d'Educalc où souvent plusieurs versions d'un même programme ont été réalisées à différentes dates où ont été collectées par diverses personnes.

Guy Toublanc (276)

LEX 'TRICATLX'

```
* TRI des fichiers-catalogues générés par les mots
* DDIR et PDIR de JPC Rom.
* Conception et codage : Guy Toublanc (276)
* 3 octobre 89
* ajout possibilité d'utilisation d'un numéro de
* canal et suppression d'une bogue relatives aux
* catalogues de MAIN. 6 novembre 89 .
        ID      #5C
t       EQU     #31
        MSG     0
        POLL    0
        ENTRY   OCTe
        CHAR    #F
        ENTRY   TRIe
        CHAR    #F
        KEY     'OCTCAT'
        TOKEN   t
        KEY     'TRICAT'
        TOKEN   1+t
        ENDTXT
=?PRFI+ EQU     #17380
=A-MULT EQU     #1B349
=ARGERR EQU     #0BF19
=AVMEME EQU     #2F599
=CHKmem EQU     #012C7
=CSLW5  EQU     #0ED3D
=CSRW5  EQU     #0ED2C
=FILXQ$ EQU     #09B95
=FINDF+ EQU     #09F63
=FLOAT  EQU     #1B322
=FNRTN1 EQU     #0F216
=F-RO-2 EQU     #2F8A5
=FIBADR EQU     #11457
=HDFLT  EQU     #1B31B
=IDIVA  EQU     #0EC6E
=MFERR  EQU     #09393
=RAMROM EQU     #0A5F7
=REVPOP EQU     #0BD31
=RNDAHX EQU     #136CB
=RSTD0  EQU     #06832
=RSTD1  EQU     #1C596
=SAVD0  EQU     #1C587
=SAVD1  EQU     #1C578
=eFACCS EQU     #3C
=eFTYPE EQU     #3F

        NIBHEX 4C12
OCTe    ST=1    3
        GONC    SORT

        NIBHEX 4C12
TRIe
SORT
        GOSBVL  =SAVD0
        P=C     15
        C=0     A
        ?P=     1
        GOYES   filsp
        GOSBVL  =REVPOP  routine de vérification
        CD1EX            de l'option choisie
        D0=C             puis de la nature
        A=A+C   A        du fichier catalogue
        D1=A
        C=0     W
        LCASC   'N'
        A=DAT0  2
        ?A#C    B
        GOYES   nonumb
        ST=1    0
```

```
        D0=D0+ 2
nonumb
        C=DAT0 4
filsp   GOSBVL =CSLW5
        R4=C
        A=DAT1 S
        A=A+1  S
        GOC    filxq
        GOSBVL =RNDAHX
        D1=D1+ 16
        B=A    A
        BSR    A
        BSR    A
        ?B=0   A
        GOYES  cgood
        GOTO   argerr
cgood   GOSBVL =SAVD1
        GOSBVL =FIBADR
        D1=D1+ 5
        C=0    A
        C=DAT1 4
        D1=D1+ 4
        A=DAT1 S
        D1=D1+ 3
        C=DAT1 S
        D=C    S
        D1=D1+ 9
        A=DAT1 A
        D1=A
        D1=D1- 5
        GONC   tsttyp
filxq   GOSBVL =FILXQ$
        GONC   mferr
        GOSBVL SAVD1
        GOSBVL =FINDF+
        GOC    mferr
        D1=D1+ 16
        C=0    A
        C=DAT1 4
        D1=D1+ 4
        A=DAT1 S
        D1=D1+ 12
tsttyp
        C=C-1  A
        C=C-1  A
        GOC    textf
etype   LC(2)  =eFTYPE
mferr   GOVLNG =MFERR
textf   GOSBVL =RAMROM
        LC(2)  =eFACCS
        GONC   mferr
        GOSBVL =?PRFI+
        GOC    mferr
        A=DAT1 A
        R2=A
        D1=D1+ 9
        AD1EX
        R0=A
        AD1EX
        D1=D1- 4
        C=0    A
        P=     2
        LC(2)  40
        ST=0   1
        A=0    A
        A=DAT1 4
        ?A=C   A
        GOYES  disk
        ST=1   1
        C=C+1  P
        C=C+1  P
        ?A=C   A
        GOYES  port
        LC(2)  37
        ?A#C   A
        GOYES  BAD
        ST=1   2
        C=C+1  P
port
disk    CSR    A
        CSR    A
        C=C+C  A
        D=C    A
        D=D+1  A
        D=D+1  A
        D=D+1  A
        D=D+1  A
        A=R2
        P=     0
        LC(2)  9
        A=A-C  A
        C=D    A
        R1=C
        GOSBVL IDIVA
        P=     0
        GONC   OK
BAD     P=     0
        GOTO   etype
OK      R3=A
        ?A=0   A
        GOYES  BAD
        A=A-1  A
        C=0    A
        P=     2
        LC(2)  40
        ?ST=0  1
        GOYES  disk2
        C=C+1  P
        C=C+1  P
        ?ST=0  2
        GOYES  port2
        LC(2)  37
port2
disk2   B=C    A
        P=     0
TEST    C=DAT1 4
```

```
        ?C=B    A
        GOYES   start
        C=R3
        C=C-A   A
        C=C-1   A
        ?C=0    A
        GOYES   BAD
        R3=C
        GONC    fin??
start   CD1EX
        C=C+D   A
        CD1EX
        A=A-1   A
        GONC    TEST
fin??   C=0     A
        C=C-1   A
        C=DAT1  4
        C=C+1   A
        GONC    BAD
        D1=(5)  =AVMEME
        A=DAT1  A
        C=D     A
        GOSBVL  =CHKmem
        GONC    OK2
        GOTO    mferr
OK2     D1=A
        C=0     A
        D0=(5)  =F-R0-2
        DAT0=C  A
        C=R4
        GOSBVL  =CSRW5
        D=C     A
        ?C#0    A       recherche du type
        GOYES   tal     de fichiers choisi
        ?ST=0   3       ou s'il s'agit du
        GOYES   gtalph  du calcul de la somme
gtsoct  GOTO    soct    des octets pour
gtalph  GOTO    alpha   les fichiers retenus
tal     LCASC   'LA'
        ?C#D    A
        GOYES   ba
        ?ST=1   3
        GOYES   gtsoct
        ST=1    5
ba      LCASC   'AB'
        GOC     alnul
        A=R3
        D0=D0+  6
        DAT0=A  A
        GOSUB   CATRI
        GOTO    ib
alnul   GOSUB   choix
ib      LCASC   'IB'
        GOSUB   choix
        LCASC   'AD'
        GOSUB   choix
        LCASC   '-D'
        GOSUB   choix
        LCASC   'OF'
        GOSUB   choix
        LCASC   'EK'
        GOSUB   choix
        LCASC   'EL'
        GOSUB   choix
        LCASC   'OR'
        GOSUB   choix
        LCASC   'DS'
        GOSUB   choix
        LCASC   'ET'
        GOSUB   choix
        LCASC   '14'
        GOSUB   choix
        LCASC   '57'
        GOSUB   choix
        ?ST=1   5
        GOYES   sorta
argerr  GOVLNG  =ARGERR
alpha   GOSUB   TRIA
        GOTO    sorta
choix   ?ST=0   3
        GOYES   all?
        ?C#D    A
        GOYES   return
        ST=1    4
        R2=C
        C=RSTK
        GOTO    soct
all?    ?ST=0   5
        GOYES   choixy
        GOTO    CATRI2
choixy  ?C=D    A
        GOYES   gsbcat
return  RTN
gsbcat  C=RSTK
        C=D     A
        GOSUB   CATRI
sorta   D0=(5)  =F-R0-2
sortal  A=DAT0  A
        GOSBVL  HDFLT
SORTIE  GOSBVL  RSTD0
        GOSBVL  RSTD1
        C=A     W
        GOVLNG  FNRTN1
CATRI   R2=C
CATRI1  A=R3            préparation
        A=A-1   A       pour le tri
        C=R1            calcul des adresses
        GOSBVL  =A-MULT et des limites
        C=R0            puis tri des fichiers
        C=C+A   A       pour le type choisi
        CD1EX
        RSTK=C
        A=R0
        D0=A
        D=0     A
        C=R2
```

```
        B=C    A
        C=R3
        C=C-1  A
        D0=D0+ 14
        D0=D0+ 12
        D1=D1+ 14
        D1=D1+ 12
        ?ST=0  1
        GOYES  disk3
        D0=D0- 4
        D1=D1- 4
disk3   A=0    A
        A=DAT0 4
        ?A#B   A
        GOYES  nobah
        AD0EX
        CR1EX
        A=A+C  A
        AD0EX
        CR1EX
        D=D+1  A
        ?C>D   A
        GOYES  disk3
fin
        GOTO   TRIAC
nobah   A=DAT1 4
        ?A=B   A
        GOYES  bxh
        C=C-1  A
        ?C<=D  A
        GOYES  fin
        A=R1
        CD1EX
        C=C-A  A
        CD1EX
        A=0    A
        GONC   nobah
bxh     CD1EX
        RSTK=C
        CD1EX
        CD0EX
        RSTK=C
        CD0EX
        RSTK=C
        D0=D0- 14
        D0=D0- 12
        D1=D1- 14
        D1=D1- 12
        ?ST=0  1
        GOYES  disk4
        D0=D0+ 4
        D1=D1+ 4
disk4   P=     4
ex      A=DAT0 16
        C=DAT1 16
        DAT0=C 16
        DAT1=A 16
        D0=D0+ 16
        D1=D1+ 16
        P=P-1
        GONC   ex
        ?ST=0  1
        GOYES  rtex
        ?ST=0  2
        GOYES  port3
        D0=D0- 4
        D1=D1- 4
port3   A=DAT0 4
        C=DAT1 4
        DAT0=C 4
        DAT1=A 4
rtex    A=R1
        C=RSTK
        D1=C
        C=RSTK
        C=C+A  A
        D0=C
        C=RSTK
        C=C-A  A
        CD1EX
        D=D+1  A
        C=C-1  A
        ?C>D   A
        GOYES  gdisk
        GOTO   fin
gdisk   GOTO   disk3
TRIAC
        C=RSTK
        A=C    A
        C=0    A
        C=DAT0 4
        ?C#B   A
        GOYES  ston
        D=D+1  A
ston    C=D    A
        R3=C
        P=     0
        D1=A
TRIA    A=R3            tri alpha
        D0=(5) =F-R0-2  tri shell
        C=DAT0 A
        C=A+C  A
        DAT0=C A
        D0=D0+ 11
        DAT0=A A
        C=R1
        ?ST=1  0
        GOYES  res
        A=A-1  A
        ?A#0   A
        GOYES  dif1
res     R2=C
        RTN
dif1
        GOSBVL =A-MULT
        C=R0
```

```
        A=A+C  A
        AR3EX
        A=A+1  A
        C=0    A
        LC(1)  9
        GOSBVL =IDIVA
        C=0    A
        C=C+1  A
detP    ?C>=A  A
        GOYES  Pfait
        B=C    A
        C=C+C  A
        C=B+C  A
        C=C+1  A
        GONC   detP
Pfait   A=C    A
        C=R1
        RSTK=C
whileP  GOSBVL =A-MULT
        ?A#0   A
        GOYES  contP
        C=RSTK
        R2=C
        P=     0
        RTN
contP   R2=A
        C=R0
        C=A+C  A
        D=C    A
        C=RSTK
        RSTK=C
        C=A-C  A
        A=R0
        C=A+C  A
        RSTK=C
        C=R0
        R1=C
forK    C=D    A
forI    RSTK=C
        D0=C
        AD1EX
        D1=A
        GOSUB  TRANS
        D1=D1+ 16
        P=     7
S$      D1=D1- 2
        CSL    W
        CSL    W
        C=DAT1 2
        P=P-1
        GONC   S$
        B=C    W
        C=RSTK
        RSTK=C
whileJ  A=R2
        C=C-A  A
        RSTK=C
        ?C<D   A
        GOYES  endwJ
        GOSUB  eq
        GOC    endwJ
        C=RSTK
        RSTK=C
        D0=C
        A=R2
        A=A+C  A
        AD1EX
        GOSUB  TRANS
        C=RSTK
        GONC   whileJ
endwJ   C=R1
        GOSUB  eq
        GOC    CSJ+P
        C=RSTK
        A=R2
        C=R1
        C=C-A  A
        RSTK=C
        C=R1
        D0=C
        C=D    A
        CD1EX
        A=C    A
        GOSUB  TRANS
CSJ+P   C=RSTK
        A=R2
        A=A+C  A
        AD1EX
        D0=A
        GOSUB  TRANS
        C=RSTK
        A=R2
        C=A+C  A
        A=R3
        ?C>A   A
        GOYES  nextK
        GOTO   forI
nextK   C=RSTK
        B=C    A
        C=RSTK
        RSTK=C
        A=R1
        A=A+C  A
        ?A>B   A
        GOYES  calcP
        R1=A
        D=D+C  A
        C=B    A
        RSTK=C
        GOTO   forK
calcP   A=C    A
        D=C    A
        C=C+C  A
        C=A+C  A
        A=R2
        GOSBVL =IDIVA
```

```
        C=D    A
        GOTO   whileP

TRANS   ?ST=0  2        routine d'echange
        GOYES  nomain   de deux enregistrements
        P=     3
        GONC   RI
nomain  P=     4
RI      C=DAT0 16
        DAT1=C 16
        D0=D0+ 16
        D1=D1+ 16
        P=P-1
        GONC   RI
        ?ST=0  1
        GOYES  retour
        ?ST=0  2
        GOYES  port4
        C=DAT0 12
        DAT1=C 12
        GONC   retour
port4   C=DAT0 4
        DAT1=C 4
retour  D1=A
        RTNCC

eq      D0=C           routine de comparaison
        A=B    W       des noms de deux
        P=     7       fichiers
eqSCI   C=DAT0 2       jusqu'à 8 caractères
        ?A>C   B
        RTNYES
        ?A<C   B
        GOYES  retrn
        D0=D0+ 2
        ASR    W
        ASR    W
        P=P-1
        GONC   eqSCI
        RTN
retrn   RTNCC

CATRI2  C=P    4       prépare le tri
        CR2EX          par type de fichiers
        R1=C           pour l'option 'all'
        D0=(5) =F-R0-2
        C=DAT0 A
        D0=D0+ 6
        A=DAT0 A
        A=A-C  A
        R3=A
        A=A-1  A
        GOC    finall
        ?A=0   A
        GOYES  finall
        D0=D0+ 5
        A=DAT0 A
        C=R1
        GOSBVL =A-MULT
        C=R0
        A=A+C  A
        R0=A
        GOTO   CATRI1
finall  C=RSTK
        GOTO   sortal

soct    A=R0           routine de calcul
        D1=A           du nombre d'octets
        D1=D1+ 16      par type de fichiers
        D1=D1+ 6       ou pour tous les
        ?ST=1  1       fichiers
        GOYES  hp71
        D1=D1+ 4
hp71    C=R3
        C=C-1  A
        D=C    A
        C=0    W
        CR1EX
        ?ST=1  4
        GOYES  chx
        D1=D1+ 12
        CD1EX
        D1=D1- 10
        CD1EX
chx     B=C    A
        GOC    chx2
        C=0    W
        A=0    W
        GOSUB  clcoct
chx2
bcchx   A=R2
        C=0    A
        C=DAT1 4       identifie les fichiers
        ?A#C   A       du type choisi
        GOYES  nofind
        D1=D1+ 12
        C=R1
        GOSUB  clcoct
        R1=C
        D1=D1- 16
        D1=D1- 6
nofind  AD1EX
        A=A+B  A
        D1=A
        D=D-1  A
        GONC   bcchx
        C=R1
        GOSUB  gtout
clcoct  P=     4
bcoct   ASL    A       calcul de la somme
        A=DAT1 1       des octets pour
        D1=D1+ 2       les fichiers retenus
        P=P-1
        GONC   bcoct
        SETDEC
        C=A+C  W
```

```
        SETHEX
        ?ST=1  4
        RTNYES
        CD1EX
        C=C+B  A
        CD1EX
        D=D-1  A
        GONC   clcoct
gtout   A=C    W
        C=RSTK
        GOSBVL =FLOAT
        GOTO   SORTIE


        END
```

DISCUSSIONS SUR LES PROGRAMMES INTERNES DE LA HP71

La compréhension des programmes internes de la HP71 (les IDS) représente une étape nécessaire pour pouvoir attaquer une programmation en langage Assembleur. Cette tache est à la fois difficile et elle nécessite un emploi du temps qui n'est pas à la portée de tous les membres du club. Mais il reste à verifier si cette présentation trouvera des gens intéressés non seulement dans la lecture mais dans une participation active capable d'étendre le texte des articles présentés. Avant d'attaquer le fonds du problème on va essayer d'expliquer la signification d'une série de termes.

Il y a 5 mots anglais dont on essayera d'élucider le sens à la fois comme signification technique et générale. Les mots qu'on veut présenter sont: token, trap, hook, poll, parse.

token = quelque chose qui sert comme signe, symbole, ou signification. Le terme qu'on utilisera sera 'symbit' et la définition est un bit qui représente l'ensemble sous forme codée d'une instruction, fonction ou texte.

trap = quelque chose qui provoque subitement un arrêt ou une entrave; un nib du code hexa qui établit l'action à suivre dans le cas d'une exception arithmétique. Le terme qu'on va utiliser sera entrave d'erreur.

hook = un levier pour actionner sur un dispositif; terme utilisé dans le mécanisme de sélection (polling). on utilisera le terme crochet.

poll = faire campagne dans une élection. Demander à chaque membre d'une délégation de préciser son vote individuellement. Etablir si chaque fichier LEX présent dans la machine est capable d'assumer une tache specifiée ;le terme qu'on utilisera sera mécanisme de sélection.

parse = faire une analyse grammaticale d'un mot ou groupe des mots. Coder selon les règles de base du logiciel, xrom1 et les tables LEX, le texte, les instructions, les mots-clef, les fonctions, etc. On utilisera le terme analyse.

Le mécanisme de sélection contient des crochets qui per- mettent aux fichiers LEX d'intercepter l'opération qu'ils doivent executer sur la machine. Il existe plus de 80 endroits dans le code opératoire du système qui donnent la possibilité d'exécuter une tache spéciale, par ex. analyser le nom d'un dispositif ou finir l'exécution d'un programme. Ces points permettent de vérifier si les fichiers LEX présents dans la mémoire de la machine peuvent remplir la tache demandée. Le système opératif est connecté au code opératif du LEX ou il présente un symbit (sous forme de nombre), qui identifie la tache qui doit être executée. Il est clair que le fichier LEX peut accepter l'exécution de la tache s'il contient cette procédure adéquate. De cette façon on particularise très avantageusement l'utilsation du HP71.
En ce qui concerne l'analyse lexicologique quand l'analyseur lexique établit la succesion des symbits (tokens) on cherche dans les tables textes asociées aux LEX pour déterminer une chaîne concordante et on associe le symbit respectif qui correspond au mot-clef. A partir de ce symbit qui constitue un index dans la table principale qui fournit l'adresse d'exécution. Pour une instruction l'adresse du code exécution est precedée par une adresse (5 nibs) qui donne la distance relative vers la procédure d'analyse (parsing). Pour une fonction le code d'exécution est precedée par le nombre des paramètres et leurs descriptions ce qui permet l'analyse complète de la fonction.
Le mécanisme de la décompilation utilise les symbits et on cherche à identifier les fichiers LEX qui possèdent un identificateur identique. Le symbit sert comme index dans la table principale. L'analyseur trouve donc dans la table principale :(1) l'emplacement du mot-clef dans la table texte, (2)l'adrese d'exécution. Une instruction contient 10 nibs avant l'adresse d'exécution la distance (5 nibs) vers la procédure de décompilation. Pour une fonction on a précisé que cet emplacement est reservé à la description des paramètres.

LE BUFFER DE CONFIGURATION.

ADRESSE LISTE DE LA MEMOIRE DU BUFFER DE

```
          CONFIGURATION
2F9E6     FFE1001010E0030300  --  ID du Buffer
2F9F6     20E023030030E043
2FA06     030EF230000E0050        Longueur du Buffer
2FA16     30050A0041000019
2FA26     00B000002800C100
2FA36     004A0841000FA000        Fin du Buffer
2FA46     40F000F000034F4E        Chainage du fichier
2FA56     4020202020210000
2FA66     42100082118850209
```

Il y a trois buffers : (1) FF System RAM, (2) FE IRAM, ROM, EEPROM, etc. (3) FD Memory-mapped I/O (HPIL mailbox, etc). On va analyser la structure des buffers de configuration. On a d'abord l'identificateur (FF pour le premier buffer) L'identificateur est suivi par 3 nibs, un pointeur de longueur qui fixe la distance vers le suivant buffer ou vers un octet 0 ,qui marque la fin du buffer de configuration et le commencement du chaînage des fichiers. Il faut préciser que tous les fichiers forment une chaîne et que chaque fichier pointe sur le fichier suivant.

Comme on peut voir dans le listing presenté plus haut l'adresse où commence la chaîne des fichiers est '2FA51' et, on peut verifier que le pointeur 'MAINST'(=2F558) donne effectivement PEEK$('2F558',5)=15AF2 . Le premier fichier étant "CON", on a ADDR$('CON')=2FA51 et HTA$(PEEK$('2FA51',16))='CON'. Le contenu de chaque buffer de configuration forment une structure de 10 nibs selon le format de la table qui suit:

```
N Signification        FFFFFFFEFEFEFEFEFD  mss=module à
                                           sequence simple
0 Position Sequece     000000000 Position dans le module
                                 (0 pour mss)
1 Numero Ensemble      123050004 Position du module
                                 (pour un port-ext.)
2 Numero Port          000001240 Les RAMs du chaînage
                                 peuvent être groupées
3 Log2(Grandeur)       EEEEA98AF en modules logiques comme
                                 par ex. port .05)
4                      000000000 Pour des éléments de
                                 mémoire les 3 nibs
5 ADRESSE(kbit)        024000080 supér. sont donnés
                                 (les 2 inf. sont 0)
6                      333300000 Pour MMp-I/O device les
                                 3 nibs au milieu
7 Type du Dispositif   00001011F sont donnés (le nib. sup.
                                 est toujours 1
8 NO chip/seq-1 ou     333300000 nib. inf. est 0. Pour
                                 la première catégorie
                                 l'unité est le knib &
                                 pour le second il existe
9 Nibble 1 du   ID     000000000 se sont des mots relatifs
                                 à l'adresse 20000
La table de grandeur:
```

Nibble	F	E	D	C	B	A	9	8	7	6
Kbyte	0.5	1	2	4	8	16	32	64	128	256

Le type de dispositif identifie si on a affaire à un élément mémoire ou entrée/sortie realisée en mémoire (MM E/S). Les valeurs des nibs sont: 0 RAM, 1 ROM, 2 EEPROM, F MM E/S, 3-5 non alloués, 6-E non-permis.

La classe de l'élément est un index précisant le type du MM E/S. Il n'y a pas de classe pour les éléments mémoire. La valeur du nib est 0 pour le bloc HPIL; les autres valeurs ne sont pas asignées.\ \Le nombre du CHIP dans la sequence précise combien il y en a. Il est retenu sous la forme #chip-1. Ce index est 0 pour MM E/S parce que cette catégorie a une table propre. Enfin nib 9 est reservé et d'habitude il n'est pas défini; il représente nib 1 de l'identificateur du CHIP.

Il existe cinq sous-buffers FE représentant les ports: 0, .05, 1, 2, 4. L'allocation était la suivante : port(4) FORTH/ASSEMBLER , port(2) EPROM-64k , port(1) RAM-32k. Il faut souligner que chaque module indépendant a un identificateur de 8 nibs avec la valeur hex 'B3DDDDDE' ou '00000000' . En consequence le titre du fichier, dans chaque port, commence 8 nibs plus loin que le commencement du buffer. En ce qui concerne la partie incipiente du fichier elle a le format suivant:

TITRE FICHIER	Nib.
Nom fichier	16
Type fichier	4
Drapeaux	1
Code pour copier	1
Temps de création	4
Date de création	6
Longueur du chain.	5

Fich.Type Table	N	S	P	E
BASIC	E214	E215	E216	E217
BIN	E204	E205	E206	E207
DATA	E0F0	E0F1		
LEX	E208	E209	E20A	E20B
KEY	E20C	E20D		
SDATA	E0D0			
TEXT	0001	E0D5		
FORTH	E218	E219	E21A	E21B

Les quatre bits du 'drapeau' ont la signification suivante: bit 0 amorcé signifie que le fichier est securisé, pour le bit 1 le fichier est privatisé. Dans le cas ou les deux bits 0&1 sont excités le fichier est de type E ,à la fois securisé et privatisé.

Le titre du fichier en fonction du code copie(voir table suivante) peut être suivi d'un champ 'implementation' et d'un sous-titre. L'implementation , est une zone de 8 nibs qui, pour fichier type DATA ,contient le numero (4 nibs) et la longueur de l'enregistrement (4 nibs). Le sous-titre (d'un fichier BASIC) a d'abord un entête (5 nibs) qui donne le chaînage vers le premier sous-programme et après un autre groupe de 5 nibs qui pointe vers une etiquette ou fonction definie par l'utilisateur. Un fin de ligne(EOL) finit le sous-titre. Cet ensemble a le rôle d'accelerer la recherche d'une etiquette ou fonction utilisateur.

LA STRUCTURE DE L'ENTETE D'UN FICHIER EN FONCTION DU CODE COPIE

Code copie	0	1	2	4	8
Type fich(ex.)	BASIC	DATA	SDATA	TEXT	UD
Champs IMPLEM.	Non	Oui	Non	Non	Oui
I.F.(ext media)	A	B	C	No	UD
Soustitre	Oui	Non	Non	Oui	Oui

UD = definit par l'utilisateur

Le champs 'implementation' d'un fichier sur un milieu exteriur (e.m.) suit le format A,B ou C. Les fichiers BASIC,LEX et KEY utilisent 6 nibs donnant la longueur (format A),les fichiers DATA utilisent le format B, et les fichiers SDATA le format C qui a 6 nibs, 2 pour protection et 4 pour specifier le nombre du registre (le registre étant un enregistrement d'une longueur fixe de 8 bytes).
Le programme FILECH vous permet d'obtenir une liste (dump) de la mémoire avec les adresses de chaque fichiers et les chaînages corespondantes et cette liste est accesible même si un fichier est protégé. Enfin l'annexe 1 présente un dump de la mémoire du HP71. Je souligne que en envoyant une disquette vous pouvez obtenir un fichier source qui par l'utilisation du programme REPORT vous permettra d'avoir un tableau imprimé en charactères gras.

Aurel ROTTMAN

REVISEZ VOS MATH(LEX)

Certains d'entre vous ont peut-être remarqué que le programme MATHLEX (*JPC 64*) réagisait de façon plus ou moins curieuse. En effet, pour exécuter la fonction NEP par exemple, il fallait taper DTB$!

L'explication est simple : Dans l'entête du Lex, si les déclarations des mnémoniques dans la Text Table étaient bien dans l'ordre alphabétique, les références des points d'entrées dans la Main Table auraient dû être *dans l'ordre des Tokens*, alors qu'elles étaient aussi dans l'ordre alphabétique.

Voici donc un extrait de la version corrigée.

```
      LEX    'MATHLEX'

AD2-12 EQU    #0C35F
AD2-15 EQU    #0C363
       .
       .
uRES12 EQU    #0C994

       ID     #5E
       MSG    0
       POLL   p

       ENTRY  decl
       CHAR   #D
       ENTRY  conje
       CHAR   #F
       ENTRY  impte
       CHAR   #F
       ENTRY  mage
       CHAR   #F
       ENTRY  repte
       CHAR   #F
       ENTRY  acoshe
       CHAR   #F
       ENTRY  asinhe
       CHAR   #F
       ENTRY  btde
       CHAR   #F
       ENTRY  cbre
       CHAR   #F
       ENTRY  coshe
       CHAR   #F
       ENTRY  dtbe
       CHAR   #F
       ENTRY  nepe
       CHAR   #F
       ENTRY  sinhe
       CHAR   #F

       KEY    'ACOSH'
       TOKEN  32
       KEY    'ASINH'
       TOKEN  33
       KEY    'BTD'
       TOKEN  34
       KEY    'CBR'
       TOKEN  35
       KEY    'COMPLEX'
       TOKEN  27
       KEY    'CONJ'
       TOKEN  28
       KEY    'COSH'
       TOKEN  36
       KEY    'DTB$'
       TOKEN  37
       KEY    'IMPT'
       TOKEN  29
       KEY    'MAG'
       TOKEN  30
       KEY    'NEP'
       TOKEN  38
       KEY    'REPT'
       TOKEN  31
```

```
        KEY    'SINH'
        TOKEN  39
        ENDTXT


        NIBHEX 811
dtbe    GOSUB  pop1r
        D1=D1+ 16
          .
          .
          .
```

Jacques Belin (123)

HP71 MEMORY MAP	ADDRESS	LABEL	REMARKS
	00000		
Operating System ROM			
	1FFFF		
	20000		
Memory Mapped I/O			
	2BFFF		
	2C000		
Card Reader			
	2C01F		
	2E100		
Display RAM			
	2E3FF		
	2F400		
System RAM			
	2F986		Display Driver
Reserved RAM			
	2F9E6	CONFST	RAM
Configuration BUFFER			
	2F558	*)MAINST	*)from here pointers
Main File Chain			
	2F571	MAINEN	
			Addr. 30000
	2F571	IOBFST	
System BUFFERS			
	2F576	IOBFEN	Soft configured
	2F57b	CLCBFR	and
Command Stack			
	2F57B	RFNBFR	Plug-in
Calc Mode BUFFERS			
	2F58F	OUTBS	RAM
Output BUFFERS			
	2F594	AVMEMS	
Free User RAM			
	2F599	AVMEME	
	2F599	MTHSTK	
Mathematic Stack			
FOR/NEXT Stack	2F59E	FORSTK	
GOSUB Stack	2F5A3	GSBSTK	
Active Variables	2F5A8	ACTIVE	
	2F5AD	CALSTK	
Prior Environments (from CALLs) includes: FOR/NEXT Stack, GOSUB Stack, Variables			
	2F5B2	RAMEND	
Plug-in ROM, Independent RAM			
	FFC00		
Configuration Reserved Area	FFFFF		

```
0010 ! FILECHAIN RAM VISUALISATION
0020 CALL FILECH @ SUB FILECH
0030 DIM A$(7)
0040 PRINT 'MAIN'
0050 A$(1)=XFN82012(PEEK$('2F558',5))
0060 CALL AD(A$(),B$) @ IF PEEK$(B$,2)='00' THEN 80 ELSE A$(1)=B$ @ GOTO 60
0070 IF PEEK$(B$,2)='00' THEN 80 ELSE A$(1)=B$ @ GOTO 60
0080 A$(1)='2F9E6'
0090 IF PEEK$(A$(1),2)#'FF' THEN DISP 'CONFIG.ERROR' @ STOP
0100 C=HTD(A$(1))+HTD(XFN82012(PEEK$(A$(1),5))[1,3])+5
0110 IF PEEK$(DTH$(C),2)='EF' THEN N=HTD(XFN82012(PEEK$(DTH$(C+2),3)))/10
0120 FOR J=1 TO N
0130 D=C+(J-1)*10+5 @ SFLAG 1
0140 A$(1)=XFN82012(PEEK$(DTH$(D+4),3))&'08'
0150 D1=D+2 @ D2=D1-1
0160 E$=PEEK$(DTH$(D1),1) @ F$=PEEK$(DTH$(D2),1)
0170 IF F$='0' THEN PRINT 'Port....',E$ @ GOTO 190 ELSE 180
0180 PRINT 'Port....',E$&'.0'&F$
0190 CFLAG 1
0200 CALL AD(A$(),B$)
0210 IF PEEK$(B$,2)='00' THEN 240
0220 A$(1)=B$
0230 GOTO 200
0240 NEXT J
0250 END
0260 SUB AD(A$(),B$)
0270 A$(2)=DTH$(HTD(A$(1))+16)
0280 A$(3)=DTH$(HTD(A$(2))+4)
0290 A$(4)=DTH$(HTD(A$(3))+1)
0300 A$(5)=DTH$(HTD(A$(4))+1)
0310 A$(6)=DTH$(HTD(A$(5))+4)
0320 A$(7)=DTH$(HTD(A$(6))+6)
0330 D$=XFN82012(PEEK$(A$(7),5))
0340 B$=DTH$(HTD(A$(7))+HTD(D$))
0350 PRINT A$(1),'File Name..',XFN113004(PEEK$(A$(1),16))
0360 N$=XFN82012(PEEK$(A$(2),4)) @ CALL COM(N$,R$,I1)
0370 PRINT 'File Type..',N$&'='&R$
0380 N=HTD(D$)/2
0390 IF I1=1 THEN N=CEIL(N-8.5) @ GOTO 410 ELSE 400
0400 IF I1=3 THEN N=CEIL(N-6.5) @ GOTO 410 ELSE N=CEIL(N-2.5)
0410 PRINT 'File length',D$&'=',N,' bytes'
0420 IF PEEK$(A$(3),1)#'0' THEN PRINT 'Flag  .....',PEEK$(A$(3),1)
0430 IF PEEK$(A$(4),1)#'0' THEN PRINT 'Copy Code..',PEEK$(A$(4),1)
0440 C$=XFN82012(PEEK$(A$(5),4))
0450 PRINT 'Creation Time',C$[1,2]&':'&C$[3,4]
0460 C$=XFN82012(PEEK$(A$(6),6))
0470 PRINT 'Creation Date',C$[1,2]&'/'&C$[3,4]&'/'&C$[5,6]
0480 END
0490 SUB COM(Z$,R$,I1)
0500 DIM U$(8)
0510 DATA BASIC,BIN,DATA,KEY,LEX,SDATA,TEXT,FORTH
0520 DATA E214,E204,E0F0,E20C,E208,E0D0,0001,E218
0530 DATA E215,E205,E0F1,E20D,E209,0,E025,E219
0540 DATA E216,E206,0,0,E20A,0,0,E219
0550 DATA E217,E207,0,0,E20B,0,0,E21B
0560 J=1
0570 IF J>4 THEN PRINT 'FILE CODE ERROR',Z$,'NON EXISTENT' @ STOP
0580 FOR I=1 TO 8
```

```
0590 IF I>=2 THEN 650
0600 ON J GOTO 610,620,630,640
0610 RESTORE 520 @ GOTO 650
0620 RESTORE 530 @ GOTO 650
0630 RESTORE 540 @ GOTO 650
0640 RESTORE 550
0650 READ U$(I) @ IF U$(I)=Z$ THEN J1=J @ I1=I @ GOTO 680
0660 IF I=8 THEN J=J+1 @ GOTO 570
0670 NEXT I
0680 RESTORE 510
0690 FOR K=1 TO I1
0700 READ X$
0710 NEXT K
0720 IF J=1 THEN R$=X$
0730 IF J=2 THEN R$=X$&' S'
0740 IF J=3 THEN R$=X$&' P'
0750 IF J=4 THEN R$=X$&' E'
0760 END
```

LE COIN DES LHEX

Comme de coutume, cette rubrique contient la liste des codes hexadécimaux des fichiers Lex parus ce mois-ci.

Rappelons ce qu'est un fichier Lex : c'est un programme pour le HP-71, en assembleur, qui apporte de nouvelles fonctions. Celles-ci sont utilisables directement, ou dans des programmes Basic.

Pour bénéficier de ces nouvelles fonctions, vous n'avez pas besoin de programmer vous-même en assembleur, ni de posséder un module Forth/Assembleur.

Il suffit de recopier le petit programme basic "MAKELEX" ci-dessous, de le lancer et de recopier les codes du fichier Lex désiré. Quand vous avez fini, les nouvelles fonctions sont accessibles, après avoir éteint et rallumé votre HP-71.

Si l'erreur "Erreur de somme" apparaît, vérifiez la ligne que vous avez introduite.

Vous trouverez donc le Lex CHARLEX nécessaire à la rédaction de votre article (voir "Ah ! Vous écrivez !"), ainsi que le Lex de programmation structurée de ce mois-ci.

```
CHARLEX

TRICATLX        OCTCAT  XWORD 092049    TRICAT  XWORD   092050
```

```
 10 CALL MLEX @ SUB MLEX @ SFLAG -1 @ PURGE AH @ INPUT "Nb. d'octets: ";N @ LC OFF
 20 CREATE DATA AH,1,N-4 @ A=HTD(ADDR$("AH")) @ B=A @ GOSUB 130
 30 Q=1 @ X=0 @ INPUT "000: ",P$;A$ @ C$=A$ @ S=0 @ GOSUB 90
 40 Q=2 @ X=1 @ GOSUB 80 @ A$=A$&C$ @ A=A+37 @ N=N*2+37 @ Q=3 @ SFLAG 5 @ FOR X=2 TO N DIV 16-1
 50 GOSUB 80 @ C$=C$[5*FLAG(5)+1] @ POKE DTH$(A),C$ @ A=A+16-5*FLAG(5,0) @ NEXT X @ Q=4
 60 DISP DTH$(X)[3]; @ INPUT ": ",P$[1,MOD(N,16)];C$ @ GOSUB 90
 70 POKE DTH$(A),C$ @ POKE DTH$(B),A$ @ CFLAG -1 @ END
 80 DISP DTH$(X)[3]; @ INPUT ": ",P$;C$
 90 DISP DTH$(X)[3]; @ INPUT " sm ","---";D$
100 M=S @ FOR Z=1 TO LEN(C$) @ M=NUM(C$[Z])+M+1 @ NEXT Z
110 IF D$=DTH$(MOD(M,4096))[3] THEN GOSUB 130 @ S=M @ RETURN
120 DISP "Erreur de somme" @ BEEP @ P$=C$ @ POP @ ON Q GOTO 30,40,50,60
130 P$="----------------" @ RETURN
```

```
CHARLEX          624 octets

      0123456789ABCDEF sm

000: 34841425C4548502 35E
001: 802E000000000000 68D
002: 5E4001EFF0000000 9FD
003: FE0000000800001F D57
004: F31BF961400032BF 0EA
005: 38F14A11DB10AD23 484
006: 07D532BFB8FD7911 837
007: 11AD754D7A101743 BBA
008: 11014D1CB15D0000 F25
009: 71450375FF864834 2A2
00A: 5655581008355654 5F9
00B: 5810070507701724 93F
00C: 7700775070077517 C92
00D: 2077040708364545 FE0
00E: 4A30000A49724000 333
00F: 0808094A2C180814 69C
010: A464242008355455 9F6
011: 581000054C714000 D3C
012: 0C3142404C700832 098
013: 41414A70002078A0 3F0
014: 2F30000000000000 71B
015: 0000000000000000 A2B
016: 0000000000000000 D3B
017: 0000000000000000 04B
018: 0000000000000000 35B
019: 0000000000000000 66B
01A: 0000000000000000 97B
01B: 0000000000000000 C8B
01C: 0000000000000000 F9B
01D: 0000000000000000 2AB
01E: 0000000000000000 5BB
01F: 0000000000000000 8CB
020: 0000000000000000 BDB
021: 000000000000080C F06
022: 1A28080008080A2C 270
023: 180008040E340800 5B9
024: 08001E3018000000 8F3
025: 0000000000000000 C03
026: 0000000000000000 F13
027: 0000000000000000 223
028: 0201000000010200 539
029: 0000000201020000 84E
02A: 0001000100000002 B62
02B: 0102010000000000 E76
02C: 0000000000000000 186
02D: 045E755142400101 4D2
02E: 0101010000000000 7E5
02F: 0000000000000000 AF5
030: 0000070507000000 E18
031: 0000000008344C4 156
032: 44400D7901112D70 4B6
033: 050D750509700000 800
034: 0D70000000384540 B43
035: 4020014E322E3140 E97
036: 084E794142400000 1E7
037: 000000000002E4559 525
038: 3200000000000000 83A
039: 0000000000000026 B52
03A: 5556587008365556 EB1
03B: 5810083645464830 202
03C: 0832414248700024 543
03D: 5655587008345655 8A0
03E: 5810083446454830 BEF
03F: 0C3042414C700024 F44
040: 5556587008355654 2A1
041: 5810083546444830 5F0
042: 0C3142404C700025 946
043: 5455587008355455 CA0
044: 5810083544454830 FEE
045: 0C3140414C700875 350
046: 14141870000A4972 6A1
047: 40000E3159454E30 A01
048: 0C7A0F7949400024 D79
049: 5554587000084A71 0D5
04A: 40000C523A262D10 436
04B: 0424587458400875 78D
04C: 1415187000094A70 ADD
04D: 4000083544454830 E21
04E: 0C3140414C300C74 189
04F: 5655545000054C71 4E0
050: 40000            5D9

CURLEX           90 octets

      0123456789ABCDEF sm

000: 345525C454850202 359
001: 802E000000000000 688
002: 9B0001EFF0000000 9F5
003: FE0000000800001F D4F
004: F31B196140001103 0A5
005: 1B09664FDBD5317F 457
006: 8FC4631D9D75ED1F 823
007: 998F2D231C28FFA0 BD9
008: B11FE74F2AF214F8 F97
009: F4BCE0B64041F998 34C
00A: F23103814908E0A6 6C5
00B: A149171AE0810A6A A54
00C: 1491F998F28F7415 DEA
00D: 182184001        FBC

TRICATLX         962 octets

      0123456789ABCDEF sm

000: 452594341445C485 366
001: 802E000000000000 695
002: 98700C5132300000 9DE
003: F020000000000000 D06
004: 043000FF0053000F 067
005: BF4344534144513B 3DE
006: 452594341445231F 73C
007: F4C128535604C128 AB9
008: F785C180DFD2891E E76
009: 28F13DB0137134CA 207
00A: 131AF231E415A196 589
00B: 68085016115E38FD 90C
00C: 3DE010C1534B4442 C89
00D: 58FBC63117FD8F5F 056
00E: 58A96066528F875C 3EB
00F: 18F75411174D215F 76A
010: 317315341721574A AC1
011: C71781431311C450 E25
012: 38F59B905138F875 1BA
013: C18F36F9040217FD 55C
014: 215F3173153417BC 8D2
015: ECE4D031F38D3939 C88
016: 08F7F5A031C35DE8 039
017: F0837143E1431021 39A
018: 781331001331C3D2 6F5
019: 223182841D015B38 A5B
01A: A2F1851B06B068A2 DEE
01B: 1131528A643852B0 152
01C: 6F6F6C6D7E7E7E7E 53D
01D: 7112203190EADB10 8B4
01E: 98FE6CE020580206 C45
01F: 66F1038A84FCCD22 FF4
020: 2318286111B06B06 355
021: 862603152D52015F 6BD
022: 38A14111BE2CE8AA A6C
023: DB10B5F0137CB137 E0B
024: CC5ADD2CE15F3E65 1E8
025: D91F995F2143DB8F 5A7
026: 7C21056061FE131D 92A
027: 21B5A8F214411C8F CC1
028: C2DE0D78AEF08636 085
029: 068646EA03314C48 403
02A: A7A0873AE8553324 792
02B: 1443111316514071 ACD
02C: E067007280332494 E29
02D: 78703344147E6033 18A
02E: 44D274603364F47A 50B
02F: 5033B454705033C4 86E
030: 5476403325F47C30 BDC
031: 3335447230334554 F26
032: 78203343137E1033 27B
033: 73537410875F38D9 5FB
034: 1FB07C8161308633 975
035: 18A7C185410A076B CFC
036: 938656067438A340 065
037: 0107DB79201B5A8F 3FC
038: 21428FB13B18F238 787
039: 608F695C1AF68D61 B33
03A: 2F010A113CC1198F EC1
03B: 943B1118C2137061 225
03C: 10130D311AD511BC 5A4
03D: E16D16B17D17B861 942
03E: 801831C3D015A38A CC4
03F: 4B1132129CA13212 02A
040: 9E78B72E60B015B3 3CA
```

041: 8A091CE8BBEE1111 77F
042: 37E2137D052E1370 AF6
043: 6137136061360618 E40
044: D18B1CD1CB861801 1E5
045: 631732415AF15FF1 56F
046: 5CF159F16F17F0D5 926
047: 7E861D1862801831 C9B
048: C315A315F315C315 01C
049: 931110713507C213 36B
04A: 407E2137E7CE8B76 713
04B: 0665F623F07DAD21 AAE
04C: 5E38A540E7DB10B2 E54
04D: 01311131B5A8F214 1BC
04E: 6C214416A1401198 520
04F: 7090CC8AC7010A01 8AC
050: 8F943B1118CA123E C46
051: 4D23098FE6CE0D2E 008
052: 68BAD0D5C6C9E652 3CD
053: FDA119068F943B18 772
054: ACB00710A2001102 AD7
055: 118C2D70706EE110 E5A
056: C206118109DB0613 1C9
057: 41331317BC017F27 53C
058: 1C1BF2BF215F10D5 8EB
059: 0FAF50706112E206 C6B
05A: 8B3127ED04A10706 FEE
05B: 134112CA13377800 34B
05C: 757D1197CB04F107 6DB
05D: 112119E206119134 A2A
05E: DB137DA7B5007112 DB9
05F: CA1331307A400711 11D
060: 2C21138B260606F0 48D
061: 7D50706111CA8B0F 821
062: 0101C3D9066F3FDA BC6
063: D7C6C21128FE6CE0 F84
064: DB65FE8627023540 315
065: 2415EF15DF16F17F 6C7
066: 0D5FE861A1862D01 A61
067: 5EB15DB5A015E315 E03
068: D313103134AF4271 16F
069: 5E19E6009E221161 4E9
06A: BF4BF40D56E01038 891
06B: 0C412A1091B5A8F2 C1F
06C: 146165142EA103CC F9D
06D: 4328A8E116414211 301
06E: 98F943B1118CA100 68A
06F: 65BC076A8C110131 A0A
070: 17F1758715017311 D67
071: BCED7AF2129874E0 129
072: 17B1371C9137D54C 4B6
073: 0AF2AF07530112D2 83F
074: 15F38A65117B1197 BBF
075: D101091CF1C5133C F4B
076: 0131CF54D1197920 2C3
077: 24F015B01710D54F 643
078: 05A720487400137C 9A7
079: 9137CF59DAFA078F D73
07A: 223B165EB F78

```
0001 00000                      TITLE   Editeur, utilitaires <xutil.as>
0002 00000
0003 00000              *
0004 00000              * Ce module contient les utilitaires propres à l'éditeur
0005 00000              * et ses fonctions.
0006 00000              * Les utilitaires pris dans JPC Rom sont dans jpcutil.as
0007 00000              *
0008 00000
0009 00000              ********************************************************
0010 00000              * KEYWT
0011 00000              *
0012 00000              * But: attendre une pression de touche de l'utilisateur. Les
0013 00000              *   touches de curseur horizontal sont autorisées pour le
0014 00000              *   déroulement de l'affichage.
0015 00000              * Entree: -
0016 00000              * Sortie:
0017 00000              *   - A(B) := B(B) := code physique (IDS2, p. 12-13)
0018 00000              *   - Cy = 1
0019 00000              * Abime: A-D, D0, D1, P, 32 nibs at SCRTCH (CKSREQ)
0020 00000              * Appelle: SCRLLR, CKSREQ, POPBUF, ATNCLR
0021 00000              * Niveaux: 6
0022 00000              * Detail: Cette routine n'est pas l'équivalent exact
0023 00000              *   de keywt dans KA car celle de KA détecte kOFF et
0024 00000              *   branche directement à l'extinction de KA.
0025 00000              * Historique:
0026 00000              *   86/08/01: ajout de documentation
0027 00000              *   88/05/23: PD/JT pompage dans KA pour T/XEDIT
0028 00000              ********************************************************
0029 00000
0030 00000 8F00000  *KEYWT  GOSBVL =SCRLLR  Voir KEYWAIT$ (JPC No 20), sauf que
0031 00007 5D0              GONC   kwt10        là, eh bien les touches de
0032 0000A 8F00000          GOSBVL =CKSREQ      déroulement del'affichage sont
0033 00011 6EEF             GOTO   =KEYWT       supportées.
0034 00015 8F00000   kwt10  GOSBVL =POPBUF
0035 0001C 8F00000          GOSBVL =ATNCLR  Abime A(A)
0036 00023 D4               A=B    A        Pour compatibilité avec FINDA
0037 00025 01               RTN
0038 00027
0039 00027              ********************************************************
0040 00027              * posmsg, poscmd
0041 00027              *
0042 00027              * But: réaliser POS(MSG$(C), A)
0043 00027              * Entree:
0044 00027              *   - A(B) = caractère à chercher
0045 00027              *   - posmsg seulement :
0046 00027              *       C(A) = numéro de message
0047 00027              * Sortie:
0048 00027              *   - Cy = 0 : non trouvé
0049 00027              *   - Cy = 1 : trouvé
0050 00027              *       C(A) = D(A) = 0.."LEN(MSG$(..)-1)"
```



```
0051 00027              * Abime: A-D, D0, D1, R0, R1, R2 si message insert, OUTBS
0052 00027              * Appelle: CONVUC, FPOLL, D0=AVS, TBMSG$
0053 00027              * Note: la routine "poscmd" est reservee à l'editeur de
0054 00027              *   textes. Le numéro du message à chercher est codé en dur.
0055 00027              * Note: La documentation de TBMSG$ est archi-fausse:
0056 00027              *   Il faudrait lire :
0057 00027              *       ** Exit:
0058 00027              *       **      D0 = ^ start of string
0059 00027              *       **      C(A) = B(A) = ^ end of string (final FF)
0060 00027              *   Merci HP !
0061 00027              * Niveaux: 3
0062 00027              * Historique:
0063 00027              *   88/10/29: PD/JT conception & codage
0064 00027              ********************************************************
0065 00027
0066 00027 8F00000  =poscmd GOSBVL  =CONVUC A(B) := commande en majuscule
0067 0002E 330000           LC(4)   (=id)^(=teVCMD)
0068 00034 108      =posmsg R0=C            R0(3-0) := numéro du message
0069 00037 101              R1=A            R1(B) := caractère à chercher
0070 0003A              *
0071 0003A              * Y-a-t-il un traducteur pour ce message ?
0072 0003A              *
0073 0003A 8F00000          GOSBVL  =FPOLL  abime A(A)-D(A), D0, D1
0074 00041 00               CON(2)  =pTRANS abime A-D, D0, D1, P
0075 00043              *
0076 00043              * R0(3-0) = nouveau numéro de message s'il y a eu traduction
0077 00043              * R1(B) = caractère à chercher
0078 00043              *
0079 00043 8F00000          GOSBVL  =D0=AVS D0 := (AVMEMS)
0080 0004A 118              C=R0            C(A) := numéro de message
0081 0004D 8F00000          GOSBVL  =TBMSG$
0082 00054              *
0083 00054              * D0 = ^ début de la chaine
0084 00054              * C(A) = ^ fin
0085 00054              * R1(B) = caractère à chercher
0086 00054              * Modifie pour tenir compte des erreurs de la documentation
0087 00054              * de TBMSG$.
0088 00054              *
0089 00054 111              A=R1            A(B) := caractère à chercher
0090 00057 D5               B=C     A       B(A) := ^ fin du message
0091 00059 D3               D=0     A
0092 0005B 6010             GOTO    posm20
0093 0005F              *
0094 0005F              * Invariant de boucle :
0095 0005F              *   A(B) = caractère à chercher
0096 0005F              *   D0 = ^ caractère du message à tester
0097 0005F              *   D(A) = indice du caractère à tester (0..LEN-1)
0098 0005F              *   B(A) = ^ terminateur de la table
0099 0005F              *
0100 0005F 14E       posm10 C=DAT0  B
```



```
0101 00062 962              ?A=C    B
0102 00065 51               GOYES   posm50  trouvé !
0103 00067 161              D0=D0+  2
0104 0006A E7               D=D+1   A
0105 0006C 136       posm20 CD0EX
0106 0006F 134              D0=C
0107 00072 881              ?C<B    A       Peut-on continuer ?
0108 00075 AE               GOYES   posm10  oui.
0109 00077 400              RTNC            pas trouvé (Cy = 0)
0110 0007A
0111 0007A DB        posm50 C=D     A       C(A) := index
0112 0007C 01               RTN             trouvé (Cy = 1)
0113 0007E
0114 0007E              ********************************************************
0115 0007E              * PURGE
0116 0007E              *
0117 0007E              * But: détruire le fichier que l'on vient de créer par les
0118 0007E              *   commandes D ou X.
0119 0007E              * Entree:
0120 0007E              *   - EXTFIL = adresse du header du fichier créé
0121 0007E              * Sortie: -
0122 0007E              * Abime: A, B, C(15-5), D, R0-R2, D0, D1, SCRTCH(4-0)
0123 0007E              * Appelle: MVMEM+
0124 0007E              * Niveaux: 5 (MVMEM+ plus un pour sauver C(A))
0125 0007E              * Note: le fichier doit faire exactement 37+4 quartets.
0126 0007E              * Historique:
0127 0007E              *   88/11/11: PD/JT conception & codage
0128 0007E              ********************************************************
0129 0007E
0130 0007E 1B00000  *PURGE  D0=(5)  =EXTFIL Destruction "interne" ?
0131 00085 142              A=DAT0  A
0132 00088 8A8              ?A=0    A       Pas de fichier externe
0133 0008B 00               RTNYES          alors retour
0134 0008D 860              ?ST=0   =sCREAT
0135 00090 00               RTNYES          Le fichier existait déjà
0136 00092
0137 00092 06               RSTK=C          RSTK := C(A)
0138 00094              *
0139 00094              * Purger le fichier
0140 00094              *
0141 00094 347DFFF          LC(5)   -(37+4) taille du fichier créé
0142 00098 EA               A=A-C   A       A(A) := end of file
0143 0009D D5               B=C     A       B(A) := offset (dest - source)
0144 0009F D6               C=A     A       C(A) := end of file
0145 000A1              *
0146 000A1              * Le fichier ayant été créé par nos soins, il l'a été
0147 000A1              * forcément après notre Lex. En conséquence, notre Lex
0148 000A1              * ne peut bouger.
0149 000A1              *
0150 000A1 8F00000          GOSBVL  =MVMEM+
```



```
0151 000A8              *
0152 000A8              * Il ne peut pas y avoir d'erreur (Cy = 1) car
0153 000A8              * - le fichier est supprimé (donc suffisamment de mémoire)
0154 000A8              * - le fichier est créé (donc pas en Rom)
0155 000A8              *
0156 000A8 07               C=RSTK
0157 000AA 01               RTN
0158 000AC
0159 000AC              ********************************************************
0160 000AC              * setf11
0161 000AC              *
0162 000AC              * But: mettre le flag 1 (le vrai flag 1, celui de
0163 000AC              *   l'utilisateur) à l'état indiqué par C(0).
0164 000AC              * Entree:
0165 000AC              *   - C(0) = 0 ou #0
0166 000AC              * Sortie:
0167 000AC              *   - P = 0
0168 000AC              * Abime: A(S), C(S), D0, P
0169 000AC              * Appelle: -
0170 000AC              * Niveaux: 0
0171 000AC              * Historique:
0172 000AC              *   88/11/12: PD/JT conception & codage
0173 000AC              ********************************************************
0174 000AC
0175 000AC 1B00000  =setf11 D0=(5)  =FLGREG D0 := ^ flags utilisateurs
0176 000B3 1524             A=DAT0  S       A(S) := flags 0 à 3
0177 000B7 2F               P=      15
0178 000B9 300              LC(1)   =FL1MSK C(S) := FL1MSK
0179 000BC 20               P=      0
0180 000BE
0181 000BE 90A              ?C=0    P       Mettre le flag à 0 ?
0182 000C1 A0               GOYES   setf20  oui
0183 000C3 0E4E             A=A!C   S       non : mettre le flag à 1
0184 000C7 6A00             GOTO    setf30
0185 000CB BCE       setf20 C=-C-1  S       C(S) := not FL1MSK
0186 000CE 0E46             A=A&C   S
0187 000D2
0188 000D2 1504      setf30 DAT0=A  S       flags 0 à 3 := nouveaux flags
0189 000D6 01               RTN
0190 000D8
0191 000D8              ********************************************************
0192 000D8              * extsek, extsk+
0193 000D8              *
0194 000D8              * But: chercher des lignes dans un fichier externe.
0195 000D8              * Entree:
0196 000D8              *   - A(A) = adresse du fichier externe
0197 000D8              *   - C(A) = numéro de ligne cible
0198 000D8              *   - extsk+ seulement :
0199 000D8              *       - B(A) = numéro d'une ligne <= C dans le fichier
0200 000D8              *       - D0 = adresse de cette ligne
```

```
0201 000D8          • Sortie:
0202 000D8          •   - A(A) = numéro de ligne trouvé (<= C en entree)
0203 000D8          •   - D0 = ^ ligne trouvée (longueur LIF)
0204 000D8          • Abime: A-D, D0
0205 000D8          • Appelle: -
0206 000D8          • Niveaux: 1 (utilisé pour sauvegarde dans RSTK)
0207 000D8          • Historique:
0208 000D8          •   88/11/13: PD/JT extraction de D et généralisation
0209 000D8          •   88/11/27: PD/JT modification de l'interface
0210 000D8          ***************************************************
0211 000D8
0212 000D8 130      *extsek D0=A            D0 := ^ header du fichier externe
0213 000D8 16F              D0=D0+  16      D0 := ^ file type
0214 000DE 16F              D0=D0+  16      D0 := ^ REL(5) FiLeNd
0215 000E1 164              D0=D0+  5       D0 := ^ début des données
0216 000E4 D1               B=0     A
0217 000E6 E5               B=B+1   A       B(A) := 1 (pointe première ligne)
0218 000E8          •
0219 000E8          • A(A) = ^ file header
0220 000E8          • B(A) = numéro d'une ligne référence
0221 000E8          • D0 = ^ début de cette ligne
0222 000E8          • C(A) = numéro de la ligne à trouver
0223 000E8          •
0224 000E8 D7       =extsk* D=C     A       D(A) := numéro de ligne à chercher
0225 000EA 136              CD0EX           C(A) := ^ ligne référence
0226 000ED 06               RSTK=C          RSTK := ^ ligne référence
0227 000EF          •
0228 000EF          • A(A) = ^ file header
0229 000EF          • B(A) = numéro de la ligne référence
0230 000EF          • D(A) = numéro de la ligne à chercher
0231 000EF          • RSTK = ^ ligne référence
0232 000EF          •
0233 000EF 130              D0=A            D0 := ^ header
0234 000F2 D4               A=B     A       A(A) := numéro de la ligne référence
0235 000F4 16F              D0=D0+  16
0236 000F7 16F              D0=D0+  16      D0 := ^ REL(5) FiLeNd
0237 000FA 136              CD0EX           C(A) := ^ REL(5) FiLeNd
0238 000FD 134              D0=C
0239 00100 D5               B=C     A       B(A) := ^ REL(5)
0240 00102 146              C=DAT0  A       C(A) := REL(5)
0241 00105 C1               B=B+C   A       B(A) := ^ FiLeNd
0242 00107          •
0243 00107          • A(A) = numéro de la ligne référence
0244 00107          • B(A) = ^ FiLeNd
0245 00107          • D(A) = numéro de la ligne à chercher
0246 00107          • RSTK = ^ ligne référence
0247 00107          •
0248 00107 07               C=RSTK
0249 00109 134              D0=C            D0 := ^ ligne référence
0250 0010C
```



```
0251 0010C D6               C=A     A       C(A) := numéro ligne référence
0252 0010E
0253 0010E          •
0254 0010E          • pour i := (ligne référence) jusque (ligne à chercher)
0255 0010E          •   faire
0256 0010E          •     si D0 >= FiLeNd ou *D0 = #FFFF
0257 0010E          •       alors sortir de la boucle "pour"
0258 0010E          •     fin si
0259 0010E          •     passer à la ligne suivante
0260 0010E          • fin pour
0261 0010E          • renvoyer i
0262 0010E          •
0263 0010E
0264 0010E          •
0265 0010E          • Affectation des registres pendant la boucle :
0266 0010E          • - B(A) = ^ FiLeNd
0267 0010E          • - C(A) = numéro de la ligne courante (i)
0268 0010E          • - D(A) = numéro de la ligne à chercher
0269 0010E          • - D0 = ^ ligne courante (longueur LIF)
0270 0010E          •
0271 0010E 6540             GOTO    ske200  aller directement au test sur i
0272 00112
0273 00112 06       ske100 RSTK=C           RSTK := i
0274 00114 136              CD0EX           C(A) := ^ ligne courante
0275 00117 134              D0=C            D0 := ^ ligne courante
0276 0011A 88D              ?C>=B   A       at EOF ?
0277 0011D F3               GOYES   ske300  oui : D0 = ^ fin du fichier
0278 0011F
0279 0011F 15A3             A=DAT0  4
0280 00123          • SWPBYT (#17A24) recopié pour éviter un GOSBVL
0281 00123 D6               C=A     A
0282 00125 F0               ASL     A
0283 00127 F0               ASL     A
0284 00129 F6               CSR     A
0285 0012B F6               CSR     A       C(4) := 0 (entre autres)
0286 0012D AEA              A=C     B       A(A) := longueur LIF
0287 00130          • fin du pompage
0288 00130 23               P=      3
0289 00132 A96              C=A     WP      on sait que C(4) = 0
0290 00135 DA               A=C     A
0291 00137 B14              A=A+1   WP      Cy = 1 si C(A) = FFFF
0292 0013A 20               P=      0
0293 0013C 4F1              GOC     ske300  D0 = ^ FFFF
0294 0013F          • conversion en nombre de quartets à sauter
0295 0013F 81C              ASRB
0296 00142 E4               A=A+1   A
0297 00144 C4               A=A+A   A
0298 00146 C4               A=A+A   A       A(A) := nb de quartets à sauter
0299 00148
0300 00148 136              CD0EX
```



```
0301 0014B C2               C=C+A   A
0302 0014D 134              D0=C            D0 := ^ ligne suivante
0303 00150
0304 00150          •
0305 00150          • D0 = ^ ligne suivante
0306 00150          • RSTK = numéro de la ligne précédente
0307 00150          • B(A) = ^ FiLeNd
0308 00150          • D(A) = numéro de ligne à chercher
0309 00150          •
0310 00150 07               C=RSTK
0311 00152 E6               C=C+1   A       i++
0312 00154
0313 00154 8BF      ske200 ?C>=D    A       i >= ligne à chercher ?
0314 00157 70               GOYES   ske310  oui : on arrête
0315 00159 58B              GONC    ske100
0316 0015C
0317 0015C          •
0318 0015C          • Sortie sur EOF
0319 0015C          • RSTK  = i
0320 0015C          • D0 = ^ EOF
0321 0015C          •
0322 0015C 07       ske300 C=RSTK           C(A) := i
0323 0015E DA       ske310 A=C      A       A(A) := i
0324 00160 01               RTN
0325 00162
0326 00162          ***************************************************
0327 00162          • YNQ
0328 00162          •
0329 00162          • But: prendre un code physique de touche, et tester si la
0330 00162          •   touche est une de celles autorisées par le message
0331 00162          •   *teKEYS, ce qui permet de traduire les reponses
0332 00162          •   autorisées à une demande "Yes/No/Quit".
0333 00162          • Entree:
0334 00162          •   - A(B) = code physique de la touche (renvoyé par KEYWT)
0335 00162          • Sortie:
0336 00162          •   - C(0) = 0 : non reconnu
0337 00162          •            1 : [Y]
0338 00162          •            2 : [N]
0339 00162          •            3 : [Q]
0340 00162          • Abime: A-C, R0-R1, S0-S2, D0, D1, OUTBS
0341 00162          • Appelle: KEYNAM, posmsg
0342 00162          • Niveaux: 4 (posmsg)
0343 00162          • Historique:
0344 00162          •   88/11/13: PD/JT conception & codage
0345 00162          ***************************************************
0346 00162
0347 00162 8F00000  =YNQ    GOSBVL  =KEYNAM Conversion en nom de touche
0348 00169          •
0349 00169          • A(WP) = ASCII for keycode
0350 00169          • P = word thru pointer length of text
```



```
0351 00169          •
0352 00169 881              ?P#     1       # 1 caractère ?
0353 0016C 72               GOYES   YNQ500  non reconnu
0354 0016E 20               P=      0
0355 00170 8F00000          GOSBVL  =CONVUC Conversion en majuscules
0356 00177 330000           LC(4)   (=id)^(=teKEYS)
0357 0017D 738E             GOSUB   =posmsg
0358 00181 511              GONC    YNQ500  non trouvé
0359 00184 D0               A=0     A
0360 00186 E4               A=A+1   A
0361 00188 E4               A=A+1   A       A(A) := 2
0362 0018A 8B2              ?C>A    A
0363 0018D 60               GOYES   YNQ500  Non reconnu ! Message erroné !
0364 0018F E6               C=C+1   A       renvoyer numéro dans (1..3)
0365 00191 01               RTN
0366 00193
0367 00193 D2       YNQ500 C=0      A       C(0) := 0
0368 00195 20               P=      0
0369 00197 01               RTN
0370 00199
0371 00199          ***************************************************
0372 00199          • CHKTXT
0373 00199          •
0374 00199          • But: vérifier que le fichier pointé par D1 est du type
0375 00199          •   TEXT.
0376 00199          • Entrée:
0377 00199          •   - D1 = ^ file header
0378 00199          • Sortie:
0379 00199          •   - Cy = 0 : pas d'erreur
0380 00199          •        D1 inchangé
0381 00199          •   - Cy = 1 : ce n'est pas un TEXT
0382 00199          •        C(3-0) = eFTYPE
0383 00199          • Abime: A(A), A(S), C
0384 00199          • Appelle: -
0385 00199          • Niveaux: 0
0386 00199          • Historique:
0387 00199          •   88/11/27: PD/JT conception & codage
0388 00199          •   89/06/11: PD/JT reconception & recodage
0389 00199          ***************************************************
0390 00199
0391 00199 17F      =CHKTXT D1=D1+  16      D1 := ^ file type
0392 0019C D0               A=0     A
0393 0019E 1583             A=DAT1  4       A(A) := file type
0394 001A2 D2               C=0     A
0395 001A4 E6               C=C+1   A       C(A) := FTEXT
0396 001A6 8A2              ?A=C    A
0397 001A9 A0               GOYES   CKTXT1  Ok
0398 001AB          •
0399 001AB          • Invalid File Type
0400 001AB          •
```

```
0001 001AB 330000          LC(4)   =eFTYPE
0002 001B1 02              RTNSC           Cy = 1
0003 001B3
0004 001B3 1CF      CKTXT1 D1=D1-  16      remet D1 à l'original
0005 001B6 03              RTNCC           Cy = 0
0006 001B8
0007 001B8          *******************************************************
0008 001B8          * FILE1
0009 001B8          *
0010 001B8          * But: analyser une description de fichier et mémoriser
0011 001B8          *   certaines informations pour FILE2 dans STMTR0/R1
0012 001B8          * Entree:
0013 001B8          *   - D0 = ^ chaine tokenisée
0014 001B8          * Sortie:
0015 001B8          *   - STMTR0 = D(W) à la sortie de FSPECx
0016 001B8          *   - STMTR1 = nom du fichier
0017 001B8          *   - D0 = réactualisé
0018 001B8          * Abime: A-D, R0-R3, D0, D1, STMTD0, STMTR1, FUNCxx
0019 001B8          * Appelle: FSPECx
0020 001B8          * Niveaux: 6 (FSPECx)
0021 001B8          * Note:
0022 001B8          *   - en cas d'erreur, sortie par BSERR
0023 001B8          * Historique:
0024 001B8          *   89/06/17: PD/JT mise en commun
0025 001B8          *******************************************************
0026 001B8
0027 001B8          =FILE1
0028 001B8          *
0029 001B8          * Sauvegarde d'un niveau de RSTK
0030 001B8          *
0031 001B8 1F00000          D1=(5)  =S-R0-0
0032 001BF 07               C=RSTK
0033 001C1 145              DAT1=C  A       S-R0-0 := sauvegarde d'un niveau
0034 001C4          *
0035 001C4          * Evaluation de la spécification
0036 001C4          *
0037 001C4 8F00000          GOSBVL  =FSPECx 6 niveaux pour lui tout seul !
0038 001CB          *
0039 001CB          * Restauration d'un niveau de RSTK
0040 001CB          *
0041 001CB 1F00000          D1=(5)  =S-R0-0
0042 001D2 147              C=DAT1  A       C(A) := niveau sauvé au dessus
0043 001D5 06               RSTK=C
0044 001D7          *
0045 001D7          * L'évaluation s'est-elle bien passée ?
0046 001D7          *
0047 001D7 442              GOC     bserr   Cy = 1 <=> erreur dans l'analyse
0048 001DA 978              ?A=0    W       A = 0 <=> pas de nom de fichier
0049 001DD 91               GOYES   fspece
0050 001DF          *
```



```
0051 001DF          * Mémoriser les informations renvoyées par FSPECx
0052 001DF          *
0053 001DF 1F00000          D1=(5)  =STMTR0
0054 001E6 AFB              C=D     W       Sauvegarde de D(W) pour CRETF+
0055 001E9 1557             DAT1=C  W
0056 001ED 17F              D1=D1+  16      D1=(5) STMTR1
0057 001F0 1517             DAT1=A  W       Sauvegarde du nom du fichier
0058 001F4 01               RTN
0059 001F6
0060 001F6 D2       fspece  C=0     A
0061 001F8 3100             LC(2)   =eFSPEC
0062 001FC 8D00000  bserr   GOVLNG  =BSERR
0063 00203
0064 00203          *******************************************************
0065 00203          * FILE2
0066 00203          *
0067 00203          * But: à partir des descriptions laissées par FILE1 dans
0068 00203          *   STMTR0/R1, chercher et créer éventuellement le fichier.
0069 00203          * Entree:
0070 00203          *   - STMTR0 = D(W) à la sortie de FSPECx
0071 00203          *   - STMTR1 = nom du fichier
0072 00203          *   - S1 = 1 si le fichier doit être copié en mémoire
0073 00203          *   - S2 = 1 si le fichier doit être créé
0074 00203          * Sortie:
0075 00203          *   - C(A) = adresse du fichier
0076 00203          * Abime: A-D, D0, D1, R0-R3, S0-S9, S12, STMTR0
0077 00203          * Appelle: SALLOC, SVINF+, SVINFO, PSHUPD, COPYu, POPUPD,
0078 00203          *   FINDF, CRETF+
0079 00203          * Niveaux: 7 (S1=1/COPYu), 6 (S2=1/CRETF+) ou 3 (Sx=0/FINDF)
0080 00203          * Historique:
0081 00203          *   89/06/17: PD/JT mise en commun
0082 00203          *   89/06/17: PD/JT utilisation des flags S1 et S2
0083 00203          *******************************************************
0084 00203
0085 00203          =FILE2
0086 00203          *
0087 00203          * Analyse du résultat renvoyé par FSPECx
0088 00203          *
0089 00203 1F00000          D1=(5)  =STMTR0
0090 0020A 1577             C=DAT1  W
0091 0020E AF7              D=C     W       Restaurer pour CRETF+
0092 00211 17F              D1=D1+  16      D1=(5) =STMTR1
0093 00214 1537             A=DAT1  W       Comme en sortie de FSPECx
0094 00218 AC2              C=0     S
0095 0021B 943              ?C=D    S
0096 0021E C7               GOYES   FL2-20  :MAIN
0097 00220 846              C=C+1   S
0098 00223 943              ?C=D    S
0099 00226 47               GOYES   FL2-20  :PORT
0100 00228 AC2              C=0     S
```



```
0101 00228 A4E              C=C-1   S
0102 0022E 943              ?C=D    S
0103 00231 96               GOYES   FL2-20  no device specified
0104 00233          *
0105 00233          * C'est un spécificateur externe
0106 00233          *
0107 00233 871              ?ST=1   1       fichier doit être copié en mémoire ?
0108 00236 A0               GOYES   FL2-10  oui
0109 00238
0110 00238 D2               C=0     A
0111 0023A 3100             LC(2)   =eFSPEC
0112 0023E 02               RTNSC           Cy = 1 : erreur
0113 00240
0114 00240          *
0115 00240          * Recopie en mémoire du fichier
0116 00240          *
0117 00240 101      FL2-10  R1=A
0118 00243 3123             LC(2)   50
0119 00247 8F00000          GOSBVL  =SALLOC
0120 0024E          * Abime : A, B(A), C(A), D1
0121 0024E 111              A=R1
0122 00251 843              ST=0    3       Source
0123 00254 8F00000          GOSBVL  =SVINF+
0124 0025B          * Abime : A, C, D, R0, D1, S3, S4
0125 0025B AF0              A=0     W
0126 0025E D3               D=0     A
0127 00260 853              ST=1    3
0128 00263 8F00000          GOSBVL  =SVINFO
0129 0026A          *
0130 0026A          * Le fichier est copié derrière le dernier, mais une
0131 0026A          * reconfiguration peut intervenir, la mémoire serait alors
0132 0026A          * déplacée. Il faut donc utiliser MGOSUB, mais également
0133 0026A          * sauvegarder l'adresse de l'appelant
0134 0026A          *
0135 0026A 07               C=RSTK          C(A) := adresse de l'appelant
0136 0026C DA               A=C     A
0137 0026E 8F00000          GOSBVL  =PSHUPD
0138 00275
0139 00275 8F00000          GOSBVL  =MGOSUB
0140 0027C 00000            CON(5)  =COPYu  copie
0141 00281          * Abime : A-D, D0, D1, R0-R2, S0-S9, S12, STMTR0
0142 00281 441              GOC     Bserr   Erreur lors de la copie
0143 00284
0144 00284 8F00000          GOSBVL  =POPUPD
0145 0028B DB               C=D     A       C(A) := adresse dépilée
0146 0028D 06               RSTK=C
0147 0028F
0148 0028F 119              C=R1            C(A) = ^ début du fichier
0149 00292 6680             GOTO    FL2-40
0150 00296
```



```
0151 00296 656F     Bserr  GOTO    bserr   Rallonge
0152 0029A
0153 0029A          *
0154 0029A          * C'est un spécificateur interne
0155 0029A          *
0156 0029A 1F00000  FL2-20 D1=(5)  =STMTR0
0157 002A1 AFB              C=D     W
0158 002A4 1557             DAT1=C  W
0159 002A8 17F              D1=D1+  16
0160 002AB 1517             DAT1=A  W
0161 002AF 8F00000          GOSBVL  =FINDF
0162 002B6          * Abime : A-D, S6, S8, R1-R3
0163 002B6 137              CD1EX
0164 002B9 5F5              GONC    FL2-40  C(A) := ^ début du fichier
0165 002BC          *
0166 002BC          * Le fichier n'existe pas
0167 002BC          *
0168 002BC 872              ?ST=1   2       Faut-il le créer ?
0169 002BF 70               GOYES   FL2-30  Oui !
0170 002C1          *
0171 002C1          * Le fichier n'existe pas et il ne faut pas le créer.
0172 002C1          * Renvoyer un erreur
0173 002C1          *
0174 002C1 137              CD1EX           C(A) := numéro d'erreur
0175 002C4 02               RTNSC           Cy = 1 => il y a eû erreur
0176 002C6          *
0177 002C6          * Il faut le créer
0178 002C6          *
0179 002C6 1F00000  FL2-30 D1=(5)  =STMTR0
0180 002CD 1577             C=DAT1  W
0181 002D1 AF7              D=C     W
0182 002D4 D2               C=0     A
0183 002D6 3192             LC(2)   37+4
0184 002DA 8F00000          GOSBVL  =CRETF+ Théoriq., le Lex ne doit pas bouger
0185 002E1          * Abime : A-D, D0, D1, R0-R1, S0-S7
0186 002E1 448              GOC     Bserr
0187 002E4 119              C=R1
0188 002E7 1F00000          D1=(5)  =STMTR1
0189 002EE 1537             A=DAT1  W
0190 002F2 135              D1=C
0191 002F5 1517             DAT1=A  W       Nom du fichier
0192 002F9 17F              D1=D1+  16      D1 := ^ file type
0193 002FC 35100004         LCHEX   400001  text + copy code
0194 00304 15D5             DAT1=C  6
0195 00308 17F              D1=D1+  16
0196 0030B 174              D1=D1+  5
0197 0030E D2               C=0     A       EOF mark
0198 00310 CE               C=C-1   A
0199 00312 15D3             DAT1=C  4
0200 00316 119              C=R1            C(A) = ^ début du fichier
```

```
0601 00319          •
0602 00319          • Le fichier est trouvé. On a son adresse dans C(A)
0603 00319          •
0604 00319 03        FL2-40 RTNCC            Cy = 0 : pas d'erreur
0605 0031B
0606 0031B          ************************************************************
0607 0031B          • CHKWRT
0608 0031B          •
0609 0031B          • But: vérifier si le fichier spécifié est modifiable ou non
0610 0031B          • Entree:
0611 0031B          •   - C(A) = adresse (du header) du fichier à tester
0612 0031B          • Sortie:
0613 0031B          •   - C(S) = 0 : fichier modifiable
0614 0031B          •   - C(S) = 1 : fichier en Rom (eFACCS)
0615 0031B          •   - C(S) = 2 : fichier SECURE (eFPROT)
0616 0031B          •   - C(S) = 3 : fichier ouvert (eFOPEN)
0617 0031B          •   - C(A) = adresse du fichier
0618 0031B          • Abime: A, B(A), C, D1
0619 0031B          • Appelle: ISRAM?, GETPR1, CHKOPN
0620 0031B          • Niveaux: 2
0621 0031B          • Historique:
0622 0031B          •   89/06/17: PD/JT conception & codage
0623 0031B          ************************************************************
0624 0031B
0625 0031B          =CHKWRT
0626 0031B          •
0627 0031B          • Vérifier si le fichier est en Rom
0628 0031B          •
0629 0031B 8F00000          GOSBVL  =ISRAM? Abime A, B(A), C, D1
0630 00322 D9               C=B     A       C(A) := ^ header sauvé par ISRAM?
0631 00324 512              GONC    CKWrom
0632 00327          •
0633 00327          • Vérifier si le fichier est protégé (SECURE)
0634 00327          •
0635 00327 135              D1=C            D1 := ^ file header
0636 0032A 8F00000          GOSBVL  =GETPR1
0637 00331 D9               C=B     A       C(A) := B(A) non modifié par GETPR1
0638 00333          • SB = 1 si le fichier est SECURE
0639 00333 832              ?SB=0           Fichier non sécurisé
0640 00336 50               GOYES   CKW010
0641 00338 580              GONC    CKWsec  B.E.T.
0642 0033B          CKW010
0643 0033B          •
0644 0033B          • Vérifier si le fichier est déjà ouvert
0645 0033B          •
0646 0033B 7730             GOSUB   CHKOPN
0647 0033F 580              GONC    CKWok   Ok, il n'est pas déjà ouvert
0648 00342
0649 00342 0C       CKWopn P=P+1
0650 00344 0C       CKWsec P=P+1
```



```
0651 00346 0C       CKWrom P=P+1
0652 00348 80CF     CKWok  C=P      15
0653 0034C 20              P=      0
0654 0034E 01              RTN
0655 00350
0656 00350          ************************************************************
0657 00350          • WRITE?
0658 00350          •
0659 00350          • But: renvoyer un numéro d'erreur connaissant le code
0660 00350          •   délivré par CHKWRT
0661 00350          • Entree:
0662 00350          •   - C(S) = 0 : fichier modifiable
0663 00350          •   - C(S) = 1 : fichier en Rom (eFACCS)
0664 00350          •   - C(S) = 2 : fichier SECURE (eFPROT)
0665 00350          •   - C(S) = 3 : fichier ouvert (eFOPEN)
0666 00350          • Sortie:
0667 00350          •   - Cy = 1 : pas d'erreur
0668 00350          •   - Cy = 0 : erreur
0669 00350          •       C(A) = numéro d'erreur
0670 00350          • Abime: C(S), C(A)
0671 00350          • Appelle: -
0672 00350          • Niveaux: 0
0673 00350          • Historique:
0674 00350          •   89/06/17: PD/JT conception & codage
0675 00350          ************************************************************
0676 00350
0677 00350 A4E      =WRITE? C=C-1   S
0678 00353 400              RTNC            pas d'erreur
0679 00356
0680 00356 D2               C=0     A
0681 00358 A4E              C=C-1   S
0682 0035B 441              GOC     WRTrom
0683 0035E A4E              C=C-1   S
0684 00361 480              GOC     WRTsec
0685 00364 3100     WRTopn LC(2)    =eFOPEN
0686 00368 03               RTNCC
0687 0036A 3100     WRTsec LC(2)    =eFPROT
0688 0036E 03               RTNCC
0689 00370 3100     WRTrom LC(2)    =eFACCS
0690 00374 03               RTNCC
0691 00376
0692 00376          ************************************************************
0693 00376          • CHKOPN
0694 00376          •
0695 00376          • But: vérifier si le fichier spécifié est déjà ouvert
0696 00376          •   ou non.
0697 00376          • Entree:
0698 00376          •   - C(A) = adresse du fichier à tester
0699 00376          • Sortie:
0700 00376          •   -C(A) = adresse du fichier
```



```
0701 00376          •   - Cy = 0 : le fichier n'est pas déjà ouvert
0702 00376          •   - Cy = 1 : le fichier est déjà ouvert
0703 00376          • Abime: A, B(A), C, D1
0704 00376          • Appelle: I/OFND
0705 00376          • Niveaux: 1
0706 00376          • Historique:
0707 00376          •   89/06/17: PD/JT mise en commun
0708 00376          ************************************************************
0709 00376
0710 00376 D5       CHKOPN B=C      A       Sauver l'adresse dans B(A)
0711 00378          •
0712 00378          • Chercherle buffer contenant le FIB
0713 00378          •
0714 00378 32000            LC(3)   =bFIB
0715 0037D 8F00000          GOSBVL  =I/OFND Abime A, C(A), C(S), D1
0716 00384          •
0717 00384          • Le FIB existe forcément
0718 00384          • D1 = ^ past buffer header
0719 00384          •
0720 00384
0721 00384          •
0722 00384          • Préparation de la valeur à tester.
0723 00384          • Les quartets 6-5 contiennent F0
0724 00384          • Les quartets 4-0 contiennent l'adress du début des données
0725 00384          •
0726 00384 36520000         LCHEX   F000025 C(6-5) := F0 (fichier en Ram)
     00388 F
0727 0038D C9               C=C*B   A       C(4-0) := adresse des données
0728 0038F          •
0729 0038F          • Code extrait de OPENF- (#11B54 dans Rom du HP-71).
0730 0038F          • Merci HP !
0731 0038F          •
0732 0038F 26               P=      6
0733 00391 148      CHK010 A=DAT1   B
0734 00394 968              ?A=0    B
0735 00397 32               GOYES   CHKO90  On n'a pas trouvé ! C'est donc ok
0736 00399 170              D1=D1+  (=oFBEGb)-(=oFILfb)
0737 0039C 170              D1=D1+  (=oDBEGb)-(=oFBEGb)
0738 0039F 1586             A=DAT1  7
0739 003A3 912              ?A=C    WP
0740 003A6 E0               GOYES   CHKO80  On a trouvé ! Beeep ! "File Open"
0741 003A8 170              D1=D1+  (=oRECLb)-(=oDBEGb)
0742 003AB 170              D1=D1+  (=oRLENb)-(=oRECLb)
0743 003AE 170              D1=D1+  (=IFIB)-(=oRLENb)
0744 003B1 5FD              GONC    CHK010  B.E.T.
0745 003B4
0746 003B4 20       CHK080 P=       0
0747 003B6          •
0748 003B6          • B(A) = adresse du fichier
0749 003B6          •
```



```
0750 003B6 D9               C=B     A       On remet les choses en place
0751 003B8 02               RTNSC           Cy = 1
0752 003BA
0753 003BA 20       CHKO90 P=       0
0754 003BC          •
0755 003BC          • B(A) = adresse du fichier
0756 003BC          •
0757 003BC D9               C=B     A       On remet les choses en place
0758 003BE 03               RTNCC
0759 003C0
0760 003C0                  END
```

```
*ATNCLR       Extrn Ukn       -  0035
*BSERR        Extrn Ukn       -  0462
 Bserr        00296 Rel  0551 -  0542 0586
*CHKTXT       00199 Rel  0391 -
*CHKWRT       0031B Rel  0625 -
*CKSREQ       Extrn Ukn       -  0032
*CONVUC       Extrn Ukn       -  0066 0355
*COPYu        Extrn Ukn       -  0540
*CRETF*       Extrn Ukn       -  0584
 CHK010       00391 Rel  0733 -  0744
 CHK030       00384 Rel  0746 -  0740
 CHK090       0038A Rel  0753 -  0735
 CHKOPN       00376 Rel  0710 -  0646
 CKTXT1       00183 Rel  0404 -  0397
 CKW010       00338 Rel  0642 -  0640
 CKWok        00348 Rel  0652 -  0647
 CKWopn       00342 Rel  0649 -
 CKWrom       00346 Rel  0651 -  0631
 CKWsec       00344 Rel  0650 -  0641
*D0*AVS       Extrn Ukn       -  0079
*EXTFIL       Extrn Ukn       -  0130
*FILE1        00188 Rel  0427 -
*FILE2        00203 Rel  0485 -
*FINDF        Extrn Ukn       -  0561
*FL1MSK       Extrn Ukn       -  0178
*FLGREG       Extrn Ukn       -  0175
*FPOLL        Extrn Ukn       -  0073
*FSPECx       Extrn Ukn       -  0437
 FL2-10       00240 Rel  0517 -  0508
 FL2-20       0029A Rel  0556 -  0496 0499 0503
 FL2-30       002C6 Rel  0579 -  0569
 FL2-40       00319 Rel  0604 -  0549 0564
 FiLeNd       003C0 Rel  0760 -
*GETPR1       Extrn Ukn       -  0636
*I/OFND       Extrn Ukn       -  0715
*ISRAM?       Extrn Ukn       -  0629
*KEYNAM       Extrn Ukn       -  0347
*KEYWT        00000 Rel  0030 -  0033
*MGOSUB       Extrn Ukn       -  0539
*MVMEM*       Extrn Ukn       -  0150
*POPBUF       Extrn Ukn       -  0034
*POPUPD       Extrn Ukn       -  0544
*PSHUPD       Extrn Ukn       -  0537
*PURGE        0007E Rel  0130 -
*S-R0-0       Extrn Ukn       -  0431 0441
*SALLOC       Extrn Ukn       -  0519
*SCRLLR       Extrn Ukn       -  0030
*STMTR0       Extrn Ukn       -  0453 0489 0556 0579
*STMTR1       Extrn Ukn       -  0588
*SVINF*       Extrn Ukn       -  0523
```



```
*SVINFO       Extrn Ukn       -  0528
*TBMSG$       Extrn Ukn       -  0081
*WRITE?       00350 Rel  0677 -
 WRTopn       00354 Rel  0685 -
 WRTrom       00370 Rel  0689 -  0682
 WRTsec       0036A Rel  0687 -  0684
*YNQ          00162 Rel  0347 -
 YNQ500       00193 Rel  0367 -  0353 0358 0363
*bFIB         Extrn Ukn       -  0714
 bserr        001FC Rel  0462 -  0447 0551
*eFACCS       Extrn Ukn       -  0689
*eFOPEN       Extrn Ukn       -  0685
*eFPROT       Extrn Ukn       -  0687
*eFSPEC       Extrn Ukn       -  0461 0511
*eFTYPE       Extrn Ukn       -  0401
*extsek       000D8 Rel  0212 -
*extsk*       000E8 Rel  0224 -
 fspece       001F6 Rel  0460 -  0449
*id           Extrn Ukn       -  0067 0356
 kwt10        00015 Rel  0034 -  0031
*lFIB         Extrn Ukn       -  0743
*oDBEGb       Extrn Ukn       -  0737 0741
*oFBEGb       Extrn Ukn       -  0736 0737
*oFILEb       Extrn Ukn       -  0736
*oRECLb       Extrn Ukn       -  0741 0742
*oRLENb       Extrn Ukn       -  0742 0743
*pTRANS       Extrn Ukn       -  0074
*poscmd       00027 Rel  0066 -
*posmsg       00034 Rel  0068 -  0357
 posm10       0005F Rel  0100 -  0108
 posm20       0006C Rel  0105 -  0092
 posm50       0007A Rel  0111 -  0102
*sCREAT       Extrn Ukn       -  0134
*setf11       000AC Rel  0175 -
 setf20       000CB Rel  0185 -  0182
 setf30       000D2 Rel  0188 -  0184
 ske100       00112 Rel  0273 -  0315
 ske200       00154 Rel  0313 -  0271
 ske300       0015C Rel  0322 -  0277 0293
 ske310       0015E Rel  0323 -  0314
*teKEYS       Extrn Ukn       -  0356
*teVCMD       Extrn Ukn       -  0067
```

Source : xutil.as

Object : obj/xutil.ao

Listing : list/xutil.al

Date : Sat Aug 12 17:41:27 1989

Errors : 000

Areuh Assembler/Linker V2.4, (c) P. David & J. Taillandier 1986 Paris, France

```
0001 00000                      TITLE   Utilitaires empruntés à JPC Rom <jpcutil.as>
0002 00000
0003 00000              ********************************************************
0004 00000              * Num2D1
0005 00000              *
0006 00000              * But: traduire un nombre en décimal et transférer sa
0007 00000              *   représentation alphanumérique dans le buffer pointé par
0008 00000              *   D1.
0009 00000              * Entree:
0010 00000              *   - Nombre en hexa dans A(A)
0011 00000              *   - D1 pointe sur le premier caractère à renvoyer
0012 00000              *   - P = 0
0013 00000              * Sortie:
0014 00000              *   - D1 pointe après le dernier caractère envoyé
0015 00000              *   - mode = hex
0016 00000              *   - P = 0
0017 00000              *   - Cy = 1
0018 00000              *   - C(S) = nombre de chiffres affichés - 1
0019 00000              * Abime: A(A), B(A), C(A), D1
0020 00000              * Appelle: HEXDEC
0021 00000              * Niveaux: 1
0022 00000              * Historique:
0023 00000              *   86/07/31: supprimer la boucle infinie quand A(A) = 0
0024 00000              *   86/08/01: ajout de documentation
0025 00000              *   88/05/23: PD/JT pompage dans KA pour T/XEDIT
0026 00000              ********************************************************
0027 00000
0028 00000 8A8      =Num2D1 ?A=0    A       Nb dans A(A), P=0
0029 00003 41               GOYES   num15
0030 00005 8F00000          GOSBVL  =HEXDEC Résultat dans A,B,C
0031 0000C 04               SETHEX          Car mode = DEC après HEXDEC
0032 0000E              * Boucle de comptage du nombre de chiffres
0033 0000E 27               P=      7       Max=1048575
0034 00010 0D       num10   P=P-1
0035 00012 908              ?A=0    P
0036 00015 BF               GOYES   num10
0037 00017              * num15: P = nombre de chiffres à envoyer - 1
0038 00017 80CF     num15   C=P     15      C(S) := nb chiffres à afficher - 1
0039 00018              * préparation de A(W) pour la boucle d'envoi
0040 0001B 814      num20   ASRC
0041 0001E 0D               P=P-1
0042 00020 88F              ?P#     #F
0043 00023 8F               GOYES   num20
0044 00025 20               P=      0
0045 00027              * Fin de la boucle, A(W) := C1 C2 ... Cn 0 ... 0
0046 00027 3103             LCASC   '0'     Pour ajouter à tous les chiffres Ci
0047 00028 80DF             P=C     15      P := nombre de chiffres à envoyer
0048 0002F              * Boucle d'envoi des chiffres
0049 0002F 810      num30   ASLC            A(W) := Ci+1 ... Cn 0 ... 0 Ci
0050 00032 A6A              A=A+C   B       A(B) := valeur Ascii du chiffre
```



```
0051 00035 149              DAT1=A B        envoi dans le buffer
0052 00038 171              D1=D1+ 2
0053 0003B D0               A=0     A       Pour le chiffre suivant
0054 0003D 0D               P=P-1
0055 0003F 5FE              GONC    num30
0056 00042 20               P=      0       En sortie: A(W)=0, P=0, Cy=1
0057 00044 01               RTN
0058 00046
0059 00046              ********************************************************
0060 00046              * fkey
0061 00046              *
0062 00046              * But: mettre une touche en premier dans le key buffer.
0063 00046              * Entree:
0064 00046              *   - R3(B) = code physique de la touche (cf IDS2, p 12-13)
0065 00046              * Abime: A(A), B(A), C(A), D0
0066 00046              * Niveaux: 0
0067 00046              * Detail: INTON et INTOFF sont utilisés ici
0068 00046              * Historique:
0069 00046              *   86/08/01: ajout de documentation
0070 00046              *   88/05/23: PD/JT pompage dans KA pour T/XEDIT
0071 00046              ********************************************************
0072 00046
0073 00046 D2       =fkey   C=0     A       Entrée: R3(B)=physical key-code
0074 00048 31C1             LC(2)   28      Nb de quartets du KEYBUF deplacer
0075 0004C D5               B=C     A       > Nombre de quartets à déplacer
0076 0004E 3400000          LC(5)   =KEYBUF
0077 00055 DA               A=C     A       > Adresse origine
0078 00057 E6               C=C+1   A
0079 00059 E6               C=C+1   A       > Adresse destination
0080 0005B 808F             INTOFF
0081 0005F 8F00000          GOSBVL  =MOVE=M Déplacer 14 entrées du KEYBUF
0082 00066 11B              C=R3            La touche à insérer
0083 00069 1B00000          D0=(5) =KEYBUF
0084 00070 14C              DAT0=C B        Insertion effectivement réalisée
0085 00073 180              D0=D0- 1        D0=(5) KEYPTR
0086 00076 1564             C=DAT0 S
0087 0007A B46              C=C+1  S        Une touche de plus si possible
0088 0007D 460              GOC     fkey10
0089 00080 1544             DAT0=C S
0090 00084 8080     fkey10  INTON
0091 00088 01               RTN
0092 0008A
0093 0008A                  END
```


**** SYMBOL TABLE ****

```
 FiLeNd         0008A Rel  0093 -
=HEXDEC         Extrn Ukn       -  0030
=KEYBUF         Extrn Ukn       -  0076 0083
=MOVE=M         Extrn Ukn       -  0081
=Num2D1         00000 Rel  0028 -
=fkey           00046 Rel  0073 -
 fkey10         00084 Rel  0090 -  0088
 num10          00010 Rel  0034 -  0036
 num15          00017 Rel  0038 -  0029
 num20          0001B Rel  0040 -  0043
 num30          0002F Rel  0049 -  0055
```

Source : Jpcutil.as

Object : obj/Jpcutil.ao

Listing : list/jpcutil.al

Date : Sat Aug 12 17:41:28 1989

Errors : 000

Areuh Assembler/Linker V2.4, (c) P. David & J. Taillandier 1986 Paris, France

Le Journal JPC est le bulletin de liaison entre les membres de l'Association "PPC Paris", régie par la loi de 1901. Le Club est éditeur de JPC, et son siège social est au 56, rue Jean-Jacques Rousseau, 75001 Paris.

La maquette de ce numéro a été préparée et réalisée par Laurent Chouraki, Jean Reibel et Aurel Rottman grâce à un système comprenant un HP71B, un lecteur de disquettes HP9114A, un HP9807A, deux HP9154 et une imprimante LaserJet Séries II.

Les dessins sont de Jean-Jacques Dhénin et Paul Courbis.

Directeur de la publication : Jean Reibel
Numéro ISSN : 0762 - 381X

Veuillez adresser toute correspondance à :
PPC Paris, BP 604, 75028 Paris Cedex 01.