JPC
JUILLET-AOUT 1990
NUMERO 71
Le numéro 40 F

## A PROPOS DU CLUB


| | | |
|---|---|---|
| Le Bureau | Editorial | *1* |
| | Courrier du coeur | *2* |
| | Assemblée générale extraordinaire | *2* |


## HP-28


| | | |
|---|---|---|
| G.Toublanc | Ecran virtuel | *4* |
| G.Toublanc | A proros de phi | *5* |


## HP-71


| | | |
|---|---|---|
| A.Rottman | Programmes en Basic écrivant l'adresse du destinataire | *10* |
| A.Rottman | Le "Mondiale" | *10* |
| A.Rottman | La transformation des instructions Basic dans une suite de symbits | *10* |
| A.Rottman | Les instructions de l'assembleur | *12* |
| J.P. Nalin | Conversion d'un nombre décimal en binaire | *14* |
| G.Toublanc | Ecran virtuel | *14* |
| J.Elhay | Vérification de la boucle HP-IL | *17* |
| J.P. Nalin | Editeur de Graphiques (Programme PIXEL) | *18* |

# EDITORIAL

Des difficultés matérielles, longtemps restées sans solution, ne nous ont pas permis de sortir plus tôt ce numéro 71.
Ces difficultés ont été, dans l'ordre, l'impossibilité de trouver un endroit pour installer l'Integral PC, ses disques durs et, surtout, l'imprimante LaserJet; puis les démissions successives de ceux des Membres qui savaient se servir de l'IPC; comme ils sont partis sans avoir transmis leur savoir à ceux qui restaient, et que les documentations des programmes n'étaient pas toujours faites, on imaginera sans peine que la reprise n'était guère facile.

Or, ce matériel étant le seul qui nous permette de tirer la maquette du Journal dans un temps raisonnable, il fallait réapprendre à s'en servir.
Pierre Franck, malgré qu'il ne dispose pas de beaucoup d'espace s'est proposé pour le garder chez lui. Dès lors, nous avons pu, au prix de nombreux essais et erreurs, réapprendre tout ce qui nous manquait, et documenter au passage ce qui ne l'était pas... Ce qui a représenté un très gros travail, qui nous permet de pouvoir relancer l'édition de JPC.
Les activités professionnelles de Jean Reibel ne lui permettant plus d'assumer la Présidence du Club, il a à son tour démissionné, mais continue à nous donner des coups de main, chaque fois qu'il le peut.
Le Bureau est donc à l'heure actuelle composé de seulement deux Membres, Aurel Rottmann et Pierre Franck. Eric Gengoux, dont les articles ont été appréciés tant dans JPC que dans certaines Revues Etrangères, s'est proposé pour assurer un certain nombre de tâches. Il va sans dire qu'il faut que d'autres membres se manifestent pour remplacer ceux qui sont partis.
Le plus gros travail pour la réalisation de JPC est la saisie des textes et leur correction. Il peut être considérablement réduit si les articles nous parviennent sur disquette; nous pouvons exploiter les formats HP-71, IPC (HP-UX), MS-DOS (tous types de disquettes). Si vous ne disposez pas de micro, vous pouvez nous envoyer vos articles manuscrits - mais ceci doit rester limité, et nous recherchons des bonnes volontés pour aider Guy Toublanc dans ce travail de resaisie.
A l'évidence, le Bureau ne pourra pas continuer à fonctionner à deux ou trois : aussi une Assemblée Générale Extraordinaire devra-t-elle avoir lieu en octobre, pour élire un nouveau Bureau et envisager l'avenir du Club Pour que PPC-Paris puisse tourner dans les meilleures conditions, et que les attributions de chacun des Membres du Bureau soient réalisables raisonnablement, il faut que le Bureau compte au moins cinq Membres... Envoyez-nous donc dès à présent vos candidatures, l'avenir du Club en dépend et rien ne pourra se faire sans bonnes volontés.

JPC ne manque pas de matière première pour le moment, nombre d'entre vous continuant à se servir de leurs 41, 71 et même 75; les programmes pour la 28 sont nombreux, montrant bien le succès de cette machine, et la 48 vient à son tour montrer son nez, avec déjà 3 possesseurs rien que parmi les Membres que nous connaissons personnellement. Sont-ils les seuls? Faites-nous connaître vos réflexions, remarques et découvertes.
Nous vous souhaitons à tous de bonnes vacances.

LE BUREAU

## LE COURRIER DU COEUR

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

---

## ASSEMBLEE GENERALE EXTRAORDINAIRE

Nous allons organiser une assemblée générale extraordinaire,prévue pour le mois d'Octobre,mais la date est encore inconnue,pour des raisons de réservation de salle (nous sommes au mois d'août).
Il s'agit d'élire des membres du bureau (Nous ne sommes que deux,c'est insuffisant) Il faut notamment un président et un secrétaire s'occupant du courrier (vous avez pu constater de nombreuses déficiences dans ce domaine !) Nous attendons donc de nombreuses candidatures.

## HP-28


| | | |
|---|---|---|
| G.Toublanc | Ecran virtuel | *4* |
| G.Toublanc | A propos de phi | *5* |

# ECRAN VIRTUEL

La TOOLBOX de Jean-Paul Barre (JPC 67) permettait d'obtenir un écran virtuel de 166 pixels de large correspondant à la longueur des lignes graphiques de l'imprimante HP82240. Je vous propose quelques petites améliorations permettant de gagner un peu de rapidité et d'encombrement mémoire.
Le programme PSET a été réécrit en assembleur avec intégration des vérifications de validité des paramètres ainsi que des calculs d'arrondis, donc les STOF, RCLF,et 0 FIX, ne sont plus nécessaires dans les programmes LINE, VSP, VPAR, et VCAR.
La routine LINE a été réécrite pour tracer en continu des droites de pente supérieure à 1.
Les routines DISPFEN et VCAR ont été allégées.
D'autre part PSET permet maintenant d'aoir les AXES dans la DISPOSITION HABITUELLE.
Il resterait encore à faire une routine LINE en assembleur ce qui permettrait de tracer plus confortablement des axes ou des asymptotes par exemple.

Le programme assembleur pour HP-28S

```
      CON(5) #02C67
      CON(5) #0BC97
      CON(5) #07610
      CON(5) #0BC97
      CON(5) #02D12
      CON(2) 4
      NIBHEX 19353425
      CON(5) #02C96
debut CON(5) (fin)-(debut)
      GOSBVL #05081
      GOSUB  pop1n
      GOC    gtout
      C=0    W
      LC(2)  166
      D=C    W
      ?A>=C  W
      GOYES  gtout
      R0=A
      GOSUB  pop1n
      GOC    gtout
      R1=A
      ASRB
      ASRB
      ASRB
      A=A+1  W
      C=D    W
      ACEX   W
      GOSBVL #01FB3
      B=B+B  W
      D1=D1- 10
      C=DAT1 A
      D1=C
      D1=D1+ 5
      A=DAT1 A
      C=0    W
      LC(1)  5
      C=A-C  A
      B=C-B  W
gtout GOC    OUT
      D1=D1+ 5
      AD1EX
      A=A+B  A
      C=R0
      C=C+C  A
      A=A+C  A
      D1=A
      A=R1
      C=0    A
      LC(1)  7
      A=A&C  A
      C=C-A  A
      A=0    B
      A=A+1  B
      GONC   pset20
pset10 A=A+A  B
pset20 C=C-1  P
      GONC   pset10
      C=DAT1 B
      A=A!C  B
      DAT1=A B
OUT   GOSBVL #050B8
      A=DAT0 A
      D0=D0+ 5
      CON(4) #C808
pop1n D1=D1+ 5
      C=DAT1 A
      CD1EX
      D1=D1+ 10
      A=DAT1 16
      D1=C
      A=A-1  W
      RTN
fin   CON(5) #02A4E
      CON(5) #11
      NIBHEX 721935342572
      CON(5) #079C8
      CON(5) #0A220
      CON(5) #07644
      CON(5) #02F90
```

Voir chaîne de caractères à assembler en rubrique "Le coin des LHEX"

LINE

```
<< R->C 3 ROLLD DUP2 PSET R->C SWAP OVER -
DUP C->R ABS SWAP ABS MAX DUP
 IF
```

```
 THEN SWAP OVER / SWAP ROT SWAP 1 SWAP
  START OVER + DUP C->R PSET
  NEXT
 END DROP2
>>
```

DISPFEN

```
<< SCR SIZE 166 / IP -> X Y S << "" Y DUP 31 +
 FOR YY SCR YY 8 / IP DUP S
   IF <
   THEN 166 * X + DUP 136 + SUB +
   ELSE DROP2
   END 8
 STEP CLLCD ->LCD
 >>
>>
```

VCAR

```
<< VSP -> p x v y w s <<
  FOR P P DUP 'X' STO x * v + EQ ->NUM y * w
  + PSET s
  STEP 'X' PURGE
 >>
>>
```

Pour l'emploi de la toolbox relisez JPC 67.

Guy Toublanc (276)

---

## A PROPOS DE PHI

Il y a déjà un certain temps, j'avais créé un lex avec six fonctions mathématiques, quatre traitant des nombres premiers. Mon but avait été de rendre plus facile la vie des programmeurs d'utilités ou de divertissements mathématiques. Or il ne semble pas que les gens intéressés par ce domaine très particulier, se soient manifestés. Je les invite à regarder du côté des articles pour HP28 où j'ai amorcé un bout d'explication sur la fonction PHI avec un petit problème d'application. Peut-être que cela incitera les imaginatifs à se manifester pour nous gratifier d'articles divertissants dans dans ce domaine et nous éviter les "vilaines pages blanches" .

Guy Toublanc (276)

### L'IMPOSSIBLE ETOILE DE DAVID

Il existe plus d'une demi-douzaine de fonctions arithmétiques et parmi celles-ci l'indicateur d'Euler ou fonction phi joue un rôle important en théorie des nombres ainsi que dans de multiples domaines d'applications.
Si phi a sa place dans JPC Rom, il n'y a aucune raison qu'elle soit la belle inconnue de nos HP28.

phi indique le nombre de nombres inférieurs à un nombre donné et relativement premiers avec lui.
Par exemple phi(10)=4 car les nombres 1,3,7,9 sont relativement premiers avec 10.
par convention phi(1)=1
si n est décomposable en facteurs premiers tel que n = p1^a1*p2^a2*....*pk^ak

phi(n) = n(1-1/p1)(1-1/p2).....(1-1/pk)

si n est premier phi(n) = n-1
phi est une fonction multiplicative comme les autres fonctions arithmétiques et a de nombreuses propriétés. Elle est liée à une autre fonction arithmétique: la fonction de Moebius.
Pour calculer phi(n) il faut donc connaître les facteurs premiers de n. Or, il y a déjà quelques temps, j'avais proposé des fonctions en assembeur qui sont tout indiquées pour calculer phi très rapidement. Il s'agit des fonctions DIVI (JPC 61) et EDIVI (JPC 62).
DIVI et EDIVI renvoient le plus petit facteur premier de n ou 0 si n est premier.
DIVI admet les arguments jusqu'à 999999999999 tandis que EDIVI admet les arguments jusque 2^64-1 c'est-à-dire # 18446744073709551615d mais en mode binaire decimal et EDIVI est deux fois plus rapide.
On peut utiliser EDIVI indifféremment avec les deux types d'arguments à l'aide du programme :

```
R?B
<< DUP TYPE 0 ==
  IF
  THEN DEC R->B
  END
>>
```

et donc pour ceux qui utilisent EDIVI il sera possible de mettre dans la variable DIVI le bout de programme :
<< R?B EDIVI >>

Ainsi pour tout le monde les arguments devront être inférieurs à 1E12.
Voici le programme PHI :

```
<< DUP
  WHILE DUP DIVI DUP DUP 1 <> AND
  REPEAT ROT OVER / OVER 1 - * ROT ROT
   WHILE DUP2 MOD NOT 3 PICK 1 <> AND
   REPEAT SWAP OVER / SWAP
```

```
  END DROP
 END DROP DUP 1 = =
 << DROP >>
 << SWAP OVER / SWAP 1 - * >>
 IFTE
>>
```

exemples :
540 PHI -> 144
752329095943 PHI -> 752163703380

La cryptographie pourrait être un domaine de choix comme utilisation de phi, mais cela demande des développements qui nécessitent un article entier. Aussi je me contenterai cette fois d'un modeste problème.

Partons de n points également espacés sur un cercle. De combien de manières peut-on tracer une étoile à n sommets, sans soulever le crayon, en joignant les sommets de 2 en 2, de 3 en 3 etc... si cela est possible.

La réponse est :
( phi(n) - 2 ) / 2
par exemple pour n = 7 (phi(7)-2)/2=2

il n'y a que 2 polygones étoilés à 7 branches et je vous laisse le soin de faire les figures.
Pour n=6 (phi(6)-2)/2=0
le polygone étoilé à 6 branches est impossible (d'où le titre de l'article).
Maintenant voyons le problème inverse: pour un nombre j d'étoiles différentes qu'elle est la valeur de n ?
réponse : on doit avoir phi(n)=2j+2
est-ce toujours possible ?
Non. Par exemple pour j=6 2j+2=14 phi(n)=14 n'existe pas. 26 et 34 sont dans le même cas.

JPC n'ayant pas de "coin des codes" pour HP28" comme la HP71 a son coin des lhex, je vous propose un rappel des listes de codes à assembler avec le programme de votre choix.

On remarquera que la présentation des listings est la même que celle adoptée par Paul Courbis et Sébastien Lalande dans leur livre bien connu. Je me rallie donc à cette méthode et je propose, pour ceux qui ne possèdent pas le livre cité, deux programmes équivalents aux leurs, pour contrôler les codes entrés.
Lorsque les codes ont été entrés dans une seule chaîne faire PIL5 puis après le programme terminé faire EDIT. Les codes pourront ainsi être relus très facilement. Puis faire DEPIL5 pour obtenir la chaîne à assembler.

```
PIL5
<< ""
  DO OVER 1 5 SUB 10 CHR
 + + SWAP 6 9000 SUB SWAP
  UNTIL OVER "" = =
  END SWAP DROP
>>

DEPIL5
<<
  WHILE DUP 10 CHR POS DUP
  REPEAT DUP2 1 SWAP 1 - SUB ROT ROT 1 +
      9000 SUB +
  END DROP
>>
```

Toujours pour les gens qui n'auraient pas pu se procurer le livre voici une variante du programme qui permet de doubler la vitesse d'une HP28S. Cette vitesse redevenant normale si la touche <ON> a été actionnée.

```
      CON(5) #02C96
debut CON(5) (fin)-(debut)
      AD0EX
      LCHEX #FFF00
      D0=C
      DAT0=C XS
      D0=A
      A=DAT0 A
      D0=D0+ 5
      CON(4) #C808
fin
```

Voici la liste des codes à assembler.

HP28S DVIT 20 octets
69C20 32000 13234 00FFF 13415 42130 14216 4808C

Guy Toublanc (276)

## ECRAN VIRTUEL

La TOOLBOX de Jean-Paul Barre (JPC 67) permettait d'obtenir un écran virtuel de 166 pixels de large correspondant à la longueur des lignes graphiques de l'imprimante HP82240. Je vous propose quelques petites améliorations permettant de gagner un peu de rapidité et d'encombrement mémoire.
Le programme PSET a été réécrit en assembleur avec intégration des vérifications de validité des paramètres ainsi que des calculs d'arrondis, donc les STOF, RCLF,et 0 FIX, ne sont plus nécessaires dans les programmes LINE, VSP, VPAR, et VCAR.

La routine LINE a été réécrite pour tracer en continu des droites de pente supérieure à 1.
Les routines DISPFEN et VCAR ont été allégées.
D'autre part PSET permet maintenant d'aoir les AXES dans la DISPOSITION HABITUELLE.
Il resterait encore à faire une routine LINE en assembleur ce qui permettrait de tracer plus confortablement des axes ou des asymptotes par exemple.

Le programme assembleur pour HP-28S

```
      CON(5) #02C67
      CON(5) #0BC97
      CON(5) #07610
      CON(5) #0BC97
      CON(5) #02D12
      CON(2) 4
      NIBHEX 19353425
      CON(5) #02C96
debut CON(5) (fin)-(debut)
      GOSBVL #05081
      GOSUB pop1n
      GOC   gtout
      C=0   W
      LC(2) 166
      D=C   W
      ?A>=C  W
      GOYES gtout
      R0=A
      GOSUB pop1n
      GOC   gtout
      R1=A
      ASRB
      ASRB
      ASRB
      A=A+1 W
      C=D   W
      ACEX  W
      GOSBVL #01FB3
      B=B+B W
      D1=D1- 10
      C=DAT1 A
      D1=C
      D1=D1+ 5
      A=DAT1 A
      C=0   W
      LC(1) 5
      C=A-C A
      B=C-B W
gtout GOC   OUT
      D1=D1+ 5
      AD1EX
      A=A+B A
      C=R0
      C=C+C A
      A=A+C A
      D1=A
      A=R1
      C=0   A
      LC(1) 7
      A=A&C A
      C=C-A A
      A=0   B
      A=A+1 B
      GONC  pset20
pset10 A=A+A B
pset20 C=C-1 P
      GONC  pset10
      C=DAT1 B
      A=A!C B
      DAT1=A B
OUT   GOSBVL #050B8
      A=DAT0 A
      D0=D0+ 5
      CON(4) #C808
pop1n D1=D1+ 5
      C=DAT1 A
      CD1EX
      D1=D1+ 10
      A=DAT1 16
      D1=C
      A=A-1 W
      RTN
fin   CON(5) #02A4E
      CON(5) #11
      NIBHEX 721935342572
      CON(5) #079C8
      CON(5) #0A220
      CON(5) #07644
      CON(5) #02F90
```

Voir chaîne à assembler en rubrique "Le coin des LHEX".

LINE

```
<< R->C 3 ROLLD DUP2 PSET R->C SWAP OVER -
DUP  C->R ABS SWAP ABS MAX DUP
 IF
 THEN SWAP OVER / SWAP ROT SWAP 1 SWAP
  START OVER + DUP C->R PSET
  NEXT
 END DROP2
>>
```

DISPFEN

```
<< SCR SIZE 166 / IP -> X Y S << "" Y DUP 31 +
 FOR YY SCR YY 8 / IP DUP S
   IF <
   THEN 166 * X + DUP 136 + SUB +
   ELSE DROP2
   END 8
 STEP CLLCD ->LCD
```

```
 >>
>>
```

VCAR

```
<< VSP -> p x v y w s <<
  FOR P P DUP 'X' STO x * v + EQ ->NUM y * w + PSET s
  STEP 'X' PURGE
 >>
>>
```

Pour l'emploi de la toolbox relisez JPC 67.

Guy Toublanc (276)

## HP-71


| | | |
|---|---|---|
| A.Rottman | Programmes en Basic écrivant l'adresse du destinataire | *10* |
| A.Rottman | Le "Mondiale" | *10* |
| A.Rottman | La transformation des instructions Basic dans une suite de symbits | *10* |
| A.Rottman | Les instructions de l'assembleur | *12* |
| J.P. Nalin | Conversion d'un nombre décimal en binaire | *14* |
| G.Toublanc | Ecran virtuel | *14* |
| J.Elhay | Vérification de la boucle HP-IL | *17* |
| J.P. Nalin | Editeur de Graphiques (Programme PIXEL) | *18* |

## PROGRAMMES EN BASIC QUI ECRIVENT L'ADRESSE DU DESTINATAIRE ET LA VOTRE.

En partant d'une présentation de A. BOLDIREFF j'ai écrit 2 programmes qui vous permettent avec la Thinkjet d'avoir un en-tête et votre adresse bien présentés dans vos lettres. L'exécution du programme est coservée dans le vecteur D$, et peut etre répétée par une exécution après le STOP. Vous pouvez particulariser ces programmes à votre guise.
Le premier programme ADRESSE est conversationnel, le second programme ADNOM utilise vos coordonnées dans le DATA.

Aurel ROTTMAN (289)

---

## LE "MONDIALE"

L'expression est sans doute à la mode et notre amour du football bien acquis mais en fin de compte on a une autre façon d'interpréter cette passion sportive . Parce que par ailleurs J.P. Nalin attaque le problème du graphisme du HP71 je vous donne un programme que j'ai publié dans DATAFILE V7N1 et vous suggère de lire sans faute l'article de Brian D. STEEL (DF V4N5/6 p.39).
Le programme montre le travail et le but (au carré) du footballeur.

A. ROTTMAN (258)

---

## LA TRANSFORMATION DES INSTRUCTIONS BASIC DANS UNE SUITE DE SYMBITS.

Rappel de terminologie: symbit = token ; parse = analyse ; buffer = entasseur.
Pour comprendre l'essentiel du système opérationel il faut pouvoir répondre à la suivante question: qu'est-ce qu'il se passe quand on tape quelque chose sur le display du HP71 et on envoie 'ENDLINE' ?
Pour pouvoir donner une réponse il faut savoir que les instructions BASIC, qu'elles soient précedées par un numéro de ligne ,ou non, sont représentées intérieurement par une suite de symbits. Si le numéro de ligne est absent l'instruction est immédiatement exécutable (après avoir actionné la clef 'ENDLINE'). Les plus usuels symbits sont donnés dans la table qui suit:

```
-----------------------------------------
Symbole .Sb. Symbole .Sb. Symbole .Sb.

-----------------------------------------
t!     .CF.Comment  .D0+.tREM     .E6.
tIF    .DF.tTHEN    .F4 .tELSE    .F5.
tCOMMA .F1.tIMMAGE  .FF .tLBLRF   .0E.
tPRMST .F3.tPRMEN   .F8 .tCVAL    .E1.
.........................................
-----------------------------------------
Symbole .Sb. Symbole .Sb. Symbole .Sb.

-----------------------------------------
t@     .4F.tLINE#   .F0 .tREM     .E6.
tEXTIF .F4.tGOTO    .D0 .tCALL    .F9.
tLBLST .6F.tLITRL   .C4 .tSUB     .C1.
tCREF  .E0.tXFN     .B3 .tFFN     .B4.
.........................................
```

Après chaque no. de ligne ou t@ il y a un octet qui donne la longueur de l'instruction et qui pointe sur un symbit final (@ ou EOL). Le principe de la suite des symbits pour l'interprétation du BASIC est le même que le chaînage des fichiers. Si on note les séquences formelles :
A = #Ligne,long.inst.,inst.
B = Long.inst.,inst.
C = #ligne,long.inst.
on a les suites des symbits suivantes:
A,4F,B,tEOL
A,4F,B,t!,comment.,D0,F0
C,tLBLST,Etiq.,4F,B,0F
C,tIF,Expression,tTHEN,tEXTIF,B,0F
C,tIF,Expression,tTHEN,tLINE#,LINE#,0F
C,tIF,Expression,tTHEN,tLBLRF,String expr.,0F, (Une etiquette est une chaîne de charactères)
C,tIF,Expression,tTHEN, tLBLRF,tLITRL,ASCII Label,0F
#Ligne,tCALL,Nom,tPRMEN,0F

#Ligne,tCALL,Nom,tPRMST, P1,tCVAL,P2,tCREF, t#,#tCREF,tPRMEN,0F
#Ligne,tCALL,Nom,tPRMEN, tIN,Nom Fichier,4F Line#,tCALL,chaîne ASCII entre quotes,0F (Nom du Programme entre quotes)
Line#,tCALL,tLITRL,chaîne ASCII sans quotes,0F (Nom du Programme sans quotes)
Pour un sousprogramme l'obtention des symbits est similaire mais tCOMMA existe entre les paramètres. Dans tous les cas le nom du sousprogramme est précédé de tLITRL. Les constantes mono-numériques ont comme symbit la valeur ASCII correspondante. Les entiers (2-12 dig.) sont représentés par un octet qui précise le nombre des chiffres et un nib pour chaque chiffre (dans l'ordre renversée). Une constante flotante est représentée

par un octet définissant le nBombre des chiffres de la mantisse et 3 nibs sous forme de complément par rapport à 9 qui représente l'exponent. Les chaînes constantes se présentent sous la forme "quote,charactères,quote". Les variables sont représentées par 1 H 3 octets. (Ex. A$=E0D214=tLBLRF,tSVAR,A ; A=0814=tINIG,A)

Les fonctions et opérateurs sont codées de la façon suivante: Fonctions implicites (internes) qui sont codées avec un seul octet.

Les opérateurs de relation qui ont un octet+ un nib, qui sert d'identificateur de la relation respective. Fonctions externes (XFN llleee) ou mots-clef (XWORD) ou 'lll' représente l'identificateur du LEX en décimal et 'eee' représente le no. de l'entrée du mot-clef toujours en décimal sont codées sous la forme d'un symbit tXFN et un autre octet qui donne le no. d'ordre de l'ID. On a après un nib, qui donne le nombre des paramltres. Une fonction qui n'a pas de paramètres est traitée comme un opérateur.

Les tableaux sont codés sous forme d'un tARRAY symbit plus un jusqu'à trois octets qui décrivent le nom du tableau et un nib qui donne le nombre d'indices. Les fonctions d'exception (Funny funct.) sont codées sous la forme: 'tFFN+ID+Fn#+Long.'

Quand le LEX définissant le mot-clef est absent, on décompile seulement le nom du XWORD et dans le cas d'une XFN on donne la notation 'XFNlllleee'.

EXEMPLE D'UNE SERIE DES SYNBITS POUR UN PROGRAMME BASIC.

Le detail concernant l'exemple pratique est illustré plus bas en prenant comme source le programme BASIC qui suit:

```
0010 A1=SIN(30) @ DISP A
0015 N2=COS(A1) @ GOTO 33
0020 B=PI
0030 CALL C
0040 IF X=Y THEN 20 ELSE B=C
0045 END
```

Titre du fichier (16)-en paranthèses le nombre de nibs.

---

2FA82 4554354502020202

---

CODE du fichier (4), $ (2), DATE&HEURRE (10) ; ($=DRAPEAU & CODE COPIE)

---

2FA92 412E005191133098

---

Pointeur longueur (5) 9E+2FAA2=2FB40
Commenc. du fichier qui suit.
Soustitre (12) (5)=ChaQnage des sousprogrammes FFFFF (inexistants)
(5)=Etiq. ou Fonct. définie par l'usager FFFFF (inexistants) tEOL (2) F0\

---

2FAA2 E9000FFFFFFFFFF0

2FAB2 F

---

#ligne (4) 0010, (2) 0E Long. instruction E+2FAB7=2FAC5 pointe vers t@ tADIG1 (2) 61, A (2) 41, tCOMMA (2) F1, tINT2 (2) 0C, Param. (2) 30 tSIN (2) 96, t@ (2) 4F

---

2FAB3 0100E016141FC00
2FAC2 3694F

---

Long. instruct. (2) 06 6+2FAC7=2FACD pointe vers tEOL F0, tDISP (2) C5, A (2) 41, tEOL (2) F0, line# (4) 0015

---

2FAC7 605C140F510

---

Long.Instr. (2) 0E 2FAD3+E=2FAE1 e pointe vers 4F tADIG2 (2) 62, N (2) 4E,tCOMMA (2) F1, tADIG1 (2) 61, A (2) 41, tCOS (2) 97

---

2FAD2 0E026E41F1614794

---

t@ (2) 4F, 0F (2) Long.Instr. 2FAE3+F =2FAD2 pointe vers EOL F0, tGOTO (2) DD, tEOL (2) F0, 20 BCD (8)

---

2FAE2 FF0DDF0000000200

---

tEOL (2) F0, LINE# (4) 0020, Long. Instr. (2) 8+2FAF7=2FAFF pointe vers tEOL, B (2) 42, tCOMMA (2) F1,tPI (2) 79, tEOL (2) F0

---

2FAF2 0F020080241F970F

---

LINE# (4) 0030, Long.Instr. (2) A+2FAB05=2FAB0F pointe vers F0, tCALL (2) F9 tLITRL (2) 4C, C (2) 43, tPRMEN (2) F8, tEOL (2) F0

---

2FB02 0300A09F4C348F0F

---

LINE# (4) 0040, Long.Instr. (2) 18 2FB16+18=2FB2E pointe vers tEXTIF F4, tIF (2) DF, X (2) 58, Y (2) 59, tRELOP (2) 8A, (1) 2, aprIs ETIQ. (2) 4F.

---

2FB12 040081FD8595A824

---

1) 4, tTHEN (2) 4F, tEOL (2) 0F, (8) BCD 20, tEXTIF (2) F4,tINT10 (2) 04

---

2FB22 FF000000002004F40

----------\ \tELSE (2) F5, tEXTIF (2) F4, Long.Instr. (2) 08 8+2FB36=2FB3E pointe vers EOL, B (2) 42, tCOMMA (2) F1, C (2) 43, tEOL (2) F0

---

2FB32 5F4F80241F340F54

---

Aurel ROTTMAN (289)

---

## LES INSTRUCTIONS DE L'ASSEMBLEUR.

Les instruction d'un langage ASSEMBLEUR diffèrent par rapport aux instructions BASIC parce qu'elles sont des operations qui s'effectuent au niveau des registres du microprocesseur (CPU). La cellule unitaire d'un registre est le demi-octet (nib) c'est à dire un ensemble de 4 bits.
Il y a 4 registres opérationnels pour le traitememt des données : A,B,C,D et 5 registres de stockage passager (temporel) : R0,R1,R2,R3,R4 qui gardent pour un certain temps les données des registres usuels. Ces registres ont une longueur de 16 nibs (16*4=64 bits) qui ont la structure suivante (le nib le plus H droite est notE 0 et celui le plus H gauche 15) :

0-1 B (champs byte ou exponent) (code 6E)
2 XS signe de l'exponent (code 2A)
0-2 X exponent et son signe (code 3B)
3-14 M la mantisse (code 5D)
15 S le signe de la mantisse (code 4C)
0-15 W l'ensemble (whole) du mot (code 7F)

Un registre spécial de 1 nib , le registre P (pointeur) permet de pointer sur un certain nib du registre de travail. Le nib pointé est noté aussi avec P et il permet la définition d'un nouveau champs:

0-P WP mot partiel ou mot pointé (code 19)

Deux autres registres pointeurs ont un rôle essentiel; il s'aggit des registres H 5 nibs D0 et D1 qui sont utilisés pour garder l'adresse de la pile mathématique et du compteur du programme. Un autre registre de 5 nibs PC détermine l'instruction executée par le CPU. La partie correspondante aux 12 nibs inférieurs constitue le registre d'état ST (utilisé par le logiciel d'exploitation). Il existe aussi deux registres d'entrée et sortie IN de 4 nibs et OUT de 3 nibs. Un registre spécial de 1 bit est le registre du report (CARRY register). L'état du registre report est modifié par certains calculs et tests logiques. Enfin pour le logiciel intérieur il y a encore un registre de 1 nib dont les bits ont la signification suivantes:
Bit 0 XM module externe qui manque.
Bit 1 SB Bit sticky (utilisé dans des opérations de transfert)
Bit 2 SR demande de service.
Bit 3 MR Module retiré.

Quand on lit des données de la mémoire pour le transférer dans les registres du CPU le microprocesseur place les nibs avec une adresse inférieure dans les nibs droits du registre. Quand les données sont inscrites en mémoire le CPU place les nibs les moins significatifs dans les nibs inférieurs de l'adresse. De cette façon dans les deux cas les données se trouvent transférer en ordre renversée.

L'ensemle des instructions de l'ASSEMBLEUR.

On classifie les instructions en 17 categories:
1. Instructions GOTO ; elles sont c,l ou tl (c=courtes, l=longues,tl=trIs longues) et ic ou c(voir plus bas)
2. Instructions GOSUB qui sont c,l,tl.
3. Instructions SUBROUTINES RETURN qui sont ic,c (ic= inconditionnelles, c=conditionnelles)
4. Instructions de test (pour les registres,pointeurs, état logiciel,état du programmme)
5. Instructions qui comportent un pointeur.
6. Instructions sur l'état du programme ou du logiciel.
7. Instructions de contrôle du système.
8. Instructions d'exploration des clefs.
9. Instructions d'opérations sur les registres.

10. Instructions de manipulation des pointeurs des données.
11. Instructions de transfert des données.
12. Instructions de transfert de charges.
13. Instructions de decalage.
14. Instructions des opérations logiques.
15. Instructions arithmétiques.
16. Instructions NO-OP (pas d'opération).
17. Instructions PSEUDO-OP.

Sous la forme la plus compacte l'ensemble des instructions a été présenté par J. BAKER (V12 N2-p.10/12). Il y a la pour chaque instruction le code décimal et hexadécimal et le type d'instruction. Il faut considérer que le codage d'une instruction peut demander de 2 H 18 nibs. Certaines instructions contiennent dans le code une sélection du champs du registre, une distance de transfert ou une constante. J'ai choisi une présentation sur plusieurs colonnes qui est donnée en annexe.
L'Etude de l'ASSEMBLEUR implique d'abord une bonne connaissance des programmes internes qui forment le système opérationnel de la HP71. L'annexe donne des exemples des programmes utilitaires.
L'âme du système opérationnel est le programme 'MAIN-LOOP' (Boucle principale voir description qui suit). Chaque fois quand on attend quelque chose, on est automatiquement connecté à ce programme. Quand on appuie une touche la représentation ASCII de cette touche est placée dans l'entasseur de l'Ecran de visualisation. (98 octets à partir de l'adresse 2F480). L'entaseur de visualisation est distinct de l'affichage LCD. Cet affichage est adressable en trois segments (les adresses 2E100,2E200, 2E300). Le système opérationnel fait que les LCD reflètent la configuration de l'entasseur de visualisation.
Ex. HTA$(PEEK$('2F480',48)) donne HTA$(PEEK$('2F480',48)) sur l'afficheur.
L'examen syntactique commence quand :(1) la clef 'ENDLINE' est actionnée , (2) quand le calculateur est connecté et un entasseur extérieur de commande ou un entasseur de demarage existe et le sousprogramme INBS est prêt et pointe vers cet entasseur , (3) une clé d'execution directe est actionnée (INBS pointe vers la définition de la clé dans le fichier des clefs) ,(4) la commande TRANSFORM initialise aussi ce processus. Le texte sous forme analysé (symbits) est comparé aux dictionnaires de base. Tout les mots clefs de base se trouvent dans 2 fichiers LEX (ROM1 et ROM2).

Le programme WRDSCN( analyse d'un mot) essaie, pour le texte pointé par D1, de trouver un même symbit dans un des fichiers LEX. WRDSCN appelle NTOKEN pour trouver un certain lexème (symbit donné) .Cette opération est ce qu'on définie par 'polling' (elle précise si un LEX peut répondre positivement (il contient un symbit donnE). Le processus se fait par un LEX 'poll-handler' qui au moment il est activé précise s'il posède ou non ce symbit. Le code est dans le registre B(A) et le registre D(A) est le pointeur dans l'entaseur des LEX. La réponse d'un p.h.LEX est positive si le registre XM (le bit du module ext.manquant) est XM=0, sinon XM=1.

Programmes Utilitaires.

GTEXT1 Pour un synbit donné fournit en sortie le texte correspondant.
EOLDC Verifie l'existence d'une expression terminale t@,t!,tEOL.\ \EOLXC* Appelle EOLDC si l'expression terminale est trouvée; on traite toute l'instruction en passant H OUTELA sinon retour avec 'carry clear'.
VARDC Décompilation des variables.
LIN#DC décompile et met en sortie un no. de ligne.
ASCICK Copie des charactIres de la suite d'entrée vers l'entasseur de sortie jusqu'à ce qu'on trouve un charactIre dont le bit supérieur est activé.
EXPRDC Décompile les expressions pointées par D1.
FILDC* Décompile le fichier spécifié.
ARYDC Décompile les tableaux compilés par ARRYCK.
LBLDC Au cas ou D1 est sur tLBLRF (le sunbit de l'etiquette) examine l'etiquette ; si elle est littérale, elle est ssortie entre quotes ,sinon la chaQne littérale est dEcompilée et le 'carry clear'.
SKIPDC Si un XWORD inconnu est rencontré, provoque le saut de D1 à la fin de l'instruction et on passe à OUTELA.
DSPCHA et DSPCHC Envoie, dans l'entasseur de l'affichage, les charactères du registre A (respectivement B). On a affaire à des programmes de visualisation ('display is C(D) device') au niveau du CPU du HP71 L'action est effective sur l'entasseur de sortie mais par sur les LCD. La réactualisation des LCD se fait par BLDDSP.
BLDDSP est automatiquement appelé par un retour charriot ; l'appel est explicite si le display LCD impose une mise à jour. Un bit status (EXACT) est desactivé si une modification de l'entasseur de visualisation implique une modification de l'ensemble des bits LCD.
BLDBIT Etablit a un autre niveau (inférieur) le format des bits LCD suivant la situation d'un autre entasseur. De cette façon on peut introduire les clefs VIEW et ERRM.
KEYSCN réalise le balayage des clefs. Le point d'entrée est DEBNCE qui introduit un retardement pour empêcher l'omission de certaines clefs qui sont mises dans l'entasseur des clefs jusqu'au moment où celui-ci est complètement rempli.

POPBUF enlève les clefs de l'entasseur.

DSPRST rétablit l'affichage.
EXPEXC Execute les expressions par l'examen de la suite des synbits et les valeurs respectives sont mises sur la pile.
FNRTN1 Résume l'execution des expressions après avoir mis une valeur sur la pile.
BF2STK ajoute sur la pile une chaîne de la mémoire.
STKCHR crée une chaîne sur la pile en utilisant ADHEAD.
ADHEAD ajoute le titre de la chaîne à une chaîne placée sur la pile par STKCHR.
POPIN extrait un argument numérique de la pile.
POF2N extrait deux arguments.
POPIS verifie l'existence d'une chaîne sur la pile.
CLOSET ferme un fichier ouvert.
CRTF crée un fichier d'un type arbitraire.
FIBADR détermine l'adresse de l'entasseur des fichiers dans le registre D0.
FINDF trouve un fichier en mémoire d'après son nom et type de mémoire.
FSPEC* execute un fichier spécifié par un synbit afin d'établir le nom et type.
MVMEM* elargit ou contracte le contenu d'un fichier. Peut-être utilisé pour effacer un fichier.
OPENF ouvre un fichier donné par son nom et type.
PRGFMF efface un fichier de la mémoire.
PURGET efface un fichier de la mémoire ou disque.

(voir l'annexe)

Aurel ROTTMAN (289)

---

## CONVERSION D'UN NOMBRE DECIMAL EN BINAIRE

Ce programme a pour objet de convertir un entier en base décimale en un entier en base binaire. Son usage est des plus simple ; on lance le programme et un masque de saisie demande l'entier décimal en affichant la valeur par défaut de 10 ; une fois entré l'entier au clavier, faire ENDLINE ; il apparait alors à l'affichage l'entier en base décimale et son équivalent en binaire.
Ce petit utilitaire sait se faire apprécier lorsque l'on veut faire des travaux graphiques sur son HP 71 ou sur sa Thinkjet ; dans le premier cas, la configuration des bits est verticale, et dans le second, la configuration est horizontale... De quoi vous éviter d'avoir à faire des noeuds avec vos méninges !.
Bon travail... !

Jean Paul NALIN

---

## ECRAN VIRTUEL

Par deux fois les possibilités d'affichage de graphes sur l'écran du HP71 ont été concretisées dans JPC :
par Xavier Bille (JCP 42 et 47)
par Bruno Gil (JPC 59)
Certains adhérents possèdent le module "GRAPHIC" de Pierre David. Ce module est integré dans la version E de JPC Rom. Pierre met ainsi un logiciel très pratique, à la disposition d'un plus grand nombre d'utilisateurs potentiels pour faire du graphisme sur imprimante PCL et en particulier sur Thinkjet.
Pour ceux qui ne possèdent pas ou plus ce type d'imprimante, "GRAPHIC" ne leur est, à priori, d'aucune utilité, car les fichiers GRAPH générés par ce logiciel ne sont qu'imprimables.
Il fallait donc remédier à cela pour les deux types d'affichage "portatifs" :
- l'écran LCD du HP71 lui-même
-l'écran LCD HPIL CMT décrit par Eric Gengoux dans JPC 61.
Je vous propose deux nouvelles fonctions qui transforment une partie de fichier GRAPH en une chaîne affichable sous forme graphique :
- LCD$ pour l'écran HP71
- TRAN$ pour l'écran CMT
Deux exemples de fichiers basic utilisant ces fonctions permettent de se déplacer dans les fichiers GRAPH et même de numériser un peu comme avec une HP28.

MODE D'EMPLOI

Syntaxe des fonctions

LCD$ (D)
D = adresse en décimal du début de la zone du graphique à afficher

utilisation : GDISP LCD$ (D)

TRAN$ (H,D)
H = hauteur, en pixels, de la zone à visualiser
D = adresse en décimal du début de zone graphique à afficher

utilisation : OUTPUT :LCD ;TRAN$ (H,D)

PROGRAMME GLCD71

affichage du graphique sur l'écran LCD HP71 et numérisation.

RUN GLCD71 :
le programme demande le nom du fichier GRAPH, puis affiche le graphique

actions des touches :

[ LEFT ] déplace à gauche suivant le pas
[ RIGHT ] déplace à droite suivant le pas
[ UP ] déplace vers le haut suvant le pas
[ DOWN ] déplace vers le bas suivant le pas
[ TOP ] affiche le haut du graphique
[ BOTTOM ] affiche le bas du graphique
[ ( ] affiche la partie gauche du graphique
[ ) ] affiche la partie droite du graphique
[ S ] demande le pas
[ M ] affiche la partie centrale du graphique par défaut ou choisie
[ C ] mémorise le centre du graphique
[ E ] mémorise l'extrémité droite du graphique
[ V ] défilement vertical de haut en bas
[ H ] défilement horizontal gauche-droite
[ N ] demande de numérisation : appel du sous-programme DIGIT
[ F ] sortie du programme d'affichage sur HP71 puis sortie du programme GLCDCMT si l'appel s'est fait depuis celui-ci
[ Q ] sortie du programme

Lorsque la demande de NUMERISATION a été faite par la touche [ N ] un point clignote.

actions des touches :

[ LEFT ] déplace le point clignotant vers la gauche de 1 ou 2 pixels
[ RIGHT ] déplace le point clignotant vers la droite de 1 ou 2 pixels
[ UP ] déplace le point clignotant vers le haut de 1 ou 2 pixels
[ DOWN ] déplace le point clignotant vers le bas de 1 ou 2 pixels
[ S ] bascule : fait passer alternativement le pas de 1 à 2 ou de 2 à 1 pixels
[ O ] mémorise l'origine des coordonnées si on a fait coïncider celle-ci avec le point clignotant, puis rappelle qu'il faut numériser la graduation horizontale
[ U ] mémorise la graduation horizontale si on a fait coïncider celle-ci avec le point clignotant, demande sa valeur et rappelle qui'il faut numériser la graduation verticale
[ V ] mémorise la graduation verticale si on a fait coïncider celle-ci avec le point clignotant et demande sa valeur
[ = ] donne les coodonnées du point clignotant si l'origine des axes et les graduations ont été mémorisées, sinon fait BEEP
[ ATTN ] sort du mode numérisation et revient à l'affichage graphique simple du HP71

ATTENTION : respectez l'ordre [ O ] [ U ] [ V ]

## PROGRAMME GLCDCMT

affichage du graphique sur l'écran LCD CMT et possibilité d'appel des sous-programmes de GLCD71

RUN GLCDCMT :
le programme demande le nom du fichier GRAPH puis affiche la partie centrale du graphique

actions des touches :

[ LEFT ] [ RIGHT ] [ UP ] [ DOWN ] [ TOP ] [ BOTTOM ] [ ( ] [ ) ] [ S ] [ C ] [ M ] [ E ] [ Q ]
identiques à celles de GLCD71 hors numérisation

[ P ] demande d'impression de la zone affichée sur l'écran CMT
[ H ] demande d'affichage du graphique sur l'écran HP71 via le programme GLCD71 pour numérisation

REMARQUES :
Le programme GLCDCMT appelle le sous-programme GPARA du programe GLCD71, aussi ceux qui utilisent l'écran CMT mais n'ont que faire de la numérisation doivent intégrer le sous-programme GPARA dans le programme GLCDCMT.
Le programme GLCDCMT est conçu pour des graphiques situés au-dessus d'une zone réservée aux données du graphique ( fonction,limites,pas etc..). Cette zone fait 45 pixels de hauteur. Si ce n'est pas le cas remplacez ligne 50 : M=(F-45) par M=F.

Dans le coin des lhex vous trouverez les 2 lex : LCDLEX pour LCD$ et TRANSLEX pour LCD$ et TRAN$, à vous de choisir suivant vos besoins.

LE SOURCE DE TRANSLEX

```
LEX   'TRANSLEX'
ID    #5C
MSG   0
POLL  0
```

```
        ENTRY LCD
        CHAR  #F
        ENTRY TRAN
        CHAR  #F
        KEY   'LCD$'
        TOKEN 51
        KEY   'TRAN$'
        TOKEN 52
        ENDTXT
ADHEAD  EQU   #181B7
D=AVMS  EQU   #1A460
EXPR    EQU   #0F23C
MFERR   EQU   #09393
MEMERR  EQU   #0944D
POP1S   EQU   #0BD38
RNDAHX  EQU   #136CB
XXHEAD  EQU   #1A44E
pop     GOSBVL RNDAHX
        D1=D1+ 16
        RTN
        NIBHEX 8822
TRAN    CD0EX
        RSTK=C
        GOSUB pop
        D0=A
        D0=D0- 2
        GOSBVL RNDAHX
        B=A   A
        GOSBVL D=AVMS
        R1=C
        A=C   A
        LC(5) 4084
        C=A-C A
        ?C<=D  A
        GOYES memerr
        LCHEX 1B3C
        D1=D1- 4
        DAT1=C 4
        LCHEX 1B2A6233305701D
        R0=C
        C=0   A
        LC(2) 100
        D=C   A
        P=    1
LOOPT   D1=D1- 12
        C=R0
        DAT1=C M
loopt   D0=D0+ 2
        A=DAT0 B
        D1=D1- 1
        DAT1=A 1
        D1=D1- 1
        DAT1=A P
        C=C-1 B
        GONC  loopt
        CD0EX
        C=C+D A
        D0=C
        B=B-1 B
        GONC  LOOPT
OUT     GOSBVL D=AVMS
        C=RSTK
        D0=C
        ST=0  0
        P=    0
        GOVLNG ADHEAD
memerr  GOVLNG MEMERR
        NIBHEX 811
LCD     CD0EX
        RSTK=C
        GOSUB pop
        D0=A
        GOSBVL D=AVMS
        R1=C
        A=C   A
        C=0   A
        LC(3) 288
        A=A-C A
        LC(3) 1282
        R0=C
        LC(3) 161
        CDEX  A
        ?A<=C A
        GOYES memerr
        CD0EX
        C=C-D A
        D0=C
        D0=D0- 2
        LC(2) 16
LOOP    RSTK=C
        A=R0
        CD0EX
        C=A+C A
        D0=C
        P=    8
loop1   ASL   W
        A=DAT0 1
        D0=D0+ 1
        ASL   W
        A=DAT0 1
        CD0EX
        C=C-D A
        D0=C
        P=P+1
        GONC  loop1
        P=    15
        LCHEX 7
        B=C   S
        C=0   W
loop2   P=    14
        C=C+C W
loop3   A=A+A W
        GONC  decrp
        C=C+1 P
decrp   P=P-1
        GOC   decrb
```

```
    P=P-1
    GOTO  loop3
decrb  B=B-1  S
    GONC  loop2
PILE   D1=D1- 16
    DAT1=C W
    C=RSTK
    C=C-1  B
    GONC  LOOP
    GOTO  OUT

    END
```

Guy Toublanc (276)

---

## VERIFICATION DE LA BOUCLE HP-IL.

Ce LEX vérifie la continuité de la boucle HP-IL. Il retourne un code 1 si c'est le cas, 0 sinon. On l'appelle par le mot-clé IOQ. Lex écrit pratiquement sans connaissance de l'HP-IL. Fonctionne correctement avec des périphériques sans "power down". Je remercie d'avance ceux qui voudront bien m'aider à l'améliorer, pour que le 71 ne se bloque plus lorsqu'un appareil s'éteint sur la boucle..

Jack Elhay, 9/494-498 Mitcham Rd.,
Mitcham, Victoria,3132,AUSTRALIA.

```
    LEX   'IOTEST' *
    ID    #EC     *
    MSG   0       *
    POLL  0       *
    ENTRY SIDY    *
    CHAR  #F      *
    KEY   'IOQ'   *
    TOKEN 55      *
    ENDTXT        *
FNRTN1 EQU   #0F216  *
GETLPs EQU   #01D15  *
PUTC   EQU   #06B1C  *
GET    EQU   #06751  *
ERRORX EQU   #0342C  *
SNAPBF EQU   #2F7F0  *
bLEX   EQU   #00BFC  *
I/OFND EQU   #118BA  *
LEXPIL EQU   #000FF  *
eXWORD EQU   #00023  *
BSERR  EQU   #0939A  *
D1C=R3 EQU   #03047  *
R3=D10 EQU   #03526  *
    NIBHEX 00     *
SIDY   GOSBVL R3=D10  *
    GOSUB JUMPER  *
    CON(5) GETLPs  *
    GOC   Err     *
    LC(4)  #1E00   * charge message à envoyer
    GOSUB JUMPER   * envoie le message
    CON(5) PUTC    * avec PUTC
    GOC   Err      * indicateur carry armé par erreur
    GOSUB JUMPER   * reçoit message de mb
    CON(5) GET     * si bit carry =0, OK,
                   * message dans C(7,0) etc.
    P=    0        * si erreur, #erreur dans P
    GOSBVL D1C=R3  * restaure les pointeurs
    D0=C
    C=0   W        * efface C(W)
    GOC   ZERO     * bit carry armé
                   * si cond."loop broken",
                   * sort un 0
    P=    1        * bit "carry" désarmé = boucle OK,
                   * sort un 1
    CPEX  14       * Attention, astuce!
ZERO   CLRST       * ST(3:0) doit Atre effacé
    GOVLNG FNRTN1  * sortie
Err    GOSUB JUMPER  * Routine d'erreur de l'HP-IL
    CON(5) ERRORX  * point d'entrée non supporté,
* valable seulement avec HP-IL Versions A ou B.
* Le code ci-dessous a été emprunté au fichier JUMPER
* du swapdisc 3 de Mike Markov.
* La table des EQUATE a juste été reportée  avec
* ceux du reste du Lex.
* JUMPER: routine d'accès à la ROM HP-IL.
*
* Routine transparente pour le CPU sauf SB
*
* (code assembleur)
* (prépare CPU pour appel)
* GOSUB JUMPER
* CON(5) (target routine offset)
* (resume assembly code)
************************************************
JUMPER RSTK=C
    CD1EX
    D1=(5) SNAPBF
    DAT1=C A     write D1 @ SNAPBF
    D1=(2) (SNAPBF)+5
    C=RSTK
    DAT1=C W     write C(W) @ SNAPBF + 5
    D1=(4) (SNAPBF)+21
    DAT1=A W     write A(W) @ SNAPBF + 21
    D1=(2) (SNAPBF)+37
    C=B   A
    CPEX  5      save P @ SNAPBF + 42
    P=    6
    C=0   P
    GONC  JUMP05 C:6='0' = bit carry non armé
    C=C-1 P      C:6#'0' = bit carry armé
JUMP05 P=    7
    C=0   P
    C=C-1 P      C:7='9' = mode décimal
    DAT1=C 8     write B(A),P,Carry,mode
                 @ SNAPBF + 37
```

```
    SETHEX      mode hexa pour I/OFND
************
* H présent A(W), B(A), C(W) et D1 disponibles
*
    P=   0
    LC(3) bLEX  find the LEX buffer
    GOSBVL I/OFND
    GONC  JUMP90 si absent, erreur et sortie
************
* LEX buffer found; D1 points to it
* search the LEX buffer for the HPIL ROM entry
*
    LC(2) LEXPIL C(B)=HPIL ROM ID #
    B=C  A
    A=0  A
    A=A+1 A     A(B)=No. de Token
          dans la ROM HP-IL
JUMP10 C=DAT1 6
    ?C=0  B     fin du buffer?
    GOYES JUMP90 oui, sortie
    ?B#C  B     ID correcte?
    GOYES JUMP20 non, essayer suivante
************
* LEX ID correspond; voir si No.token valide
*
    CSR  W
    CSR  A     C 3:0 intervalle No.token
    ?A<C  B     trop petit?
    GOYES JUMP20 oui, continuer
    CSR  A
    CSR  A
    C=C-A B     pas de No. de token dans intervalle
    GONC  JUMP30 dans intervalle
JUMP20 D1=D1+ 11    pas dans intervalle;
          entrée suivante
    GONC  JUMP10 B.E.T.
************
* entrée routine d'erreur
*
JUMP90 LC(4) eXWORD "XWORD Not Found"
    GOVLNG BSERR  pas de retour erreur
************
* LEX table trouvée
*
JUMP30 D1=D1+ 6     pointeur adresse main table
    C=DAT1 A     adresse main table dans C(A)
    B=C  A
************
* lire offset cible dans RSTK
*
    C=RSTK      lire address de l'offset
    D1=C        mettre dans D1
    D1=D1+ 5    step over offset field
    CD1EX
    RSTK=C      place adresse retour finale sur RSTK
    C=DAT1 A    lire offset dans main table
    C=C+B  A    ajouter adresse de la main table
    RSTK=C      placer adresse cible sur STK
************
* restaurer registres & saut à routine
*
    D1=(5) (SNAPBF)+21
    A=DAT1 W    restore A(W)
    D1=D1+ 16   position to carry/mode/B(A) save area
    C=DAT1 8
    B=C  A      restore B(A)
    P=   7      vérifier mode
    C=C+1 P     si bit carry on, mode hex
    GOC  JUMP50
    SETDEC      si carry off, decimal mode
JUMP50 P=   6
    ?C#0  P
    GOYES JUMP60 bit carry on si C:7 # 0
JUMP60 P=C  5     restore P from C:5
    D1=(4) (SNAPBF)+5
    C=DAT1 W    restore C(W)
    D1=(2) SNAPBF
    RSTK=C      mettre C(A) temporairement sur RSTK
    C=DAT1 A
    D1=C        restore D1
    C=RSTK      restore C(A)
    RTN         saut à routine cible
```

## EDITEUR DE GRAPHIQUES.(PROGRAMME PIXEL)

Ce programme a pour objet d'éditer des graphiques sur le HP 71 et sur l'imprimante Thinkjet, et plus particulièrement de faire des impressions de pages graphiques en continu; il faut être patient, étant donné que toutes les routines sont en Basic.

J'ai voulu ce programme aussi simple et facile à comprendre que possible ; je crois y être parvenu.

Le listing du programme contient de nombreux commentaires pour vous expliquer son fonctionnement.

C'est le fruit de nombreuses recherches personnelles qui s'expliquent d'autant mieux lorsque l'on sait qu'à l'affichage, la configuration des bits est faite de 132 barres verticales de 8 pixels chacune, alors que pour l'imprimante, celle-ci se fait par "caractères" horizontaux de 8 pixels, chacun combinables entre eux (allumés/éteints)...

Voici quelques petits exemples, histoire de se mettre dans le bain :

Soit à l'affichage en mode graphique (GDISP$ & Cie) l'allumage de tous les points de la ligne du bas : la variable (chaine) GDISP$ sera donc faite de 132 caractères de code ASCII 128 ; si on veut la même chose avec la ligne du haut, cela donnera une GDISP$ de 132 caractères de code ASCII 1 ; ou encore la ligne du milieu (la 4ème en partant du bas)

donnera une GDISP$ de 132 caractères de code ASCII 16 ; pour en finir, avec un affichage oK tous les pixels de la ligne horizontale du bas sont allumés et idem ceux de la ligne horizontale du haut (lignes 1 et 8), on obtient une GDISP$ de 132 caractères de code ASCII 129... ! Je vous fait grâce des autres combinaisons possibles...!

C'est dans ce genre de situation que l'on aimerait bien connecter un gazogène à son HP 71 préfèré...! (inhumain, quoique...).

Il est à noter, qu'en tout état de cause, au stade primaire un pixel à l'affichage ne peut (en principe, donc) faire l'objet d'une impression distincte. Pour mémoire, en ce qui concerne la Thinkjet j'ai retenu deux codes de caractères distincts : le caractère de code 0 (n'imprime rien, marque un blanc) et le caractère de code 255 (imprime huit points joints à l'horizontale) ; de plus on peut, à l'impression, moduler la dimension verticale du caractère, c'est à dire soit imprimer une suite de simples traits horizontaux, ou de "pavés", ou bien encore de barres verticales : ceci fait qu'un même graphique à l'écran peut avoir un nombre infini de représentations à l'impression ! Dans le cas d'impressions multifichiers en continu, les graphiques seront imprimés dans l'ordre et le degré de verticalite de votre choix sans espaces entre eux. De même, si vous faites des impressions de fichiers uns à uns, il n'y aura pas d'espaces entre eux (si vous n'emettez pas d'avance ligne/page) ; de cette façon, vous pouvez définir au coup par coup le degré de verticalité de votre impression...!.

La notion d'image graphique contrôlable devient donc une réalité sur le HP 71 !.

Chaque fichier graphique est un fichier DATA de 2120 octets ; si vous disposez d'un espace RAM de 320 K (programme déduit), vous pouvez donc disposer de 150 fichiers graphiques, ou bien encore de 60 fichiers graphiques par cassette, et, si vous préfèrez de 330 fichiers graphiques par unité de disque 9114...!. Sans préjuger de l'adressage étendu HP-IL et des 961 périphériques que l'on peut y mettre !.

De quoi imprimer en continu plusieurs centaines de pages graphiques, ce qui je crois, reste un record à battre.

Il faut peut-être se modérer dans ses ambitions compte tenu de la lenteur du Basic...

Mais je n'ai pas pour autant dit mon dernier mot, car je vous prépare actuellement une transposition en Basic décompilé (fichier BIN) qui résoudra le problème une fois pour toutes !. De quoi donner des crampes aux utilisateurs d'IBM-PC et autres MS/DOS (à moëlle) !.

Il est aisément concevable d'associer ce programme à des fonctions mathématiques/statistiques moyennant une légère modification des structures de celui-ci.

Voici maintenant le mode d'emploi de ce programme :

Vous le lancez et il apparait un masque de saisie demandant la largeur de texte en affichant la valeur par défaut de 80 :

Ce paramètre à renseigner n'est autre que le nombre de caractères par ligne que vous avez sélectionné sur votre imprimante Thinkjet ; cette imprimante vous permet d'imprimer diffèrentes grosseurs de caractères et selon le cas, vous pouvez imprimer en 40,71,80,142 caractères/ligne ; ce paramètre est donc indispensable pour alterner correctement entre le mode texte et le mode graphique ; aux curieux, je signale une possibilité non utilisée ici : il est notement possible d'imprimer du texte superposé à du graphique ; je ne manquerai pas d'y revenir.

Ce après quoi il apparait un autre masque vous demandant (répondre par [O] pour oui ou [N] pour "non") et avec pour réponse par défaut [O] ("oui") si vous voulez l'impression de matrice(s) graphiques, puis un autre masque vous en demandant le nombre avec 1 comme valeur par défaut ; cette routine a pour but d'imprimer un ou des documents représentant schématiquement l'écran de votre HP 71 avec indications des rangs verticaux et horizontaux ; chaque case correspond à un point à allumer sur l'écran ; après avoir décidé du ou des motifs de votre dessin, vous remplissez au crayon les cases que voulez allumer... Bien entendu, vous pouvez définir plusieurs écrans successifs pour un grand dessin ; de cette façon, il vous sera très facile, lors de la session, d'enregistrer vos coordonnés de points. L'imprimante vous imprime ce genre de document, vous évitant ainsi la tâche fastidieuse de faire votre/vos croquis à la main (que de temps gagné !).

Ensuite, il apparait un masque vous demandant si voulez retravailler sur un fichier déjà crée avec pour réponse par défaut [N] ("non") : si tel est le cas répondez en tapant le nom du fichier, sinon tapez [N] (non). Il est à noter que si vous répondez par non, cela entraine l'élaboration d'une variable "vierge" d'impression durant laquelle votre HP 71 émet des beeps. Si, par mégarde, vous répondez par "oui", il y a un beep et le masque de saisie réapparaît, vous demandez une réponse correcte. Cette opération ne peut se faire que fichier par fichier îtant entendu qu'à ce stadî vous ne faites que des créations ou des mises à jour.

Ensuite, vient un autre masque vous demandant (répondre par [O] "oui" ou [N] "non") avec comme réponse par défaut [O] ("oui") si voulez l'impression des coordonnées du fichier en cours d'édition (s'il ne s'agit pas d'une création) pixels par pixels (points) ; si vous répondez affirmativement, les coordonnées des 1056 pixels seront imprimées en commençant par celles du pixel le plus en bas à gauche ; il est judicieux de combiner cela avec votre matrice précedement évoquée ; comme cela, vous savez en permanence oK vous en êtes : simple, n'est-ce pas ?. Là encore, cette opération ne peut se faire que fichier par fichier pour

les mêmes raisons que précedement. Durant l'impression de ces coordonnées, le nom du fichier et son image graphique apparaissent alternativement, eficacité oblige. Bien entendu, le nom du fichier est imprimé en entête du document.

Ensuite, le graphique apparait à l'écran puis un masque apparait à son tour, vous demandant si vous voulez l'impression du graphique (répondre par [O] "oui" ou [N] "non") avec [O] ("oui") comme réponse par défaut ; si vous répondez par l'affirmative, un autre masque vous demande l'échelle verticale du point graphique à l'impression ; répondre par un entier positif supérieur ou égal à 1 (une échelle de 1 se traduit à l'impression par un point d'environ 1mm d'épaisseur verticale). Durant cette routine, le nom du fichier et sa représentation graphique apparaissent alternativement à l'écran à titre d'aide mémoire ; bien entendu, cette routine est d'autant plus longue que l'échelle verticale du point est élevée... Cette séquence n'effectue pas d'avance-page/ligne en début comme en fin, vous permettant ainsi de superposer plusieurs graphiques sans espaces entre eux si vous le désirez. Bien entendu, cette séquence ne peut se faire que fichier par fichier pour les mêmes raisons déja évoquées plus haut.
ensuite, apparaît un autre masque ("DESSIN GRAPHIQUE ?") vous demandant si vous voulez entrer en session graphique pour allumer des points ou en éteindre selon le/les cas avec comme réponse par défaut [O] ("oui") ; si vous répondez par l'affirmative, l'affichage du graphique apparait à l'écran, puis l'on vous demande si vous voulez allumer un pixel (répondre par [O] pour "oui" ou par [N] pour "non") ; vous répondrez par l'affirmative pour allumer un pixel et par la négative pour le cas contraire ; dans chaque cas vous aurez un masque vous demandant tout d'abord le rang vertical (1 a 8) puis le rang horizontal (1 a 132). Puis vient un autre masque vous demandant la suite ("SUITE ? ") avec [N] ("non") comme valeur par défaut. Si vous répondez [O] ("oui"), réapparaitront alors les masques de manipulations de pixels (pixels par pixels)... et ainsi de suite jusqu'à ceque vous répondiez [N] ("non")...
J'oubliais presque un détail : à chaque fois que vous manipulez un pixel, l'ancien affichage apparait, puis vous voyez le pixel manipulé s'allumer ou s'éteindre selon le cas, puis le nouvel affichage et enfin la valeur de code ASCII du caractIre de la chaine GDISP$ concernée. Simultanément, la variable d'impression est mise à jour...
Ensuite, apparait un masque vous demandant si vous voulez imprimer le graphique que vous venez d'éditer, avec comme réponse par défaut [O] ("oui") (répondre par [O] "oui" ou par [N] "non selon le cas)... Mêmes remarques que précèdent...
Il est à noter qu'à chaque séquence d'impression graphique, le flag(-60) est armé, ce qui se concrétise par l'apparition à droite de l'écran du gros point entouré de parenthèses...
Ensuite, apparait un masque vous demandant si vous voulez créer un fichier à partir du graphique que vous venez d'éditer (suite à création, mise à jour...) ; dans l'affirmative, vous tapez le nom du fichier où stocker votre travail (et non pas en tapant [O] ("oui")) ; attention toutefois ; si vous avez déja un fichier sous un même nom, l'ancien est purgé et le nouveau prend alors sa place !. Dans la négative, répondre par [N] ("non"). Comme déja évoqué, il s'agit de fichiers DATA de 2120 octets chacun contenant deux enregistrements, le premier pour l'affichage et le second pour l'impression.
ensuite apparait un masque vous demandant si vous voulez procéder à l'impression en continu de plusieurs fichiers en continu (impression multifichiers) (répondre par [O] pour "oui" et par [N] pour "non") ; dans l'affirmative, apparait un masque vous demandant le nombre de fichier à imprimer en continu ; répondre par un entier positif supérieur ou égal à 1 ; après apparaissent une série de masques (autant que de fichiers à imprimer) oK figurent à chaque fois le numéro d'ordre ; cette série de masques disparait une fois le nombre de fichiers atteint. Puis apparait un masque vous demandant l'échelle verticale d'impression (déjà évoqué) ; dans ce cas l'échelle sera la même pour tous les fichiers à imprimer !. Bien entendu, le nom du fichier en cours d'impression et son graphique apparaissent alternativement à l'écran ainsi qu'un message demadant d'attendre la fin des opérations. Selon la nature des paramètres fournis, la durée de cette séquence peut varier considérablement ; il convient donc de prendre toutes dispositions utiles dans ce cas (alimentation secteur, heure de départ de la session et emploi du temps personnel...).
Enfin, apparaît un masque vous demandant si vous voulez procéder à une autre session d'édition graphique, avec [N] ("non") comme réponse par défaut ; si vous répondez négativement, vous terminez le programme ; dans l'affirmative [O] ("oui"), vous revenez en début de programme, et ainsi de suite, l'ensemble effectuant une boucle parfaite, il vous est donc inutile de relancer le programme à chaque fois...
Ceci est mon premier (et fait entièrement par moi) programme sur HP 71. L'essayer, c'est l'adopter !. Bonne et heureuse programmation à tous !.

Jean Paul NALIN

PROGRAMME ADR

```
0100 DIM D$(3)[95],A$[95],B$[230],C$(5) @ OPTION BASE 1
0101 Z$='------------------------------'
0103 PRINT USING "40x,41A";Z$ @ PRINT
0105 B$=XFN82097("#,40x,b'&k3S',8a,x,20a")
0115 INPUT 'TITRE,PREN.&NOM: ';X$,Y$
0120 XWORD82098
0125 PRINT A$ @ PRINT @ D$(1)=A$
0126 PRINT USING "40x,41A";Z$ @ PRINT
0130 B$=XFN82097("#,40x,b'&k3S',3d,b'&k0S',8a,b'&k3S',20a")
0132 INPUT 'No.,Rue,Nom';A,X$,Y$
0135 XWORD82098
0140 PRINT A$ @ PRINT @ D$(2)=A$
0145 B$=XFN82097("#,40x,b'&k3S',5d,2x,20a")
0147 INPUT 'CODE,LOCALITE';A,X$
0150 XWORD82098
0160 PRINT A$ @ D$(3)=A$ @ STOP
0165 PRINT
0170 PRINT USING "40x,41A";Z$ @ PRINT
0180 PRINT D$(1) @ PRINT
0185 PRINT USING "40x,41A";Z$ @ PRINT
0190 PRINT D$(2) @ PRINT
0200 PRINT D$(3) @ PRINT
0210 E$=DATE$[7,8]&'.'&DATE$[4,5]&'.'&DATE$[1,2]
0220 PRINT USING "30x,8a";E$ @ PRINT
```

**********

PROGRAMME ADNOM

```
0100 DIM D$(3)[80],A$[80],B$[200],C$(5) @ OPTION BASE 1
0105 B$=IMAGE$("#,40x,b'&k0S',8a,3x,b'&k3S',14A")
0110 DATA Monsieur,Aurel  ROTTMAN,rue du,Bas HUET,St. GERMAIN en LAYE
0115 READ C$
0120 PRINT TO A$ USING B$;27,C$(1),27,C$(2)
0125 PRINT A$ @ PRINT @ D$(1)=A$
0130 B$=IMAGE$("#,40x,b'&k3S',3a,2x,b'&k0S',6a,2x,b'&k3S',8a")
0135 PRINT TO A$ USING B$;27,'47,',27,C$(3),27,C$(4)
0140 PRINT A$ @ PRINT @ D$(2)=A$
0145 B$=IMAGE$("#,40x,b'&k3S',5a,2x,19a")
0150 PRINT TO A$ USING B$;27,'78100',C$(5)
0155 B$=IMAGE$("#,40x,b'&k3S',5a,2x,19a")
0160 PRINT A$ @ D$(3)=A$ @ STOP
0170 PRINT
0180 PRINT D$(1) @ PRINT
0190 PRINT D$(2) @ PRINT
0200 PRINT D$(3) @ PRINT
```

**********

PROGRAMME ANIM

```
0010 ! ANIM 25.12.1986
0020 DESTROY ALL
0030 DIM B$[180]
0040 M$=CHR$(192)&CHR$(90)&CHR$(83)&CHR$(202)&CHR$(136)
0050 N$=CHR$(0)&CHR$(0)&CHR$(2)&CHR$(255)&CHR$(178)
0060 P$=CHR$(64)&CHR$(160)&CHR$(64)&CHR$(0)&CHR$(0)
0070 Q$=CHR$(255)&CHR$(129)&CHR$(145)&CHR$(145)&CHR$(231)
0080 R$=CHR$(255)&CHR$(129)&CHR$(129)&CHR$(129)&CHR$(255)
0090 S$=CHR$(255)&CHR$(17)&CHR$(17)&CHR$(17)&CHR$(255)
0100 T$=CHR$(255)&CHR$(128)&CHR$(128)&CHR$(128)&CHR$(128)
0110 FOR I=1 TO 40
0120 IF I=21 THEN P$=CHR$(0)&CHR$(32)&CHR$(80)&CHR$(32)&CHR$(0)
0130 IF I=22 THEN P$=CHR$(0)&CHR$(0)&CHR$(16)&CHR$(40)&CHR$(16)
0140 IF I>=24 THEN P$=CHR$(0)&CHR$(2)&CHR$(5)&CHR$(2)&CHR$(0)
0150 IF I=25 THEN M$=CHR$(192)&CHR$(88)&CHR$(61)&CHR$(202)&CHR$(141)
0160 A=.08
0170 IF I>=4 THEN A=.05
0180 IF I>=6 THEN A=.02
0190 IF I>26 THEN A=0
0200 IF I=40 THEN A=.45
0210 IF I#40 THEN GDISP B$&N$
0220 WAIT A
0230 GDISP B$&M$&P$
0240 WAIT A
0250 IF I>25 THEN GDISP B$
0260 WAIT A
0270 IF I<5 THEN K=5
0280 IF I>5 THEN K=10
0290 IF I>10 THEN K=20
0300 FOR J=1 TO K STEP 8
0310 B$=B$&CHR$(0)
0320 IF B$[180]#"" THEN STOP
0330 WAIT A
0340 NEXT J
0350 NEXT I
0360 GDISP B$&Q$
0370 WAIT A
0380 GDISP B$&Q$&R$
0390 WAIT A
0400 GDISP B$&Q$&R$&S$
0410 WAIT A
0420 GDISP B$&Q$&R$&S$&T$
```

**********

PROGRAMME DECBIN

```
0010 DESTROY N,N$ @ DEF FNT(N,J)=INT(2*FP(N/2^(J+1)))
0020 N$=""
0030 INPUT "NBR DECIMAL ? ","10";N
0040 FOR J=7 TO 0 STEP -1
```

```
0050 N$=N$&STR$(FNT(N,J))[1,1]
0060 NEXT J
0070 DISP "DEC";N;"BIN ";N$
0080 DESTROY N,N$
```

**********

PROGRAMME PIXEL

```
0002 DESTROY A$,B$,C$,M$,M1$,M2$,R$,R1$,R2$,R3$,R4$,P$,C,E9,M,M1,X,Y,Z,Z1 @ STD
0004 DEF FNP(Y,C)=INT(2*FP(Y/2^(C+1)))
0006 DIM A$[132],C$[1056] @ CFLAG 0,1,3
0008 INPUT "LARGEUR DE TEXTE ? ","80";Z @ Z1=CEIL((Z-28)/2) @ IF Z=40 THEN Z1=1
0010 IF Z<40 OR Z>142 THEN BEEP @ GOTO 8
0012 LINPUT "IMPRESSION MATRICE ? ","O";R3$
0014 IF R3$="O" THEN GOTO 16 ELSE IF R3$="N" THEN GOTO 42 ELSE GOTO 12
0016 INPUT "NOMBRE DE MATRICES ? ","1";M
0018 FOR I=1 TO M @ PRINT CHR$(12)
0020 PRINT TAB(Z1);"   MATRICE GRAPHIQUE   " @ PRINT TAB(Z1);"########################"
0022 PRINT TAB(Z1);" 1  2  3  4  5  6  7  8"
0024 PRINT CHR$(27)&"&l8D" @ PRINT TAB(Z1);"########################" @ FOR E=1 TO 132
0026 DESTROY E1$ @ IF E<10 THEN E1$="00"&STR$(E)
0028 IF E<100 AND E>=10 THEN E1$="0"&STR$(E)
0030 IF E>=100 THEN E1$=STR$(E)
0032 PRINT TAB(Z1);CHR$(27)&"&dD"&"[]:[]:[]:[]:[]:[]:[]:[]"&CHR$(27)&"&d@"&"  "&E1$
0034 NEXT E @ PRINT TAB(Z1);"########################" @ PRINT CHR$(27)&"&l6D" @ DESTROY E1$
0036 PRINT TAB(Z1);" 1  2  3  4  5  6  7  8" @ PRINT CHR$(12)
0038 PRINT CHR$(12)
0040 NEXT I
0042 LINPUT "REPRISE FICHIER ? ","N";R2$
0044 IF R2$="N" THEN DISP "ATTENDEZ..." @ GOTO 46 ELSE GOTO 48
0046 FOR P=1 TO 1056 @ BEEP 50,.02 @ C$[P,P]=CHR$(0) @ NEXT P @ GOTO 98
0048 IF R2$="O" THEN BEEP @ DISP "NOM DU FICHIER ? " @ WAIT 1 @ GOTO 42
0050 IF R2$#"N" AND R2$#"O" THEN ASSIGN #1 TO R2$ @ READ #1,0;C$ @ READ #1,1;A$
0052 IF R2$#"N" AND R2$#"O" THEN ASSIGN #1 TO *
0054 IF R2$#"N" AND R2$#"O" THEN DISP "FICHIER "&R2$ @ WAIT 1 @ GDISP A$ @ WAIT 5
0056 LINPUT "LISTE COORDONNEES ? ","O";P$
0058 IF P$#"O" AND P$#"N" THEN BEEP @ GOTO 56
0060 IF P$="O" AND R2$#"N" AND R2$#"O" THEN GOSUB 'XYCORD' @ SFLAG 1 @ GOTO 180
0062 IF P$="O" AND R2$="N" THEN BEEP @ DISP "FICHIER CREE VIDE" @ WAIT 1 @ GOTO 106
0064 IF P$="N" AND R2$#"N" AND R2$#"O" THEN SFLAG 1 @ GOTO 180
0066 IF P$="N" AND R2$="N" THEN GOTO 98
0068 'XYCORD':
0070 PRINT CHR$(12) @ X=1 @ N1=925 @ N2=1056 @ E1$=CHR$(27)&"&dD" @ E2$=CHR$(27)&"&d@"
0072 PRINT TAB(Z/4);E1$&"COORDONNEES DU FICHIER "&R2$&E2$&CHR$(10)&CHR$(13)&CHR$(13)
0074 DIM C5$[132] @ C5$=C$[N1,N2] @ GDISP A$
0076 FOR I=1 TO 132
0078 IMAGE "[ Rang vertical vvv , Rang horizontal hhh ] PIXEL "3Z,","3Z" ETEINT"
0080 IMAGE K,"[ Rang vertical vvv , Rang horizontal hhh ] PIXEL "3Z,","3Z" ALLUME",K
0082 IF C5$[I,I]=CHR$(0) THEN PRINT USING 78;X,I
0084 IF C5$[I,I]=CHR$(255) THEN PRINT USING 80;E1$,X,I,E2$
0086 NEXT I
0088 X=X+1 @ N1=N1-132 @ N2=N2-132
0090 IF X<9 THEN DISP "FICHIER ";R2$ @ WAIT 2 @ GOTO 74
0092 PRINT CHR$(12) @ PRINT CHR$(12)
```

```
0094 DESTROY X,N1,N2,C5$,E1$,E2$
0096 RETURN
0098 GDISP A$ @ WAIT 1 @ LINPUT "IMPRESSION ? ","N";R$
0100 IF R$#"O" AND R$#"N" THEN BEEP @ GOTO 98
0102 IF R$="O" THEN GOSUB "PRGRAPH"
0104 DESTROY R$
0106 LINPUT "DESSIN GRAPHIQUE ? ","O";R$ @ IF R$#"O" AND R$#"N" THEN BEEP @ GOTO 106
0108 IF R$="N" THEN GOTO 190
0110 LINPUT "ALLUMER PIXEL ? ","O";R$
0112 IF R$="O" THEN GOTO 126
0114 IF R$="N" THEN GOTO 118
0116 IF R$#"O" AND R$#"N" THEN BEEP @ GOTO 110
0118 INPUT "RANG VERTICAL ? ","1";Y @ IF Y<1 OR Y>8 THEN BEEP @ GOTO 118
0120 INPUT "RANG HORIZONTAL ? ","1";X @ IF X<1 OR X>132 THEN BEEP @ GOTO 120
0122 I=0
0124 GOTO 132
0126 INPUT "RANG VERTICAL ? ","1";Y @ IF Y<1 OR Y>8 THEN BEEP @ GOTO 126
0128 INPUT "RANG HORIZONTAL ? ","1";X @ IF X<1 OR X>132 THEN BEEP @ GOTO 128
0130 I=1
0132 IF I=0 THEN DISP "PIXEL E {";Y;":";X;"}" @ BEEP @ WAIT 1
0134 IF I=0 AND Y=1 THEN C$[924+X,924+X]=CHR$(0)
0136 IF I=0 AND Y=2 THEN C$[792+X,792+X]=CHR$(0)
0138 IF I=0 AND Y=3 THEN C$[660+X,660+X]=CHR$(0)
0140 IF I=0 AND Y=4 THEN C$[528+X,528+X]=CHR$(0)
0142 IF I=0 AND Y=5 THEN C$[396+X,396+X]=CHR$(0)
0144 IF I=0 AND Y=6 THEN C$[264+X,264+X]=CHR$(0)
0146 IF I=0 AND Y=7 THEN C$[132+X,132+X]=CHR$(0)
0148 IF I=0 AND Y=8 THEN C$[X,X]=CHR$(0)
0150 IF I=1 THEN DISP "PIXEL A {";Y;":";X;"}" @ BEEP @ WAIT 1
0152 IF I=1 AND Y=1 THEN C$[924+X,924+X]=CHR$(255)
0154 IF I=1 AND Y=2 THEN C$[792+X,792+X]=CHR$(255)
0156 IF I=1 AND Y=3 THEN C$[660+X,660+X]=CHR$(255)
0158 IF I=1 AND Y=4 THEN C$[528+X,528+X]=CHR$(255)
0160 IF I=1 AND Y=5 THEN C$[396+X,396+X]=CHR$(255)
0162 IF I=1 AND Y=6 THEN C$[264+X,264+X]=CHR$(255)
0164 IF I=1 AND Y=7 THEN C$[132+X,132+X]=CHR$(255)
0166 IF I=1 AND Y=8 THEN C$[X,X]=CHR$(255)
0168 DISP CHR$(27)&"J" @ IF NOT FLAG(0) THEN DIM B$[132]
0170 IF NOT FLAG(0) THEN A$=GDISP$ @ C=NUM(A$[X])
0172 IF FLAG(0) THEN GDISP B$ @ A$=GDISP$ @ C=NUM(A$[X])
0174 IF I AND NOT FNP(C,8-Y) THEN C=C+2^(8-Y)
0176 IF NOT I AND FNP(C,8-Y) THEN C=C-2^(8-Y)
0178 A$[X,X]=CHR$(C) @ B$=A$ @ DISP "CHR RANG "&STR$(X)&" = "&STR$(C) @ WAIT 1
0180 GDISP A$ @ WAIT 5 @ BEEP @ IF FLAG(1) THEN DIM B$[132] @ B$=A$ @ CFLAG 1
0182 LINPUT "SUITE ? ","N";R1$
0184 IF R1$="N" THEN GOTO 190
0186 IF R1$="O" THEN SFLAG 0 @ GOTO 98
0188 IF R1$#"O" AND R1$#"N" THEN BEEP @ GOTO 182
0190 LINPUT "IMPRIMER GRAPHIQUE ? ","O";R4$ @ IF R4$="O" THEN GOSUB 'PRGRAPH' @ GOTO 230
0192 IF R4$="N" THEN GOTO 230 ELSE IF R4$#"O" AND R4$#"N" THEN BEEP @ GOTO 190
0194 'PRGRAPH':
0196 IF NOT FLAG(3) THEN INPUT "ECHELLE VERTICALE ? ","6";E9
0198 IF E9<1 AND NOT FLAG(3) THEN BEEP @ GOTO 196
0200 DISP "ATTENDEZ..." @ POKE "2F6E7","8" @ PWIDTH INF @ ENDLINE "" @ PRINT CHR$(27)&"*r1280sA"
0202 E=1 @ N1=1 @ N2=132 @ C2$="" @ FOR E0=1 TO 14 @ C2$=CHR$(0)&C2$ @ NEXT E0
0204 BEEP @ DIM C1$[146] @ C1$=C2$&C$[N1,N2] @ N1=N1+132 @ N2=N2+132
0206 BEEP @ IF FLAG(3) THEN DISP "FICHIER "&M$(I)&"..." @ WAIT 1
0208 BEEP @ IF FLAG(3) THEN DISP "EN COURS D'IMPRESSION" @ WAIT 1 @ GDISP A$
```

```
0210 BEEP @ IF NOT FLAG(3) THEN GDISP A$
0212 FOR E2=1 TO E9 @ PRINT CHR$(27)&"*b146W" @ FOR E3=1 TO 146 @ PRINT CHR$(NUM(C1$[E3,E3]))
0214 NEXT E3 @ NEXT E2 @ E=E+1 @ IF E<9 THEN DISP "ATTENDEZ..." @ WAIT 1 @ GOTO 204
0216 BEEP
0218 BEEP
0220 BEEP
0222 PRINT CHR$(27)&"*rB"
0224 PWIDTH Z @ ENDLINE CHR$(10)&CHR$(13) @ POKE "2F6E7","0"
0226 DESTROY C1$,C2$,E,N1,N2 @ IF NOT FLAG(3) THEN DESTROY E9
0228 RETURN
0230 LINPUT "CREER FICHIER ? ";R4$ @ IF R4$="N" THEN GOTO "MGRAPH"
0232 IF R4$#"N" AND R4$#"O" THEN DISP "ATTENDEZ..." @ GOTO 236
0234 IF R2$="O" THEN BEEP @ DISP "NOM DU FICHIER ? " @ WAIT 1 @ GOTO 230
0236 SFLAG -1 @ PURGE R4$ @ CREATE DATA R4$,2,1060 @ CFLAG -1
0238 ASSIGN #1 TO R4$ @ PRINT #1,0;C$ @ PRINT #1,1;A$
0240 ASSIGN #1 TO *
0242 'MGRAPH':
0244 LINPUT "IMPR MULTIFICHIERS ? ";M1$
0246 IF M1$#"O" AND M1$#"N" THEN BEEP @ GOTO 244
0248 IF M1$="N" THEN GOTO 274
0250 IF M1$="O" THEN SFLAG 3 @ INPUT "NBRE DE FICHIERS ? ","1";M1
0252 OPTION BASE 1 @ DIM M$(M1)
0254 FOR I=1 TO M1 @ ON ERROR GOTO 256
0256 DISP STR$(I)&" " @ WINDOW 4 @ LINPUT "EME FICHIER ? ";M$(I) @ WINDOW 1 @ OFF ERROR
0258 IF M$(I)="" OR LEN(M$(I))>8 THEN BEEP @ GOTO 256
0260 NEXT I
0262 INPUT "ECHELLE VERTICALE ? ","6";E9
0264 IF E9<1 THEN BEEP @ GOTO 262
0266 FOR I=1 TO M1 @ DIM C$[1056],A$[132]
0268 ASSIGN #1 TO M$(I) @ READ #1,0;C$ @ READ #1,1;A$ @ ASSIGN #1 TO * @ GOSUB "PRGRAPH"
0270 NEXT I
0272 DESTROY E9 @ CFLAG 3
0274 LINPUT "AUTRE JOB ? ","N";M2$
0276 IF M2$#"O" AND M2$#"N" THEN BEEP @ GOTO 274
0278 IF M2$="O" THEN GOTO 2
0280 IF M2$="N" THEN GOTO 282
0282 DESTROY A$,B$,C$,M$,M1$,M2$,R$,R1$,R2$,R3$,R4$,P$,C,M,M1,X,Y,Z,Z1 @ CFLAG 0,1,3 @ FIX 2


0001 REM PROGRAMME EDITEUR DE GRAPHIQUES SUR IMPRIMANTE THINKJET (C) JEAN-PAUL NALIN 1990
0003 REM LIBERATION DES VARIABLES UTILISEES DANS LE PROGRAMME
0005 REM IL S'AGIT LA DE LA FONCTION PERMETTANT "L'INTERCONNEXION" ENTRE L'AFFICHAGE ET...
0007 REM L'IMPRIMANTE. A$ EST LA VARIABLE D'ECRAN ET C$ POUR L'IMPRESSION SUR LA THINKJET
0009 REM POUR MEMORISER LE NOMBRE DE CARACTERES PAR LIGNES (40,71,80,142) SUITE A CHOIX PREALABLE
0011 REM CELA EST NECESSAIRE POUR PASSER DU MODE TEXTE AU MODE GRAPHIQUE ET VICE VERSA
0013 REM CETTE OPTION SERT A L'IMPRESSION DE BROUILLON(S) DE TRAVAIL, CE QUI EST MOINS FASTIDIEUX
0015 REM QUE DE LES FAIRE A LA MAIN : IL FAUT UN BROUILLON PAR ECRAN A DESSINER
0017 REM ON A LA FACULTE D'EN IMPRIMER AUTANT QUE NECESSAIRE A CE STADE
0019 REM EDITION FORMATTEE, BIEN ENTENDU
0021 REM ENTETE DU DOCUMENT
0023 REM IDEM
0025 REM IMPRESSION COMME EVOQUE
0027 REM IDEM ET FORMATTAGE DU NUMERO DE RANG
0029 REM IDEM
0031 REM IDEM
0033 REM IMPRESSION LIGNE A LIGNE DES RANGS VERTICAUX
0035 REM FIN DE DOCUMENT
0037 REM IDEM
```

```
0039 REM EN OUTRE, EFFECTUE UNE FIN DE PAGE POUR EVITER TOUTE CONFUSION
0041 REM FIN DE CETTE PREMIERE ROUTINE
0043 REM CHOIX ENTRE UN FICHIER DEJA CREE (FICHIER DATA) QUE L'ON PEUT DE CE FAIT, SOIT MODIFIER
0045 REM SOIT IMPRIMER TEL QUE OUBIEN ENCORE POUR EN IMPRIMER LES COORDONNEES
0047 REM CAS OU L'ON VEUT EN CREER UN NOUVEAU INITIALISATION DE LA VARIABLE D'IMPREESION
0049 REM CAS OU L'ON REPOND OUI AU LIEU DE DONNER LE NOM DU FICHIER
0051 REM LECTURE DU FICHIER DATA QUE L'ON REPREND LEQUEL CONTIENT LA VARIABLE D'AFFICHAGE...
0053 REM ET CELLE D'IMPRESSION
0055 REM LE NOM DU FICHIER REAPPARAIT A L'ECRAN, PUIS UNE SECONDE APRES L'AFFICHAGE GRAPHIQUE
0057 REM POUR IMPRIMER LES COORDONNEES POINT/POINT DU FICHIER REPRIS (INUTILE SI CREE, CAR VIDE)
0059 REM CAS OU L'ON NE REPOND PAS CORRECTEMENT A LA QUESTION
0061 REM SI REPONSE AFFIRMATIVE, EXECUTE L'IMPRESSION DES COORDONNEES DU FICHIER REPRIS
0063 REM CAS OU IL S'AGIT D'UN FICHIER CREE DONC VIDE ; INUTILE D'IMPRIMER ; SAUT AUX OPTIONS
0065 REM SI ON NE VEUT PAS LA LISTE ET QUE L'ON A REPRIS UN FICHIER, ARME FLAG 1 POUR OPTIONS
0067 REM SI ON NE VEUT PAS LA LISTE ET QUE L'ON A CREE UN FICHIER, VA EN SESSION GRAPHIQUE
0069 REM C'EST LA SOUS ROUTINE D'IMPRESSION DES COORDONNEES POINTS PAR POINTS
0071 REM L'ECRAN DU HP 71 COMPREND 8 LIGNES DE 132 CARACTERES => 1056 POINTS
0073 REM EN TETE DU DOCUMENT
0075 REM C5$ EST UNE SUBDIVISION DE C$ REPRENANT CHAQUE LIGNE HORIZONTALE D'ECRAN
0077 REM FORMATTAGE (ECRAN DE BAS EN HAUT)
0079 REM IDEM
0081 REM LE PIXEL (1,1) OU (Y,X) EST CELUI EN BAS A GAUCHE DE L'AFFICHAGE
0083 REM FORMATTAGE
0085 REM IDEM
0087 REM POUR DES RAISONS TECHNIQUES EVIDENTES, ON COMMENCE PAR LE BAS DE L'AFFICHAGE
0089 REM PASSAGE A LA LIGNE SUIVANTE
0091 REM AFFICHE LE NOM DU FICHIER REPRIS ET SON GRAPHIQUE CORRESPONDANT DURANT L'IMPRESSION
0093 REM FIN DE DOCUMENT : AVANCE DE DEUX PAGES POUR COMMODITE
0095 REM FIN DE ROUTINE
0097 REM IDEM ET RETOUR A LA ROUTINE APPELANTE POUR SUITE DU TRAVAIL
0099 REM AFFICHE LE GRAPHIQUE ET DEMANDE SI L'ON VEUT L'IMPRIMER
0101 REM CAS OU L'ON NE REPOND PAS CORRECTEMENT A LA QUESTION
0103 REM SAUT A LA ROUTINE D'IMPRESSION GRAPHIQUE ET AVANCE AGE
0105 REM LIBERATION DE LA VARIABLE R$
0107 REM SELECTION POUR APPEL A LA ROUTINE POUR ALLUMER OU ETEINDRE DES PIXELS A L'ECRAN
0109 REM SI ON NE VEUT PAS SAUT AUX OPTIONS RESTANTES
0111 REM REPONDRE O SI AFFIRMATIF OU N SI NEGATIF
0113 REM SI AFFIRMATIF, VA A LA LIGNE DEMANDANT LES COORDONNEES DU PIXEL A ALLUMER
0115 REM SI NEGATIF, VA A LA LIGNE DEMANDANT LES COORDONNEES DU PIXEL A ETEINDRE
0117 REM CAS OU L'ON NE REPOND PAS CORRECTEMENT A LA QUESTION
0119 REM D'UNE PART...
0121 REM ET D'AUTREPART
0123 REM LA VARIABLE I ETANT A ZERO, LE PIXEL EST CONC ETEINT
0125 REM POUR INTERCONNEXION AVEC LA VARIABLE D'IMPRESSION
0127 REM D'UNE PART... (DE BAS EN HAUT)
0129 REM ET D'AUTREPART (DE GAUCHE A DROITE)
0131 REM LA VARIABLE I ETANT A 1, LE PIXEL EST DONC ALLUME
0133 REM AFFICHE "PIXEL E (ETEINT)"...ETC
0135 REM TRANSPOSITION ENTRE VARIABLE D'AFFICHAGE ET D'IMPRESSION (1ERE LIGNE HORIZONTALE)
0137 REM IDEM (2EME LGNE HORIZONTALE)
0139 REM IDEM (3EME LIGNE HORIZONTALE)
0141 REM IDEM (4EME LIGNE HORIZNTALE)
0143 REM IDEM (5EME LIGNE HORIZONTALE)
0145 REM IDEM (6EME LIGNE HORIZONTALE)
0147 REM IDEM (7EME LIGNE HORIZONTALE)
0149 REM IDEM (8EME LIGNE HORIZONTALE)
0151 REM TRANSPOSITION ENTRE VARIABLE D'AFFICHAGE ET D'IMPRESSION
0153 REM IDEM
```

```
0155 REM IDEM
0157 REM IDEM
0159 REM IDEM
0161 REM IDEM
0163 REM IDEM
0165 REM IDEM
0167 REM IDEM
0169 REM VIDE L'AFFICHAGE ET SI DEUXIEME INTERVENTION OU PLUS SUR UN PIXEL INITIALISE
0171 REM LA VARIABLE RELAIS D'AFFICHAGE, L'UTILISE ET MET A JOUR LA VARIABLE D'AFFICHAGE
0173 REM IDEM : ON VOIT DONC DISTINCTEMENT S'ALLUMER OU S'ETEIDRE LE PIXEL TRAITE A L'ECRAN
0175 REM CALCULE L'EQUIVALENCE DES CONFIGURATIONS DE BITS D'AFFICHAGE
0177 REM IDEM : LA VARIABLE D'AFFICHAGE AFFICHE 132 BARRES VERTICALES DE 6 PIXELS DE HAUT
0179 REM MET A JOUR LA VARIABLE D'AFFICHAGE ET AFFICHE LE RANG ET LA VALEUR ASCII DU RANG
0181 REM D'AFFICHAGE, AFFICHE ET SI NECESSAIRE REINITIALISE LA VARIABLE RELAIS D'AFFICHAGE
0183 REM DEMANDE SI L'ON VEUT METTRE A JOUR UN AUTRE PIXEL
0185 REM SI NEGATIF, SAUT AUX OPTIONS
0187 REM SI AFFIRMATIF ARME LE FLAG O ET SAUT A LA ROUTINE POUR ALLUMER/ETEINDRE UN AUTRE PIXEL
0189 REM POUR IMPRIMER LE GRAPHIQUE MIS A JOUR OU TEL QUE (SELON OPTIONS PRECEDEMENT CHOISIES)
0191 REM SELON LE CAS, SAUT A LA ROUTINE D'IMPRESSION PUIS AUX OPTIONS
0193 REM CAS OU L'ON NE REPOND PAS CORRECTEMENT A LA QUESTION
0195 REM C'EST LE DEBUT DE LA ROUTINE D'IMPRESSION GRAPHIQUE (#TEXTE)
0197 REM SELECTION DE L'EPAISSEUR VERTICALE DU PIXEL IMPRIME (1=1MM ENVIRON) CAS MONOFICHIER
0199 REM CAS OU L'ON NE REPOND PAS CORRECTEMENT A LA QUESTION
0201 REM DEBUT DE SESSION GRAPHIQUE ET AFFICHAGE D'UN TEMOIN A DROITE DE L'ECRAN
0203 REM C2$ SERT AU CENTRAGE DU GRAPHIQUE (DENSITE 160/132)
0205 REM ON COMMENCE PAR LA LIGNE "D'AFFICHAGE" DU HAUT
0207 REM SI IMPRESSION MULTIFICHIER AFFICHE LE NOM DE CELUI EN COURS D'IMPRESSION
0209 REM POUR VOUS RECONFORTER, PUIS AFFICHE LE GRAPHIQUE EN COURS D'IMPRESSION
0211 REM SELON CRITER MONO/MULTI FICHIER
0213 REM IMPRESSION DE LA LIGNE GLOBALE A L'ECHELLE VERTICALE PREALABLEMENT CHOISIE
0215 REM ET... D'UNE
0217 REM ILLUSTRATION SONORE
0219 REM IDEM
0221 REM IDEM
0223 REM FIN D'IMPRESSION EN MODE GRAPHIQUE POUR LA THINKJET
0225 REM ET RETOUR AU MODE TEXTE POUR LA THINKJET
0227 REM LIBERATIONS DE VARIABLES UTILISEES TEMPORAIREMENT
0229 REM FIN DE ROUTINE
0231 REM POUR CREER UN FICHIER DATA DE CE QUE L'ON VIENT DE FAIRE (MISE A JOUR OU CREATION)
0233 REM SI AFFIRMATIF... LE FICHIER DATA SERA TOUJOURS DE 2120 OCTETS
0235 REM CAS OU L'ON REPOND O (OUI) AU LIEU DE PRECISER LE NOM DU FICHIER
0237 REM CREE OU MET A JOUR LE FICHIER DATA EN QUESTION
0239 REM IDEM
0241 REM FIN DE ROUTINE
0243 REM ROUTINE POUR IMPRIMER PLUSIEURS GRAPHIQUES SUPERPOSES SANS BLANCS ENTRE EUX (IMAGE)
0245 REM AINSI L'ON PEUT IMPRIMER UN NOMBRE ILLIMITE DE PAGES GRAPHIQUES EN CONTINU (PATIENCE)
0247 REM CAS OU L'ON NE REPOND PAS CORRECTEMENT A LA QUESTION
0249 REM SI NEGATIF, SAUT AUX OPTIONS
0251 REM SI AFFIRMATIF, ARME LE FLAG 3 ET DEMANDE LE NOMBRE DE FICHIERS A TRAITER
0253 REM LESQUELS PEUVENT RESIDER INDIFFEREMENT EN RAM OU CASSETTES
0255 REM SOLLOCITATION DU NOM DE CHAQUE FICHIER ET TEST DE REPONSE VALIDE
0257 REM IDEM
0259 REM IDEM
0261 REM FIN DE SAISIE DES NOMS DE FICHIERS
0263 REM SELECTION DE L'EPAISSEUR VERTICALE DU PIXEL IMPRIME (1=1MM ENVIRON) ET ARME LE FLAG 3
0265 REM CAS DE REPONSE INVALIDE
0267 REM ACCES SEQUENTIELS AUX FICHIERS DATA CONCERNES AINSI QU'A LA ROUTINE D'IMPRESSION
0269 REM EN MODE GRAPHIQUE
```

```
0271 REM FIN DE ROUTINE
0273 REM LIBERATION DES VARIABLES TEMPORAIRES ET DESARMEMENT DU FLAG 3
0275 REM POUR CONTINUER SUR UNE AUTRE SESSION GRAPHIQUE
0277 REM CAS OU L'ON NE REPOND PAS CORRECTEMENT A LA QUESTION
0279 REM SI AFFIRMATIF, RETOUR EN DEBUT DE PROGRAMME
0281 REM SI NEGATIF, LIBERE LES VARIABES ET DESARME LES FLAGS DU PROGRAMME
```

# 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: FE00000000800001F D57
004: F31BF9614000032BF 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: 000000000834444C4 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: 581008346454830 BEF
03F: 0C3042414C700024 F44
040: 5556587008355654 2A1
041: 5810083546444830 5F0
042: 0C3142404C700025 946
043: 5455587008355455 CA0
044: 581008354454830 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

IOTEST          191 octets

      0123456789ABCDEF sm

000: 94F4455435450202 35E
001: 802E000000000000 68D
002: 28100CE737300000 9E4
003: F710000000000000 D12
004: 041000F594F41573 079
005: 1FF008F625307D40 403
006: 51D104E33300E17B 77F
007: 30C1B604C27F2015 AFC
008: 760208F74030134A E60
009: F24802180FE088D6 1F4
00A: 12F07500C2430061 54C
00B: 371F0F7F21451D5F 8EC
00C: 0715571E508F1517 C5C
00D: 1D51D980F526A825 FEF
00E: 50A0E27A82A0E15D 38E
00F: 7042032CFB8FAB81 735
010: 154331FFD5D0E415 ACE
011: F596A1296561BF6F E7B
012: 69E2C0F6F6B62551 221
013: 17A5AD3332008DA9 5B5
014: 390175147D507135 913
015: 17413706147C9061 C6F
016: F508F2153717F15F 004
017: 7D527B0644005269 373
018: 0E2080D51E5F7F15 70F
019: 771D0F0614713507 A7A
01A: 01F                B24

TRANSLEX        219 octets

      0123456789ABCDEF sm

000: 452514E435C45485 373
001: 802E000000000000 6A2
002: AB100C5334300000 9FB
003: F020000000000000 D23
004: 0BE000FB0013000F 09C
005: 7C43444423394525 3FE
006: 14E442431FF8FBC6 7AF
007: 3117F01882213606 B06
008: 77EF1301818FBC63 EA9
009: 1D88F064A1109DA3 242
00A: 44FF00EE8BB7733C 603
00B: 3B11C315D33ED107 991
00C: 5033326A2B1108D2 CFC
00D: 3146D7211CB11815 06E
00E: 5516114A1C015901 3CA
00F: C01510A6E58E136C 75F
010: B134A6D50D8F064A B06
011: 107134840208D7B1 E6A
012: 818DD44908111360 1D8
013: 6763F1308F064A11 553
014: 09DAD232021EA322 8DC
015: 05108321A0DF8BA3 C69
016: C136EB1341813101 FD4
017: 06110136C213428B 32F
018: F015A0160BF015A0 6B2
019: 136EB1340C54E2F3 A47
01A: 07AC5AF22EA76A74 E01
01B: 550B060D4800D6FE 198
01C: FA4D53E1CF155707 547
01D: A6E5C9633FF        7E7
```

## ANNEXE I INSTRUCTIONS DE L'ASSEMBLEUR

0 bits
00 RTNSXM

2 bits

| | | | | |
|---|---|---|---|---|
| 01 RTN | 02 RTNSC | 03 RTNCC | 04 SETHEX | 05 SETDEC |
| 06 RSTK=C | 07 C5RSTK | 08 CDRST | 09 C=ST | 0A ST=C |
| 0B CSTEX | 0C P=P+1 | 0D P=P-1 | | |

4 bits

| | | | | |
|---|---|---|---|---|
| 0E00 A=A&B | 0E01 B=B&C | 0E02 C=C&A | 0E03 D=D&C | 0E04 B=B&A |
| 0E05 C=C&B | 0E06 A=A&C | 0E07 C=C&D | 0E08 A=A!B | 0E09 B=B!C |
| 0E0A C=C!A | 0E0B D=D!C | 0E0C B=B!A | 0E0D C=C!B | 0E0E A=A!C |
| 0E0F C=C!D | 0EF0 A=A&B A | 0EF1 B=B&C A | 0EF2 C=C&A A | 0EF3 D=D&C A |
| 0EF4 B=B&A A | 0EF5 C=C&B A | 0EF6 A=A&C A | 0EF7 C=C&D A | 0EF8 A=A!B A |
| 0EF9 B=B!C A | 0EFA C=C!A A | 0EFB D=D!C A | 0EFC B=B!A A | 0EFD C=C!B A |
| 0EFE A=A!C A | 0EFF C=C!D A | | | |

2 bits
0F RTI

3 bits

| | | | | |
|---|---|---|---|---|
| 100 R0=A | 101 R1=A | 102 R2=A | 103 R3=A | 104 R4=A |
| 108 R0=C | 109 R1=C | 10A R2=C | 10B R3=C | 10C R4=C |
| 110 A=R0 | 111 A=R1 | 112 A=R2 | 113 A=R3 | 114 A=R4 |
| 118 C=R0 | 119 C=R1 | 11A C=R2 | 11B C=R3 | 11C C=R4 |
| 120 AR0EX | 121 AR1EX | 122 AR2EX | 123 AR3EX | 124 AR4EX |
| 128 CR0EX | 129 CR1EX | 12A CR2EX | 12B CR3EX | 12C CR4EX |
| 130 D0=A | 131 D1=A | 132 AD0EX | 133 AD1EX | 134 D0=C |
| 135 D1=C | 136 CD0EX | 137 CD1EX | 138 D0=AS | 131 D1=AS |
| 13A AD0XS | 13B AD1XS | 13C D0=CS | 13D D1=CS | 13E CD0XS |
| 13F CD1XS | 140 DAT0=A A | 141 DAT1=A A | 142 A=DAT0 A | 143 A=DAT1 A |
| 144 DAT0=C A | 145 DAT1=C A | 146 C=DAT0 A | 147 C=DAT1 A | 148 DAT0=A B |
| 149 DAT1=A B | 14A A=DAT0 B | 14B A=DAT1 B | 14C DAT0=C B | 14D DAT1=C B |
| 14E C=DAT0 B | 14F C=DAT1 B | | | |

4 bits

| | | | | |
|---|---|---|---|---|
| 1500 DAT0=A | 1510 DAT1=A | 1520 A=DAT0 | 1530 A=DAT1 | 1540 DAT0=C |
| 1550 DAT1=C | 1560 C=DAT0 | 1570 C=DAT1 | 1580 DAT0=A | 1590 DAT1=A |
| 15A0 A=DAT0 | 15B0 A=DAT1 | 15C0 DAT0=C | 15D0 DAT1=C | 15E0 C=DAT0 |
| 15F0 C=DAT1 | | | | |

3 bits

| | | |
|---|---|---|
| 160 D0=D0+ | 170 D1=D1+ | 180 D0=D0- |

4 bits
19hh D0=(2)

6 bits
1Ahhhh D0=(4)

7 bits
1Bhhhhh D0=(5)

3 bits
1C0 D1=D1-

4 bits
1Dhh D1=(2)

```
6 bits
1Ehhhh D1=(4)

7 bits
1Fhhhhh D1=(5)

2 bits
20 P=

6 bits
30h..h LCHEX

3 bits
4xx GOC        4hh RTFC       420 NOP3       5hh GONC       500 RTNNC
6000 GOTO

4 bits
6300 NOP4

5 bits
64000 NOP5

4 bits
7000 GOSUB

3 bits
800 OUT=CS     801 OUT=C      802 A=IN       803 C=IN       804 UNCNFG
805 CONFIG     806 C=ID       807 SHUTDN

4 bits
8000 INTON     808F INTOFF

3 bits
809 C+P+1      80A RESET      80B BUSCC

4 bits
80C0 C=P       80D0 P=C

3 bits
80E SREQ?

4.bits
80F0 CPEX

3 bits
810 ASLC       811 BSLC       012 CSLC       813 DSLC       814 ASRC
815 BSRC       816 CSRC       817 DSRC       81C ASRB       81D BSRB
01E CSRB       81F DSRB       821 XM=0       822 SB=0       824 SR=0
828 MP=0       82F CLRHST

5 bits
83100 ?XM=0    83200 ?SB=0    83400 ?SR=0    83800 ?MP=0

3 bits
840 ST=0       850 ST=1

5 bits
86000 ?ST=0    87000 ?ST=1    88000 ?P#      89000 ?P=      8A000 ?A=B A
8A100 ?B=C A   8A200 ?C=A A   8A300 ?C=D A   8A400 ?B#A A   8A500 ?B#C A
8A600 ?A#C A   8A700 ?C#D A   8A800 ?A=0 A   8A900 ?B=0 A   8AA00 ?C=0 A
```

```
8AB00 ?D=0 A   8AC00 ?A#0 A   8AD00 ?B#0 A   8AE00 ?C#0 A   8AF00 ?D#0 A
8B000 ?A>B A   8B100 ?B>C A   8B200 ?C>A A   8B300 ?D>C A   8B400 ?A<B A
8B500 ?B<C A   0B600 ?C<A A   8B700 ?D<C A   8B800 ?A>=B A  8B900 ?B>=C A
8BA00 ?C>=A A  8BB00 ?D>=C A  8BC00 ?A<=B A  8BD00 ?B<=C A  0BE00 ?C<=A A
8BF00 ?D<=C A
```

6 bits

```
8Chhhh GOLONG
```

7 bits

```
8Dhhhhh GOVLNG
```

6 bits

```
8Ehhhh GOSUBL
```

7 bits

```
8Fhhhhh GOSBVL
```

5 bits

```
90000 ?A=B     90100 ?B=C     90200 ?C=A     90300 ?C=D     90400 ?A#B
90500 ?B#C     90600 ?C#A     90700 ?C#D     90800 ?A=0     90900 ?B=0
90A00 ?C=0     90B00 ?D=0     90C00 ?A#0     90D00 ?B#0     90E00 ?C#0
90F00 ?D#0     98000 ?A>B     98100 ?B>C     98200 ?C>A     98300 ?D>C
98400 ?A<B     98500 ?B<C     98600 ?C<A     98700 ?D<C     98800 ?A>=B
98900 ?B>=C    98A00 ?C>=A    98B00 ?D>=C    98C00 ?A<=B    98D00 ?B<=C
98E00 ?C<=A    98F00 ?D<=C
```

3 bits

```
A00 A=A+B      A01 B=B+C      A02 C=C+A      A03 D=D+C      A04 A=A+A
A05 B=B+B      A06 C=C+C      A07 D=D+D      A08 B=B+A      A01 C=C+B
A0A A=A+C      A0B C=C+D      A0C A=A-1      A0D B=B-1      A0E C=C-1
A0F D=D-1      A80 A=0        A81 B=0        A82 C=0        A83 D=0
A84 A=B        A85 B=C        A86 C=A        A87 D=C        A88 B=A
A81 C=B        A8A A=C        A8B C=D        A8C ABEX       A8D BCEX
A8E CAEX       A8F CDEX       B00 A=A-B      B01 B=B-C      B02 C=C-A
B03 D=D-C      B04 A=A+1      B05 B=B+1      B06 C=C+1      B07 D=D+1
B08 B=B-A      B01 C=C-B      B0A A=A-C      B0B C=C-D      B0C A=B-A
B0D B=C-B      B0E C=A-C      B0F D=C-D      B80 ASL        B81 BSL
B82 CSL        B83 DSL        B84 ASR        B85 BSR        B86 CSR
B87 DSR        B88 A=-A       B89 B=-B       B8A C=-C       B8B D=-D
B8C A=-A-1     B8D B=-B-1     B8E C=-C-1     B8F D=-D-1
```

2 bits

```
C0 A=A+B A     C1 B=B+C A     C2 C=C+A A     C3 D=D+C A     C4 A=A+A A
C5 B=B+B A     C6 C=C+C A     C7 D=D+D A     C8 B=B+A A     C9 C=C+B A
CA A=A+C A     CB C=C+D A     CC A=A-1 A     CD B=B-1 A     CE C=C-1 A
CF D=D-1 A     D0 A=0 A       D1 B=0 A       D2 C=0 A       D3 D=0 A
D4 A=B A       D5 B=C A       D6 C=A A       D7 D=C A       D8 B=A A
D9 C=B A       DA A=C A       DB C=D A       DC BAEX A      DD CBEX A
DE CAEX A      DF DCEX A      E0 A=A-B A     E1 B=B-C A     E2 C=C-A A
E3 D=D-C A     E4 A=A+1 A     E5 B=B+1 A     E6 C=C+1 A     E7 D=D+1 A
E0 B=B-A A     E9 C=C-B A     EA A=A-C A     EB C=C-D A     EC A=B-A A
ED B=C-B A     EE C=A-C A     EF D=C-D A     F0 ASL A       F1 BSL A
F2 CSL A       F3 DSL A       F4 ASR A       F5 BSR A       F6 CSR A
F7 DSR A       F8 A=-A A      F9 B=-B A      FA C=-C A      FB D=-D A
```

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.