Le BASICIUM

Gérard Michel

Editions MCV - 64/78 rue des Chantiers - 78000 Versailles

Le BASICIUM

Présentation

Le BASICIUM constitue un ensemble de routines en langage-machine et d'utilitaires destinés à faciliter et à améliorer la gestion d'écran et la saisie de données réalisées par vos programmes en Applesoft. Il est conçu pour fonctionner sur un Apple //e ou un Apple II+ 48K.

Il vous permettra de "dessiner" des masques en écran texte en donnant libre cours à vos talents esthétiques, et de les utiliser ensuite très simplement dans vos programmes. Les variables saisies à l'intérieur de ces masques pourront être définies sous forme de tableaux qu'il sera également facile de rentrer, modifier, afficher ou effacer dans vos programmes. Le déplacement entre les variables d'un tableau à l'écran sera pris en charge par le BASICIUM. Celui-ci vous offre en outre des instructions synthétiques pour gérer les messages que vous adressez à l'utilisateur (questions, messages d'erreur...). Enfin, il vous apportera une assistance dans la mise en forme de vos états imprimés, par le biais d'un dessin préalable de ces derniers sous forme de masque, puis d'une copie d'écran sélective ligne à ligne.

Entre autres, il reprend sous une forme plus élaborée la gestion de masques et l'INPUT généralisé de tableaux déjà présentés par Pom's.

L'appel des routines ne se fait pas par le biais de l'ampersand (&). De ce fait, et dans la mesure où elles n'occupent pas le même espace-mémoire que celles du BASICIUM, vous devriez pouvoir utiliser vos routines "&" favorites dans un programme faisant également usage du BASICIUM.

Le branchement sur ce dernier se fait au moyen du caractère "]" (qui s'obtient directement sur un Apple //e, et par SHIFT-M sur un Apple II+). Le sous-programme de l'Apple baptisé CHRGET, situé de l'adresse $B1 à l'adresse $C8, et qui fournit à l'interpréteur chaque caractère du programme Basic en cours d'exécution, est donc modifié par l'initialisation du BASICIUM. Celle-ci place à l'adresse $BA un "JMP $9190", $9190 étant l'adresse de la routine qui controle la présence éventuelle d'un "]" et assure le branchement vers les instructions concernées si cette recherche est fructueuse (elle correspond aux fichiers ANA.U.SCE et ANA.U.OBJ analysés plus loin).

Le caractère "]" doit toujours être précédé par ":", même s'il constitue la première instruction dans une ligne et même s'il vient après THEN, sous peine de provoquer une SYNTAX ERROR. Le petit exemple ci-dessous présente des formats corrects :

```
10 :]IZ:PRINT
20 PRINT A:]IZ
30 IF B=1 THEN:]IZ:]PW
```

Le BASICIUM se loge en mémoire entre les adresses décimales 35860 e 38399. Un programme utilisateur doit donc fixer la HIMEM à 35859 a maximum. En pratique, la valeur définitive de HIMEM dépendra de masques et variables gérés; les éléments nécessaires à son calcul son indiqués dans le cours de cette documentation.

Les instructions autonomes

1) Nettoyage mémoire

Le BASICIUM utilise l'une des routines de nettoyage publiées dan Pom's 2 et destinées à remplacer "l'escargot" FRE(0). Il s'agit de l version traitant 64 chaines de caractères par exploration, avec tes préalable sur la place mémoire disponible pour juger de l'utilité d nettoyage. L'appel de la routine se fait par l'instruction :

```
]F
```

Cette instruction peut être utilisée notamment dans des boucle manipulant de nombreuses chaines, ou dans les opérations de lecture e écriture sur des fichiers de données. Par exemple :

```
10 FOR I=500 TO 1 STEP-1:A$(I+1)=A$(I):]F:NEXT

20 FOR I=1 TO 500:PRINT D$"READ FICHIER,R"I:FOR J=1 TO 10:
   INPUT W$(J):PRINT W$(J):NEXT:]F:NEXT
```

Il n'est pas nécessaire de se préoccuper du nettoyage-mémoire lorsqu vous faites de la saisie de tableaux par le BASICIUM. En effet, l routine d'INPUT se charge de faire appel à ce dernier lors de l rentrée de chaines de caractères.

2) Affichage de messages appelant une réponse O ou N

Syntaxe : `]QMEi`
ou : `]Q"MEc"`

Supposons que l'on dispose d'un tableau ME$ dans lequel son stockées les différentes questions qui doivent être posées à l'écran (saisie confirmée, êtes-vous bien d'accord, imprimante branchée...). L'instruction]QMEi aura les conséquences suivantes :

- Affichage en inverse à la ligne 23 du message contenu dans ME$(I), suivi d'un "?".
- Attente d'une réponse-clavier sur un caractère qui ne peut être que O ou N, puis RETURN. Si, après avoir tapé O par exemple, vous voulez rectifier votre réponse, vous pouvez revenir en arrière par la flè-

che à gauche et taper N (la flèche à droite n'est jamais acceptée).
- Après la frappe du RETURN, l'adresse 6 contient le code de la réponse : 1 si c'est O, et O si c'est N. Au niveau du programme BASIC, la lecture de PEEK(6) permet donc d'orienter le traitement en fonction du sens de la réponse.

Avec la syntaxe ci-dessus, l'indice i peut varier de O à 9. Pour les indices supérieurs à 9, les choses sont un peu plus compliquées car le BASICIUM travaille sur le code ASCII d'un seul caractère auquel il retire $30 pour calculer la valeur réelle de l'indice dans le tableau. Il faut alors utiliser la syntaxe]Q"MEc" où c est le caractère dont le code ASCII diminué de $30 (48 en décimal) donne la valeur désirée. Par exemple :

`"ME;" pour traiter ME$(11)
]Q"ME<" pour traiter ME$(12)
]Q"ME=" pour traiter ME$(13)...

Dans le cas de l'indice 10, toutefois, vous pouvez utiliser la syntaxe]QME: car le ":" n'est pas "tokenisé" et se trouve stocké à l'intérieur du programme sous son code ASCII ($3A soit 58 en décimal).

Si vous avez trop de questions à poser pour qu'un seul code ASCII puisse correspondre à l'indice maximum, il vous faudra définir plusieurs tableaux.

Pour baptiser vos tableaux de messages, vous pouvez utiliser n'importe quel groupe de deux lettres :]QZZi pour le tableau ZZ$,]QYUi pour le tableau YU$...
Les noms de variables à une lettre ou plus de deux lettres ne sont pas acceptés (SYNTAX ERROR). En outre, employer une instruction]Q avec un tableau non encore défini ou avec un indice supérieur à la dimension provoque une telle "perturbation" dans le système que vous n pourrez sortir que par RESET.

3) Affichage de messages appelant RETURN ou ? pour réponse

Syntaxe :]RMEi
ou :]R"MEc"

Mêmes remarques que ci-dessus en ce qui concerne les indices et les noms de variables.

Cette instruction est notamment utile chaque fois qu'il y a une consultation de données à l'écran, avec des messages du type 'RETURN' OU '?' POUR HARD-COPY.
Le message est affiché en inverse à la ligne 23 et seul RETURN (qui met l'adresse 6 à O) ou ? (qui met l'adresse 6 à 1) sont autorisés comme réponse au clavier.
L'option "?" a été prévue pour orienter le traitement suivant vers u copie de l'écran sur imprimante (voir infra).

4) Affichage de messages avec "beep"

Syntaxe :]BMEi
ou :]B"MEc"

Mêmes remarques que ci-dessus en ce qui concerne les indices et les noms de variables.

Cette instruction peut servir notamment à l'émission de messages d'erreur. Le texte est affiché en inverse à la ligne 23, accompagné de deux "beep", et il s'y maintient quelques instants pour permettre la lecture avant de disparaître.

A noter qu'un même tableau peut bien entendu contenir des messages destinés à une utilisation par]Q,]R ou]B.

5) Hard-copy d'écran TEXT

Syntaxe :]H

Cette instruction réalise une copie de l'écran sur une imprimante préalablement branchée et sélectionnée par PR#s (où s est le slot de l'imprimante). Le programme de démonstration listé plus loin donne un exemple d'utilisation de]H à la suite d'une question posée par]R.

Gestion des masques

La gestion des masques doit vous permettre de réaliser facilement une présentation complexe ou "esthétique" de vos écrans, tout en réduisant sensiblement la taille des parties de programmes correspondantes (pas de VTAB, HTAB et autres PRINT à répétition...). Elle s'applique à la fois aux écrans purement "visuels", ce en quoi elle ressemble à un éditeur graphique basse résolution, et aux masques de saisie de données.

De façon générale, le BASICIUM permet de dissocier assez nettement la conception de la gestion des écrans et la phase de programmation des traitements. Vous pouvez ainsi définir dans un premier temps les différentes données à entrer au clavier, la position où elles seront affichées sur l'écran et les textes que l'on trouvera en regard (masques de saisie), les noms de variables, puis constituer tous ces éléments à l'aide des utilitaires du BASICIUM. L'exploitation au niveau de votre programme Basic se fera ensuite avec quelques instructions simples et synthétiques, sans que vous ayez à vous préoccuper de la mise au point des routines.

1) Mise au point des masques

Elle se fait au moyen de l'utilitaire GESMASK. Concrètement, la création d'un masque consiste tout simplement à "dessiner" celui-ci à l'écran exactement tel qu'il se présentera pour l'utilisateur du programme de traitement.

Le premier écran de GESMASK vous permet de choisir la fonction (en toute logique, nous commencerons par CREATION) et de donner un nom au masque traité. Le deuxième demande le drive où sera stocké le masque (ou celui sur lequel il se trouve déjà s'il s'agit d'un masque existant); les informations concernant l'imprimante ne sont réclamées que si vous avez choisi l'option IMPRESSION.
A noter que GESMASK, et c'est bien la moindre des choses, utilise la routine d'INPUT de tableaux que nous analyserons plus tard. En particulier, cela vous permet de "remonter" du nom de masque au choix de la fonction en tapant ESC en premier caractère sur la zone "nom" et de revenir du deuxième écran au premier en tapant ESC en premier caractère sur la zone "drive". De même, dans l'option IMPRESSION, vous pouvez remonter sur le deuxième écran au moyen de cette touche ESC, ceci en vue de la modification éventuelle des données entrées. Le passage d'une zone à la suivante se fait par RETURN.

Pour en revenir à notre création, GESMASK vérifie, après que vous lui ayez fourni les informations nécessaires, que le masque indiqué n'existe pas déjà et, si tel est bien le cas, il vous "livre" ensuite un écran "blanc" et un curseur. Il ne vous reste plus qu'à écrire ce que vous voulez voir apparaitre dans votre programme (nous vous donnons plus loin des exemples de masques ainsi créés).
Quelques touches de fonction doivent vous faciliter ce travail :

- CTRL-N : écriture en mode normal
- CTRL-I : écriture en mode inverse
- CTRL-V : répétition verticale d'un caractère (il faut se placer sur le caractère à répéter puis taper CTRL-V)
- CTRL-R : remontée sur l'écran sans affecter celui-ci
- RETURN : descente dans les mêmes conditions que ci-dessus
- Flèches à gauche et à droite : déplacement latéral
- CTRL-G : décalage vers la gauche à partir de la position du curseur (les caractères suivants se déplacent d'une position vers la gauche)
- CTRL-D : décalage vers la droite.

Lorsque vous avez terminé votre création, tapez ESC pour sauver votre masque. Attention, il vaut mieux ne pas faire ce ESC lorsque le curseur est tout en bas de l'écran car le PRINT D1$ de GESMASK risque de vous effacer une ligne; il suffit de remonter de quelques positions.

Les autres fonctions de GESMASK ne posent pas de problème d'utilisation une fois que l'on a fait un test de création. En MODIFICATION, vous disposez des mêmes touches de fonction et le masque modifié sera sauvé sur disquette. En CONSULTATION, le retour au menu se fait toujours par ESC, sans sauvegarde.

Notons simplement que, pour l'impression d'un masque, les caractères affichés en inverse à l'écran sont recodés en normal pour l'imprimante (un cadre de "blancs" en inverse, par exemple, ne ressort pas sur le papier).

Sous cette forme "développée", les masques sont utilisables à partir d'un programme par PRINT D$"BLOAD Nom du Masque", où vous aurez défini D$ par CHR$(4), et ils viennent se charger directement sur l'écran à partir de la disquette. Cette solution est évidemment la plus économe en place-mémoire, mais elle impose un temps de chargement entre chaque écran et pose quelques problèmes de paramètres aux périphériques ("bruits" de lecteur en particulier - voir sur ce point les remarques publiées dans Pom's 9).
C'est pourquoi le BASICIUM vous propose deux autres options d'utilisation, avec des masques réduits et chargés en mémoire.

2) Manipulation et utilisation des masques

La manipulation des masques est réalisée par l'utilitaire GPMASK qui, comme GESMASK, utilise les routines du BASICIUM pour les traitements d'écran et la saisie des données.

2.1 - Contraction des masques

Cette opération est la première à effectuer. Elle consiste à réduire la taille du masque (1024 octets sous sa forme développée) en analysant les séquences de caractères semblables. Ainsi, dès lors que plus de 2 caractères consécutifs sur une ligne d'écran sont identiques, on ne stocke plus que le code du caractère et le nombre de ses occurences. Les caractères isolés ou doublés sont repris tels quels. Le seuil de modification de la forme de stockage étant fixé à 3 caractères semblables au minimum, il est certain qu'un masque "contracté" n'occupera jamais plus de place que l'original (au pire, il prendra le même nombre d'octets, mais on arrive en fait dans la très grande majorité des cas à une réduction sensible).
Le masque ainsi analysé est copié au fur et à mesure dans une zone tampon qui est ensuite sauvée sur disquette.

Les informations à fournir à l'utilitaire GPMASK pour cette opération sont le nom du masque "original", le nom du masque réduit (sous lequel il sera sauvegardé sur disquette), le lecteur d'origine (original) et le lecteur de destination. En fin de traitement, la longueur du masque contracté est affichée. Il est conseillé de

toujours conserver le masque original en prévision des modifications éventuelles qu'il faudrait lui apporter (par GESMASK), car la forme contractée n'est pas aisément modifiable.

Un masque réduit peut être utilisé tel quel à partir d'un programme selon la procédure suivante :

- BLOAD du masque à l'adresse ADR (au maximum, ADR est égal à 35859 diminué de la longueur du masque).
- POKE 25,INT(ADR/256):POKE 24,ADR-256*INT(ADR/256). L'adresse du tampon dans lequel est chargé le masque doit être placée en 24-25.
- CALL 38304 : appel de la routine qui reconstitue un masque à l'écran à partir de sa forme contractée.

Cette procédure étant encore relativement lourde, le BASICIUM vous propose une autre possibilité beaucoup plus facile à gérer pour le programme utilisateur.

2.2 - Constitution d'un bloc de masques réduits

Vous pouvez par cette opération regrouper des masques préalablement contractés au sein d'un même fichier en les repérant par un code, qui peut être un chiffre entre 0 et 9 ou une lettre entre A et Z, à l'exception du T (nous verrons plus loin pour quelle raison). Deux masques différents ne peuvent avoir le même code.

Dans le programme d'application, il suffit ensuite de charger ce fichier et d'appeler les masques par leur code pour les afficher. On peut évidemment utiliser plusieurs blocs différents à l'intérieur d'un même programme, et les codes définissant les masques peuvent être identiques d'un bloc à l'autre.

En ce qui concerne l'exploitation de GPMASK pour ce traitement, la seule donnée pouvant poser problème est la réponse à la question "HIMEM DONT ON PART", qui doit fournir l'adresse à partir de laquelle sera constituée le bloc (en descendant). Pour réduire l'encombrement de la mémoire, il est conseillé de placer le bloc juste en-dessous du descriptif des variables, élément qui sera discuté ci-dessous. La HIMEM en question serait alors l'adresse donnée en fin de traitement par CRETAB et diminuée de 1.

GPMASK demande en séquence des noms de masque et des codes correspondants (rien ne s'oppose à ce qu'un bloc soit constitué d'un seul masque); lorsque le dernier élément constitutif du bloc a été traité, la touche ESC amène à la zone de saisie du nom du bloc et du drive de destination. Après sauvegarde, l'adresse de départ et la longueur du bloc sont affichées, ce qui doit permettre notamment de déterminer la valeur de HIMEM à donner dans le programme d'application.

Supposons, à titre d'exemple, que l'on ait constitué un groupe de masques réduits baptisé BLOC et comprenant :

- MASK-R1 = code 1
- MASK-RA = code A
- MASK-R9 = code 9.

L'utilisation des masques dans le programme de traitement repose sur les instructions suivantes :

- Chargement du bloc : PRINT D$"BLOAD BLOC":POKE 37254,PEEK(43634): POKE 37255,PEEK(43635). Les adresses 37254 et 37255 se situent dans le BASICIUM; on y stocke une fois pour toutes l'adresse du début du bloc que les routines chargeront ensuite aux adresses 24 et 25 à chaque appel d'affichage, comme dans la première procédure présentée plus haut.

- Affichage de MASK-R1 :]A1

- Affichage de MASK-RA :]AA

- Affichage de MASK-R9 :]A9

- Les instructions]A1:]AA:]A9 auraient pour conséquence l'affichage successif et "instantané" (sans même que vous ayez le temps de lire les deux premiers) des trois masques.

L'instruction]A, combinée à un bloc de masques réduits, permet donc de supprimer les problèmes de délai avant affichage (lecture disque) et de paramètres des périphériques, au prix d'un encombrement minimum du fait de la compression, et au moyen d'une syntaxe simple pour le programmeur. Cette syntaxe explique d'ailleurs pourquoi la lettre T ne peut pas être utilisée comme code puisque AT est un mot réservé de l'Applesoft, et le reste même lorsqu'il est précédé de].

A noter que l'emploi de]A avec un code qui n'existe pas dans le bloc plonge l'Apple dans un abime de réflexion dont seul RESET peut le tirer !

2.3 Impression paramétrée

Il s'agit d'une application particulière des masques destinée à faciliter l'édition sur imprimante, par le biais d'une copie d'écran "sélective".

Les lignes à imprimer sont enregistrées dans un masque et repérées par des paramètres sous forme de lettres en inverse (il ne doit pas y avoir de caractères en inverse dans les lignes elles-mêmes).

Supposons ainsi que l'on veuille éditer une liste de ce type :

```
**************************************************************************
*                        *                        *                       *
*         NOM            *       PRENOM           *      PROFESSION       *
*                        *                        *                       *
**************************************************************************
*                        *                        *                       *
* liste des noms         * liste des prénoms      * liste des professions *

          ....                     ....                     ....

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

Le masque utilisé pourrait se présenter ainsi, les lettres entre parenthèses symbolisant les paramètres en inverse :

```
(A)**************************************************************************(A)
(B)*                        *                        *                       *(B)
(C)*         NOM            *       PRENOM           *      PROFESSION       *(C)
(D)*                        *                        *                       *(D)
```

Les instructions permettant l'édition seraient alors les suivantes :

-]AO affichage du masque que l'on suppose repéré par le code 0 dans un bloc.
- Sélection de l'imprimante.
-]CA:]CB:]CC:]CB:]CA:]CB impression du cadre-titre de l'état.
- Retour à l'écran (PRØ0).
- Lecture du fichier dans lequel sont stockées les données, avec pour chaque enregistrement :
  - réaffichage du masque par]AO (les instructions PRINT D$ de sélection imprimante-écran pouvant avoir des conséquences sur la présentation de l'écran)
  - affichage des données à l'intérieur de la ligne (D), notamment au moyen d'un PRINT de tableau (voir plus loin)
  - sélection de l'imprimante, puis]CD, puis retour à l'écran.
- En fin de fichier, sélectionner l'imprimante et finir par]CB:]CA.

L'utilisation de]C avec un paramètre inexistant sur le masque peut avoir toutes sortes de conséquences, à l'exception du résultat escompté !

Saisie des données - Input généralisé de tableaux

Là encore, le BASICIUM doit vous permettre de préparer la saisie-clavier des données, ainsi que leur restitution à l'écran, indépendamment des traitements auxquels elles seront soumises.
Dans la mesure où vous avez déterminé les informations nécessaires à votre programme, nous vous conseillons de procéder de la manière suivante :

a) Définir sur le papier la présentation des écrans et notamment le libellé des questions posées à l'utilisateur. Mettre en regard des libellés les variables correspondantes, par exemple :

```
NOM DU PRODUIT : Z$(0)          )
FOURNISSEUR : Z$(1)             ) -> masque ECR1
QUANTITE EN STOCK : Z$(2)       )
```

Vous savez déjà à ce point quels seront le type (alphanumérique, numérique ou date) et la longueur des informations à rentrer dans le tableau Z$.

b) Créer les masques avec GESMASK. Au cours de la création, noter à quel endroit sont placées les zones de saisie des variables sur l'écran (VTAB est compté à partir de la position 0, de même que HTAB). Vous pouvez constituer sur un morceau de papier ou de carton un modèle quadrillé pour relever plus facilement les coordonnées à l'écran.

c) Passer à l'utilitaire CRETAB pour enregistrer les descriptifs des tableaux qui seront exploités dans le programme d'application, et ce avant de manipuler les masques par GPMASK.

1) Fonctionnement de CRETAB

Deux options sont offertes : création d'un descriptif de tableaux, ou modification d'un descriptif déjà enregistré.

1.1 - Création

Il faut indiquer le nombre de tableaux qui seront intégrés dans le descriptif (le maximum est fixé à 10 dans CRETAB, mais vous pouvez modifier cette valeur si vous ne craignez pas de surcharger la mémoire avec un descriptif trop long). Un même descriptif peut comporter des tableaux qui interviennent sur des écrans différents, plusieurs tableaux intervenant sur le même écran... Toutes les combinaisons sont possibles puisque les instructions d'affichage des masques et celles de gestion des tableaux sont dissociées. Un programme d'application peut également appeler différents descriptifs.

La dimension maximale correspond bien entendu à celle du tableau le plus grand (indiquer x pour un tableau à x indices, de 0 à x-1). CRETAB vérifie qu'aucun des tableaux du descriptif ne dépasse cette valeur maximale.

Vous indiquez ensuite pour chaque tableau :

- Une lettre qui servira pour le tableau Applesoft (Z pour Z$, A pour A$...). La routine d'INPUT ne peut traiter que des tableaux alphanumériques dont le nom comporte une seule lettre (la lettre F pose un problème que nous examinerons plus loin). Si vous avez besoin de plus de 26 tableaux dans votre programme, vous pouvez utiliser plusieurs descriptifs; ils sont "étanches" en ce qui concerne les noms de variables utilisés. Au sein d'un même descriptif, deux tableaux ne peuvent cependant être repérés par une lettre identique.
- La dimension du tableau.

Pour un tableau donné, CRETAB vous demandera ensuite les paramètres de chaque variable (Z$(0), Z$(1)...) : type (1 pour alphanumérique, 2 pour numérique, 3 pour une date), longueur maximale autorisée, VTAB et HTAB. En principe, il devrait vous suffire de reprendre le papier rempli tout à l'heure pour réaliser cette opération.

CRETAB utilise bien évidemment les routines d'INPUT du BASICIUM, ce qui donne à la touche ESC un role particulier :

- ESC en premier caractère, à la demande du nombre de tableaux, renvoie au menu;
- ESC sur la dimension maximale renvoie au nombre de tableaux;
- ESC sur la lettre du premier tableau provoque une sortie de CRETAB (pour éviter une erreur de redimensionnement, comme le montre l'analyse du programme);
- ESC sur la dimension renvoie à la lettre;
- à l'intérieur de la saisie des paramètres pour une variable donnée, ESC ramène au paramètre précédent; ESC sur le premier d'entre eux renvoie à la variable précédente si elle existe et à la lettre du tableau dans le cas contraire;
- ESC sur la lettre d'un tableau qui n'est pas le premier ramène à la lettre du tableau précédent;
- ESC à la demande du nom de fichier permet le retour à la lettre du dernier tableau, tandis que sur le numéro de lecteur il renvoie au nom du fichier.

En phase de "modification" de données déjà entrées, un RETURN en premier caractère sur la valeur affichée la conserve inchangée. Le passage d'une zone à la suivante se fait toujours par RETURN (l'utilisation des touches ESC et RETURN sera discutée un peu plus loin sur le plan général).

Aussi longtemps que l'on est en traitement de création, il est possible de changer à tout moment la lettre ou la dimension d'un tableau; ce n'est plus possible en mode modification après enregistrement du fichier (voir infra).

En fin de création, le descriptif est sauvé sur disquette et CRETAB affiche son adresse de début, qui vous servira notamment pour indiquer le HIMEM de départ d'un bloc de masques dans GPMASK (normalement, un descriptif de tableaux est logé pour sa part juste sous le BASICIUM).

1.2 - Modification d'un descriptif

Il doit tout d'abord être rechargé en mémoire à l'appel de son nom, première information demandée dans ce cas par CRETAB. Deux options sont offertes :

a) Ajout de tableaux : la procédure est alors la même que pour la création, le nombre de tableaux créés en sus venant simplement compléter le descriptif et le fichier correspondant.

b) Modification pure : la lettre du premier tableau est affichée et il vous est demandé de préciser s'il est correct ou non. Dans l'affirmative et s'il reste des tableaux, on vous demande FINI ? (sans commentaires); si vous n'avez pas fini, on passe au tableau suivant, jusqu'à épuisement du stock ou indication de la fin. Si un tableau est signalé incorrect, les paramètres de sa première variable sont affichés, avec demande de confirmation; l'absence de cette dernière permet de les modifier (dans un cas comme dans l'autre, le traitement affiche ensuite la variable suivante si elle existe). Le descriptif ainsi modifié est finalement sauvé sur disquette sous son nom d'origine.

Vous aurez noté qu'il n'est plus possible à ce niveau de changer la lettre ou la dimension d'un tableau.

2) Utilisation des tableaux dans un programme d'application

Il sera plus simple, pour illustrer notre propos, de prendre l'exemple d'un descriptif VAR, comprenant les tableaux Z (indices de 0 à 20), X (de 0 à 5) et Y (de 0 à 3).

Le chargement se fera en début de programme par PRINT D$"BLOAD VAR" : POKE 37191,PEEK(43634) : POKE 37192,PEEK(43635) - Les routines iront toujours rechercher à ces deux adresses celle du début du descriptif qui leur est nécessaire. Il est INDISPENSABLE de donner ensuite la dimension des tableaux : DIM Z$(20),X$(5),Y$(3).

Les instructions possibles sont ensuite, par exemple :

- Pour saisir toutes les variables de Z$:]IZ (ceci suffit pour saisir les 21 variables, avec tabulation, controle sur le type et la longueur, et retrouver leurs valeurs en Z$(0) à Z$(20)).
- Pour modifier les valeurs de X$:]MX (les valeurs sont en X$(0) à X$(5)).
- Pour afficher les valeurs de Y$:]PY.

- Pour effacer les valeurs affichées de Z$, sans affecter le reste de l'écran :]EZ.

De façon générale, les instructions qui comportent une saisie-clavier (]I et]M) possèdent les caractéristiques suivantes :

- Un ESC en premier caractère sur la première zone de saisie provoque une sortie de la routine d'INPUT. L'adresse 6, normalement mise à 0 à l'appel de la routine, est alors mise à 1, d'où test sur PEEK(6) dans le programme utilisateur pour savoir si la saisie a été abandonnée ou non. Ceci permet notamment de revenir à une autre phase de traitement sans rester "bloqué" dans la saisie.
- Un ESC en premier caractère sur une zone autre que la première provoque la remontée à la zone précédente pour en modifier la valeur.
- Le passage d'une zone à la suivante se fait toujours par RETURN, qui valide en même temps comme donnée la suite de caractères qui le précède à l'écran.
- Lors de la modification d'une valeur, un RETURN en premier caractère la conserve inchangée (valeur par défaut).
- Les flèches à gauche et à droite ont le même effet que dans l'instruction INPUT de l'Applesoft.

A noter que le BASICIUM autorise la frappe de ":" et ";" au clavier, mais les problèmes que pourraient poser ces deux caractères avec le DOS demeurent. La virgule est, pour sa part, systématiquement remplacée par un point.

L'emploi de]I,]M,]P,]E avec un tableau qui n'existe pas dans le descriptif ou qui n'a pas été préalablement dimensionné entraine bel et bien toutes les conséquences désastreuses auxquelles vous pensez !

Par ailleurs, la syntaxe]I explique la difficulté que présente un tableau F$, puisque IF est un mot réservé de l'Applesoft, même précédé de].

Le BASICIUM vous propose enfin une dernière série d'instructions permettant de bénéficier des caractéristiques de l'INPUT généralisé de tableaux pour des données qui ne se prêtent pas naturellement à un traitement par tableau. Il s'agit notamment des données saisies isolément avant un autre type de travail (date ou titre avant l'impression d'un état) ou qui réclament un controle immédiat avant la suite de la saisie (vérification de l'existence de la clé d'un article dans une table de références, par exemple).

Plutot que de créer un "pseudo-tableau" à un indice pour chacune de ces entrées, il est conseillé de les regrouper toutes dans un seul tableau spécial et de leur appliquer les instructions fonctionnant "par indice", dont la syntaxe est la suivante (le tableau Z$ est pris à titre d'exemple) :

-]UiIZ ou]U"cIZ" pour la saisie proprement dite.
-]UiMZ ou]U"cMZ" pour la modification.
-]UiPZ ou]U"cPZ" pour affichage (PRINT).
-]UiEZ ou]U"cEZ" pour effacer.

Les règles d'utilisation d'un indice i chiffré ou d'un caractère c sont les mêmes que pour les instructions]Q,]R et]B (voir supra).

Contenu de la disquette BASICIUM

1) START : programme de présentation qui vous propose :

- initialisation du BASICIUM pour l'écriture de vos propres programmes;
- programme de démonstration (DEMO-PGM);
- gestion de masques (GESMASK);
- gestion de tableaux de variables (CRETAB);
- manipulation de masques (GPMASK);
- accès au Basic normal (Applesoft);
- accès au BASICIUM version 80 colonnes.

START utilise le bloc de masques PRE-B.

2) Gestion de masques :
- GESMASK : programme en BASICIUM
- MASKIN.OBJ : routine en langage machine pour le "dessin" à l'écran (voir Pom's 7);
- GESMASK-BLOC : bloc des deux masques réduits;
- GESMASK-VAR : descriptif des tableaux de variables utilisés dans GESMASK.

3) Gestion des tableaux de variables :
- CRETAB : programme en BASICIUM;
- CRETAB-B : "bloc" de masques, se réduisant à un seul masque, mais néanmoins constitué pour des raisons de facilité d'emploi dans CRETAB;
- CRETAB-V : variables.

4) Manipulation des masques :
- GPMASK : programme en BASICIUM;
- GPM-B : bloc des masques;
- GPM-V : variables.

5) Programme de démonstration : DEMO-PGM en BASICIUM qui utilise les variables DEMO-VAR et le bloc de masques DEMO-BLOC.

CRE-FIC-POUR DEMO PARA est un petit programme qui crée un fichier de valeurs numériques aléatoires FIC que peut utiliser ensuite le programme DEMO-PARA-AVEC FIC pour illustrer l'intérêt de l'impression paramétrée lors de l'édition des données d'un fichier.

6) Routines du BASICIUM (les fichiers sources sont toujours en assembleur LISA 1.5 - suffixe .SCE - les fichiers objets sont suffixés par .OBJ) :

- IG2.SCE et IG2.OBJ : INPUT de tableaux;
- HCT2.SCE et HCT2.OBJ : hard-copy d'écran TEXT;
- PARA2.SCE et PARA2.OBJ : impression paramétrée;
- QRB.SCE et QRB.OBJ : traitement des instructions]Q,]R et]B;
- IG.U.SCE et IG.U.OBJ : analyse des instructions d'INPUT par indice avant le branchement sur IG2;
- ANA.U.SCE et ANA.U.OBJ : "interpréteur" général des instructions] qui organise le branchement sur les routines de traitement concernées. C'est à lui que conduit le JMP $9190 implanté dans CHRGET.
- SERMASK.SCE et SERMASK.OBJ : contraction des masques;
- DESERMASK.SCE.ORG$95A0 et DESM95A0 : restitution d'un masque réduit sous forme développée à l'écran;
  SAVPG4.SCE et SAVPG4.OBJ : sauvegarde des paramètres-périphériques de la page 4 avant affichage d'un masque;
- RESPG4.SCE et RESPG4.OBJ : restauration des paramètres dans la page 4 après affichage;
- LOCCOD.SCE et LOCCOD.OBJ : localisation du code d'un masque (à l'appel de]A) dans le bloc; donne à DESM95A0 l'adresse du "tampon";
- BLOC.U : regroupe toutes les routines (codes-objets) en un seul fichier pour un chargement plus rapide et plus simple. CRE-BLOC-U, comme son nom l'indique, peut servir à reconstituer BLOC.U à partir des différentes routines en cas de destruction accidentelle (une destruction volontaire nous causerait le plus grand chagrin...).

7) TEST-DESER-95A0 est un tout petit programme en Applesoft qui permet de vérifier, indépendamment du BASICIUM, la restitution d'un masque réduit à l'écran.

PARANA contient simplement le code hexa de JMP $9190 qui doit être chargé pour initialiser le BASICIUM. De même CHRGET contient la version "standard" de la routine CHRGET qui permet de revenir à l'Applesoft.

INIT.ANA.U vous permet, par un RUN, d'initialiser le BASICIUM sans avoir à passer par le programme START.

Il faut noter que le BASICIUM reste initialisé même après RESET et que seuls PR£6 ou un passage par START restaurent de façon simple l'Applesoft (la façon compliquée, si l'on peut dire, consiste à faire BLOAD CHRGET puis à remettre la valeur de HIMEM à 38400).

Programme DEMO-PGM : commentaires

Ligne 5 : HIMEM est placée sous le bloc DEMO-BLOC.
Ligne 20 : chargement du descriptif des tableaux (DEMO-VAR), du bloc de masques, et stockage de leurs adresses de début.
Ligne 25 : dimensionnement des tableaux de saisie (Z$, Y$ et X$) et du tableau des messages ZZ$ (ce dernier n'est dimensionné que parce qu'il dépasse 10 indices, car la ligne suivante suffirait autrement à le faire enregistrer par Applesoft).
Lignes 26 à 29 : lecture des messages, regroupés dans un seul tableau.
Ligne 30 : affichage du masque de menu (code D) et saisie du choix.
Ligne 100 : affichage du masque DEMO-M1 (code A), message ZZ$(2) avec "beep" (SAISIE), saisie de l'ensemble du tableau Z$, avec retour au menu si cette saisie est abandonnée (PEEK(6)=1).
Ligne 110 : "SAISIE CONFIRMEE ?" et passage à 150 dans l'affirmative.
Ligne 120 : on est toujours en saisie (ZZ$(2)) et l'on modifie les valeurs entrées pour Z$ (■MZ).
Ligne 150 : efface le tableau Z$ à l'écran et annonce la modification (message ZZ$(3) avec "beep").
Ligne 160 : modification, avec retour à la création si abandon.
Ligne 170 : voir ligne 110.
Ligne 180 : efface le tableau, annonce l'affichage, affiche Z$ à l'écran, puis le message ZZ$(13) avec demande de RETURN ou "?". Si c'est RETURN (PEEK(6)=0), on passe à 200.
Ligne 190 : "IMPRIMANTE BRANCHEE ?". Si oui, sélection du slot 1 et hard-copy (■H). Si votre imprimante ne fait pas normalement d'affichage simultané à l'écran, il faut supprimer PRINT CHR$(9)"80N".
Ligne 200 : réaffiche le masque et le tableau, demande un RETURN (la frappe de "?" ne serait pas prise en compte), efface Z$ et demande "VOULEZ-VOUS RECOMMENCER".
Ligne 1000 : affiche le masque DEMO-M2.
Lignes 1010 à 1110 : bien que le tableau Y$ se prête par nature à un INPUT complet, on l'utilise ici pour la démonstration des traitements "par indice". L'indice traité étant donné par la valeur de I, l'analyse de ces lignes ne devraient pas poser de problème.
Ligne 1115 : voir ligne 180.
Ligne 1120 : voir ligne 190.
Ligne 1130 : voir ligne 200.
Ligne 2000 : affiche le masque DEMO-M3 (code C) de l'impression paramétrée. A l'écran, les lignes à imprimer sont délimitées par les caractères en inverse. Demande ensuite "IMPRIMANTE BRANCHEE ?".
Ligne 2010 : le tableau X$ a été défini de telle manière que les variables X$(0) à X$(3) viennent s'insérer dans la ligne codée (D) à l'écran lors de l'instruction ■PX (il suffit pour cela de choisir les valeurs correctes de VTAB et HTAB lors de la création par CREATAB). La fin de la ligne 2010 assure donc l'impression du cadre titre de l'état et de la première ligne de données avec les valeurs prises dans Z$.
Ligne 2020 : la nouvelle ligne de valeurs est prise dans Y$, puis imprimée, et l'état se termine par les lignes (B) et (A).

Pour tout ce qui concerne les ordres d'impression, vous pouvez supprimer les PRINT CHR$(9)"80N" si votre imprimante ne fait pas d'affichage simultané à l'écran (ceci s'applique de façon générale, y compris au programme DEMO-PARA-AVEC FIC). Lorsque l'impression paramétrée est utilisée de manière répétitive, comme c'est le cas dans la démonstration "AVEC FIC", il vaut mieux réafficher le masque après chaque retour à l'écran, et avant ■P, car les ordres PRINT D1$ affectent l'écran, notamment par des suppressions de lignes, et perturbent par conséquent le fonctionnement de ■C.

En pratique, pour éviter les interférences écran/imprimante, il peut être utile de sélectionner cette dernière par une petite procédure comme :

```
50 VTAB 23:PRINT D1$"PRE1":PRINT CHR$(9)"80N";:RETURN:REM sous-
   routine de branchement.

1000 GOSUB 50:■H:PRINT D1$"PRE0" ou encore GOSUB 50:■CA:PRINT
D$"PRE0"...
```

Le BASICIUM version 80 colonnes

Les différents utilitaires, masques et routines composant le BASICIUM ont été adaptés à la carte 80 colonnes de l'Apple //e, pour constituer une version "80 colonnes". Tous les fichiers nécessaires sont également fournis sur la disquette.

La manipulation des utilitaires est parfaitement identique à celle de la version 40 colonnes et l'on retrouve les mêmes possibilités dans le menu de départ du BASICIUM 80 colonnes (gestion de masques, de descriptifs de tableaux de variables...). Les explications données précédemment restent donc valables ici.

L'utilisation des instructions du BASICIUM dans un programme sont également transparentes au choix "40 colonnes/80 colonnes". Vous disposez exactement des mêmes possibilités (INPUT de tableaux, messages, Hard-Copy...), avec une syntaxe inchangée. Les précautions éventuelles à prendre au niveau de vos programmes relèvent uniquement des "particularités" de la carte 80 colonnes. Pour éviter certains problèmes d'incompatibilité entre la carte et le DOS ou l'imprimante, nous vous conseillons ainsi de faire un "CALL 1002" avant tout appel à une commande DOS et de faire "CALL 49920", en lieu et place de "PR#3", au retour du DOS ou d'une impression, pour reprendre l'affichage à l'écran.
La re-initialisation de la carte (par CALL 49920, par exemple) provoque un HOME de l'écran; vous apprécierez donc tout particulièrement ici la possibilité que vous donne le BASICIUM d'afficher très rapidement un écran complet au moyen d'une instruction très courte (■A...). Reportez-vous au programme de démonstration 80 colonnes pour un exemple d'utilisation.

Tous les fichiers modifiés pour la version 80 colonnes sont reconnaissables sur la disquette à la présence du nombre 80 dans leur nom (IG2.80.SCE, ANA.U80.OBJ...).
L'ensemble des routines du BASICIUM est regroupé dans le fichier BLOC.U80 et l'initialisation se fait par RUN INIT.ANA.U80. A partir de l'un de vos programmes, si vous n'êtes pas certain que le BASICIUM est chargé au moment de son lancement, l'initialisation se fera de même dans les toutes premières lignes par un BLOAD BLOC.U80 suivi de BLOAD PARANA (système identique pour la version 40 colonnes).

BLOC.U80 commence à l'adresse 35760 (contre 35860 pour BLOC.U) en raison, entre autres, de la présence de la routine supplémentaire LDA80.OBJ, destinée à simuler la routine de lecture de mémoire d'écran de la carte 80 colonnes. Cette routine est utilisée lors des instructions d'impression (en effet, lorsque vous sélectionnez l'imprimante, la ROM de la carte imprimante prend la place de la ROM de la carte 80 colonnes en mémoire centrale, ce qui interdit d'employer directement les routines de la carte). Cette petite complication reste bien entendu transparente pour l'utilisateur au niveau du programme Basic.

Le BASICIUM sous ProDOS

Le BASICIUM est "en lui-même" compatible avec ProDOS, c'est-à-dire que les routines correspondant aux différentes instructions ne sont pas allergiques à la présence du ProDOS, et réciproquement, tant en 80 colonnes qu'en 40 colonnes. Toutefois, la méthode d'interprétation des commandes par ProDOS et la façon dont il gère la mémoire imposent quelques précautions supplémentaires par rapport au DOS 3.3.

Pour disposer d'un BASICIUM en ProDOS, vous devrez convertir sur une disquette ProDOS les fichiers suivants :

- 40 et 80 colonnes: PARANA.PRODOS (mise en place de l'aiguillage dans CHRGET)
  : CHRGET.PRODOS (retour à la routine standard)
- Pour 40 colonnes : BLOC.U
  : IN40.PRODOS (initialisation du BASICIUM)
- Pour 80 colonnes : BLOC.U80
  : INIT.PRODOS (initialisation version 80 colonnes)

Dans l'un et l'autre cas, la mise en route du BASICIUM se fera par un RUN du programme d'initialisation adéquat et le retour à l'Applesoft par BRUN CHRGET.PRODOS.

En ce qui concerne les blocs de masques et les tableaux de variables nécessaires à vos programmes, il faudra les préparer et les finaliser sous DOS 3.3 au moyen des utilitaires proposés, puis les convertir sous ProDOS. Une fois les masques et variables disponibles, il ne vous restera plus qu'à écrire les programmes directement sous ProDOS. Pour cette dernière étape, plusieurs points réclament votre attention :

1) Ne mettez pas de HIMEM au début du programme avant de BLOADer les blocs de masques et tableaux de variables. Laissez le HIMEM à la valeur que lui donne le programme d'initialisation du BASICIUM (38400).
2) Vous ne pourrez plus utiliser PEEK(43634) et PEEK(43635) pour l'adresse de début des fichiers BLOADés mais devrez POKEr les adresses "en clair".
3) Fixez le HIMEM en-dessous des blocs et tableaux après les avoir chargés.
4) Si vous voulez définir une variable D$ contenant CTRL-D, faites-le par D$="", en tapant un CTRL-D entre guillemets, et non par D$=CHR$(4).
5) La méthode d'interprétation des commandes par ProDOS provoque des "SYNTAX ERRORs" en sortie des instructions du BASICIUM. Pour éviter ces problèmes, il faut déconnecter ProDOS avant d'utiliser ces dernières, au moyen des instructions "CALL 39936 : NOTRACE".

6) Pour utiliser les commandes d'impression (]H et]C), ProDOS étant déconnecté, il faut sélectionner l'imprimante par PR#s et non par PRINT D$"PR#s". Le retour à l'écran se fera par CALL 49920 avec la carte 80 colonnes et par PR#0 en 40 colonnes.
7) Pour reconnecter ProDOS avant de lui faire appel, utilisez les instructions "CALL 40071 : PRINT". Avant de revenir à une portion de programme en BASICIUM, refaites alors "CALL 39936 : NOTRACE".
8) Si vous devez BLOADer des fichiers en cours de programme, remontez d'abord HIMEM à 38400 puis redonnez-lui sa valeur initiale à l'issue du chargement (si vous oubliez cette précaution, une partie du BASICIUM se fera "écraser"...).

Vous trouverez ci-après un petit programme d'illustration de l'emploi du BASICIUM 80 colonnes sous ProDOS.

Réaumé des instructions

]F : nettoyage mémoire (remplace le FRE(0) de l'Applesoft).

]QMEi ou]Q"MEc" : pose la question dont le libellé est donné par ME$(i) ou par ME$(j) avec j=ASC(c)-48 (c est un caractère comme "A" ou ";"...). Attend une réponse O ou N suivis de RETURN uniquement.
]RMEi ou]R"MEc" : affiche le message dont le libellé est donné comme ci-dessus et attend une réponse ne pouvant être que RETURN ou "?".
]BMEi ou]B"MEc" : affiche le message dont le libellé est donné comme ci-dessus, avec deux "beep", puis l'efface après lecture.

]H : hard-copy d'écran texte.

]AC : affiche à l'écran le masque dont le code est C dans le bloc de masques chargé en mémoire.

]CC : imprime la ligne de l'écran qui est comprise entre deux caractères C affichés en INVERSE.

]IZ : saisie du tableau de variables Z$ précédemment défini au moyen de CRETAB.
]MZ : modification du tableau Z$.
]PZ : affiche le contenu des variables du tableau Z$.
]EZ : efface à l'écran le contenu affiché des variables du tableau Z$.

]UiIZ ou]U"cIZ" : saisie de l'élément du tableau Z$ dont l'indice est donné par i ou par ASC(c)-48 (c est un caractère comme "A" ou ";" ou "="...).
]UiMZ ou]U"cMZ" : modification de l'élément du tableau Z$ dont l'indice est donné comme ci-dessus.
]UiPZ ou]U"cPZ" : affichage de l'élément du tableau Z$ dont l'indice est donné comme ci-dessus.
]UiEZ ou]U"cEZ" : efface à l'écran le contenu de l'élément du tableau Z$ dont l'indice est donné comme ci-dessus.

Différences entre les routines 80 et 40 colonnes

Les listes sources de toutes les routines en 40 colonnes vous sont données en annexe, ainsi que les sources de LDA.OBJ qui n'existe que dans la version 80 colonnes et de la routine de gestion de masques 80 colonnes qui n'a pas été publiée dans Pom's. Vous trouverez ci-après les modifications apportées aux sources 40 colonnes pour obtenir le BASICIUM 80 colonnes.

1) INPUT généralisé de tableaux

Ligne 10 : H EQU $57B remplace H EPZ $24
Lignes 16 et 17 : supprimées
Ligne 26 : TAB2 EPZ $EE remplace TAB2 EPZ $1E
Lignes 365 à 368 : remplacées par LDY H
JSR $CF01
En fin de programme, rajouter les routines suivantes :

```
390 ;ICI COMMENCENT LES ROUTINES DE SAISIE ET
391 ;D'AFFICHAGE DE CARACTERES CONCUES POUR
392 ;LA ROM DE LA CARTE 80 COLONNES A//E
393 GET     TYA
394         PHA
395 GET0    LDY H
396         JSR $CF01
397         PHA
398         LDA #$DF
399         JSR $CEF2
400         LDA $C000
401         BMI GET1
402         PLA
403         JSR $CEF2
404         JMP GET0
405 GET1    STA $C010
406         STA $67B
407         PLA
408         JSR $CEF2
409         PLA
410         TAY
411         LDA $67B
412         RTS
413 OUT     STY $2F
414         LDY H
415         JSR $CEF2
416         INC H
417         LDY $2F
418         RTS
```

2) Routine de Hard-Copy

Ligne 7 : ORG $8BE0 remplace ORG $9114
Ligne 16 : JSR $8BB0 remplace LDA (ADR),Y
Ligne 28 : CPY £$50 remplace CPY £$28

3) Impression paramétrée

Ligne 18 : JSR $8BB0 remplace LDA (ADR),Y
Ligne 32 : CPY £$50 remplace CPY £$28

4) Gestion de message (QRB)

```
Ligne 12 : TAB2 EPZ $EE remplace TAB2 EPZ $1E
Ligne 16 : H EQU $57B remplace H EPZ $24
Ligne 18 : OUT EQU $912B remplace OUT EQU $FDED
Ligne 19 : GET EQU $9101 remplace GET EQU $FDOC
Ligne 71 : rajouter une étiquette SB0 devant LDA £0
Ligne 83 : la remplacer par BNE SC0
                            DEC H
                            JMP SB0
Ligne 84 : rajouter une étiquette SC0 devant CMP £$8D
```

5) Traitement "IG" sur un indice

```
Ligne 26 : LDA $EE remplace LDA $1E
Ligne 29 : STA $EE remplace STA $1E
Ligne 31 : INC $EF remplace INC $1F
Ligne 42 : JSR $8FEA remplace JSR $8FE7
Ligne 54 : JSR $904D remplace JSR $904A
```

6) Interpréteur BASICIUM

```
Ligne 36 : JSR $8BE0 remplace JSR $9114
Ligne 71 : JSR $8D34 remplace JSR $8D2C
Ligne 76 : JSR $8D56 remplace JSR $8D4D
```

7) Compression des masques

```
Ligne 21 : LDA (BASE),Y devient JSR $CF01
                                STA $8
Ligne 23 : CMP (BASE),Y devient JSR $CF01
                                CMP $8
Ligne 26 : idem ligne 23
Ligne 30 : idem ligne 23
Ligne 33 : CPY £$4F remplace CPY £$27
Après la ligne 36, rajouter LDA $8
Ligne 45 : CPY £$4D remplace CPY £$25
Ligne 47 : CPY £$50 remplace CPY £$28
Ligne 49 : idem ligne 21
Après la ligne 53, rajouter LDA $8
```

8) Restitution des masques

```
Ligne 11 : CH EQU $57B remplace CH EPZ $24
Ligne 24 : JSR $CEF2 remplace STA (BASE),Y
Ligne 27 : CPY £$50 remplace CPY £$28
Ligne 40 : idem ligne 24
```

```
1  TEXT : HOME
5  HIMEM: 34550
10 D$ =  CHR$ (4):D1$ =  CHR$ (13) + D$
20  PRINT D$"BLOAD DEMO-VAR": POKE 37191, PEEK (43634): POKE 37192
     , PEEK (43635): PRINT D$"BLOAD DEMO-BLOC": POKE 37254, PEEK (
     43634): POKE 37255, PEEK (43635)
25  DIM Z$(11),Y$(13),X$(3),ZZ$(14)
26  FOR I = 0 TO 13: READ ZZ$(I): NEXT
28  DATA  SAISIE CONFIRMEE,MODIFICATION CONFIRMEE,SAISIE,MODIFICAT
     ION,AFFICHAGE,ON EFFACE APRES LE RETURN,VOULEZ-VOUS RECOMMENC
     ER,ON RENTRE L'INDICE,ON MODIFIE L'INDICE,ON AFFICHE L'INDICE

29  DATA    ON EFFACE L'INDICE,REAFFICHAGE DU TABLEAU COMPLET,IMPR
     IMANTE BRANCHEE,'RETURN' OU '?' POUR HARD-COPY
30 :JAD: VTAB 21: HTAB 26: GET C$:C =  VAL (C$): IF C < 1 OR C > 4
      THEN 30
40  IF C = 4 THEN  TEXT : HOME : PRINT D1$"RUN START"
50  ON C GOTO 100,1000,2000
100 :JAA:JBZZ2:JIZ: IF  PEEK (6) THEN 30
110 :JQZZ0: IF  PEEK (6) THEN 150
120 :JBZZ2:JMZ: IF  PEEK (6) THEN 30
130  GOTO 110
150 :JEZ:JBZZ3
160 :JMZ: IF  PEEK (6) THEN 100
170 :JQZZ1: IF  NOT  PEEK (6) THEN 160
180 :JEZ:JBZZ4:JPZ:JR"ZZ=": IF  NOT  PEEK (6) THEN 200
190 :JQ"ZZ<": IF  PEEK (6) THEN  POKE 34,24: PRINT D1$"PR#1": PRINT
      CHR$ (9)"80N":JH: PRINT D1$"PR#0": POKE 34,1
200 :JAA:JPZ:JRZZ5:JEZ:JQZZ6: IF  PEEK (6) THEN 100
210  GOTO 30
1000 :JAB:ZZ$ = ZZ$(7): GOTO 1010
1005 ZZ$(14) = ZZ$ + " " +  STR$ (I):JB"ZZ)": RETURN
1010 I = 0: GOSUB 1005:JU0IY: IF  PEEK (6) THEN 30
1020 I = 1: GOSUB 1005:JU1IY: IF  PEEK (6) THEN 1010
1030 I = 2: GOSUB 1005:JU2IY: IF  PEEK (6) THEN 1020
1040 I = 3: GOSUB 1005:JU3IY: IF  PEEK (6) THEN 1030
1050 I = 10: GOSUB 1005:JU:IY: IF  PEEK (6) THEN 1040
1060 I = 11: GOSUB 1005:JU";IY": IF  PEEK (6) THEN 1050
1070 I = 12: GOSUB 1005:JU"<IY": IF  PEEK (6) THEN 1060
1080 I = 13: GOSUB 1005:JU"=IY": IF  PEEK (6) THEN 1080
1090 ZZ$ = ZZ$(8):I = 0: GOSUB 1005:JU0MY: IF  PEEK (6) THEN 1000
1100 I = 13: GOSUB 1005:JU"=MY": IF  PEEK (6) THEN 1090
1110 ZZ$ = ZZ$(9):JEY:I = 1: GOSUB 1005:JU1PY:I = 12: GOSUB 1005:J
     U"<PY":ZZ$ = ZZ$(10): GOSUB 1005:JU"<EY":I = 1: GOSUB 1005:JU
     1EY
1115 :JEY:JB"ZZ;":JPY:JR"ZZ=": IF  NOT  PEEK (6) THEN 1130
1120 :JQ"ZZ<": IF  PEEK (6) THEN  POKE 34,24: PRINT D1$"PR#1": PRINT
      CHR$ (9)"80N":JH: PRINT D1$"PR#0": POKE 34,1
1130 :JAB:JPY:JQZZ6: IF  PEEK (6) THEN 1000
1140  GOTO 30
2000 :JAC:JQ"ZZ<": IF  NOT  PEEK (6) THEN 30
2010  POKE 34,24: FOR I = 0 TO 3:X$(I) = Z$(2 + I): NEXT :JPX: PRINT
     D1$"PR#1": PRINT  CHR$ (9)"80N":JCA:JCB:JCC:JCB:JCA:JCB:JCD
2020  PRINT D$"PR#0":X$(0) = Y$(0):X$(1) = Y$(2):X$(2) = Y$(10):X$
     (3) = Y$(12):JPX: PRINT D1$"PR#1": PRINT  CHR$ (9)"80N";:JCD:
     JCB:JCA: PRINT D1$"PR#0": GOTO 30
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;* INPUT GENERALISE DE TABLEAUX  *
 4 ;*        CODE = I62.OBJ         *
 5 ;*                               *
 6 ;*********************************
 7        ORG $8E20
 8        OBJ $800
 9 ADR    EPZ $28            ;SE REPORTER A LA ROUTINE PUBLIEE
10 H      EPZ $24            ;DANS POM'S POUR LES COMMENTAIRES
11 LC     EPZ $9
12 E      EPZ $6
13 LO     EPZ $7
14 TY     EPZ $8
15 ZY     EQU $200
16 GET    EQU $FD0C
17 OUT    EQU $FDED
18 ADT    EPZ $18
19 VAR    EPZ $1A
20 VP     EPZ $6
21 ADT2   EPZ $1B
22 NVAR   EPZ $1D
23 V      EPZ $25
24 ME     EPZ $D7
25 TAB    EPZ $6B
26 TAB2   EPZ $1E
27 VTAB   EQU $FC22
28 ADV    EPZ $CE
29 FRETOP EPZ $6F
30 NVT    EPZ $D6
31 FRE    EQU $927C          ;SEULE EST RAJOUTEE LA VARIABLE
32 NVP    EPZ $EB            ; TI POUR TENIR COMPTE DES
33 VAR2   EPZ $EC            ;INSTRUCTIONS FONCTIONNANT PAR
34 TI     EPZ $ED            ;PAR INDICE (JU)
35        LDY #$80
36        STY VAR2
37 EXA0   STA TI
38        CMP #'I'
39        BNE EXA1
40        JMP I0
41 EXA1   CMP #'P'
42        BNE EXA2
43        JMP P0
44 EXA2   CMP #'M'
45        BEQ M0
46        CMP #'E'
47        BEQ E0
48        RTS
49        JMP M0
50 I0     JSR $B1
51        STA VAR
52        LDA #0
53        STA ME
54        STA NVP
55        JMP I1
56 P0     JSR $B1
57        STA VAR
58        JMP SS0
59 M0     JSR $B1
60        STA VAR
61        LDA #1
62        STA ME
63        JMP SSS0
64 E0     JSR $B1
65        STA VAR
66        JMP E1
67 I1     JSR S0
68        JMP S60
69 S0     LDY #0
70        LDA (ADT),Y
71        CMP VAR
72        BEQ S1
73        INY
74        INY
75        LDA (ADT),Y
76        CLC
77        ADC ADT
78        STA VP
79        INY
80        LDA (ADT),Y
81        ADC ADT+1
82        STA ADT+1
83        LDA VP
84        STA ADT
85        JMP S0
86 S1     LDA ADT
87        STA ADT2
88        LDA ADT+1
89        STA ADT2+1
90        INY
91        LDA (ADT),Y
92        STA NVAR
93        LDA #0
94        STA NVT
95        LDA TAB
96        STA TAB2
97        LDA TAB+1
98        STA TAB2+1
99 S2     LDY #0
100       LDA (TAB2),Y
101       CMP VAR
102       BNE S3
103       INY
104       LDA (TAB2),Y
105       CMP VAR2
106       BEQ S4
107       DEY
108 S3    INY
109       INY
110       LDA (TAB2),Y
111       CLC
112       ADC TAB2
113       STA VP
114       INY
115       LDA (TAB2),Y
116       ADC TAB2+1
117       STA TAB2+1
118       LDA VP
119       STA TAB2
120       JMP S2
```

```
121 S4      LDA TAB2
122         CLC
123         ADC #7
124         STA TAB2
125         BCC S5
126         INC TAB2+1
127 S5      RTS
128 S60     JSR S6
129         JMP S70
130 S6      LDA ADT2
131         CLC
132         ADC #4
133         STA ADT2
134         BCC S61
135         INC ADT2+1
136 S61     RTS
137 S70     JSR S7
138         JMP S71
139 S7      LDY #0
140         LDA (ADT2),Y
141         STA TY
142         INY
143         LDA (ADT2),Y
144         STA LO
145         INY
146         LDA (ADT2),Y
147         STA V
148         INY
149         LDA (ADT2),Y
150         STA H
151         STA VP
152         JSR VTAB
153         RTS
154 S71     LDX LO
155         LDA #$AE
156 S8      JSR OUT
157         DEX
158         BNE S8
159         LDA VP
160         STA H
161         LDA ME
162         BNE S80
163         LDA NVT
164         CMP NVP
165         BCC S80
166         LDA #0
167         STA NVP
168         JMP S90
169 S80     JSR SS10
170         LDA VP
171         STA H
172 S90     JSR INPUT
173         LDA E
174         BEQ S9
175         JMP S16
176 S9      LDX LC
177         BNE S11
178 S10     LDA ZY,X
179         CMP #$80
180         BEQ S11
181         JSR OUT
182         INC LC
183         INX
184         JMP S10
185 S11     JSR S11A
186         JMP S110
187 S11A    CPX LO
188         BEQ S11B
189         LDA #$A0
190         JSR OUT
191         INX
192         JMP S11A
193 S11B    RTS
194 S110    LDY #0
195         LDA LC
196         STA (TAB2),Y
197         BEQ S12
198         LDA FRETOP
199         SEC
200         SBC LC
201         STA FRETOP
202         LDA FRETOP+1
203         SBC #0
204         STA FRETOP+1
205 S13     LDA ZY,Y
206         SEC
207         SBC #$80
208         STA (FRETOP),Y
209         INY
210         CPY LC
211         BNE S13
212         LDY #1
213         LDA FRETOP
214         STA (TAB2),Y
215         LDA FRETOP+1
216         INY
217         STA (TAB2),Y
218 S12     INC NVT
219         LDA NVT
220         CMP NVAR
221         BNE S14
222         JSR FRE
223         RTS
224 S14     LDA TAB2
225         CLC
226         ADC #3
227         STA TAB2
228         BCC S15
229         INC TAB2+1
230 S15     JMP S60
231 S16     LDA TI              ;CAS PARTICULIER DES INSTRUCTIONS
232         CMP #'U'            ;PAR INDICES (JU)
233         BEQ S161
234 S160    LDA NVT
235         BNE S17
236 S161    JSR FRE
237         RTS
238 S17     LDX NVP
239         BNE S170
240         STA NVP
```

```
241 S170   DEC NVT
242        LDA TAB2
243        SEC
244        SBC #3
245        STA TAB2
246        LDA TAB2+1
247        SBC #0
248        STA TAB2+1
249        LDA ADT2
250        SEC
251        SBC #4
252        STA ADT2
253        LDA ADT2+1
254        SBC #0
255        STA ADT2+1
256        JMP S70
257 SS0    JSR SS00
258        RTS
259 SS00   JSR S0
260 SS1    JSR S6
261        JSR S7
262        JSR SS10
263        JMP SS3
264 SS10   LDX #0
265        JSR S11A
266        JSR S7
267        LDY #0
268        LDA (TAB2),Y
269        STA LC
270        BEQ SS20
271        INY
272        LDA (TAB2),Y
273        STA ADV
274        INY
275        LDA (TAB2),Y
276        STA ADV+1
277        LDY #0
278 SS2    LDA (ADV),Y
279        CLC
280        ADC #$80
281        STA ZY,Y
282        JSR OUT
283        INY
284        CPY LC
285        BNE SS2
286 SS20   LDA #$80
287        STA ZY,Y
288        RTS
289 SS3    INC NVT
290        LDA NVT
291        CMP NVAR
292        BNE SS4
293        RTS
294 SS4    LDA TAB2
295        CLC
296        ADC #3
297        STA TAB2
298        BCC SS5
299        INC TAB2+1
300 SS5    JMP SS1
```

```
301 SSS0   JSR SS00
302        LDY #0
303        JSR S1
304        JMP S60
305 E1     JSR S0
306 E4     JSR S6
307        JSR S7
308        LDX #0
309        JSR S11A
310        INC NVT
311        LDA NVT
312        CMP NVAR
313        BNE E2
314        RTS
315 E2     LDA TAB2
316        CLC
317        ADC #3
318        STA TAB2
319        BCC E3
320        INC TAB2+1
321 E3     JMP E4
322 INPUT  LDA ME
323        BNE DEP
324        LDA NVT
325        CMP NVP
326        BCC DEP
327        LDA #$80
328        LDX LO
329 DEP0   STA ZY,X
330        DEX
331        BPL DEP0
332 DEP    LDX #0
333        STX E
334 DEP1   JSR GET
335        CMP #$80
336        BEQ FIN1
337        CPX LO
338        BCC SUIT5
339        CMP #$88
340        BEQ RETOUR
341        JSR $FBDD
342        JMP DEP1
343 SUIT5  CMP #$9B
344        BNE SUIT1
345        CPX #0
346        BNE SUIT1
347        LDA #9
348        STA E
349        RTS
350 SUIT1  CMP #$AC
351        BNE SUIT2
352        LDA #$AE
353        JMP SUIT8
354 SUIT2  CMP #$AE
355        BEQ SUIT8
356        CMP #$88
357        BNE SUIT3
358        CPX #0
359        BEQ SUIT3
360 RETOUR DEX
```

```
361        DEC H
362        JMP DEP1
363 SUIT3  CMP #$95
364        BNE SUIT4
365        LDA $25
366        JSR $FBC1
367        LDY H
368        LDA (ADR),Y
369        JMP SUIT8
370 SUIT4  CMP #$A0
371        BCC DEP1
372        CMP #$DB
373        BCS DEP1
374        LDY TY
375        CPY #1
376        BEQ SUIT6
377        CMP #$AF
378        BCS SUIT7
379        CMP #$A0
380        BNE DEP1
381 SUIT7  CMP #$BA
382        BCS DEP1
383 SUIT6  CPX #0
384        BNE SUIT8
385        CMP #$A2
386        BEQ DEP1
387 SUIT8  JSR OUT
388        STA ZY,X
389        INX
390        JMP DEP1
391 FIN1   STX LC
392        RTS
393        DCM 'INT'
394        END
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;*    IMPRESSION PARAMETREE      *
 4 ;*       CODE = PARA2.OBJ        *
 5 ;*                               *
 6 ;*********************************
 7        ORG $9150
 8        OBJ $800
 9 V      EPZ $25
10 ADR    EPZ $28
11 C      EPZ $6
12 PR     EQU $FDED
13        LDA #0
14        STA V
15        LDX #0
16 HP4    LDY #0
17        JSR $FBC1
18 HP3    LDA (ADR),Y
19        CMP C
20        BNE HP1
21        CPX #1
22        BNE HP0
23        LDA #$80
24        JSR PR
25        RTS
26 HP0    LDX #1
27        JMP HP2
28 HP1    CPX #1
29        BNE HP2
30        JSR PR
31 HP2    INY
32        CPY #$28
33        BNE HP3
34        INC V
35        LDA V
36        CMP #$18
37        BNE HP4
38        RTS
39        DCM 'INT'
40        END
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;*    ROUTINE DE HARD-COPY       *
 4 ;*       CODE = HCT2.OBJ         *
 5 ;*                               *
 6 ;*********************************
 7        ORG $9114
 8        OBJ $800
 9 V      EPZ $25
10 ADR    EPZ $28
11 PR     EQU $FDED
12        LDA #0
13        STA V
14 HC0    LDY #0
15        JSR $FBC1
16 HC1    LDA (ADR),Y
17        CMP #$40
18        BCS HC2
19        CMP #$20
20        BCS HC3
21        CLC
22        ADC #$C0
23        JMP HC2
24 HC3    CLC
25        ADC #$80
26 HC2    JSR PR
27        INY
28        CPY #$28
29        BNE HC1
30        LDA #$8D
31        JSR PR
32        INC V
33        LDA V
34        CMP #$18
35        BNE HC0
36        RTS
37        DCM 'INT'
38        END
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;*     QUESTION A REPONSE O/N    *
 4 ;*     MESSAGE =) RETURN OU ?    *
 5 ;*        MESSAGE AVEC BEEP      *
 6 ;*          CODE = ORB.OBJ       *
 7 ;*                               *
 8 ;*********************************
 9         ORG $8CA0
10         OBJ $800
11 IND     EPZ $ED          ;INDICE DANS LE TABLEAU
12 TAB2    EPZ $1E          ;DEBUT TABLEAU DANS APPLESOFT
13 LC      EPZ $9           ;VOIR IG2.SCE
14 ADV     EPZ $CE
15 V       EPZ $25
16 H       EPZ $24
17 VTAB    EQU $FC22
18 OUT     EQU $FDED
19 GET     EQU $FD0C
20 E       EPZ $6
21 CLER    EQU $FC9C        ; CALL -868
22 RT      EQU $8E9E        ;RECHERCHE TABLEAU (SOUS-ROUTINE DE IG2)
23         JSR RT           ;DEBUT INSTRUCTION "O"
24         JSR SS0
25         JMP SS1
26 SS0     LDY IND
27 S0      BEQ S1           ;POSITIONNE EN DEBUT DE L'INDICE
28         LDA TAB2         ;DANS LE TABLEAU APPLESOFT
29         CLC
30         ADC #3
31         STA TAB2
32         BCC S2
33         INC TAB2+1
34 S2      DEY
35         JMP S0
36 S1      LDY #0
37         LDA (TAB2),Y
38         STA LC           ;LONGUEUR DE LA CHAINE DE CARACTERES
39         BEQ S5
40         INY
41         LDA (TAB2),Y
42         STA ADV          ;ADV ET ADV+1 = ADRESSE DE LA CHAINE
43         INY
44         LDA (TAB2),Y
45         STA ADV+1
46         LDA #$16         ;EQUIVALENT DE VTAB 22 - HTAB 1
47         STA V
48         JSR $FBC1
49         LDA #0
50         STA H
51         JSR VTAB
52         LDY #0
53         LDA #$3F         ;PASSE EN MODE INVERSE
54         STA $32
55 S4      LDA (ADV),Y      ;AFFICHE LE MESSAGE
56         CLC
57         ADC #$80
58         JSR OUT
59         INY
60         CPY LC
61         BNE S4
62         LDA #$FF         ;RETOUR AU MODE NORMAL
63         STA $32
64 S5      RTS
65 SS1     LDA #$A0         ;AFFICHE " ? "
66         JSR OUT
67         LDA #$BF
68         JSR OUT
69         LDA #$A0
70 SB      JSR OUT
71         LDA #0
72         STA E
73 S6      JSR GET
74         CMP #$CE         ; REPONSE = N ?
75         BEQ S7
76         CMP #$CF         ; REPONSE = O ?
77         BNE S6
78         LDY #1           ;SIGNALE LE O POUR LE BASIC
79         STY E
80 S7      JSR OUT
81 S70     JSR GET
82         CMP #$88         ; FLECHE A GAUCHE ?
83         BEQ SB
84         CMP #$8D         ; RETURN ?
85         BNE S70
86         LDA #0           ;EFFACE LA LIGNE
87         STA H
88         JSR CLER
89         RTS
90         JSR RT           ;DEBUT INSTRUCTION "R"
91         JSR SS0
92         LDA #0
93         STA E
94 T0      JSR GET
95         CMP #$8D         ; RETURN ?
96         BEQ T1
97         CMP #$BF         ; "?" ?
98         BNE T0
99         LDY #1           ;SIGNALE LE ? POUR LE BASIC
100        STY E
101 T1     LDA #0
102        STA H
103        JSR CLER
104        RTS
105        JSR $FBDD        ; DEBUT INSTRUCTION "B" (BEEP)
106        JSR RT
107        JSR SS0
108        JSR $FBDD
109        LDY #$10         ; PAUSE
110 T2     LDA #0
111        JSR $FCA8
112        DEY
113        BNE T2
114        JMP T1
115        DCM "INT"
116        END
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;* TRAITEMENT '16" SUR UN INDICE *
 4 ;*        CODE = IG.U.OBJ        *
 5 ;*                               *
 6 ;*********************************
 7        ORG $8D70
 8        OBJ $800
 9        LDY #3
10        LDA ($B8),Y
11        STA $1A          ;1ERE LETTRE NOM DE LA VARIABLE
12        LDA #$80
13        STA $EC          ;2EME LETTRE NOM DE LA VARIABLE
14        JSR $8E70        ;CHERCHE TABLEAU DANS DESCRIPTIF
15 ; ET DANS ZONE APPLESOFT (SOUS-ROUTINE DE IG2)
16        JSR $B1
17        SEC
18        SBC #$30         ;DONNE L'INDICE
19        STA $D6          ; NVT
20        STA $EB          ; NVP
21        STA $1D          ; NVAR
22        INC $1D
23        TAX
24 U2     BEQ U0           ;PLACE AU DEBUT DE L'INDICE DANS
25        JSR $8EE1        ;DESCRIPTIF ET TABLEAU APPLESOFT
26        LDA $1E
27        CLC
28        ADC #3
29        STA $1E
30        BCC U1
31        INC $1F
32 U1     DEX
33        JMP U2
34 U0     STX $D7          ;ME=0 (VOIR IG2)
35        JSR $B1
36        CMP #'I'
37        BNE U3
38        JSR $8EDB
39        RTS
40 U3     CMP #'P'
41        BNE U4
42        JSR $8FE7
43        RTS
44 U4     CMP #'M'
45        BNE U5
46        JSR $8EE1
47        LDA #1
48        STA $D7
49        JSR $8EED
50        RTS
51 U5     CMP #'E'
52        BEQ U6
53        RTS
54 U6     JSR $904A
55        RTS
56        DCM 'INT'
57        END
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;*     INTERPRETEUR BASICIUM     *
 4 ;*       CODE = ANA.U.OBJ        *
 5 ;*                               *
 6 ;*********************************
 7        ORG $9190
 8        OBJ $800
 9        CMP #$50      ;CARACTERE "]" ?
10        BEQ A1
11        CMP #$3A      ; TRAITEMENT STANDARD DE CHRGET
12        BCS A0
13        JMP $BE
14 A0     JMP $C8
15 A1     JSR $B1       ;LECTURE CARACTERE SUIVANT LE ]
16        CMP #'F'
17        BNE A2
18        JSR $927C     ;ROUTINE NETTOYAGE-MEMOIRE
19        JMP A21
20 A2     LDX $9147     ;INITIALISE AOT POUR IG2
21        STX $18       ;$9147=37191 ET $9148=37192
22        LDX $9148
23        STX $19
24        CMP #'I'
25        BNE A3
26 A20    JSR $8E20     ;ROUTINE IG2
27 A21    JMP $B1       ;RETOUR A INTERPRETEUR APPLESOFT
28 A3     CMP #'P'
29        BEQ A20
30        CMP #'M'
31        BEQ A20
32        CMP #'E'
33        BEQ A20
34        CMP #'H'
35        BNE A4
36        JSR $9114     ;ROUTINE HARD-COPY
37        JMP A21
38 A4     CMP #'C'
39        BNE A5
40        JSR $B1       ;CARAC. SUIVANT = PARAMETRE IMPRESSIO
41        SEC           ;TRANSFORME EN CODE ECRAN INVERSE
42        SBC #$40
43        STA $6
44        JSR $9150     ;IMPRESSION PARAMETREE
45        JMP A21
46 A5     CMP #'O'
47        BNE A6
48        JSR A50
49        JMP A51
50 A50    LDY #1        ;RECHERCHE " EVENTUEL
51        LDA ($B8),Y
52        CMP #$22
53        BNE A500
54        JSR $B1       ;AVANCE POINTEUR DERRIERE LE "
55 A500   JSR $B1
56        STA $1A       ;1ERE LETTRE NOM DU TABLEAU
57        JSR $B1
```

```
58          CLC
59          ADC #$80
60          STA $EC     ;2EME LETTRE NOM DU TABLEAU
61          JSR $B1
62          SEC
63          SBC #$30
64          STA $ED     ;INDICE MESSAGE DANS LE TABLEAU
65          RTS
66 A51      JSR $8CA0   ;ROUTINE INSTRUCTION "Q"
67          JMP A81
68 A6       CMP #'R'
69          BNE A7
70          JSR A50
71          JSR $8D2C   ;ROUTINE INSTRUCTION "R"
72          JMP A81
73 A7       CMP #'B'
74          BNE A8
75          JSR A50
76          JSR $8D4D   ;ROUTINE INSTRUCTION "B"
77          JMP A81
78 A8       CMP #'U'
79          BNE A9
80          STA $ED     ;TYPE INSTRUCTION POUR I62
81          LDY #1      ;RECHERCHE " EVENTUEL
82          LDA ($B8),Y
83          CMP #$22
84          BNE A80
85          JSR $B1     ;AVANCE POINTEUR DERRIERE LE "
86 A80      JSR $8D70   ;ROUTINE I6.U
87          JSR $B1     ;POSITIONNE CORRECTEMENT LE POINTEUR
88 A81      LDY #1      ;RECHERCHE LE 2EME " EVENTUEL
89          LDA ($B8),Y
90          CMP #$22
91          BNE A82
92          JSR $B1     ;AVANCE LE POINTEUR DERRIERE LE "
93 A82      JMP A21
94 A9       CMP #'A'
95          BEQ A90
96          JMP A21
97 A90      JSR $B1
98          STA $6      ;CODE DU MASQUE
99          JSR $8C14   ;ROUTINE LOCCOD
100         JSR $8DCF   ;ROUTINE SAVPG4
101         JSR $95A0   ;ROUTINE DESM95A0
102         JSR $8C64   ;ROUTINE RESPG4
103         JMP A21
104         DCM "INT"
105         END
```

```
1  ;*********************************
2  ;*                               *
3  ;*  SAUVEGARDE DES 64 OCTETS     *
4  ;* RESERVES AUX PERIPHERIQUES    *
5  ;*       DANS LA PAGE 4          *
6  ;*      CODE = SAVPG4.OBJ        *
7  ;*                               *
8  ;*********************************
9          ORG $3DCF
10         OBJ $300
11 PG4     EPZ $6              ;ADRESSE BASE PAGE 4 A SAUVER
12 IN      EQU $9418           ;ZONE DE SAUVEGARDE PARAMETRES
13         LDA #$78
14         STA PG4
15         LDA #4
16         STA PG4+1
17         LDX #0
18 S1      LDY #0
19 S0      LDA (PG4),Y         ;TRANSFERT DES 8 OCTETS
20         STA IN,X            ;RESERVES AUX PERIPHERIQUES
21         INX
22         INY
23         CPY #8
24         BNE S0
25         LDA PG4             ;PASSE A ADRESSE-BASE SUIVANTE
26         CLC
27         ADC #$80
28         STA PG4
29         BCC S2
30         INC PG4+1
31 S2      LDA PG4+1           ;DERNIERE ADRESSE-BASE = $7F8
32         CMP #8
33         BNE S1
34         RTS
35         DCM "INT"
36         END
```

```
1  ;*********************************
2  ;*                               *
3  ;* RESTITUTION EN PAGE 4 DES 64  *
4  ;*   OCTETS DE PERIPHERIQUES     *
5  ;*      CODE = RESPG4.OBJ        *
6  ;*                               *
7  ;*********************************
8          ORG $8C64
9          OBJ $300
10 PG4     EPZ $6              ;VOIR SAVPG4
11 IN      EQU $9418
12         LDA #$78
13         STA PG4
14         LDA #4
15         STA PG4+1
16         LDX #0
17 T1      LDY #0
18 T0      LDA IN,X
19         STA (PG4),Y
20         INX
21         INY
22         CPY #8
23         BNE T0
24         LDA PG4
25         CLC
26         ADC #$80
27         STA PG4
28         BCC S2
29         INC PG4+1
30 S2      LDA PG4+1
31         CMP #8
32         BNE T1
33         RTS
34         DCM "INT"
35         END
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;* ROUTINE DE GESTION DE MASQUES *
 4 ;*     VERSION 80 COLONNES       *
 5 ;*      CODE = MASK.80.OBJ       *
 6 ;*                               *
 7 ;*                               *
 8 ;*********************************
 9 ;
10        ORG $86E4
11        OBJ $800
12 V2     EQU $18
13 H2     EQU $19
14 V1     EPZ $8
15 H1     EPZ $9
16 M      EPZ $7      ;DRAPEAU POUR INVERSE
17 C      EPZ $6
18 ADR    EPZ $28     ;ADRESSE 1ERE COLONNE
19 H      EQU $57B
20 V      EPZ $25     ;POSITION VER. CURSEUR
21 VTAB   EQU $FC22   ;DEPLACE CURSEUR EN V
22 HOME   EQU $FC58
23        JSR HOME
24        LDA #0
25        STA H
26        STA M
27 DEB    LDA V
28        CMP #$18    ;DERNIERE LIGNE ?
29        BNE S0
30        LDA #0      ; RETOUR 1ERE COL/1ERE LIGNE
31        STA V
32        STA H
33        JMP S1
34 S0     LDA H
35        CMP #$50
36        BNE S1
37        LDA #0      ;PASSER A 1ERE COL. LIGNE SUIVANTE
38        STA H
39        INC V
40        JMP DEB
41 S1     JSR VTAB    ;POSITIONNE CURSEUR
42        JSR GET     ;ENTREE CARACTERE
43        CMP #$9B    ;ESC ?
44        BNE S2
45        RTS
46 S2     CMP #$8D    ;RETURN ?
47        BNE S3
48        INC V       ;OUI => LIGNE SUIVANTE
49        JMP DEB
50 S3     CMP #$95    ; "-)" ?
51        BNE S4
52        INC H       ;OUI => CARACTERE SUIVANT
53        JMP S0
54 S4     CMP #$88    ; "(-" ?
55        BNE S5
56        LDA H       ;OUI => CARACTERE PRECEDENT
57        CMP #0      ; "(-" SUR 1ER CARAC. DE LA LIGNE
58        BEQ S6
59        DEC H
60        JMP S1
61 S6     LDA V
62        CMP #0      ;SI 1ER CARAC./1ERE LIGNE : ON RESTE LA
63        BEQ S1
64        LDA #$4F    ;DERNIER CARAC./LIGNE PRECEDENTE
65        STA H
66        DEC V
67        JMP S1
68 S5     CMP #$92    ; CTRL-R ?
69        BNE S51
70        DEC V       ;OUI => LIGNE PRECEDENTE
71        BPL S1      ;SI V>=0 ON POSITIONNE EN V
72        LDA #$17    ;ON SAUTE A LA DERNIERE LIGNE
73        STA V
74        JMP S1
75 S51    CMP #$89    ; CTRL-I ?
76        BNE S52
77        LDA #1
78 S53    STA M       ;SIGNALER LE "INVERSE"
79        JMP S1
80 S52    CMP #$8E    ; CTRL-N ?
81        BNE S7
82        LDA #0      ;SIGNALER LE "NORMAL"
83        JMP S53
84 S7     CMP #$96    ; CTRL-V ?
85        BNE S8
86        LDA V
87        CMP #$17    ;DERNIERE LIGNE => ON IGNORE
88        BEQ S1
89        JSR $FBC1   ;CALCUL ADRESSE BASE LIGNE V
90        LDY H
91        JSR $CF01   ;PREND CARAC. H DE LIGNE V
92        INC V       ;PASSE A LA LIGNE SUIVANTE
93        JSR 01      ;AFFICHE LE CARACTERE
94        DEC H       ; REMET LE CURSEUR DANS LA MEME COLONNE
95        JMP S0
96 S8     CMP #$84    ; CTRL-D ?
97        BEQ D0
98        JMP S9
99 D0     LDA V
100       CMP #$17
101       BNE D1
102       LDA H
103       CMP #$4F
104       BNE D1
105       JMP S1      ;DERNIERE COL/DERNIERE LIGNE : ON IGNORE
106 D1    LDA V       ;SAUVEGARDE POSITION CURSEUR
107       STA V2
108       LDA H
109       STA H2
110       CMP #$4F    ;SI DERNIER CARAC. DE LA LIGNE => PRENDRE
111 ;LE PREMIER DE LA SUIVANTE
112       BNE D6
113       LDA #0
114       STA H1
115       INC V
116       LDA V
117       STA V1
118       JMP D7
119 D6    INC H       ;DECALAGE A PARTIR DU CARAC. SUIVANT
120       LDA H
```

```
121         STA H1
122         LDA V
123         STA V1
124 D7      LDA #$17            ;ON DECALE LES CARACTERES DE
125 ; L'AVANT DERNIER DE L'ECRAN JUSQU'A H1/V1
126         STA V
127 D3      LDY #$4E
128 D5      JSR $FBC1
129         JSR $CF01
130         INY
131         JSR $CEF2
132         LDA V
133         CMP V1
134         BNE D2
135         CPY H1
136         BNE D2
137 D11     LDA V2              ;ON EST A H1/V1 => RESTITUER H/V ET
138 ; METTRE UN "BLANC" EN H/V
139         STA V
140         LDA H2
141         STA H
142         LDA #$A0
143         JSR O1
144         DEC H
145         JMP S1
146 ;
147 ; PROCEDURE EN D2 => COMMENT COPIER LE DERNIER CARACTERE
148 ; D'UNE LIGNE SUR LE PREMIER DE LA SUIVANTE ET REVENIR
149 ; A CETTE LIGNE
150 ;
151 D2      CPY #1
152         BNE D4
153         DEC V
154         LDA V
155         JSR $FBC1
156         LDY #$4F
157         JSR $CF01
158         STA C
159         INC V
160         LDA V
161         JSR $FBC1
162         LDY #0
163         LDA C
164         JSR $CEF2
165         CPY H1
166         BNE D10
167         LDA V
168         CMP V1
169         BNE D10
170         JMP D11
171 D10     DEC V
172         LDA V
173         JMP D3
174 D4      DEY                 ;DECALE CARACTERE PRECEDENT
175         DEY
176         JMP D5
177 S9      CMP #$87            ; CTRL-G ?
178 ;
179 ; CTRL-G => DECALAGE A GAUCHE
180 ; (MEMES PRINCIPES GENERAUX QUE POUR CTRL-D)
181 ;
182         BEQ G0
183         JMP S10
184 G0      LDA V
185         CMP #$17
186         BNE G1
187         LDA H
188         CMP #$4F
189         BNE G1
190         JMP S1
191 G1      LDA V
192         STA V2
193         LDA H
194         STA H2
195 G7      CMP #$4F
196         BNE G4
197         INC V
198         LDA #0
199         STA H
200         JMP G2
201 G4      INC H
202 G2      LDA V
203         LDY H
204         JSR $FBC1
205         JSR $CF01
206         STA C
207         CPY #0
208         BNE G3
209         DEC V
210         LDA V
211         JSR $FBC1
212         LDY #$4F
213         LDA C
214         JSR $CEF2
215         INC V
216         LDA #0
217         STA H
218         JMP G5
219 G3      LDA V
220         JSR $FBC1
221         DEY
222         LDA C
223         JSR $CEF2
224 G5      LDA V
225         CMP #$17
226         BEQ G6
227         LDA H
228         JMP G7
229 G6      LDA H
230         CMP #$4F
231         BEQ G8
232         JMP G4
233 G8      LDA #$A0
234         JSR O1
235         LDA V2
236         STA V
237         LDA H2
238         STA H
239         JMP S1
240 S10     JSR OUT
```

```
241         JMP S0
242 OUT     LDX M
243 ; M=1 => TRANSFORMER LES CODES-ECRAN DES CARACTERES
244 ; POUR AVOIR LES "INVERSE"
245         CPX #1
246         BNE C1
247         CMP #$C0
248         BCS 02
249         SEC
250         SBC #$80
251         JMP 01
252 02      SEC
253         SBC #$C0
254 ; PROCEDURE 01 : AFFICHE LE CARACTERE ET AVANCE CURSEUR
255 01      STA C
256         LDA V
257         JSR $FBC1
258         LDY H
259         LDA C
260         JSR $CEF2
261         INC H
262         RTS
263 GET     LDY H               ;ROUTINE POUR CARTE 80 COLONNES
264         JSR $CF01
265         PHA
266         LDA #$DF
267         JSR $CEF2
268 GET1    LDA $C000
269         BPL GET1
270         STA $C010
271         STA $67B
272         PLA
273         JSR $CEF2
274         LDA $67B
275         RTS
276         DCM "INT"
277         END
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;* ROUTINE DE LECTURE DE MEMOIRE *
 4 ;* D'ECRAN SIMULANT CELLE DE LA  *
 5 ;*   CARTE 80 COLONNES A//E      *
 6 ;*       CODE = LDA80.OBJ        *
 7 ;*                               *
 8 ;*********************************
 9         ORG $300
10         CLV
11         STY $1F
12         PHA
13         LDA $C01F
14         BPL S0
15         LDA $1F
16         LSR
17         TAY
18         PHP
19         SEI
20         LDA $C055
21         BCC S1
22         LDA $C054
23 S1      LDA ($28),Y
24         TAY
25         LDA $C054
26         PLP
27         PLA
28         TYA
29         PHA
30         BVC S2
31 S0      LDY $1F
32         PLA
33         LDA ($28),Y
34         PHA
35 S2      PLA
36         LDY $1F
37         RTS
38         DCM "INT"
39         END
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;* RECHERCHE DU CODE D'UN MASQUE *
 4 ;*        CODE = LOCCOD.OBJ      *
 5 ;*                               *
 6 ;*********************************
 7         ORG $8C14
 8         OBJ $300
 9 TAMP    EPZ $18          ;TAMPON POUR DESM95A0
10 CM      EPZ $6           ;CODE-MASQUE RECHERCHE
11 TAMPRO  EQU $9186        ;POKE PAR BASIC
12         LDA TAMPRO
13         STA TAMP
14         LDA TAMPRO+1
15         STA TAMP+1
16 C2      LDY #0
17         LDA (TAMP),Y
18         CMP CM           ;BON CODE ?
19         BNE C0
20         LDA TAMP         ;PLACE TAMP AU DEBUT DU MASQUE
21         CLC
22         ADC #3
23         STA TAMP
24         BCC C1
25         INC TAMP+1
26 C1      RTS
27 C0      INY
28         LDA TAMP         ;AJOUTE A TAMP LA LONGUEUR DU MASQUE
29         CLC
30         ADC (TAMP),Y
31         PHA
32         INY
33         LDA TAMP+1
34         ADC (TAMP),Y
35         STA TAMP+1
36         PLA
37         STA TAMP
38         JMP C2           ;PASSE AU MASQUE SUIVANT
39         DCM "INT"
40         END
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;*  COMPRESSION DES MASQUES      *
 4 ;*     CODE = SERMASK.OBJ        *
 5 ;*                               *
 6 ;*********************************
 7       ORG $300
 8 TAMP  EPZ $18       ;ADRESSE DU TAMPON DE RECOPIE
 9 LMC   EPZ $1A       ;LONGUEUR MASQUE COMPRESSE
10 CV    EPZ $25       ;POSITION VERTICALE CURSEUR
11 BASE  EPZ $28       ;ADRESSE BASE LIGNE COURANTE
12 BASCAL EQU $FBC1    ;ROUTINE CALCUL ADRESSES-BASES
13 IN    EPZ $6        ; VARIABLE DE TRAVAIL
14 SY    EPZ $7        ;STOCKAGE PROVISOIRE DE Y
15       LDA #0
16       STA CV
17       STA LMC
18       STA LMC+1
19       TAY
20 B7    JSR BASCAL
21 B2    LDA (BASE),Y  ;1ER CARACTERE EXAMINE
22       INY
23       CMP (BASE),Y  ;1ER = SUIVANT ?
24       BNE B3
25       INY
26       CMP (BASE),Y  ;1ER = 3EME ?
27       BNE B4
28       LDX #3        ;AU MOINS 3 IDENTIQUES
29 B0    INY           ;SUITE DE LA COMPARAISON
30       CMP (BASE),Y
31       BNE B1
32       INX
33       CPY #$27      ;DERNIERE COLONNE ?
34       BNE B0
35       INY
36 B1    STY SY
37       STA IN
38       LDA #$80      ;SIGNALE SUITE DE CARAC. IDENTIQUES
39       JSR S0
40       TXA
41       JSR S0        ;STOCKE NOMBRE DE CARACTERES
42       LDA IN
43 B5    JSR S0        ;STOCKE CODE DU CARACTERE
44       LDY SY
45       CPY #$25
46       BCC B2        ;RESTE PLUS DE 3 COLONNES SUR LIGNE
47       CPY #$28
48       BEQ B6        ;ON A FINI LA LIGNE
49       LDA (BASE),Y  ;STOCKE 3 DERNIERS CARAC.
50       INY
51       JMP B3
52 B4    DEY
53 B3    STY SY        ;STOCKAGE CARAC. INCHANGE
54       JMP B5
55 B6    INC CV
56       LDA CV
57       CMP #$18      ;DERNIERE LIGNE DEJA TRAITEE ?
58       BEQ S2
59       LDY #0
60       JMP B7        ;PASSE A LIGNE SUIVANTE
61 S0    LDY #0        ;STOCKAGE DANS LE TAMPON
62       STA (TAMP),Y
63       INC TAMP
64       BNE S1
65       INC TAMP+1
66 S1    INC LMC
67       BNE S2
68       INC LMC+1
69 S2    RTS
70       DCM 'INT'
71       END
```

```
 1 ;*********************************
 2 ;*                               *
 3 ;*    RESTITUTION DE MASQUES     *
 4 ;*        CODE = DESM95A0        *
 5 ;*                               *
 6 ;*********************************
 7       ORG $95A0
 8       OBJ $300
 9 TAMP  EPZ $18
10 CV    EPZ $25
11 CH    EPZ $24
12 BASE  EPZ $28
13 BASCAL EQU $FBC1
14 SY    EPZ $6
15       LDA #0
16       STA CV
17       STA CH
18       TAY
19 C1    JSR BASCAL
20 C0    JSR L0
21       CMP #$80      ;SUITE CARAC. SEMBLABLES ?
22       BEQ C2
23       LDY CH
24       STA (BASE),Y  ;AFFICHE CARAC. TEL QUEL
25       INC CH
26       INY
27 C4    CPY #$28
28       BNE C0        ;DERNIERE COLONNE NON TRAITEE
29       LDY #0
30       STY CH        ;DEBUT LIGNE SUIVANTE
31       INC CV
32       LDA CV
33       CMP #$18
34       BNE C1        ;DERNIERE LIGNE NON TRAITEE
35       RTS
36 C2    JSR L0
37       TAX           ;X = NOMBRE DE CARACTERES
38       JSR L0        ;LECTURE CODE DU CARAC.
39 C3    LDY CH
40       STA (BASE),Y  ;AFFICHE CARACTERE
41       INC CH
42       DEX
43       BNE C3        ;X=0 =) FIN DE SUITE
44       LDY CH
45       JMP C4
46 L0    LDY #0        ;RELECTURE DU TAMPON
47       LDA (TAMP),Y
48       INC TAMP
49       BNE L1
50       INC TAMP+1
51 L1    RTS
52       DCM 'INT'
53       END
```