9/8.20

Transmac, l'utilitaire pour le macro-assembleur MAC-80

POSONS LE PROBLÈME

Nous allons ici nous intéresser aux programmeurs en langage machine, qui développent des routines assembleur, et qui voudraient bien les exploiter facilement sous Basic.

Notre expérience nous a prouvé que certains programmes assembleurs sous Basic ne disposent pas suffisamment de mémoire pour générer le code machine, et il est « rageant » de se voir refuser un assemblage après avoir laborieusement entré 20 Ko de mnémoniques.

La méthode consisterait à découper le programme et à l'assembler par morceaux. Mais cela est plus facile à dire qu'à réaliser.

La solution fut trouvée à l'aide du macro-assembleur MAC-80, un cousin de MAC et RMAC que vous possédez sur votre disquette CP/M : nous avons réalisé un utilitaire de transfert de code machine sous Basic, qui permet de loger ce dernier à une adresse différente de &100, à laquelle il est implanté habituellement sous CP/M.

RAPPEL SUR MAC-80

MAC-80 est un logiciel d'assemblage, édité par la société Microsoft, qui permet d'assembler du code machine Z-80 et 8080, ainsi que la création d'une bibliothèque de macro-instructions.

Sous CP/M, ce logiciel se lance par la commande :

A> M80 <RETURN>

A l'apparition du prompt *, vous entrez le nom du fichier à assembler en majuscule et précédé du signe **=** :

* = NOMFIC.EXT

Il est possible de fixer des directives pour l'assemblage, en les notant à la suite du nom, séparées par le signe slash (**/**). Par exemple :

* = NOMFIC.EXT /C/I

Ces directives peuvent éventuellement être intégrées au fichier et sont en résumé :

O impression des adresses en octal
H impression des adresses en hexadécimal
R génération d'un fichier objet
L génération d'un fichier texte
C génération d'un fichier de références croisées
Z utilisation des mnémoniques Z-80
I utilisation des mnémoniques 8080

Le rappel de ces directives nous permet tout de suite de repérer les directives /Z et /L que nous utiliserons afin de traiter les fichiers générés avec Transmac.

Rappelons aussi que, une fois l'assemblage réalisé, il est nécessaire d'effectuer une édition des liens du programme à l'aide de **LINK-80 (L80).**

Le fichier utile pour Transmac

Suite à l'édition des liens, le programme est exploitable sous CP/M à partir de l'adresse **&100**, or les programmeurs Basic savent que, sous Basic, cette adresse est trop basse pour y placer du code machine, un programme Basic débutant en **&170**.

Aussi, notre idée fut d'utiliser un fichier généré par MAC-80, en y extrayant le code machine, pour le sauvegarder sur disquette, et ainsi le récupérer dans un programme quelconque.

Sitôt dit, sitôt fait, un rapide aperçu des fichiers générés par MAC-80 nous fait découvrir le fichier texte, suivi de l'extension **.PRN**, dans lequel on retrouve le code machine et le listing du fichier source (voir figure 1).

```
                                ;************************
                                ;
                                DEBUT:
A000'   21 A12D'                LD HL,KERNEL;4 OCTETS
                                ;RESERVES AU SYSTEME
A003'   01 A00F'                LD BC, VECTEU;VECTEUR
                                ;POUR INSTAALTION DES INSTRUCTIONS
A006'   CD BCD1                 CALL KLOGTX;CREER INSTRUCTIONS
A009'   3E C9                   LD A,0C9H;RET POUR EVITER
A00B'   32 A000'                LD (DEBUT),A;TOUT NOUVEL APPEL
A00E'   C9                      RET;FIN INSTALLATION
                                ;
                                ;
A00F'   A083'                   VECTEU: DEFW TABLE;VECTEUR
                                ;
                                ;***   TABLE DES SAUTS AUX  ***
                                ;*** NOUVELLES INSTRUCTIONS ***
                                ;
A011'   C3 A2FB'                JP SEND;ENVOI CODE
A014'   C3 A30B'                JP SENCAR;ENVOI CARACTERE
```

Fig. 1

Il faudra ainsi assembler tous les programmes assembleurs que l'on désire traiter avec Transmac à l'aide de l'option /L.

Arrêtons-nous sur l'organisation de ce fichier :

– difficilement visible sur la figure 1, il y a une colonne de deux caractères espace sur la gauche ;

– en colonnes 3, 4, 5 et 6 se trouvent les chiffres hexadécimaux composant les adresses d'implantation du code machine ;

– en colonne 7, on trouve une apostrophe, signalant la fin de l'adresse ;

– suivent ensuite 3 espaces ;

– en colonne 11 débute le code machine, qui peut se présenter sous deux formes : s'il comporte des adresses, les deux octets de celles-ci ne sont pas séparés par un espace, et elles sont suivies par une apostrophe ; s'il ne comporte pas d'adresses, chaque octet est séparé par un espace (voir adresse A009) ;

– enfin, après une longue série d'espaces, arrive le texte source.

L'ALGORITHME DE TRANSMAC

Notre idée est donc de lire chacun des octets fournis dans le fichier texte **.PRN**, et de les sauvegarder sur disquette, sachant que les adresses utiles sont composées de 2 octets.

Un problème se pose. Observons le contenu de l'adresse **A000** :

A000' 21 A12D'

Si vos connaissances en assembleur sont assez pointues, vous remarquerez en effet que l'adresse **A12D** est inscrite ici dans le désordre. Il s'agit bien de l'adresse **A12D**, mais lors de la mise en mémoire, ses deux octets doivent être inversés, pour former le code suivant : **2DA1**. Notre programme devra en tenir compte.

A partir de toutes ces remarques concernant l'organisation du fichier texte, nous pouvons nous pencher sur l'algorithme général de traitement de ce fichier :

```
– DEBUT
    – TANTQUE   la fin du fichier texte n'est pas atteinte
        – Prendre une ligne du fichier
        – TANTQUE l'on ne lit pas deux espaces consécutifs
                – Lire le caractère en position 7
                – SI c'est une apostrophe
                        – CO
                          une apostrophe signale une
                          adresse où se trouvent des
                          codes machines
                        – FINCO
                    – Lire l'octet en position 9 et 10
                    – Lire le caractère de la position suivante
                    – SI c'est un espace
                        – CO
                          on a affaire à un octet isolé
                        – FINCO
                    – Sauver l'octet
                    – SINON
                    – Prendre l'octet suivant
                    – Sauver celui-ci
                    – Sauver l'octet précédent
                        – CO
                          On vient d'inverser l'ordre des deux
                          octets
                        – FINCO
                    – FINSI
```

- FINSI
- FINTANQUE
- Passer à la ligne suivante

- FINTANTQUE

- FIN

LE LISTING DE TRANSMAC

A partir de cet algorithme, il est possible de passer facilement à l'écriture du programme.

Nous n'avons pas non plus résisté à la tentation d'écrire le programme afin qu'il ait une présentation presque professionnelle, en s'approchant de celle des utilitaires performants, existants sous CP/M.

Le programme comporte en plus une phase de récupération des octets sauvegardés, pour les réimplanter à l'adresse mémoire prévue, et ainsi sauvegarder la zone RAM utile dans un fichier .BIN.

```
10 REM  ***********************************
20 REM  *                                   *
30 REM  *      CREATION D'UN FICHIER        *
40 REM  *       BINAIRE UTILISABLE          *
50 REM  *       SOUS BASIC A PARTIR         *
60 REM  *      D'UN FICHIER ASSEMBLE        *
70 REM  *        A L'AIDE DE MAC-80         *
80 REM  *                                   *
90 REM  *    Version 1.03 - 15.02.91        *
100 REM *                                   *
110 REM ***********************************
120 REM
130 MODE 2
140 PRINT "  TRANSMAC Version 1.03"
150 PRINT
160 PRINT "  Creation d'un fichier .BIN
pour chargement sous BASIC"
170 PRINT "  d'un fichier assemble a l'a
ide de MAC-80"
180 PRINT
190 PRINT "  Votre programme assembleur
doit avoir ete assemble avec"
200 PRINT "  MAC-80 en version hexadecim
ale, et avec le mode choisi"
210 PRINT "  selon le code source (/Z ou
 /I)"
220 PRINT
230 PRINT "  De plus, vous devez creer u
n fichier listing (.PRN)"
240 PRINT "  avec l'option /L"
250 PRINT
260 PRINT "  Exemple:"
270 PRINT "              Pour le code mach
ine Z-80, lancez MAC-80 sous la forme"
280 PRINT
290 PRINT "                              M-80 =
 NOMFIC.EXT /Z/L"
300 PRINT
310 PRINT
320 PRINT "Pressez <RETURN> pour continu
er";
330 CALL &BB06
340 PRINT CHR$(13);
350 PRINT SPC(40)
360 PRINT "Entrez le nom du fichier a tr
aduire (sans extension)"
370 INPUT nomdufichier$
380 PRINT
390 PRINT "Entrez l'adresse de chargemen
t ( ORG ) sous forme hexadecimale"
```

```
400 INPUT "&",adresse$
410 adresse$ = "&" + UPPER$(adresse$)
420 adresse = VAL(adresse$)
430 MEMORY adresse - 1
440 REM
450 fichierrel$ = nomdufichier$ + ".rel"
460 fichierrel$ = UPPER$(fichierrel$)
470 fichierprn$ = nomdufichier$ + ".prn"
480 fichierprn$ = UPPER$(fichierprn$)
490 fichierhex$ = nomdufichier$ + ".hex"
500 fichierhex$ = UPPER$(fichierhex$)
510 fichierbin$ = nomdufichier$ + ".bin"
520 fichierbin$ = UPPER$(fichierbin$)
530 REM
540 MODE 2
550 PRINT "  Fichier de reference : ";
560 PRINT fichierprn$
570 PRINT
580 PRINT "  Fichier hexadecimal  : ";
590 PRINT fichierhex$
600 PRINT
610 PRINT "Traitement en cours ..."
620 PRINT
630 OPENOUT "E"
640 MEMORY HIMEM - 1
650 CLOSEOUT
660 REM
670 REM **  TRAITEMENT DU FICHIER   **
680 REM ** CREATION DU FICHIER .HEX **
690 adr = adresse
700 REM
710 REM ouverture des fichiers
720 OPENIN fichierprn$ : REM fichier a t
raiter
730 OPENOUT fichierhex$: REM fichier cib
le
740 REM
750 WHILE NOT EOF
760     REM
770     REM drapeau pour adresse
780     FLAG = 0
790     REM
800     LINE INPUT#9,A$
810     POSITION = 7
820     REM
830     REM recherche code machine
840     REM
850     B$ = MID$(A$,POSITION,1)
860     IF B$ <> CHR$(39) THEN 1560 : REM
```

```
 ce n'est pas du code
870     REM
880     REM traitement du code
890     REM
900     POSITION = 11
910     B$ = MID$(A$,POSITION,1)
920     POSITION = POSITION + 1
930     IF B$ = CHR$(39) THEN 1560 : REM
fin de code
940     IF B$ = CHR$(32) THEN 1500 : REM
aller voir si adresse
950     FLAG = 0 : REM ce n'est pas une a
dresse
960     REM
970     REM traitement de code operation
980     REM
990     C$ = MID$(A$,POSITION,1)
1000    POSITION = POSITION + 1
1010    REM
1020    REM reconstitution code
1030    REM
1040    OCTET1$ = "&" + B$ + C$
1050    OCTET1 = VAL (OCTET1$)
1060    B$ = MID$(A$,POSITION,1)
1070    IF B$ <> CHR$(32) THEN 1210: REM
verification adresse
1080    REM
1090    REM sauvegarde code
1100    REM
1110    WRITE#9,OCTET1
1120    REM
1130    REM affichage
1140    REM
1150    PRINT HEX$(OCTET1,2);
1160    PRINT "..";
1170    GOTO 910 : REM suite traitement
1180    REM
1190    REM inversion octets d'adresses
1200    REM
1210    POSITION = POSITION + 1
1220    REM
1230    REM lecture octet bas pour adress
e
1240    C$ = MID$(A$,POSITION,1)
1250    POSITION = POSITION + 1
1260    OCTET2$ = "&" + B$ + C$
1270    OCTET2 = VAL (OCTET2$)
1280    REM
1290    REM ecriture octet bas
```

```
1300    REM
1310    WRITE#9,OCTET2
1320    REM
1330    REM affichage
1340    REM
1350    PRINT HEX$(OCTET2,2);
1360    PRINT "..";
1370    REM
1380    REM ecriture octet haut
1390    REM
1400    WRITE#9,OCTET1
1410    REM
1420    REM affichage
1430    REM
1440    PRINT HEX$(OCTET1,2);
1450    PRINT "..";
1460    GOTO 910
1470    REM
1480    REM adresse en cours ?
1490    REM
1500    IF FLAG = 1 THEN 1560
1510    FLAG = 1
1520    REM
1530    REM suite traitement
1540    REM
1550    GOTO 910
1560 WEND
1570 REM
1580 REM ** FIN DE TRAITEMENT **
1590 REM ** FERMETURE FICHIER **
1600 REM
1610 CLOSEIN
1620 CLOSEOUT
1630 REM
1640 REM ** CREATION DU FICHIER **
1650 REM **     BINAIRE .BIN      **
1660 REM
1670 MODE 2
1680 REM
1690 PRINT "  Fichier ";
1700 PRINT fichierprn$;
1710 PRINT " traite"
1720 PRINT
1730 PRINT "  Fichier ";
1740 PRINT fichierhex$;
1750 PRINT " cree"
1760 PRINT
1770 PRINT "  Creation du fichier ";
1780 PRINT fichierbin$
```

```
1790 PRINT
1800 REM
1810 REM ouverture fichier
1820 REM
1830 OPENIN fichierhex$
1840 REM
1850 REM TRAITEMENT EN COURS
1860 REM
1870 WHILE NOT EOF
1880   REM
1890   REM lecture octet
1900   REM
1910   INPUT#9,octet
1920   REM
1930   REM mise en memoire
1940   REM
1950   POKE adr,octet
1960   REM
1970   REM affichage
1980   REM
1990   PRINT HEX$(octet,2);
2000   PRINT "..";
2010   REM
2020   REM octet suivant
2030   adr = adr + 1
2040 WEND
2050 CLOSEIN
2060 REM
2070 PRINT
2080 PRINT
2090 PRINT "  Donnees en memoire ..."
2100 PRINT
2110 nombeoctets = adr - adresse + 1
2120 PRINT
2130 nombreoctets = adr - adresse
2140 SAVE fichierbin$,b,adresse,nombreoc
tets
2150 MODE 2
2160 PRINT
2170 PRINT "  Fichier binaire sauvegarde
: ";
2180 PRINT fichierbin$
2190 PRINT
2200 PRINT "  Fin de traitement."
2210 PRINT
2220 PRINT "  Effectuez un RESET"
2230 PRINT
2240 PRINT "  Vous pourrez effacer les f
```

```
ichiers:"
2250 PRINT TAB(20);fichierrel$
2260 PRINT TAB(20);fichierprn$
2270 PRINT TAB(20);fichierhex$
2280 REM
2290 END
```

Lignes 10 à 330 : présentation et rappel de la directive de sauvegarde à utiliser avant d'utiliser Transmac.

Lignes 360 à 400 : acquisition nom du fichier et adresse d'implantation en mémoire.

Lignes 410 à 430 : réservation mémoire à partir de l'adresse fournie.

Lignes 450 à 520 : création des noms des différents fichiers utiles.

Lignes 550 à 620 : présentation du travail en cours.

Lignes 630 à 650 : ajustement **HIMEM** avant ouverture fichiers.

Lignes 670 à 1560 : application de l'algorithme précédemment décrit pour créer le fichier hexadécimal **.HEX**.

Lignes 1640 à 1790 : présentation de la deuxième phase du travail.

Ligne 1830 : ouverture du fichier **.HEX**.

Lignes 1850 à 2040 : création du fichier **.BIN** par mémorisation des codes hexadécimaux en mémoire.

Lignes 2090 à 2140 : sauvegarde d'une partie de **RAM** en fichier binaire.

Lignes 2150 à fin : rappel du nom du fichier binaire, et des fichiers crées pour effacement.

UTILISATION ET FONCTIONNEMENT DE TRANSMAC

Une fois sauvegardé, vous pouvez lancer Transmac pour traiter un fichier texte de type **.PRN** créé à partir de MAC-80 :

a – Dès le lancement, Transmac se présente et vous signale le type de fichier qu'il traite.

b – Après confirmation, il vous demande le nom et l'adresse d'implantation comme présenté en figure 2. Attention, le nom du fichier doit être entré sans extension. Transmac se charge d'ajouter les extensions nécessaires.

```
TRANSMAC Version 1.03

Creation d'un fichier .BIN pour chargement sous BASIC
d'un fichier assemble a l'aide de MAC-80

Votre programme assembleur doit avoir ete assemble avec
MAC-80 en version hexadecimale, et avec le mode choisi
selon le code source (/Z ou /I)

De plus, vous devez creer un fichier listing (.PRN)
avec l'option /L

Exemple:
          Pour le code machine Z-80, lancez MAC-80 sous la forme

                    M-80 = NOMFIC.EXT /Z/L


Entrez le nom du fichier a traduire (sans extension)
? minit3

Entrez l'adresse de chargement ( ORG ) sous forme hexadecimale
&a000
```

Fig. 2

Vous n'avez plus qu'à suivre les différentes phases qui sont affichées à l'écran :

c – Création du fichier hexadécimal .HEX à partir du fichier .PRN. Les différents codes hexadécimaux traités sont affichés l'un après l'autre à l'écran, les adresses étant reconstituées dans le sens octet bas puis octet haut (voir figure 3).

```
  Fichier de reference : MINIT3.PRN

  Fichier hexadecimal  : MINIT3.HEX

Traitement en cours ...

21..2D..A1..01..0F..A0..CD..D1..BC..3E..C9..32..00..A0..C9..83..A0..C3..FB..A2.
C3..0B..A3..C3..26..A3..C3..3F..A3..C3..4D..A3..C3..62..A3..C3..51..A1..C3..97.
A1..C3..9D..A1..C3..A2..A1..C3..AD..A1..C3..A7..A2..C3..D1..A2..C3..31..A1..C3.
37..A1..C3..B8..A1..C3..BF..A1..C3..C5..A1..C3..CB..A1..C3..E0..A1..C3..F5..A1.
C3..00..A2..C3..0B..A2..C3..16..A2..C3..21..A2..C3..2C..A2..C3..37..A2..C3..42.
A2..C3..4D..A2..C3..53..A2..C3..59..A2..C3..5F..A2..C3..65..A2..C3..6B..A2..C3.
71..A2..C3..77..A2..C3..7D..A2..C3..92..A2..53..45..4E..C4..57..52..49..54..C5.
53..45..4E..44..53..45..D1..50..52..49..4E..D4..52..45..43..45..50..D4..52..45.
43..41..D2..4C..4F..43..41..54..C5..43..55..52..53..4F..CE..43..55..52..53..4F.
46..C6..4C..4F..57..56..49..44..45..CF..4E..4F..52..4D..56..49..44..45..CF..49.
4E..CB..50..41..50..45..D2..43..4C..D3..41..4C..4C..43..4C..D3..47..B0..47..B1.
47..B2..55..50..50..45..D2..
```

Fig. 3

d – Après un temps plus ou moins long selon la taille du fichier texte, Transmac sauvegarde le fichier .HEX, puis le réutilise pour créer le fichier .BIN. Les octets mis en mémoire sont alors affichés comme indiqué sur la figure 4.

```
FB..79..E6..7F..C1..C9..CD..4C..A4..38..FB..CD..3F..A4..CD..4C..A4..38..F3..CD.
8B..A4..C9..E5..F5..21..36..00..2B..7D..B4..20..FB..F1..E1..C9..C5..01..00..F5.
ED..78..17..17..C1..C9..E6..7F..EA..5D..A4..F6..80..C9..C5..2E..00..26..00..06.
08..1F..ED..6A..10..FB..37..ED..6A..29..10..FD..C1..C9..C5..D5..F5..F3..01..00.
EF..1E..0A..AF..29..17..ED..79..CD..8B..A4..1D..20..F5..FB..F1..D1..C1..C9..E5.
F5..21..6D..00..2B..7D..B4..20..FB..F1..E1..C9..0A..0D..45..52..52..45..55..52.
20..44..45..20..43..4F..4F..52..44..4F..4E..4E..45..45..53..0A..0D..00..0A..0D.
45..52..52..45..55..52..20..43..4F..55..4C..45..55..52..20..44..45..20..43..41.
52..41..43..54..45..52..45..0A..0D..00..0A..0D..45..52..52..45..55..52..20..43.
4F..55..4C..45..55..52..20..44..45..20..46..4F..4E..44..0A..0D..00..
  Fichier MINIT3.PRN traite

  Fichier MINIT3.HEX cree

  Creation du fichier MINIT3.BIN

21..2D..A1..01..0F..A0..CD..D1..BC..3E..C9..32..00..A0..C9..83..A0..C3..FB..A2.
C3..0B..A3..C3..26..A3..C3..3F..A3..C3..4D..A3..C3..62..A3..C3..51..A1..C3..97.
A1..C3..9D..A1..C3..A2..A1..C3..AD..A1..C3..A7..A2..C3..D1..A2..C3..31..A1..C3.
37..A1..C3..B8..A1..C3..BF..A1..C3..C5..A1..C3..CB..A1..C3..E0..A1..C3..F5..A1.
C3..00..A2..C3..0B..A2..C3..16..A2..C3..21..A2..C3..2C..A2..C3..37..A2..C3..42.
A2..C3..4D..A2..C3..53..A2..C3..59..A2..C3..5F..A2..C3..65..A2..C3..6B..A2..C3.
71..A2..C3..77..A2..C3..7D..A2..C3..92..A2..53..45..4E..C4..57..52..49..54..C5.
53..45..4E..44..53..45..D1..50..52..49..4E..D4..52..45..43..45..50..D4..52..45.
43..41..D2..4C..4F..43..41..54..C5..43..55..52..53
```

Fig. 4

e – Les données étant toutes en mémoire, le fichier binaire est placé sur disquette. Transmac signale alors qu'il a terminé, propose un reset du CPC, et affiche le nom des fichiers devenus inutiles :

– .REL créé automatiquement par MAC-80 ;
– .PRN créé par MAC-80 et traité par Transmac ;
– .HEX créé et traité par Transmac.

```
F5..21..6D..00..2B..7D..B4..20..FB..F1..E1..C9..0A..0D..45..52..52..45..55..52..
20..44..45..20..43..4F..4F..52..44..4F..4E..4E..45..45..53..0A..0D..00..0A..0D..
45..52..52..45..55..52..20..43..4F..55..4C..45..55..52..20..44..45..20..43..41..
52..41..43..54..45..52..45..0A..0D..00..0A..0D..45..52..52..45..55..52..20..43..
4F..55..4C..45..55..52..20..44..45..20..46..4F..4E..44..0A..0D..00..
  Donnees en memoire ...



  Fichier binaire sauvegarde: MINIT3.BIN

  Fin de traitement.

  Effectuez un RESET

  Vous pourrez effacer les fichiers:
                    MINIT3.REL
                    MINIT3.PRN
                    MINIT3.HEX
```

Fig. 5