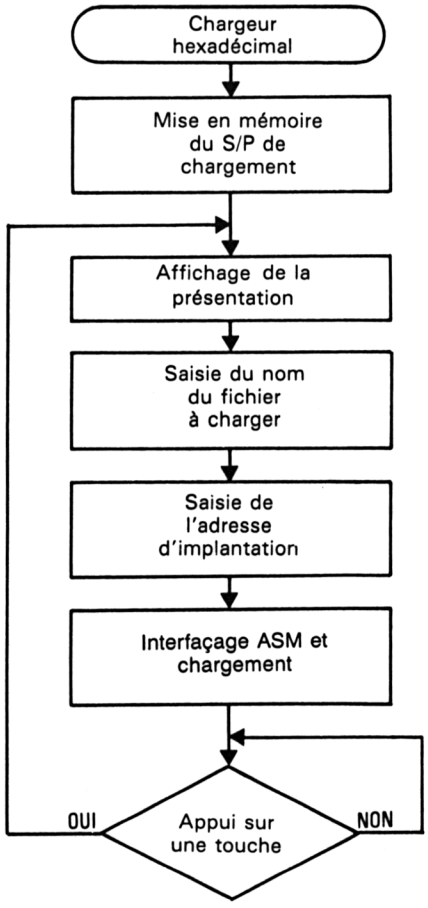
# 9/8.13

# Chargeur hexadécimal

Ce petit utilitaire écrit en Assembleur vous permet de charger un programme ou des données à n'importe quel endroit en mémoire RAM (chose impossible sous Basic: essayez par exemple de charger des données en &H100 !! Vous verrez apparaître le message **MEMORY FULL** et le chargement ne s'effectuera pas).

Le chargeur hexadécimal présenté ici est appelé sous Basic. Il vous demande le nom du fichier à charger, puis son implantation en mémoire. Le programme (ou les données) correspondant est alors chargé.

Sa logique de programmation est la suivante :



Le listing du programme est le suivant :

```
1000 REM Chargeur de programmes ou fichiers
1010 '
1020 '------------------------------------
1030 'Programme Assembleur de chargement
1040 '
1050 FOR I=&9000 TO &9014:READ A:POKE I,A:NEXT
1060 DATA &6,&0,&21,&15,&90,&11,&0,&A6,&CD,&77,&BC,&21,&0,&C
0,&CD,&83,&BC,&CD,&7A,&BC,&C9
1070 '------------------------------------
1080 INK 0,0:INK 1,10:BORDER 0:MODE 1:PEN 3:PRINT"         Ch
argeur de programmes":PEN 1
1090 LOCATE 1,10:INPUT"Nom du fichier ";N$
1100 PRINT:INPUT"Implantation memoire ";IM
1110 IF IM<0 THEN IM=IM+2^16
1120 '------------------------------------
1130 A=LEN(N$):POKE &9001,A 'Longueur du nom
1140 MSB=INT(IM/256):LSB=IM-MSB*256 'Poids Fort et Faible à
Implantation
1150 POKE &900C,LSB:POKE &900D,MSB 'à implantation du Fichie
r
1160 FOR I=0 TO A-1
1170   POKE &9015+I,ASC(MID$(N$,I+1,1)) 'Nom du prog a charg
er
1180 NEXT I
1190 '------------------------------------
1200 CALL &9000 'Chargement
1201 PRINT:PRINT"Appuyez sur une touche"
1202 A$=INKEY$:IF A$="" THEN 1202
1210 GOTO 1080 'Autre chargement
1220 END
```

— Lignes 1050 à 1060 : Chargement du programme assembleur.

— Lignes 1080 à 1110 : Présentation.

— Lignes 1130 à 1180 : Interfaçage avec l'assembleur.

— Lignes 1200 : Chargement.

— Lignes 1210 : Poursuite sur un autre chargement.

Le programme de chargement est écrit en Assembleur et fait appel à plusieurs macros du firmware :

CAS IN OPEN → Ouverture d'un fichier
CAS IN DIRECT → Lecture de données dans un fichier
CAS IN CLOSE → Fermeture d'un fichier

Rapportez-vous aux points d'entrées OBC77H, OBC83H et OBC7AH pour avoir plus de détails sur ces macros.

Son listing est le suivant :

```
 1                ;
 2                ;Chargeur HEXA
 3                ;Entree interfacee par BASIC:
 4                ;Lgr du nom du fichier a charger,
 5                ;et à Stockage Fichier.
 6                ;
 7                              ORG  9000H
 8                              LOAD 9000H
 9                OPEN:         EQU  0BC77H          ;CAS IN OPEN
10                DIRECT:       EQU  0BC83H          ;CAS IN DIRECT
11                CLOSE:        EQU  0BC7AH          ;CAS IN CLOSE
12                ;-------------------------------
13 9000 0600                    LD   B,0             ;Longueur Nom Fichier
14 9002 211590                  LD   HL,AFFICH       ;à Nom Fichier
15 9005 1100A6                  LD   DE,0A600H       ;à Buffer 2KO
16 9008 CD77BC                  CALL OPEN            ;Ouverture fichier
17                ;
18 900B 2100C0                  LD   HL,0C000H       ;à Stockage Fichier
19 900E CD83BC                  CALL DIRECT          ;Chargement Fichier
20                ;
21 9011 CD7ABC                  CALL CLOSE           ;Fermeture Fichier
22 9014 C9                      RET
23                AFFICH:       EQU  $               ;à Nom Fichier
24                              END
```

Le fichier est ouvert (ligne 16) en ayant pris soin de déclarer une zone tampon de 2 KOctets nécessaire au système pour le chargement (ligne 15).

Ensuite, le fichier est chargé (ligne 19) en ayant au préalable précisé son adresse d'implantation (ligne 18).

Enfin, le fichier est fermé (ligne 21).

*Remarque :*

Ce programme peut s'avérer très utile dans le cas où une grande partie de la mémoire est occupée (par exemple par un programme de jeu). En effet, le Basic détruit les données situées trop bas ou trop haut en mémoire. Pour pallier à ce problème, le chargeur ne doit plus être activé par un **RUN BASIC** mais par un **CALL.** Il doit être constitué d'une suite de modules identiques au programme de chargement décrit ci-dessus (un module par fichier à charger).