La protection des disquettes selon Daniel Duffau (2e partie)

Utilisation du debugger d'Euphoric pour récupérer Mluch

par André C.

Ecran-menu de la disquette CEO-soft 2

Ecran-titre du jeu de Daniel Duffau

Récupération de Mluch

Nous en étions restés à l'idée d'utiliser la fonction Dump (touche F9) d'Euphoric pour récupérer les programmes tout décodés et ainsi contourner la protection "DD" de la disquette CEO-soft 2 !

Je commence par Mluch, non parce qu'il est le premier de la liste, mais parce que c'est un programme monobloc en Basic et que je le connais bien.

Après quelques tâtonnements, je me rends compte que le meilleur moment pour presser F9 est d'attendre que Mluch se soit lancé, afin que tout le programme soit transcodé et bien en place.

L'analyse du fichier de Dump avec un éditeur hexadécimal montre facilement que le fichier Basic commence en #0501 et se termine en #3E4C.

Là, ma connaissance de MLUCH m'a joué des tours (ou m'a peut-être évité des errances, qui sait ?).

En effet, je savais comment fini le programme et cela m'a évité de chercher les triple #00 qui marquent la fin de tout programme Basic. Nous verrons les conséquences de ma paresse.

Dans l'éditeur hexadécimal, je copie donc la zone #0501 à #3E4C du fichier Dump vers un fichier vierge et y ajoute un entête K7 approprié avec les bonnes adresses de début et de fin. Je sauve sous le nom "MLUCH.TAP".

NB. Pour éviter de se casser la tête, le plus simple pour "fabriquer" cet entête est d'aller sur un Atmos et de sauver la zone de mémoire correspondante, avec les bons paramètres, ici CSAVE"MLUCH",A#0501,E#3E4C
puis on ouvre le TAP obtenu, dans l'éditeur hexadécimal et on récupère l'entête (tous les octets du début, jusqu'au #00 qui suit le nom du fichier TAP).

Reprenant ma disquette de travail, je récupère le fichier "MLUCH.TAP" issu du collage de l'entête ad hoc devant les octets récupérés dans le fichier Dump d'Euphoric, avec un CLOAD"MLUCH.TAP" suivi d'un SAVE"MLUCH".

La fièvre du succès me gagne et je lance MLUCH. Patatras ! Il manque la plus grande partie du programme !

Le listing ne va pas plus loin que la ligne 9510 !

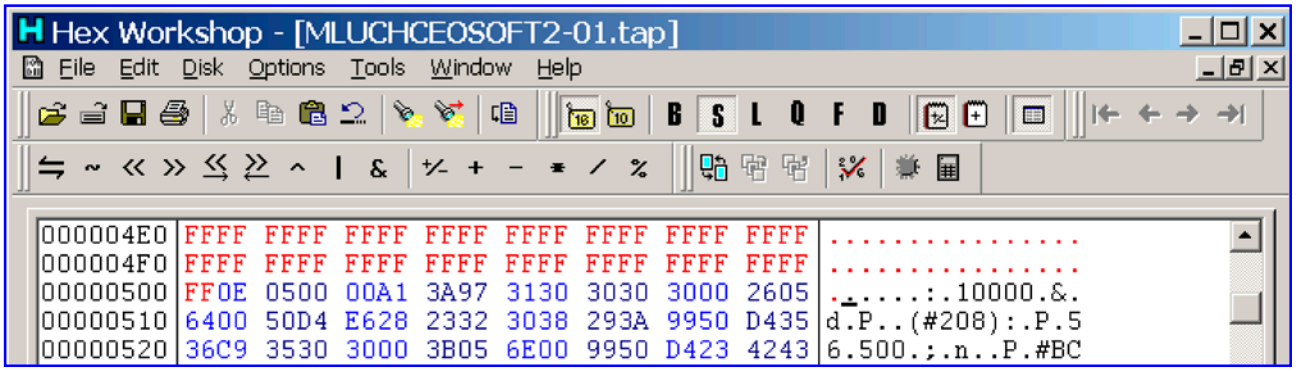
Je vérifie l'adresse de fin du Basic avec un PRINTHEX$(#9C) et j'obtiens bien #3E4C.

Il y a probablement une rupture des liens de lignes.

Je tente de restaurer ces liens de lignes avec un CALL#C55F en Rom, mais ça ne marche pas. Il faut donc restaurer manuellement.

Avec mon éditeur hexadécimal, je repars du fichier TAP (plus facile à éditer que ma disquette de travail). Je coupe l'entête K7 (et le sauve pour le remettre en place plus tard).

Devant le programme j'ajoute 501 octets #FF (dont les adresses iront donc de #0000 à #500) afin que l'offset du programme corresponde à sa place en mémoire. Il va bien maintenant de #0501 à #3E4C.



Le début du programme Basic commence bien à l'offset #0501 dans l'éditeur hexadécimal.

```
…
9040 GOTO15085
9500 FORI=25TO26:PLOT5,I,CHR$(1)+CHR$(10)+"BONUS 1000 POINTS + UNE VIE":NEXT
9510 ZAP:ZAP:ZAP:WAIT300:RETURN
```

La fin du listing tronqué que j'avais obtenu

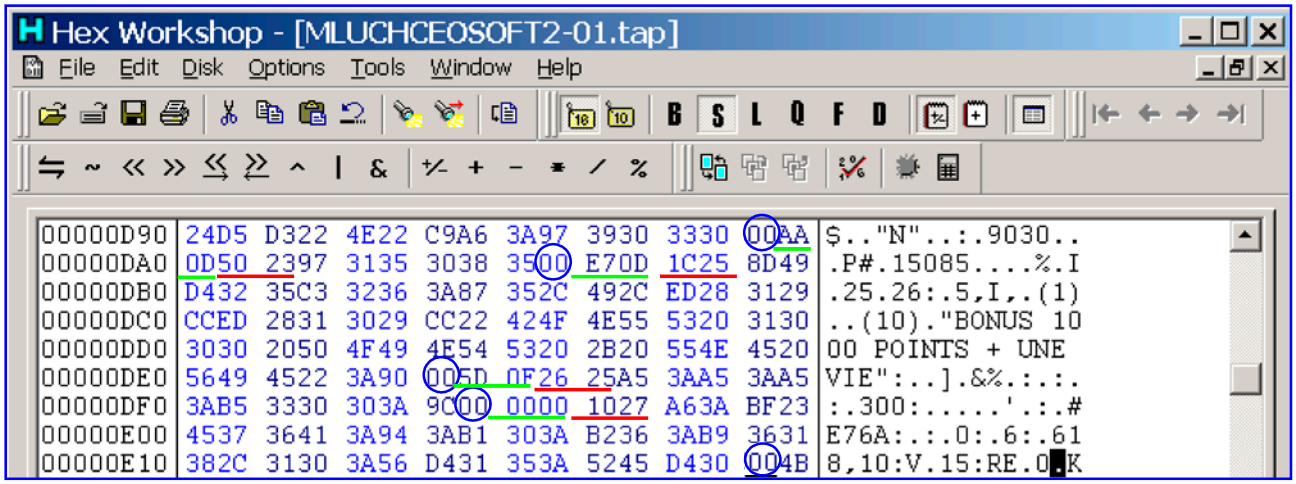
Il doit donc exister une anomalie à la fin de cette ligne 9510, par exemple un triple #00.

Dans la figure ci-dessous, les #00 cerclés en bleu sont les marques de fin des lignes Basic. Il est suivit d'un nombre sur deux octets souligné en vert qui est le lien (adresse du lien de la ligne suivante). Puis vient un autre nombre sur deux octets souligné en rouge qui est le n° de ligne. Enfin suit le code Basic, puis le #00 de fin de ligne etc.

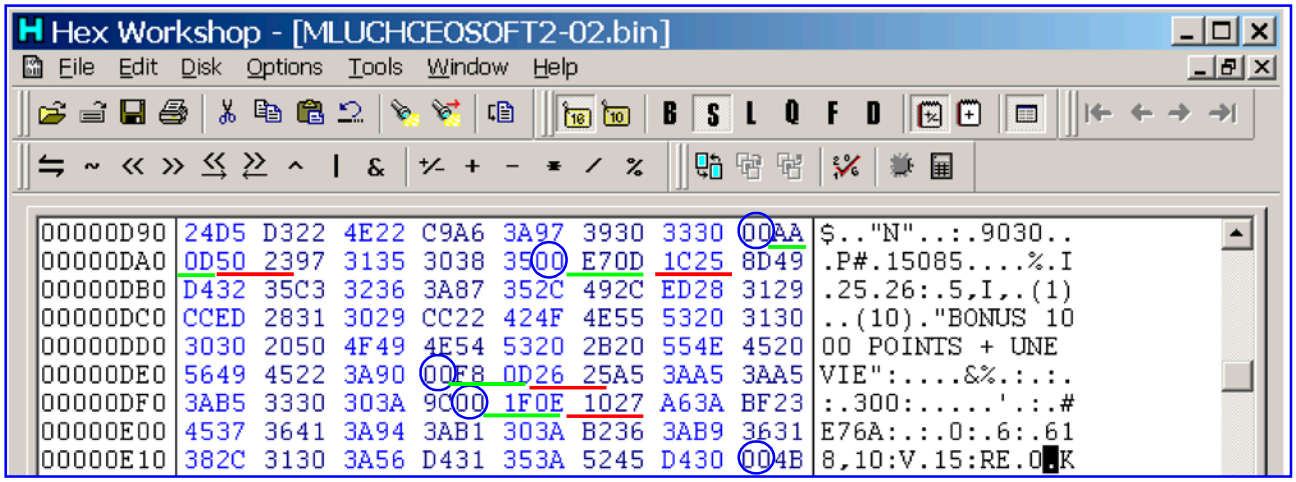
La dernière ligne affichée par le LIST étant la ligne 9510, je recherche la suite d'octets #26 #25.

Rappel : En mémoire, les nombres sur deux octets sont écrits dans l'ordre inverse. L'octet de poids faible vient en premier, puis l'octet de poids fort. Par exemple pour rechercher le n° de ligne 9510, qui s'écrit #2526 en hexadécimal, il faut cherche la suite d'octets #26 #25.

Le lien qui précède ce n° de ligne est l'adresse de la ligne suivante (celle qui ne s'affiche pas). On lit #0F #5D (soit l'adresse #5D0F). Mais ce lien est faux et ne correspond à rien. Je le remplace par #F8 #0D (soit l'adresse #0DF8), qui l'adresse de la ligne suivante de n°10000 (soit #2710 en hexadécimal, on voit bien les valeurs #10 #27).



Etat du programme Basic avant la correction de plusieurs liens de lignes



Etat du programme Basic après correction de plusieurs liens de lignes

Pendant que j'y suis, je vérifie le lien suivant, situé juste avant le n°10000 et bingo ! Au lieu d'avoir l'adresse de la ligne suivant la 10000, je trouve #00 #00 (lien nul), qui avec le #00 de fin de la ligne 9510 forment une suite de 3 octets #00. Or un triple #00 marque la fin du programme Basic. C'est pourquoi le listing s'est arrêté là.

Je remplace ces #00 #00 par l'adresse de la ligne située après la ligne 10000, soit #1F #0E (pour l'adresse #0E1F). Je vérifie les liens suivants : ils sont bons. Je mets ces erreurs de liens sur le compte d'un "accident" dont l'origine m'échappe. Et j'ai tort comme la récupération de "L'oeil de Zoltec" qui suivra va le montrer. Il s'agit en fait d'une ultime protection !

Pour en terminer, je remplace les #501 octets #FF placés au début du programme Basic par l'entête K7 que j'avais mis de côté. Je sauve ce TAP corrigé, reprends ma disquette de travail et récupère ce fichier avec un CLOAD suivi d'un SAVE"MLUCH". Je teste : Hioupi ! Ça marche !

Bilan des différentes versions de Mluch en ma possession :

1) MLUCH utilisé pour l'application PB5, qui est une version pour Oric-1 et Atmos ne comportant que 10 phases (10 tableaux). Cette version nécessite un QUIT car elle utilise la page 4, réservée à Sedoric. C'est cette même version que j'avais ensuite adaptée au Super-Oric. Elle m'avait été fournie par un Oricien il y a fort longtemps (vers 1995 ?) Elle est commentée et comporte des instructions pour créer de nouvelles phases. Tout ceci me laisse penser qu'il s'agit de la toute première version mise au point par Daniel Duffau, voire d'une version en cours de développement. Elle est marquée "COPYRIGHT 1985 D. DUFFAU", sa checksum est de #73BD, c'est un Basic "AUTO", qui va de #0501 à #3533.

2) MLUCH de la disquette trimestrielle de juin 1998, qui est une version pour Atmos comportant 19 phases et ne nécessite pas de QUIT. Le code langage machine à été déplacé de #400 en #B400. Cette version est également marquée "COPYRIGHT 1985 D. DUFFAU".

3) MLUCH de CEO-soft 2 (disquette publiée en 1989), semblable à la version précédente, mais vraisemblablement postérieure (bien que toujours marquée "COPYRIGHT 1985 D. DUFFAU"). J'ai tenté de décoder le petit programme langage machine supplémentaire (accompagné de 2 lignes de DATA en plus, les lignes n°19140 et 19150). Mais il est très obscur (utilise par exemple des mnémoniques non documentés). Après coup, je suppose que son rôle est de "boucher" le triple #00 qui bloque le programme. Sinon, ça ne peut pas marcher. Il y a aussi quelques autres petites différences :

- Ligne 10 rebaptisée ligne 0 (dispositif élémentaire de protection) et augmentée de la commande TEXT.
- Ligne 10015 modifiée pour implanter le code supplémentaire.
- Ligne 10270 prolongée par un CALL#B44D appelant le code supplémentaire.
- Ligne 10680 déplacement du CALL#E93D (reconfiguration du VIA).

Et c'est tout. Je pense que cette dernière version a été spécialement adaptée pour la disquette CEO-soft 2.

Nous verrons ce que ça donne pour les 3 autres jeux dans les prochains articles.

à suivre…