SEDORIC? DO IT YOURSELF! (12)

LINPUT re-suite et fin finale : Dans le CEO-MAG de septembre 91 (cf. n°17), notre ami Yann Legrand, exaspéré par la bogue de la commande LINPUT, a cherché de l'aide "pour créer une rubrique dans laquelle SEDORIC serait disséqué une bonne fois pour toutes, afin que l'on puisse exploiter toutes ses routines les plus secrètes". Je me suis immédiatement porté à son secours, en effet j'avais déjà pas mal travaillé à "dépiauter" Sédoric.

Petit historique de la longue quête du LINPUT :

Nous avons attaqué le problème à bras le corps, mais ce n'est qu'en mars 92 (CEO-MAG n°23) que la nouvelle rubrique "Sédoric à nu" a vu le jour. Par la suite, nous avons publié une série sur la commande LINPUT ("Sédoric à nu" n° 10, 11 & 12 dans les CEO-MAG n°32 de décembre 92, n° 34 de février 93 et n°36 d'avril 93).

Mise en évidence de la bogue de LINPUT :

Tapez donc la commande suivante : LINPUT@10,10,"X",43;F$↵ qui demande à LINPUT d'afficher à partir de la position x = 10 de la ligne y = 10 une fenêtre de saisie de 43 cases matérialisées par des "X" et de saisir une chaîne de 43 caractères dans F$. Vous n'obtiendrez jamais le résultat escompté, mais quelque chose d'aléatoire en fonction des commandes précédentes ayant affecté la position du curseur (voir recopie d'écran à la fin de cet article).

Une première "médication" :

Malgré nos recherches, nous n'avons pas su trouver l'origine de la bogue. Toutefois, nous avons proposé une parade : ne pas utiliser les paramètres "@x,y," avec la commande LINPUT, mais faire précéder celle-ci d'un "PRINT@x,y;CHR$(18);". Le CHR$(18) correspond au CTRL/R qui n'est pas utilisé par l'oric, mais n'est pas rejeté. En fait la commande PRINT est trompée : elle positionne le curseur à x,y et envoie ce CTRL/R qui ne fait rien. La position est néanmoins validée (ce que ne fait pas PRINT@x,y;"";) et le curseur reste en place grâce au ";". Le LINPUT qui suit affiche le début de sa fenêtre à ce point x,y. Le tour est joué.

Une parade définitive :

Le tour est joué,... mais nous n'avions toujours pas compris pourquoi, lorsque le nombre de caractères à saisir est supérieur à 38, la fenêtre est mal positionnée, de la mauvaise taille, ni pourquoi le curseur est généralement hors de la fenêtre de saisie ! Je me suis donc remis au travail. Après plusieurs jours d'efforts caniculaires, j'ai finalement trouvé que cette bogue spectaculaire est due à 2 petits octets qui manquent dans le code de LINPUT ! Pour que tout fonctionne normalement il aurait fallu qu'un STX 30 soit ajouté en ECB5 (RAM OVERLAY) entre le STX 0269 et le JSR D73E.

Mise en oeuvre :

Nous allons donc corriger la commande LINPUT de notre disquette "DO IT 1". Le problème est qu'il n'y a pas de place dans le code de LINPUT pour ajouter 2 octets. Il va donc falloir remplacer le JSR D73E par un JSR XXXX et à l'adresse XXXX insérer le code STX 30 suivi de JMP D73E, soit 5 octets. Simple, mais les places sont chères dans la RAM OVERLAY. Heureusement, nos travaux précédents en ont dégagé un peu : il nous reste 65 octets libres à l'emplacement de l'ex-"EXT" (c'est joli ça !), c'est à dire de E9FA à EA3A. Cette zone nous sert à greffer les nouvelles commandes insérées dans la banque 7 (exemple VH). Nous allons squatter les 5 dernières positions de cette zone libre, soit de EA36 à EA3A. Nous en profiterons pour remplacer les restes de l'ex-"EXT" par des NOP (No Operation) de E9FA à EA35, ce qui fait plus

propre et évite par la suite de se demander ce que c'est que ce code (comme c'est arrivé plusieurs fois dans les banques interchangeables).

Booter donc votre Oric avec votre disquette "DO IT 1" dans le drive A et votre disquette "DO IT 2" dans le drive B (si vous avez deux drives, sinon il faudra adapter mes directives). Tapez les commandes suivantes : HIMEM#13FF↵ puis NOYAU↵ et enfin B-MONAC1↵ (ou tout autre moniteur que celui d'André Chénière, les commandes ont généralement une syntaxe proche).

Dumper le code en 3CB5 (correspondant à ECB5) avec D3CB5↵ vous devez obtenir la suite d'octets suivante : 20 3E D7 20 2C D2 etc. Assemblez en 3CB5 avec A3CB5↵ entrez JSR EA36↵ vérifiez avec un nouveau dump qui doit vous donner : 20 36 EA 20 2C D2 etc.

Dumper le code en 39F7 (correspondant à E9F7) avec D39F7↵ vous devez obtenir la suite d'octets suivante : 4C 5E F1 60 C9 BA etc. Les 3 premiers octets correspondent au JMP F15E que nous avions laissé en attente. Les restes de l'ex-"EXT" commencent donc en 39FA avec 60 C9 BA etc. et se terminent en EA3A par les octets 4C AC D5. A partir de 39FA, entrez 60 fois la valeur #EA (code de NOP), c'est à dire jusqu'en 3A35 inclus. Assemblez en 3A36 avec A3A36↵ entrez STX 30↵ et JMP D73E↵. Vérifiez avec un nouveau dump en 39F7 qui doit vous donner : 4C 5E F1 suivi de 60 NOPs suivi de 86 30 4C 3E D7 (notre greffon) suivi de 20 38 D2 etc. (début de la commande SWAP). Sortez de votre moniteur, sauvez vos modifications : SAVEU"B-NOYAU",A#1400,E#4FFF↵ et faites ensuite un B-MISEAJOUR↵ (ou équivalant, voir les rubriques précédentes).

Test du débogage effectué :

Re-bootez avec la disquette "DO IT 1" et essayez à nouveau : LINPUT@10,10,"X",43;F$↵ La fenêtre s'affiche bien à partir des coordonnées x = 10 et y = 10. Elle est bien matérialisée par 43 "X". Le curseur est en place au début de la fenêtre de saisie (voir recopie d'écran ci-dessous). Tapez un petit texte, commençant par exemple par TOTO et se terminant par FIN. Sortez avec ↵ et vérifiez avec un PRINT F$↵. La chaîne complète (43 caractères commençant par TOTO et se terminant par FIN, s'affiche. Vous pouvez maintenant utiliser LINPUT sans crainte pour saisir jusqu'à 255 caractères.

Notez au passage que la commande PRINT@ de la ROM présente aussi un défaut de positionnement du curseur lorsque la chaîne à afficher comporte plus de 38 caractères. Mais déboguer la ROM n'est pas mon propos. Si nécessaire utilisez PRINT@x,y;CHR$(18);:PRINT F$↵

André Chéramy 54 rue de Sours 28000 Chartres