Cherche première case de champ présente dans le masque en C400/C7E7

```
F309- A9 1E     LDA #1E    caract de contrôle pour placer curseur en (0,1)
c'est à dire au début de l'écran (1ère case de 1ère ligne) (fonction HOME)
F30B- 20 2A D6  JSR D62A   XAFCAR "Affiche" ce caractère ASCII (CTRL/^)
F30E- 20 06 D2  JSR D206   JSR CBF0/ROM (CRLF) curseur au début du masque
F311- 20 CA F2  JSR F2CA   lit dans le masque l'octet corresp au curseur
F314- F0 0E     BEQ F324   simple RTS si c'est #7F (indicateur de champ)
F316- A9 09     LDA #09    sinon, A = curseur vers la droite (CTRL/I)
F318- 20 EC F2  JSR F2EC   Gestion déplacements du curseur dans le masque
F31B- 10 F4     BPL F311   reboucle si curseur est toujours dans le masque
F31D- 68        PLA si fin de masque atteinte sans trouver de #7F, F31E- 68
retour et DONC sort de WINDOW.
F31F- 28        PLP        récupère les indicateurs 6502
F320- 68        PLA        récupère A
F321- 8D 6A 02  STA 026A   et le remet en 026A (flags d'état console)
F324- 60        RTS
```

Copie de WI$ dans SCRN et de SCRN dans WI$

Il s'agit de 2 routines complexes qui s'entremêlent constamment, ce qui gagne quelques octets, mais perd beaucoup en clarté! En résumé, on a:
(1) l'écran avec le curseur et les pointeurs 0269, 0268 et 12/13
(2) la zone intermédiaire BUF1 (C100 à C1FF) pour stocker les chaînes.
(3) la zone de stockage finale sous HIMEM (selon descripteurs) et
(4) le tableau WI$ contenant la liste de descripteurs de chaîne.

A chaque champ de données (dans le masque) correspond une chaîne (stockée sous HIMEM et repérée par un descripteur dans WI$) qui sera copie de ou à l'écran (à la place correspondante). L'exploration du masque par un "curseur" fictif (qui suit les déplacements du vrai curseur dans l'écran) permet de savoir où sont les cases de champs (#7F). Afin de comprendre ce qui suit, nous vous conseillons de suivre d'abord le fil du sous-programme WI$ -> SCRN qui est plus facile.

Organigramme de la routine WI$ -> SCRN

a) Cherche le premier champ dans le masque et positionne le curseur.
b) Initialise la recherche de la première chaîne dans WI$.
c) Cherche un descripteur de chaîne dans WI$, met son adr dans B6/B7.
d) Ecrit la longueur de la chaîne dans F2 et son adr dans 91/92.
e) Copie dans l'écran la chaîne lue dans la zone sous HIMEM (cette opération est pilotée par le déplacement simultané du "curseur" dans le masque, afin de détecter la présence de "#7F" matérialisant le champ).
f) Tant que la fin du masque n'est pas atteinte, reboucle en (c).

Organigramme de la routine SCRN -> WI$

a) Cherche le premier champ dans le masque et positionne le curseur.
b) Initialise la recherche de la première chaîne dans WI$.
c) Cherche un descripteur de chaîne dans WI$, met son adr dans B6/B7.
d) Ecrit la longueur dans F2 et l'adr dans 91/92 (bogue, c'est inutile).
e) Copie dans BUF1 la chaîne présente à l'écran (opération pilotée par le déplacement du "curseur" dans le masque pour détecter les "#7F" de champ).
f) Réserve sous HIMEM une chaîne de longueur D0 et d'adr D1/D2.
g) Copie sous HIMEM (selon D0/D1/D2) la chaîne en attente dans BUF1.
h) Met à jour le descripteur (pointé par B6/B7) dans WI$ selon D0/D1/D2.
i) Tant que la fin du masque n'est pas atteinte, reboucle en (c).

Rappel: le tableau WI$ ne contient pas les chaînes, mais leurs descripteurs. Les chaînes proprement dites sont stockées dans la zone sous HIMEM. Par simplification de langage, on "lit" ou on "écrit" une chaîne dans WI$.

Recopie les champs de l'écran dans le tableau WI$ (SCRN -> WI$)

```
F325- 18        CLC C = 0, entrée appelée avant de sortir de WINDOW
F326- 24 38     BIT 38     et continue en F328
```

Recopie les chaînes de WI$ dans les champs de l'écran (WI$ -> SCRN)

```
F327- 38        SEC C = 1, entrée appelée au début de WINDOW
```

F328- 6E 72 C0 ROR C072   C -> b7 de C072 qui permet désormais de savoir si on est entré en F325 ou en F327, c'est à dire, s'il faut copier de **champ à l'écran -> tableau (SCRN -> WI$)** ou de **tableau -> champ (WI$ -> SCRN).**
F32B- 20 09 F3 JSR F309   Cherche 1er champ dans le masque en C400/C7E7
F32E- A9 57   LDA #57   place "WI$" en B4/B5 (caractères significatifs
F330- A0 C9   LDY #C9   d'une variable) (#57="W" et #C9="I"+128 pour $)
F332- 85 B4   STA B4   bogue: pas de vérification concernant
F334- 84 B5   STY B5   l'existence de WI$, ni sa validité,
F336- A9 00   LDA #00   ni s'il contient quelque chose à copier.
F338- 85 F6   STA F6   #00 -> F6/F7 (n° de 1ère chaîne à chercher)
F33A- 85 F7   STA F7

Cherche une chaîne dans le tableau WI$

F33C- A0 01   LDY #01
F33E- 84 26   STY 26   #01 -> 26 (nombre de dimensions du tableau)
F340- 88   DEY #00 -> 29 (b7 = 0 flag "non entier")
F341- 84 29   STY 29   #00 -> 27 (flag consultation tableau, inhibe
F343- 84 27   STY 27   "REDIM'D ARRAY ERROR", si pas nul déclenche
F345- 88   DEY un nouveau DIM cf "Oric à Nu" pages 154/155)
F346- 84 28   STY 28   #FF -> 28 (b7 = 1 flag "chaîne")
F348- A4 F6   LDY F6   {
F34A- A6 F7   LDX F7   { YX reçoit F6/F7 (n°de la chaîne à chercher)
F34C- E6 F6   INC F6   { puis F6/F7 est incrémenté (chaîne suivante)
F34E- D0 02   BNE F352   {
F350- E6 F7   INC F7   {
F352- 20 D1 04 JSR 04D1   D306/ROM Cherche la chaîne dans le tableau WI$ retourne avec l'adresse du descripteur de chaîne dans B6/B7
F355- A0 00   LDY #00   prépare index Y pour lire ce descripteur
F357- B1 B6   LDA (B6),Y lit 1er octet et le place en F2
F359- 85 F2   STA F2   (longueur de la chaîne)
F35B- C8   INY
F35C- B1 B6   LDA (B6),Y lit 2ème octet et le place en 91
F35E- 85 91   STA 91   (LL de l'adresse de la chaîne)
F360- C8   INY
F361- B1 B6   LDA (B6),Y lit 3ème octet et le place en 92
F363- 85 92   STA 92   (HH de l'adresse de la chaîne)
F365- A2 00   LDX #00   X pointe dans la chaîne (X=0 pour 1er caract)
F367- 2C 72 C0 BIT C072   teste le flag b7 de C072 (0 si juste avant la sortie de WINDOW, cas où il faut copier de l'écran vers le tableau WI$)
F36A- 10 14   BPL **F380**   si c'est le cas, continue en F380 (SCRN -> WI$)

Recopie la chaîne de WI$ dans le champ à l'écran (WI$ -> SCRN)

F36C- E4 F2   CPX F2   Pointeur X comparé à longueur de chaîne F2, ce qui positionne C à 0 si X < F2 ou C à 1 si X >= F2, c'est à dire si le pointeur a atteint la fin de la chaîne (sera testé plus loin en E375).
F36E- 8A   TXA sauve le pointeur X dans A (= pointeur actuel)
F36F- E8   INX X vise le prochain caractère (Z = 1 si X = #00)
F370- F0 59   BEQ F3CB   si X=#00, erreur n°19 "STRING TOO LONG" Lorsque X = #100, le pointeur actuel A = #FF ce qui indique que la chaîne est trop longue. En effet sous Sédoric les chaînes ne peuvent avoir plus de 255 caractères or 1er caract est visé par X = #00 et dernier par X = #FE (254)
F372- A8   TAY récupère le pointeur actuel dans Y pour index
F373- B1 91   LDA (91),Y lit octet selon adr de chaîne en 91/92 + index
F375- 90 1C   BCC **F393**   si la fin de la chaîne n'était pas atteinte, continue en F393 pour affichage de cet octet à l'écran

Suite WI$ -> SCRN: cas où la chaîne est plus courte que le champ (C=1)

F377- A9 7F   LDA #7F   si la fin de la chaîne dans WI$ était atteinte, remplace cet octet par #7F (carré de couleur INK), pour compléter le champ
F379- AC 69 02 LDY 0269   Y = n° de colonne où est le curseur TEXT
F37C- 91 12   STA (12),Y copie #7F dans l'écran sous le curseur TEXT
NB: 12/13 contient l'adresse du début de la ligne dans l'écran et Y la position du curseur TEXT sur cette ligne.
F37E- B0 11   BCS **F391**   suite forcée en F391 car C = 1 (chaîne < champ)

SCRN -> WI$: copie chaîne de l'écran dans BUF1 (avant d'aller dans WI$)

```
F380- AC 69 02   LDY 0269    Y = n° de colonne où est le curseur TEXT
F383- B1 12      LDA (12),Y  lit le caractère sous le curseur TEXT
F385- C9 7F      CMP #7F     est-ce #7F? (carré couleur INK utilisé par DEL)
F387- D0 02      BNE F38B    si non, saute l'instruction suivante
F389- A9 20      LDA #20     remplace #7F par un espace
F38B- 9D 00 C1   STA C100,X  écrit le contenu de A dans l'octet n° X de BUF1
```

où la chaîne est assemblée avant "copie" dans WI$.

```
F38E- E8         INX vise le prochain caractère à copier
F38F- F0 3A      BEQ F3CB    erreur n°19 "STRING TOO LONG"
```

si X = #00, on a écrit un caractère de trop , celui visé par X = #FF

Suite commune pour SCRN -> WI$ ou chaîne < champ

Le STA (12),Y en F37C et le STA C100,X en F38B n'ont ni fait avancer le curseur, ni mis à jour les pointeurs 0269, 0268 et 12/13. Ceci sera fait ci-après, en "affichant" une flèche droite.

```
F391- A9 09      LDA #09     A = curseur à droite (CTRL/I) (case suivante)
```

Suite commune pour tous (SCRN -> WI$ et WI$ -> SCRN)

```
F393- 20 2A D6   JSR D62A    XAFCAR Affiche caractère ASCII contenu dans A
F396- 20 CA F2   JSR F2CA    lit dans le masque l'octet correspondant au
```

curseur, au retour Z=1 si une case de champ a été trouvée.

```
F399- F0 CC      BEQ F367    reboucle si #7F (lire caract suiv de la chaîne)
F39B- 2C 72 C0   BIT C072    teste le flag b7 de C072 (0 si juste avant la
```

sortie de WINDOW, cas où il faut copier de l'écran vers le tableau WI$)

```
F39E- 30 1C      BMI F3BC    si ce n'est pas le cas, continue en F3BC
```

Suite pour SCRN -> WI$: recopie de BUF1 dans WI$

```
F3A0- 86 F2      STX F2      sauve X dans F2
F3A2- 8A         TXA et dans A (rappel: X est le pointeur de chaîne)
F3A3- 20 64 D2   JSR D264    Réserv chaîne de lg A, sauv lg D0 et adr D1/D2
F3A6- A0 00      LDY #00     prépare index pour copie de BUF1 -> chaîne
F3A8- B9 00 C1   LDA C100,Y  lit octet n°Y dans BUF1
F3AB- 91 D1      STA (D1),Y  écrit cet octet dans chaîne réservée
F3AD- C8         INY octet suivant
F3AE- C4 F2      CPY F2      compare Y et valeur dans F2 (longueur chaîne)
F3B0- D0 F6      BNE F3A8    reboucle tant que pas fini
F3B2- A0 02      LDY #02     prépare Y pour copier 3 octets lg & adr chaîne
F3B4- B9 D0 00   LDA 00D0,Y  lit octet du descripteur de chaîne (lg & adr)
F3B7- 91 B6      STA (B6),Y  et le copie en B6/B7/B8
F3B9- 88         DEY pour l'octet précédent
F3BA- 10 F8      BPL F3B4    reboucle tant que Y n'est pas négatif
```

Suite pour tous (SCRN -> WI$ et WI$ -> SCRN): recherche le champ suivant

```
F3BC- A9 09      LDA #09     A = curseur vers la droite (CTRL/I)
F3BE- 20 EC F2   JSR F2EC    Gestion déplacements du curseur dans le masque
F3C1- 30 0B      BMI F3CE    simple RTS si N = 1 (fin de masque atteinte)
F3C3- 20 CA F2   JSR F2CA    lit dans le masque l'octet corresp au curseur
F3C6- D0 F4      BNE F3BC    reboucle en F3BC tant que Z = 0 (cherche champ)
F3C8- 4C 3C F3   JMP F33C    champ trouvé, reboucle en F33C pour chercher
```

l'adresse de la chaîne suivante dans le tableau WI$.

```
F3CB- 4C 77 E9   JMP E977    relais vers erreur n°19 "STRING TOO LONG"
```

Nous travaillons actuellement à dépouiller la routine LINPUT qui n'est pas mal non plus dans le même genre. Si vous avez des questions concernant le travail déjà publié ou des souhaits concernant les prochaines routines que nous pourrions examiner, n'hésitez pas à nous écrire.

André Chéramy, 54, rue de Sours, 28000 CHARTRES et Yann Legrand, 608, rue de l'Eglise, "Les Templiers", 62610 LANDRETHUN LES ADRES

à suivre...