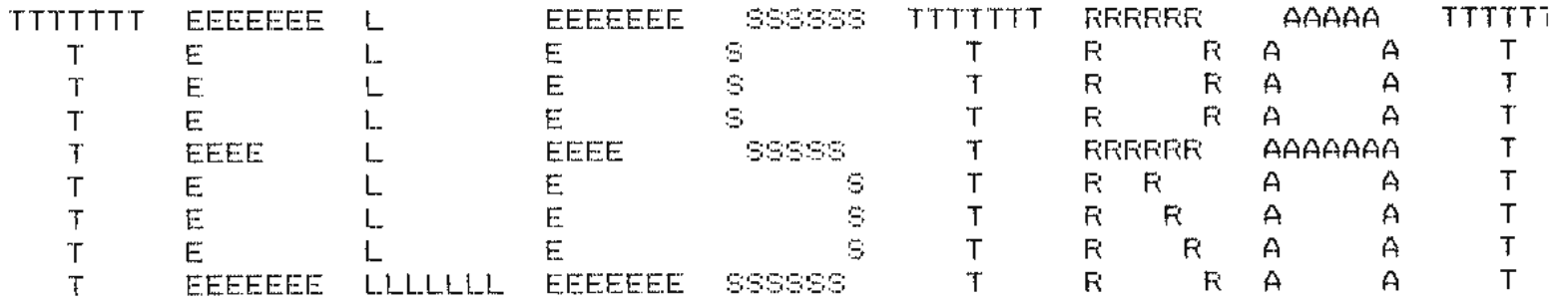
Guillaume MEISTER

Présente



SYSTEME M'ETAIT CONTE

Date de parution : Le 20 Mai 1990

Distribution : Guillaume MEISTER

TABLE DES MATIERES

INTRODUCTION

Il y a déja deux ans, mon premier livre "TELESTRAT à COEUR OUVERT" sortait, distribué par l'AEDIT, à grand renforts de pub sur TELSTAR et par le Club Disc'ORIC.

Aujourd'hui l'AEDIT n'est plus, TELSTAR reste l'unique serveur visité régulièrement par tous les Oriciens actifs et le CDO est devenu le CEO, Club EUROP'ORIC, association 1901.
Cette évolution montre clairement que les oriciens ne baissent pas les bras, loin s'en faut. Et je suis un oricien avant tout !

Alors deux ans après "TELESTRAT à COEUR OUVERT", voici "TELESTRAT, système m'était conté", titre léger mais éloquent quant à la matière de cet ouvrage.
Il est bien difficile d'écrire un livre intéressant sur une machine; ma première idée a été de réecrire le manuel d'utilisation de l'HYPER-BASIC, mais je ne pense pas que l'avenir soit dans le basic.
En outre, il est très important, afin de profiter pleinement de sa machine, de bien la connaitre. "TELESTRAT à COEUR OUVERT" présentait le matériel et la programmation matérielle, "TELESTRAT, système m'était conté" présente le logiciel du TELESTRAT : le TELEMON V2.4.

Ce livre représente, après STRATDISK, le plus gros travail que j'ai effectué sur et pour le TELESTRAT.
Il m'a d'abord fallut écrire un traitement de textes plus souple que QVC, notemment au niveau du centrage et des fonctions de bloc, toujours en 80 à 132 colonnes. D'où le logiciel WINTEX.
Ensuite, il m'a fallut construire, en assembleur, un désassembleur très court qui créait, à partir d'un listing désassemblé, un fichier au format de WINTEX, de telle sorte que je n'ai plus qu'à y adjoindre flèches et commentaires. D'où DESAS.
Pour finir, il m'a fallut sortir le source désassemblé de TELEMON V2.4 en totalité afin de l'annoter et de le comprendre. Au terme de ces 3 parties, le vrai travail à commencé. ET voici le résultat.

Dans ce livre, 16 Ko de langage machine 6502 les plus extraordinaires qui soient : Le cerveau du TELESTRAT.
L'empreinte de tous ceux qui ont jamais donné une partie d'eux-mêmes à l'ORIC : Andy Brown, Peter Halford, Denis Sebbag et enfin Fabrice Broche.
Ce livre est une mine d'or pour qui désire mieux comprendre comment fonctionne le logiciel TELESTRAT. De surcroit, il remplace avantageusement le manuel développeur, vendu 190 francs à une certaine époque, et quasiment inutilisable par le novice.

Alors lisez-le, relisez-le et imprégnez-vous des milliers d'instructions commentées qui le composent. On n'apprend jamais autant que lorsque l'on lit un programme bien écrit !

G. MEISTER

LE SYSTEME TELEMON

Le système TELEMON, au même titre que MS-DOS, UNIX, STOS, etc... est la base de fonctionnement du TELESTRAT. Il est le fruit de longues recherches (Théoriquement) et d'une programmation très fine (toujours théoriquement).

A sa racine, et à celle du TELESTRAT, figurent les routines de BOOT.

On appelle le BOOT le point 0 , ou point d'entrée, d'un système informatique ici en l'occurence : le BOOT consiste : vérifier que tout fonctionne ; placer quelques valeurs en mémoire et passer la main au langage.
Voici, détaillés, les diverses taches du BOOT :

- Le BOOT initialise les ports d'E/S (ACIA,VIAs,FDC,PSG)
- il installe logiquement le(s) drive(s) branché(s).
- il teste les périphériques
- il place en RAM les routines systèmes
- il ajuste les BUFFERS
- il lit les cartouches enfichées
- il place et active toutes les interruptions (IRQ,NMI,BRK...)

En outre, il existe deux entrées physiques au BOOT. Si le boot est activé par RESET ou la commande BASIC NMI, on appelle cela un RESET à chaud; c'est à dire que la RAM n'est pas vidée et que les variables systèmes sont, pour la plupart, inchangées. En outre, le vecteur VIRQ est laissé tel quel.

Si le BOOT est activé par la touche RESET+DEL ou la commande BASIC RESET il s'en suit un RESET à froid, c'est à dire que la RAM est initialisée et avec elle toutes les variables, buffers et routines système. Cela correspond presque à la mise sous tension de l'appareil. Presque car en fait la RAM utilisateur n'est pas modifiée et en outre le SED en mémoire y reste...

Le BOOT est la partie du TELEMON la plus importante. En effet, il est important d'être exhaustif dans la programmation des taches du BOOT sans quoi il s'en suit des bugs des plus droles aux plus gênants !
Sur la dernière version du TELEMON (2.4), la totalité des bugs ont été corrigés (au moins dans le BOOT). Ce qui ne veut pas dire que le BOOT est "optimisé", pour reprendre l'expression chère à Fabrice BROCHE...

On retrouve beaucoup d'erreurs simples d'optimisation (JMP au lieu de Bxx, JSR/RTS au lieu de JMP, peu de BYT $2C ou 24,etc...) et un certain gaspillage de la mémoire. Gaspillage qui, vous le constaterez par la suite, rend désuets tous les efforts ultérieurs pour économiser les octets.

LES INTERRUPTIONS

Fabrice BROCHE, le concepteur du TELEMON, a voulu (ou peut-être n'avait-il pas le choix ?) faire du TELESTRAT un machine "multitaches", entre guillemets car le multi-tasking va un peu plus loin qu'une bonne gestion de buffers et d'E/S par interruptions.

Il en résulte plusieurs niveaux d'interruptions, plus ou moins complexes, mais toujours très efficaces. Voici ces différents niveaux :

- Les IRQs, ou Interrupt ReQuest, qui sont les interruptions périphériques. Ces interruptions proviennent indifféremment du FDC ou des VIA. Elles induisent deux actions :

  * traitement de l'information dans le cas d'une IRQ FDC

  * traitement du travail invisible dans le cas d'une IRQ VIA

    Ce travail "invisible" consiste en :

    . Gestion des buffers selon les E/S (RS,CENTRONICS,MINITEL...)
    . Gestion du clavier, du joystick et/ou de la souris
    . Gestion des variables TIMER (horloges,TIMEU...)
    . Gestion de la vidéo (écrans, curseurs...)
    . Gestion des E/S et IRQ utilisateur.

- L'IRQ logicielle, le BRK en l'occurence, qui exécute une routine demandée par un pseudo paramètre lié au BReaK.

- Les NMI, Non Masquable Interruption, ou interruption mécanique (!), qui correspond au RESET à chaud par pression du RESET latéral.

Chaque type d'interruption dispose de ses routines de gestions, assez complexes, étant donnés le travail à accomplir, et surtout pour lesquelles il a fallut innover totalement.

On trouve par exemple la gestion de la prise RS232 éparpillée dans toutes les routines visitées lors d'une IRQ par VIA ou BRK...

En tous les cas, il y a beaucoup à apprencre de la partie du TELEMON dédiée à la gestion des interruptions.

LA GESTION DU CLAVIER

La gestion du clavier du TELESTRAT est à la fois le problème le plus simple et le plus compliqué. Le plus simple car la gestion du clavier est en fait la même que celle de l'ATMOS. La plus compliquée car il faut y ajouter les ports joystick et la bufférisation.

Pour ce qui est de la gestion de base, le principe est des plus simples : Pour savoir si une touche est pressée, on passe par le PSG et le VIA1 ses numéros de ligne et de colonne, et si on reçoit un "strobe", c'est que le contact est fait est que la touche est pressée. Trivial...

Quant il s'agit de mémoriser les dernières touches pressées, pour permettre une frappe "en avance", donc plus économique, le problème est plus ardu. Il faut en effet avant de savoir si une autre touche est pressée, vérifier d'abord si la touche précédemment pressée n'est pa sujette à une répétition... allez donc jeter un coup d'oeil au enchevêtrements de boucles, ça vaut le détour !

A cela, on ajoute les tests port gauche et port droit, afin d'envoyer éventuellement des valeurs de touches clavier (prioritaires d'ailleurs) dans le buffers. Le joystick ne pouvant se brancher qu'à gauche, quiqu'en dise la Doc..

Pour en finir, voici le shéma de la matrice du clavier :

colonne ligne	0	1	2	3	4	5	6	7
0	7	N	5	V		1	X	3
1	J	T	R	F		esc	Q	D
2	M	6	B	4	ctl	Z	2	C
3	K	9	;	-			\	'
4	spc	,	.	↑	sG	<-	↓	->
5	U	I	O	P	fc	del	]	[
6	Y	H	G	E		A	S	W
7	8	L	0	/	sD	ret		=

Vous remarquerez que lq colonne 4 est reservée au touches de fonctions, alors que la ligne 4 est reservée aux touches de déplacements...rusé !

LES FONCTIONS MATHEMATIQUES

Les calculs mathématiques représentent sans conteste la partie de loin la plus complexe du TELEMON. En effet, si après un peu de réflexion logique pure, on peut tracer un algorithme de gestion des E/S bufférisées, écrire des routines de calcul réel, ou pseudo-réel, et implémenter des fonctions transcendantes, ne peut se faire sans au moins les bases d'un raisonnement purement mathématique.

Avant tout un peu de théorie pour remttre les choses en place :

* Un nombre réel est un nombre qui ne peut s'écrire comme la division de deux entiers signés (ou relatifs). Son codage ne peut donc se faire en codant numérateur et dénominateur.

* Un polynome à variable réelle est une fonction injective (tout x a une image par cette fonction) qui à un réel associe un autre réel selon la formule : P(x)=a(n).x^n+a(n-1)x^(n-1)+..+a1x+a0.

  On appelle degré du polynome le plus grand n pour lequel a(n) est non nul.

* On appelle fonction transcendante toutes fonction qui ne peut avoir comme fonction égale, un polynome. Un nouveau problème de codage se pose puisque l'on ne peut pas coder simplement des coefficients a(n).

* On appelle limite d'une fonction au voisinage d'un point a la valeur l que tend à prendre cette fonction quand sa variable tend vers a. En théorie, les limites utiles sont remarquables (0,1,pi/4,etc...)

* Pour finir, on appelle développement limité (d.l.) d'une fonction au voisinage d'un point a à l'ordre n, le polynome dont les valeurs sont les plus proches de celles de la fonction quand la variable est proche de a. Le polynome est de degre n et la fonction est dérivable n fois et chaque dérivée doit etre définie au voisinage de a.

Pour construire un d.l., il suffit de considerer un polynome...

soit p(x)=4x2+5x-1. Supposons que P(x)=Q(x)=ax2+bx+c.
pour trouver c, il suffit de poser P(0)=-1 donc c=-1.
pour trouver b, il suffit de poser P'(0)=5 donc b=5/1=5.
pour trouver a, il suffit de poser P''(0)=8 donc a=8/2=4.

on a calculé un développement limité en 0 à l'ordre 3 de P(x).
un d.l. en 0 à l'ordre 1 est Q(x)=1, tout simplement.
on a divisé chaque facteur a,b,c par i!, le produit des nombres de 1 à son rang car à chaque dérivée, l'exposant de la variable est multiplié au coefficient.

Voyons comment on s'y prend en règle générale.
Cette formule est la formule de Taylor-Young :

$$Q(x)= \frac{P(a)}{n!}.(x-a)^n$$

Q(x) est un d.l. à l'ordre n, en a, de P(x).
parmi les d.l. connus, on trouve :

SIN(x)=x-x3/6+x5/120+...
COS(x)=1-x2/3+x4/24+...
EXP(x)=1+x+x2/2+x3/6+...
$(1-x)^n = 1-nx+.....$

ces D.L sont calculés en 0, ce qui est le plus courant.

Un russe notoire, et profondément haï de la gent estudiantine des facs de maths et des prépas de France et de Navarre, nommé Chebyshev, a eu un jour l'idée saugrenue de tenter de donner au D.L. une précision plus grande Il à alors trouvé des suites récurrentes permettant, non plus de calculer clairement les coefficients du d.l., mais de les approximer, de telle sorte que plus l'ordre du d.l. est grand, plus les coefficients s'éloignent des coeff. de Taylor-Young.
Ce qui est drole, c'est que si l'erreur maxi à été minimisée, l'erreur min, elle a été maximisée !!! mais bon, on calcule rarement sin(0), mais plutot sin(pi/6)...

Finalement, le seul vrai problème est de trouver un moyen de faire converge la fonction vers une limite, et qu'elle ne s'en eloigne pas trop ...
Ce qui est aisé pour les fonctions périodiques, mais bien plus complexe pour les autres (EXP,LOG,etc...). Voyez dans le listing.

Les d.l. de Chebyshev ont donc servi au codage des fonctions transcendantes sur ORIC, depuis le début.
Quant aux nombres réels, rien de plus simple :

si 1256=1,256*10^3 alors 127,5=1111111,1=1,1111111*2^6

avec les conventions suivantes pour l'exposant :

* s'il est à 128, alors il est nul
* s'il est >128 alors positif
* s'il est <128 alors négatif
* s'il est nul, alors le nombre est nul

En outre, le premier bit du nombre etant obligatoirement un 1, on se sert de sa position pour coder le signe du nombre (0=+, 1=-).

Quoi qu'il en soit, et pour plus de renseignements, reportez-vous au bouqui très exhaustif de F. Broche : L'ORIC à NU.

ET POUR FINIR...

Et pour finir, ma foi que dire ?

Que la majorité des routines du moniteur du TELESTRAT sont, entre 90 et 100 identiques à celles de la V1.1 de l'ATMOS. Parmi ces routines figurent les routines suivantes :

- fonctions maths (100%)
- graphismes (95%)
- mode texte (90%)
- imprimante (90%)
- calculs entiers (95%)
- clavier (100%)
- sons et musique (100%)

et j'en oublie surement. En bref, tout ce qui existait sur la V1.1 a été transporté et très peu modifié. A peine débuggé.

On retrouve donc, vraiment de Broche, et neuf sur le TELESTRAT :

- gestion VIDEOTEX
- gestion disquettes
- ports cartouches
- E/S plus diverses (VIA2,ACIA,etc..)
- CANAUX et BUFFERS
- interruptions

Et le boot ! vraiment la plus belle routine.

Quoiqu'il en soit, le TELEMON reste, pour le commun des mortels, donc moi, et pas mal d'entre vous, je disais donc le TELEMON reste un Chef-d'oeuvre de programmation. Vous pourrez en juger...

Et de la même manière que l'ORIC A NU a été mon tremplin dans la programmation, j'espère que ce livre sera, sinon le votre, au moins un outil précieux dans vos déambulations à travers cette merveilleuse machine qu'est

LE TELESTRAT

RESET

Action: Réinitialise les vecteurs et les routines système. Vérifie et installe les adresses systèmes.
Si DEL est pressée, on remet absolument tout à 0.
Si SHIFT droit est pressé, on place le minitel en sortie vidéo.

```
C000-78          SEI              on ne doit pas être dérangé
C001-D8          CLD              pas décimal
C002-A2 FF       LDX #$FF         pile à 0
C004-9A          TXS
C005-E8          INX
C006-8E 18 04    STX $0418        met à 0 la pile des banques
C009-20 A6 C2    JSR $C2A6        initialise les E/S (ACIA,VIAs...)
C00C-20 4F DA    JSR $DA4F        éteint le PSG
C00F-20 03 D9    JSR $D903        balaye le clavier dans KBDCOL
C012-A2 0F       LDX #$0F
C014-5E AE 02    LSR $02AE,X      met à 0 les 16 vecteurs d'E/S possibles
C017-CA          DEX              pour les 4 canaux
C018-10 FA       BPL $C014
C01A-A9 D0       LDA #$D0         envoie la commande %1101000 au FDC
C01C-20 D1 C4    JSR $C4D1        soit interdiction d'interruption
C01F-AD FA 02    LDA $02FA        le vecteur IRQ est OK ?
C022-C9 4C       CMP #$4C
C024-D0 0F       BNE $C035        non ------------------------------------------
C026-AD 6D 02    LDA $026D        oui, est-ce un DEL+RESET (reset à froid)?   I
C029-29 20       AND #$20                                                     I
C02B-D0 08       BNE $C035        non ----------------------------------------C
C02D-AD 0D 02    LDA $020D        oui, on isole b1 de FLGTEL (SED présent?)   I
C030-29 01       AND #$01                                                     I
C032-18          CLC              C=0                                         I
C033-90 03       BCC $C038     I---inconditionnel                             I
C035-A9 01       LDA #$01      I   force SED présent <--------------------------
C037-38          SEC           I   et C=1
C038-8D 0D 02    STA $020D     I-->dans FLGTEL
C03B-6E EE 02    ROR $02EE         b7 de FLGRST à 1 si DEL+RESET
C03E-30 03       BMI $C043            DEL+RESET -------------------------------
C040-4C AC C0    JMP $C0AC            RESET                                   I
C043-A2 2F       LDX #$2F          pour 48 <-----------------------------------
C045-BD 38 C8    LDA $C838,X       déplacements. On installe les vecteurs
C048-9D BE 02    STA $02BE,X       d'E/S par défaut en ADIOB
C04B-CA          DEX
C04C-10 F7       BPL $C045
C04E-A2 04       LDX #$04
C050-BD 5F C4    LDA $C45F,X       on place la racine du générateur aléatoire
C053-9D EF 02    STA $02EF,X       dans CSRND (réel 5 octets)
C056-CA          DEX
C057-10 F7       BPL $C050
C059-EA          NOP               ???
C05A-EA          NOP               ???
C05B-A2 03       LDX #$03
C05D-BD DC C2    LDA $C2DC,X       les lecteurs A à D sont-ils branchés ?
C060-8D 14 03    STA $0314
C063-A9 08       LDA #$08          commande SEEK au FDC (placer la tête sur piste 0)
C065-8D 10 03    STA $0310         dans FDCCR
C068-A8          TAY
```

```
C069-C8     INY          I->248 boucles
C06A-D0 FD  BNE $C069    I--pour attente de réponse
C06C-EA     NOP          ??? décidément
C06D-A0 40  LDY #$40
C06F-86 00  STX $00
C071-AD 10 03 LDA $0310     si b1 de ACIACR=0
C074-4A     LSR
C075-90 0A  BCC $C081       on installe le drive ----------------------------I
C077-E6 00  INC $00         on teste plus de 200 fois pour être sur          I
C079-D0 F6  BNE $C071                                                        I
C07B-88     DEY             mais alors vraiment,vraiment sur !               I
C07C-D0 F3  BNE $C071                                                        I
C07E-98     TYA             y=0, le nombre de pistes sera 0,drive absent     I
C07F-F0 05  BEQ $C086    I--inconditionnel                                   I
C081-4E 0D 02 LSR $020D  I  on indique SED présent <-------------------------I
C084-A9 AA  LDA #$AA     I  on indique 42 pistes double face
C086-9D 08 02 STA $0208,X I->dans le drive trouvé
C089-CA     DEX
C08A-10 D1  BPL $C05D       et on les teste tous
C08C-EA     NOP             ??? aie,aie,aie
C08D-E8     INX             X=0
C08E-BD 24 C5 LDA $C524,X   on installe la gestion IRQ,BRK et banques en $0400
C091-9D 00 04 STA $0400,X
C094-BD 64 C4 LDA $C464,X
C097-9D 00 B8 STA $B800,X   routine de chargement du STRATSED
C09A-BD E0 C2 LDA $C2E0,X
C09D-9D 00 06 STA $0600,X   routine d'installation de la routine des buffers
C0A0-BD EB C5 LDA $C5EB,X
C0A3-9D 00 07 STA $0700,X   routine des buffers qui sera en $C500 banque 0
C0A6-E8     INX
C0A7-D0 E5  BNE $C08E
C0A9-20 03 06 JSR $0603     installe la routine de gestion des buffers
C0AC-A9 00  LDA #$00        X=buffer clavier
C0AE-48     PHA          I->on sauve le numéro
C0AF-AA     TAX          I
C0B0-20 EA C4 JSR $C4EA  I  on initialise le buffer X
C0B3-68     PLA          I
C0B4-18     CLC          I
C0B5-69 0C  ADC #$0C     I  on passe au buffer suivant
C0B7-C9 30  CMP #$30     I  est-ce après le buffer SORTIE PARALLELE ?
C0B9-D0 F3  BNE $C0AE    I--non,on passe au suivant
C0BB-A9 00  LDA #$00
C0BD-8D 0E 02 STA $020E     0 Ko de ROM
C0C0-A9 40  LDA #$40        64 Ko de RAM
C0C2-8D 0F 02 STA $020F
C0C5-A2 07  LDX #$07
C0C7-BC 00 02 LDY $0200,X   on lit le statut des banques
C0CA-98     TYA
C0CB-29 10  AND #$10        on ignore ?
C0CD-D0 1C  BNE $C0EB       oui ----------------------------------------------
C0CF-98     TYA             non,on sauve la valeur                           I
C0D0-48     PHA                                                              I
C0D1-29 0F  AND #$0F        combien de mémoire ?                             I
C0D3-A8     TAY             +1                                               I
C0D4-C8     INY             dans Y                                           I
C0D5-68     PLA                                                              I
C0D6-29 20  AND #$20        est-ce une RAM ou une ROM                        I
C0D8-18     CLC                                                              I
C0D9-F0 09  BEQ $C0E4                                                        I
C0DB-98     TYA             + une ROM                                        I
```

```
C0DC-6D 0E 02   ADC $020E       on ajoute à la ROM totale
C0DF-8D 0E 02   STA $020E
C0E2-90 07      BCC $C0EB       inconditionnel
C0E4-98         TYA           + une RAM
C0E5-6D 0F 02   ADC $020F       on ajoute à la RAM totale
C0E8-8D 0F 02   STA $020F
C0EB-CA         DEX           banque suivante <
C0EC-D0 D9      BNE $C0C7
C0EE-2C EE 02   BIT $02EE       b7 de FLGRST=1 ?
C0F1-10 0E      BPL $C101       non, c'est un RESET à chaud
C0F3-A2 0B      LDX #$0B        oui, on place les vecteur VNMI,VIRQ et VAPLIC
C0F5-BD 86 C3   LDA $C386,X
C0F8-9D F4 02   STA $02F4,X
C0FB-CA         DEX
C0FC-10 F7      BPL $C0F5
C0FE-20 B1 D9   JSR $D9B1       et installe les valeurs clavier
C101-AD 6C 02   LDA $026C           <
C104-29 90      AND #$90        a-t-on pressé SHIFT droit ?
C106-F0 08      BEQ $C110     I--non
C108-AD 0D 02   LDA $020D     I  oui,on force b6 de FLGTEL à 1
C10B-09 40      ORA #$40      I  pour indiquer le mode Minitel
C10D-8D 0D 02   STA $020D     I
C110-20 5B DF   JSR $DF5B     I->initialise les valeurs écran TEXT
C113-20 56 DA   JSR $DA56       initialise l'imprimante
C116-20 BD D5   JSR $D5BD       initialise variables VDT
C119-20 AB DF   JSR $DFAB       initialise JOYstick,SOURIS et TIMERs
C11C-20 54 DB   JSR $DB54       initialise RS232
C11F-AD 75 02   LDA $0275
C122-4A         LSR             on isole le mode clavier
C123-29 03      AND #$03        dans b1-2-3 de FLGKBD
C125-00 52      BRK $52         et on place le type de clavier
C127-A9 80      LDA #$80
C129-00 00      BRK $00         on ouvre le clavier en entrée sur le canal 0
C12B-A9 88      LDA #$88
C12D-00 00      BRK $00         et l'écran fenêtre 0 en sortie
C12F-00 3C      BRK $3C         on met l'horloge à 0

C131-A9 8F      LDA #$8F        on ouvre le Minitel en sortie
C133-00 01      BRK $01         sur le canal 1
C135-A9 13      LDA #$13        indexe mode non rouleau et
C137-A0 C4      LDY #$C4        curseur éteint sur Minitel
C139-2C 0D 02   BIT $020D       est-on en mode minitel ?
C13C-50 08      BVC $C146       non
C13E-A9 8F      LDA #$8F        oui,on ouvre le minitel en sortie
C140-00 00      BRK $00         sur le canal 1
C142-A9 0D      LDA #$0D        on indexe mode rouleau et curseur allumé
C144-A0 C4      LDY #$C4
C146-00 15      BRK $15         on envoie les code au Minitel <
C148-2C EE 02   BIT $02EE       RESET à froid ?
C14B-10 26      BPL $C173       non
C14D-A9 9B      LDA #$9B
C14F-A0 C3      LDY #$C3
C151-00 14      BRK $14         on affiche TELESTRAT...
C153-A9 E1      LDA #$E1
C155-A0 C3      LDY #$C3
C157-00 14      BRK $14         on affiche Copyright ...
C159-00 25      BRK $25         on passe à la ligne
C15B-AD 0F 02   LDA $020F       on affiche la RAM
C15E-20 9B C2   JSR $C29B       en décimal deux chiffres
C161-A9 B9      LDA #$B9
C163-A0 C3      LDY #$C3
```

```
:C15B-A9 14     BRK $14          Ko RAM
:C15D-AD 0E 02  LDA $020E        puis la ROM                                  I
:C160-20 9B C2  JSR $C29B                                                     I
:C163-A9 C2     LDA #$C2         Ko ROM                                       I
:C165-A0 C3     LDY #$C3                                                      I
:C167-00 14     BRK $14                                                       I
:C169-20 BF C6  JSR $C6BF        l'imprimante est prête ? <-------------------
:C16C-D0 05     BNE $C173        oui ----------------------------------------
:C16E-00 25     BRK $25          non,on saute une ligne                       I
:C170-4C 88 C1  JMP $C188        BNE aurait été mieux...----------------------I I
:C173-2C EE 02  BIT $02EE        RESET à froid ? <----------------------------+--
:C176-10 06     BPL $C17E        non ----------------------------------------0
:C178-A9 01     LDA #$01         oui, on affiche "Imprimante"                 I
:C17A-A0 C4     LDY #$C4                                                      I
:C17C-00 14     BRK $14                                                       I
:C17E-2C EE 02  BIT $02EE        RESET à froid ? <----------------------------I
:C181-30 0F     BMI $C192        oui ----------------------------------------
:C183-20 68 C2  JSR $C268        on coupe l'entrée Minitel sur le canal 1     I
:C186-AE F4 02  LDX $02F4        et on exécute VNMI - banque X                I
:C189-AD F5 02  LDA $02F5        adresse AY                                   I
:C18C-AC F6 02  LDY $02F6                                                     I
:C18F-4C 5C C2  JMP $C25C        car changement de banque probable            I
:C192-A2 00     LDX #$00         on va afficher les drives <-----------------
:C194-BD 08 02  LDA $0208,X      le drive X est branché ?
:C197-D0 07     BNE $C1A0      ---oui
:C199-E8        INX            I  non
:C19A-E0 04     CPX #$04       I  on fait les quatre possibles
:C19C-D0 F6     BNE $C194      I
:C19E-F0 36     BEQ $C1D6      I  inconditionnel ---------------------------------
:C1A0-8A        TXA            -->                                             I
:C1A1-48        PHA                                                            I
:C1A2-AD 20 02  LDA $0220        sommes-nous au début de la ligne ?           I
:C1A5-F0 04     BEQ $C1AB      ---oui                                         I
:C1A7-A9 2C     LDA #$2C       I  non, on affiche ","                         I
:C1A9-00 10     BRK $10        I                                              I
:C1AB-A9 CC     LDA #$CC       -->on affiche "Drive :"                        I
:C1AD-A0 C3     LDY #$C3                                                      I
:C1AF-00 14     BRK $14                                                       I
:C1B1-68        PLA                                                           I
:C1B2-48        PHA                                                           I
:C1B3-AA        TAX                                                           I
:C1B4-BD DC C2  LDA $C2DC,X      on indique au FDC la commande ixx00100       I
:C1B7-8D 14 03  STA $0314        soit activer le drive xx ,face B             I
:C1BA-8E 0C 02  STX $020C        et drive courant dans DRVDEF,drive par défaut I
:C1BD-68        PLA                                                           I
:C1BE-AA        TAX                                                           I
:C1BF-8A        TXA              on prend le numéro du drive <---------------  I
:C1C0-18        CLC                                                          I  I
:C1C1-69 41     ADC #$41         on ajoute "A"                               I  I
:C1C3-00 10     BRK $10          et on affiche                               I  I
:C1C5-E8        INX                                                          I  I
:C1C6-E0 04     CPX #$04         pour tous les drives                        I  I
:C1C8-F0 11     BEQ $C1DB      ---(4 maxi)                                   I  I
:C1CA-BD 08 02  LDA $0208,X I    on lit le drive X                           I  I
:C1CD-F0 F6     BEQ $C1C5   I    vide ?                                      I  I
:C1CF-A9 2D     LDA #$2D    I    non,on affiche "-"                          I  I
:C1D1-00 10     BRK $10     I    et on continue                              I  I
:C1D3-4C C9 C1  JMP $C1C9   I    pourquoi pas BNE ??? ------------------------  I
:C1D6-AD 20 02  LDA $0220    I    on est au début de la ligne ? <---------------
:C1D9-F0 02     BEQ $C1DD   I    oui
```

```
:1E5-00 25      BRK $25      -->non on saute une ligne
:1E7-A9 D3      LDA #$D3       on affiche TELEMON V2.4 etc...
:1E9-A0 C3      LDY #$C3
:1EB-00 14      BRK $14
:1ED-A9 00      LDA #$00       on met 0 en définition pour la banque 0 (RAM)
:1EF-8D 00 02   STA $0200
:1F2-AD 0D 02   LDA $020D
:1F5-4A         LSR            STRATSED présent ?
:1F6-B0 0F      BCS $C207      non ----------------------------------------
:1F8-A9 19      LDA #$19       oui, on affiche "insérez une disquette"     I
:1FA-A0 C4      LDY #$C4       clignotant                                  I
:1FC-00 14      BRK $14                                                    I
:1FE-20 00 B8   JSR $B800      on attend une disquette STRATSED,on charge le SED
:201-A9 30      LDA #$30       on arrête de clignoter                      I
:203-A0 C4      LDY #$C4                                                   I
:205-00 14      BRK $14                                                    I
:207-A2 02      LDX #$02       <-------------------------------------------
:209-BD 5C C4   LDA $C45C,X    on place l'extension COM
:20C-9D 5D 05   STA $055D,X    dans l'extension par défaut
:20F-CA         DEX
:210-10 F7      BPL $C209
:212-20 00 06   JSR $0600      affiche le copyright des banques
:215-20 68 C2   JSR $C268      ferme le minitel sur le canal 1
:218-AD 0D 02   LDA $020D      SED présent ?
:21B-4A         LSR
:21C-B0 2F      BCS $C24D      non ----------------------------------------
:21E-A9 55      LDA #$55       indexe BONJOUR                              I
:220-A0 C4      LDY #$C4                                                   I
:222-A2 07      LDX #$07                                                   I
:224-00 24      BRK $24        dans BUFNOM                                 I
:226-A2 00      LDX #$00                                                   I
:228-A9 7D      LDA #$7D       appel a XTRVNM                              I
:22A-A0 FF      LDY #$FF                                                   I
:22C-20 5C C2   JSR $C25C      BONJOUR.COM est-il présent ?                I
:22F-F0 1C      BEQ $C24D      non ----------------------------------------0
:231-AD 0D 02   LDA $020D      oui, on l'indique dans FLGTEL               1
:234-09 04      ORA #$04                                                   I
:236-8D 0D 02   STA $020D                                                  I
:239-A2 06      LDX #$06                                                   I
:23B-BD 00 02   LDA $0200,X    on lit le statut des banques                I
:23E-C9 EF      CMP #$EF       %11101111, il ne faut pas ignorer la banque I
:240-F0 05      BEQ $C247 ------alors on se branche dessus                 I
:242-CA         DEX          I                                             I
:243-10 F6      BPL $C23B    I                                             I
:245-30 06      BMI $C24D    I ---inconditionnel,on exécute l'APLIC        I
:247-A9 00      LDA #$00     I-+->X contient le numéro de la banque        I
:249-A0 C0      LDY #$C0       I  on exécute le logiciel qui s'y trouve    I
:24B-D0 0F      BNE $C25C      I  inconditionnel ------------------------- I
:24D-A2 00      LDX #$00       I->on affiche le curseur <-----------------+--
:24F-00 35      BRK $35                                                   I
:251-AE FD 02   LDX $02FD        y-a-t-il une cartouche application ?      I
:254-30 30      BMI $C286        non, on saute                             I
:256-AD FE 02   LDA $02FE        on indexe l'adresse de l'application      I
:259-AC FF 02   LDY $02FF        VAPLIC (port droit)                       I
:25C-8D 15 04   STA $0415        et on exécute sur la banque X <-----------
:25F-8C 16 04   STY $0416
:262-8E 17 04   STX $0417
:265-4C 0C 04   JMP $040C        par EXBNK
```

ETEINT L'ENTREE MINITEL SUR LE CANAL 1

Action : vide le buffer d'entrée du Minitel et ferme l'entrée série sur le canal

numéro 1.

```
C268-58        CLI             en $C268, on autorise les interruptions.
C269-A9 02     LDA #$02        en $C269, on ne touche pas aux interruptions.
C26B-85 44     STA $44
C26D-A5 44     LDA $44
C26F-D0 FC     BNE $C26D       on attend 3 dixièmes de secondes
C271-A2 0C     LDX #$0C        on vide le buffer série ACIA entrée
C273-00 57     BRK $57         par XVIDBU
C275-AD 1E 03  LDA $031E
C278-29 F3     AND #$F3        on force b2 à 0
C27A-09 08     ORA #$08        et       b3 à 1
C27C-8D 1E 03  STA $031E       dans le registre de commande de l'ACIA
C27F-A9 8F     LDA #$8F        on ferme le minitel en sortie sur le canal 1
C281-00 05     BRK $05         pourquoi pas JMP $C723 ???
C283-60        RTS
```

????????

Action: lit un code au clavier, si c'est CTRL-C, saute en $9000, si c'est CTRL-A, se branche en $E000 sur la banque 0. Cette routine n'est en fait jamais appelée (il faut qu'aucun langage ne soit présent, ni aucune application) il s'agit donc probablement d'une routine de mise au point.

```
C284-00 10     BRK $10         'lit un code <--------------------
C286-00 0C     BRK $0C         'demande une touche au clavier   I
C288-C9 03     CMP #$03        'CTRL-C ?                        I
C28A-D0 03     BNE $C28F       'non, on saute ----------------  I
C28C-20 00 90  JSR $9000       'oui, on exécute en $9000     I  I
C28F-C9 01     CMP #$01        'CTRL-A ? <-------------------  I
C291-D0 F1     BNE $C284       'non, on insiste... ------------
C293-A9 00     LDA #$00        'oui,on indexe $E000, banque 0 (RAM)
C295-A0 E0     LDY #$E0
C297-A2 00     LDX #$00
C299-F0 C1     BEQ $C25C       'et on saute
```

AFFICHE A EN DECIMAL SUR 3 CHIFFRES

Action: En entrée, A contient un nombre qui sera affiché sur le canal 0 par le biais de la routine XDECIM.
En sortie, Y=0.

```
C29B-A0 00     LDY #$00        'AY contient A
C29D-A2 20     LDX #$20        'le caractère par défaut est " "
C29F-86 14     STX $14         'dans DEFAFF
C2A1-A2 01     LDX #$01        'il faut afficher 3 caractères (X+2)
C2A3-4C 39 CE  JMP $CE39       'on affiche par XDECIM
```

INITIALISE LES E/S

Action:place les premières valeurs, ou valeurs neutres dans les deux VIA et dans
l'ACIA 6551, ainsi que dans le FDC.
En sortie, X et Y inchangés, A vaut 204, #CC.

```
C2A6-A9 7F     LDA #$7F        interdit toute interruptions
C2A8-8D 0E 03  STA $030E       dans VIA 6522-1
C2AB-8D 2E 03  STA $032E       et VIA 6522-2
C2AE-8D 1D 03  STA $031D       et ACIA 6551
C2B1-A9 00     LDA #$00
C2B3-8D 14 03  STA $0314       aucune directive au FDC
C2B6-A9 FF     LDA #$FF        %11111111
C2B8-8D 03 03  STA $0303       port A en sortie sur VIA1
C2BB-A9 F7     LDA #$F7        %11110111
C2BD-8D 00 03  STA $0300       dans le port B du VIA1
C2C0-8D 02 03  STA $0302       port B en sortie sauf PB3
C2C3-A9 17     LDA #$17        %00010111
C2C5-8D 21 03  STA $0321       dans port A du VIA2
C2C8-8D 23 03  STA $0323       PA0-1-2-4 en sortie et PA3-5-6-7 en entrée
C2CB-A9 E0     LDA #$E0        %11100000
C2CD-8D 20 03  STA $0320       dans port B du VIA2
C2D0-8D 22 03  STA $0322       PB0..4 en entrée et PB5-6-7 en sortie
C2D3-A9 CC     LDA #$CC        %10101010 CA2 et CB2 en sortie/écriture VIADR
C2D5-8D 0C 03  STA $030C       dans V1PCR
C2D8-8D 2C 03  STA $032C       et V2PCR
C2DB-60        RTS
```

```
C2DC BYT %10000100,%10100100,%11000100,%11100100  octets de tests de drives ABCD
```

ROUTINE A TRANSFERER EN $0600

Action:en $0600, elle affiche les copyright des banques et éventuellement
exécute le logiciel qui s'y trouve.
en $0603, elle teste si les banques sont active, RAM ou ROM et place leu
octet de statut au cas échéant.

```
C2E0-4C 50 06  JMP $0650       ------------------------------------------------
C2E3-A9 00     LDA #$00                                                        I
C2E5-8D 21 03  STA $0321       force banque 0                                  I
C2E8-AA        TAX                                                             I
C2E9-BD 00 07  LDA $0700,X     envoie 256 octets de $0700                      I
C2EC-9D 00 C5  STA $C500,X     à $C500, routine de gestion des buffers         I
C2EF-E8        INX                                                             I
C2F0-D0 F7     BNE $C2E9                                                       I
C2F2-A2 07     LDX #$07        force banque 7                                  I
C2F4-8E 21 03  STX $0321       force banque X                                  I
C2F7-A0 00     LDY #$00                                                        I
C2F9-B9 00 FF  LDA $FF00,Y     lit un octet                                    I
C2FC-48        PHA                                                             I
C2FD-69 04     ADC #$04        boucle d'attente                                I
C2FF-90 FC     BCC $C2FD       en sortie, C=1                                  I
C301-68        PLA                                                             I
C302-D9 00 FF  CMP $FF00,Y     le code a-t-il été rafraichi ou gardé ?          I
```

```
C305-D0 14     BNE $C31B      non, on doit ignorer la banque ----------------  I
C307-C8        INY            a-t-on fait 256 tests ?                      I  I
C308-D0 EF     BNE $C2F9                                                   I  I
C30A-8C FB FF  STY $FFFB      oui, on met 0 en définition banque 0         I  I
C30D-AD FB FF  LDA $FFFB      on lit la définition.                        I  I
C310-CC FB FF  CPY $FFFB      est-elle toujours la même ?                  I  I
C313-D0 08     BNE $C31D   ---non, c'est de la ROM                         I  I
C315-C8        INY            I  Y=1, c'est de la RAM                      I  I
C316-D0 F2     BNE $C30A      I  a-t-on passé 256 valeurs ?                I  I
C318-A9 0F     LDA #$0F       I  oui, A=%00001111 (16 Ko RAM)              I  I
C31A-2C                       I  on saute l'instruction suivante           I  I
C31B-A9 10     LDA #$10       I  A=%0001000 (banque à ignorer) <-----------  I
C31D-9D 00 02  STA $0200,X --->banque X selon A (ignorer ou RAM 16Ko)          I
C320-C9 02     CMP #$02        (en fait, ici A=0 ou A=15 ou A=16)             I
C322-D0 03     BNE $C327       inconditionnel                                 I
C324-6C FC FF  JMP ($FFFC)                                                    I
C327-CA        DEX                                                            I
C328-D0 CA     BNE $C2F4       on fait toutes les banques                     I
C32A-A9 07     LDA #$07                                                       I
C32C-8D 21 03  STA $0321       on force la banque 7 et on sort.               I
C32F-60        RTS                                                            I
C330-A2 00     LDX #$00        banque 0  <-$0650-------------------------------
C332-20 5E 06  JSR $065E-----
C335-A2 06     LDX #$06     I  banque 6
C337-20 5E 06  JSR $065E---0  on affiche le copyright et on exécute
C33A-CA        DEX          I  exécute si une banque le demande.
C33B-D0 FA     BNE $C337    I
C33D-60        RTS          I
C33E-BD 00 02  LDA $0200,X I->on prend l'octet de définition de la banque X
C341-10 22     BPL $C365       y-a-t-il un copyright à afficher ?
C343-8E 21 03  STX $0321       oui,on force la banque X
C346-AD F8 FF  LDA $FFF8       on stocke l'adresse du copyright dans RESB
C349-85 02     STA $02
C34B-AD F9 FF  LDA $FFF9
C34E-85 03     STA $03
C350-A0 00     LDY #$00
C352-8E 21 03  STX $0321       on force banque X
C355-B1 02     LDA ($02),Y     on lit le copyright
C357-48        PHA
C358-A9 07     LDA #$07        bancue 7
C35A-8D 21 03  STA $0321
C35D-68        PLA
C35E-F0 05     BEQ $C365       est-ce 0 ? ------------------------------------
C360-00 10     BRK $10         non, on affiche et on continue                 I
C362-C8        INY                                                            I
C363-D0 ED     BNE $C352                                                      I
C365-BD 00 02  LDA $0200,X     <-------------------------------------------------
C368-0A        ASL             le programme doit-il se lancer automatiquement ?
C369-10 17     BPL $C382       non -------------------------------------------
C36B-8E 21 03  STX $0321                                                      I
C36E-AD FC FF  LDA $FFFC       on lit l'adresse d'exécution                   I
C371-AC FD FF  LDY $FFFD                                                      I
C374-8D FE 02  STA $02FE       dans VAPLIC                                    I
C377-8C FF 02  STY $02FF                                                      I
C37A-8E FD 02  STX $02FD                                                      I
C37D-A9 07     LDA #$07        on force banque 7                              I
C37F-8D 21 03  STA $0321       et on sort                                     I
C382-60        RTS             <-----------------------------------------------
```

```
C386 BYT 07,92,C3              $C392,banque 7,adresse par défaut du RESET à chauc
C389 JMP $0000                 vecteur non répertorié
C38C JMP $0406                 vecteur IRQ
C38F BYT 80,00,00              valeurs nulle du APLIC. banque et adresse.
```

NMI PAR DEFAUT

Action:Cette routine est exécutée si on actionne le poussoir NMI (RESET à chaud) avant qu'une application ait été lancée. Elle affiche la fameuse "Logiciel ecrit par Fabrice BROCHE" qui ne peut que nous rappeler la non moins fameuse "SOFTWARE BY PETER HALFORD AND ANDY BROWN" de la ROM V1.0.

```
C392-A9 33     LDA #$33        indexe "Logiciel..."
C394-A0 C4     LDY #$C4
C396-00 14     BRK $14         affiche
C398-4C 98 C3  JMP $C398       et boucle indéfiniment
```

CODES ET CHAINES DIVERSES

Usage:ces codes sont utilisés lors de l'initialisation par RESET ou NMI.

```
C398 BYT 0C                    code CLS
C39C BYT 96,95,94,93,92,91,90  codes de couleurs dégradés
C3A4 BYT " TELESTRAT "
C3AF BYT 90,91,92,93,94,95,96  codes de couleur
C3B8 BYT 00                    terminateur de chaine

C3B9 BYT " Ko RAM, ",00

C3C2 BYT " Ko ROM",CR,LF,00

C3CC BYT " Drive:",00

C3D3 BYT CR,LF,"TELEMON V2.4"
C3E1 BYT CR,LF,"(c) 1986 "
C3EC BYT "ORIC International"
C3FE BYT CR,LF,00

C401 BYT LF,"Imprimante",0

C40D BYT 1B,3A,69,43,11,00     ESC,PRO2,START,ROULEAU,Con - instructions VDT
C413 BYT 1B,3A,6A,43,14,00     ESC,PRO2,STOP,ROULEAU,Coff

C419 BYT 8C                    code clignotant
C41A BYT "Inserez une disquette",00

C430 BYT CR,18,00              annulation clignotant

C433 BYT "Logiciel ecrit par Fabrice BROCHE",00

C455 BYT "BONJOUR","COM"

C45F BYT 80,4F,C7,52,58        racine du générateur aléatoire, soit 0.811630249
```

ZONE A TRANSFERER EN B800

```
ction:Charge le STRATSED s'il y a lieu et exécute l'amorce du SED (programme
      situé à la fin du premier secteur de SED).


:464-A9 E8      LDA #$E8        %11101000 soit banque 0
:466-8D 21 03   STA $0321       dans V2DRA
:469-A9 00      LDA #$00
:46B-A0 C1      LDY #$C1
:46D-85 00      STA $0C         $C100 dans RES
:46F-84 01      STY $01
:471-A2 01      LDX #$01        on lit piste 0 secteur 1
:473-8E 12 03   STX $0312       1 dans le registre de secteur
:476-20 4F B8   JSR $B84F       on lit le secteur
:479-AD 00 C1   LDA $C100       on prend l'indicateur
:47C-D0 2F      BNE $C4AD       est-ce le bon SED ?
:47E-AD 03 C1   LDA $C103       oui, on lit l'adresse
:481-AC 04 C1   LDY $C104       ou il faut charger le SED
:484-85 00      STA $00         dans RES
:486-84 01      STY $01
:488-EC 01 C1   CPX $C101       a-t-on lu toute la piste ? <---------------
:48B-D0 09      BNE $C496    ---non                                       I
:48D-A9 58      LDA #$58     I  oui, %01011000                           I
:48F-20 6D B8   JSR $B86D     I  soit deplacer la tete d'une piste        I
:492-A2 00      LDX #$00     I  compteur de secteur à 0                  I
:494-EA         NOP          I  tss tss...                               I
:495-EA         NOP          I                                           I
:496-E8         INX          -->on lit un secteur du SED                 I
:497-8E 12 03   STX $0312                                                I
:49A-20 4F B8   JSR $B84F                                                I
:49D-E6 01      INC $01         on passe à la page suivante              I
:49F-CE 02 C1   DEC $C102       a-t-on fait tous les secteurs du SED ?   I
:4A2-D0 E4      BNE $C488       non ------------------------------------
:4A4-20 05 C1   JSR $C105       oui, on exécute l'amorce du SED.
:4A7-AD FB FF   LDA $FFFB       on lit l'état de la banque
:4AA-8D 00 02   STA $0200       dans l'octet de STATUS banque 0
:4AD-A9 EF      LDA #$EF        et on repasse sur la banque 7
:4AF-8D 21 03   STA $0321
:4B2-60         RTS
:4B3-A9 88      LDA #$88        %10001000 dans FDCCR <---$B84F-------------------
:4B5-8D 10 03   STA $0310       soit lire un secteur                            I
:4B8-A0 04      LDY #$04                                                        I
:4BA-88         DEY             délai pour attendre la fin de la lecture        I
:4BB-D0 FD      BNE $C4BA                                                       I
:4BD-AD 10 03   LDA $0310    -->on lit le registre de commande                  I
:4C0-4A         LSR          I  opération terminée ?                            I
:4C1-90 1E      BCC $C4E1    I  oui ------------------------------------------- I
:4C3-AD 18 03   LDA $0318<-II   non, est-on en DATA READY ?                   I I
:4C6-30 F5      BMI $C4BD    I --non, on boucle                               I I
:4C8-AD 13 03   LDA $0313    I  oui,on lit le numéro de piste                 I I
:4CB-91 00      STA ($00),YI    dans le buffer                                I I
:4CD-C8         INY          I  et on pointe sur la position suivante         I I
:4CE-4C 5F B8   JMP $B85F---    pourquoi pas BNE ???                          I I
:4D1-8D 10 03   STA $0310       ($B86D) on envoie la commande A               I I
:4D4-A0 03      LDY #$03        on attend la réponse                          I I
:4D6-88         DEY                                                           I I
:4D7-D0 FD      BNE $C4D6                                                     I I
```

```
C4D9-AD 10 03   LDA $0310 .  -->ordre traité ?                          I I
C4DC-4A         LSR          I                                          I I
C4DD-B0 FA      BCS $C4D9    ---non                                     I I
C4DF-60         RTS             oui, on sort                            I I
C4E0-EA         NOP                                                     I I
C4E1-AD 10 03   LDA $0310       on lit le retour <----------------------- I
C4E4-29 1C      AND #$1C        y-a-t-il une erreur ?                     I
C4E6-F0 F7      BEQ $C4DF       non, RTS                                  I
C4E8-D0 C9      BNE $C4B3       oui, on recommence ------------------------
```

ROUTINES DE GESTION DES BUFFER

```
Principe:X contient le numéro du buffer, A la donnée et AY s'il y a deux données
         + pour lire une donnée          V=0 C=0
         + pour ecrire une donnée        V=1 C=0
         + pour initialiser un buffer    V=0 C=1
         + pour vider un buffer          V=1 C=1 Z=1
         + pour tester un buffer         V=1 C=1 Z=0
```

BUFFER PAR DEFAUT

```
C4EA-86 02      STX $02         on sauve l'index du buffer
C4EC-8A         TXA
C4ED-A2 FF      LDX #$FF        on calcule la postion des 4 octets
C4EF-38         SEC             de définition du buffer X
C4F0-E9 03      SBC #$03
C4F2-E8         INX
C4F3-B0 FA      BCS $C4EF
C4F5-BD AF C6   LDA $C6AF,X     on lit l'adresse de debut
C4F8-85 00      STA $00         dans RES
C4FA-BD B0 C6   LDA $C6B0,X
C4FD-85 01      STA $01
C4FF-BD B1 C6   LDA $C6B1,X     et l'adresse de fin (non incluse dans le buffer)
C502-BC B2 C6   LDY $C6B2,X     dans AY
C505-A6 02      LDX $02         et on initialise
```

INITIALISE UN BUFFER

```
C507-2C 18 C5   BIT $C518       adresse de début dans RES
C50A-50 08      BVC $C514       adresse de fin dans AY
```

VIDE UN BUFFER

```
C50C-A9 00      LDA #$00
C50D-2C         BYT $2C
```

TESTE UN BUFFER

```
C50F-A9 01      LDA #$01
C511-2C 24 C5   BIT $C524
C514-38         SEC
```

```
C515-4C 09 04   JMP $0409
```

LIT UNE DONNEE DANS UN BUFFER

```
C518-2C 18 C5   BIT $C518
C51B-50 03      BVC $C520
```

ECRIT UNE DONNEE DANS UN BUFFER

```
C51D-2C 24 C5   BIT $C524
C520-18         CLC
C521-4C 09 04   JMP $0409
```

ROUTINE A TRANSFERER EN $0400

Action: La page 4 contient les routines d'interfaçage avec les banques. Les routines sont les suivantes :
$0400 : termine une IRQ non BRK
$0403 : termine une IRQ BRK
$0406 : vecteur terminal des IRQ
$040C : EXBNK exécution d'une routine sur une banque donnée
$0411 : lecture d'un octet sur la banque contenant le BRK

Tous ces vecteurs peuvent bien sur être détournés, à condition bien sur de respecter le protocole.

```
C524-4C 93 04   JMP $0493       $0400 - fin d'IRQ
C527-4C A1 04   JMP $04A1       $0403 - fin de BRK
C52A-4C 7E 04   JMP $047E       $0406 - IRQ                          EXBNK
C52D-4C 19 04   JMP $0419       $0409 - gestion des buffers.
C530-4C 36 04   JMP $0436       $040C - gestion des banques
C533            RES 1           $040F - tampon pour banque lors d'IRQ
C534            RES 1           $0410 - tampon pour banque lors de BRK
C535-4C AF 04   JMP $04AF       $0411 - lecture d'une donnée durant BRK
C538-4C 00 00   JMP $0000       $0414 - vecteur d'execution inter-banque  VEXBNK
C53B            RES 1           $0417 - banque cible pour $040C           BNKCIB
C53C            RES 1           $0418 - pointeur de pile des banques

C53D-08         PHP             ($419) on sauve P
C53E-78         SEI             car on interdit les interruptions
C53F-48         PHA
C540-AD 21 03   LDA $0321       on force la banque 0
C543-29 F8      AND #$F8        (RAM) car les buffers
C545-8D 21 03   STA $0321       sont en RAM
C548-68         PLA
C549-20 00 C5   JSR $C500       et on exécute la routine des buffers
C54C-A8         TAY             on préserve la donnée
C54D-AD 21 03   LDA $0321
C550-09 07      ORA #$07        on repasse sur la banque 7
C552-8D 21 03   STA $0321
C555-6A         ROR             on préserve C
C556-28         PLP
```

```
:557-0A        ASL
:558-98        TYA             C et A sont représentatifs du résultat
:559-60        RTS

:55A-08        PHP             ($436)
:55B-78        SEI
:55C-48        PHA             on sauve A et X
:55D-8A        TXA
:55E-48        PHA
:55F-AD 21 03  LDA $0321       on sauve la banque actuelle dans la pile locale
:562-AE 18 04  LDX $0418       dont le pointeur est en $0418
:565-9D C8 04  STA $04C8,X
:568-EE 18 04  INC $0418
:56B-68        PLA
:56C-AA        TAX
:56D-AD 17 04  LDA $0417       A=banque cible
:570-20 6A 04  JSR $046A       on force la banque A
:573-68        PLA
:574-28        PLP
:575-20 14 04  JSR $0414       on execute le vecteur VEXBNK
:578-08        PHP
:579-78        SEI
:57A-48        PHA
:57B-8A        TXA
:57C-48        PHA
:57D-CE 18 04  DEC $0418       on restaure la banque d'avant
:580-AE 18 04  LDX $0418       l'appel
:583-BD C8 04  LDA $04C8,X
:586-20 6A 04  JSR $046A
:589-68        PLA
:58A-AA        TAX
:58B-68        PLA
:58C-28        PLP             et on sort avec A,X,Y et P préservés et
:58D-60        RTS             représentatifs de l'appel.

:58E-08        PHP             ($46A)
:58F-78        SEI
:590-29 07     AND #$07        il n'y a que 7 banques...
:592-8D C7 04  STA $04C7       dans une variable temporaire
:595-AD 21 03  LDA $0321
:598-29 F8     AND #$F8        on force la banque A
:59A-0D C7 04  ORA $04C7
:59D-8D 21 03  STA $0321
:5A0-28        PLP
:5A1-60        RTS

:5A2-85 21     STA $21         ($47E) on sauve A
:5A4-AD 21 03  LDA $0321       on force la banque 7
:5A7-29 07     AND #$07
:5A9-8D 0F 04  STA $040F       après avoir sauve la banque courante
:5AC-AD 21 03  LDA $0321
:5AF-09 07     ORA #$07
:5B1-8D 21 03  STA $0321
:5B4-4C 68 C8  JMP $C868       et on execute les IRQ

:5B7-AD 21 03  LDA $0321       ($493)
:5BA-29 F8     AND #$F8        on restaure la banque par $40F
:5BC-0D 0F 04  ORA $040F       donc en sortie d'IRQ
:5BF-8D 21 03  STA $0321
:5C2-A5 21     LDA $21         on recupère A
```

```
C5C4-40        RTI              et on sort

C5C5-48        PHA              ($4A1)
C5C6-AD 21 03  LDA $0321        on force la banque selon
C5C9-29 F8     AND #$F8         le contenu de $0410
C5CB-0D 10 04  ORA $0410
C5CE-8D 21 03  STA $0321        tout en préservant A
C5D1-68        PLA
C5D2-60        RTS

C5D3-AD 21 03  LDA $0321        ($4AF)  on force la banque selon $0410
C5D6-29 F8     AND #$F8         sauvé par BRK
C5D8-0D 10 04  ORA $0410
C5DB-8D 21 03  STA $0321
C5DE-B1 15     LDA ($15),Y      on lit l'octet pointé indirectement
C5E0-48        PHA              sur la banque de l'appelant
C5E1-AD 21 03  LDA $0321        on repasse sur la banque 7
C5E4-09 07     ORA #$07
C5E6-8D 21 03  STA $0321
C5E9-68        PLA              et on sort avec la donnée lue dans A
C5EA-60        RTS
```

ROUTINE A TRANSFERER EN $0700

```
Action:Cette routine, normalisée selon la gestion d'E/S, gère les buffers
       intégrés du TELESTRAT (clavier, série in/out, parrallèle) mais aussi les
       buffer crées par l'utilisateur.
       le protocole est ainsi fait :
       C=0 V=0       lecture dans A (C=1 si buffer vide)
           V=1       écriture de A  (C=1 si buffer plein)
       C=1 V=0       initialise un buffer (en sortie, A=0, Y=255,Z=1)
           V=1 Z=0   teste le buffer (C=1 si vide)
               Z=1   vide le buffer (A et P comme pour C=1/V=0)
       En sortie, en règle générale, C est à 1 si l'opération ne s'est pas
       déroulée correctement (buffer plein en écriture, vide en lecture, etc...)


C5EB-90 4C     BCC $C639        si C=0, c'est une lecture/écriture ----------------
C5ED-50 0F     BVC $C5FE        C=1 si V=0 on initialise un buffer ------------- I
C5EF-A8        TAY                                                              I I
C5F0-F0 2C     BEQ $C61E        C=1 V=1 si Z=1 on vide le buffer  ------------- I I

C5F2-BD 88 C0  LDA $C088,X      sinon, on teste le buffer                      I I I
C5F5-1D 89 C0  ORA $C089,X      longueur=0 ?                                   I I I
C5F8-F0 02     BEQ $C5FC        oui C=1                                        I I I
C5FA-18        CLC              non C=0                                        I I I
C5FB-60        RTS                                                             I I I
C5FC-38        SEC                                                             I I I
C5FD-60        RTS                                                             I I I

C5FE-85 02     STA $02          on initialise avec fin+1 dans RESB <----------+-- I
C600-84 03     STY $03                                                        I   I
C602-38        SEC                                                            I   I
C603-E5 00     SBC $00          fin-debut                                     I   I
C605-9D 8A C0  STA $C08A,X      dans longueur du buffer                       I   I
C608-98        TYA                                                            I   I
C609-E5 01     SBC $01                                                        I   I
C60B-9D 8B C0  STA $C08B,X                                                    I   I
```

```
C60E-8A        TXA                                                    I   I
C60F-69 03     ADC #$03                                               I   I
C611-AA        TAX                                                    I   I
C612-A0 03     LDY #$03                                               I   I
C614-B9 00 00  LDA $0000,Y    pourquoi pas LDA $00,Y ? étiquette peut-être I   I
C617-9D 7F C0  STA $C07F,X    on place adresse de début et de fin (exclue) I   I
C61A-CA        DEX                                                    I   I
C61B-88        DEY                                                    I   I
C61C-10 F6     BPL $C614                                              I   I
C61E-A9 00     LDA #$00       vide le buffer <-------------------------------   I
C620-9D 88 C0  STA $C088,X    longueur utilisée = 0                            I
C623-9D 89 C0  STA $C089,X                                                     I
C626-BD 82 C0  LDA $C082,X    adresse de fin                                   I
C629-9D 84 C0  STA $C084,X    dans pointeur de lecture                         I
C62C-9D 86 C0  STA $C086,X    et ponteur d'écriture                            I
C62F-BD 83 C0  LDA $C083,X                                                     I
C632-9D 85 C0  STA $C085,X                                                     I
C635-9D 87 C0  STA $C087,X                                                     I
C638-60        RTS            et on sort                                      <

C639-70 26     BVS $C661      C=0 si V=1 on écrit -============================
C63B-20 07 C5  JSR $C507      sinon, lecture. buffer vide ?                    >
C63E-B0 20     BCS $C660      oui on sort C=1                                  I
C640-BD 86 C0  LDA $C086,X    prend pointeur de lecture dans AY                I
C643-BC 87 C0  LDY $C087,X                                                     I
C646-20 A6 C5  JSR $C5A6      on calcule la prochaine position de lecture      I
C649-9D 86 C0  STA $C086,X    dans le pointeur de lecture                      I
C64C-98        TYA                                                            I
C64D-9D 87 C0  STA $C087,X                                                     I
C650-BD 88 C0  LDA $C088,X    on décrémente le pointeur d'écriture             I
C653-D0 03     BNE $C658                                                       I
C655-DE 89 C0  DEC $C089,X                                                     I
C658-DE 88 C0  DEC $C088,X                                                     I
C65B-A0 00     LDY #$00       on lit le code dans A                            I
C65D-B1 24     LDA ($24),Y                                                     I
C65F-18        CLC            lecture OK, C=0                                  I
C660-60        RTS                                                             I

C661-48        PHA            <-----------------------------------------------
C662-BD 88 C0  LDA $C088,X    longueur utilisée < longueur du buffer ?
C665-DD 8A C0  CMP $C08A,X
C668-BD 89 C0  LDA $C089,X
C66B-FD 8B C0  SBC $C08B,X
C66E-B0 1F     BCS $C68F      non, on récupère la donnée et on sort
C670-BD 84 C0  LDA $C084,X    on calcule la prochaine position
C673-BC 85 C0  LDY $C085,X    d'écriture
C676-20 A6 C5  JSR $C5A6
C679-9D 84 C0  STA $C084,X    dans le pointeur d'écriture
C67C-98        TYA
C67D-9D 85 C0  STA $C085,X
C680-FE 88 C0  INC $C088,X    on ajoute 1 à la longueur utilisée
C683-D0 03     BNE $C688
C685-FE 89 C0  INC $C089,X
C688-A0 00     LDY #$00
C68A-68        PLA
C68B-91 24     STA ($24),Y    on écrit la donnée
C68D-18        CLC            C=0 et A est intact
C68E-60        RTS
C68F-68        PLA            écriture ratée, C=1, A restauré
C690-60        RTS
```

```
C691-18        CLC            ($C5A6)
C692-69 01     ADC #$01       on incrémente l'adresse AY
C694-90 01     BCC $C697
C696-C8        INY
C697-DD 82 C0  CMP $C082,X
C69A-85 24     STA $24        AY < adresse de fin ?
C69C-98        TYA
C69D-FD 83 C0  SBC $C083,X
C6A0-90 08     BCC $C6AA      oui dans $24-25 ----------------------------------
C6A2-BD 80 C0  LDA $C080,X    non, on prend adresse de début du buffer          I
C6A5-BC 81 C0  LDY $C081,X                                                      I
C6A8-85 24     STA $24        dans $24-25                                       I
C6AA-84 25     STY $25        <-------------------------------------------------
C6AC-A5 24     LDA $24        AY contient l'adresse de la prochaine
C6AE-60        RTS            position dans le buffer
```

ADRESSES DES BUFFERS

contenu: deux adresses définissent l'adresse de début et de fin (exclue) par défaut. Les buffers utilisateurs n'ont pas de valeur par défaut, donc si on tente des leur rendre une valeur par défaut, cette valeur sera prise dans la routine suivante !!!

```
C6AF BYT $C5C4,$C680          buffer clavier
C6B3 BYT $C680,$C800          buffer entrée série ACIA/MINITEL
C6B7 BYT $C800,$CA00          buffer sortie série ACIA/MINITEL
C6BB BYT $CA00,$D200          buffer sortie parrallèle (imprimante)
```

TESTE SI L'IMPRIMANTE EST BRANCHEE

Principe: On génère un STROBE et on attend la réponse (ACKnowledge) de l'imprimante. Si un ACK est lu, l'imprimante est bien branchée.

```
C6BF-A2 00     LDX #$00       on met un 0 sur le port A
C6C1-8E 01 03  STX $0301      du VIA 1
C6C4-AD 00 03  LDA $0300      on force PB4 à 0
C6C7-29 EF     AND #$EF       %11101111, soit pas de STROBE
C6C9-8D 00 03  STA $0300      on dépose B sur le bus
C6CC-09 10     ORA #$10       puis on génère un STROBE par PB4
C6CE-8D 00 03  STA $0300
C6D1-AD 0D 03  LDA $030D   -->on lit IFR
C6D4-29 02     AND #$02    I  transition CA1 (ACK) ?
C6D6-D0 04     BNE $C6DC   I  oui, l'imprimante est branchée ------------------
C6D8-CA        DEX         I  non                                              I
C6D9-D0 F6     BNE $C6D1   ---on fait le test 256 fois pour être sur           I
C6DB-60        RTS            et on sort si l'imprimante n'a pas répondu       I
C6DC-AD 0D 02  LDA $020D      <-------------------------------------------------
C6DF-09 02     ORA #$02       on indique imprimante détectée
C6E1-8D 0D 02  STA $020D      dans FLGTEL
C6E4-60        RTS            et on sort
```

OUVRIR UNE E/S SUR UN CANAL

Principe: X contient 4 fois le numéro du canal (4 E/S par canal) et A le numéro

de l'E/S à ouvrir. Si elle est déja ouverte sur ce canal, on sort.
Sinon, on vérifie si elle est déja ouverte sur un des autres canaux.
Si ce n'est pas le cas, on l'active avant de l'ouvrir sur le canal.
En sortie, C=0 si l'E/S a été ouverte, C=1 si elle était déja ouverte.

```
C6E5-A2 00     LDX #$00        pour canal 0
C6E6           BYT $2C         sauter l'instruction suivante

C6E7-A2 04     LDX #$04        pour canal 1
C6EA           BYT $2C         sauter l'instruction suivante

C6EA-A2 08     LDX #$08        pour canal 2
C6ED           BYT $2C         sauter l'instruction suivante

C6ED-A2 0C     LDX #$0C        pour canal

C6F0-48        PHA             on sauve l'E/S
C6F1-68        PLA             on lit l'E/S
C6F2-DD AE 02  CMP $02AE,X     est-elle déja ouverte sur le canal en question ?
C6F5-F0 0D     BEQ $C704       oui C=1 et on sort ----------------------------------
C6F7-BC AE 02  LDY $02AE,X     le canal est-il saturé ?                           I
C6FA-10 09     BPL $C705       non, on peut ouvrir l'E/S ------------------------ I
C6FC-E8        INX                                                              I I
C6FD-48        PHA                                                              I I
C6FE-8A        TXA                                                              I I
C6FF-29 03     AND #$03        on teste les 4 E/S du canal                      I I
C701-D0 EE     BNE $C6F1                                                        I I
C703-68        PLA                                                              I I
C704-60        RTS             En sortie, A contient l'E/S <--------------------+--
C705-A0 0F     LDY #$0F        pour 4*4 E/S possibles <-------------------------
C707-D9 AE 02  CMP $02AE,Y     l'E/S est-elle ouverte sur un des 4 canaux ?
C70A-F0 0F     BEQ $C71B       oui,on ouvre simplement l'E/S sur le canal -------
C70C-88        DEY                                                              I
C70D-10 F8     BPL $C707                                                        I
C70F-86 19     STX $19         on sauve le numéro du canal                      I
C711-48        PHA                                                              I
C712-A0 80     LDY #$80        N=1 Z=0                                          I
C714-AA        TAX                                                              I
C715-20 1C C8  JSR $C81C       on active l'E/S dans X                           I
C718-A6 19     LDX $19         on récupère l'index                              I
C71A-68        PLA             l'E/S                                            I
C71B-9D AE 02  STA $02AE,X     et on ouvre l'E/S <-----------------------------
C71E-18        CLC             C=0
C71F-60        RTS
```

FERMER UNE E/S SUR UN CANAL

Principe: identique à l'ouverture, si ce n'est que si une E/S n'était ouverte que sur ce canal, elle est désactivée.
De plus C n'a pas de valeur particulière en sortie.
Si A contient la valeur 0, toutes les E/S sont fermées sur le canal en question.

```
C720-A2 00     LDX #$00        indexe canal 0
C722           BYT $2C         sauter instruction suivante
```

```
C723-A2 04      LDX #$04        indexe canal 1
C725            BYT $2C         sauter instruction suivante

C726-A2 08      LDX #$08        indexe canal 2
C728            BYT $2C         sauter instruction suivante

C729-A2 0C      LDX #$0C        indexe canal 3

C72B-A0 03      LDY #$03        pour 4 E/S
C72D-C9 00      CMP #$00        doit-t-on fermer toutes les E/S ?
C72F-F0 1D      BEQ $C74E       oui ----------------------------------------------
C731-DD AE 02   CMP $02AE,X     l'E/S est-elle ouverte ?                          I
C734-F0 05      BEQ $C73B       oui ------------------------                      I
C736-E8         INX                                        I                      I
C737-88         DEY                                        I                      I
C738-10 F7      BPL $C731                                  I                      I
C73A-60         RTS             l'E/S n'était pas ouverte  I                      I
C73B-5E AE 02   LSR $02AE,X     on ferme l'E/S <------------                      I
C73E-A2 0F      LDX #$0F        si l'E/S n'est ouverte sur aucun                  I
C740-DD AE 02   CMP $02AE,X     autre canal,                                      I
C743-F0 F5      BEQ $C73A                                                         I
C745-CA         DEX                                                               I
C746-10 F8      BPL $C740                                                         I
C748-AA         TAX                                                               I
C749-A0 81      LDY #$81        N=1 C=1 V=0                                       I
C74B-4C 1C C8   JMP $C81C       on désactive l'E/S                                I
C74E-5E AE 02   LSR $02AE,X     on ferme les 4 E/S sur le canal <-----------------
C751-E8         INX
C752-88         DEY
C753-10 F9      BPL $C74E
C755-60         RTS             et on sort
```

SAUTER UNE LIGNE SUR LE CANAL 0

```
C756-A9 0A      LDA #$0A        on envoie un CR (ASCII 13)
C758-20 5D C7   JSR $C75D       sur le canal 0
C75B-A9 0D      LDA #$0D        et un LF (ASCII 11)
```

ENVOYER UN CODE SUR UN CANAL

Principe:pour ne pas détruire les registres, on sauve la donnée à écrire et on met dans A 4 fois le numéro du canal. On envoie la donnée ensuite à toutes les E/S ouvertes sur le canal
Pour écrire une donnée sur une E/S, N=0 et C=1 avec la donnée dans A.

```
C75D-48         PHA             on sauve la donnée
C75E-A9 00      LDA #$00        indexe canal 0
C760-F0 0D      BEQ $C76F       inconditionnel

C762-48         PHA
C763-A9 04      LDA #$04        indexe canal 1
C765-D0 08      BNE $C76F       ..

C767-48         PHA
C768-A9 08      LDA #$08        indexe canal 2
```

```
C76A-D0 03      BNE $C76F

C76C-48         PHA
C76D-A9 0C      LDA #$0C        indexe canal 3

C76F-85 19      STA $19         on sauve le numéro du canal
C771-68         PLA
C772-85 1B      STA $1B         et la donnée
C774-A9 04      LDA #$04
C776-85 1A      STA $1A         pour 4 E/S à tester
C778-8A         TXA             on sauve X et Y
C779-48         PHA
C77A-98         TYA
C77B-48         PHA
C77C-A6 19      LDX $19         donnée dans X
C77E-BD AE 02   LDA $02AE,X     l'E/S est elle une sortie ?
C781-C9 88      CMP #$88
C783-90 16      BCC $C79B       non, on passe ----------------------------------
C785-0A         ASL                                                            I
C786-AA         TAX                                                            I
C787-BD BE 02   LDA $02BE,X     oui, on met son adresse dans                   I
C78A-8D F8 02   STA $02F8       $2F8-9                                         I
C78D-BD BF 02   LDA $02BF,X                                                    I
C790-8D F9 02   STA $02F9                                                      I
C793-A5 1B      LDA $1B         donnée dans A                                  I
C795-2C 95 C7   BIT $C795       N=0 C=1                                        I
C798-20 F7 02   JSR $02F7       on écrit la donnée sur l'E/S                   I
C79B-E6 19      INC $19         <---------------------------------------------
C79D-C6 1A      DEC $1A
C79F-D0 DB      BNE $C77C       et on scrute 4 E/S
C7A1-68         PLA
C7A2-A8         TAY
C7A3-68         PLA             on récupère A,X et Y
C7A4-AA         TAX
C7A5-A5 1B      LDA $1B
C7A7-60         RTS
```

ENVOIE UNE SERIE DE CODES SUR UN CANAL

Principe:On envoie en fait les codes un par un par une boucle sur le canal.
Etant sur la banque 7, pour lire les données sur la banque d'appel,on stocke l'adresse en $15-16 et on utilise $0411.
En entrée, la série de codes, terminée par un 0, est à l'adresse AY.

```
C7A8-A2 00      LDX #$00        on indexe le canal 0
C7AA            BYT $2C         et on saute l'instruction suivante

C7AB-A2 04      LDX #$04        on indexe le canal 1
C7AD            BYT $2C         et on saute l'instruction suivante

C7AE-A2 08      LDX #$08        on indexe le canal 2
C7B0            BYT $2C         et on saute l'instruction suivante

C7B1-A2 0C      LDX #$0C        on indexe le canal 3

C7B3-86 1C      STX $1C         on sauve le canal
C7B5-85 15      STA $15         et l'adresse de la chaine
```

```
:7B7-84 16      STY $16
:7B9-A5 1C      LDA $1C         <-------------------------------------------
:7BB-85 19      STA $19         on positionne le canal                     I
:7BD-A0 00      LDY #$00        Y=0 pour adressage indirect                I
:7BF-20 11 04   JSR $0411       et on lit un octet                         I
:7C2-F0 E3      BEQ $C7A7       est-ce 0 ? oui on sort.                    I
:7C4-20 72 C7   JSR $C772       non, on l'affiche directement              I
:7C7-E6 15      INC $15         on incrémente l'adresse                    I
:7C9-D0 EE      BNE $C7B9       et on boucle ------------------------------0
:7CB-E6 16      INC $16                                                    I
:7CD-D0 EA      BNE $C7B9       inconditionnel ----------------------------
```

LIRE UN CODE SUR UN CANAL

Principe:on indexe le canal comme d'habitude et on positionne N à 0 et C à 0 avant l'appel à chaque routine d'E/S. Dès qu'une donnée est lue, on sort sans scruter les autres E/S. Ce qui fait qu'une E/S ouverte avant une autre est prioritaire.
Comme pour la routine d'écriture, les registres A,X et Y sont conservés
En sortie, C=0 si une donnée à été lue, C=1 sinon.

```
:7CF-A9 00      LDA #$00        on indexe canal 0
:7D1            BYT $2C         sauter l'instruction suivante

:7D2-A9 04      LDA #$04        on indexe canal 1
:7D4            BYT $2C         on saute ...

.7D5-A9 08      LDA #$08        canal 2
:7D7            BYT $2C         on saute...

:7D8 A9 0C      LDA #$0C        ou canal 3

:7DA-85 19      STA $19         on sauve le canal
:7DC-A9 04      LDA #$04        pour 4 E/S
:7DE-85 1A      STA $1A
:7E0-8A         TXA             on sauve les registres
:7E1-48         PHA
:7E2-98         TYA
:7E3-48         PHA
:7E4-A6 19      LDX $19
:7E6-BD AE 02   LDA $02AE,X     y a-t-il une E/S ouverte ?
:7E9-10 0E      BPL $C7F9       non, suivante -----------------------------
:7EB-C9 88      CMP #$88        oui, une entrée ?                          I
:7ED-B0 0A      BCS $C7F9       non, suivante -----------------------------0
:7EF-AA         TAX             on prend l'index                           I
:7F0-A0 40      LDY #$40        N=0 V=1 C=0                                I
:7F2-20 1C C8   JSR $C81C       on lit la donnée                           I
:7F5-85 1D      STA $1D         et on la sauve                             I
:7F7-90 06      BCC $C7FF       C=0, on sort si une donnée à été lue       I
:7F9-E6 19      INC $19              <-------------------------------------
:7FB-CE 1A      DEC $1A
:7FD-D0 E5      BNE $C7E4       on scrute les 4 E/S si besoin est
:7FF-68         PLA             on récupère X et Y
:800-A8         TAY
:801-68         PLA
:802-AA         TAX
:803-A5 1D      LDA $1D         et la donnée
:805-60         RTS             C=1 si aucune donnée n'est lue
```

ATTENDRE UN CODE SUR UN CANAL

Principe:on lit un code jusqu'à ce qu'un code soit véritablement lu...
Cette routine est baclée en deux points. D'une part le STA/LDA qui aurait avantageusement pu être remplacé par TAX/TXa puisque X et Y ne sont pas modifiés, et d'autre part le SEC final qui met C à 1 en sortie alors que le protocole observé jusqu'ici donnait C=0 pour une opération correctement exécutée...

```
C806-A9 00     LDA #$00        indexer canal 0
C808           BYT $2C         sauter instruction suivante

C809-A9 04     LDA #$04        indexer canal 1
C80B           BYT $2C         et passer...

C80C-A9 08     LDA #$08        indexer canal 2
C80E           BYT $2C         et passer

C80F-A9 0C     LDA #$0C        indexer canal 3

C811-85 1B     STA $1B           on sauve le canal
C813-A5 1B     LDA $1B       -->pourquoi pas TAX/TXA ? deux octets de perdus...
C815-20 DA C7  JSR $C7DA     I  on lit un code
C818-B0 F9     BCS $C813     ---si pas de code lu, on boucle
C81A-38        SEC               sinon, C=1, ça cloche !
C81B-60        RTS
```

ENVOYER UNE COMMANDE A UNE E/S

principe:Y contient de quoi influencer N,V et C et X contient le numéro de l'E/S à activer. Pour l'activation, on place l'adresse de l'E/S dans le vecteur $2F8-9 et on se branche dessus.
Pour lire une donnée: Y=%0100000, pour fermer l'E/S, Y=%10000001 et pour ouvrir l'E/S, Y=%10000000. Curieusement, suite à un manque d'optimisation flagrant, cette routine ne sert pas à la lecture d'une donnée, ce qui fait perdre à la routine $C75D et qui remet en question la présence du PHA/PLA.

```
C81C-84 17     STY $17         on sauve la commande
C81E-84 18     STY $18
C820-48        PHA             et la donnée (pourquoi ???)
C821-8A        TXA
C822-0A        ASL             on multiplie le numéro de l'E/S par deux
C823-AA        TAX
C824-BD BE 02  LDA $02BE,X     et on prépare le saut à l'adresse
C827-8D F8 02  STA $02F8       de gestion de l'E/S
C82A-BD BF 02  LDA $02BF,X
C82D-8D F9 02  STA $02F9
C830-68        PLA             on récupère la donnée
C831-46 17     LSR $17         on positionne C
C833-24 18     BIT $18         N et V
C835-4C F7 02  JMP $02F7       et on exécute la routine de l'E/S
```

TABLE DES ROUTINES D'E/S

Remarque:Les vecteurs sans valeurs pointent sur SEC/RTS, donc l'opération s'est mal déroulée. Ce qui est clair.

```
C838    WRD $D95C          routine KBD gestion du clavier
C83A    WRD $C81A          rien
C83C    WRD $DAF7          routine MDE entrée Minitel
C83E    WRD $DB5D          routine RSE entrée RS232
C840    WRD $C81A          rien (aurait du être MIE, entrée MIDI)
C842    WRD $C81A          rien
C844    WRD $C81A          rien (US 1 libre pour l'utilisateur)
C846    WRD $C81A          rien (US 2 libre pour l'utilisateur)
C848    WRD $DB86          routine SCR écran fenêtre 0
C84A    WRD $DB8C          routine SC1 fenêtre 1
C84C    WRD $DB92          routine SC2 fenêtre 2
C84E    WRD $DB98          routine SC3 fenêtre 3
C850    WRD $C81A          rien (aurait du être MIS, sortie MIDI)
C852    WRD $C81A          rien
C854    WRD $DA70          routine LPR sortie CENTRONICS parrallèle
C856    WRD $DB12          routine MDS sortie Minitel
C858    WRD $DB79          routine RSS sortie RS232
C85A    WRD $D5C6          routine VDT sortie HIRES émulation Videotex
C85C    WRD $C81A          rien
C85E    WRD $C81A          rien
C860    WRD $C81A          rien (US 4 libre pour l'utilisateur)
C862    WRD $C81A          rien (US 5 libre pour l'utilisateur)
C864    WRD $C81A          rien (US 6 libre pour l'utilisateur)
C866    WRD $C81A          rien (US 7 libre pour l'utilisateur)
```

TRAITEMENT DU BRK

Principe:Trivial mais o combien génial ! Seul un farfelu comme BROCHE pouvait y penser. L'appel à une routine du moniteur se fait non pas par des JMP mais par le BRK qui pour l'occasion devient une instruction page 0. Voici le principe:

- Après un BRK, donc à l'entrée de la routine, la pile contient dans l'ordre d'entrée PC+2 (PC est l'adresse exacte du BRK) et P.
- On dépile P que l'on sauve car on en aura besoin pour le RTI
- On enlève 1 à l'adresse de retour de l'IRQ, donc il y a dans la pile PC+1. C'est en PC+1 que le code de la routine se trouve ! on le lit.
- On empile l'adresse de retour du BRK décrémentée de 1.
- On empile l'adresse exacte de la routine.
- Et on empile P.

La routine est prête à être exécutée :

- RTI : on dépile P et l'adresse de la routine que l'on exécute.
- RTS : fin de la routine, on dépile $402 donc on se branche en $403.
- RTS : fin de la $403, on dépile l'adresse de retour de base que l'on a déja décrementé de 1 (par chance, le BRK sauve l'adresse d'appel+2 car à la base, il remplaçait des codes sur 2 octets le plus souvent). Le RTS incérementé et se branche, le programme se poursuit naturellement !!!

```
C868-86 22      STX $22         on sauve X
C86A-84 23      STY $23         Y
C86C-68         PLA
C86D-85 24      STA $24         et P dans IRQSVX, IRQSVY et $24
C86F-29 10      AND #$10        B=1 ? (est-ce un BRK)
C871-F0 40      BEQ $C8B3       non, on passe aux IRQ système ----------------
C873-BA         TSX             on prend                                      I
C874-68         PLA             on décremente l'adresse de retour             I
C875-D0 03      BNE $C87A       car RTS incrémente après dépilement           I
C877-DE 02 01   DEC $0102,X                                                   I
C87A-38         SEC                                                           I
C87B-E9 01      SBC #$01                                                      I
C87D-48         PHA                                                           I
C87E-85 15      STA $15         on sauve l'adresse de retour-1                I
C880-BD 02 01   LDA $0102,X     dans $15-16 pour lire le code suivant le BRK  I
C883-85 16      STA $16                                                       I
C885-AD 0F 04   LDA $040F       on place la banque d'appel comme banque de BRK I
C888-8D 10 04   STA $0410                                                     I
C88B-A0 00      LDY #$00                                                      I
C88D-20 11 04   JSR $0411       on lit le numéro de la routine appelée        I
C890-0A         ASL             *2 car une adresse a deux octets              I
C891-AA         TAX                                                           I
C892-A9 04      LDA #$04        on met dans la pile $403-1                    I
C894-48         PHA             car on va s'y brancher par RTS                I
C895-A9 02      LDA #$02        pourquoi pas LSR ?                            I
C897-48         PHA                                                           I
C898-BD A5 CA   LDA $CAA5,X     on lit l'adresse de la routine                I
C89B-BC A4 CA   LDY $CAA4,X                                                   I
C89E-90 06      BCC $C8A6       si C=1                                        I
C8A0-BD A5 CB   LDA $CBA5,X     on lit dans la deuxième table car             I
C8A3-BC A4 CB   LDY $CBA4,X     le code était > 127                           I
C8A6-48         PHA             on empile l'adresse exacte                    I
C8A7-98         TYA             car on appelle par RTI                        I
C8A8-48         PHA             on empile P (RTI oblige)                      I
C8A9-A5 24      LDA $24                                                       I
C8AB-48         PHA                                                           I
C8AC-A5 21      LDA $21         on prend A,X et Y                             I
C8AE-A4 23      LDY $23                                                       I
C8B0-A6 22      LDX $22                                                       I
C8B2-40         RTI             et on exécute la routine                      I
                                                                              I
                                                                              I
                GESTION NORMALE DES IRQ                                       I
                                                                              I
C8B3-A5 24      LDA $24         on replace P <--------------------------------
C8B5-48         PHA
C8B6-38         SEC
C8B7-66 1E      ROR $1E         on force b7($1E) à 1 pour indiquer premier passage
C8B9-20 BF C8   JSR $C8BF       on gère la RS232 (entrée et sortie)
C8BC-4C B9 C9   JMP $C9B9       et on poursuit le traitement des IRQ
```

GERE LA RS232

Principe: A chaque IRQ, on va tester si la ligne est libre et au cas échéant lire ou écrire une donnée. Si la ligne était libre, on force b7($1E) à 0 afin d'indiquer qu'une donnée est en cours de lecture ou d'écriture. Les octets $20 et $3F servent au décompte des bits lors de la

transmission. De plus, avant de latcher une donnée, on teste si on l'envoie sur une imprimante ou non. Y sort indemne de la routine.

```
C8BF-98        TYA            on sauve Y
C8C0-48        PHA
C8C1-AD 1D 03  LDA $031D      on prend l'état de la RS232
C8C4-10 55     BPL $C91B      occupée ?        on saute ----------------------
C8C6-46 1E     LSR $1E        oui, on annule b7($1E)                            I
C8C8-48        PHA            on sauve ACIASR                                   I
C8C9-29 08     AND #$08       b4=1 ?                                            I
C8CB-F0 12     BEQ $C8DF      non ----------------------------------------------I
C8CD-AE 1C 03  LDX $031C      oui, on lit la donnée                          I  I
C8D0-68        PLA            et l'état                                      I  I
C8D1-48        PHA                                                           I  I
C8D2-29 07     AND #$07       dont on isole b2b1b0                           I  I
C8D4-F0 03     BEQ $C8D9      tous à 0 ? oui ----------------                I  I
C8D6-09 B0     ORA #$B0       non %1011000 on force les bits I               I  I
C8D8           BYT $24        et on saute la suivante        I               I  I
C8D8-8A        TXA            donnée dans A <----------------                I  I
C8DA-A2 0C     LDX #$0C       indexe buffer ACIA entrée                      I  I
C8DC-20 1D C5  JSR $C51D      et on inscrit la donnée                        I  I
C8DF-68        PLA            on sort l'état sauvegardé <--------------------  I
C8E0-29 10     AND #$10       libre en écriture ?                               I
C8E2-F0 37     BEQ $C91B      non ----------------------------------------------0
C8E4-A2 18     LDX #$18       oui, on lit le buffer ACIA sortie                 I
C8E6-20 0F C5  JSR $C50F      y-a-t-il une donnée ?                             I
C8E9-B0 16     BCS $C901      non ------------------------------------          I
C8EB-AD 1D 03  LDA $031D      oui, on lit l'état                 I              I
C8EE-29 20     AND #$20       écriture possible ?                I              I
C8F0-D0 29     BNE $C91B      non, on sort ----------------------+--------------0
C8F2-20 18 C5  JSR $C518      oui, on lit une donnée ds le buffer I             I
C8F5-8D 1C 03  STA $031C      on la dépose sur le registre de données I         I
C8F8-AD 1F 03  LDA $031F      on envoie la                       I              I
C8FB-29 07     AND #$07       pour 7 bits                        I              I
C8FD-85 3F     STA $3F        dans $3F                           I              I
C8FF-90 1A     BCC $C91B      inconditionnel                     I              I
C901-E6 20     INC $20        si $20 non nul <-------------------               I
C903-D0 16     BNE $C91B      on sort ------------------------------------------0
C905-C6 3F     DEC $3F        $3F non nul, on sort                              I
C907-D0 12     BNE $C91B      ------------------------------------------------0
C909-AD 8A 02  LDA $028A      si imprimante série, C=1                          I
C90C-4A        LSR                                                              I
C90D-4A        LSR                                                              I
C90E-4A        LSR                                                              I
C90F-AD 1E 03  LDA $031E      on force b4b3 de ACIA Command Register            I
C912-29 F3     AND #$F3                                                         I
C914-90 02     BCC $C918      si imprimante centronics, on passe ------------   I
C916-29 FE     AND #$FE       si RS232, on force b1 à 0                      I  I
C918-8D 1E 03  STA $031E      on place la commande <-------------------------   I
C91B-68        PLA            <-------------------------------------------------
C91C-A8        TAY            on récupère Y et on sort
C91D-60        RTS
```

GESTION TIMERS, IMPRIMANTE ET CURSEUR

Principe:toutes les 0,1 secondes (en fait 4 fois toutes les 25000 us décomptées sur T1), l'horloge utilisateur et temps réel sont mises à jour. Si on

passe une seconde; l'heure est affiché si nécessaire. L'imprimante se voit éventuellement envoyer une donnée si elle est libre.
4 fois par secondes (ou 10 toutes les 25000 us décomptées), on inverse l'état du curseur dans la fenêtre courante selon les paramètres de la fenêtre 0. Tout va bien si on ne travaille qu'avec la fenêtre 0, mais ça bugge dès qu'on jongle avec les fenêtres... pas très sérieux

```
C91E-CE 15 02  DEC $0215      faut-il gérer horloges et imprimante ?
C921-D0 50     BNE $C973      non ----------------------------------------------
C923-A9 04     LDA #$04       on replace le comoteur pour 1/10' de seconde     I
C925-8D 15 02  STA $0215                                                        I
C928-2C 8A 02  BIT $028A      imprimante prête ?                                I
C92B-10 03     BPL $C930                                                        I
C92D-20 2F CA  JSR $CA2F      oui, on gère l'imprimante                         I
C930-A5 44     LDA $44        on décomote les dixièmes utilisateurs             I
C932-D0 02     BNE $C936      dans 16 bits de TIMEUD                            I
C934-C6 45     DEC $45                                                          I
C936-C6 44     DEC $44                                                          I
C938-38        SEC                                                              I
C939-EE 10 02  INC $0210      on compte 0,1 seconde dans l'horloge temps réel   I
C93C-AD 10 02  LDA $0210                                                        I
C93F-E9 0A     SBC #$0A                                                         I
C941-90 30     BCC $C973      a-t-on 10 dixièmes ?                              I
C943-8D 10 02  STA $0210      oui, on met 0                                     I
C946-2C 14 02  BIT $0214      faut-il afficher l'horloge                        I
C949-10 03     BPL $C94E                                                        I
C94B-20 75 CA  JSR $CA75      oui, so do we !                                   I
C94E-EE 11 02  INC $0211      on ajuste les secondes                            I
C951-A5 42     LDA $42        et le timer utilisateur en secondes               I
C953-D0 02     BNE $C957      (pourquoi ?? rien ne dit qu'il concorde           I
C955-C6 43     DEC $43        avec l'horloge !!!)                               I
C957-C6 42     DEC $42        donc TIMEUD et TIMEUS sont désynchronisés         I
C959-AD 11 02  LDA $0211      60 secondes ?                                     I
C95C-E9 3C     SBC #$3C                                                         I
C95E-90 13     BCC $C973                                                        I
C960-8D 11 02  STA $0211      oui, on ajoute une minute                         I
C963-EE 12 02  INC $0212                                                        I
C966-AD 12 02  LDA $0212                                                        I
C969-E9 3C     SBC #$3C       60 minutes ?                                      I
C96B-90 06     BCC $C973                                                        I
C96D-8D 12 02  STA $0212                                                        I
C970-EE 13 02  INC $0213      oui on ajoute une heure                           I
C973-CE 16 02  DEC $0216      faut-il gérer le curseur <-------------------------
C976-D0 19     BNE $C991
C978-A9 0A     LDA #$0A       oui, on remet le compteur de quart de secondes
C97A-8D 16 02  STA $0216
C97D-AD 17 02  LDA $0217      on inverse l'état du curseur
C980-49 80     EOR #$80
C982-8D 17 02  STA $0217
C985-2C 48 02  BIT $0248      test pour la fenêtre 0
C988-10 07     BPL $C991      si curseur absent, on sort
C98A-70 05     BVS $C991      si curseur fixe aussi
C98C-A6 28     LDX $28        on gère le curseur ds la fenêtre courante
C98E-4C 2D DE  JMP $DE2D      (et si ce n'est pas la fenêtre 0 ?!)
C991-60        RTS
```

GESTION DES IRQ T1 et T2

```
Principe: Si c'est T2 qui a déclenché l'IRQ, on teste la souris.
          Si c'est T1, on teste le clavier et le joystick gauche et la souris.
          Tant que b7($1E) est nul, on poursuit la transmission série.
          A noter que la gestion du joystick droit n'est que fumisterie
          puisqu'elle pointe sur un RTS.


C992-AD 0D 03  LDA $030D      IRQ par T2 ?
C995-29 20     AND #$20
C997-F0 20     BEQ $C9B9      non, on passe sur T1
C999-AD 8F 02  LDA $028F      oui, on replace la valeur de
C99C-AC 90 02  LDY $0290      base (10000) dans T2
C99F-8D 08 03  STA $0308
C9A2-8C 09 03  STY $0309
C9A5-AD 8C 02  LDA $028C      souris branchée ?
C9A8-4A        LSR
C9A9-90 06     BCC $C9B1    ---non
C9AB-20 85 E0  JSR $E085    I  oui, on gère la souris
C9AE-4C B9 C8  JMP $C8B9    I  et on poursuit les IRQ
C9B1-A9 FF     LDA #$FF     I->on place le compteur largement pour
C9B3-8D 09 03  STA $0309       ne pas être dérangé par une souris absente...
C9B6-4C B9 C8  JMP $C8B9       et on poursuit (BNE, non ?)

C9B9-2C 0D 03  BIT $030D      IRQ traitée ?
C9BC-30 0E     BMI $C9CC      non ----------------------------------------------
C9BE-24 1E     BIT $1E        oui, transmission série en cours ?               I
C9C0-10 07     BPL $C9C9    ---oui                                             I
C9C2-A6 22     LDX $22      I  non, on termine                                 I
C9C4-A4 23     LDY $23      I  normalement les IRQ                             I
C9C6-4C 00 04  JMP $0400    I                                                  I
C9C9-4C B6 C8  JMP $C8B6    I->on poursuit la transmission série               I
C9CC-46 1E     LSR $1E        on indique transmission en cours <---------------
C9CE-2C 0D 03  BIT $030D      IRQ par T1 ?
C9D1-50 4C     BVC $CA1F      non
C9D3-2C 04 03  BIT $0304      on annule T1L
C9D6-20 1E C9  JSR $C91E      on gère les timers, l'imprimante et le curseur
C9D9-CE A6 02  DEC $02A6      $2A6=0 ?
C9DC-D0 22     BNE $CA00      non, on passe
C9DE-20 DF D7  JSR $D7DF      oui, on gère le clavier
C9E1-20 BF C8  JSR $C8BF      et la transmission série
C9E4-2C 70 02  BIT $0270      b7 de $270 ?
C9E7-10 07     BPL $C9F0      0, on passe ----------------------------------------
C9E9-A9 14     LDA #$14       on met 20 en $2A7                                 I
C9EB-8D A7 02  STA $02A7                                                        I
C9EE-D0 0B     BNE $C9FB    ---et on saute                                     I
C9F0-AD A8 02  LDA $02A8    I  on prend $2A8 <-----------------------------------
C9F3-2C A7 02  BIT $02A7    I  b7 de $2A7=1 ?
C9F6-30 05     BMI $C9FD    I  oui, on met $2A8 dans $2A6
C9F8-CE A7 02  DEC $02A7    I  non
C9FB-A9 01     LDA #$01     -->on met 1
C9FD-8D A6 02  STA $02A6       dans $2A6
CA00-2C 8C 02  BIT $028C       joystick droit connecté ?
CA03-10 06     BPL $CA0B
CA05-20 FA DF  JSR $DFFA       on gère le joy droit (enfin, on devrait -> RTS)
CA08-2C 8C 02  BIT $028C       joystick gauche ?
CA0B-50 03     BVC $CA10
CA0D-20 FB DF  JSR $DFFB       oui, on gère le port gauche
CA10-AD 8C 02  LDA $028C       souris connectée ?
CA13-4A        LSR
CA14-90 03     BCC $CA19    ---non
```

```
CA16-20 E1 E0   JSR $E0E1    I  oui, on gère la souris (double gestion donc)
CA19-4C B9 C8   JMP $C8B9    -->on poursuit la transmission série
CA1C-4C 92 C9   JMP $C992       on gère les IRQ par T2 <---
CA1F-AD 0D 03   LDA $030D       l'IRQ ne vient pas de T1  I
CA22-29 02      AND #$02        ACK imprimante ?          I
CA24-F0 F6      BEQ $CA1C       non ----------------------
CA26-2C 01 03   BIT $0301       oui, on annule le ACK
CA29-20 2F CA   JSR $CA2F       on gère l'imprimante
CA2C-4C B9 C8   JMP $C8B9       et on poursuit la transmission série
```

GERE L'IMPRIMANTE

Principe:Lorsqu'un ACK est détecté, ou que l'imprimante est prête, lors d'une IRQ, on teste si une donnée est dans le buffer. Dans ce cas, on l'envoie et on indique imprimante occupée. Sinon on indique imprimante libre.

```
CA2F-A2 24      LDX #$24        indexe buffer imprimante
CA31-20 18 C5   JSR $C518       on lit une donnée
CA34-90 08      BCC $CA3E       buffer vide ? non ------------------------------
CA36-0E 8A 02   ASL $028A       oui                                           I
CA39-38         SEC             on indique imprimante libre                   I
CA3A-6E 8A 02   ROR $028A                                                     I
CA3D-60         RTS             et on sort                                    I
CA3E-8D 01 03   STA $0301       on met la donnée sur le port A <---------------
CA41-AD 00 03   LDA $0300       et on génère un STROBE
CA44-29 EF      AND #$EF        en mettant succesivement à 0
CA46-8D 00 03   STA $0300
CA49-09 10      ORA #$10        et à 1 la broche PB4
CA4B-8D 00 03   STA $0300
CA4E-0E 8A 02   ASL $028A       et on indique imprimante occupée
CA51-4E 8A 02   LSR $028A
CA54-60         RTS
```

REMET L'HORLOGE A 0

Principe:on met dixièmes, secondes, minutes et heures à 0 et on indique au compteur de secondes système ($215) qu'il faut ajuster l'horloge dès la prochaine IRQ.

```
CA55-A9 00      LDA #$00        on met 0
CA57-A2 04      LDX #$04
CA59-9D 10 02   STA $0210,X     dans TIMEH,TIMEM,TIMES et TIMED
CA5C-CA         DEX
CA5D-10 FA      BPL $CA59
CA5F-A9 01      LDA #$01        puis on force l'ajustement de l'horloge
CA61-8D 15 02   STA $0215       à la prochaine IRQ
CA64-60         RTS
```

STOPPE L'AFFICHAGE DE L'HORLOGE

```
CA65-4E 14 02   LSR $0214       on indique pas d'affiche dans FLGCLK
```

```
:A68-60         RTS
```

DEMANDE L'AFFICHAGE REGULIER DE L'HORLOGE

```
:A69-08         PHP             on interdit les interruptions
:A6A-78         SEI             pour ne pas qu'un affichage se fasse entre
:A6B-85 40      STA $40         CA6B et CA6D.
:A6D-84 41      STY $41         on stocke l'adresse d'affichage dans ADCLK
:A6F-38         SEC
:A70-6E 14 02   ROR $0214       et on indique qu'il faut afficher l'horloge toutes
:A73-28         PLP             les secondes
:A74-60         RTS
```

AFFICHE L'HORLOGE

```
ction:Affiche l'horloge selon ADCLK sous la forme HH:MM:SS. Aucun test n'est
      fait sur ADCLK ce qui permet d'envoyer l'horloge n'importe ou en RAM.

:A75-A0 00      LDY #$00        on positionne l'affiche en ADCLK
:A77-AD 13 02   LDA $0213       on lit l'heure
:A7A-20 90 CA   JSR $CA90       et on affiche deux chiffres
:A7D-A9 3A      LDA #$3A        on affiche ":"
:A7F-91 40      STA ($40),Y
:A81-C8         INY
:A82-AD 12 02   LDA $0212       on lit les minutes
:A85-20 90 CA   JSR $CA90       que l'on affiche
:A88-A9 3A      LDA #$3A        suivies de ":"
:A8A-91 40      STA ($40),Y
:A8C-C8         INY
:A8D-AD 11 02   LDA $0211       et on affiche les secondes
```

AFFICHER A EN DECIMAL SUR DEUX CHIFFRES

```
rincipe:on retire 10 à A jusqu'a ce que A devienne négatif, alors on affiche
        A/10, puis on affiche le reste soit A MOD 10. Cette routine, courte,
        est un bon exemple d'optimisation : Pas de CLC ou SEC en trop, et une
        habile gestion des registres et des valeurs.

:A90-A2 2F      LDX #$2F        X contient "0"-1
:A92-38         SEC
:A93-E9 0A      SBC #$0A        on soustrait 10 à A
:A95-E8         INX             et on ajoute 1 à X
:A96-B0 FB      BCS $CA93       on a dépassé ?
:A98-48         PHA             oui, on sauve le reste
:A99-8A         TXA             et on affiche X, le premier chiffre
:A9A-91 40      STA ($40),Y
:A9C-68         PLA             on reprend le reste
:A9D-C8         INY             on indexe la position suivante
:A9E-69 3A      ADC #$3A        on ajoute "0"+10 au reste
:AA0-91 40      STA ($40),Y     et on l'affiche
:AA2-C8         INY             puis on indexe la prochaine position
:AA3-60         RTS
```

TABLE DES VECTEURS BRK

Principe: Tous les vecteurs BRK, appelés X..... sauf pour ZADCHA (sans doute une faute de frappe de BROCHE...), sont rangés ici selon leur numéro. L'adresse codée est ici l'adresse exacte puisqu'elle sera appélée par un RTI. Parmi les vecteurs non répertoriés, certains pointent sur 0, un RTS aurait été mieux, d'autre sur des adresses bidons et d'autres encore sur des routines intéressantes.

CAA4	WRD $C6E5	00	XOP0	ouverture d'une E/S sur le canal 0
CAA6	WRD $C6E8	01	XOP1	ouverture d'une E/S sur le canal 1
CAA8	WRD $C6EB	02	XOP2	...canal 2
CAAA	WRD $C6EE	03	XOP3	...canal 3
CAAC	WRD $C720	04	XCL0	fermeture d'une E/S sur le canal 0
CAAE	WRD $C723	05	XCL1	...canal 1
CAB0	WRD $C726	06	XCL2	...canal 2
CAB2	WRD $C727	07	XCL3	...canal 3
CAB4	WRD $C7CF	08	XRD0	lecture d'un code sur le canal 0
CAB6	WRD $C7D2	09	XRD1	...canal 1
CAB8	WRD $C7D5	0A	XRD2	...canal 2
CABA	WRD $C7B8	0B	XRD3	...canal 3
CABC	WRD $C806	0C	XRDW0	attente d'un code sur le canal 0
CABE	WRD $C809	0D	XRDW1	...canal 1
CAC0	WRD $C80C	0E	XRDW2	...canal 2
CAC2	WRD $C80F	0F	XRDW3	...canal 3
CAC4	WRD $C75D	10	XWR0	écriture d'un code sur le canal 0
CAC6	WRD $C762	11	XWR1	...canal 1
CAC8	WRD $C767	12	XWR2	...canal 2
CACA	WRD $C76C	13	XWR3	...canal 3
CACC	WRD $C7A8	14	XWSTR0	écriture d'une chaine sur le canal 0
CACE	WRD $C7AB	15	XWSTR1	...canal 1
CAD0	WRD $C7AE	16	XWSTR2	...canal 2
CAD2	WRD $C7B1	17	XWSTR3	...canal 3
CAD4	WRD $CD6C	18	XDECAL	déplacement d'un bloc mémoire
CAD6	WRD $CF75	19	XTEXT	passage en mode TEXT
CAD8	WRD $CF45	1A	XHIRES	passage en mode HIRES
CADA	WRD $CF06	1B	XEFFHI	effacer la page HIRES
CADC	WRD $CF14	1C	XFILLM	remplir un bloc mémoire
CADE	WRD $FF31	1D	ZADCHA	trouver l'adresse d'un code ASCII
CAE0	WRD $C6BF	1E	XTSTLP	tester l'imprimante
CAE2	WRD $D0F0	1F	XMINMA	passe A en majuscules s'il y a lieu
CAE4	WRD $CE69	20	XMUL40	multiplication par 40
CAE6	WRD $CE97	21	XMULT	multiplication
CAE8	WRD $CE89	22	XADRES	addition
CAEA	WRD $CEDC	23	XDIVIS	division
CAEC	WRD $CFF0	24	XNOMFI	envoie un nom dans BUFNOM
CAEE	WRD $C756	25	XCRLF	saute une ligne sur le canal 0
CAF0	WRD $E749	26	XDECAY	passage décimal-ASCII -> binaire AY
CAF2	WRD $0000	27	rien	(pourquoi pas CAA3 par exemple ?)
CAF4	WRD $CDEF	28	XBINDX	conversion décimal-ASCII
CAF6	WRD $CE39	29	XDECIM	conversion avec afichage sur le canal 0
CAF8	WRD $CE54	2A	XHEXA	conversion hexa
CAFA	WRD $F49B	2B	XA1AFF	affiche ACC1 sur le canal 0
CAFC	WRD $CBE0	2C	XMENU	gestion de menu déroulant
CAFE	WRD $E435	2D	XEDT	édition pleine page sur le canal 0
CB00	WRD $E6EB	2E	XINSER	insertion de ligne dans un listing

```
CB02        WRD $E680     2F XSCELG recherge de ligne dans un listing
CB04        WRD $D140     30        gère le curseur HIRES émulation VIDEOTEX
CB06        WRD $D711     31        affiche un motif mosaique en HIRES VDT
CB08        WRD $E537     32 XEDTIN (que fait-elle ?)
CB0A        WRD $E66C     33 XECRPR écrit le prompt
CB0C        WRD $DE1E     34 XCOSCR efface le curseur dans une fenêtre
CB0E        WRD $DE20     35 XCSSCR affiche le curseur dans une fenêtre
CB10        WRD $DEFB     36 XSCRSE initialise une fenêtre
CB12        WRD $DE54     37 XSCROH scrolle une fenêtre vers le haut
CB14        WRD $DE5C     38 XSCROB id. mais vers le bas
CB16        WRD $FEF7     39 XSCRNE redéfinisseur de caractères
CB18        WRD $D442     3A        gère les séquences VIDEOTEX
CB1A        WRD $D4F1     3B        calcule une séquence VIDEOTEX
CB1C        WRD $CA55     3C        remet l'horloge à 0
CB1E        WRD $CA65     3D XCLCL  stoppe l'affichage de l'horloge
CB20        WRD $CA69     3E XWRCLK force l'affichage de l'horloge
CB22        WRD $0000     3F rien
CB24        WRD $D9E9     40 XSONPS envoie 14 paramètres au PSG 8912
CB26        WRD $DA1A     41 XEPSG  envoie une valeur au PSG 8912
CB28        WRD $DDD8     42 XOUPS  émet un OUPS
CB2A        WRD $EB0D     43 XPLAY  commande PLAY de l'HYPER-BASIC
CB2C        WRD $EB73     44 XSOUND commande SOUND de l'HYPER-BASIC
CB2E        WRD $EB5A     45 XMUSIC commande MUSIC de l'HYPER-BASIC
CB30        WRD $EBEC     46 XZAP   émet un ZAP
CB32        WRD $EBDF     47 XSHOOT émet un SHOOT
CB34        WRD $DA72     48 XLPRBI envoie un code à l'imprimante
CB36        WRD $DAE4     49 XLPCRL saute une ligne sur l'imprimante
CB38        WRD $E189     4A XHCSCR hard-copy TEXT
CB3A        WRD $E209     4B XHCVDT hard-copy HIRES émulation VIDEOTEX
CB3C        WRD $E250     4C XHCHRS hard-copy HIRES
CB3E        WRD $0000     4D rien
CB40        WRD $0000     4E rien
CB42        WRD $0000     4F rien
CB44        WRD $D903     50 XALLKB scrute le clavier dans KBDCOL (8 col.)
CB46        WRD $D81F     51 XKBDAS convertit en ASCII KBDCOL ds le buffer
CB48        WRD $FF4C     52 XGOKBD change le type de clavier
CB4A        WRD $0000     53 rien
CB4C        WRD $C51D     54 XECRBU écrit une donnée dans le un buffer
CB4E        WRD $C518     55 XLISBU lit une donnée dans une buffer
CB50        WRD $C50F     56 XTSTBU teste si un buffer est vide
CB52        WRD $C50C     57 XVIDBU vide un buffer
CB54        WRD $C507     58 XINIBU initialise un buffer
CB56        WRD $C4EA     59 XDEFBU donne les valeurs par défaut à un buffer
CB58        WRD $CFB1     5A XBUSY  teste si un des buffer est non vide
CB5A        WRD $0000     5B rien
CB5C        WRD $ED9A     5C XSDUMP dumpe l'entrée RS232
CB5D        WRD $ED77     5D XCONSO commande console de l'HYPER-BASIC
CB60        WRD $EDE5     5E XSLOAD charge un fichier via la RS232
CB62        WRD $EDCA     5F XSSAVE sauve un fichier via la RS232
CB64        WRD $EDFC     60 XMLOAD charge un fichier via le Minitel
CB66        WRD $EDD7     61 XMSAVE sauve un fichier via le Minitel
CB68        WRD $EEA5     62 XRING  commande RING de l'HYPER-BASIC
CB6A        WRD $EF4A     63 XWCXFI commande WCXFIN de l'HYPER-BASIC
CB6C        WRD $EF20     64 XLIGNE prise de ligne
CB6E        WRD $EF3F     65 XDECON commande UNCONNECT de l'HYPER-BASIC
CB70        WRD $EF7A     66 XMOUT  envoie un code au Minitel
CB72        WRD $EF85     67 XSOUT  envoie un code à la RS232
CB74        WRD $F4A5     68 XA1DEC convertit ACC1 en décimal-ASCII
CB76        WRD $F91E     69 XDECA1 convertit une chaine déciamle dans ACC1
CB78        WRD $EF82     6A XA1PA2 somme
```

```
CB7A        WRD $EF3B     6B XA2NA1 soustraction
CB7C        WRD $F18B     6C XA1MA2 multiplication
CB7E        WRD $F28A     6D XA2DA1 division
CB80        WRD $F61A     6E XA1EA2 exponentiation
CB82        WRD $F653     6F XNA1   négation
CB84        WRD $F78E     70 XSIN   sinus
CB86        WRD $F781     71 XCOS   cosinus
CB88        WRD $F80A     72 XTAN   tangente
CB8A        WRD $F835     73 XATAN  arctangente
CB8C        WRD $F68C     74 XEXP   exponentielle
CB8E        WRD $F146     75 XLN    logarithme népérien
CB90        WRD $F26F     76 XLOG   logarithme décimal
CB92        WRD $F735     77 XRND   renvoie un nombre aléatoire dans ACC1
CB94        WRD $F610     78 XSQR   racine carrée
CB96        WRD $F8B6     79 XRAD   conversion degré -> radian
CB98        WRD $F8AA     7A XDEG   conversion radian -> degré
CB9A        WRD $F46A     7B XINT   calcul de INT(ACC1)
CB9C        WRD $F314     7C XPI    place la valeur PI dans ACC1
CB9E        WRD $F771     7D XRAND  ACC1=RAND(ACC1)
CBA0        WRD $F387     7E XA1A2  met ACC1 dans ACC2
CBA2        WRD $F377     7F XA2A1  met ACC2 dans ACC1
CBA4        WRD $F3ED     80 XIYAA1 YA (et non AY !) dans ACC1
CBA6        WRD $F323     81 XAYA1  (AY) dans ACC1
CBA8        WRD $F3A6     82 XA1IAY INT(ACC1) dans AY
CBAA        WRD $F352     83 XA1XY  ACC1 en (XY)
CBAC        WRD $F396     84 XAA1   arrondit ACC1 selon ACC1EX
CBAE        WRD $F8CD     85 XADNXT (AY)+ACC1 -> (AY)
CBB0        WRD $FA12     86 XINTEG teste si ACC1 est un entier
CBB2        WRD $0000     87 rien
CBB4        WRD $E7E7     88        déplace le curseur hires vers la gauche
CBB6        WRD $E7D9     89        id. vers la droite
CBB8        WRD $E7C1     8A        id. vers le bas
CBBA        WRD $E7CD     8B        id. vers le haut
CBBC        WRD $E792     8C XHRSSE place le curseur en X,Y
CBBE        WRD $E866     8D XDRAWA commande ADRAW de l'HYPER-BASIC
CBC0        WRD $E885     8E XDRAWR commande DRAW de l'HYPER-BASIC
CBC2        WRD $E9CB     8F XCIRCL commande CIRCLE de l'HYPER-BASIC
CBC4        WRD $E92F     90 XCURSE commande CURSET de l'HYPER-BASIC
CBC6        WRD $E93C     91 XCURMD commande CURMOV de l'HYPER-BASIC
CBC8        WRD $E95D     92 XPAPER commande PAPER de l'HYPER-BASIC
CBCA        WRD $E95F     93 XINK   commande INK de l'HYPER-BASIC
CBCC        WRD $E819     94 XBOX   commande BOX de l'HYPER-BASIC
CBCE        WRD $E82C     95 XABOX  commande ABOX de l'HYPER-BASIC
CBD0        WRD $EA73     96 XFILL  commande FILL de l'HYPER-BASIC
CBD2        WRD $EAAF     97 XCHAR  commande CHAR de l'HYPER-BASIC
CBD4        WRD $EA93     98 XSCHAR commande SCHAR de l'HYPER-BASIC
CBD6        WRD $0000     99 rien
CBD8        WRD $0000     9A rien
CBDA        WRD $0000     9B rien
CBDC        WRD $EBE5     9C XEXPLO émet un EXPLODE
CBDE        WRD $EBD9     9D XPING  émet un PING
```

GESTION D'UN MENU DEROULANT

Action:Les choix du menu doivent se trouver à l'adresse ADMEN, séparés par un 0. Le dernier choix est terminé par un double 0. MENDDY et X contiennent respectivement l'ordonnée de la première et de la dernière ligne de la fenêtre menu dans l'écran.MENDDX contient l'abscisse de la fenêtre

dans l'écran. MENLX contient la largeur de la barre en vidéo inverse.
A contient le premier choix à afficher (0 pour le premier) et Y le numéro du choix à afficher (id.)
En sortie, X contient le numéro du choix (id.) et A le mode de sortie:
13 pour RETURN
27 pour ESC
32 pour barre d'espace
Le CTRL-C n'est pas géré.

```
BE0-84 60      STY $60        on sauve les données
BE2-85 66      STA $66        passées par registre
BE4-86 63      STX $63
BE6-A2 00      LDX #$00
BE8-20 1E DE   JSR $DE1E      on éteint le curseur dans la fenêtre 0
BEB-A4 62      LDY $62        Y = première ligne dans la fenêtre
BED-A6 66      LDX $66        X = premier choix à afficher
BEF            BYT $2C      ----et on saute
BF0-E8         INY          I  on passe à la ligne et <----------------------------
BF1-C8         INX          I  au choix suivant                                    I
BF2-20 F9 CC   JSR $CCF9    -->on affiche le choix                                 I
BF5-30 04      BMI $CBFB    ----si dernier choix on passe                          I
BF7-C4 63      CPY $63      I  sinon, fin de la fenêtre ?                          I
BF9-D0 F5      BNE $CBF0    I  non ---------------------------------------------------
BFB-86 67      STX $67      I->on stocke le numéro du dernier choix
BFD-A6 60      LDX $60        X=choix ou placer la barre
BFF-38         SEC
C00-8A         TXA
C01-E5 66      SBC $66        on enleve le premier choix affiché
C03-18         CLC
C04-65 62      ADC $62        et on ajoute la première ligne de la fenêtre
C06-A8         TAY
C07-20 D3 CC   JSR $CCD3      on affiche la barre
C0A-20 06 C8   JSR $C806      on attend une touche
C0D-48         PHA            on sauve la touche
C0E-2C 0D 02   BIT $020D      est-on en mode Minitel ?
C11-50 0D      BVC $CC20      non -------------------------------------------------
C13-A9 08      LDA #$08       oui, on enlève le marqueur                          I
C15-20 5D C7   JSR $C75D                                                          I
C18-A9 20      LDA #$20                                                           I
C1A-20 5D C7   JSR $C75D                                                          I
C1D-4C 2E CC   JMP $CC2E      pourquoi pas BNE ???                                I
C20-A4 61      LDY $61        <-----------------------------------------------------
C22-A6 65      LDX $65
C24-B1 26      LDA ($26),Y    on efface la barre en vidéo inverse
C26-29 7F      AND #$7F
C28-91 26      STA ($26),Y
C2A-C8         INY
C2B-CA         DEX
C2C-D0 F6      BNE $CC24
C2E-68         PLA            on lit la touche
C2F-C9 20      CMP #$20       espace ?
C31-F0 08      BEQ $CC3B      oui, on sort ----------------------------------------
C33-C9 1B      CMP #$1B       ESC ?                                                 I
C35-F0 04      BEQ $CC3B      oui, on sort ----------------------------------------
C37-C9 0D      CMP #$0D       RETURN ?
C39-D0 03      BNE $CC3E      non, on passe
C3B-A6 60      LDX $60        sortie, A contient le code de la touche de sortie <-
C3D-60         RTS
C3E-C9 0A      CMP #$0A       flèche bas ?
C40-D0 2B      BNE $CC6D      non, on passe ---------------------------------------
```

```
CC42-A5 60     LDA $60        la barre est en bas ?
CC44-C5 67     CMP $67
CC46-F0 05     BEQ $CC4D
CC48-E6 60     INC $60        non, on descend la barre
CC4A-4C FD CB  JMP $CBFD      et on recommence
CC4D-24 68     BIT $68        oui, y a-t-il encore des choix ?
CC4F-30 AC     BMI $CBFD      non, on recommence
CC51-E6 60     INC $60        oui,on ajuste les variables
CC53-E6 67     INC $67
CC55-E6 66     INC $66
CC57-2C 0D 02  BIT $020D      mode Minitel ?
CC5A-70 8F     BVS $CBEB      oui, on reaffiche tout
CC5C-A6 62     LDX $62        non, on scrolle la fenêtre
CC5E-A4 63     LDY $63
CC60-20 54 DE  JSR $DE54      vers le haut
CC63-A4 63     LDY $63
CC65-A6 60     LDX $60
CC67-20 F9 CC  JSR $CCF9      on affiche le choix suivant
CC6A-4C FD CB  JMP $CBFD      et on boucle
CC6D-C9 0B     CMP #$0B       flèche haut ? <----------------------------------
CC6F-D0 29     BNE $CC9A      non, on passe ------------------------------------
CC71-A5 60     LDA $60        on est en haut de la fenêtre ?
CC73-C5 66     CMP $66
CC75-D0 1B     BNE $CC92      non-----------------------------------------------
CC77-A5 60     LDA $60        oui, premier choix ?                          I I
CC79-F0 19     BEQ $CC94   ---oui
CC7B-C6 66     DEC $66     I  on ajuste les variables                       I I
CC7D-C6 67     DEC $67     I                                                I I
CC7F-C6 60     DEC $60     I                                                I I
CC81-2C 0D 02  BIT $020D   I  mode Minitel ?                                I I
CC84-70 11     BVS $CC97  -+--oui, on repart                               I I
CC86-A6 62     LDX $62    II  non, on scrolle vers le bas                  I I
CC88-A4 63     LDY $63    II                                               I I
CC8A-20 5C DE  JSR $DE5C  II                                               I I
CC8D-A4 62     LDY $62    II                                               I I
CC8F-4C 65 CC  JMP $CC65  II  et on affiche le choix et on boucle          I I
CC92-C6 60     DEC $60    II  <-------------------------------------------- I
CC94-4C FD CB  JMP $CBFD  I-->on boucle                                     I
CC97-4C EB CB  JMP $CBEB  --->                                              I
CC9A-C9 30     CMP #$30       est-on entre "0" <-------------------------------
CC9C-90 F6     BCC $CC94
CC9E-C9 3A     CMP #$3A       et "9" inclus ?
CCA0-B0 F2     BCS $CC94      non, on repart
CCA2-A6 60     LDX $60        oui,le choix courant
CCA4-E0 19     CPX #$19       est inférieur à 25 ?
CCA6-90 06     BCC $CCAE   ---oui
CCA8-A6 66     LDX $66     I  non, on place la barre
CCAA-86 60     STX $60     I  sur le premier choix
CCAC-B0 E6     BCS $CC94   I  et on boucle
CCAE-48        PHA         --->on sauve la touche
CCAF-06 60     ASL $60        choix actuel   *2
CCB1-A5 60     LDA $60
CCB3-06 60     ASL $60        *4
CCB5-06 60     ASL $60        *8
CCB7-65 60     ADC $60        *10
CCB9-85 60     STA $60
CCBB-68        PLA
CCBC-29 0F     AND #$0F       on isole le numéro demandé
CCBE-65 60     ADC $60        on ajoute au choix courant
CCC0-E9 00     SBC #$00       -1 si on dépasse
```

```
CCC2-85 60      STA $60
CCC4-90 E2      BCC $CCA8       et on boucle tant qu'on dépasse
CCC6-C5 66      CMP $66
CCC8-90 DE      BCC $CCA8       ou jusqu'a ce qu'on soit à la fin de la fenêtre
CCCA-C5 67      CMP $67
CCCC-F0 02      BEQ $CCD0
CCCE-B0 D8      BCS $CCA8
CCD0-4C FD CB   JMP $CBFD       et on boucle
```

AFFICHER LA BARRE EN VIDEO INVERSE

```
CCD3-20 5A CD   JSR $CD5A       on positionne le curseur en X,Y
CCD6-2C 0D 02   BIT $020D       mode Minitel ?
CCD9-50 0F      BVC $CCEA       non ----------------------------------------------
CCDB-A2 02      LDX #$02        oui, on décale le curseur à droite 3 fois         I
CCDD-A9 09      LDA #$09                                                          I
CCDF-20 5D C7   JSR $C75D                                                         I
CCE2-CA         DEX                                                               I
CCE3-10 F8      BPL $CCDD                                                         I
CCE5-A9 2D      LDA #$2D        et on affiche "-"                                 I
CCE7-4C 5D C7   JMP $C75D                                                         I
CCEA-A4 61      LDY $61         on prend l'abscisse de la fenêtre <---------------
CCEC-A6 65      LDX $65         on prend la longueur de la barre
CCEE-B1 26      LDA ($26),Y
CCF0-09 80      ORA #$80        et on inverse la barre
CCF2-91 26      STA ($26),Y
CCF4-C8         INY
CCF5-CA         DEX
CCF6-D0 F6      BNE $CCEE
CCF8-60         RTS
```

AFFICHE LE CHOIX X

```
CCF9-98         TYA             on va afficher le choix X
CCFA-48         PHA             en $61,Y
CCFB-8A         TXA
CCFC-48         PHA
CCFD-48         PHA
CCFE-20 5A CD   JSR $CD5A       positionnement
CD01-E8         INX             on est déja sur le choix 0
CD02-A5 69      LDA $69         on sauve l'adresse de la table des choix
CD04-A4 6A      LDY $6A
CD06-85 15      STA $15         car on lit sur une autre banque
CD08-84 16      STY $16
CD0A-A0 00      LDY #$00        on indexe le premier caractère
CD0C-CA         DEX           -->on est sur le bon choix ?
CD0D-F0 11      BEQ $CD20     I   oui ------------------------------------------
CD0F-C8         INY           I   non, on indexe le code suivant                I
CD10-D0 02      BNE $CD14     I                                                 I
CD12-E6 16      INC $16       I   sur 16 bits                                   I
CD14-20 11 04   JSR $0411     I   et on lit le caractère suivant               I
CD17-D0 F6      BNE $CD0F     I   0 ?                                          I
CD19-C8         INY           I   oui, on a passé un choix                     I
CD1A-D0 F0      BNE $CD0C     I                                                 I
CD1C-E6 16      INC $16       I                                                 I
CD1E-D0 EC      BNE $CD0C     ---inconditionnel                                 I
```

```
CD20-A6 16      LDX $16         <----------------------------------------
CD22-18         CLC
CD23-98         TYA
CD24-65 15      ADC $15
CD26-90 01      BCC $CD29
CD28-E8         INX
CD29-85 02      STA $02         on met l'adresse du choix
CD2B-86 03      STX $03         dans $02-03
CD2D-A9 20      LDA #$20        et un espace comme valeur par défaut en aff. dec.
CD2F-85 14      STA $14
CD31-68         PLA             on sort le numéro du choix
CD32-18         CLC
CD33-69 01      ADC #$01        +1 car le premier est 0
CD35-A0 00      LDY #$00        AY=A
CD37-A2 01      LDX #$01        pour 3 codes à afficher
CD39-20 39 CE   JSR $CE39       et on affiche le numéro du choix en décimal
CD3C-A9 20      LDA #$20        puis un espace
CD3E-20 5D C7   JSR $C75D
CD41-A5 02      LDA $02         puis le choix lui-même
CD43-A4 03      LDY $03
CD45-20 A8 C7   JSR $C7A8       par XWSTR0
CD48-A0 01      LDY #$01
CD4A-20 11 04   JSR $0411       on lit le code suivant le choix
CD4D-38         SEC             si 0, C=1
CD4E-F0 01      BEQ $CD51
CD50-18         CLC             sinon C=0
CD51-66 68      ROR $68         et on ajuste MENFLG
CD53-68         PLA             on restaure les registres
CD54-AA         TAX
CD55-68         PLA
CD56-A8         TAY
CD57-24 68      BIT $68         et N=1 si dernier choix
CD59-60         RTS
```

POSITIONNE LE CURSEUR EN $61,X

Principe:Classique, dans la norme ASCII, on envoie la séquence US/Y+64/X+64 pour positionner le curseur en X,Y.

```
CD5A-A9 1F      LDA #$1F
CD5C-20 5D C7   JSR $C75D       on envoie US
CD5F-98         TYA             Y est inférieur à 64
CD60-09 40      ORA #$40        donc on force le bit 6 à 1 pour ajouter 64
CD62-20 5D C7   JSR $C75D       Y+64
CD65-A5 61      LDA $61
CD67-09 40      ORA #$40
CD69-4C 5D C7   JMP $C75D       $61+64
```

DECALAGE D'UN BLOC MEMOIRE

Principe:Etant donné l'usage sur-fréquent qu'il est fait de cette routine, elle a été optimisée en temps, au détriment de la place mémoire. Deux routines ont été écrites, une pour les décalages vers le haut et une pour les décalages vers le bas. Le principe consiste d'abord à décaler le petit morceau final de moins de 256 octets pour se ramener ensuite à un décalage de pages. Ce sont en effet les ajustements des adresses de transfert qui sont les plus gourmand en temps. Il faut ici près de

14 millisecondes pour décaler l'écran d'une ligne, ce qui est un temps près de 2 fois plus court que sur la V1.1. Le décalage vers le bas est d'ailleurs très sensiblement plus lent que le décalage vers le haut.

```
CD6C-48        PHA             on sauve les registres
CD6D-8A        TXA
CD6E-48        PHA
CD6F-98        TYA
CD70-48        PHA
CD71-38        SEC
CD72-A5 06     LDA $06         on calcule le nombre d'octets à décaler
CD74-E5 04     SBC $04         dans YX
CD76-A8        TAY
CD77-A5 07     LDA $07
CD79-E5 05     SBC $05
CD7B-AA        TAX
CD7C-90 3B     BCC $CDB9       si début>fin, on sort
CD7E-86 0B     STX $0B         nombre de pages à décaler dans $0B
CD80-A5 08     LDA $08         on compare la cible
CD82-C5 04     CMP $04         à l'adresse de début
CD84-A5 09     LDA $09
CD86-E5 05     SBC $05
CD88-B0 35     BCS $CDBF       cible>=début, on décale de bas en haut ----------
CD8A-98        TYA             décalage de haut en bas                          I
CD8B-49 FF     EOR #$FF        on complémente le nombre d'octets hors pages     I
CD8D-69 01     ADC #$01        a 2 car on change de sens                        I
CD8F-A8        TAY                                                              I
CD90-85 0A     STA $0A                                                          I
CD92-90 03     BCC $CD97                                                        I
CD94-CA        DEX             au besoin, on enlève une page                    I
CD95-E6 07     INC $07                                                          I
CD97-38        SEC                                                              I
CD98-A5 08     LDA $08         on enlève le nombre d'octets hors-pages          I
CD9A-E5 0A     SBC $0A         à l'adresse cible                                I
CD9C-85 08     STA $08                                                          I
CD9E-B0 02     BCS $CDA2                                                        I
CDA0-C6 09     DEC $09                                                          I
CDA2-18        CLC                                                              I
CDA3-A5 07     LDA $07         et le nombre de pages +1                         I
CDA5-E5 0B     SBC $0B         à l'adresse de fin                               I
CDA7-85 07     STA $07                                                          I
CDA9-E8        INX                                                              I
CDAA-B1 06     LDA ($06),Y     on décale les octets hors pages d'abord          I
CDAC-91 08     STA ($08),Y     puis les pages complètes ensuite                 I
CDAE-C8        INY                                                              I
CDAF-D0 F9     BNE $CDAA                                                        I
CDB1-E6 07     INC $07         puisqu'on déplace des pages, on ajuste           I
CDB3-E6 09     INC $09         uniquement les poids fort des adresses           I
CDB5-CA        DEX                                                              I
CDB6-D0 F2     BNE $CDAA                                                        I
CDB8-38        SEC             C=1 si le décalage s'est effectué <-------------+---
CDB9-68        PLA             C=0 sinon                                        I I
CDBA-A8        TAY             on restaure les registres et on sort             I I
CDBB-68        PLA                                                              I I
CDBC-AA        TAX                                                              I I
CDBD-68        PLA                                                              I I
CDBE-60        RTS                                                              I I
CDBF-8A        TXA             décalage vers le haut <--------------------------
CDC0-18        CLC                                                                I
CDC1-65 05     ADC $05         on ajoute le nombre de pages à l'adresse           I
```

```
CDC3-85 05      STA $05         de début du bloc
CDC5-8A         TXA
CDC6-18         CLC
CDC7-65 09      ADC $09         et à l'adresse cible
CDC9-85 09      STA $09
CDCB-E8         INX             et on décale comme précedemment
CDCC-88         DEY
CDCD-B1 04      LDA ($04),Y
CDCF-91 08      STA ($08),Y
CDD1-98         TYA
CDD2-D0 F8      BNE $CDCC
CDD4-C6 05      DEC $05
CDD6-C6 09      DEC $09
CDD8-CA         DEX
CDD9-D0 F1      BNE $CDCC
CDDB-F0 DB      BEQ $CDB8       inconditionnel ----------
```

DONNEES POUR CONVERSION DECIMALE

```
CDDD            BYT $0A         poids faible de 000A=10
CDDE            BYT $64                         0064=100
CDDF            BYT $E8                         03E8=1000
CDE0            BYT $E1                         2710=10000
CDE1            BYT $00         poids forts des nombres ci-dessus
CDE2            BYT $00
CDE3            BYT $03
CDE4            BYT $27
```

CONVERSION DECIMALE 0-99

```
CDE5-A2 00      LDX #$00        on indique 2 chiffres
CDE7-A0 00      LDY #$00        AY=A
CDE9            BYT $2C         et on saute la suivante
```

CONVERSION DECIMAL 0-65535

```
CDEA-A2 03      LDX #$03        on indique 5 chiffres
CDEC            BYT $2C         et on saute la suite
```

CONVERSION DECIMALE 0-9999

```
CDED-A2 02      LDX #$02        on indique 4 chiffres
```

CONVERSION EN DECIMAL

Principe: En entrée, AY contient le nombre et X le nombre de chiffres du nombre (son logarithme décimal en fait). On soustrait successivement des puissances de dix au nombre pour trouver ses chiffres. Les 0 devant le chiffre seront remplacés par le contenu de DEFAFF. En sortie, les ($10) nombres du chiffre sont dans ($11-12).

```
CDEF-85 0D      STA $0D         on sauve le nombre
CDF1-84 0E      STY $0E
CDF3-A9 00      LDA #$00        et on met 0 dans l'index d'écriture
CDF5-85 0F      STA $0F         et l'indicateur de 0 en début de nombre
CDF7-85 10      STA $10
CDF9-A9 FF      LDA #$FF        et 255 dans $0C
```

```
:CDFB-85 0C      STA $0C
:CDFD-E6 0C      INC $0C          pour calculer le chiffre courant
:CDFF-38         SEC
:CE00-A5 0D      LDA $0D          on enlève la puissance de dix courante
:CE02-A8         TAY
:CE03-FD DD CD   SBC $CDDD,X      au poids faible
:CE06-85 0D      STA $0D
:CE08-A5 0E      LDA $0E
:CE0A-48         PHA
:CE0B-FD E1 CD   SBC $CDE1,X      puis au poids fort
:CE0E-85 0E      STA $0E
:CE10-68         PLA
:CE11-B0 EA      BCS $CDFD        jusqu'à ce qu'on déborde
:CE13-84 0D      STY $0D          on reprend le nombre d'avant qu'il
:CE15-85 0E      STA $0E          déborde.
:CE17-A5 0C      LDA $0C          le chiffre est 0 ?
:CE19-F0 04      BEQ $CE1F        oui ----------------------
:CE1B-85 0F      STA $0F          non, on le stocke        I
:CE1D-D0 07      BNE $CE26    ---inconditionnel            I
:CE1F-A4 0F      LDY $0F      I  0 en début de nombre ?<-I
:CE21-D0 03      BNE $CE26    0--non
:CE23-A5 14      LDA $14      I  oui, on prend le caractère par défaut
:CE25            BYT $2C      I  et on saute l'instruction suivante
:CE26-09 30      ORA #$30     -->on ajoute "0"
:CE28-20 32 CE   JSR $CE32        et on le stocke dans le buffer
:CE2B-CA         DEX              pour tous les chiffres sauf les unités
:CE2C-10 CB      BPL $CDF9
:CE2E-A5 0D      LDA $0D          puis les unités
:CE30-09 30      ORA #$30         +"0" (un nombre nul affiche 0)
:CE32-A4 10      LDY $10          on prend l'index
:CE34-91 11      STA ($11),Y      et on stocke le chiffre
:CE36-E6 10      INC $10          puis on augmente l'index
:CE38-60         RTS
```

CONVERSION DECIMALE AVEC AFFICHAGE SUR CANAL 0

Action:En entrée, tout comme pour la conversion décimale. Le nombre est stocké dans le bas de la pile (BUFTRV).

```
:CE39-48         PHA
:CE3A-A9 00      LDA #$00         On sauve BUFTRV ($100)
:CE3C-85 11      STA $11
:CE3E-A9 01      LDA #$01
:CE40-85 12      STA $12          Dans $11,$12
:CE42-68         PLA
:CE43-20 EF CD   JSR $CDEF        on convertit
:CE46-A0 00      LDY #$00
:CE48-B9 00 01   LDA $0100,Y
:CE4B-20 5D C7   JSR $C75D        et on affiche
:CE4E-C8         INY
:CE4F-C4 10      CPY $10
:CE51-D0 F5      BNE $CE48
:CE53-60         RTS
```

CONVERSION HEXADECIMALE

Action:Renvoie dans AY les deux chiffres Hexa du nombre contenu dans A en entrée

```
:CE54-48         PHA              on sauve le nombre
```

```
CE55-29 0F      AND #$0F        on isole le quartet de droite
CE57-20 60 CE   JSR $CE60       on le convertit en un chiffre hexa
CE5A-A8         TAY             dans Y
CE5B-68         PLA
CE5C-4A         LSR             puis on isole le quartet de gauche
CE5D-4A         LSR
CE5E-4A         LSR
CE5F-4A         LSR             que l'on convertit à son tour
CE60-09 30      ORA #$30        on ajoute "0"
CE62-C9 3A      CMP #$3A        supérieur à 9 ?
CE64-90 02      BCC $CE68       non
CE66-69 06      ADC #$06        oui, on en fait une lettre
CE68-60         RTS
```

MULPTIPLICATION PAR 40

Action:Par glissements et additions, on multiplie A (en fait AY avec Y=0) par 40. Le résultat est à la fois dans AY et dans RES. Cette routine sert au calculs de position sur l'écran puisqu'il a 40 colonnes.

```
CE69-A0 00      LDY #$00        AY=A
CE6B-85 00      STA $00         dans RES
CE6D-84 01      STY $01
CE6F-0A         ASL             *2
CE70-26 01      ROL $01
CE72-0A         ASL             *4
CE73-26 01      ROL $01
CE75-65 00      ADC $00         *5
CE77-90 02      BCC $CE7B
CE79-E6 01      INC $01
CE7B-0A         ASL             *10
CE7C-26 01      ROL $01
CE7E-0A         ASL             *20
CE7F-26 01      ROL $01
CE81-0A         ASL             *40
CE82-26 01      ROL $01
CE84-85 00      STA $00         dans RES et AY
CE86-A4 01      LDY $01
CE88-60         RTS
```

ADDITION D'ENTIERS

Action:Additionne RES et AY dans RES et AY.

```
CE89-18         CLC
CE8A-65 00      ADC $00         poids faible de RES
CE8C-85 00      STA $00         + poids faible de AY
CE8E-48         PHA             sauvé
CE8F-98         TYA
CE90-65 01      ADC $01         poids fort de RES
CE92-85 01      STA $01         + poids fort de AY
CE94-A8         TAY             sauvé
CE95-68         PLA
CE96-60         RTS
```

MULTIPLICATION ENTIERE

Principe:On teste chaque bit du mutiplicateur, tout en décalant le multiplicande à gauche à chaque fois (comme en décimal). Si ce bit est 1, on ajoute le multiplicande décalé au résultat partiel. Et ainsi de suite.
Cette routine aurait pu être encore plus rapide en testant au début si le multiplicande est bien plus grand que le multiplicateur, en les permutant sinon. En effet, multiplier 1 (RES) par 65535 (AY) revient au même (en temps) que multiplier 65535 par lui même !
En entrée, AY et RES contiennent les facteurs. En sortie, TR0-1-2-3 contient le résultat.

```
CE97-85 10      STA $10         multiplicateur dans TR5-6
CE99-84 11      STY $11
CE9B-A2 00      LDX #$00        0 dans :
CE9D-86 0C      STX $0C         le résultat
CE9F-86 0D      STX $0D         -
CEA1-86 0E      STX $0E         -
CEA3-86 0F      STX $0F         -
CEA5-86 02      STX $02         et dans l'extension du multiplicande
CEA7-86 03      STX $03
CEA9-A2 10      LDX #$10        pour 16 décalages (inutile ! voir plus loin)
CEAB-46 11      LSR $11         on isole le bit 16-X du multiplicateur
CEAD-66 10      ROR $10
CEAF-90 19      BCC $CECA       0 ----------------------------------------------
CEB1-18         CLC             1 on additionne le multiplicande au             I
CEB2-A5 00      LDA $00         résultat, octet 0                               I
CEB4-65 0C      ADC $0C                                                         I
CEB6-85 0C      STA $0C                                                         I
CEB8-A5 01      LDA $01         octet 1                                         I
CEBA-65 0D      ADC $0D                                                         I
CEBC-85 0D      STA $0D                                                         I
CEBE-A5 02      LDA $02         octet 2                                         I
CEC0-65 0E      ADC $0E                                                         I
CEC2-85 0E      STA $0E                                                         I
CEC4-A5 03      LDA $03         octet 3                                         I
CEC6-65 0F      ADC $0F                                                         I
CEC8-85 0F      STA $0F                                                         I
CECA-06 00      ASL $00         on multiplie le multiplicande par 2 <-----------
CECC-26 01      ROL $01
CECE-26 02      ROL $02
CED0-26 03      ROL $03
CED2-A5 10      LDA $10         le multiplicateur est nul ?
CED4-05 11      ORA $11
CED6-F0 03      BEQ $CEDB       oui, on sort -----------------------------------
CED8-CA         DEX             inutile ! après 16 glissements, le multiplicateurI
CED9-D0 D0      BNE $CEAB       est toujours nul ... BNE suffit                 I
CEDB-60         RTS             Z=1 en sortie <--------------------------------
```

DIVISION ENTIERE

Principe:comme pour la multiplication, on teste les bit du diviseur et on enlève ou non le diviseur au reste selon les bits. On sort après 16 tests.
Divise RES par AY. Le résultat est dans RES et le reste dans RESB
Cette routine est bien plus optimisée que la précedente...

```
CEDC-85 0C      STA $0C         diviseur dans TR0-1
```

```
:DE-84 0D      STY $0D
:E0-A2 00      LDX #$00        reste est 0
:E2-86 02      STX $02
:E4-86 03      STX $03
:E6-A2 10      LDX #$10        pour 16 décalages du diviseur
:E8-06 00      ASL $00         on décale le dividende
:EA-26 01      ROL $01         avec report dans le reste partiel
:EC-26 02      ROL $02
:EE-26 03      ROL $03
:F0-38         SEC
:F1-A5 02      LDA $02         on soustrait le diviseur
:F3-E5 0C      SBC $0C         au reste dans YA
:F5-A8         TAY
:F6-A5 03      LDA $03
:F8-E5 0D      SBC $0D         diviseur > reste ?
:FA-90 06      BCC $CF02       oui, on saute ----------------------------------
:FC-84 02      STY $02         non, on sauve le reste partiel                  I
:FE-85 03      STA $03                                                         I
:00-E6 00      INC $00         et on ajoute 1 au dividende                     I
:02-CA         DEX             <-----------------------------------------------
:03-D0 E3      BNE $CEE8
:05-60         RTS
```

EFFACE L'ECRAN HIRES

```
:06-A9 00      LDA #$00        RES=$A000, début de l'écran HIRES
:08-A0 A0      LDY #$A0
:0A-85 00      STA $00
:0C-84 01      STY $01
:0E-A0 68      LDY #$68        YX=$BF68, fin de l'écran
:10-A2 BF      LDX #$BF
:12-A9 40      LDA #$40        on remplit avec %01000000, soit pixels éteints
```

REMPLIT UNE ZONE MEMOIRE

```
rincipe:identique au décalage, pour se ramener à un remplissage de pages, on
        commence par remplir la zone hors-page (- de 256 octets).
        Remplit la zone de RES à YX avec le code dans A.

:F14-48        PHA             on sauve le code
:F15-38        SEC
:F16-98        TYA
:F17-E5 00     SBC $00         on calcule dans YX
:F19-A8        TAY             la taille de la zone à remplit
:F1A-8A        TXA
:F1B-E5 01     SBC $01         (YX-RES)
:F1D-AA        TAX
:F1E-84 02     STY $02         et si RES est supérieur à YX ?!?!
:F20-68        PLA             on sort le code de remplissage
:F21-A0 00     LDY #$00        premier octet hors page
:F23-C4 02     CPY $02         on à fini la zone hors-page ?
:F25-B0 05     BCS $CF2C       oui ----------------------------------------
:F27-91 00     STA ($00),Y     non, on remplit la zone hors-page             I
:F29-C8        INY                                                           I
:F2A-D0 F7     BNE $CF23                                                     I
:F2C-48        PHA             on sauve la donnée <-------------------------
:F2D-98        TYA             AY=Y
```

```
:F2E-A0 00     LDY #$00
:F30-20 89 CE  JSR $CE89      on augmente l'adresse de début de Y (déja remplie)
:F33-68        PLA            on prend la donnée
:F34-E0 00     CPX #$00       y a-t-il des pages à remplir ?
:F36-F0 0C     BEQ $CF44      non ----------------------------------------------
:F38-A0 00     LDY #$00       oui, on remplit une page                         I
:F3A-91 00     STA ($00),Y                                                     I
:F3C-C8        INY                                                             I
:F3D-D0 FB     BNE $CF3A                                                       I
:F3F-E6 01     INC $01        et toutes les autres                             I
:F41-CA        DEX                                                             I
:F42-D0 F6     BNE $CF3A                                                       I
:F44-60        RTS            Z=1 en sortie <-----------------------------------
```

PASSAGE EN HIRES

```
:F45-A2 00     LDX #$00
:F47-A0 FF     LDY #$FF
:F49-8C AA 02  STY $02AA      pattern = %11111111, ligne pleine
:F4C-C8        INY            Y=0 X=0
:F4D-20 F3 E7  JSR $E7F3      on initialise les données HRS
:F50-AD 0D 02  LDA $020D      est-on en HIRES ?
:F53-30 B1     BMI $CF06      oui, on efface simplement
:F55-09 80     ORA #$80       non, on force mode HIRES
:F57-8D 0D 02  STA $020D
:F5A-08        PHP            on inhibe les interruptions
:F5B-78        SEI
:F5C-A9 1F     LDA #$1F       place attribut HIRES
:F5E-8D 67 BF  STA $BF67
:F61-20 A4 CF  JSR $CFA4      on attend 1/3 de seconde
:F64-20 D8 FE  JSR $FED8      on déplace la table des caractères
:F67-A9 5C     LDA #$5C       fenêtre TEXT en HIRES
:F69-A0 02     LDY #$02
:F6B-A2 00     LDX #$00       fenêtre 0
:F6D-20 FD DE  JSR $DEFD      on initialise la fenêtre TEXT
:F70-20 06 CF  JSR $CF06      on efface l'écran HIRES
:F73-28        PLP            pourquoi pas PLP/JMP $CF06 ???
:F74-60        RTS
```

PASSAGE EN MODE TEXT

```
:F75-AD 0D 02  LDA $020D      est-on déja en TEXT ?
:F78-10 29     BPL $CFA3      BPL $CF74 aurait été mieux -----------------------
:F7A-08        PHP            on inhibe les IRQ                                I
:F7B-78        SEI                                                             I
:F7C-29 7F     AND #$7F       on indique mode TEXT                             I
:F7E-8D 0D 02  STA $020D      dans FLGTEL                                      I
:F81-20 DB FE  JSR $FEDB      on replace la table des caractères               I
:F84-A9 56     LDA #$56                                                        I
:F86-A0 02     LDY #$02                                                        I
:F88-A2 00     LDX #$00                                                        I
:F8A-20 FD DE  JSR $DEFD      on initialise la fenêtre 0 (tout l'écran)        I
:F8D-A9 1A     LDA #$1A       on pose un attribut TEXT                         I
:F8F-8D DF BF  STA $BFDF      à la fin de l'écran                              I
:F92-20 A4 CF  JSR $CFA4      on attend 1/3 de seconde                         I
:F95-A2 28     LDX #$28       on efface l'écran                                I
:F97-A9 20     LDA #$20                                                        I
```

```
F99-9D 7F BB   STA $BB7F,X                                              I
F9C-CA         DEX                                                     I
F9D-D0 FA      BNE $CF99                                               I
F9F-20 20 DE   JSR $DE20       on affiche le curseur                   I
FA2-28         PLP             PLP/JMP aurait été mieux ...            I
FA3-60         RTS             <-------------------------------------------
```

ATTEND 1/3 DE SECONDES

```
FA4-A0 1F      LDY #$1F        on boucle 8192 fois
FA6-A2 00      LDX #$00        avec un temps moyen de 4 us par boucle
FA8-CA         DEX             ce qui fait pres de 32000 us
FA9-D0 FD      BNE $CFA8       donc 1/3 de seconde
FAB-88         DEY
FAC-D0 FA      BNE $CFA8
FAE-60         RTS
```

TESTE SI TOUS LES BUFFERS SONT VIDES

rincipe:On scrute les buffers 1 par 1, si on en trouve un non vide, on sort avec C=0, sinon C=1. Cette routine ne teste pas les buffers définis par l'utilisateur. De plus, que viennent faire les deux entrées (CFBF et CFB1) puisque qu'à part le fait de faire perdre 5 octets, elles ont le même effet ?

```
FAF-38         SEC             C=1
FB0            BYT $24
FB1-18         CLC             ou C=0
FB2-66 15      ROR $15         dans b7($15) (?)
FB4-A2 00      LDX #$00
FB6-20 0F C5   JSR $C50F    -->on teste si le buffer X est vide
FB9-90 08      BCC $CFC3    I  non, on sort avec C=0 ----------------------
FBB-8A         TXA          I                                             I
FBC-69 0B      ADC #$0B     I  on passe au buffer suivant (C=1 donc +11)  I
FBE-AA         TAX          I                                             I
FBF-E0 30      CPX #$30     I  a-t-on fait les 4 buffers ?                I
FC1-D0 F3      BNE $CFB6    ---non, on boucle                             I
FC3-08         PHP             on sauve C <-----------------------------------
FC4-A9 DC      LDA #$DC        on indexe prompt plein
FC6-A0 CF      LDY #$CF        si tous les buffers sont vides
FC8-B0 04      BCS $CFCE
FCA-A9 E6      LDA #$E6        ou prompt vide sinon
FCC-A0 CF      LDY #$CF
FCE-24 15      BIT $15         ???
FD0-10 05      BPL $CFD7       -------------------------
FD2-20 F9 FE   JSR $FEF9       on redéfinit le prompt    I
FD5-28         PLP             et on sort                I
FD6-60         RTS                                       I
FD7-20 F9 FE   JSR $FEF9       id. (quel intérêt ?) <-----
FDA-28         PLP
FDB-60         RTS
```

TABLES DE DEFINITION DU PROMPT

```
FDC BYT $7F                          code ASCII 127 (prompt)
FDD BYT 00,00,08,3C,3E,3C,08,00      valeurs prompt plein
FE5 BYT 0                            terminateur
```

```
CFE6 BYT $7F                          code du prompt
CFE7 BYT 00,00,08,34,32,34,08,00      valeurs prompt vide
CFEF BYT 0                            terminateur
```

PLACE UN NOM DE FICHIER DANS BUFNOM

```
Action:place le nom de fichier de longueur X, à l'adresse AY dans BUFNOM.
       En sortie, X=0 si longueur du nom nulle; X=1 si le nom n'a qu'une lettre,
       C=1 s'il contient des jokers; X>127 s'il est invalide.
       X=128 si un des caractères du nom est invalide
       X=129 si drive spécifié incorrect
       X=130 si nom de fichier trop long
       X=131 si plusieurs * dans le nom
       X=132 si l'extension est invalide
```

```
CFF0-85 15     STA $15          on sauve l'adresse du nom dans $15-16
CFF2-84 16     STY $16          pour permettre la lecture inter-banques
CFF4-86 00     STX $00          longeur dans $0
CFF6-E6 00     INC $00          +1 pour tests
CFF8-AC 0C 02  LDY $020C        on prend le drive courant
CFFB-8C 17 05  STY $0517        dans BUFNOM
CFFE-8C 00 05  STY $0500        et DRIVE
D001-A0 0C     LDY #$0C         on place 12 jokers ("?") dans BUFNOM
D003-A9 3F     LDA #$3F         pour nom="???????????.???"
D005-99 17 05  STA $0517,Y
D008-88        DEY
D009-D0 FA     BNE $D005
D00B-8A        TXA              longueur nulle ?
D00C-F0 3B     BEQ $D049        oui on sort
D00E-E0 01     CPX #$01         longeur=1 ?
D010-D0 22     BNE $D034        non, on passe  ----------------------------------
D012-20 DF D0  JSR $D0DF        oui, on lit le caractère                        I
D015-38        SEC                                                              I
D016-E9 41     SBC #$41         on lui enlève "A"                               I
D018-C9 04     CMP #$04         supérieur à 3 ?                                 I
D01A-B0 40     BCS $D05C        oui, ce n'est pas un drive                      I
D01C-8D 17 05  STA $0517        non, on place le drive demandé                  I
D01F-8D 00 05  STA $0500        dans BUFNOM et DRIVE                            I
D022-A2 01     LDX #$01         y a-t-il encore des jokers ?                    I
D024-A0 0C     LDY #$0C                                                         I
D026-A9 3F     LDA #$3F                                                         I
D028-D9 17 05  CMP $0517,Y                                                      I
D02B-F0 05     BEQ $D032        oui, on sort avec C=1 --                        I
D02D-88        DEY                                    I                        I
D02E-D0 F8     BNE $D028                              I                        I
D030-18        CLC              non, on sort avec C=0  I                        I
D031-60        RTS                                     I                        I
D032-38        SEC              <----------------------                         I
D033-60        RTS                                                              I
D034-A0 01     LDY #$01         on lit le premier caractère <-------------------
D036-20 DF D0  JSR $D0DF
D039-C9 2D     CMP #$2D         est-ce un trait "-" ?
D03B-D0 1F     BNE $D05C        non, ok on lit le nom ---------------------------
D03D-A0 00     LDY #$00         oui, on reprend le numéro du drive              I
D03F-20 DF D0  JSR $D0DF                                                        I
D042-38        SEC                                                              I
D043-E9 41     SBC #$41         on enlève "A"                                   I
```

```
D045-B0 03     BCS $D04A      si < 0
D047-A2 81     LDX #$81       on indique drive invalide
D049-60        RTS
D04A-C9 04     CMP #$04
D04C-B0 F9     BCS $D047      (ou supérieur à 4)
D04E-E0 02     CPX #$02       sinon, X=2 (juste drive+"-")
D050-D0 04     BNE $D056      non, on lit la suite du nom
D052-8D 0C 02  STA $020C      oui, on change le drive par défaut
D055-60        RTS
D056-8D 17 05  STA $0517      on met le drive dans BUFNOM
D059-A0 02     LDY #$02       et on lit la suite du nom
D05B           BYT $2C        sauter l'instruction suivante
D05C-A0 00     LDY #$00       pas de drive dans le nom, on lit tout le nom
D05E-A2 00     LDX #$00
D060-20 DF D0  JSR $D0DF    -->on lit le caractère suivant
D063-B0 1D     BCS $D082    I  fin du nom ? oui
D065-C9 2E     CMP #$2E     I  on a un point ?
D067-F0 19     BEQ $D082    I  oui
D069-C9 2A     CMP #$2A     I  non, une étoile ?
D06B-F0 22     BEQ $D08F    I  oui
D06D-20 FB D0  JSR $D0FB    I  non, caractère valide ?
D070-90 06     BCC $D078    I  oui
D072-A2 80     LDX #$80     I  non, on sort avec X=128
D074-60        RTS          I
D075-A2 82     LDX #$82     I
D077-60        RTS          I
D078-E0 09     CPX #$09     I  X=9 ?
D07A-F0 F9     BEQ $D075    I  oui, on sort avec X=130
D07C-9D 18 05  STA $0518,X  I  non, on place le caractère dans BUFNOM
D07F-E8        INX          I
D080-D0 DE     BNE $D060    ---et on boucle
D082-A9 20     LDA #$20       on complète la fin du nom avec des espaces
D084-E0 09     CPX #$09       nom fini ?
D086-F0 06     BEQ $D08E      oui
D088-9D 18 05  STA $0518,X    non, on complète
D08B-E8        INX
D08C-D0 F6     BNE $D084
D08E-88        DEY            on lit le caractère suivant
D08F-A2 00     LDX #$00
D091-20 DF D0  JSR $D0DF
D094-90 0E     BCC $D0A4      on a fini ? non, on saute
D096-A0 02     LDY #$02       oui, on met l'extension par défaut
D098-B9 5D 05  LDA $055D,Y    (on devrait la passer en majuscules avant...)
D09B-99 21 05  STA $0521,Y
D09E-88        INX
D09F-10 F7     BPL $D098
D0A1-4C 2C D0  JMP $D02C      et on termine
D0A4-C9 2E     CMP #$2E       a-t-on "." ? <-----------------------------------
D0A6-D0 CA     BNE $D072      pas de ".", nom de fichier invalide
D0A8-20 DF D0  JSR $D0DF      caractère suivant
D0AB-B0 E1     BCS $D08E      il n'y en a pas, on met l'extension par défaut
D0AD-88        DEY
D0AE-20 DF D0  JSR $D0DF    -->on lit le code
D0B1-90 0E     BCC $D0C1    I  on l'interprète
D0B3-A9 20     LDA #$20     I  il n'y en a plus,
D0B5-E0 03     CPX #$03     I  on complète l'extension avec des espaces
D0B7-F0 E8     BEQ $D0A1    I
D0B9-9D 21 05  STA $0521,X  I
D0BC-E8        INX          I
D0BD-D0 F6     BNE $D0B5    I
D0BF-F0 E0     BEQ $D0A1    I  et on sort
```

```
D0C1-C9 2A      CMP #$2A     I  est-ce "*" <----------------------------------
D0C3-D0 08      BNE $D0CD    I  non, on passe ----------------------------------
D0C5-20 DF D0   JSR $D0DF    I  oui, elle est seule ?                          I
D0C8-B0 D7      BCS $D0A1    I  oui, on  termine                               I
D0CA-A2 83      LDX #$83     I  non, trop de jokers                            I
D0CC-60         RTS          I                                                 I
D0CD-20 FB D0   JSR $D0FB    I  caractère valide ? <-----------------------------
D0D0-B0 A0      BCS $D072    I  non, nom invalide et on sort
D0D2-E0 03      CPX #$03     I  oui, fin de l'extension ?
D0D4-F0 06      BEQ $D0DC    I  oui, extension invalide et on sort ------------
D0D6-9D 21 05   STA $0521,X  I  non, on stocke le caractère                    I
D0D9-E8         INX          I                                                 I
D0DA-D0 D2      BNE $D0AE    ---et on poursuit                                 I
D0DC-A2 84      LDX #$84        <---------------------------------------------
D0DE-60         RTS

D0DF-20 11 04   JSR $0411       on lit le caractère courant <-----------
D0E2-20 F0 D0   JSR $D0F0       en majuscules                          I
D0E5-C8         INY                                                    I
D0E6-C4 00      CPY $00         est-ce le dernier ?                    I
D0E8-B0 05      BCS $D0EF       oui, C=1 non, C=0                      I
D0EA-C9 20      CMP #$20        est-ce un espace ?                     I
D0EC-F0 F1      BEQ $D0DF       oui, on lit le suivant -----------------
D0EE-18         CLC
D0EF-60         RTS
```

PASSE A EN MAJUSCULES

```
D0F0-C9 61      CMP #$61        entre "a"
D0F2-90 06      BCC $D0FA
D0F4-C9 7B      CMP #$7B        et "z"
D0F6-B0 02      BCS $D0FA
D0F8-E9 1F      SBC #$1F        oui, on retire 32 (31+(1-C) avec C=0)
D0FA-60         RTS
```

TESTE SI A EST VALIDE DANS UN NOM DE FICHIER

```
D0FB-C9 3F      CMP #$3F        est-ce "?"
D0FD-F0 0E      BEQ $D10D       oui, ok
D0FF-C9 30      CMP #$30        entre 0
D101-90 0C      BCC $D10F
D103-C9 3A      CMP #$3A        et 9 ?
D105-90 06      BCC $D10D       oui, ok
D107-C9 41      CMP #$41        supérieur à "A" ?
D109-90 04      BCC $D10F
D10B-C9 5B      CMP #$5B        ridicule ! on met C à 0 de toutes façons !!!
D10D-18         CLC             C=0, caractère OK
D10E-60         RTS
D10F-38         SEC             C=1, caractère invalide
D110-60         RTS
```

INITIALISE LES TABLES VDT

Action:initialise la table ASCII VDT ($9000-$9400), la table des couleurs VDT ($9400-$9800) et l'écran VDT ($A000-$BF3F, HIRES).
VDT signifie mode HIRES en émulation VIDEOTEX.

```
D111-A0 00      LDY #$00        AY=$9000
```

```
D113-A9 90      LDA #$90
D115-84 00      STY $00         dans RES
D117-85 01      STA $01
D119-A2 94      LDX #$94        YX=$9400
D11B-A9 00      LDA #$00        A=0
D11D-20 14 CF   JSR $CF14       on remplit de $9000 à $9400 de 0
D120-A0 00      LDY #$00
D122-A9 94      LDA #$94
D124-84 00      STY $00
D126-85 01      STA $01
D128-A2 98      LDX #$98
D12A-A9 87      LDA #$87
D12C-20 14 CF   JSR $CF14       on remplit de $9400 à $9800 de %10000111
D12F-A0 00      LDY #$00
D131-A9 A0      LDA #$A0
D133-84 00      STY $00
D135-85 01      STA $01
D137-A0 3F      LDY #$3F
D139-A2 BF      LDX #$BF
D13B-A9 10      LDA #$10
D13D-4C 14 CF   JMP $CF14       on remplit de $A000 à $BF3F de %0001000
```

GERE LE CURSEUR VDT

Action:Calcule dans VDTASC,VDTATR et ADVDT la position du code, de la couleur et de l'affichage VDT du point courant VDT et gère le curseur VDT.

```
D140-A5 39      LDA $39         prend VDTY, ordonnée du curseur VDT
D142-20 69 CE   JSR $CE69       calcul VDTY*40
D145-48         PHA             dans AY et RES
D146-98         TYA
D147-48         PHA
D148-A9 00      LDA #$00        ajoute l'adresse de la table ASCII
D14A-A0 90      LDY #$90
D14C-20 89 CE   JSR $CE89
D14F-85 2E      STA $2E         dans ADASC (table ASCII VDT)
D151-84 2F      STY $2F
D153-85 30      STA $30         ajoute 4 pages ($400)
D155-C8         INY
D156-C8         INY
D157-C8         INY
D158-C8         INY
D159-84 31      STY $31         et stocke ADATR (table des couleurs)
D15B-68         PLA             on sort poids fort de VDTY*40
D15C-85 01      STA $01         dans $01
D15E-68         PLA             et poids faible dans A
D15F-0A         ASL             on multiplie par 8
D160-26 01      ROL $01         car chaque caractère fait 8 lignes
D162-0A         ASL
D163-26 01      ROL $01
D165-0A         ASL
D166-26 01      ROL $01
D168-85 00      STA $00         dans RES
D16A-A9 00      LDA #$00        on ajoute l'adresse de l'écran HIRES
D16C-A0 A0      LDY #$A0
D16E-20 89 CE   JSR $CE89
D171-85 2C      STA $2C         dans ADVDT (adresse du point courant en HIRES)
D173-84 2D      STY $2D
```

```
D175-4C 56 D7  JMP $D756     et on gère le curseur VDT
```

..

ENVOIE A SUR L'ECRAN VIDEOTEX

Action:Cette routine est la routine principale de l'émulateur VIDEOTEX intégré. Elle envoie A, code ASCII, en HIRES émulant la norme VIDEOTEX, en gérant tous les codes (REP,SEP,SYN, etc...). Les séquences sont affichées récursivement. La routine est compliquée à suivre car très optimisée.

```
D178-85 3E     STA $3E       on sauve la donnée
D17A-98        TYA           et les registres
D17B-48        PHA
D17C-8A        TXA
D17D-48        PHA
D17E-A5 39     LDA $39       VDTY=0 ?
D180-F0 08     BEQ $D18A     oui ----------------------------------------------
D182-8D 81 02  STA $0281     non, on sauve VDTX et VDTY                      I
D185-A5 38     LDA $38       dans $280 et $281                               I
D187-8D 80 02  STA $0280                                                     I
D18A-24 3C     BIT $3C       séquence en cours ? <--------------------------
D18C-30 5F     BMI $D1ED     oui, on passe ...
D18E-A5 3E     LDA $3E       on prend la donnée
D190-20 42 D4  JSR $D442     on la code
D193-85 00     STA $00       et on la stocke
D195-F0 4F     BEQ $D1E6     est-ce 0 ? oui, on débute un séquence
D197-C9 20     CMP #$20      non, est-ce un code de controle ?
D199-90 58     BCC $D1F3     oui, on le gère
D19B-C9 A0     CMP #$A0      est-ce un caractère OK ?
D19D-B0 42     BCS $D1E1     oui, on le sauve
D19F-8D 85 02  STA $0285
D1A2-29 7F     AND #$7F      b7 à 0
D1A4-AA        TAX           dans X
D1A5-A5 34     LDA $34       on prend VDTATR
D1A7-30 12     BMI $D1BB     mode mosaique (G1) ? oui --------------------
D1A9-A4 38     LDY $38       VDTX=39 ?                                       I
D1AB-C0 27     CPY #$27                                                      I
D1AD-D0 02     BNE $D1B1     non ------------------------------------------0
D1AF-29 DF     AND #$DF      oui, on interdit la double largeur              I
D1B1-A4 39     LDY $39       VDTY<2                                          I
D1B3-C0 02     CPY #$02                                                      I
D1B5-B0 04     BCS $D1BB     non, VDTY>1-----------------------------------0
D1B7-29 EF     AND #$EF      oui, on interdit la double hauteur              I
D1B9-85 34     STA $34       on sauve VDTATR <------------------------------
D1BB-48        PHA           on sauve encore VDTATR
D1BC-20 30 D5  JSR $D530     on place le code dans les tables VDT
D1BF-20 DC D5  JSR $D5DC     on affiche le code à l'écran
D1C2-68        PLA           on lit VDTATR
D1C3-48        PHA
D1C4-30 0A     BMI $D1D0     mode G1 ? oui ------------------------------------
D1C6-29 20     AND #$20      mode double largeur ?                           I
D1C8-F0 05     BEQ $D1D0     non ----------------------------------------------
D1CA-20 91 D3  JSR $D391     oui, on déplace le curseur à droite deux fois   I
D1CD-20 59 D7  JSR $D759     on éteint le curseur                            I
D1D0-20 91 D3  JSR $D391     on déplace le curseur à droite <-----------------
D1D3-68        PLA           on sort VDTATR
D1D4-30 0B     BMI $D1E1     mode G1 ? oui ------------------------------------
D1D6-A6 38     LDX $38       non, VDTX=0 ?                                   I
D1D8-D0 07     BNE $D1E1     non ----------------------------------------------0
```

```
)1DA-29 10      AND #$10        oui, double hauteur ?                          I
)1DC-F0 03      BEQ $D1E1       non ---------------------------------------->0
)1DE-20 A0 D3   JSR $D3A0       oui on saute une ligne                         I
)1E1-AD 85 02   LDA $0285       en sortie $00 contient la donnée codée <-------
)1E4-85 00      STA $00         (pour récursivité)
)1E6-68         PLA             une chance que les séquences on au plus 3
)1E7-AA         TAX             caractères, sans quoi la pile déborderait vite.
)1E8-68         PLA             (4 appels récursifs empilent 8 valeurs pour les
)1E9-A8         TAY             registres et 8 pour les adresses, soit 16...)
)1EA-A5 00      LDA $00
)1EC-60         RTS
```

POURSUIT UNE SEQUENCE

```
)1ED-20 2E D2   JSR $D22E       on traite la séquence
)1F0-4C E6 D1   JMP $D1E6       et on sort
```

GERE LES CODES DE CONTROLE

```
)1F3-20 F9 D1   JSR $D1F9       on traite le code
)1F6-4C E6 D1   JMP $D1E6       et on sort
```

GERE LES CODES DE CONTROLE VIDEOTEX

Principe:En entrée, A contient un code ASCII entre 0 et 31. On calcule
l'adresse de gestion du code d'après une table d'octets en ajoutant
$D37E à l'octet correspondant au code. Un JMP indirect et on exécute...

```
)1F9-AA         TAX             code dans X
)1FA-18         CLC
)1FB-BD 0B D2   LDA $D20B,X     on lit l'octet correspondant
)1FE-69 7E      ADC #$7E        on ajoute le poids faible de $D37E
)200-85 00      STA $00         on sauve
)202-A9 D3      LDA #$D3        et le poids fort de $D37E
)204-69 00      ADC #$00        que l'on sauve aussi
)206-85 01      STA $01
)208-6C 00 00   JMP ($0000)     et on exécute la routine de gestion du code.
```

TABLE DE GESTION DES CODES DE CONTROLE

```
)20B            BYT 0,0,0,0,0,0,0  Les codes de 0 à 6 ne sont pas traités
)212            BYT $01    code 07 BEL - $D37F  emét un bip
)213            BYT $04         08 BS  - $D382  déplace le curseur à gauche
)214            BYT $10         09 HT  - $D38E  déplace le curseur à droite
)215            BYT $22         10 LF  - $D3A0  déplace le curseur vers le bas
)216            BYT $44         11 VT  - $D3C2  déplace le curseur vers le haut
)217            BYT $53         12 FF  - $D3D1  efface l'écran
)218            BYT $59         13 CR  - $D3D7  curseur en début de ligne
)219            BYT $63         14 SO  - $D3E1  passe en G1
)21A            BYT $76         15 SI  - $D3F4  passe en G0
)21B            BYT 0
)21C            BYT $7D         17 Con - $D3FB  allume le curseur
)21D            BYT $B3         18 REP - $D431  répète un code
)21E            BYT $B6         19 SEP - $D434  séquence clavier et accents
)21F            BYT $80         20 Coff- $D3FE  éteint le curseur
)220            BYT 0
)221            BYT $B9         22 SYN - $D437
```

```
D222            BYT 0
D223            BYT $83         24 CAN - $D401  efface la fin de la ligne
D224            BYT $B9         25 SS2 - $D437 (comme SYN)
D225            BYT 0
D226            BYT $BC         27 ESC - $D43A  séquence à quatre caractères
D227            BYT 0,0
D229            BYT $A7         30 RS  - $D425  ramène le curseur en 0,0
D22A            BYT $BF         31 US  - $D43D  positionne le curseur en X,Y


D22B-4C B7 D2  JMP $D2B7  saut à la gestion de séquence ESC <------------------
                                                                              I
                                                                              I
                          GERE UNE SEQUENCE                                   I
                                                                              I
                                                                              I
Action:Gère totalement la séquence codée dans FLGVDO. Cette routine est       I
       récursive, c'est à dire qu'elle appelle la $D182 (son appelant). En    I
       tout cas, il semble bien difficile de respecter la norme VIDEOTEX !    I
                                                                              I
                                                                              I
D22E-A5 3C      LDA $3C         on isole le nombre de caractère (-1)          I
D230-29 03      AND #$03                                                      I
D232-85 36      STA $36         dans $36                                      I
D234-A5 3C      LDA $3C         on prend la séquence                          I
D236-0A         ASL             est-ce ESC ?                                  I
D237-30 F2      BMI $D22B       oui ------------------------------------------
D239-0A         ASL             non
D23A-0A         ASL             US ?
D23B-30 11      BMI $D24E       oui ------------------------------------------
D23D-A5 3E      LDA $3E         non, c'est REP.on lit le code actuel à envoyer I
D23F-29 3F      AND #$3F        on élimine b7b6 (entre 0 et 63)               I
D241-AA         TAX             dans X                                        I
D242-46 3C      LSR $3C         on indique fin de séquence                    I
D244-AD 85 02   LDA $0285       et on répète l'envoi du                       I
D247-20 78 D1   JSR $D178       caractère                                     I
D24A-CA         DEX             X fois                                        I
D24B-D0 F7      BNE $D244                                                     I
D24D-60         RTS                                                           I
D24E-A5 36      LDA $36         séquence US <---------------------------------
D250-F0 1D      BEQ $D26F       fin de séquence ? oui ------------------------
D252-A5 3E      LDA $3E         non, on lit la donnée                         I
D254-C9 30      CMP #$30        entre "0" et "0"+39 (?) ?                     I
D256-90 12      BCC $D26A                                                     I
D258-C9 59      CMP #$59                                                      I
D25A-B0 0E      BCS $D26A       non, incorrect -------------------------------I
D25C-8D 82 02   STA $0282       on stocke l'ordonnée dans VDTPIL              I I
D25F-C6 3C      DEC $3C         on indique un code de moins en séquence       I I
D261-A9 07      LDA #$07        encre blanche                                 I I
D263-85 34      STA $34                                                       I I
D265-A9 00      LDA #$00        fond noir                                     I I
D267-85 32      STA $32                                                       I I
D269-60         RTS                                                           I I
D26A-46 3C      LSR $3C         on indique fin de séquence <------------------<
D26C-4C 78 D1   JMP $D178       et on envoie simplement la donnée             I I
D26F-46 3C      LSR $3C         fin de séquence US <--------------------------+---
D271-A5 3E      LDA $3E         on lit le code                                I
D273-C9 30      CMP #$30        entre "0" et "0"+56 (?)                       I
D275-90 F3      BCC $D26A                                                     I
```

```
D277-C9 69      CMP #$69
D279-B0 EF      BCS $D26A       non ----------------------------------------
D27B-29 3F      AND #$3F        oui, on isole b3b2b1b0                       I
D27D-8D 83 02   STA $0283       dans VDTPIL+1                                I
D280-A5 3E      LDA $3E         la donnée est-elle > "A"                     I
D282-C9 40      CMP #$40                                                     I
D284-B0 1D      BCS $D2A3       oui ----------------------------------------I
D286-AD 82 02   LDA $0282       non, on prend l'ordonnée                     I I
D289-29 03      AND #$03        que l'on triture                             I I
D28B-0A         ASL             pour ramener de "0" à "0"+39                 I I
D28C-0A         ASL             à 0 .. 25                                    I I
D28D-6D 82 02   ADC $0282                                                    I I
D290-E9 2F      SBC #$2F                                                     I I
D292-0A         ASL                                                          I I
D293-6D 83 02   ADC $0283                                                    I I
D296-E9 2F      SBC #$2F                                                     I I
D298-85 39      STA $39         dans VDTY                                    I I
D29A-20 D7 D3   JSR $D3D7       on ramène le curseur en début de ligne       I I
D29D-20 59 D7   JSR $D759       on éteint  le curseur                        I I
D2A0-4C 40 D1   JMP $D140       on ajuste les tables et affiche le curseur   I
D2A3-20 59 D7   JSR $D759       on éteint le curseur >-----------------------I
D2A6-AD 83 02   LDA $0283       on prend l'abscisse
D2A9-85 38      STA $38         dans VDTX
D2AB-C6 38      DEC $38
D2AD-AD 82 02   LDA $0282       l'ordonnée
D2B0-29 3F      AND #$3F        dans VDTY
D2B2-85 39      STA $39
D2B4-4C 40 D1   JMP $D140       on ajuste les tables et on affiche le curseur
```

GESTION DE SEQUENCE ESC

Entrée: $36 contient le nombre de caractères de la séquence, $3E contient le caractère à traiter.
Le codage est aisé car PRO1, PRO2 et PRO3 sont suivis respectivement de 1,2 et 3 codes.
Le masquage et le démasquage devant certainement être gérés dans une version ultérieure, mais ne l'étant pas dans celle-ci, ont laissé de vilaines traces ...

```
D2B7-A6 36      LDX $36         on lit le nombre de caractères à traiter
D2B9-A5 3E      LDA $3E         on lit la donnée
D2BB-E0 03      CPX #$03        4 caractères à traiter ?
D2BD-D0 22      BNE $D2E1       non ----------------------------------------
D2BF-8D 82 02   STA $0282       on sauve le code (PROx ou 3/6)               I
D2C2-A2 00      LDX #$00        on met 0                                     I
D2C4-86 35      STX $35         dans VDTFT (pas SEP ni SS2)                  I
D2C6-C9 36      CMP #$36        est-ce 3/6 ?                                 I
D2C8-F0 0D      BEQ $D2D7       oui  ----------------------------------------I
D2CA-C9 39      CMP #$39        est-ce entre PRO1 (3/9)                      I I
D2CC-90 22      BCC $D2F0                                                    I I
D2CE-C9 3C      CMP #$3C        et PRO3 (3/B) ?                              I I
D2D0-B0 1E      BCS $D2F0       non, on passe                                I I
D2D2-29 03      AND #$03        on isole le PRO                              I I
D2D4-E9 00      SBC #$00        -1 (donc 0,1,2 pour PRO1,2,3)                I I
D2D6            BYT $2C         et on passe l'instruction suivante           I I
D2D7-A9 00      LDA #$00        on  force PRO1 <-----------------------------I
D2D9-09 C0      ORA #$C0        et on indique                                I
```

```
D2DB-85 3C      STA $3C                                                  I
D2DD-60         RTS                                                      I
                                                                         I
D2DE-46 3C      LSR $3C         fin de séquence sans action              I
D2E0-60         RTS                                                      I
                                                                         I
D2E1-E6 35      INC $35         on indexe le paramètre suivant <---------
D2E3-A6 35      LDX $35
D2E5-9D 81 02   STA $0281,X     et on le stocke dans la PILE VDT
D2E8-C6 3C      DEC $3C         on indique un code de moins
D2EA-C6 36      DEC $36
D2EC-10 EF      BPL $D2DD       on sort simplement s'il en reste
D2EE-30 EE      BMI $D2DE       on sort avec fin de séquence si c'était le dernier
```

GESTION DE SEQUENCE C1

Action:Les séquences C1 sont les séquences ESC xx, qui donnent accès aux attributs caractère (couleur, lignage, clignotement, etc...). Leur gestion est longue, mais c'est la plus importante !

```
D2F0-C9 40      CMP #$40        est-ce un attribut C1 ?
D2F2-90 E9      BCC $D2DD       non, on sort
D2F4-46 3C      LSR $3C         oui, on indique fin de séquence
D2F6-C9 48      CMP #$48        est-ce un code couleur texte ?
D2F8-B0 0D      BCS $D307       non ------------------------------------------
D2FA-29 07      AND #$07        oui, on isole la couleur                 I
D2FC-85 36      STA $36                                                  I
D2FE-A5 34      LDA $34                                                  I
D300-29 F8      AND #$F8                                                 I
D302-05 36      ORA $36                                                  I
D304-85 34      STA $34         dans VDTATR                              I
D306-60         RTS                                                      I
D307-C9 4A      CMP #$4A        est-ce un code clignotement ? <-----------
D309-B0 0C      BCS $D317       non ------------------------------------------
D30B-4A         LSR             oui, C=1 si fixe, 0 si glignotement      I
D30C-A5 34      LDA $34                                                  I
D30E-29 F7      AND #$F7                                                 I
D310-B0 02      BCS $D314                                                I
D312-09 08      ORA #$08                                                 I
D314-85 34      STA $34         dans VDTATR                              I
D316-60         RTS                                                      I
D317-C9 4C      CMP #$4C        est-ce rien ? <--------------------------
D319-90 2F      BCC $D34A       oui... on sort ---------------------------
D31B-C9 50      CMP #$50        est-ce un attribut taille ?              I
D31D-B0 11      BCS $D330       non ------------------------------------  I
D31F-29 03      AND #$03        oui                                   I  I
D321-0A         ASL                                                   I  I
D322-0A         ASL                                                   I  I
D323-0A         ASL                                                   I  I
D324-0A         ASL                                                   I  I
D325-85 36      STA $36                                               I  I
D327-A5 34      LDA $34                                               I  I
D329-29 CF      AND #$CF                                              I  I
D32B-05 36      ORA $36         on indique lequel dans                I  I
D32D-85 34      STA $34         VDTATR                                I  I
D32F-60         RTS                                                   I  I
D330-C9 58      CMP #$58        est-ce un attribut fond ? <------------  I
D332-B0 17      BCS $D34B       non ------------------------------------  I
D334-29 07      AND #$07        oui, on isole la couleur              I  I
D336-0A         ASL                                                   I  I
```

```
D337-0A       ASL
D338-0A       ASL
D339-0A       ASL
D33A-85 36    STA $36
D33C-A5 32    LDA $32
D33E-29 84    AND #$84
D340-05 36    ORA $36        dans A
D342-24 34    BIT $34        est-on en G1 ?
D344-30 02    BMI $D348      oui, on passe ----------------------------
D346-09 80    ORA #$80       non, on indique attribut à valider        I
D348-85 32    STA $32        et on stocke la couleur de fond <---------I
D34A-60       RTS            <-----------------------------------------+---
D34B-D0 00    BNE $D34D      (aie ! aie ! aie ! voila qui est louche !) <---
D34D-C9 5B    CMP #$5B       est-ce le lignage ?
D34F-B0 1B    BCS $D36C      non ----------------------------------------------
D351-4A       LSR            C=0 si lignage, 1 si fin de lignage               I
D352-24 34    BIT $34        est on en G1                                      I
D354-10 09    BPL $D35F      non----------------------------------------------- I
D356-A9 00    LDA #$00       oui, ça devient vidéo normale                  I  I
D358-90 02    BCC $D35C                                                     I  I
D35A-A9 40    LDA #$40       et vidéo inverse                               I  I
D35C-85 33    STA $33                                                       I  I
D35E-60       RTS                                                           I  I
D35F-A5 32    LDA $32        pas G1 <-------------------------------------- I
D361-29 70    AND #$70                                                        I
D363-B0 02    BCS $D367      on indique lignage ou non                        I
D365-09 04    ORA #$04                                                        I
D367-09 80    ORA #$80       et attribut à valider                            I
D369-85 32    STA $32        dans VDTPAR                                      I
D36B-60       RTS                                                             I
D36C-F0 FD    BEQ $D36B      on ne gère pas l'échappement ISO 6429 <----------
D36E-C9 5E    CMP #$5E       est-ce l'inversion vidéo ?
D370-B0 0C    BCS $D37E      non ----------------------------------------------
D372-4A       LSR            C=1 si inverse, 0 si normal                      I
D373-A5 34    LDA $34                                                         I
D375-29 BF    AND #$BF                                                        I
D377-90 02    BCC $D37B                                                       I
D379-09 40    ORA #$40                                                        I
D37B-85 34    STA $34        on indique dans VDTATR                           I
D37D-60       RTS                                                             I
D37E-60       RTS            holala !!! quelle horreur ! <--------------------
```

ROUTINES DE GESTION DES CODES DE CONTROLE

CODE 7 - CTRL G

```
D37F-4C D8 DD  JMP $DDD8      envoie un OUPS
```

CODE 8 - CTRL H

```
D382-20 59 D7  JSR $D759      éteint le curseur
D385-A5 38     LDA $38        on est en colonne 0 ?
D387-F0 35     BEQ $D3BE      oui ----------------------------------------
D389-C6 38     DEC $38        non, on déplace vers la gauche <-------------- I
D38B-4C 56 D7  JMP $D756      et on affiche le curseur <-------------------- I I
                                                                           I I I
```

CODE 9 - CTRL I

```
D38E-20 59 D7  JSR $D759      on éteint le curseur                         I I I
```

```
D391-E6 38      INC $38      on déplave à droite                    I I I
D393-A5 38      LDA $38                                             I I I
D395-C9 28      CMP #$28     on est en colonne 40 ?                 I I I
D397-90 F2      BCC $D38B    non ------------------------------------ I I
D399-A5 39      LDA $39      oui, on est sur la ligne d'état ?       I I
D39B-F0 EC      BEQ $D389    oui ------------------------------------- I
D39D-20 D7 D3   JSR $D3D7     non, on descend d'une ligne              I
                                                                      I
                          CODE 10 - CTRL J                            I
                                                                      I
D3A0-20 59 D7   JSR $D759     on éteint le curseur                    I
D3A3-A6 39      LDX $39      est on en ligne 0 ?                     I
D3A5-F0 0C      BEQ $D3B3    oui, on reste ou on est ----------------- I
D3A7-E0 18      CPX #$18     non, en dernière ligne ?                 I I
D3A9-D0 02      BNE $D3AD    non ---------------------                I I
D3AB-A2 00      LDX #$00     oui, on passe en ligne 1   I             I I
D3AD-E8         INX          <--------------------------              I I
D3AE-86 39      STX $39     -->dans VDTY                              I I
D3B0-4C 40 D1   JMP $D140   I  on ajuste tables et curseur            I I
                            I                                         I I
D3B3-AD 80 02   LDA $0280   I  on est sur la ligne d'état <------------ I
D3B6-AE 81 02   LDX $0281   I                                         I
D3B9-85 38      STA $38     I                                         I
D3BB-4C AE D3   JMP $D3AE   ---et on y reste...                       I
                                                                      I
D3BE-A9 27      LDA #$27     on indique colonne 39 <-------------------
D3C0-85 38      STA $38      et on remonte

                          CODE 11 - CTRL K

D3C2-20 59 D7   JSR $D759     on éteint le curseur
D3C5-A6 39      LDX $39      on déplace vers le haut
D3C7-CA         DEX          on est en 0 ?
D3C8-D0 02      BNE $D3CC   ---non
D3CA-A2 18      LDX #$18    I  oui, on passe en ligne 24
D3CC-86 39      STX $39     -->dans VDTY
D3CE-4C 40 D1   JMP $D140     on ajuste tables et curseur

                          CODE 12 - CTRL L

D3D1-20 11 D1   JSR $D111     on initialise l'écran VIDEOTEX
D3D4-4C 25 D4   JMP $D425     et on met le curseur en 0,0

                          CODE 13 - CTRL M

D3D7-20 59 D7   JSR $D759     on éteint le curseur
D3DA-A9 00      LDA #$00     on ramène le curseur en début de ligne
D3DC-85 38      STA $38      et on remet le curseur
D3DE-4C 56 D7   JMP $D756

                          CODE 14 - CTRL N

D3E1-A9 40      LDA #$40     on indique G1' joint
D3E3-85 33      STA $33      dans VDTASC
D3E5-A5 32      LDA $32
D3E7-29 74      AND #$74     %01110100
D3E9-85 32      STA $32      et pas d'attribut G1 dans VDTPAR
D3EB-A5 34      LDA $34      on force G1
D3ED-29 0F      AND #$0F
D3EF-09 80      ORA #$80
```

```
D3F1-85 34      STA $34         dans VDTATR
D3F3-60         RTS
```

CODE 15 - CTRL O

```
D3F4-A5 34      LDA $34         on force G1 à 0 et pas d'attributs
D3F6-29 0F      AND #$0F        (passage en G0)
D3F8-85 34      STA $34         dans VDTATR
D3FA-60         RTS
```

CODE 17 - CTRL Q

```
D3FB-4C 4F D7   JMP $D74F       on allume le curseur
```

CODE 20 - CTRL T

```
D3FE-4C 4D D7   JMP $D74D       on éteint le curseur
```

CODE 23 - CTRL X

```
D401-A5 38      LDA $38         on sauve VDTX et VDTY
D403-48         PHA
D404-A5 39      LDA $39
D406-48         PHA
D407-A9 20      LDA #$20        on envoie un espace
D409-20 78 D1   JSR $D178
D40C-A5 38      LDA $38         on a fini la ligne ?
D40E-F0 09      BEQ $D419       oui ----------------------------------------
D410-C9 27      CMP #$27        inutile ! on perd des octets bêtement ...    I
D412-D0 F3      BNE $D407       BNE suffit amplement                         I
D414-A9 20      LDA #$20                                                     I
D416-20 78 D1   JSR $D178                                                    I
D419-20 59 D7   JSR $D759       on éteint le curseur <-----------------------
D41C-68         PLA             on restaure VDTX et VDTY
D41D-85 39      STA $39
D41F-68         PLA
D420-85 38      STA $38
D422-4C 40 D1   JMP $D140       et on ajuste tables et curseur
```

CODE 30 - CTRL ^

```
D425-20 D7 D3   JSR $D3D7       curseur en début de ligne
D428-20 61 D2   JSR $D261       pas d'attributs
D42B-20 59 D7   JSR $D759       curseur éteint
D42E-4C AB D3   JMP $D3AB       VDTY=1 et on ajuste les tables
```

CODES 18,19,22,24,27,31

Principe:On met dans FLGVDO les bits nécessaires à un pour expliciter la
séquence décodée : %1EYURSnn avec
E à 1 si séquence ESC
Y à 1 si séquence SYN ou CAN
U à 1 si séquence US
R à 1 si séquence REP
S à 1 si séquence SEP
nn contient le nombre de codes de la séquence-1 (00=1, 11=4 etc...)

```
)431-A9 89      LDA #$89        %10001001 , séquence REP avec deux codes
)433            BYT $2C         et on saute

)434-A9 84      LDA #$84        %10000100 , séquence SEP avec un code
)436            BYT $2C         sautons...

)437-A9 A1      LDA #$A1        %10100001 , séquence SYN avec deux codes
)439            BYT $2C         etc...

)43A-A9 C3      LDA #$C3        %11000011 , séquence ESC avec trois codes
)43C            BYT $2C         ...caetera...

)43D-A9 91      LDA #$91        %10010001 , séquence US avec deux codes

)43F-85 3C      STA $3C         dans VDTFLG0
)441-60         RTS
```

CODE UNE DONNEE A ENVOYER SUR L'ECRAN VIDEOTEX

```
rincipe:Dès le début, la routine teste si on est en séquence. Si tel est le cas
        on essaye d'interpréter le deuxème caractère. S'il est bon, on le
        traite, sinon, on sort avec A=#$5F (95).
        Voici les registres et flags en sortie:
        Si on débute une séquence SEP, $37=%01000000 et A=0
                                   SYN  $37=%10000000 et A=0
                                   SS2  (id.)
        Si on a poursuivi une sequence :
        SEP:A contient le code majoré de 95
        SS2:si c'est un code caractère unique, A contient son code VDT
            si c'est un accent, $37=%1100000 et A=0
        En fin de séquence SS2, A contient le code du caractère accentué s'il
        existe ou 95 sinon.

)442-A4 37      LDY $37         on lit l'indicateur de séquence
)444-24 37      BIT $37
)446-08         PHP             on sauve P
)447-A2 00      LDX #$00        indique pas de séquence
)449-86 37      STX $37
)44B-28         PLP
)44C-30 1C      BMI $D46A       N=1 (SS2) ----------------------------------------
)44E-70 16      BVS $D466       V=1 (SEP) ------------------------------------------ I
)450-C9 13      CMP #$13        est-ce un SEP ?                                    I  I
)452-F0 08      BEQ $D45F       oui ---------------------------------------------- I  I
)454-C9 19      CMP #$19        est-ce un SS2 ?                                 I  I  I
)456-F0 04      BEQ $D45C       oui ------------------------------------------- I  I  I
)458-C9 16      CMP #$16        ou un SYN                                    I  I  I  I
)45A-D0 09      BNE $D465   ---non                                       I  I  I  I
)45C-A9 80      LDA #$80    I   on indique SS2 ou SYN <------------------ I  I  I
)45E            BYT $2C     I   et on saute l'instruction suivante       I  I  I
)45F-A9 40      LDA #$40    I   on indique SEP <---------------------------- I  I
)461-85 37      STA $37     I   dans $37                                     I  I
)463-A9 00      LDA #$00    I   et A=0 (début de séquence)                   I  I
)465-60         RTS         --->code de séquence caractère invalide         I  I
)466-18         CLC             séquence SEP <------------------------------- I
)467-69 5F      ADC #$5F        on ajoute 95 au code dans A                    I
)469-60         RTS                                                            <
```

```
D46A-70 1E      BVS $D48A       V=1 et N=1 fin de séquence SS2 (accent)
D46C-A2 14      LDX #$14        le code est-il dans la table SS2/SYN ?
D46E-DD A7 D4   CMP $D4A7,X
D471-F0 06      BEQ $D479       oui
D473-CA         DEX
D474-10 F8      BPL $D46E
D476-A9 5F      LDA #$5F        non, A=95 et on sort
D478-60         RTS
D479-E0 05      CPX #$05        est-ce un accent ?
D47B-B0 08      BCS $D485       non, un caractère
D47D-8A         TXA             X contient le numéro de l'accent
D47E-09 C0      ORA #$C0        on met b7 et b6 à 1 pour indiquer séquence
D480-85 37      STA $37         SS2 deuxième phase dans $37
D482-A9 00      LDA #$00        A=0 (séquence en cours)
D484-60         RTS             et on sort
D485-29 1F      AND #$1F        on calcule son code
D487-09 80      ORA #$80        dans A
D489-60         RTS
D48A-48         PHA             on sauve la caractère à accentuer
D48B-98         TYA             on prends le numéro de l'accent
D48C-29 07      AND #$07        entre 0 et 7 (il n'y a que 5 accents)
D48E-AA         TAX             dans X
D48F-BD C1 D4   LDA $D4C1,X     on lit le nombre de caractères ayant cet accent
D492-A8         TAY             dans Y
D493-BD BC D4   LDA $D4BC,X     et la position de ces caractères dans X
D496-AA         TAX
D497-68         PLA             on prend le code à accentuer
D498-DD C6 D4   CMP $D4C6,X     peut-il être accentué ?
D49B-F0 06      BEQ $D4A3       oui
D49D-E8         INX
D49E-88         DEY
D49F-D0 F7      BNE $D498
D4A1-AA         TAX             non, A contient le code en sortie
D4A2-60         RTS
D4A3-BD DE D4   LDA $D4DE,X     on lit le code du caractère ainsi accentué
D4A6-60         RTS             et on sort
```

TABLES DE SEQUENCES CARACTERES

TABLES DES CODES SS2 ET SYN

```
D4A7            BYT $41,$42,$43,$48,$4B   codes accents grave,aigu,circonflexe,
                                          tréma et cédille.
D4AC            BYT $20,$23,$24,$26       rien,livre sterling,dollar,dièse
D4AF            BYT $2C,$2D,$2E,$2F       flèches gauche, haut, droite, bas
D4B4            BYT $30,$31,$38,$3C       degré,plus ou moins,divisé,un quart
D4B8            BYT $3D,$3E,$6A,$7A       un demi, un tiers, E dans O, e dans o
```

TABLES DES POSITIONS

```
D4BC            BYT 0,5,7,11    position des caractères par accent dans la table
```

NOMBRES D'ACCENT PAR CARACTERE

```
D4C1            BYT 5           accents graves (A,a,E,e,u)
D4C2            BYT 2           accents aigus  (E,e)
D4C3            BYT 7           accents circonflexes (A,a,E,e,u,i,o)
D4C4            BYT 3           trémas (E,e,i)
D4C5            BYT 2           cédilles (C,c)
```

CARACTERES RANGES PAR CATEGORIE D'ACCENT

```
D4C6            BYT "A","a","E","e","u"          accent grave
D4CB            BYT "E","e"                      accent aigu
D4CD            BYT "A","a","E","e","u","i","o" accent circonflexe
D4D4            BYT "E","e","i"                  tréma
D4D7            BYT "C","c"                      cédilles

D4D9            BYT $41,$42,$43,$48,$4B   et de deux ! une n'était pas suffisante
```

CODES CORRESPONDANT AUX CARACTERES ACCENTUES

```
D4DE            BYT $87,$97,$89,$99,$88
D4E2            BYT $82,$92
D4E5            BYT $81,$86,$8B,$9B,$96,$80,$9F
D4EC            BYT $84,$93,$94

D4EF            BYT $85,$95
```

EFFECTUE LE CODAGE VDT HIRES -> VIDEOTEX

Action:Un caractère lu sur l'écran HIRES en émulation VIDEOTEX est donnée en entrée. En sortie, Y contient le code de séquence (si séquence), A le premier code de la séquence (si séquence) et X le deuxième si la séquence est un caractère accentué. Cette routine effectue en fait le codage inverse de celui effectué par la $D44E.

```
D4F1-A2 00      LDX #$00        X=0
D4F2-C9 A0      CMP #$A0        la donnée est-elle un caractère ?
D4F4-90 05      BCC $D4FF       non ----------------------------------------------
D4F6-E9 5F      SBC #$5F        on lui enlève 95 (#A0-"A"=95)                    I
D4F9-A0 13      LDY #$13        et Y=1/3 (SEP)                                   I
D4FB-60         RTS                                                              I
D4FC-A8         TAY             donnée dans Y  <-------------------------------------
D4FD-30 04      BMI $D503       si > 128, on passe -------------------------------
D4FF-A8         TAY             non                                              I
D500-A9 00      LDA #$00        A=0                                              I
D502-60         RTS             et on sort                                       I
D503-A0 12      LDY #$12        pour 18 tests <-------------------------------------
D505-D9 DE D4   CMP $D4DE,Y     est-ce un caractère accentué ?
D508-F0 13      BEQ $D51D       oui ----------------------------------------------
D50A-88         DEY                                                              I
D50B-10 F8      BPL $D505       non, on boucle                                   I
D50D-18         CLC                                                              I
D50E-69 A0      ADC #$A0        on ajoute 160                                    I
D510-C9 2A      CMP #$2A        est-ce "-"                                       I
D512-F0 04      BEQ $D518       oui ------------------------------------------   I
D514-C9 3A      CMP #$3A        est-ce "="                                   I   I
D516-D0 02      BNE $D51A    ---non                                          I   I
D518-09 40      ORA #$40     I  on force b6 à 1 <-------------------------------   I
D51A-A0 19      LDY #$19     -->Y=1/9 SS2                                        I
D51C-60         RTS                                                              I
D51D-98         TYA             position de l'accent dans A <-------------------
D51E-A2 04      LDX #$04        on calcule sa position dans la table des
D520-DD BC D4   CMP $D4BC,X     caractères
D523-B0 03      BCS $D528       on a trouvé -------
D525-CA         DEX                             I
D526-D0 F8      BNE $D520                       I
```

```
D528-BD D9 D4  LDA $D4D9,X     on lit l'accent <-
D52B-BE C6 D4  LDX $D4C6,Y     et le caractère
D52E-D0 EA     BNE $D51A       inconditionnel, Y=SS2 et on sort
```

PLACE UNE DONNEE DANS LES TABLES VIDEOTEX

Action:En entrée, X contient le code à placer et A VDTATR. Le code modifie les tables VDTASC (ASCii), VDTATR (attributs) mais pas le pointeur écran.

```
D530-85 36     STA $36         on sauve VDTATR
D532-06 36     ASL $36         b5 dans b7
D534-06 36     ASL $36         mosaique (G1) ?
D536-A8        TAY
D537-10 26     BPL $D55F       non ----------------------------------------------
D539-48        PHA             on sauve VDTATR                                   I
D53A-8A        TXA             donnée dans A                                     I
D53B-C9 60     CMP #$60        inférieure à "a"-1 ?                              I
D53D-90 02     BCC $D541    ---oui                                               I
D53F-E9 20     SBC #$20     I  non, on lui enlève 32                             I
D541-38        SEC          -->on lui enlève 32                                  I
D542-E9 20     SBC #$20                                                          I
D544-85 36     STA $36         dans $36                                          I
D546-A5 33     LDA $33         on prend VDTASC                                   I
D548-29 40     AND #$40        on isole b6 (code ASCII ou type de mosaique)      I
D54A-05 36     ORA $36         ajouté à la donnée                                I
D54C-AA        TAX             dans X                                            I
D54D-A5 32     LDA $32         on lit VDTPAR                                     I
D54F-29 70     AND #$70        %0111000, on isole la couleur de fond             I
D551-85 36     STA $36         dans $36                                          I
D553-68        PLA             on prend VDTATR                                   I
D554-29 8F     AND #$8F        %10001111, on isole G1,Viédo et couleur de texte  I
D556-05 36     ORA $36         A contient la donnée et ses attributs             I
D558-A4 38     LDY $38         VDTX                                              I
D55A-84 36     STY $36         dans $36                                          I
D55C-4C AF D5  JMP $D5AF       pourquoi pas BPL ? on stocke dans les tables      I
D55F-E0 20     CPX #$20        la donnée est un espace ? <-----------------------
D561-D0 2B     BNE $D58E       non ----------------------------------------------
D563-24 32     BIT $32         doit-on valider l'attribut ?                      I
D565-10 27     BPL $D58E       non ----------------------------------------------D
D567-29 70     AND #$70        %01110000, on isole les attributs caractère       I
D569-85 35     STA $35         dans VDTFT                                        I
D56B-A5 32     LDA $32                                                           I
D56D-29 04     AND #$04        on isole le lignage                               I
D56F-09 80     ORA #$80        on force b7 à 1                                   I
D571-05 35     ORA $35         et les attributs dans A                           I
D573-AA        TAX             dans X                                            I
D574-A5 32     LDA $32         on lit VDTPAR                                     I
D576-29 74     AND #$74        %01110100, on isole couleur de fond et lignage    I
D578-85 32     STA $32         dans VDTPAR                                       I
D57A-29 70     AND #$70        on isole couleur de fond dans VDTFT               I
D57C-85 35     STA $35                                                           I
D57E-4A        LSR             on met la couleur dans b2b1b0                     I
D57F-4A        LSR                                                               I
D580-4A        LSR                                                               I
D581-4A        LSR                                                               I
D582-24 34     BIT $34         vidéo inverse ?                                   I
D584-50 04     BVC $D58A    ---non                                               I
```

```
D586-A5 34      LDA $34      I  oui, on prend la couleur de texte                I
D588-29 07      AND #$07     I  comme couleur de fond                            I
D58A-05 35      ORA $35      I->on ajoute l'attribut                             I
D58C-09 80      ORA #$80        et b7 à 1 (attribut à valider)                   I
D58E-24 36      BIT $36         double hauteur ? <-------------------------------
D590-50 13      BVC $D5A5       non ---------------------------------------------
D592-C6 2F      DEC $2F         oui, on enlève une page                          I
D594-C6 31      DEC $31         aux tables ASCII et ATTRIBUT                     I
D596-48         PHA             on sauve la donnée                               I
D597-38         SEC                                                              I
D598-A5 38      LDA $38         on enlève 40 à VDTX                              I
D59A-E9 28      SBC #$28                                                         I
D59C-A8         TAY             dans Y                                           I
D59D-68         PLA             on prend la donnée                               I
D59E-20 A7 D5   JSR $D5A7       on envoie le code une ligne plus haut donc       I
D5A1-E6 2F      INC $2F         puis on ajoute une page de nouveau               I
D5A3-E6 31      INC $31         et on envoie le code sur la ligne courante       I
D5A5-A4 38      LDY $38         on prend VDTX dans Y <---------------------------

D5A7-20 AF D5   JSR $D5AF       on envoie le code
D5AA-24 36      BIT $36         double largeur ?
D5AC-10 08      BPL $D5B6       non, on sort ------------------------------------
D5AE-C8         INY             oui, on écrit le code une deuxième fois          I
D5AF-48         PHA             on sauve l'attribut                              I
D5B0-8A         TXA                                                              I
D5B1-91 2E      STA ($2E),Y     on écrit le code ASCII                           I
D5B3-68         PLA                                                              I
D5B4-91 30      STA ($30),Y     et l'attribut                                    I
D5B6-60         RTS             <-----------------------------------------------
```

STOPPE L'EMULATION VIDEOTEX

```
D5B7-20 75 CF   JSR $CF75       on passe en TEXT
D5BA-4C 4D D7   JMP $D74D       on éteint le curseur VIDEOTEX
```

INITIALISE LES FLAGS VIDEOTEX

```
D5BD-A9 00      LDA #$00        on met 0
D5BF-85 3C      STA $3C         dans FLGVD0
D5C1-85 3D      STA $3D         FLGVD1
D5C3-85 37      STA $37         et le flag de séquence
D5C5-60         RTS
```

GESTION DES E/S VIDEOTEX

Action: Cette routine gère les entrée/sortie avec l'écran HIRES émulation VIDEOTEX.
En entrée N=0 pour écrire un code ASCII, C=0 pour ouvrir l'E/S et 1 pour fermer l'E/S. C'est le système standard de gestion des E/S.

```
D5C6-10 11      BPL $D5D9       N=0, on écrit une donnée sur l'écran ------------
D5C8-B0 ED      BCS $D5B7       C=1, on ferme l'E/S                              I
D5CA-20 45 CF   JSR $CF45       on passe en HIRES                                I
D5CD-20 BD D5   JSR $D5BD       on initialise les flags VIDEOTEX                 I
D5D0-A9 96      LDA #$96        AY=$D796                                         I
D5D2-A0 D7      LDY #$D7                                                         I
D5D4-20 F9 FE   JSR $FEF9       on redéfinit les caractères VIDEOTEX             I
```

```
D5D7-A9 0C     LDA #$0C        et on efface l'écran
D5D9-4C 78 D1  JMP $D178       <-----------------------------------------
```

AFFICHAGE EN MODE VIDEOTEX

Action:En entrée, X contient le code ASCII et A les attributs à afficher.

```
D5DC-46 3D     LSR $3D         on sort N/B dans C
D5DE-08        PHP
D5DF-26 3D     ROL $3D
D5E1-28        PLP
D5E2-B0 2B     BCS $D60F       si mode N/B, on saute
D5E4-A4 38     LDY $38         on prend VDTX
D5E6-F0 27     BEQ $D60F       si 0, on saute aussi
D5E8-48        PHA             on sauve le code et son attribut
D5E9-8A        TXA
D5EA-48        PHA
D5EB-88        DEY             on revient un cran à gauche
D5EC-B1 2E     LDA ($2E),Y     on lit le code
D5EE-30 1B     BMI $D60B       >128 ? oui, on saute
D5F0-AA        TAX             non, on sauve l'ASCii dans X
D5F1-B1 30     LDA ($30),Y     on lit l'attribut
D5F3-30 06     BMI $D5FB       >128 ? oui, on saute ----------------------
D5F5-E0 20     CPX #$20        est-ce un espace ?                         I
D5F7-D0 12     BNE $D60B       non, on passe                              I
D5F9-F0 05     BEQ $D600   ---inconditionnel                             I
D5FB-8A        TXA         I   on prend le code <-------------------------
D5FC-29 3F     AND #$3F    I   %00111111, on enlève b6 et b7
D5FE-D0 0B     BNE $D60B   I   s'il n'est pas nul, on saute
D600-A5 34     LDA $34     -->on prend VDTATR
D602-29 07     AND #$07        on isole la couleur du texte
D604-C6 38     DEC $38         on décale d'une colonne écran à gauche
D606-20 48 D6  JSR $D648       on affiche le code
D609-E6 38     INC $38         et on revient une colonne à droite
D60B-68        PLA             on récupère code dans X
D60C-AA        TAX             et attribut dans A
D60D-68        PLA
D60E-18        CLC
D60F-A8        TAY             b7 d'attribut=1 ?
D610-10 41     BPL $D653       non, on passe
D612-8A        TXA             b7 de code =1 ?
D613-30 25     BMI $D63A       oui , on saute ----------------------------
D615-A0 00     LDY #$00        nn, on met $9C00                           I
D617-A9 9C     LDA #$9C        dans RES                                   I
D619-84 00     STY $00         (table G2, caractères accentués)           I
D61B-85 01     STA $01                                                    I
D61D-8A        TXA             et le code                                 I
D61E-85 03     STA $03         dans $03                                   I
D620-20 B2 FE  JSR $FEB2       on place le code G1 dans la table          I
D623-A2 07     LDX #$07        pour 8 éléments                            I
D625-BD 00 9C  LDA $9C00,X     on prend un code                           I
D628-24 03     BIT $03         b6 du code=1 ?                             I
D62A-70 03     BVS $D62F       oui, on passe ----------------             I
D62C-3D 09 D7  AND $D709,X     non, on disjoint les pavés    I            I
D62F-09 40     ORA #$40        on force b6 à 1 <--------------             I
D631-9D 00 9C  STA $9C00,X     et on sauve dans la table                  I
D634-CA        DEX                                                        I
D635-10 EE     BPL $D625                                                  I
D637-4C B0 D6  JMP $D6B0       et on affiche ------------------------------ I
```

```
D63A-A9 00      LDA #$00        A=0 <-------------------------------------+---
D63C-B0 02      BCS $D640       si C=1, on passe ----------------------    I
D63E-A5 32      LDA $32         sinon, A=VDTPAR                        I   I
D640-4A         LSR             on isole la couleur de fond <----------    I
D641-4A         LSR                                                        I
D642-4A         LSR                                                        I
D643-4A         LSR                                                        I
D644-29 07      AND #$07                                                   I
D646-09 10      ORA #$10        force b4 à 1                               I
D648-A2 0F      LDX #$0F        et on stocke 16 fois de suite dans la table I
D64A-9D 00 9C   STA $9C00,X                                               I
D64D-CA         DEX                                                        I
D64E-10 FA      BPL $D64A                                                  I
D650-4C 9F D6   JMP $D69F       ------------------------------------------ I
D653-8A         TXA             caractère normal                         I I
D654-A2 13      LDX #$13        on calcule son adresse                   I I
D656-86 03      STX $03         #1300*8=#9800                            I I
D658-0A         ASL             auquel on ajoute 8*ASCii                 I I
D659-26 03      ROL $03                                                  I I
D65B-0A         ASL                                                      I I
D65C-26 03      ROL $03                                                  I I
D65E-0A         ASL                                                      I I
D65F-26 03      ROL $03                                                  I I
D661-85 02      STA $02         dans $02-03                              I I
D663-A0 07      LDY #$07        on lit les 8 codes                       I I
D665-B1 02      LDA ($02),Y                                              I I
D667-09 40      ORA #$40        force b6 à 1 (pixel HIRES)               I I
D669-99 00 9C   STA $9C00,Y     dans la table                            I I
D66C-88         DEY                                                      I I
D66D-10 F6      BPL $D665                                                I I
D66F-A4 38      LDY $38         on prend VDTX                            I I
D671-B1 2E      LDA ($2E),Y     on prend le code en VDTX,Y  <----        I I
D673-10 06      BPL $D67B       si b7=0 on passe ------------  I
D675-29 04      AND #$04        on isole b2 (souligné        I  I        I I
D677-D0 07      BNE $D680   ----oui                          I  I        I I
D679-F0 0A      BEQ $D685   I---non                          I  I        I I
D67B-88         DEY         II  on lit le code précédent <---- I        I I
D67C-10 F3      BPL $D671   II  pas d'attribut ? on passe -------        I I
D67E-30 05      BMI $D685   II  attribut, on gère --------------------   I I
D680-A9 3F      LDA #$3F    I+->on souligne le code                  I     I I
D682-8D 07 9C   STA $9C07    I  dans la dernière ligne du caractère  I     I I
D685-24 36      BIT $36      -->double largeur ?                    I     I I
D687-10 16      BPL $D69F       non ------------------------------------0 I
D689-A2 07      LDX #$07        oui,on calcule                           I I
D68B-BD 00 9C   LDA $9C00,X                                              I I
D68E-85 02      STA $02                                                  I I
D690-20 F5 D6   JSR $D6F5       le code de droite                        I I
D693-9D 08 9C   STA $9C08,X                                              I I
D696-20 F5 D6   JSR $D6F5       et celui de gauche                       I I
D699-9D 00 9C   STA $9C00,X                                              I I
D69C-CA         DEX                                                      I I
D69D-10 EC      BPL $D68B                                                I I
D69F-24 34      BIT $34         inverse vidéo ? <------------------------- I
D6A1-50 0D      BVC $D6B0       non ----------------------------------------0
D6A3-A0 0F      LDY #$0F        oui, on force b7 des octets              I
D6A5-B9 00 9C   LDA $9C00,Y     HIRES à 1                                I
D6A8-09 80      ORA #$80        pour inversion vidéo                     I
D6AA-99 00 9C   STA $9C00,Y                                              I
D6AD-88         DEY                                                      I
D6AE-10 F5      BPL $D6A5                                                I
```

```
D6B0-A5 2C     LDA $2C        AY contient l'adresse du code <----------------
D6B2-A4 2D     LDY $2D
D6B4-24 36     BIT $36        double hauteur ?
D6B6-50 07     BVC $D6BF      non ------------------------------------------------
D6B8-38        SEC            oui, on passe une ligne au dessus                 I
D6B9-E9 40     SBC #$40                                                         I
D6BB-88        DEY                                                              I
D6BC-B0 01     BCS $D6BF                                                        I
D6BE-88        DEY                                                              I
D6BF-85 00     STA $00        et on stocke l'adresse dans RES <----------------
D6C1-84 01     STY $01
D6C3-A2 00     LDX #$00       Y=0
D6C5-46 03     LSR $03        b7 de $03=0
D6C7-A4 38     LDY $38        on lit l'abcisse du caractère
D6C9-BD 00 9C  LDA $9C00,X    on lit le code
D6CC-91 00     STA ($00),Y    et on l'affiche
D6CE-24 36     BIT $36        double largeur ?
D6D0-10 06     BPL $D6D8      non ------------------------------------
D6D2-BD 08 9C  LDA $9C08,X    oui, on lit la deuxième moitié         I
D6D5-C8        INY            et on l'affiche                        I
D6D6-91 00     STA ($00),Y                                           I
D6D8-18        CLC            <-------------------------------------
D6D9-A5 00     LDA $00        on ajoute 40 à l'adresse du caractère
D6DB-69 28     ADC #$28
D6DD-85 00     STA $00
D6DF-90 02     BCC $D6E3
D6E1-E6 01     INC $01
D6E3-24 36     BIT $36        double hauteur
D6E5-50 08     BVC $D6EF      non ----------------------------
D6E7-A5 03     LDA $03        oui, on inverse b7               I
D6E9-49 80     EOR #$80       de $03                           I
D6EB-85 03     STA $03                                         I
D6ED-30 D8     BMI $D6C7      on affiche deux fois la ligne    I
D6EF-E8        INX            <--------------------------------
D6F0-E0 08     CPX #$08
D6F2-D0 D3     BNE $D6C7
D6F4-60        RTS
```

CALCULE A ET $02 POUR DOUBLE LARGEUR

Principe:on sort à droite 3 bits de $02 et on les double dans A.
$02 contient en entrée une ligne de définition d'un caractère, et en sortie les 3 bits de gauche de cette ligne. Deux appels successifs à cette routine retournent donc les parties droite et gauche d'une ligne de caractère doublées.

```
D6F5-A9 00     LDA #$00       A=0
D6F7-A0 03     LDY #$03       pour 3 décalages
D6F9-46 02     LSR $02        on sort un bit dans C
D6FB-08        PHP            on sauve C
D6FC-6A        ROR            on entre ce bit dans A
D6FD-28        PLP
D6FE-6A        ROR            deux fois
D6FF-88        DEY
D700-D0 F7     BNE $D6F9
D702-4A        LSR            on ramène à b0 dans A
D703-4A        LSR
D704-29 3F     AND #$3F       inutile ! A contenait 0 en entrée !
D706-09 40     ORA #$40       et on force b6 pour affichage HIRES
```

```
D708-60         RTS
```

TABLE DE CONVERSION POUR G1 DISJOINT

```
D709            BYT $1B,$1B   ($1B=%00011011, ce qui disjoint %00xxxxxx par EOR )
D70B            BYT $00,$1B   (EOR %00000111                                    )
D70D            BYT $00,$1B   (-------------                                    )
D70F            BYT $1B,$00   (     %00011100 ce qui est bien le disjoint de %111)
```

AFFICHE UN MOTIF MOSAIQUE EN HIRES VIDEOTEX

Action:Affiche un morceau de mosaique (colonne dans VDTGX et ligne dans VDTGY).
Pour afficher le bloc, il faut savoir qu'un bloc median a trois ligne alors qu'un bloc bordant a deux lignes. D'ou le calcul du départ. Ensuite on affiche simplement le bloc sans déplacer les attributs.

```
D711-A9 02      LDA #$02        A=2
D713-A2 03      LDX #$03        X=3
D715-A4 3B      LDY $3B         on lit VDTGY (0,1,ou 2 selon la ligne du motif)
D717-F0 09      BEQ $D722       Y=0 X=3 A=2 ----------------------------------------
D719-CA         DEX             X=X-1                                           I
D71A-A9 03      LDA #$03        A=3 si X=2                                      I
D71C-88         DEY             si Y était 1                                    I
D71D-F0 03      BEQ $D722       X=2 et A=3 ok ----------------------------------0
D71F-E8         INX                                                             I
D720-A9 05      LDA #$05        X=3 et A=5                                      I
D722-20 69 CE   JSR $CE69       A*40 dans AY on calcule la ligne Y <-------------
D725-A5 2C      LDA $2C         on additionne
D727-A4 2D      LDY $2D
D729-20 89 CE   JSR $CE89       l'adresse du motif
D72C-A9 38      LDA #$38        %00111000 dans A
D72E-A4 3A      LDY $3A         Y=VDTGX, 0 ou 1 (mosaique a deux colonnes)
D730-F0 02      BEQ $D734       si 0, on saute, A a le motif colonne 0 -----------
D732-A9 07      LDA #$07        sinon, motif colonne 1 = %00000111              I
D734-85 02      STA $02         dans RESB <-------------------------------------
D736-A4 38      LDY $38
D738-B1 00      LDA ($00),Y     et on prend l'octet
D73A-0A         ASL             on sort le bit de vidéo inverse
D73B-30 02      BMI $D73F       si pixels, on saute
D73D-A9 80      LDA #$80        sinon, on force pixel
D73F-6A         ROR             sans altérer la vidéo inverse
D740-45 02      EOR $02         on dépose le motif
D742-91 00      STA ($00),Y     et on le stocke
D744-98         TYA
D745-18         CLC
D746-69 28      SBC #$28        +40 pour ligne suivante
D748-A8         TAY
D749-CA         DEX
D74A-D0 EC      BNE $D738
D74C-60         RTS             et on sort
```

ETEINT LE CURSEUR VIDEOTEX

```
D74D-18         CLC             C=0
D74E            BYT $24         et on saute le SEC
```

ALLUMER LE CURSEUR VIDEOTEX

```
D74F-38         SEC             C=1
D750-08         PHP             on sauve P
```

```
D751-06 3D      ASL $3D
D753-28         PLP
D754-66 3D      ROR $3D         et on met C dans b7 de FLGVD1
```

AFFICHER LE CURSEUR VIDEOTEX

```
D756-A9 80      LDA #$80        A=%10000000
D758            BYT $2C         et on saute la suite
```

EFFACER LE CURSEUR VIDEOTEX

```
D759-A9 00      LDA #$00        A=%00000000
D75B-25 3D      AND $3D         si pas de curseur, alors on ne fait rien.
D75D-24 3D      BIT $3D         est-on en mode graphique ?
D75F-50 02      BVC $D763       non, ok ---------------
D761-A9 00      LDA #$00        oui, pas de curseur   I
D763-85 02      STA $02         dans RES8 <------------
D765-A5 2C      LDA $2C         adresse de la ligne du curseur dans RES
D767-A4 2D      LDY $2D
D769-85 00      STA $00
D76B-84 01      STY $01
D76D-A4 38      LDY $38
D76F-B1 30      LDA ($30),Y     on lit le code actuel
D771-30 0A      BMI $D77D       si inverse video, on saute
D773-29 40      AND #$40        si attribut aussi
D775-F0 06      BEQ $D77D       ------------------------------------------------
D777-A5 02      LDA $02         on prend l'octet                                I
D779-49 80      EOR #$80        et on lui accole le curseur                     I
D77B-85 02      STA $02         dans RES8                                       I
D77D-A2 08      LDX #$08        on prend la position du curseur <---------------
D77F-A4 38      LDY $38
D781-B1 00      LDA ($00),Y     on prend le code qui s'y trouve
D783-29 7F      AND #$7F        on lui met le curseur ou non
D785-05 02      ORA $02
D787-91 00      STA ($00),Y     et on le stocke
D789-18         CLC
D78A-98         TYA
D78B-69 28      ADC #$28        +40 pour ligne suivante
D78D-A8         TAY
D78E-90 02      BCC $D792
D790-E6 01      INC $01         éventuellement poids fort aussi
D792-CA         DEX             pour 8 lignes
D793-D0 EC      BNE $D781
D795-60         RTS
```

TABLE DE REDEFINITION POUR CARACTERES VIDEOTEX SPECIAUX

```
D796            BYT $2F         "/"  devient plus longue
D797            BYT 01,02,04,04,08,08,10,20  codes hexadécimaux de redéfinition

D79F            BYT $5C         "ç" devient grand backslash (inverse de ci-dessus)
D7A0            BYT 20,10,08,08,04,04,02,01

D7A8            BYT $5F         livre sterling devient trait bas
D7A9            BYT 00,00,00,00,00,00,00,3F

D7B1            BYT $60         copyright devient trait mi-hauteur
D7B2            BYT 00,00,3F,00,00,00,00

D7BA            BYT $7B         accolade gauche devient trait vertical à gauche
```

```
D7BB            BYT 20,20,20,20,20,20,20,20

D7C3            BYT $7C         barre verticale devient verticale médiane
D7C4            BYT 08,08,08,08,08,08,08,08

D7CC            BYT $7D         accolade fermée devient trait vertical à droite
D7CD            BYT 01,01,01,01,01,01,01,01

D7D5            BYT $7E         pave strié devient trait haut
D7D6            BYT 3F,00,00,00,00,00,00,00
D7DE            BYT 00          fin de table de redéfinition
```

GESTION DU CLAVIER

Action:Scrute le clavier et éventuellement, si une touche est pressée, stocke la donnée dans le buffer et gère la répétition.
étant donné que l'on teste la répétition sur la colonne et non sur un caractère précis, la pression simultanée (en fait impossible) sur deux touches fait répéter la plus proche de la ligne 0 même si on la relache mais qu'on garde l'autre pressée.

```
D7DF-20 03 D9   JSR $D903       on scrute le clavier
D7E2-F0 2E      BEQ $D812  ----si aucune touche, on sort
D7E4-AE 70 02   LDX $0270  I    touche pressée au tour précédent ?
D7E7-10 08      BPL $D7F1  I    non ----------------------------------------------
D7E9-AD 71 02   LDA $0271  I    la touche est celle que l'on avait préssée ?     I
D7EC-3D E8 01   AND $01E8,XI    oui (X=128+col, $1E8+128=$268, KBDCOL !)         I
D7EF-D0 16      BNE $D807  I---on gère la répétition                             I
D7F1-88         DEY        II   Y contient la colonne où la touche a été pressée <
D7F2-B9 68 02   LDA $0268,YII   on lit la colonne
D7F5-8D 71 02   STA $0271  II   dans $271
D7F8-98         TYA        II
D7F9-09 80      ORA #$80   II   et b7 de $270 à 1
D7FB-8D 70 02   STA $0270  II
D7FE-20 1F D8   JSR $D81F  II   on convertit en ASCII et on gère le buffer
D801-AD 72 02   LDA $0272  II   on lit le nombre avant répétition
D804-4C 18 D8   JMP $D818  II   dans le compteur
D807-CE 74 02   DEC $0274  I--->même touche pressée, faut-il répéter ?
D80A-D0 0F      BNE $D81B  I    non ----------------------------------------------
D80C-20 1F D8   JSR $D81F  I    oui, on convertit la même touche dans le buffer  I
D80F-4C 15 D8   JMP $D815  I    et on indique touche pressée et délai --------    I
D812-8D 70 02   STA $0270  ---->on indique pas de touche pressée           I     I
D815-AD 73 02   LDA $0273       on place le diviseur <-------------------------  I
D818-8D 74 02   STA $0274       de répétition dans le délai                      I
D81B-60         RTS             <-----------------------------------------------
```

```
D81C-4C DD D8   JMP $D8DD       saut à la gestion de la touche FUNCT
```

CONVERSION EN ASCII DANS LE BUFFER

Action:D'après le motif de la colonne et le numéro de la colonne où une touche a été pressée. Les codes de controle sont gérés immédiatement et la gestion de la touche FUNCT est séparée. En sortie, on trouve dans le buffer clavier deux codes, un code de type KBDSHT et le code ASCII de la touche:
KBDSHT : %CFxxxxxS, C=1 si CTRL, F=1 si FUNCT et S=1 si SHIFT pressé.

```
81F-20 BF C8   JSR $C8BF      on gère la RS232 (décidément, on la gère partout)
822-A9 00      LDA #$00       on pousse 0
824-48         PHA
825-AD 70 02   LDA $0270      on lit le numéro de colonne
828-0A         ASL            dans b6b5b3
829-0A         ASL
82A-0A         ASL
82B-A8         TAY            et Y
82C-AD 71 02   LDA $0271      on lit la colonne
82F-4A         LSR          -->est-on sur la bonne ligne ?
830-B0 03      BCS $D835    I  oui ----------------------------------------------
832-C8         INY          I  non, ligne suivante                              I
833-90 FA      BCC $D82F    ---inconditionnel                                   I
835-AD 6C 02   LDA $026C      on lit colonne 4 <-------------------------------
838-AA         TAX            dans X
839-29 90      AND #$90       %10010000, SHIFT pressé ?
83B-F0 08      BEQ $D845      non, on passe ----------------------------------
83D-68         PLA            on met 1 dans la pile                           I
83E-09 01      ORA #$01       on indique SHIFT dans la pile                   I
840-48         PHA                                                            I
841-98         TYA            on prend l'index caractère                      I
842-69 3F      ADC #$3F       on ajoute 64 (C=1) pour majuscule               I
844-A8         TAY                                                            I
845-98         TYA            <-----------------------------------------------
846-C9 20      CMP #$20       colonne 4 ?
848-90 09      BCC $D853      en dessous -------------------------------------
84A-E9 08      SBC #$08       on enlève une ligne                             I
84C-C9 58      CMP #$58       %01011000, on dépasse ?                         I
84E-90 02      BCC $D852    ---non                                            I
850-E9 08      SBC #$08     I on enlève une colonne                           I
852-A8         TAY          ---->on replace                                   I
853-8A         TXA            on prend la colonne 4 <-------------------------
854-29 20      AND #$20       bit 5 ? (FUNCT)
856-D0 C4      BNE $D81C      oui, on gère la touche FUNCT
858-B1 2A      LDA ($2A),Y    non, on lit le code ASCII de la touche demandée
85A-2C 75 02   BIT $0275      majuscule ?
85D-10 0A      BPL $D869      non --------------------------------------------
85F-C9 61      CMP #$61       oui, on passe en majuscule                      I
861-90 06      BCC $D869      le code si il y a lieu                          I
863-C9 7B      CMP #$7B                                                       I
865-B0 02      BCS $D869                                                      I
867-E9 1F      SBC #$1F       en enlevant 32 (C=0)                            I
869-A8         TAY            code ASCII dans Y <-----------------------------
86A-8A         TXA
86B-29 04      AND #$04       touche CTRL ?
86D-F0 12      BEQ $D881      non --------------------------------------------
86F-2D 6F 02   AND $026F      C pressé ?                                      I
872-F0 05      BEQ $D879      non ------------------------------------------- I
874-A9 80      LDA #$80       oui, on indique CTRL-C                        I I
876-8D 7E 02   STA $027E      pourquoi pas SEC/ROR ?                        I I
879-68         PLA            <------------------------------------------------
87A-09 80      ORA #$80       on indique CTRL dans la pile                    I
87C-48         PHA                                                            I
87D-98         TYA            puisque CTRL, on limite le code                 I
87E-29 1F      AND #$1F       entre 0 et 31 (1 pour A,2 pour B, etc...)       I
880-A8         TAY                                                            I
881-98         TYA            on lit le code dans A <-------------------------
882-A2 00      LDX #$00       X=0
884-48         PHA            on pousse le code ASCII
```

```
D885-C9 06      CMP #$06        est-ce CTRL-F ?
D887-D0 07      BNE $D890     ---non
D889-AD 75 02   LDA $0275     I  oui, on bascule le bip clavier
D88C-49 40      EOR #$40      I
D88E-B0 23      BCS $D8B3     I  inconditionnel ----------------------------------
D890-C9 14      CMP #$14      -->est-ce CTRL-T ?                                  I
D892-F0 1A      BEQ $D8AE     ----oui                                             I
D894-C9 17      CMP #$17      I   est-ce CTRL-W ?                                 I
D896-D0 07      BNE $D89F     I---non                                             I
D898-AD 75 02   LDA $0275     II  on force ESC=CTRL-T                             I
D89B-49 20      EOR #$20      II                                                  I
D89D-B0 14      BCS $D8B3     II  inconditionnel ----------------------------------0
D89F-C9 1B      CMP #$1B      I-->est-ce ESC ?                                    I
D8A1-D0 13      BNE $D8B6     I   non ------------------------------------------ I
D8A3-AD 75 02   LDA $0275     I   oui, ESC=CTRL-T ?                          I   I
D8A6-29 20      AND #$20      I                                              I   I
D8A8-F0 0C      BEQ $D8B6     I   non ----------------------------------------0   I
D8AA-68         PLA           I   oui                                            I
D8AB-A9 00      LDA #$00      I   on pousse 0 dans la pile                      I
D8AD-48         PHA           I   (on devrait pousser CTRL-T/#14 !)             I
D8AE-AD 75 02   LDA $0275     --->on bascule le mode Majuscules                  I
D8B1-49 80      EOR #$80                                                       I
D8B3-8D 75 02   STA $0275        et on stocke FLGKBD <-----------------------+-------
D8B6-68         PLA              on sort la donnée <------------------------
D8B7-A2 00      LDX #$00         X=buffer clavier (et de deux !)
D8B9-20 1D C5   JSR $C51D        et on envoie le code ASCII
D8BC-68         PLA              on sort le code KBDSHT
D8BD-A2 00      LDX #$00         encore ?!?!?! décidément...
D8BF-20 1D C5   JSR $C51D        et on écrit le code KBDSHT
D8C2-2C 75 02   BIT $0275        bip clavier actif ?
D8C5-50 07      BVC $D8CE        BVC $D8F9 aurait permis de gagner un octet...
D8C7-A2 CF      LDX #$CF         on indexe bip clavier
D8C9-A0 D8      LDY #$D8
D8CB-4C E7 D9   JMP $D9E7        et on émat.
D8CE-60         RTS
```

DONNEES BIP CLAVIER

```
D8CF            BYT 1F,00,00,00,00,00  soit période 0,5 ms ou fréquence 2 KHz
D8D6            BYT 3E,10,00,00        pour canal A , volume géré par l'enveloppe
D8DA            BYT 1F,00,00           pour enveloppe 0, période 0,5 ms
```

GESTION DE LA TOUCHE FUNCT

Action:On teste d'abord si FUNCT sert à obtenir le copyright ou le pavé strié. Si c'est le cas, on n'indique pas FUNCT pressé. Sinon, on teste s'il y a une gestion utilisateur de la touche FUNCT. Sinon, on sort. Si oui, on saute à la gestion utilisateur dont les premiers codes doivent être PHA pour sortir KBDSHT de la pile et LDA ($2A),Y pour lire l'ASCII de la touche de fonction.

```
D8DD-B1 2A      LDA ($2A),Y      on lit le code ASCII de la touche pressée
D8DF-C9 2D      CMP #$2D         est-ce "="
D8E1-F0 15      BEQ $D8F8        oui, on affiche le copyright ------------------
D8E3-C9 3D      CMP #$3D         est-ce "-"                                    I
D8E5-F0 14      BEQ $D8FB        oui, on affiche le pavé strié ---------------- I
D8E7-68         PLA              aucun des deux                                I I
```

```
D8E8-09 40      ORA #$40        on indique FUNCT pressée
D8EA-48         PHA
D8EB-AD 75 02   LDA $0275       faut-il gérer les touches de fonction ?
D8EE-4A         LSR
D8EF-B0 0F      BCS $D900       oui, Y contient l'index de la touche ---------
D8F1-B1 2A      LDA ($2A),Y     on lit le code
D8F3-29 1F      AND #$1F        comme CTRL
D8F5-09 80      ORA #$80        mais en inverse vidéo
D8F7            BYT $2C         sauter l'instruction suivante
D8F8-A9 60      LDA #$60        indexe (c) <----------------------------------+--+
D8FA            BYT $2C         sauter l'instruction suivante                 I  I
D8FB-A9 7E      LDA #$7E        indexe pavé strié <---------------------------+--+
D8FD-4C 82 D8   JMP $D882       et on envoie les données dans le buffer       I
D900-6C 76 02   JMP ($0276)     gestion FUNCT utilisateur <--------------------
```

SCRUTATION DU CLAVIER

Principe: On envoie au PSG le numéro de la ligne sous la forme 255-2^(colonne), c'est à dire en mettant à 0 le bit colonne, ce qui donne %10111111 pour la colonne 1. Ensuite, la donnée reste sur le port, on va donc pouvoir tester les 8 lignes sans s'en soucier. Pour tester une ligne, on met son numéro dans PB0-1-2, et on met le strobe clavier pour demander le test. Quelque microsecondes après, le strobe est à 1 si le contact était fait à l'intersection ligne/colonne. Auquel cas on indique dans KBDCOL un bit à 1 dans la bonne colonne.
Avant de sortir, on teste si une touche à été pressée, auquel cas Z=0, sinon Z=1. En sortie, Y contient le numéro de la colonne ou une touche à été pressée. Si plusieurs touches ont été pressées, c'est la plus proche de lig/col 0/0 qui sera prise en compte.

```
D903-A0 07      LDY #$07        pour 8 colonnes
D905-A9 7F      LDA #$7F        %01111111 pour motif colonne 7
D907-48         PHA             dans la pile <---------------------------------
D908-AA         TAX
D909-A9 0E      LDA #$0E        registre 14 du PSG (IOA)
D90B-20 1A DA   JSR $DA1A       on envoie la colonne
D90E-A9 00      LDA #$00        on met 0 dans la colonne pour l'instant
D910-99 68 02   STA $0268,Y
D913-20 BF C8   JSR $C8BF       on gère la RS232 en guise de délai de réponse (!)
D916-AD 00 03   LDA $0300       on lit port B
D919-29 B8      AND #$B8        dans A (on élimine le numéro de ligne)
D91B-AA         TAX             dans X
D91C-18         CLC
D91D-69 08      ADC #$08        on ajoute 8 (pour forcer le strobe)
D91F-85 1F      STA $1F         dans $1F
D921-8E 00 03   STX $0300    -->on repose le port B. b2b1b0 contient la ligne
D924-E8         INX          I  on passe à la ligne suivante
D925-A9 08      LDA #$08     I  a-t-on un strobe ?                    ---------
D927-2D 00 03   AND $0300    I                                        I
D92A-D0 06      BNE $D932    I  oui, touche pressée ------------------+--------
D92C-E4 1F      CPX $1F      --+->non, on à fini le test ?            I       I
D92E-D0 F1      BNE $D921    I---non, ligen suivante                  I
D930-F0 14      BEQ $D946    I   inconditionnel ------------------------+-----
D932-CA         DEX          I   on prend la ligne <-------------------+--+----
D933-8A         TXA          I                                          I  I
D934-48         PHA          I   que l'on sauve                         I  I
D935-29 07      AND #$07     I   isole le numéro (pas b7)               I  I
D937-AA         TAX          I                                          I  I
```

```
D938-BD A9 D9  LDA $D9A9,XI   lit la position du bit correspondant
D93B-19 68 02  ORA $0268,YI   et le place
D93E-99 68 02  STA $0268,YI   dans la bonne colonne
D941-68        PLA          I
D942-AA        TAX          I
D943-E8        INX          I  on récupère l'index
D944-D0 E6     BNE $D92C    ----inconditionnel
D946-68        PLA             on sort le motif de la colonne <----------+---
D947-38        SEC             que l'on décale                           I
D948-6A        ROR             pour colonne suivante                     I
D949-88        DEY             on compte une colonne en moins            I
D94A-10 BB     BPL $D907       fini ? non ------------------------------
D94C-A0 08     LDY #$08        on va lire les 8 colonnes
D94E-B9 67 02  LDA $0267,Y     on en lit une
D951-D0 08     BNE $D95B       non nulle, une touche a été pressée ----------
D953-C0 06     CPY #$06        colonne 4 ?                                   I
D955-D0 01     BNE $D958    ---non                                          I
D957-88        DEY          I  oui, on saute (SHIFT,CTRL et FUNCT)         I
D958-88        DEY          I->colonne suivante                            I
D959-D0 F3     BNE $D94E       si pas de touche pressée, Z=1               I
D95B-60        RTS             <- une touche à été pressée, Z=0 ----------
```

GESTION DE L'E/S CLAVIER

Principe: Comme toutes les routines d'E/S, N,V et C décident de l'opération. Pour ouvrir l'E/S, on autorise les IRQ à raison de 4 par seconde, et pour fermer l'E/S, on interdit les IRQ par T1. En sortie de lecture, est à 0 si une touche valide était présente dans le buffer. A noter le manque d'optimisation à la fin de la routine; plus on avance ...

```
D95C-30 27     BMI $D985       N=1, ouverture ou fermeture ----------------
D95E-A9 01     LDA #$01        lecture, A=1 dans $2A8 et $2A6                I
D960-8D A8 02  STA $02A8                                                     I
D963-8D A6 02  STA $02A6                                                     I
D966-08        PHP                                                           I
D967-78        SEI             on interdit les IRQ                           I
D968-A2 00     LDX #$00                                                      I
D96A-20 18 C5  JSR $C518       y a-t-il une donnée dans le buffer clavier ?  I
D96D-B0 13     BCS $D982       non, C=1 et on sort ------------------------- I
D96F-8D 79 02  STA $0279       oui, on la stocke en $279                  I I
D972-A2 00     LDX #$00        on lit le deuxième code,                   I I
D974-20 18 C5  JSR $C518       s'il n'y en a pas, on sort                 I I
D977-B0 09     BCS $D982       avec C=1 -----------------------------------0 I
D979-8D 78 02  STA $0278       on stocke le code KBDSHT                   I I
D97C-AD 79 02  LDA $0279       et dans A le code ASCII                    I I
D97F-28        PLP             et on sort avec C=0                        I I
D980-18        CLC                                                        I I
D981-60        RTS                                                        I I
D982-28        PLP             pour économiser un RTS, on aurait pu coder la <- I
D983-38        SEC             séquence PLP/CLC/RTS/PLP/SEC/RTS par PLP/CLC/    I
D984-60        RTS             BIT $3808/RTS et pointer sur 08/38 pour C=1...   <
D985-90 06     BCC $D98D       si C=0 on ouvre  ================================
D987-A9 40     LDA #$40        on interdit les interruptions par T1          >
D989-8D 0E 03  STA $030E       dans V1IER (Interrupt Enable Register)        I
D98C-60        RTS             donc Entrée clavier fermée                    I
D98D-AD 0B 03  LDA $030B       on lit V1ACR <-------------------------------
D990-09 40     ORA #$40        on force MT1 (mode roue libre sur T1)
D992-8D 0B 03  STA $030B
```

```
)995-A9 A8      LDA #$A8        AY=#$61A8, soit 25000 soit 40 IRQ par secondes
)997-A0 61      LDY #$61
)999-8D 04 03   STA $0304       dans ViTi timer T1
)99C-8C 05 03   STY $0305
)99F-A9 C0      LDA #$C0        on autorise les IRQ par T1
)9A1-8D 0E 03   STA $030E
```

VIDE LE BUFFER CLAVIER

```
Remarque:Une fonction qui aurait eu sa place dans l'HYPER-BASIC...

)9A4-A2 00      LDX #$00        on indexe buffer clavier
)9A6-4C 0C C5   JMP $C50C       et on vide

)9A9            BYT 1,2,4,8,16,32,64  puissances de deux pour scrutation clavier
```

INITIALISE LE CLAVIER

```
Remarque:L'usage des variables $2A6 $2A7 et $2A8 est encore mal défini.

D9B1-A9 FF      LDA #$FF        PORT A en en sortie (PSG)
D9B3-8D 03 03   STA $0303       dans V1DDRA
D9B6-8D A7 02   STA $02A7       et $2A7
D9B9-A9 F7      LDA #$F7        %11110111 PB3 en entrée (strobe clavier)
D9BB-8D 02 03   STA $0302       dans V1DDRB
D9BE-A9 01      LDA #$01
D9C0-8D 73 02   STA $0273       on indique vitesse
D9C3-8D 74 02   STA $0274       de répétition maximale
D9C6-8D A8 02   STA $02A8
D9C9-8D A6 02   STA $02A6
D9CC-A9 0E      LDA #$0E        et 14 IRQ (1/3 seconde) avant répétition
D9CE-8D 72 02   STA $0272
D9D1-A9 3F      LDA #$3F        table ASCII en $FA3F
D9D3-A0 FA      LDY #$FA
D9D5-85 2A      STA $2A
D9D7-84 2B      STY $2B
D9D9-4E 70 02   LSR $0270       pas de code pour répétition
D9DC-A9 C0      LDA #$C0        majuscules et bip clavier
D9DE-8D 75 02   STA $0275
D9E1-A9 00      LDA #$00        et pas CTRL-C
D9E3-8D 7E 02   STA $027E
D9E6-60         RTS
```

ENVOI DE 14 PARAMETRES AU PSG

```
Action:Envoie 14 valeurs au PSG (R0-R13), soit un son complètement défini. Par
       souci d'optimisation, la routine à deux points d'entrée selon qu'on
       l'appelle de la banque 7 ou d'une autre.
       Dans tous les cas, les 14 paramètres sont à l'adresse XY.

D9E7-18         CLC             en $D9E7, C=0, l'appel vient de la banque 7
D9E8            BYT $24         on saute le SEC
D9E9-38         SEC             en $D9E9, C=1, l'appel ne vient pas de la banque 7
D9EA-08         PHP             on sauve P
D9EB-78         SEI
D9EC-A5 16      LDA $16         et l'adresse en $15-16
D9EE-48         PHA
D9EF-A5 15      LDA $15
D9F1-48         PHA
```

```
)9F2-86 15     STX $15          et on met l'adresse des paramètres
)9F4-84 16     STY $16          dans $15-16
)9F6-08        PHP              on sauve C
)9F7-A0 00     LDY #$00         Y=0
)9F9-28        PLP              on lit C mais on le laisse dans la pile
)9FA-08        PHP
)9FB-B0 04     BCS $DA01        si C=0
)9FD-B1 15     LDA ($15),Y      on lit la donnée simplement
)9FF-90 03     BCC $DA04        sinon
)DA01-20 11 04 JSR $0411        on passe par la gestion de banques
)DA04-AA       TAX              dans X
)DA05-98       TYA              registre dans A
)DA06-48       PHA              inutile (la routine s'en occupe)
)DA07-20 1A DA JSR $DA1A        on envoie la donnée au PSG
)DA0A-68       PLA              inutile aussi ...
)DA0B-A8       TAY
)DA0C-C8       INY
)DA0D-C0 0E    CPY #$0E         et 14 données
)DA0F-D0 E8    BNE $D9F9
)DA11-28       PLP              on sort C
)DA12-68       PLA
)DA13-85 15    STA $15          on récupère $15-16
)DA15-68       PLA
)DA16-85 16    STA $16
)DA18-28       PLP              et P
)DA19-60       RTS
```

ENVOIE UNE DONNEE AU PSG

```
'rincipe:Place la valeur X dans le registre A du PSG AY3-8912.
         Les broches Bdir et BC1 du PSG sont reliées respectivement aux broches
         CB2 et CA2. Pour indiquer un registre, on met CA2 et CB2 à 1, une
         donnée, CA2 à 1 et CB2 à 0. La validation se fait avec CB2 et CA2 à 0.
         Cette simplicité est due au fait que sur l'ORIC, la broche BC2 du PSG
         est forcée à l'état haut.

)A1A-48        PHA              on sauve le numéro de registre
)A1B-8D 0F 03  STA $030F        sur V1DRAB
)A1E-C9 07     CMP #$07         registre d'autorisation ?
)A20-D0 04     BNE $DA26        non ----------------------------------------------
)A22-8A        TXA              oui, on force                                     I
)A23-09 40     ORA #$40         le port A du PSG en sortie                        I
)A25-AA        TAX                                                                I
)A26-98        TYA              registre dans A <--------------------------------
)A27-48        PHA              dans la pile
)A28-08        PHP
)A29-78        SEI              interdit les IRQ
)A2A-AD 0C 03  LDA $030C        V1PCR
)A2D-29 11     AND #$11         sans changer les modes de transition
)A2F-A8        TAY              dans Y
)A30-09 EE     ORA #$EE         on force CA2 et CB2 à 1
)A32-8D 0C 03  STA $030C        pour indiquer le registre
)A35-98        TYA              on récupère V1PCR
)A36-09 CC     ORA #$CC         CA2 et CB2 à 0
)A38-8D 0C 03  STA $030C        pour valider le registre
)A3B-8E 0F 03  STA $030F        on met la valeur sur V1DRAB
)A3E-98        TYA              on récupère V1PCR
)A3F-09 EC     ORA #$EC         CA2 à 1 et CB2 à 0
)A41-8D 0C 03  STA $030C        pour indiquer donnée
```

```
DA44-98          TYA
DA45-09 CC       ORA #$CC          CA2 et CB2 à 0
DA47-8D 0C 03    STA $030C         pour valider la donnée
DA4A-28          PLP               on sort P
DA4B-68          PLA               Y
DA4C-A8          TAY
DA4D-68          PLA               et A
DA4E-60          RTS
```

COUPE LE SON DU PSG

```
DA4F-A9 07       LDA #$07          registre 7 (autorisation)
DA51-A2 7F       LDX #$7F          %01111111, tous canaux éteints
DA53-4C 1A DA    JMP $DA1A         au PSG
```

INITIALISE L'IMPRIMANTE

Principe:Rien de bien compliqué, si ce n'est que le vecteur HCOPY est $250 et non $24E comme indiqué dans le manuel développeur.

```
DA56-A9 50       LDA #$50          80 colonnes
DA58-8D 88 02    STA $0288         dans LPRFX
DA5B-A9 00       LDA #$00          ligne 0
DA5D-8D 86 02    STA $0286         dans LPRX
DA60-A9 80       LDA #$80          imprimante prête
DA62-8D 8A 02    STA $028A         dans FLGLPR
DA65-A9 53       LDA #$53          $E253 (HCOPY)
DA67-A0 E2       LDY #$E2
DA69-8D 50 02    STA $0250         dans $250-$251
DA6C-8C 51 02    STY $0251
DA6F-60          RTS
```

GERE LA SORTIE IMPRIMANTE

Principe:L'E/S est gérée comme les autres. Les imprimantes série sont prises en compte et il n'existe pas de fermeture d'imprimante...
En ouverture, les IRQ par ACK sont autorisées pour determiner quand l'imprimante est prête à recevoir une donnée.

```
DA70-30 60       BMI $DAD2         ouverture ou fermeture ----------------------------
DA72-48          PHA               écriture, on sauve les registres                  I
DA73-8A          TXA               A et X                                            I
DA74-48          PHA                                                                 I
DA75-A9 82       LDA #$82          on autorise les IRQ par CA1 (ACK)                 I
DA77-8D 0E 03    STA $030E                                                           I
DA7A-BA          TSX                                                                 I
DA7B-BD 02 01    LDA $0102,X       on prend la donnée                                I
DA7E-20 A5 DA    JSR $DAA5         on l'envoie                                       I
DA81-2C 8A 02    BIT $028A         gérer CRLF ?                                      I
DA84-70 1B       BVS $DAA1         non ---------------------------------------------
DA86-C9 20       CMP #$20          oui, la donnée est un espace ?                    I
DA88-B0 06       BCS $DA90      ---au dessus, on passe                               I
DA8A-C9 0D       CMP #$0D       I  un RETURN ?                                       I
DA8C-D0 13       BNE $DAA1      I  non ----------------------------------------------0
DA8E-F0 0C       BEQ $DA9C      I  oui, on reprend à 0 ----------------------------   I
DA90-AE 86 02    LDX $0286      -->on ajoute 1 à l'abscisse                  I     I   I
DA93-E8          INX                                                        I     I   I
DA94-EC 88 02    CPX $0288         on est au bout ?                        I     I   I
DA97-90 05       BCC $DA9E      ---non                                     I     I   I
DA99-20 E4 DA    JSR $DAE4      I  oui, on saute une ligne                 I     I   I
```

```
)A9C-A2 00     LDX #$00    I  et on met 0 <-------------------------  I  I
)A9E-8E 86 02  STX $0286   -->dans la position du stylo               I  I
)AA1-68        PLA            <-------------------------------------  I
)AA2-AA        TAX            on restaure les registres et on sort    I
)AA3-68        PLA                                                    I
)AA4-60        RTS                                                    I
)AA5-AA        TAX            Donnée dans X                           I
)AA6-AD 8A 02  LDA $028A      mode RS232 ?                            I
)AA9-29 04     AND #$04                                               I
)AAB-F0 08     BEQ $DAB5      non, parrallèle ------------------------ I
)AAD-20 2F DB  JSR $DB2F      oui, on prépare la RS232                I  I
)AB0-8A        TXA            donnée dans A                           I  I
)AB1-A2 18     LDX #$18       X=buffer ACIA sortie                    I  I
)AB3-D0 0A     BNE $DABF    --inconditionnel                          I  I
)AB5-AD 0D 02  LDA $020D    I imprimante détectée ? <-------------------- I
)AB8-29 02     AND #$02     I                                           I
)ABA-F0 15     BEQ $DAD1    I non, on sort                              I
)ABC-8A        TXA          I oui, donnée dans A                        I
)ABD-A2 24     LDX #$24     I X=buffer CENTRONICS sortie                I
)ABF-2C 8A 02  BIT $028A    ->gérer CRLF ?                              I
)AC2-70 06     BVS $DACA      non ------------------------------------  I
)AC4-C9 7F     CMP #$7F       oui, A=DEL (127) ?                  I     I
)AC6-D0 02     BNE $DACA      non ------------------------------------0  I
)AC8-A9 20     LDA #$20       oui, on envoie un espace            I     I
)ACA-48        PHA            on sauve la donnée (inutile !!!) <-------  I
)ACB-20 1D C5  JSR $C51D      on l'écrit                                  I
)ACE-68        PLA            on la récupère (inutile ! la routine la restitue)I
)ACF-B0 EE     BCS $DABF      si l'écriture n'a pas eu lieu, on boucle    I
)AD1-60        RTS                                                        I
)AD2-B0 0C     BCS $DAE0      fermeture (rien en fait) <-------------------
)AD4-AD 8A 02  LDA $028A      ouverture, imprimante série ?
)AD7-29 04     AND #$04
)AD9-D0 06     BNE $DAE1      oui ---------------------------------
)ADB-A9 82     LDA #$82     \ on autorise les IRQ par CA1       I
)ADD-8D 0E 03  STA $030E      pour le ACK                       I
)AE0-60        RTS                                              I
)AE1-4C 87 DB  JMP $DB87      <---------------------------------
```

ENVOIE UN CR,LF SUR IMPRIMANTE

```
)AE4-48        PHA            on sauve A
)AE5-A9 0D     LDA #$0D       on envoie un CR
)AE7-20 72 DA  JSR $DA72
)AEA-AD 8A 02  LDA $028A      si pas LF après CR
)AED-4A        LSR
)AEE-B0 05     BCS $DAF5    ---on passe
)AF0-A9 0A     LDA #$0A     I  sinon, on envoie
)AF2-20 72 DA  JSR $DA72    I  un LF
)AF5-68        PLA          I->on récupère A
)AF6-60        RTS
```

GESTION DE L'ENTREE MINITEL

```
)AF7-30 05     BMI $DAFE      ouverture-fermeture ---------------------
)AF9-A2 0C     LDX #$0C       lecture : on indexe le buffer ACIA entrée  I
)AFB-4C 18 C5  JMP $C518      et on lit un code                          <
)AFE-B0 09     BCS $DB09      C=1, on ferme ==============================
)B00-AD 1E 03  LDA $031E      ouverture:on lit ACIACR                    >
)B03-29 0D     AND #$0D       %00001101 isole b0 b2 et b3                I
```

```
DB05-09 60      ORA #$60        %01100000 force b6b5 à 1
DB07-D0 3A      BNE $DB43       inconditionnel ----------------------------------
DB09-AD 1E 03   LDA $031E       <-------------------------------------------------+--
DB0C-09 02      ORA #$02        force b1 à 1 dans                                  I
DB0E-8D 1E 03   STA $031E       ACIACR                                             I
DB11-60         RTS                                                                --
```

GESTION DE LA SORTIE MINITEL

Principe:N'étant guère familiarisé avec les modes de fonctionnement de l'ACIA, je vous donne le source mais je serais bien incapable d'expliciter les modifications d'ACIACR et d'ACIACT, registres de controle et de commande de l'ACIA 6551.

```
DB12-30 26      BMI $DB3A       ouverture-fermeture ----------------------------
DB14-AA         TAX             donnée dans X                                      I
DB15-10 0F      BPL $DB26       si <128, on passe ------------------------         I
DB17-C9 C0      CMP #$C0        sinon, c'est <128+64 ?                  I          I
DB19-B0 0B      BCS $DB26                                               I          I
DB1B-09 40      ORA #$40        oui, on force b7                        I          I
DB1D-48         PHA                                                     I          I
DB1E-A9 1B      LDA #$1B        et on envoie un ESC avant               I          I
DB20-A2 18      LDX #$18        la donnée                               I          I
DB22-20 1D C5   JSR $C51D                                               I          I
DB25-68         PLA                                                     I          I
DB26-48         PHA             <---------------------------------------          I
DB27-A2 18      LDX #$18        on envoie la donnée                                I
DB29-20 1D C5   JSR $C51D       dans le BUFFER ACIA sortie                         I
DB2C-68         PLA                                                                I
DB2D-B0 F7      BCS $DB26       si la donnée n'a pas été écrite, on boucle         I
DB2F-AD 1E 03   LDA $031E       on prend V2IER                                     I
DB32-29 F3      AND #$F3        %11110011 force b2 à 0                             I
DB34-09 04      ORA #$04        et b3 à 1                                          I
DB36-8D 1E 03   STA $031E       dans ACIACR                                        I
DB39-60         RTS                                                                <
DB3A-B0 17      BCS $DB53       C=1 on ferme ==================================    =
DB3C-AD 1E 03   LDA $031E       ouverture                                          >
DB3F-29 02      AND #$02        ACIACR à 0 sauf b1                                 I
DB41-09 65      ORA #$65        %01101001, bits forcés à 1                         I
DB43-8D 1E 03   STA $031E       dans ACIACR <-------------------------------------+--
DB46-AD 21 03   LDA $0321       V2DRA                                              I
DB49-29 EF      AND #$EF        %11101111 force mode MINITEL                       I
DB4B-8D 21 03   STA $0321                                                          I
DB4E-A9 38      LDA #$38        %00111000 dans ACIACT                              I
DB50-8D 1F 03   STA $031F                                                          I
DB53-60         RTS             et on sort----------------------------------------
```

INITIALISE LES VALEURS RS232

```
DB54-A9 1E      LDA #$1E        %00011110 transmission 8 bits,horloge interne,
DB56-85 59      STA $59         1 bit de stop, 9600 bauds dans RS232T
DB58-A9 00      LDA #$00        pas d'écho et pas de parité
DB5A-85 5A      STA $5A         dans RS232C
DB5C-60         RTS
```

GESTION DE L'ENTREE RS232

```
DB5D-10 98      BPL $DAF7       lecture, voir MINITEL (pourquoi pas $DAF9 ?)
```

```
DB5F-B0 A8      BCS $DB09       C=1, on ferme
DB61-AD 1E 03   LDA $031E       on ouvre
DB64-29 0D      AND #$0D        on fixe le mode de controle
DB66-05 5A      ORA $5A         de la RS232
DB68-8D 1E 03   STA $031E
DB6B-AD 21 03   LDA $0321
DB6E-09 10      ORA #$10        %00010000 on force RS232
DB70-8D 21 03   STA $0321
DB73-A5 59      LDA $59         et on fixe le mode de transmission
DB75-8D 1F 03   STA $031F       dans ACIACR
DB78-60         RTS
```

GESTION DE LA SORTIE RS232

```
DB79-10 AB      BPL $DB26       écriture, comme MINITEL
DB7B-B0 D6      BCS $DB53       pas de fermeture (RTS)
DB7D-AD 1E 03   LDA $031E       ouverture,on lit ACIACR
DB80-29 02      AND #$02        isole b1
DB82-09 05      ORA #$05        %00000101 force b0 et b2 à 1
DB84-D0 E0      BNE $DB66       inconditionnel
```

GESTION DES SORTIES EN MODE TEXT

Principe:tellement habituel que ça en devient monotone... mais bien pratique !

```
DB86-48         PHA             on sauve A et P
DB87-08         PHP
DB88-A9 00      LDA #$00        fenêtre 0
DB8A-F0 10      BEQ $DB9C       inconditionnel

DB8C-48         PHA
DB8D-08         PHP
DB8E-A9 01      LDA #$01        fenêtre 1
DB90-D0 0A      BNE $DB9C

DB92-48         PHA
DB93-08         PHP
DB94-A9 02      LDA #$02        fenêtre 2
DB96-D0 04      BNE $DB9C

DB98-48         PHA
DB99-08         PHP
DB9A-A9 03      LDA #$03        fenêtre 3

DB9C-85 28      STA $28         stocke la fenêtre dans SCRNB
DB9E-28         PLP             on lit la commande
DB9F-10 03      BPL $DBA4       écriture -------
DBA1-4C CE DE   JMP $DECE       ouverture      I
DBA4-68         PLA             on lit la donnée <
DBA5-85 29      STA $29         que l'on sauve
DBA7-AD 8A 02   LDA $028A       écho sur imprimante ?
DBAA-29 02      AND #$02
DBAC-F0 05      BEQ $DBB3       non ----------------------------------------
DBAE-A5 29      LDA $29         oui, on envoie le code sur imprimante    I
DBB0-20 72 DA   JSR $DA72                                                I
DBB3-A5 29      LDA $29         <----------------------------------------
```

AFFICHE CODE A DANS FENETRE X

Principe:On lit le code, si c'est un caractère, on l'affiche simplement. Si c'est un caractère de controle, alors là c'est la folie !
On empile l'adresse de retour général des codes de controle (il remettre le curseur, ajuster les coordonnées et dépiler les registres), puis l'adresse de gestion dudit code de controle. Un RTS bien placé suffira à exécuter cette routine, routine dont le RTS terminera proprement l'affichage (ou la gestion) dudit code. ouf !

```
DBB5-85 29      STA $29
DBB7-48         PHA             on sauve les registres
DBB8-8A         TXA
DBB9-48         PHA
DBBA-98         TYA
DBBB-48         PHA
DBBC-A6 28      LDX $28         on lit la fenêtre
DBBE-BD 18 02   LDA $0218,X     on stocke l'adresse de la fenêtre
DBC1-85 26      STA $26         dans ADSCR
DBC3-BD 1C 02   LDA $021C,X
DBC6-85 27      STA $27
DBC8-A5 29      LDA $29         on prend le code
DBCA-C9 20      CMP #$20        espace ?
DBCC-B0 7E      BCS $DC4C       caractère
DBCE-BD 48 02   LDA $0248,X     code de controle, on prend FLGSCR
DBD1-48         PHA             dans la pile
DBD2-20 1E DE   JSR $DE1E       on éteint le curseur
DBD5-A9 DC      LDA #$DC        on empile la fin de gestion
DBD7-48         PHA             soit $DC2B-1 car on s'y branche par RTS
DBD8-A9 2A      LDA #$2A
DBDA-48         PHA
DBDB-A5 29      LDA $29         on lit le code
DBDD-0A         ASL
DBDE-A8         TAY
DBDF-B9 EC DB   LDA $DBEC,Y     on lit le poids fort
DBE2-48         PHA             dans la pile
DBE3-B9 EB DB   LDA $DBEB,Y     et le poids faible
DBE6-48         PHA             dans la pile de l'adresse de gestion du code
DBE7-A9 00      LDA #$00        A=0
DBE9-38         SEC             C=1
DBEA-60         RTS             on gère le code
```

TABLE DE GESTION DES CODES DE CONTOLE

Principe:Chaque code de controle est géré par une routine particulière que l'on appelle par RTS. Les adresses stockées ici doievent donc être incrémentées pour donner l'adresse exacte de gestion du code.
On calcule l'adresse de gestion d'un code par DEEK($DBE9+2*code)+1 .

```
DBEB    BYT $DCEA-1     code 0          non géré
DBED    BYT $DCEB-1          1  CTRL-A   Tabulation
DBEF    BYT $DCEA-1          2  CTRL-B   non géré
DBF1    BYT $DCEA-1          3  CTRL-C   déja géré (durant le codage)
DBF3    BYT $DD0D-1          4  CTRL-D   double hauteur
DBF5    BYT $DCEA-1          5  CTRL-E   non géré
DBF7    BYT $DCEA-1          6  CTRL-F   déja géré
DBF9    BYT $DDD8-1          7  CTRL-G   émet un OUPS
DBFB    BYT $DD47-1          8  CTRL-H   déplacement curseur à gauche
DBFD    BYT $DD92-1          9  CTRL-I   déplacement curseur à droite
```

```
DBFF            BYT $DD9D-1         10 CTRL-J   déplacement curseur vers le bas
DC01            BYT $DD55-1         11 CTRL-K   déplacement curseur vers le haut
DC03            BYT $DDB8-1         12 CTRL-L   efface l'écran
DC05            BYT $DD67-1         13 CTRL-M   saut de ligne
DC07            BYT $DD74-1         14 CTRL-N   efface une ligne
DC09            BYT $DCEA-1         15 CTRL-O   non géré
DC0B            BYT $DD12-1         16 CTRL-P   curseur fixe/clignotant
DC0D            BYT $DD13-1         17 CTRL-Q   curseur absent/présent
DC0F            BYT $DDCF-1         18 CTRL-R   écho imprimante
DC11            BYT $DDCC-1         19 CTRL-S   TCOPY (pourquoi pas $E1B9-1 ?!)
DC13            BYT $DCEA-1         20 CTRL-T   déja géré
DC15            BYT $DCEA-1         21 CTRL-U   non géré
DC17            BYT $DD11-1         22 CTRL-V   vidéo inverse/normale
DC19            BYT $DCEA-1         23 CTRL-W   déja géré
DC1B            BYT $DD7A-1         24 CTRL-X   effacement de fin de ligne
DC1D            BYT $DCEA-1         25 CTRL-Y   non géré
DC1F            BYT $DCEA-1         26 CTRL-Z   non géré
DC21            BYT $DD0F-1         27 ESC      codes écran
DC23            BYT $DD0F-1         28 ???      comme ESC
DC25            BYT $DD10-1         29 CTRL-$   (crochet fermé) 38/40 colonnes
DC27            BYT $DDF8-1         30 HOME     ramène le curseur en 0,0
DC29            BYT $DD0E-1         31 US       adressage direct du curseur
```

SORTIE DE GESTION DES CODES DE CONTROLE

```
DC2B-A6 28      LDX $28         on prend la fenêtre dans X
DC2D-BC 20 02   LDY $0220,X     position colonne
DC30-B1 26      LDA ($26),Y     on lit le code sous le curseur
DC32-9D 4C 02   STA $024C,X     et on le place dans CURSCR
DC35-A5 26      LDA $26         on stocke l'adresse de la fenêtre X
DC37-9D 18 02   STA $0218,X     dans ADSCRL
DC3A-A5 27      LDA $27
DC3C-9D 1C 02   STA $021C,X     ADSCRH
DC3F-68         PLA             on récupère FLGSCR
DC40-9D 48 02   STA $0248,X     que l'on replace
DC43-20 2D DE   JSR $DE2D       on inverse le curseur ($DE30 aurait été mieux !)
DC46-68         PLA             on restaure les registres
DC47-A8         TAY
DC48-68         PLA
DC49-AA         TAX
DC4A-68         PLA
DC4B-60         RTS             et on sort définitivement
```

AFFICHAGE D'UN CARACTERE

```
DC4C-BD 48 02   LDA $0248,X     on prend le flag de la fenêtre X
DC4F-29 0C      AND #$0C        %00001100, ESC ou US en cours ?
DC51-D0 47      BNE $DC9A       oui ----------------------------------------------
DC53-A5 29      LDA $29         non, on prend le code                            I
DC55-10 06      BPL $DC5D       caractère normal ----------------                I
DC57-C9 A0      CMP #$A0        <128+32 ?                        I               I
DC59-B0 02      BCS $DC5D       non ------------------------------0               I
DC5B-29 7F      AND #$7F        oui, on en fait un code simple   I               I
DC5D-85 29      STA $29         dans $29 <-------------------------              I
DC5F-20 6B DC   JSR $DC6B       on affiche le code                               I
DC62-A9 09      LDA #$09        puis on déplace le curseur vers la droite        I
DC64-85 29      STA $29                                                          I
DC66-4C CE DB   JMP $DBCE                                                        I
                                                                                 I
                                                                                 I
```

PLACEMENT D'UN CODE A L'ECRAN

Principe:Trivial dans son ensemble puisqu'il gère les séquences ESC et US séquentiellement. Ca se complique lorsque des appels récursifs croisés entrent en jeu. Cette routine fait au moins preuve d'une grande optimisation. En entrée, X contient le numéro de la fenêtre et A le code à afficher.

```
DC69-85 29      STA $29         on sauve le code
DC6B-A0 80      LDY #$80        Y=128
DC6D-BD 48 02   LDA $0248,X     on lit FLGSCR
DC70-29 20      AND #$20        vidéo inverse ?
DC72-D0 02      BNE $DC76       oui, on laisse Y ----------------------------
DC74-A0 00      LDY #$00        non, Y=0                                     I
DC76-98         TYA             dans A <-------------------------------------
DC77-05 29      ORA $29         on superpose le code
DC79-9D 4C 02   STA $024C,X     dans CURSCR
DC7C-BC 20 02   LDY $0220,X     et sur l'écran
DC7F-91 26      STA ($26),Y
DC81-BD 48 02   LDA $0248,X
DC84-29 02      AND #$02        double hauteur ?
DC86-F0 11      BEQ $DC99       non -----------------------------------------
DC88-BD 24 02   LDA $0224,X     oui                                          I
DC8B-DD 34 02   CMP $0234,X     on est  à la fin de la fenêtre ?             I
DC8E-F0 09      BEQ $DC99       oui ------------------------------------------Ù
DC90-98         TYA             non, on ajoute 40 colonnes (1 ligne)         I
DC91-69 28      ADC #$28                                                     I
DC93-A8         TAY                                                          I
DC94-BD 4C 02   LDA $024C,X                                                  I
DC97-91 26      STA ($26),Y     et on affiche encore le code                 I
DC99-60         RTS             <-------------------------------------------
DC9A-29 08      AND #$08        est-ce ESC ? <-------------------------------
DC9C-F0 1A      BEQ $DCB8       non, US -------------------------------------
DC9E-A5 29      LDA $29         oui, on lit le code                          I
DCA0-30 A4      BMI $DC46       >128, on sort                                I
DCA2-C9 40      CMP #$40        <64 ("A"-1) ?                                I
DCA4-90 A0      BCC $DC46       oui, on sort                                 I
DCA6-29 1F      AND #$1F        sinon, on isole le code (0-31)               I
DCA8-20 69 DC   JSR $DC69       on le place à l'écran                        I
DCAB-A9 09      LDA #$09        on déplace le curseur à droite               I
DCAD-20 B5 DB   JSR $DBB5                                                    I
DCB0-A9 1B      LDA #$1B        on envoie un ESC (fin de ESC)                I
DCB2-20 B5 DB   JSR $DBB5                                                    I
DCB5-4C 46 DC   JMP $DC46       et on sort                                   I
DCB8-BD 48 02   LDA $0248,X     US, on lit FLGSCR <--------------------------
DCBB-48         PHA             que l'on sauve
DCBC-20 1E DE   JSR $DE1E       on éteint le curseur
DCBF-68         PLA             on prend FLGSCR
DCC0-48         PHA
DCC1-4A         LSR             doit-on envoyer Y ou X ?
DCC2-B0 18      BCS $DCDC       X -------------------------------------------
DCC4-A5 29      LDA $29         on lit Y                                     I
DCC6-29 3F      AND #$3F        on vire b4 (protocole US)                    I
DCC8-9D 24 02   STA $0224,X     et on fixe Y                                 I
DCCB-20 07 DE   JSR $DE07       on ajuste l'adresse dans la fenêtre          I
DCCE-9D 18 02   STA $0218,X     dans ADSCRL                                  I
DCD1-98         TYA                                                          I
DCD2-9D 1C 02   STA $021C,X     et ADSCRH                                    I
DCD5-68         PLA             on indique prochain code pour X              I
```

```
)CD6-09 01      ORA #$01                                                    I
)CD8-48         PHA                                                         I
)CD9-4C 2B DC   JMP $DC2B       et on sort                                  I
)CDC-A5 29      LDA $29         on lit X <-----------------------------------
)CDE-29 3F      AND #$3F        on vire b4
)CE0-9D 20 02   STA $0220,X     dans SCRX
)CE3-68         PLA
)CE4-29 FA      AND #$FA        on indique fin de US
)CE6-48         PHA
)CE7-4C 2B DC   JMP $DC2B       et on sort
)CEA-60         RTS
```

GESTION DES CODES DE CONTROLE

Principe:Génial... la gestion des codes de controle est surement la partie la plus caractéristique de l'esprit BROCHIEN (après le BRK bien sur). La gestion est suprêmement optimisée pour tout les codes, elle est surement le fruit d'une mure reflexion. Chapeau.
En entrée de chaque routine, A=0, C=1 et la pile contient en son sommet -3, FLGSCR. Le RTS branche en fait en $DC2B, routine générale de fin de gestion de code de controle.

CODE 01 - CTRL A

Action:Place le curseur sur une tabulation, colonne multiple de 8.

```
)CEB-BD 20 02   LDA $0220,X  ->on lit ala colonne
)CEE-29 F8      AND #$F8     I on la met à 0
)CF0-69 07      ADC #$07     I on place sur une tabulation 8 (C=1)
)CF2-DD 2C 02   CMP $022C,X  I est-on en fin de ligne ?
)CF5-F0 12      BEQ $DD09    I non
)CF7-90 10      BCC $DD09    I ---------------------------------------------
)CF9-20 67 DD   JSR $DD67    I oui, on ramène le curseur en début de ligne  I
)CFC-20 9D DD   JSR $DD9D    I et on passe une ligne                        I
)CFF-A6 28      LDX $28      I                                              I
)D01-BD 20 02   LDA $0220,X  I on prend la colonne                          I
)D04-29 07      AND #$07     I est-on sur une tabulation                    I
)D06-D0 E3      BNE $DCEB    --non, on tabule...                            I
)D08-60         RTS                                                         I
)D09-9D 20 02   STA $0220,X    on sauve la colonne <-------------------------
)D0C-60         RTS            et on sort
```

CODE 4 - CTRL D

```
)D0D-6A         ROR            on prépare masque %00000010
```

CODE 31 - US

```
                               on prépare masque %00000100
)D0E-6A         ROR
```

CODE 27 - ESC

```
                               on prépare masque %00001000
)D0F-6A         ROR
```

CODE 29 - CTRL §

```
                               on prépare masque %00010000
)D10-6A         ROR
```

CODE 22 - CTRL V

```
DD11-6A          ROR              on prépare masque %00100000
```

CODE 16 - CTRL P

```
DD12-6A          ROR              on prépare masque %01000000
```

CODE 17 - CTRL Q

```
DD13-6A          ROR              on prépare masque %10000000

DD14-A8          TAY              dans Y
DD15-BA          TSX              on indexe FLGSCR dans la pile
DD16-5D 03 01    EOR $0103,X      on inverse le bit correspondant au code (bascule)
DD19-9D 03 01    STA $0103,X      et on replace
DD1C-85 00       STA $00          et dans $00
DD1E-98          TYA
DD1F-29 10       AND #$10         mode 38/40 colonne ?
DD21-D0 01       BNE $DD24        oui ----------------------------------------------
DD23-60          RTS              non on sort                                       I
DD24-A6 28       LDX $28          on prend le numéro de fenêtre <-------------------
DD26-25 00       AND $00          mode monochrome (ou 40 colonnes) ?
DD28-F0 12       BEQ $DD3C        oui ----------------------------------------------
DD2A-FE 28 02    INC $0228,X      non, on interdit la première colonne              I
DD2D-FE 28 02    INC $0228,X      et la deuxième                                    I
DD30-BD 20 02    LDA $0220,X      est-on dans une colonne                           I
DD33-DD 28 02    CMP $0228,X      interdite ?                                       I
DD36-B0 03       BCS $DD3B  ---non                                                  I
DD38-4C 67 DD    JMP $DD67  I  oui,on en sort                                       I
DD3B-60          RTS     <---                                                      I
DD3C-DE 28 02    DEC $0228,X      on autorise colonne 0 et 1 <----------------------
DD3F-DE 28 02    DEC $0228,X
DD42-60          RTS

DD43-DE 20 02    DEC $0220,X      on ramène le curseur un cran à gauche  <----------
DD46-60          RTS                                                               I
```

CODE 8 - CTRL H

```
Action:déplace le curseur vers la gauche

DD47-BD 20 02    LDA $0220,X      est-on déja au début de la fenêtre ?             I
DD4A-DD 28 02    CMP $0228,X                                                       I
DD4D-D0 F4       BNE $DD43        non, on ramène à gauche ----------------------------
DD4F-BD 2C 02    LDA $022C,X      oui, on se place à la fin de la fenêtre
DD52-9D 20 02    STA $0220,X
```

CODE 11 - CTRL K

```
Action:déplace le curseur vers le haut

DD55-BD 24 02    LDA $0224,X      et si on est pas
DD58-DD 30 02    CMP $0230,X      au sommet de la fenêtre,
DD5B-D0 11       BNE $DD6E        on remonte d'une ligne ---------------------------
DD5D-BD 30 02    LDA $0230,X      X et Y contiennent le début et la                 I
DD60-BC 34 02    LDY $0234,X      fin de la fenêtre X                               I
DD63-AA          TAX                                                               I
DD64-20 5C DE    JSR $DE5C        on scrolle l'écran vers le bas ligne X à Y       I
DD67-BD 28 02    LDA $0228,X      on place début de la fenêtre dans X              I
DD6A-9D 20 02    STA $0220,X                                                       I
DD6D-60          RTS                                                               I
```

```
)D6E-DE 24 02  DEC $0224,X     on remontre le curseur <----------------------
)D71-4C 07 DE  JMP $DE07       et on ajuste ADSCR
```

CODE 14 - CTRL N

```
ction:efface la ligne courante

)D74-BC 28 02  LDY $0228,X     on prend la première colonne de la fenetre
)D77-4C 7D DD  JMP $DD7D       et on efface ce qui suit (BPL aurait été mieux...)
```

CODE 24 - CTRL X

```
ction:efface la fin de la ligne courante

'D7A-BC 20 02  LDY $0220,X     on prend la colonne du curseur
)D7D-BD 2C 02  LDA $022C,X     et la dernière colonne de la fenêtre
)D80-85 29     STA $29         dans $29
:D82-A9 20     LDA #$20        on envoie un espace
:D84-91 26     STA ($26),Y
:D86-C8        INY             jusqu'à la fin de la ligne
:D87-C4 29     CPY $29
:D89-90 F9     BCC $DD84
:D8B-91 26     STA ($26),Y     et à la dernière position aussi
:D8D-60        RTS             (INC $29 avant la boucle aurait été mieux !)

 D8E-FE 20 02  INC $0220,X
 D91-60        RTS
```

CODE 9 - CTRL I

```
ction:déplace le curseur à droite

:D92-BD 20 02  LDA $0220,X     on lit la colonne du curseur
:D95-DD 2C 02  CMP $022C,X     dernière colonne ?
:D98-D0 F4     BNE $DD8E       non, on déplace le curseur
:D9A-20 67 DD  JSR $DD67       oui, on revient à la première colonne
```

CODE 10 - CTRL J

```
ction:déplace le curseur vers la droite

:D9D-BD 24 02  LDA $0224,X     on est en bas de la fenêtre ?
:DA0-DD 34 02  CMP $0234,X
:DA3-D0 0D     BNE $DDB2       non ------------------------------------------
:DA5-BD 30 02  LDA $0230,X     oui, X et Y contiennent début et fin de fenêtre I
:DA8-BC 34 02  LDY $0234,X                                                     I
:DAB-AA        TAX                                                             I
:DAC-20 54 DE  JSR $DE54       on scrolle la fenêtre                           I
:DAF-4C 67 DD  JSR $DD67       on revient en début de ligne                    I
:DB2-FE 24 02  INC $0224,X     on incrémente la ligne <-----------------------I
:DB5-4C 07 DE  JMP $DE07       et on ajuste ADSCR
```

CODE 12 - CTRL L

```
ction:Efface la fenêtre

:DB8-20 FB DD  JSR $DDFB       on remet le curseur en haut de la fenêtre
:DBB-20 74 DD  JSR $DD74       on efface la ligne courante
:DBE-BD 24 02  LDA $0224,X     on est à la fin de la fenêtre
```

```
)DC1-DD 34 02   CMP $0234,X
)DC4-F0 35      BEQ $DDFB       oui, on sort en replaçant le curseur en haut
)DC6-20 9D DD   JSR $DD9D       non, on déplace le curseur vers le bas
)DC9-4C BB DD   JMP $DDBB       et on boucle (Et BPL, non ?!?!)
```

CODE 19 - CTRL S

```
)DCC-4C B9 E1   JMP $E1B9       on exécute un †COPY
```

CODE 18 - CTRL R

Action:bascule l'écho imprimante du clavier

```
)DCF-A9 02      LDA #$02        on inverse b1
)DD1-4D 8A 02   EOR $028A       de FLGLPR
)DD4-8D 8A 02   STA $028A
)DD7-60         RTS
```

CODE 7 - CTRL G

Action:émet un OUPS

```
)DD8-A2 F0      LDX #$F0        on indexe les 14 données du OUPS
)DDA-A0 DD      LDY #$DD
)DDC-20 E7 D9   JSR $D9E7       et on envoie au PSG
)DDF-A0 60      LDY #$60        I
)DE1-A2 00      LDX #$00        I
)DE3-CA         DEX             I Délai d'une seconde
)DE4-D0 FD      BNE $DDE3       I
)DE6-88         DEY             I
)DE7-D0 FA      BNE $DDE3       I
)DE9-A9 07      LDA #$07        un JMP $DA4F suffisait ...
)DEB-A2 3F      LDX #$3F
)DED-4C 1A DA   JMP $DA1A

)DF0            BYT $46,00,00,00,00,00  période 1,12 ms, fréquence 880 Hz (LA 4)
)DF6            BYT 00,$3E,$0F,00,00    canal 1, volume 15 musical
```

INITIALISE UNE FENETRE

Action:on place le curseur en (0,0) et on calcule son adresse

```
)DFB-BD 28 02   LDA $0228,X     on prend la première colonne
)DFE-9D 20 02   STA $0220,X     dans SCRX
)E01-BD 30 02   LDA $0230,X     la première ligne dans
)E04-9D 24 02   STA $0224,X     SCRY
)E07-BD 24 02   LDA $0224,X     et on calcule l'adresse
)E0A-20 12 DE   JSR $DE12       de la ligne
)E0D-85 26      STA $26         dans ADSCR
)E0F-84 27      STY $27
)E11-60         RTS
```

CALCULE L'ADRESSE DE LA LIGNE A

Action:En entrée, A contient le numéro de la ligne et en sortie, RES contient
l'adresse à l'écran de cette ligne.

```
DE12-20 69 CE   JSR $CE69       RES=A*40
DE15-BD 38 02   LDA $0238,X     AY=adresse de la fenêtre
DE18-BC 3C 02   LDY $023C,X
DE1B-4C 89 CE   JMP $CE89       on calcule dans RES l'adresse de la ligne
```

ETEINDRE LE CURSEUR DANS UNE FENETRE

Action:Est censée eteindre le curseur dans une fenêtre, mais la gestion du des curseurs par fenêtre est buggée. On n'aura donc de curseur que dans la fenêtre 0. Dommage, c'était une bonne idée au demeurant...

```
DE1E-18         CLC             C=0
DE1F            BYT $24         on passe l'instruction suivante
```

ALLUMER LE CURSEUR DANS UNE FENETRE

Action:voir plus haut en remplaçant "éteindre" par "allumer"...

```
DE20-38         SEC             C=1
DE21-08         PHP             on sauve C
DE22-1E 48 02   ASL $0248,X     on décale FLGSCR
DE25-28         PLP             on reprend C
DE26-7E 48 02   ROR $0248,X     et on place le bit curseur dans FLGSCR
DE29-30 28      BMI $DE53       on à allumé le curseur, on passe ----------
DE2B-A9 80      LDA #$80        on prend A=%10000000                      I
                                                                          I
```

INVERSE LE CURSEUR

```
Action:Voir plus haut en remplaçant "allumer" par "inverser"...           I
                                                                          I
DE2D-3D 48 02   AND $0248,X     curseur à afficher ?                      I
DE30-29 80      AND #$80        prend juste b7                            I
DE32-5D 4C 02   EOR $024C,X     et on inverse l'état curseur du code concerné I
DE35-BC 20 02   LDY $0220,X                                               I
DE38-91 26      STA ($26),Y     et on place le code                       I
DE3A-48         PHA             on sauve le code                          I
DE3B-BD 48 02   LDA $0248,X                                               I
DE3E-29 02      AND #$02        double hauteur ?                          I
DE40-F0 10      BEQ $DE52       non ----------------------------------------- I
DE42-BD 24 02   LDA $0224,X     oui, si on n'est pas                    I I
DE45-DD 34 02   CMP $0234,X     en bas de la fenêtre                    I I
DE48-F0 08      BEQ $DE52       ----------------------------------------0 I
DE4A-98         TYA             on ajoute une ligne                     I I
DE4B-69 28      ADC #$28                                                I I
DE4D-A8         TAY                                                     I I
DE4E-68         PLA                                                     I I
DE4F-91 26      STA ($26),Y     et on affiche le code une deuxième fois I I
DE51-60         RTS                                                     I I
DE52-68         PLA             <---------------------------------------- I
DE53-60         RTS             <-----------------------------------------
```

SCROLLE UNE FENETRE VERS LE BAS

Action:scrolle vers le bas de la ligne X à la ligne Y la fenêtre courante.

```
DE54-A9 00      LDA #$00        on prend $0028, soit 40
DE56-85 07      STA $07
```

```
DE58-A9 28      LDA #$28
DE5A-D0 06      BNE $DE62       inconditionnel
```

SCROLLE UNE FENETRE VERS LE HAUT

Action:scrolle vers le haut de la ligne X à la ligne Y la fenêtre courante.

```
DE5C-A9 FF      LDA ##$FF       on prend $FFD8, soit -40 en complément à 2
DE5E-85 07      STA $07
DE60-A9 D8      LDA ##$D8

DE62-85 06      STA $06         $06-07 contiennent le déplacement
DE64-86 00      STX $00         on met la ligne de départ en RES
DE66-98         TYA
DE67-38         SEC
DE68-E5 00      SBC $00         on calcule le nombre de lignes
DE6A-48         PHA             on sauve le nombre de lignes
DE6B-8A         TXA             ligne de début dans A
DE6C-24 06      BIT $06
DE6E-10 01      BPL $DE71       déplacement négatif ?
DE70-98         TYA             oui, ligne de fin dans A
DE71-A6 28      LDX $28
DE73-20 12 DE   JSR $DE12       on calcule l'adresse de la ligne
DE76-18         CLC
DE77-7D 28 02   ADC $0228,X     l'adresse exacte de la ligne dans la fenêtre
DE7A-90 01      BCC $DE7D
DE7C-C8         INY
DE7D-85 08      STA $08         est dans $08-09
DE7F-84 09      STY $09
DE81-18         CLC             on ajoute le déplacement
DE82-65 06      ADC $06
DE84-85 04      STA $04
DE86-98         TYA
DE87-65 07      ADC $07
DE89-85 05      STA $05         dans $04-05
DE8B-68         PLA             on sort le nombre de lignes
DE8C-85 00      STA $00         dans RES
DE8E-F0 34      BEQ $DEC4       si nul on fait n'importe quoi ! on devrait sortir!
DE90-30 3B      BMI $DECD       si négatif, on sort ----------------------------------
DE92-38         SEC             on calcule                                          I
DE93-A6 28      LDX $28                                                             I
DE95-BD 2C 02   LDA $022C,X     la largeur de la fenêtre                            I
DE98-FD 28 02   SBC $0228,X                                                         I
DE9B-85 01      STA $01         dans RES+1                                          I
DE9D-A4 01      LDY $01                                                             I
DE9F-B1 04      LDA ($04),Y     on transfère une ligne                              I
DEA1-91 08      STA ($08),Y                                                         I
DEA3-88         DEY                                                                 I
DEA4-10 F9      BPL $DE9F                                                           I
DEA6-18         CLC                                                                 I
DEA7-A5 04      LDA $04         on ajoute le déplacement                            I
DEA9-65 06      ADC $06         à l'adresse de base                                 I
DEAB-85 04      STA $04                                                             I
DEAD-A5 05      LDA $05                                                             I
DEAF-65 07      ADC $07                                                             I
DEB1-85 05      STA $05                                                             I
DEB3-18         CLC                                                                 I
DEB4-A5 08      LDA $08         et à l'adresse d'arrivée                            I
```

```
)EB6-65 06      ADC $06                                                    I
)EB8-85 08      STA $08                                                    I
)EBA-A5 09      LDA $09                                                    I
)EBC-65 07      ADC $07                                                    I
)EBE-85 09      STA $09                                                    I
)EC0-C6 00      DEC $00         on décompte une ligne de faite             I
)EC2-D0 D9      BNE $DE9D       et on fait toutes les lignes               I
)EC4-A4 01      LDY $01         on remplit la dernière ligne               I
)EC6-A9 20      LDA #$20                                                   I
)EC8-91 08      STA ($08),Y     avec de espaces                            I
)ECA-88         DEY                                                        I
)ECB-10 FB      BPL $DEC8                                                  I
)ECD-60         RTS             <-----------------------------------------
```

???

Action: inconnue... ne semble pas être appelée et utilise des variables IRQ dont on ne sait rien.

```
)ECE-90 07      BCC $DED7       si C=0 on passe ------------
)ED0-A6 28      LDX $28                                    I
)ED2-20 1E DE   JSR $DE1E       on éteint le curseur       I
)ED5-68         PLA             et on sort A de la pile    I
)ED6-60         RTS                                        I
)ED7-A9 01      LDA #$01        on met 1 en $216 <----------
)ED9-8D 16 02   STA $0216
)EDC-A9 80      LDA #$80        on force b7 à 1 dans $217
)EDE-8D 17 02   STA $0217
)EE1-68         PLA             on sort A
)EE2-60         RTS             et on sort
```

DONNEES POUR FENETRES SYSTEME

Type: Chaque bloc de données contient 6 octets qui sont dans l'ordre : colonnes de début et de fin de la fenêtre, lignes de début et de fin, adresse de base de la fenêtre.

FENETRE TEXT

```
)EE3            BYT 0,39        de 0 à 39
)EE5            BYT 1,27        de 1 à 27
)EE7            WRD $BB80       base de la fenêtre
```

FENETRE HIRES

```
)EE9            BYT 0,39        de 0 à 39
)EEB            BYT 0,2         de 0 à 2
)EED            WRD $BF68       base de la fenêtre
```

FENETRE TRACE

```
)EEF            BYT 0,39        de 0 à 39
)EF1            BYT 26,27       de 26 à 27
)EF3            WRD $BB80       adresse de base
```

FENETRE VIDEOTEX

```
)EF5            BYT 0,39        de 0 à 39
```

```
DEF7            BYT 1,24        de 1 à 24
DEF9            WRD $BB80       adresse de base
```

CREE UNE FENETRE

Action:Crée la fenêtre X selon les paramètres en AY (voir tables ci-dessus). La routine contient, comme on l'a déja vu, deux points d'entrée par souci de rapidité : En $DEFB, C=1 et on lit les données sur la banque d'appel par BRK. En $DEFD, C=0, on lit les données sur la banque 7.

```
DEFB-38         SEC             C=1, on appelle d'une autre banque que la 7
DEFC            BYT $24         on saute le CLC
DEFD-18         CLC             C=0, on appelle de la banque 7
DEFE-08         PHP             on sauve C
DEFF-85 15      STA $15         on sauve l'adresse des paramètres
DF01-84 16      STY $16
DF03-8A         TXA             on ajoute 24 au numéro de fenêtre
DF04-18         CLC             (pour 6 paramètres)
DF05-69 18      ADC #$18
DF07-AA         TAX
DF08-A0 05      LDY #$05        on indexe six paramètres
DF0A-28         PLP             on lit C
DF0B-08         PHP
DF0C-B0 04      BCS $DF12       C=1, on change de banque ----------------------
DF0E-B1 15      LDA ($15),Y     C=0, on lit les codes sur la banque 7          I
DF10-90 03      BCC $DF15   ---                                                I
DF12-20 11 04   JSR $0411   I  <-------------------------------------------------
DF15-9D 24 02   STA $0224,X --->on sauve le paramètre
DF18-8A         TXA
DF19-38         SEC
DF1A-E9 04      SBC #$04        et on passe au paramètre suivant
DF1C-AA         TAX             (il y a 4 fenêtres)
DF1D-88         DEY
DF1E-10 EA      BPL $DF0A
DF20-A9 07      LDA #$07        texte blanc
DF22-9D 40 02   STA $0240,X
DF25-A9 00      LDA #$00        et fond noir
DF27-9D 44 02   STA $0244,X
DF2A-A9 00      LDA #$00        pas de curseur
DF2C-9D 48 02   STA $0248,X
DF2F-BD 28 02   LDA $0228,X     curseur en haut à gauche de
DF32-9D 20 02   STA $0220,X     la fenêtre
DF35-BD 30 02   LDA $0230,X
DF38-9D 24 02   STA $0224,X
DF3B-BD 38 02   LDA $0238,X     adresse courante=adresse de base
DF3E-9D 18 02   STA $0218,X
DF41-BD 3C 02   LDA $023C,X
DF44-9D 1C 02   STA $021C,X
DF47-A9 20      LDA #$20        on met un espace sous le curseur
DF49-9D 4C 02   STA $024C,X
DF4C-A5 28      LDA $28         on sauve la fenêtre
DF4E-48         PHA
DF4F-86 28      STX $28         on force fenêtre courante=fenêtre créée
DF51-A9 0C      LDA #$0C        on efface la fenêtre
DF53-20 B5 DB   JSR $DBB5
DF56-68         PLA             on replace le numéro de fenêtre courante
DF57-85 28      STA $28
DF59-28         PLP             on sort P
```

```
)F5A-60         RTS
```

INITIALISATION DE L'AFFICHAGE

```
)F5B-A9 1A      LDA #$1A        on indique mode TEXT
)F5D-8D DF BF   STA $BFDF
)F60-20 49 FE   JSR $FE49       on redéfinit les caractères
)F63-A2 27      LDX #$27        pour 40 colonnes
)F65-A9 20      LDA #$20
)F67-9D 80 BB   STA $BB80,X     on efface la ligne des statuts (ligne 0)
)F6A-CA         DEX
)F6B-10 FA      BPL $DF67
)F6D-A0 11      LDY #$11        on fixe les valeurs par défaut
)F6F-B9 E3 DE   LDA $DEE3,Y
:F72-99 56 02   STA $0256,Y     des 3 fenêtres (TEXT,HIRES et TRACE)
)F75-88         DEY
)F76-10 F7      BPL $DF6F
)F78-0E 0D 02   ASL $020D       on indique mode TEXT
:F7B-4E 0D 02   LSR $020D
)F7E-A9 F5      LDA #$F5        AY=$DEF5 (fenêtre VIDEOTEX)
)F80-A0 DE      LDY #$DE
)F82-2C 0D 02   BIT $020D       mode minitel ?
)F85-70 04      BVS $DF8B       oui, on indexe la bonne fenêtre ----------------
)F87-A9 56      LDA #$56        non, on indexe $0256 (fenêtre TEXT)              I
)F89-A0 02      LDY #$02                                                         I
.F8B-A2 00      LDX #$00        pour la fenêtre 0 <-------------------------------
)F8D-4C FD DE   JMP $DEFD       et on définit la fenêtre 0
```

LIT VALEUR JOYSTICK GAUCHE

```
)F90-AD 20 03   LDA $0320       on lit V2DRB
:F93-29 3F      AND #$3F        on force b7b6 à 0
)F95-09 40      ORA #$40        et b6 à 1
:F97-D0 07      BNE $DFA0       inconditionnel
```

LIT VALEUR SOURIS

```
)F99-AD 20 03   LDA $0320       on lit V2DRB
:F9C-29 3F      AND #$3F        on force b7b6 à 0
.F9E-09 80      ORA #$80        et b7 à 1
 FA0-8D 20 03   STA $0320       dans V2DRB
)FA3-AD 20 03   LDA $0320       on lit V2DRB
:FA6-29 1F      AND #$1F        on isole b5-b0 pour valeur
:FA8-60         RTS             et on sort

:FA9-38         SEC
)FAA-60         RTS
```

INITIALISE LES PORTS JOYSTICK/SOURIS

```
:ction:Installe le joystick gauche (le droit n'est pas géré) et la souris, fixe
       les codes ASCII par défaut retournés par les ports, et installe
       les IRQ par T2 pour se déclencher une fois toutes les 10000 us, soit
       100 fois par secondes.


)FAB-A9 41      LDA #$41        %01000001, joystick gauche et souris
:FAD-8D 8C 02   STA $028C       dans FLGJCK
:FB0-A2 06      LDX #$06        on lit les valeurs par défaut
```

```
)FB2-BD F3 DF   LDA $DFF3,X     dans les 7 octets de valeurs touche/joystick
)FB5-9D 9D 02   STA $029D,X     de JCKTAB à JCKTAB+6
)FB8-CA         DEX
)FB9-10 F7      BPL $DFB2
)FBB-A9 01      LDA #$01        1 en $297 diviseur répétition joystick
)FBD-8D 97 02   STA $0297            $29C diviseur bouton central souris
)FC0-8D 9C 02   STA $029C
)FC3-A9 06      LDA #$06        6 en $298 compteur avant répétition joystick
)FC5-8D 98 02   STA $0298            $29B compteur bouton central souris
)FC8-8D 9B 02   STA $029B
)FCB-A9 01      LDA #$01        1 en $299 diviseur répétition souris
)FCD-8D 99 02   STA $0299
)FD0-A9 0A      LDA #$0A        10 en $29A compteur avant répétition souris
)FD2-8D 9A 02   STA $029A
)FD5-A9 03      LDA #$03        3 en $2A4 diviseur déplacement souris
)FD7-8D A4 02   STA $02A4            $2A5 valeur par défaut du diviseur
)FDA-8D A5 02   STA $02A5
)FDD-A9 10      LDA #$10        $2710 = 10000
)FDF-A0 27      LDY #$27
)FE1-8D 8F 02   STA $028F       dans $28F-290 (valeur timer T2)
)FE4-8C 90 02   STY $0290
)FE7-8D 08 03   STA $0308       et dans T2
)FEA-8C 09 03   STY $0309
)FED-A9 A0      LDA #$A0        %10100000, autorisation IRQ par T2
)FEF-8D 0E 03   STA $030E       dans V1IER
)FF2-60         RTS
```

VALEURS PAR DEFAUT JOYSTICK/SOURIS

```
)FF3            BYT 11,10,32,8,9,3,3 soit les quatres flèches pour les directions
                                     un espace pour le feu joystick ou souris G.
                                     et CTRL-C pour les feux D. et C. souris
```

GESTION JOYSTICK DROIT

```
Action:succinte...

)FFA-60         RTS
```

GESTION DU JOYSTICK GAUCHE

```
Action:lit le joystick et envoie dans le buffer clavier les code ASCII qui
       correspondent aux directions et au bouton. Le code KBDSHT envoyé avec
       les touches sera 32, soit b5 à 1.

)FFB-AD 8D 02   LDA $028D       on lit JGCVAL
)FFE-29 04      AND #$04        bouton pressé ?
E000-D0 12      BNE $E014       non ----------------------------------------------
E002-20 90 DF   JSR $DF90       oui, on lit le joystick                          I
E005-29 04      AND #$04        bouton pressé ?                                  I
E007-D0 15      BNE $E01E       non ------------------------------               I
E009-CE 93 02   DEC $0293       oui, répétition ?                I               I
E00C-D0 29      BNE $E037       non, on passe   -----------------+--------       I
E00E-AE 97 02   LDX $0297       on lit diviseur avant répétition I       I      I
E011-4C 1E E0   JMP $E01E       ---------------------------------0       I      I
E014-20 90 DF   JSR $DF90       on lit le JOY <-------------------+-------+-------
E017-29 04      AND #$04        bouton pressé ?                  I       I
E019-D0 1C      BNE $E037       non ------------------------------+-------0
```

```
E01B-AE 98 02  LDX $0298      on lit compteur par défaut          I       I
E01E-8E 93 02  STX $0293      dans répétition joy <---------------       I
E021-85 58     STA $58        on sauve la valeur Feu                     I
E023-AD 8D 02  LDA $028D      on lit la valeur joystick                  I
E026-29 1B     AND #$1B       %00011011, on élimine le Feu               I
E028-05 58     ORA $58        on ajoute la valeur lue sur VIA2           I
E02A-8D 8D 02  STA $028D      dans JGCVAL                                I
E02D-A5 58     LDA $58        feu pressé ?                               I
E02F-D0 06     BNE $E037      non ---------------------------------------0
E031-AD 9F 02  LDA $029F      oui, on lit le code ASCII correspondant    I
E034-20 9F E1  JSR $E19F      et on l'envoie dans le buffer clavier      I
E037-AD 8D 02  LDA $028D      on lit JGCVAL <----------------------------
E03A-29 1B     AND #$1B       on isole les bits de direction
E03C-49 1B     EOR #$1B       et on les inverse
E03E-F0 1B     BEQ $E05B      si pas de direction, on passe --------------------
E040-20 90 DF  JSR $DF90      on lit la valeur déplacement joystick                I
E043-29 1B     AND #$1B       on isole les bits direction                          I
E045-85 58     STA $58        dans $58                                             I
E047-AD 8D 02  LDA $028D      on lit JGCVAL                                        I
E04A-29 1B     AND #$1B       on isole les bits de direction                       I
E04C-45 58     EOR $58        est-ce le même mouvement ?                           I
E04E-D0 12     BNE $E062      non ------------------------------------------       I
E050-CE 91 02  DEC $0291      oui, on répète ?                      I      I       I
E053-D0 2F     BNE $E084      non ----------------------------------+---   I       I
E055-AE 97 02  LDX $0297      oui, on prend diviseur répétition     I  I   I       I
E058-4C 65 E0  JMP $E065   ---et on saute                           I  I   I       I
E05B-20 90 DF  JSR $DF90   I  on lit la valeur <--------------------+--+---+-------
E05E-29 1B     AND #$1B    I  déplacement                           I  I
E060-85 58     STA $58     I  dans $58                              I  I
E062-AE 98 02  LDX $0298   I  on prend le nb avant répétition <-------------  I
E065-8E 91 02  STX $0291   --->dans décompte répétition                       I
E068-AD 8D 02  LDA $028D      on prend JGCVAL                                 I
E06B-29 04     AND #$04       isole Feu                                       I
E06D-05 58     ORA $58        ajoute direction                                I
E06F-8D 8D 02  STA $028D                                                      I
E072-A2 04     LDX #$04       5 valeurs                                       I
E074-09 04     ORA #$04       on indique pas de FEU                           I
E076-4A        LSR            on envoie les codes ASCII correspondant         I
E077-48        PHA                                                            I
E078-B0 06     BCS $E080                                                      I
E07A-BD 9D 02  LDA $029D,X    au directions détectées                         I
E07D-20 9F E1  JSR $E19F      dans le buffer clavier                          I
E080-68        PLA                                                            I
E081-CA        DEX                                                            I
E082-10 F2     BPL $E076                                                      I
E084-60        RTS            <-----------------------------------------------
```

GESTION DE LA SOURIS

Action: Gère la souris comme précédemment le joystick gauche, à ceci près qu'il ne s'agit plus avec la souris de gérer un délai de répétition (sauf pour les boutons), mais plutot une vitesse de répétition. Dans le buffer clavier, l'octet KBDSHT ajouté au codes ASCII souris est 8, soit b3 à 1.

```
E085-20 99 DF  JSR $DF99      on lit la valeur souris
E088-29 1B     AND #$1B       on isole les directions
E08A-85 58     STA $58        dans $58
E08C-C9 1B     CMP #$1B       la souris bouge ?
```

```
E08E-D0 05     BNE $E095      non ----------------------------------------
E090-CE A4 02  DEC $02A4      on déplace ?                                 |
E093-D0 EF     BNE $E084      non, on sort.                                |
E095-AD A5 02  LDA $02A5      on place vitesse déplacement dans  <---------
E098-8D A4 02  STA $02A4      $2A4
E09B-A5 58     LDA $58        on lit le code
E09D-C9 1B     CMP #$1B       souris fixe ?
E09F-F0 14     BEQ $E0B5      oui ----------------------------------------
E0A1-29 1B     AND #$1B       non, on isole les valeurs direction          |
E0A3-4D 8E 02  EOR $028E      et on retourne les bits de JCDVAL            |
E0A6-29 1B     AND #$1B       en isolant les bits direction                |
E0A8-D0 0B     BNE $E0B5      ce ne sont pas les mêmes exactement ---------|
E0AA-CE 92 02  DEC $0292      on répète ?                                  |
E0AD-D0 31     BNE $E0E0      non                                          |
E0AF-AE 99 02  LDX $0299      oui, on met le diviseur répétition           |
E0B2-4C BB E0  JMP $E0BB   ---dans le compteur                             |
E0B5-20 99 DF  JSR $DF99   I  on lit la souris <---------------------------
E0B8-AE 9A 02  LDX $029A   I  on place le compteur avant répétition
E0BB-8E 92 02  STX $0292   -->dans le décompteur
E0BE-29 1B     AND #$1B       on isole les bits de direction
E0C0-85 58     STA $58        dans $58
E0C2-AD 8E 02  LDA $028E      on prend JDCVAL
E0C5-29 64     AND #$64       %01100100, on isole les bits de Feu
E0C7-05 58     ORA $58        on ajoute les bits de direction
E0C9-8D 8E 02  STA $028E      dans JDCVAL
E0CC-A5 58     LDA $58
E0CE-09 04     ORA #$04       on éteint le feu principal
E0D0-A2 04     LDX #$04
E0D2-4A        LSR
E0D3-48        PHA
E0D4-B0 06     BCS $E0DC
E0D6-BD 9D 02  LDA $029D,X    et on envoie les valeurs ASCII dans le buffer
E0D9-20 9D E1  JSR $E19D
E0DC-68        PLA
E0DD-CA        DEX
E0DE-10 F2     BPL $E0D2
E0E0-60        RTS
```

GERE LES BOUTONS DE LA SOURIS

Principe: Gérés comme le feu du joystick, les boutons souris ont comme les touches du clavier un délai et un diviseur de répétition. Leur gestion est laborieuse et aurait pu être ramené à une routine moitié moins longue avec un peu de reflexion. De plus, renvoyer des codes ASCII pour un déplacement de souris n'est pas ce que l'on peut faire de mieux. Il aurait été plus judicieux d'ajourner des coordonnées en mémoire, ce qui aurait permis des applications fort intéressantes...

```
E0E1-AD 8E 02  LDA $028E      on lit JDCVAL
E0E4-29 04     AND #$04       bouton gauche ?
E0E6-D0 12     BNE $E0FA      oui ----------------------------------------
E0E8-20 99 DF  JSR $DF99      non, on lit la souris                        |
E0EB-29 04     AND #$04       bouton toujours pressé ?                     |
E0ED-D0 13     BNE $E102      non ----------------------------------      |
E0EF-CE 94 02  DEC $0294      oui, on répète ?                  I          |
E0F2-D0 27     BNE $E11B      non ------------------------------+------    |
E0F4-AE 97 02  LDX $0297      oui, on prépare vitesse           I      I   |
E0F7-4C 02 E1  JMP $E102   ---et on saute                       I      I   |
```

```
E0FA-20 99 DF   JSR $DF99     I  bouton gauche pressé ? <-------------+-----+-------
E0FD-29 04      AND #$04      I                                       I     I
E0FF-AE 98 02   LDX $0298     I  X=compteur avant rep.                I     I
E102-85 58      STA $58       I->dans $58 <---------------------------      I
E104-8E 94 02   STX $0294        on place X dans décompte souris            I
E107-AD 8E 02   LDA $028E        on lit JDCVAL                              I
E10A-29 7B      AND #$7B         %01111011 tout sauf bouton gauche          I
E10C-05 58      ORA $58          met $58 dessus                             I
E10E-8D 8E 02   STA $028E        et on replace                              I
E111-A5 58      LDA $58          bouton pressé ?                            I
E113-D0 06      BNE $E11B        non ---------------------------------------0
E115-AD 9F 02   LDA $029F        oui, on envoie l'ASCII correspondant       I
E118-20 9D E1   JSR $E19D        dans le buffer clavier                     I
E11B-AD 8E 02   LDA $028E        lit bouton central <-----------------------
E11E-29 20      AND #$20         %00100000, pressé ?
E120-D0 15      BNE $E137        non ---------------------------------------------
E122-20 99 DF   JSR $DF99        oui, bouton toujours pressé ?                      I
E125-AD 2F 03   LDA $032F                                                           I
E128-29 20      AND #$20                                                           I
E12A-D0 14      BNE $E140        non, on sauve l'image bouton central ----------   I
E12C-CE 95 02   DEC $0295       oui, on répète ?                           I   I
E12F-D0 2A      BNE $E15B     ---non, on va gérér le bouton droit           I   I
E131-AE 9C 02   LDX $029C     I  oui, on prend le diviseur répétition      I   I
E134-4C 40 E1   JMP $E140     I  et on le sauve ---------------------------0   I
E137-20 99 DF   JSR $DF99     I  on lit l'image souris <-------------------+---
E13A-AD 2F 03   LDA $032F     I                                            I
E13D-AE 9B 02   LDX $029B     I  on remet le compteur avant répétition     I
E140-8E 95 02   STX $0295     I  dans décompte bouton central <-------------
E143-29 20      AND #$20      I  bouton central pressé ?
E145-85 58      STA $58       I  dans $58
E147-AD 8E 02   LDA $028E     I  on lit JDCVAL
E14A-29 5F      AND #$5F      I  %01011111 sans bouton central
E14C-05 58      ORA $58       I  on ajoute le bouton central
E14E-8D 8E 02   STA $028E     I  dan JDCVAL
E151-29 20      AND #$20      I  on isole le bouton central
E153-D0 06      BNE $E15B     0--pressé ? non
E155-AD A2 02   LDA $02A2     I  oui, en envoie le code ASCII correspondant
E158-20 9D E1   JSR $E19D     I  dans le buffer clavier
E158-AD 8E 02   LDA $028E     --->on lit le bouton droit
E15E-29 40      AND #$40         pressé ?
E160-D0 15      BNE $E177        non
E162-20 99 DF   JSR $DF99        oui, on lit la souris
E165-AD 2F 03   LDA $032F
E168-29 80      AND #$80         b7=1 ? (mode souris ?)
E16A-D0 14      BNE $E180        non ---------------------------------------------
E16C-CE 96 02   DEC $0296        oui, on a décompté à 0 ?                           I
E16F-D0 2B      BNE $E19C        non ------------------------------------          I
E171-AE 9C 02   LDX $029C        oui, on prend le diviseur                  I       I
E174-4C 80 E1   JMP $E180     ---et on saute                                I       I
E177-20 99 DF   JSR $DF99     I  on lit la souris                           I       I
E17A-AD 2F 03   LDA $032F     I  on prend la valeur                         I       I
E17D-AE 9B 02   LDX $029B     I  et le compteur                             I       I
E180-8E 96 02   STX $0296     -->dans le décompte <-----------------------+--------
E183-4A         LSR                                                       I
E184-29 40      AND #$40         on met b7 dans b6, et on isole           I
E186-85 58      STA $58          dans $58                                 I
E188-AD 8E 02   LDA $028E        on prend JDCVAL                          I
E18B-29 3F      AND #$3F         isole directions                         I
E18D-05 58      ORA $58          ajoute b6 sauvé                          I
E18F-8D 8E 02   STA $028E        dans JDCVAL                              I
E192-29 40      AND #$40         bouton pressé ?                          I
```

```
E194-D0 06      BNE $E19C       non, on sort
E196-AD A3 02   LDA $02A3       oui, on envoie le code bouton souris
E199-4C 9D E1   JMP $E19D       ou mieux:faire pointer le branchements en $E1B6
E19C-60         RTS
```

ENVOIE UN CODE ASCII SOURIS

```
E19D-38         SEC
E19E            BYT $2C
```

ENVOIE UN CODE ASCII JOYSTICK

```
E19F-18         CLC
E1A0-08         PHP             C=1 si code souris, 0 si code joystick
E1A1-86 58      STX $58         on sauve X
E1A3-A2 00      LDX #$00        on envoie le code au buffer clavier
E1A5-20 1D C5   JSR $C51D       (buffer 0)
E1A8-A9 08      LDA #$08        on prend code souris
E1AA-28         PLP
E1AB-B0 02      BCS $E1AF
E1AD-A9 20      LDA #$20        si C=0, on prend code JOYSTICK
E1AF-A2 00      LDX #$00
E1B1-20 1D C5   JSR $C51D       et on envoie le code KBDSHT
E1B4-A6 58      LDX $58         on restaure X
E1B6-60         RTS             et on sort

E1B7-38         SEC             l'utilité de ces codes est douteuse...il y a
E1B8-60         RTS             surement d'autres SEC/RTS entre $C000 et $FFFF...
```

HARD-COPY TEXT

Action:recopie tous les caractères à l'écran sur imprimante.
La fenêtre est dans $28. Pour lire les codes, on déplace le curseur de droite à gauch et de bas en haut en envoyant tout code imprimable.

```
E1B9-A6 28      LDX $28         on sauve la position du curseur
E1BB-BD 20 02   LDA $0220,X
E1BE-48         PHA
E1BF-BD 24 02   LDA $0224,X
E1C2-48         PHA
E1C3-A9 1E      LDA #$1E        on fait un HOME
E1C5-20 B5 DB   JSR $DBB5       (curseur en haut à gauche de la fenêtre)
E1C8-20 E4 DA   JSR $DAE4       on saute une ligne sur l'imprimante
E1CB-A6 28      LDX $28     -->on prend le numéro de fenêtre
E1CD-BC 20 02   LDY $0220,X I
E1D0-B1 26      LDA ($26),Y I   on lit un code
E1D2-C9 20      CMP #$20    I   code de controle ?
E1D4-B0 02      BCS $E1D8   I
E1D6-A9 20      LDA #$20    I   oui, on affiche un espace
E1D8-20 72 DA   JSR $DA72   I   sur imprimante
E1DB-BD 20 02   LDA $0220,X I   on est à la fin de la ligne ?
E1DE-DD 2C 02   CMP $022C,X I
E1E1-F0 08      BEQ $E1EB   I   oui ----------------------------------------
E1E3-A9 09      LDA #$09    I   on déplace le curseur à droite <-----      I
E1E5-20 B5 DB   JSR $DBB5   I   à l'écran                        I         I
E1E8-4C CB E1   JMP $E1CB   ---et on tourne                      I         I
E1EB-20 E4 DA   JSR $DAE4       on passe à la ligne suivante <---+---------
E1EE-A6 28      LDX $28                                          I
E1F0-BD 24 02   LDA $0224,X     on est sur la dernière ligne ?   I
```

```
E1F3-DD 34 02  CMP $0234,X                                        I
E1F6-D0 EB     BNE $E1E3       non, on déplace encore à droite -----
E1F8-A9 1F     LDA #$1F
E1FA-20 B5 DB  JSR $DBB5
E1FD-68        PLA             on sort X
E1FE-09 40     ORA #$40
E200-20 B5 DB  JSR $DBB5
E203-68        PLA             et X
E204-09 40     ORA #$40
E206-4C B5 DB  JMP $DBB5       et on replace le curseur
```

HARD-COPY VIDEOTEX

Action:On envoie sur imprimante le contenu ASCII (contenu des tables ASCII en fait) de l'écran HIRES en mode VIDEOTEX. Un petit bug s'est glissé dans la routine : La largeur de la ligne sauvée au début n'est pas restituée à la fin, mais placée dans LPRX. Ce qui n'est pas très grave par ailleurs. Donc en sortie de cette routine, LPRFX=40.

```
E209-20 E4 DA  JSR $DAE4       on saute une ligne
E20C-AD 88 02  LDA $0288       on sauve largeur d'impression
E20F-48        PHA
E210-A9 28     LDA #$28        on met largeur à 40
E212-8D 88 02  STA $0288
E215-A9 1E     LDA #$1E        on place le curseur en 0,0
E217-20 78 D1  JSR $D178
E21A-A4 38     LDY $38
E21C-B1 30     LDA ($30),Y     on lit l'attribut du code
E21E-30 06     BMI $E226       >128, on saute  ----------------------------------
E220-B1 2E     LDA ($2E),Y     on prend le code                                  I
E222-C9 20     CMP #$20        code de controle ?                                I
E224-B0 02     BCS $E228   ---non                                               I
E226-A9 20     LDA #$20    I   on orend un espace <-------------------------------
E228-20 72 DA  JSR $DA72   -->on affiche le code
E22B-A9 09     LDA #$09        on déplace le curseur à droite
E22D-20 78 D1  JSR $D178
E230-A5 38     LDA $38         on ajuste la position suivante
E232-D0 E6     BNE $E21A
E234-A4 39     LDY $39
E236-88        DEY
E237-D0 E1     BNE $E21A       et on boucle
E239-20 E4 DA  JSR $DAE4       on saute une ligne
E23C-68        PLA             et on bugge !!!
E23D-8D 86 02  STA $0286       STA $0288 aurait été mieux...
E240-60        RTS

E241           BYT $18,$33,$1B,$0A,$0D    soit saut de ligne, interligne 24/216
E246           BYT $00,$F0,$4B,$1B,$0D,$0A    soit saut de ligne, 240 points
E24C           BYT $40,$1B,$0A,$0A        soit initialise l'imprimante
```

EXECUTION HARD-COPY HIRES

Principe:Le hard-copy HIRES est vectorisé, c'est-à-dire que l'on peut placer dans le vecteur $250 l'adresse de sa routine personnelle. pratique !

```
E250-6C 50 02  JMP ($0250)
```

ROUTINE DE HARD-COPY HIRES

Principe:Trivial, mais un bug se cache quelque part dans la routine (la 17ème ligne et la 18' sont superposées...) sans raison.apparente ! Les octets graphiques EPSON etant verticaux alors que les octets (sextet en fait) graphiques ORIC sont horizontaux, on va lire 8 sextets superposées verticalement, isoler leur bit du 5 au 0 et le faire entrer dans le motif EPSON. On envoie alors le motif et on fait de même pour les 6 bits, puis pour les 40 sextets et pour les 25 blocs de 8 fois 40 sextets... ouf ! La routine est très optimisée, profitez en !

```
E253-A2 05     LDX #$05        on va envoyer 5 codes
E255-AD 8A 02  LDA $028A       on sauve FLGLPR
E258-48        PHA
E259-09 40     ORA #$40        et on interdit le CRLF du TELEMON
E25B-8D 8A 02  STA $028A
E25E-BD 40 E2  LDA $E240,X     on envoie le passage en 24/216' de pouce
E261-20 72 DA  JSR $DA72       à l'imprimante
E264-CA        DEX
E265-D0 F7     BNE $E25E
E267-86 0C     STX $0C         on met 0 dans $0C
E269-A2 06     LDX #$06        en envoie 6 codes
E26B-BD 45 E2  LDA $E245,X
E26E-20 72 DA  JSR $DA72       soit passage en graphiques 240 points/ligne
E271-CA        DEX
E272-D0 F7     BNE $E26B
E274-86 0D     STX $0D         et 0 dans $0D
E276-A9 05     LDA #$05        on va faire 6 pixels (5+1)
E278-85 0E     STA $0E         dans $0E
E27A-A5 0C     LDA $0C         on prend la ligne
E27C-0A        ASL             *2
E27D-0A        ASL             *4
E27E-0A        ASL             *8
E27F-20 69 CE  JSR $CE69       *40 (donc on calcule la ligne suivante)
E282-85 11     STA $11         dans $11-$12
E284-98        TYA             on calcule #A000+ligne*8*40
E285-18        CLC
E286-69 A0     ADC #$A0
E288-85 12     STA $12
E28A-A9 08     LDA #$08        pour 8 ligne
E28C-85 10     STA $10         dans $10
E28E-A4 0D     LDY $0D         on prend la colonne
E290-B1 11     LDA ($11),Y     on lit le code
E292-AA        TAX             dans X
E293-29 40     AND #$40        pixels ?
E295-D0 04     BNE $E29B       oui
E297-8A        TXA             attributs, on inverse vidéo ?
E298-29 80     AND #$80        b7=1 si vidéo inverse
E29A-AA        TAX
E29B-8A        TXA
E29C-10 02     BPL $E2A0       vidéo inverse ? non ----------------------------
E29E-49 3F     EOR #$3F        oui, on inverse les pixels                      I
E2A0-A6 0E     LDX $0E         on isole le bit (numéro dans $0E) <--------------
E2A2-4A        LSR             dans C
E2A3-CA        DEX
E2A4-10 FC     BPL $E2A2
E2A6-26 0F     ROL $0F         on l'ajoute à la ligne (dans $0F)
E2A8-98        TYA
E2A9-18        CLC
E2AA-69 28     ADC #$28        on saute une ligne écran
E2AC-A8        TAY
E2AD-90 02     BCC $E2B1
E2AF-E6 12     INC $12
E2B1-C6 10     DEC $10         et on fait 8 lignes
E2B3-D0 DB     BNE $E290
E2B5-A5 0F     LDA $0F         on envoie le code calculé
```

```
E2B7-20 72 DA   JSR $DA72       à l'imprimante
E2BA-C6 0E      DEC $0E
E2BC-10 BC      BPL $E27A       on fait 6 bits
E2BE-E6 0D      INC $0D
E2C0-A5 0D      LDA $0D
E2C2-C9 28      CMP #$28
E2C4-D0 B0      BNE $E276       on fait 40 colonnes de 6 bits
E2C6-E6 0C      INC $0C
E2C8-A5 0C      LDA $0C
E2CA-C9 19      CMP #$19        et 25 lignes de 40 colonnes de 6 bits
E2CC-D0 9B      BNE $E269
E2CE-A2 04      LDX #$04
E2D0-BD 4B E2   LDA $E24B,X     puis on initialise l'imprimante
E2D3-20 72 DA   JSR $DA72
E2D6-CA         DEX
E2D7-D0 F7      BNE $E2D0
E2D9-68         PLA
E2DA-8D 8A 02   STA $028A       et on récupère l'état initial du CRLF
E2DD-60         RTS
```

PLACE LE CURSEUR SUR LE DERNIER CARACTERE DE LA LIGNE

```
E2DE-BC 2C 02   LDY $022C,X     on prend la dernière colonne de la fenêtre
E2E1            BYT $24         et on saute l'instruction suivante
E2E2-88         DEY             on décale d'une colonne à gauche <----------
E2E3-B1 00      LDA ($00),Y     est-on sur un espace ?                      I
E2E5-C9 20      CMP #$20                                                    I
E2E7-D0 07      BNE $E2F0       non ----------------------------------      I
E2E9-98         TYA             oui                                 I       I
E2EA-DD 28 02   CMP $0228,X     on est au début de la fenêtre ?     I       I
E2ED-D0 F3      BNE $E2E2       non ----------------------------------+------
E2EF-60         RTS                                                 I
E2F0-C9 7F      CMP #$7F        est-ce le prompt ? <-------------------
E2F2-D0 04      BNE $E2F8       non, on sort
E2F4-98         TYA             oui, Y contient la colonne
E2F5-DD 28 02   CMP $0228,X     et Z=1 si on est sur la première colonne
E2F8-60         RTS
```

TESTE SI UN PROMPT SE TROUVE EN DEBUT DE LIGNE

```
E2F9-BC 28 02   LDY $0228,X     on prend le début de la fenêtre
E2FC-B1 00      LDA ($00),Y     on lit le caractère qui s'y trouve
E2FE-C9 7F      CMP #$7F        on le compare au prompt
E300-60         RTS             Z=1 si il y a un prompt en début de ligne
```

CALCULE LA POSITION APRES UN PROMPT

Action:On calcule dans $60 et $61 la ligne et la colonne du premier prompt avant le curseur. En fait, $61 contient la position immédiatement après le prompt. En sortie, C=0.

```
E301-A6 28      LDX $28         X=numéro de fenêre
E303-BD 24 02   LDA $0224,X     on prend la ligne du curseur
E306-85 61      STA $61         dans $61
E308-A5 61      LDA $61      -->on prend la ligne du curseur
E30A-20 12 DE   JSR $DE12    I  AY et RES contiennent l'adresse de la ligne
E30D-20 F9 E2   JSR $E2F9    I  y a-t-il un prompt au début de la ligne ?
E310-F0 0B      BEQ $E31D    I  oui ------------------------------------------
E312-A5 61      LDA $61      I  non, le curseur est-il                       I
E314-DD 30 02   CMP $0230,X  I  en haut de la fenêtre ?                      I
```

```
E317-F0 08      BEQ $E321 ·  I  oui, on passe                                  ]
E319-C6 61      DEC $61      I  non, on remonte une ligne et                   ]
E31B-B0 EB      BCS $E308    ---on poursuit le test                            I
E31D-18         CLC             C=0 <-------------------------------------------
E31E-C8         INY
E31F-84 60      STY $60         $60 contient la colonne après le prompt
E321-60         RTS
```

RAMENE LE CURSEUR EN DEBUT DE LIGNE

Action:Calcule dans $62,$63 les coordonnées de la fin de la ligne ou de la position du premier prompt +1.

```
E322-A6 28      LDX $28         X=numéro de fenêtre courante
E324-BD 24 02   LDA $0224,X     on prend la ligne du curseur
E327-85 63      STA $63         dans $63
E329-20 12 DE   JSR $DE12       on calcule l'adresse de la ligne dans RES et AY
E32C-20 DE E2   JSR $E2DE       on place le curseur sur le dernier caractère
E32F-84 62      STY $62      -->est-on au début de la ligne ?
E331-F0 1B      BEQ $E34E    I  oui ------------------------------------------
E333-A5 63      LDA $63      I  non, on prend la ligne                         I
E335-DD 34 02   CMP $0234,X  I  est-on en bas de la fenêtre ?                  I
E338-F0 13      BEQ $E34D    I  oui ------------------------------------      I
E33A-E6 63      INC $63      I  non, on descend d'une ligne                I    I
E33C-A5 63      LDA $63      I                                             I    I
E33E-20 12 DE   JSR $DE12    I  on calcule son adresse                     I    I
E341-20 F9 E2   JSR $E2F9    I  on st sur un prompt ?                      I    I
E344-F0 05      BEQ $E34B    I  oui ---------------------------------------I    I
E346-20 DE E2   JSR $E2DE    I  non, on cherche le dernier caractère       I I  I
E349-D0 E4      BNE $E32F    ---tant qu'on est pas sur la première colonneI I  I
E34B-C6 63      DEC $63         on remonte d'une ligne <------------------- I  I
E34D-60         RTS             <-------------------------------------------  I
E34E-60         RTS             ben voyons ! le $E34D ne suffisait pas !? <-------
```

ENVOIE LA LIGNE DANS BUFEDT

```
E34F-20 01 E3   JSR $E301       on calcule la position d'un prompt ($60-61)
E352-4C 61 E3   JMP $E361       et on envoie...
```

ENVOIE LA FIN DE LA LIGNE DANS BUFEDT

Action:envoie la ligne courante à partir du curseur dans BUFEDT.

```
E355-A6 28      LDX $28
E357-BD 20 02   LDA $0220,X
E35A-85 60      STA $60         on prend les coordonnées du curseur
E35C-BD 24 02   LDA $0224,X
E35F-85 61      STA $61
E361-20 22 E3   JSR $E322       on calcule le début de la ligne ($62-63)
E364-A5 61      LDA $61         on sauve la ligne trouvée
E366-85 65      STA $65
E368-C5 63      CMP $63         est-ce après un prompt ?
E36A-D0 0C      BNE $E378       non ----------------------------------------
E36C-A5 62      LDA $62         oui, les colonnes correspondent ?             I
E36E-C5 60      CMP $60                                                       I
E370-B0 06      BCS $E378       non, le prompt est avant le début de ligne ------0
E372-A9 00      LDA #$00        oui, on indique commande vide                 I
E374-8D 90 05   STA $0590       dans BUFEDT                                   I
E377-60         RTS                                                           I
```

```
E378-A9 00     LDA #$00        on met 0 <-------------------------------------
E37A-85 64     STA $64         dans $64
E37C-46 66     LSR $66         on décale $66 (b7=0)
E37E-A5 65     LDA $65         on lit la ligne dans $65
E380-20 12 DE  JSR $DE12       on calcule son adresse dans RES
E383-A4 60     LDY $60         on prend la colonne de début de commande
E385-A5 65     LDA $65         et la ligne dans $65
E387-C5 61     CMP $61         est-ce la ligne de début de commande ?
E389-F0 05     BEQ $E390       oui ---------------------------------------------
E38B-A6 28     LDX $28                                                          I
E38D-BC 28 02  LDY $0228,X     non, alors la colonne est le début de la fenêtre I
E390-B1 00     LDA ($00),Y--->on lit le code qui s'y trouve <-------------------
E392-C9 20     CMP #$20     I  un code de controle ?
E394-B0 02     BCS $E398    I  non ------------------------------------------------
E396-09 80     ORA #$80     I  oui, on l'inverse par b7=1 (pourquoi ?)           I
E398-A6 64     LDX $64      I  X=position dans la ligne <-----------------------
E39A-24 66     BIT $66      I  b7 de $66=1 ?
E39C-10 06     BPL $E3A4    I  non ------------------------------------------------
E39E-A9 20     LDA #$20     I  oui on efface le code                             I
E3A0-91 00     STA ($00),YI  on met un espace sous le curseur                     I
E3A2-D0 0D     BNE $E3B1    I---inconditionnel                                   I
E3A4-9D 90 05  STA $0590,XII  on stocke le code <-------------------------------
E3A7-E6 64     INC $64      II  on indexe la position suivante
E3A9-E4 67     CPX $67      II  est on en fin de ligne ?
E3AB-90 04     BCC $E3B1    IO--non
E3AD-C6 64     DEC $64      II  oui, on décrémente l'index
E3AF-66 66     ROR $66      II  et on met b7 de $66 à 1
E3B1-98        TYA          I-->on sauve la position écran du curseur dans A
E3B2-C8        INY          I   on avance sur le caractère suivant
E3B3-A6 65     LDX $65      I   on prend l'indexe de ligne
E3B5-E4 63     CPX $63      I   on est sur la dernière ligne ?
E3B7-D0 0C     BNE $E3C5    I   non ------------------------------------------------
E3B9-C5 62     CMP $62      I   oui, dernière colonne ?                           I
E3BB-D0 D3     BNE $E390    ----non                                             I
E3BD-A6 64     LDX $64                                                          I
E3BF-A9 00     LDA #$00                                                         I
E3C1-9D 90 05  STA $0590,X     oui, on indique fin de commande                  I
E3C4-60        RTS             et on sort                                       I
E3C5-A6 28     LDX $28         on prend le numéro de fenêtre <------------------
E3C7-DD 2C 02  CMP $022C,X     on est sur la dernière colonne de la fenêtre ?
E3CA-D0 C4     BNE $E390       pas fin de la fenêtre, on boucle
E3CC-E6 65     INC $65         fin, on ajoute 1 à la ligne
E3CE-D0 AE     BNE $E37E       incoditionnel, on continue la lecture
```

AFFICHE LE CONTENU DE BUFEDT

```
Action:Si C=1, efface la ligne de commande.
       Si C=0, affiche le buffer d'édition sur la ligne de commande.
       Cette routine est buggée car le dernier adressage du curseur gère sans
       crier gare l'unique fenêtre 0 ! L'édition de ligne ne se fera donc que
       sur la fenêtre 0...

E3D0-66 66     ROR $66         C dans b7 de $66
E3D2-A9 00     LDA #$00        0 en $64
E3D4-85 64     STA $64
E3D6-A5 26     LDA $26         AY=ADSCR
E3D8-A4 27     LDY $27
E3DA-85 00     STA $00         dans RES
E3DC-84 01     STY $01
E3DE-A6 28     LDX $28
E3E0-BC 20 02  LDY $0220,X     on prend la colonne du curseur
```

```
E3E3-A6 64     LDX $64          <-------------------------------------------
E3E5-BD 90 05  LDA $0590,X      on lit le code indexé dans BUFEDT                I
E3E8-F0 32     BEQ $E41C        si 0, on saute ----------------------------------+--
E3EA-A9 20     LDA #$20         on met un espace dans A                          II
E3EC-24 66     BIT $66          si il faut effacer la ligne                      II
E3EE-30 0B     BMI $E3FB        on efface ----------------------------           II
E3F0-BD 90 05  LDA $0590,X      sinon, on lit un code                 I          II
E3F3-10 06     BPL $E3FB                                              I          II
E3F5-C9 A0     CMP #$A0         >32+128 ?                             I          II
E3F7-B0 02     BCS $E3FB        oui ------------------------------------O          II
E3F9-29 1F     AND #$1F         non, code de controle                 I          II
E3FB-91 00     STA ($00),Y      à l'écran <----------------------------           II
E3FD-2C 0D 02  BIT $020D        mode minitel ?                                    II
E400-50 03     BVC $E405     ---non                                             II
E402-20 56 E6  JSR $E656     I  oui, on envoie le code dans le buffer SERIE OUT  II
E405-98        TYA           -->index écran dans A                              II
E406-C8        INY              on ajoute une colonne                           II
E407-A6 28     LDX $28                                                          II
E409-DD 2C 02  CMP $022C,X      dernière colonne ?                              II
E40C-D0 0A     BNE $E418        non ----------------------------------         II
E40E-A9 28     LDA #$28         on ajoute 40 (une ligne)                 I      II
E410-A0 00     LDY #$00         à RES (adresse curseur)                  I      II
E412-20 89 CE  JSR $CE89        par ADRES                                I      II
E415-BC 28 02  LDY $0228,X      Y=première colonne de la fenêtre         I      II
E418-E6 64     INC $64          on indexe caractère suivant dans BUFEDT---      II
E41A-D0 C7     BNE $E3E3        incontionnel  <--------------------------------I
E41C-2C 0D 02  BIT $020D        mode minitel ? <-------------------------------
E41F-50 09     BVC $E42A        non, on sort ----------------------------------
E421-AE 20 02  LDX $0220        on prend la position du curseur (BUG !!!)          I
E424-AC 24 02  LDY $0224        et la ligne (RE-BUG !! pourquoi fenêtre 0 ?)       I
E427-20 2A E6  JSR $E62A        et on positionne le curseur                        I
E42A-AC 20 02  LDY $0220        on prend la position du curseur (BUG : LDY,X !) <-
E42D-B1 26     LDA ($26),Y      on lit le code
E42F-A6 28     LDX $28
E431-9D 4C 02  STA $024C,X      et on le place sous le curseur
E434-60        RTS
```

EDITION DE LIGNE DANS BUFEDT

Action: Cette macro routine lit au clavier un commande, calcule s'il y a lieu le nombre en début de ligne (supposé être le numéro de ligne), et sort en codant la ligne dans BUFEDT. Un bug malencontreux ne permet son usage que sur la fenêtre 0.
En entrée, A contient la longueur maxi de la commande (<110), Y>128 s'il faut ramener le curseur en début de ligne (il est ramené de toutes façons par le CR en E446 !) et X le nombre de caractères à sauter avant l'édition. En sortie, la ligne est dans BUFEDT terminée par un 0, A contient le mode de sortie (RETURN 13, CTRL-C 3 ou FLECHES 10,11) et RES le numéro de ligne ainsi que le nombre d'espaces bidons dans Y.

```
E435-85 67     STA $67          on sauve la longueur demandée
E437-8A        TXA              le déplacement
E438-48        PHA
E439-98        TYA
E43A-10 0A     BPL $E446        faut-il revenir au début de la ligne ? non -------
E43C-20 01 E3  JSR $E301        on calcule les coordonnées du début de ligne     I
```

```
E43F-A6 60      LDX $60
E441-A4 61      LDY $61
E443-20 2A E6   JSR $E62A       et on positionne le curseur
E446-A9 0D      LDA #$0D        on envoie un CR <----------------------------
E448-20 48 E6   JSR $E648
E44B-20 6C E6   JSR $E66C       on affiche le prompt
E44E-68         PLA
E44F-AA         TAX             on récupère le déplacement
E450-F0 08      BEQ $E45A
E452-A9 09      LDA #$09        et on se déplace
E454-20 48 E6   JSR $E648
E457-CA         DEX             X fois
E458-D0 F8      BNE $E452
E45A-A6 28      LDX $28      -->X=numéro de fenêtre courante
E45C-BD 48 02   LDA $0248,X  I  le curseur doit-être affiché ?
E45F-30 05      BMI $E466    I  oui ----------------------------------------
E461-A9 11      LDA #$11     I  non, on bascule, donc on force l'affichage
E463-20 48 E6   JSR $E648    I
E466-20 AF CF   JSR $CFAF    I  on teste les buffers  <-------------------------0
E469-20 CF C7   JSR $C7CF    I  on lit un code au clavier
E46C-B0 F8      BCS $E466    I  pas de code, on boucle ---------------------------
E46E-48         PHA          I  on sauve le code
E46F-A9 11      LDA #$11     I  on efface le curseur
E471-20 48 E6   JSR $E648    I
E474-68         PLA          I  on reprend le code
E475-C9 0D      CMP #$0D     I  est-ce RETURN ?
E477-D0 43      BNE $E4BC    I  non ----------------------------------------------
E479-48         PHA          I  <-------------------------------------------------
E47A-20 4F E3   JSR $E34F    I  oui, on code BUFEDT
E47D-68         PLA          I
E47E-48         PHA          I
E47F-C9 0B      CMP #$0B     I  est-ce flèche haut ?
E481-F0 11      BEQ $E494    I  oui ----------------------------------------------
E483-A6 62      LDX $62      I  on prend les coordonnées courantes
E485-A4 63      LDY $63      I  du curseur
E487-20 2A E6   JSR $E62A    I  et on positionne le curseur
E48A-A9 0D      LDA #$0D     I  et on emèt un CR
E48C-20 48 E6   JSR $E648    I
E48F-A9 0A      LDA #$0A     I  puis un LF
E491-20 48 E6   JSR $E648    I
E494-A2 FF      LDX #$FF     I  <-------------------------------------------------
E496-E8         INX          I  on va éliminer les espaces de début de ligne<-
E497-BD 90 05   LDA $0590,X  I
E49A-C9 20      CMP #$20     I  un espace ?
E49C-F0 F8      BEQ $E496    I  oui ----------------------------------------------
E49E-8A         TXA          I  non, on sauve le nombre d'espaces bidons
E49F-48         PHA          I
E4A0-A0 00      LDY #$00     I
E4A2            BYT $2C      I  le premier tour se fait sans incrémenter
E4A3-E8         INX          I
E4A4-C8         INY          I
E4A5-BD 90 05   LDA $0590,X  I  on ramène les caractères après les espaces
E4A8-99 90 05   STA $0590,Y  I  au début de la ligne
E4AB-D0 F6      BNE $E4A3    I  et on sort si on est sur le 0 final
E4AD-A9 90      LDA #$90     I  AY=$590, BUFEDT
E4AF-A0 05      LDY #$05     I
E4B1-20 49 E7   JSR $E749    I  on calcule le numéro de ligne dans RES
E4B4-85 00      STA $00      I
E4B6-84 01      STY $01      I
E4B8-68         PLA          I  on sort le nombre d'espaces bidon dans Y
```

```
E4B9-A8        TAY          ;
E4BA-68        PLA          ;  et le mode de sortie dans A
E4BB-60        RTS          ;
E4BC-C9 03     CMP #$03     ;  est-ce CTRL C ? <----------------------+
E4BE-F0 B9     BEQ $E479    ;  oui, on sort  ----------------------------
E4C0-C9 0E     CMP #$0E     ;  non, CTRL-N ?
E4C2-D0 0D     BNE $E4D1    ;  non ----------------------------------------
E4C4-20 01 E3  JSR $E301    ;  oui, on calcule la position du début de la ligne
E4C7-A6 60     LDX $60      ;
E4C9-A4 61     LDY $61      ;
E4CB-20 2A E6  JSR $E62A    ;  on ramène le curseur en début de ligne
E4CE-4C D5 E4  JMP $E4D5    ;  et on saute
E4D1-C9 18     CMP #$18     ;  CTRL-X ? <----------------------------------
E4D3-D0 0A     BNE $E4DF    ;  non ----------------------------------------
E4D5-20 55 E3  JSR $E355    ;  oui, on vide la ligne
E4D8-38        SEC          ;  C=1 pour vider BUFEDT
E4D9-20 D0 E3  JSR $E3D0    ;  on vide BUFEDT
E4DC-4C 5A E4  JMP $E45A    ;--et on continue l'édition
E4DF-C9 7F     CMP #$7F     ;  DEL ? <-------------------------------------
E4E1-D0 47     BNE $E52A    ;  non, on passe
E4E3-AD 78 02  LDA $0278    ;  SHIFT pressé ?
E4E6-4A        LSR          ;
E4E7-B0 0A     BCS $E4F3    ;  oui ----------------------------------------
E4E9-90 00     BCC $E4EB    ;  n'importe quoi !
E4EB-A9 08     LDA #$08     ;  A=déplacement vers la gauche du curseur
E4ED           BYT $2C      ;  et on saute l'instruction suivante
E4EE-A9 09     LDA #$09     ;  A=déplacement vers la droite
E4F0-20 48 E6  JSR $E648    ;  on envoie le déplacement
E4F3-A6 28     LDX $28      ;  X=numéro de fenêtre courante <--------------
E4F5-BD 4C 02  LDA $024C,X  ;  on prend le code sous le curseur
E4F8-C9 7F     CMP #$7F     ;  est-ce un prompt ?
E4FA-F0 F2     BEQ $E4EE    ;  oui, on déplace le curseur à gauche
E4FC-20 55 E3  JSR $E355    ;  on recopie la ligne dan BUFEDT
E4FF-AD 90 05  LDA $0590    ;  ligne vide ?
E502-D0 0D     BNE $E511    ;  non ----------------------------------------
E504-A9 20     LDA #$20     ;  on envoie un espace à l'écran
E506-20 48 E6  JSR $E648    ;
E509-A9 08     LDA #$08     ;  et on replace le curseur sur l'espace
E50B-20 48 E6  JSR $E648    ;
E50E-4C 5A E4  JMP $E45A    ;  et on continue l'édition
E511-A2 01     LDX #$01     ;  on prend un code <--------------------------
E513-BD 90 05  LDA $0590,X  ;
E516-F0 06     BEQ $E51E    ;  si 0 , on sort -----------------------------
E518-9D 8F 05  STA $058F,X  ;  sinon, on le ramène à gauche dans BUFEDT
E51B-E8        INX          ;
E51C-D0 F5     BNE $E513    ;  et on ramène ainsi la ligne
E51E-A9 20     LDA #$20     ;  et on envoie un espace en dernière position <-----
E520-9D 8F 05  STA $058F,X  ;
E523-18        CLC          ;  C=0, on affiche BUFEDT
E524-20 D0 E3  JSR $E3D0    ;
E527-4C 5A E4  JMP $E45A    ;  et on continue l'édition
E52A-C9 20     CMP #$20     ;  code CTRL ?
E52C-90 06     BCC $E534    ;  oui ----------------------------------------
E52E-20 37 E5  JSR $E537    ;  on, on envoie le code
E531-4C 5A E4  JMP $E45A    ;  et on continue l'édition
E534-4C B9 E5  JMP $E5B9    ;  on gère les codes de controle <--------------
```

TRAITE LES CARACTERES NORMAUX

Remarque:Un gros BUG vient salir la routine de son empreinte malodorante...
les caractères sortant à droite en bas de l'écran sont perdus...

```
E537-A8        TAY             code dans Y
E538-8A        TXA             on sauve X
E539-48        PHA
E53A-98        TYA             et le code
E53B-48        PHA
E53C-20 55 E3  JSR $E355       on copie la commande après le curseur dans BUFEDT
E53F-A5 62     LDA $62         A=colonne
E541-AC 90 05  LDY $0590       Y=premier code
E544-D0 02     BNE $E548       si pas fin de buffer, on saute ----------------
E546-A5 60     LDA $60         A=colonne dans la fenêtre                      I
E548-A6 28     LDX $28         X=numéro de fenêtre <--------------------------
E54A-DD 2C 02  CMP $022C,X     est-on à la fin de la fenêtre ?
E54D-D0 5F     BNE $E5AE       non ------------------------------------------
E54F-A5 63     LDA $63                                                        I
E551-DD 34 02  CMP $0234,X     fin de la fenêtre en bas ?                     I
E554-F0 58     BEQ $E5AE       oui, on sort ----------------------------------0
E556-69 01     ADC #$01        non, on ajoute 1 ligne                         I
E558-20 12 DE  JSR $DE12       on calcule l'adresse de la ligne               I
E55B-20 F9 E2  JSR $E2F9       on est sur un prompt ?                         I
E55E-D0 4E     BNE $E5AE       non, on sort ----------------------------------0
E560-BC 34 02  LDY $0234,X     oui, Y=fin de la fenêtre                       I
E563-A6 63     LDX $63         X=ligne curseur                                I
E565-E8        INX             +1                                             I
E566-20 5C DE  JSR $DE5C       on scrolle la fenêtre                          I
E569-2C 0D 02  BIT $020D       mode minitel ?                                 I
E56C-50 40     BVC $E5AE       non, on sort ----------------------------------0
E56E-A2 00     LDX #$00                                                       I
E570-A4 63     LDY $63                                                        I
E572-C8        INY                                                            I
E573-20 2A E6  JSR $E62A       on se positionne sur la colonne 1 virtuelle    I
E576-A9 18     LDA #$18        on envoie un CTRL-X                            I
E578-20 56 E6  JSR $E656                                                      I
E57B-A9 0A     LDA #$0A        et un saut de ligne                            I
E57D-20 48 E6  JSR $E648                                                      I
E580-A6 28     LDX $28         <---------------------------------------------I
E582-BD 4C 02  LDA $024C,X     on est sur un promot ?                       I I
E585-C9 7F     CMP #$7F                                                     I I
E587-D0 06     BNE $E58F                                                    I I
E589-20 6C E6  JSR $E66C       oui, on affiche le prompt                    I I
E58C-4C 97 E5  JMP $E597       et on saute ----------------------------     I I
E58F-20 56 E6  JSR $E656       non, on envoie le code sur le buffer SERIE I I I
E592-A9 09     LDA #$09        et on déplace le curseur à droite          I I I
E594-20 B5 DB  JSR $DBB5       sur la fenêtre courante                    I I I
E597-AD 24 02  LDA $0224       ,X !!! bug :on lit la fenêtre 0 ! <--------- I I
E59A-DD 34 02  CMP $0234,X     est-on en fin de page ?                      I I
E59D-D0 E1     BNE $E580       non, on boucle pour trouver le prompt--------0 I
E59F-AD 20 02  LDA $0220       fin de colonne ? (re-BUG !!!)                I I
E5A2-DD 2C 02  CMP $022C,X                                                  I I
E5A5-D0 D9     BNE $E580       non, id. ------------------------------------ I
E5A7-A4 61     LDY $61         et on positionne le curseur                    I
E5A9-A6 60     LDX $60         après le promot                                I
E5AB-20 2A E6  JSR $E62A                                                      I
E5AE-68        PLA             on sort le code <-----------------------------
E5AF-20 48 E6  JSR $E648       on l'envoie à l'écran
E5B2-18        CLC             C=0
E5B3-20 D0 E3  JSR $E3D0       et dans le buffer
```

```
E5B6-68        PLA             on récupère X
E5B7-AA        TAX
E5B8-60        RTS             et on sort
```

GERE LES CODES DE CONTROLE

```
E5B9-C9 08     CMP #$08        est-ce flèche gauche ?
E5BB-D0 18     BNE $E5D5       non ----------------------------------------------
E5BD-48        PHA                                                               I
E5BE-AD 78 02  LDA $0278       SHIFT ?                                           I
E5C1-4A        LSR                                                               I
E5C2-B0 07     BCS $E5CB       oui ------------------------------------          I
E5C4-68        PLA           ---on sort le code                          I      I
E5C5-20 48 E6  JSR $E648     I  on l'envoie                               I      I
E5C8-4C 5A E4  JMP $E45A     I  et on sort                                I      I
E5CB-20 01 E3  JSR $E301     I  on cherche le début de la ligne <---------      I
E5CE-A6 60     LDX $60       I                                                  I
E5D0-A4 61     LDY $61       I                                                  I
E5D2-4C E7 E5  JMP $E5E7     I  et on ramène le curseur                         I
E5D5-C9 09     CMP #$09      I  flèche droite ? <-------------------------------
E5D7-D0 15     BNE $E5EE     I  non ---------------------------------------------
E5D9-48        PHA           I                                                  I
E5DA-AD 78 02  LDA $0278     I  oui, SHIFT ?                                    I
E5DD-4A        LSR           I                                                  I
E5DE-90 E4     BCC $E5C4     ---non                                             I
E5E0-20 22 E3  JSR $E322       oui, on cherche la fin de la ligne               I
E5E3-A6 62     LDX $62                                                         I
E5E5-A4 63     LDY $63                                                         I
E5E7-68        PLA                                                             I
E5E8-20 2A E6  JSR $E62A       on positionne le curseur                        I
E5EB-4C 5A E4  JMP $E45A       et on sort                                      I
E5EE-C9 0A     CMP #$0A        flèche bas ? <-------------------------------------
E5F0-D0 12     BNE $E604       non ------------------------------------------------
E5F2-A6 28     LDX $28         oui                                               I
E5F4-BD 24 02  LDA $0224,X     on est en bas de la fenêtre ?                     I
E5F7-DD 34 02  CMP $0234,X                                                       I
E5FA-D0 19     BNE $E615     ---non                                             I
E5FC-A9 0A     LDA #$0A      I  oui, on indique déplacement vers le bas         I
E5FE           BYT $2C       I  et on saute...                                  I
E5FF-A9 0B     LDA #$0B      I->on indique déplacement vers le haut             I
E601-4C 79 E4  JMP $E479     II et on poursuit                                  I
E604-C9 0B     CMP #$0B      II flèche haut ? <-------------------------------------
E606-D0 0F     BNE $E617     II non ---------------------------------------------
E608-A6 28     LDX $28       II                                                 I
E60A-BD 24 02  LDA $0224,X   II début de fenêtre ?                              I
E60D-DD 30 02  CMP $0230,X   II                                                 I
E610-F0 ED     BEQ $E5FF     I--oui                                             I
E612-A9 0B     LDA #$0B      I  on indique déplacement vers le haut             I
E614           BYT $2C       I  et on saute l'instruction suivante              I
E615-A9 0A     LDA #$0A      -->on indique déplacement vers le bas              I
E617-C9 0C     CMP #$0C        CTRL-L ? <-----------------------------------------
E619-D0 09     BNE $E624       non ------------------------------------------------
E61B-20 48 E6  JSR $E648       oui, on efface l'écran                            I
E61E-20 6C E6  JSR $E66C       on affiche le prompt                              I
E621-4C 5A E4  JMP $E45A       et on boucle                                      I
E624-20 48 E6  JSR $E648       on envoie le code à l'écran <---------------------
```

```
E627-4C 5A E4  JMP $E45A       et on boucle l'édition
```

POSITIONNE LE CURSEUR EN X,Y

Action: positionne le curseur à l'écran et sur le minitel s'il est actif en tant que sortie vidéo.

```
E62A-A9 1F     LDA #$1F        on envoie un US
E62C-20 48 E6  JSR $E648
E62F-98        TYA             on envoie Y+64
E630-09 40     ORA #$40
E632-20 48 E6  JSR $E648
E635-8A        TXA             et X+64
E636-09 40     ORA #$40
E638-20 B5 DB  JSR $DBB5
E63B-2C 0D 02  BIT $020D       mode minitel ?
E63E-50 2B     BVC $E66B       non
E640-E8        INX             on ajoute une colonne
E641-8A        TXA             dans A
E642-CA        DEX             et on revient en arrière
E643-09 40     ORA #$40        on ajoute 40
E645-4C 56 E6  JMP $E656       et on envoie au minitel
```

ENVOIE UN CODE SUR LE TERMINAL VIDEO

Action: envoie un code sur l'écran et éventuellement sur le minitel s'il est actif comme sortie vidéo. Seule la fenêtre 0 est gérée, ce qui ote définitivement tout espoir de gestion d'entrée de commande sur une autre fenêtre.

```
E648-2C 0D 02  BIT $020D       mode minitel ?
E64B-50 03     BVC $E650       non ----------------------------------------
E64D-20 56 E6  JSR $E656       oui, on envoie le code au minitel            I
E650-2C 50 E6  BIT $E650       V=0 et N=0 pour écriture <-------------------
E653-4C 86 DB  JMP $DB86       dans la fenêtre 0
```

ENVOIE UN CODE AU BUFFER SERIE SORTIE

```
E656-85 0C     STA $0C         on sauve le code <-------------------------------
E658-98        TYA             on sauve Y                                     I
E659-48        PHA                                                            I
E65A-8A        TXA             et X                                           I
E65B-48        PHA                                                            I
E65C-A2 18     LDX #$18        on indexe buffer ACIA sortie (minitel sortie)  I
E65E-A5 0C     LDA $0C         on envoie le code                              I
E660-20 1D C5  JSR $C51D                                                      I
E663-68        PLA             on restaure les registres                      I
E664-AA        TAX                                                            I
E665-68        PLA                                                            I
E666-A8        TAY                                                            I
E667-A5 0C     LDA $0C                                                        I
E669-B0 EB     BCS $E656       si l'envoi s'est mal passé, on recommence ------
E66B-60        RTS
```

AFFICHE LE PROMPT

```
6C-2C 0D 02  BIT $020D       mode minitel ?
6F-50 0A     BVC $E67B       non ----------------------------------------
71-A9 19     LDA #$19        on envoie SS2 2/E au minitel                 I
73-20 56 E6  JSR $E656                                                    I
76-A9 2E     LDA #$2E        donc on affiche la flèche ->                 I
78-20 56 E6  JSR $E656                                                    I
7B-A9 7F     LDA #$7F        on affiche un prompt <-----------------------
7D-4C B5 DB  JMP $DBB5       à l'écran
```

CHERCHE UNE LIGNE D'APRES SON NUMERO

```
tion:Recherche la ligne numéro RES à partir de l'adresse SCEDEB.
     Une ligne de programme est composée de l'entête de 3 octets suivants:
     1er octet    :longeur de la ligne, ou 0 si dernière ligne
     2 et 3e octet:numéro de la ligne.
     En sortie, C=1 si la ligne a été trouvée (adresse dans RESB), 0 sinon.


80-A5 5C     LDA $5C          ,AX=adresse de base de recherche
82-A6 5D     LDX $5D
84-86 03     STX $03           dans RESB
86-85 02     STA $02       -->
88-A0 00     LDY #$00      I
8A-B1 02     LDA ($02),Y   I  on lit la longeur de la ligne
8C-F0 20     BEQ $E6AE     I  0, on sort ------------------------------------
8E-AA        TAX           I  on sauve la longueur dans X                    I
8F-A0 02     LDY #$02      I  on lit le numéro de la ligne                   I
91-A5 01     LDA $01       I                                                 I
93-D1 02     CMP ($02),Y   I  poids fort lu égal au demandé ?                I
95-90 17     BCC $E6AE     I  supérieur, on sort ----------------------------0
97-F0 02     BEQ $E69B     I  égal, on continue le test                      I
99-B0 09     BCS $E6A4     I  inférieur, on passe ----------------------     I
9B-88        DEY           I  on lit le poids faible                 I    I
9C-A5 00     LDA $00       I                                         I    I
9E-D1 02     CMP ($02),Y   I  poids faible lu égal au demandé ?      I    I
A0-90 0C     BCC $E6AE     I  supérieur, on sort ----------------------+----0
A2-F0 0B     BEQ $E6AF     I  égal, on sort avec C=1                 I    I
A4-18        CLC           I  <---------------------------------------     I
A5-8A        TXA           I                                               I
A6-65 02     ADC $02       I  on passe la ligne                            I
A8-90 DC     BCC $E686     I                                               I
AA-E6 03     INC $03       I                                               I
AC-B0 D8     BCS $E686     ---et on continue                               I
AE-18        CLC              C=0, ligne non trouvée <----------------------
AF-60        RTS
```

INSERE UNE LIGNE DANS UN LISTING

```
ction:insère la ligne numéro RES contenue à l'adresse TR0-1, de longueur A
      dans le listing commençant à l'adresse SCEDEB et finissant à l'adresse
      SCEFIN. En sortie, SCEDEB contient l'adresse de la ligne dans le listing,
      SCEFIN la nouve fin du listing et TR3-4 la différence de taille du
      listing, en complément à 2. Très pratique tout ça !


B0-85 0E     STA $0E          sauve la longueur de la ligne
B2-A9 00     LDA #$00         met 0 dans TR3-4
```

```
E6B4-85 0F      STA $0F
E6B6-85 10      STA $10
E6B8-20 80 E6   JSR $E680       cherche le numéro de la ligne à insérer
E6BB-90 2A      BCC $E6E7       la ligne n'existe pas ----------------------------
E6BD-86 0F      STX $0F         on sauve la longueur de la ligne trouvée          I
E6BF-A5 5E      LDA $5E         on met SCEFIN                                     I
E6C1-A4 5F      LDY $5F                                                           I
E6C3-85 06      STA $06         dans DECFIN                                       I
E6C5-84 07      STY $07                                                           I
E6C7-A5 02      LDA $02         adresse de la ligne trouvée                       I
E6C9-A4 03      LDY $03                                                           I
E6CB-85 08      STA $08         dans DECCIB                                       I
E6CD-84 09      STY $09                                                           I
E6CF-18         CLC                                                               I
E6D0-8A         TXA                                                               I
E6D1-65 02      ADC $02                                                           I
E6D3-90 01      BCC $E6D6       et l'adresse de la fin de la ligne                I
E6D5-C8         INY                                                               I
E6D6-85 04      STA $04         dans DECDEB                                       I
E6D8-84 05      STY $05                                                           I
E6DA-20 6C CD   JSR $CD6C       et on ramène la fin du listing (efface la ligne)  I
E6DD-A9 FF      LDA #$FF        on met -1                                         I
E6DF-85 10      STA $10         dans TR4                                          I
E6E1-45 0F      EOR $0F         on complémente TR3 à 2                            I
E6E3-85 0F      STA $0F         donc on remet dans TR3-4                          I
E6E5-E6 0F      INC $0F         l'opposé de TR3-4                                 I
E6E7-A5 0E      LDA $0E         on prend la longueur à insérer <------------------
E6E9-F0 4D      BEQ $E738       c'est 0, on devait effacer la ligne --------------
E6EB-A5 5E      LDA $5E         on prend la fin du listing                        I
E6ED-A4 5F      LDY $5F                                                           I
E6EF-85 06      STA $06         dans DECFIN                                       I
E6F1-84 07      STY $07                                                           I
E6F3-A5 02      LDA $02         on prend l'adresse de la ligne                    I
E6F5-A4 03      LDY $03                                                           I
E6F7-85 04      STA $04         dans DECDEB                                       I
E6F9-84 05      STY $05                                                           I
E6FB-18         CLC                                                               I
E6FC-A5 0E      LDA $0E         on ajoute 3 à la longueur (entête de ligne)       I
E6FE-69 03      ADC #$03                                                          I
E700-48         PHA             dans la pile                                      I
E701-65 02      ADC $02         on ajoute la longueur                             I
E703-90 01      BCC $E706       à DECDEB                                          I
E705-C8         INY                                                               I
E706-85 08      STA $08         dans DECCIB                                       I
E708-84 09      STY $09                                                           I
E70A-20 6C CD   JSR $CD6C       et on libère la place pour la ligne               I
E70D-18         CLC                                                               I
E70E-68         PLA                                                               I
E70F-48         PHA             on prend la longueur                              I
E710-65 0F      ADC $0F         on calcule longueur nouvelle ligne                I
E712-85 0F      STA $0F         - longueur ligne précédente                       I
E714-90 02      BCC $E718                                                         I
E716-E6 10      INC $10         dans TR3-4 (complément à 2)                       I
E718-A0 00      LDY #$00        on écrit la longueur de la ligne                  I
E71A-68         PLA                                                               I
E71B-91 02      STA ($02),Y                                                       I
E71D-C8         INY                                                               I
E71E-A5 00      LDA $00                                                           I
E720-91 02      STA ($02),Y     le poids faible du numéro de ligne                I
E722-C8         INY                                                               I
```

```
E723-A5 01     LDA $01
E725-91 02     STA ($02),Y     le poids fort du numéro de ligne
E727-A2 00     LDX #$00
E729-C8        INY
E72A-A1 0C     LDA ($0C,X)     et le contenu de la ligne
E72C-91 02     STA ($02),Y     à la suite
E72E-E6 0C     INC $0C
E730-D0 02     BNE $E734
E732-E6 0D     INC $0D
E734-C6 0E     DEC $0E         jusqu'à la fin
E736-D0 F1     BNE $E729
E738-18        CLC             <-----------------------------------------
E739-A5 0F     LDA $0F         on calcule dans SCEFIN
E73B-65 5E     ADC $5E
E73D-85 5E     STA $5E
E73F-A4 10     LDY $10         la nouvelle adresse de fin du listing
E741-98        TYA
E742-65 5F     ADC $5F
E744-85 5F     STA $5F
E746-A5 0F     LDA $0F         et AY=différence de longuer des lignes
E748-60        RTS
```

CONVERSION ASCII -> BINAIRE

Principe: On lit un à un les chiffres de la chaine stockée en AY jusqu'à ce qu'on ait plus de chiffres. On multiplie au fur et à mesure le resultat par 10 avant d'ajouter le chiffre trouvé. Le principe est aisé à assimiler et la routine compacte. Un bon exemple d'optimisation.
En sortie, AY et RESB contient le nombre, AY l'adresse de la chaine, et X le nombre de caractères décodés.

```
E749-85 00     STA $00         on sauve l'adresse du nombre
E74B-84 01     STY $01         dans RES
E74D-A0 00     LDY #$00        et on met RESB à 0
E74F-84 02     STY $02
E751-84 03     STY $03
E753-B1 00     LDA ($00),Y     on lit le code <-----------------------------
E755-C9 30     CMP #$30        inférieur à 0 ?                          I
E757-90 2C     BCC $E785       oui ------------------------------------+----
E759-C9 3A     CMP #$3A        supérieur à 9 ?                          I   I
E75B-B0 28     BCS $E785       oui ------------------------------------+---0
E75D-29 0F     AND #$0F        on isole le chiffre                      I   I
E75F-48        PHA             dans la pille                            I   I
E760-06 02     ASL $02         RESB*2                                   I   I
E762-26 03     ROL $03                                                  I   I
E764-A5 02     LDA $02         AX=RESB*2                                I   I
E766-A6 03     LDX $03                                                  I   I
E768-06 02     ASL $02         *4                                       I   I
E76A-26 03     ROL $03                                                  I   I
E76C-06 02     ASL $02         *8                                       I   I
E76E-26 03     ROL $03                                                  I   I
E770-65 02     ADC $02         +RESB*2                                  I   I
E772-85 02     STA $02                                                  I   I
E774-8A        TXA                                                      I   I
E775-65 03     ADC $03                                                  I   I
E777-85 03     STA $03         = RESB*10                                I   I
```

```
E779-68       PLA             plus chiffre lu                              I   I
E77A-65 02    ADC $02                                                      I   I
E77C-85 02    STA $02                                                      I   I
E77E-90 02    BCC $E782                                                    I   I
E780-E6 03    INC $03                                                      I   I
E782-C8       INY             on ajoute un chiffre lu                      I   I
E783-D0 CE    BNE $E753       et on recommence ----------------------------    I
E785-98       TYA             nombre de chiffres lus <-------------------------
E786-AA       TAX             dans X
E787-A5 02    LDA $02         nombre dans AY et RESB
E789-A4 03    LDY $03
E78B-60       RTS
```

DONNEES POUR AFFICHAGE HIRES

```
E78C          BYT 32,16,8,4,2,1  position des bits 5,4,3,2,1 et 0 pour masque.
```

AFFICHE LE CURSEUR HIRES

Action:si HRS5+1 est négatif, positionne le curseur selon HRS40,HRSX6 et ADHRS.
le FB est pris en compte dans b7b6 de HRSFB.
En fait HRS5+1 contient le pattern, d'où la rotation totale.

```
E792-18       CLC             C=0
E793-24 56    BIT $56         on fait tourner HRS5+1 sur lui-même
E795-10 01    BPL $E798       afin de conserver le pattern
E797-38       SEC
E798-26 56    ROL $56
E79A-90 24    BCC $E7C0       si b7 de $56   à 0, on saute <---------------
E79C-A4 49    LDY $49         sinon on prend X/6                           I
E79E-B1 4B    LDA ($4B),Y     on lit le code actuel                        I
E7A0-0A       ASL             on sort b7                                   I
E7A1-10 1D    BPL $E7C0       pas pixel, on sort --------------------------0
E7A3-A6 4A    LDX $4A         on prend le reste de X/6                     I
E7A5-BD 8C E7 LDA $E78C,X     on lit le bit correspondant                  I
E7A8-24 57    BIT $57         b7 de HRSFB à 1 ?                            I
E7AA-30 0E    BMI $E7BA       b7 à 1, donc 3 ou 2                          I
E7AC-50 05    BVC $E7B3       FB=0 ----------------------------------------I
E7AE-11 4B    ORA ($4B),Y     FB=1, on ajoute le code                  I   I
E7B0-91 4B    STA ($4B),Y     et on le place                           I   I
E7B2-60       RTS                                                      I   I
E7B3-49 7F    EOR #$7F        on inverse le bit <-------------------------  I
E7B5-31 4B    AND ($4B),Y     et on l'éteint                               I
E7B7-91 4B    STA ($4B),Y     avant de le placer                           I
E7B9-60       RTS                                                          I
E7BA-70 04    BVS $E7C0       FB=3, on sort -------------------------------0
E7BC-51 4B    EOR ($4B),Y     FB=2, on inverse le bit                      I
E7BE-91 4B    STA ($4B),Y     et on sort                                   I
E7C0-60       RTS             <--------------------------------------------
```

DEPLACEMENT RELATIF DU CURSEUR HIRES

Action:Ces quatres routines permettent un déplacement extrêmement rapide du curseur HIRES d'après l'adresse de la ligne ou il se trouve (ADHRS), la colonne dans laquelle il se trouve (HRSX40) et sa position dans l'octet pointé (HRSX6).
Attention:Les coordonnées HRSX et HRSY ne sont pas modifiées ni vérifiées avant le déplacement, à vous de gérer cela.

DEPLACE LE CURSEUR HIRES VERS LE BAS

```
E7C1-18        CLC             on ajoute 40
E7C2-A5 4B     LDA $4B         à ADHRS
E7C4-69 28     ADC #$28
E7C6-85 4B     STA $4B
E7C8-90 F6     BCC $E7C0
E7CA-E6 4C     INC $4C
E7CC-60        RTS             et on sort
```

DEPLACE LE CURSEUR HIRES VERS LE HAUT

```
E7CD-38        SEC             on soustrait 40
E7CE-A5 4B     LDA $4B         à ADHRS
E7D0-E9 28     SBC #$28
E7D2-85 4B     STA $4B
E7D4-B0 EA     BCS $E7C0
E7D6-C6 4C     DEC $4C
E7D8-60        RTS
```

DEPLACE LE CURSEUR VERS LA DROITE

```
E7D9-A6 4A     LDX $4A         on déplace d'un pixel
E7DB-E8        INX
E7DC-E0 06     CPX #$06        si on est à la fin
E7DE-D0 04     BNE $E7E4
E7E0-A2 00     LDX #$00        on revient au début
E7E2-E6 49     INC $49         et ajoute une colonne
E7E4-86 4A     STX $4A
E7E6-60        RTS
```

DEPLACE LE CURSEUR VERS LA GAUCHE

```
E7E7-A6 4A     LDX $4A
E7E9-CA        DEX             on déplace à gauche
E7EA-10 04     BPL $E7F0       si on sort
E7EC-A2 05     LDX #$05        on se place à droite
E7EE-C6 49     DEC $49         et on enlève une colonne
E7F0-86 4A     STX $4A
E7F2-60        RTS
```

PLACE LE CURSEUR EN X,Y

Action:calcule l'adresse du curseur en calculant la position de la ligne par $A000+40*Y, la colonne dans X/6 et la position dans l'octet par X mod 6. Suite à une erreur dans la table des vecteur TELEMON, cette routine n'est pas appelée (alors qu'elle devrait l'être) par BRK XHRSSE... En sortie, HSRX,Y,X40,X6 et ADHRS sont ajustés en fonction de X et Y.

```
E7F3-84 47     STY $47         Y dans HRSY
E7F5-86 46     STX $46         X dans HRSX
E7F7-98        TYA             et Y dans A
E7F8-A0 00     LDY #$00        AY=A, ligne du curseur
E7FA-20 69 CE  JSR $CE69       on calcule 40*ligne
E7FD-85 4B     STA $4B
E7FF-18        CLC
E800-98        TYA
```

```
E801-69 A0      ADC #$A0        et on ajoute $A000, écran HIRES
E803-85 4C      STA $4C         dans ADHRS
E805-86 00      STX $00         on met la colonne dans RES
E807-A9 06      LDA #$06        A=6
E809-A0 00      LDY #$00        et Y=0  (dans RES+1)
E80B-84 01      STY $01         AY=6 et RES=colonne
E80D-20 DC CE   JSR $CEDC       on divise la colonne par 6
E810-A5 00      LDA $00         on sauve colonne/6 dans HSRX40
E812-85 49      STA $49
E814-A5 02      LDA $02         et le reste dans HRSX6
E816-85 4A      STA $4A
E818-60         RTS
```

TRACE UN RECTANGLE EN RELATIF

Principe:On calcule les coordonnées absolues des 4 coins et on trace en absolu.
Pas très optimisé en temps tout cela, il aurait été plus simple de
de tracer directement en relatif !!!
Le rectangle est tracé comme ABOX avec les paramètres dans HRSx.

```
E819-18         CLC             C=0
E81A-A5 46      LDA $46         on place les coordonées actuelles
E81C-85 06      STA $06         du curseur dans $06-07
E81E-65 4D      ADC $4D         et les coordonnées (X+dX,Y+dY)
E820-85 08      STA $08
E822-A5 47      LDA $47
E824-85 07      STA $07
E826-65 4F      ADC $4F
E828-85 09      STA $09         dans $08-09
E82A-90 0E      BCC $E83A       inconditionnel
```

TRACE UN RECTANGLE ABSOLU

Principe:Par un procédé très astucieux, on va tracer les 4 traits (en absolu)
joignant les 4 points. Voila bien la seule astuce inutile ! Il aurait
été 100 (pourquoi pas 1000 !?) fois plus simple, puisque le rectangle
n'est fait que de verticales et d'horizontales, de tracer le rectangle
immédiatement en relatif plutot que de passer par des calculs de
tangentes lourds et donnant un résultat connu (0 et infini) !!!
Cette piètre routine nécessite les paramètres comme ABOX dans HRSx.

```
E82C-A0 06      LDY #$06        on place les 4 paramètres (poids faible seulement)
E82E-A2 03      LDX #$03
E830-B9 4D 00   LDA $004D,Y     de HRSx
E833-95 06      STA $06,X       dans $06-7-8-9
E835-88         DEY
E836-88         DEY
E837-CA         DEX
E838-10 F6      BPL $E830
E83A-A2 03      LDX #$03        on va tracer 4 traits
E83C-86 05      STX $05         dans $05 <----------------------------------------
E83E-BD 62 E8   LDA $E862,X     on lit le code coordonnées                        I
E841-85 04      STA $04         dans $04                                          I
E843-A2 06      LDX #$06        on va extraire 8 bits                             I
```

```
E845-A9 00      LDA #$00        A=0 <-------------------------------------
E847-95 4E      STA $4E,X       poids fort HRSx à 0 et positif             I
E849-46 04      LSR $04         on sort 2 bits                             I
E84B-2A         ROL             dans A                                     I
E84C-46 04      LSR $04                                                    I
E84E-2A         ROL                                                        I
E84F-A8         TAY             et Y                                       I
E850-B9 06 00   LDA $0006,Y     on lit la coordonnée correspondante        I
E853-95 4D      STA $4D,X       et on stocke dans HRSx                     I
E855-CA         DEX                                                        I
E856-CA         DEX                                                        I
E857-10 EC      BPL $E845       on fait les 4 coordonnées ADRAW ------------
E859-20 66 E8   JSR $E866       on trace le trait en absolu
E85C-A6 05      LDX $05
E85E-CA         DEX                                                        1
E85F-10 DB      BPL $E83C       et on fait 4 traits ------------------------
E861-60         RTS
```

TABLES POUR TRACE DE RECTANGLES

Principe: Chaque octet contient, codée la séquence de coordonnées à envoyer à la routine de tracé de traits absolus. Dans un octet, pour trouver la séquence de coordonnées, on isole les couples de bits de gauche à droite, on inverse les deux bits et on calcule en décimal. On obtient alors 0,1,2,3 pour respectivement X1,Y1,X2 et Y2. En fait, les Xi,Yi sont stockés en $06-7-8-9 et les valeurs trouvées sont les index de position des coordonnées par rapport à $06.

```
E862            BYT %00100110   soit 0,1,2,1 ou X1,Y1,X2,Y1
E863            BYT %01100111   soit 2,1,2,3 ou X2,Y1,X2,Y2
E864            BYT %01110011   soit 2,3,0,3 ou X2,Y2,X1,Y2
E865            BYT %00110010   soit 0,3,0,1 ou X1,Y2,X1,Y1
```

TRACE DE TRAIT EN ABSOLU

Action: on calcule dX et dY les déplacements dans HRS1 et HRS2 et on trace en relatif. En entrée, comme ADRAW dans HRSx.

```
E866-A6 4D      LDX $4D         X=colonne
E868-A4 4F      LDY $4F         Y=ligne du curseur
E86A-20 F3 E7   JSR $E7F3       on place le curseur en X,Y
E86D-A2 FF      LDX #$FF        on met -1 dans X pour un changement de signe
E86F-38         SEC             éventuel dans les paramètres
E870-A5 51      LDA $51         on prend X2
E872-E5 4D      SBC $4D         -X1
E874-85 4D      STA $4D         dans HRS1 (DX)
E876-B0 03      BCS $E87B       si DX<0, on inverse le signe de HRS1
E878-86 4E      STX $4E         DEC $4E aurait été mieux...
E87A-38         SEC
E87B-A5 53      LDA $53         on prend Y2
E87D-E5 4F      SBC $4F         -Y1
E87F-85 4F      STA $4F         dans HRS2 (DY)
E881-B0 02      BCS $E885       et si DY négatif, on met signe -1
E883-86 50      STX $50         ou DEC $50
```

TRACE DE TRAIT EN RELATIF

Principe:Le principe du tracé des droites est en fait assez complexe. On aurait aimé que F. BROCHE nous ponde une routine hyper-optimisée dont il a le secret. Ce n'est malheureusement pas le cas puisque cette routine l'algorithme des ROM V1.0 et 1.1. Sans doute parce qu'il est très efficace...
Pour tracer un trait le plus rapidement possible, on cherche lequel des deux axes est le plus grand et on trace selon cet axe. Pour tracer, on avance sur l'axe de t points (t est la valeur de la tangente) et on avance d'un point sur l'autre axe, et ainsi de suite jusqu'à ce qu'on ait parcouru tout l'axe.
Ainsi DRAW 10,2,1 donnera en fait 2 paliers de 5 pixels de large.
Le cas dX=dY (déplacements égaux) est traité avec t=-1, de plus les poids fort des déplacements gardent le signe car on prend la valeur absolue de dX et dY pour les calculs.

```
E885-AD AA 02  LDA $02AA      sauve le pattern
E888-85 56     STA $56        dans HRS1+1
E88A-20 42 E9  JSR $E942      vérifie la validité de dX et dY
E88D-86 46     STX $46        X et Y contiennent HRSX+dX et HRSY+dY
E88F-84 47     STY $47        dans HRSX et HRSY
E891-24 4E     BIT $4E        dX négatif ?
E893-10 08     BPL $E89D      non ----------------------------------------------
E895-A5 4D     LDA $4D        oui, on complémente                                I
E897-49 FF     EOR #$FF       dX                                                 I
E899-85 4D     STA $4D                                                           I
E89B-E6 4D     INC $4D        à 2                                                I
E89D-24 50     BIT $50        dY négatif ? <-------------------------------------
E89F-10 08     BPL $E8A9      non ----------------------------------------------
E8A1-A5 4F     LDA $4F        oui on complémente                                 I
E8A3-49 FF     EOR #$FF       dY                                                 I
E8A5-85 4F     STA $4F                                                           I
E8A7-E6 4F     INC $4F        à 2                                                I
E8A9-A5 4D     LDA $4D        on teste dX et dY <-------------------------------
E8AB-C5 4F     CMP $4F
E8AD-90 3E     BCC $E8ED      dX<dY --------------------------------------------
E8AF-08        PHP            dX>=dY , on trace selon dX                         I
E8B0-A5 4D     LDA $4D        on prends dX                                       I
E8B2-F0 37     BEQ $E8EB      dX=0, on sort ------------------------------       I
E8B4-A6 4F     LDX $4F        X=dY                                       I       I
E8B6-20 21 E9  JSR $E921      on calcule dY/dX                           I       I
E8B9-28        PLP                                                       I       I
E8BA-D0 04     BNE $E8C0      dX<>dY -------------------------------    I       I
E8BC-A9 FF     LDA #$FF       dX=dY, la tangente est 1                I  I       I
E8BE-85 00     STA $00        en fait, -1, mais c'est la même chose   I  I       I
E8C0-24 4E     BIT $4E    --->on teste dX <-------------------------  I       I
E8C2-10 06     BPL $E8CA  I   dX>0 ---------------------------------  I       I
E8C4-20 E7 E7  JSR $E7E7  I   dX<0, on déplace le curseur à gauche   I  I       I
E8C7-4C CD E8  JMP $E8CD  I---                                       I  I       I
E8CA-20 D9 E7  JSR $E7D9  II  on on déplace le curseur à droite <-----  I       I
E8CD-18        CLC        I-->a-t-on parcouru une valeur de la tangente  I       I
E8CE-A5 00     LDA $00    I                                              I       I
E8D0-65 02     ADC $02    I   on stocke le résultat dans $02             I       I
E8D2-85 02     STA $02    I                                              I       I
E8D4-90 0D     BCC $E8E3  I   non, on continue ------------------------  I       I
E8D6-24 50     BIT $50    I   oui, dY<0 ?                             I  I       I
E8D8-30 06     BMI $E8E0  I   oui ------------------------------      I  I       I
E8DA-20 C1 E7  JSR $E7C1  I   non, on déplace le curseur         I     I  I       I
E8DD-4C E3 E8  JMP $E8E3  I---vers le bas                        I     I  I       I
E8E0-20 CD E7  JSR $E7CD  II  on déplace vers le haut <----------     I  I       I
```

```
E8E3-20 92 E7   JSR $E792   I-->on affiche le point <----------------------  I
E8E6-C6 4D      DEC $4D    I   on décremente dX,                              I
E8E8-D0 D6      BNE $E8C0  ----on n'a pas parcouru tout l'axe                I
E8EA-60         RTS         -->sinon, on sort                               I
E8EB-28         PLP         I  <-------------------------------------------
E8EC-60         RTS         I
E8ED-A5 4F      LDA $4F     I  on trace la droite selon dY <-----------------
E8EF-F0 F9      BEQ $E8EA   ---dY=0, on sort
E8F1-A6 4D      LDX $4D        X=dX
E8F3-20 21 E9   JSR $E921      on calcule dX/dY dans RES
E8F6-24 50      BIT $50
E8F8-10 06      BPL $E900      dY>0 ------------------------------------------
E8FA-20 CD E7   JSR $E7CD      dY<0, on déplace vers le haut                   I
E8FD-4C 03 E9   JMP $E903  ---et on saute                                     I
E900-20 C1 E7   JSR $E7C1  I  on déplace vers le bas <-----------------------
E903-18         CLC        -->a-t-on parcouru la tangente ?
E904-A5 00      LDA $00
E906-65 02      ADC $02
E908-85 02      STA $02        (dans $02)
E90A-90 0D      BCC $E919      non -------------------------------------------
E90C-24 4E      BIT $4E                                                       I
E90E-10 06      BPL $E916      dX>0 ----------------------------------       I
E910-20 E7 E7   JSR $E7E7      dX<0, on déplace vers                  I      I
E913-4C 19 E9   JMP $E919  ---la gauche                                I      I
E916-20 D9 E7   JSR $E7D9  I  on déplace vers la droite <--------------       I
E919-20 92 E7   JSR $E792  -->on affiche le point <-----------------------------
E91C-C6 4F      DEC $4F        et on décrit dY
E91E-D0 D6      BNE $E8F6
E920-60         RTS            avant de sortir
```

CALCUL LA TANGENTE (*256) D'UN TRAIT

```
E921-86 01      STX $01        dX (ou dY)*256 dans RES+1
E923-A0 00      LDY #$00       dY (ou dX) dans AY
E925-84 00      STY $00
E927-20 DC CE   JSR $CEDC      calcul dX*256/dY (ou dY/dX)
E92A-A9 FF      LDA #$FF       reste =-1
E92C-85 02      STA $02        resultat dans RES
E92E-60         RTS
```

ROUTINE CURSET

```
E92F-A6 4D      LDX $4D        X=HRSX
E931-A4 4F      LDY $4F        Y=HRSY
E933-20 4E E9   JSR $E94E      on vérifie les coordonnées
E936-20 F3 E7   JSR $E7F3      on place le curseur en X,Y
E939-4C 9C E7   JMP $E79C      et on affiche sans gérer pattern
```

ROUTINE CURMOV

```
E93C-20 42 E9   JSR $E942      on vérifie les paramètres
E93F-4C 36 E9   JMP $E936      et on déplace
```

VERIFIE LA VALIDITE DES PARAMETRES RELATIFS

```
Action:Vérifie si l'adressage relatif du curseur est dans les limites de l'écran
       HIRES, soit si 0<=X+dX<240 et 0<=Y+dY<200.

E942-18         CLC
```

```
E943-A5 46      LDA $46         on prend HRSX
E945-65 4D      ADC $4D         plus le déplacement horizontal
E947-AA         TAX             dans X
E948-18         CLC
E949-A5 47      LDA $47         HRSY
E94B-65 4F      ADC $4F         plus le déplacement vertical
E94D-A8         TAY             dans Y
```

TESTE SI X ET Y SONT VALIDES

Principe:Si X>239 ou Y>199 alors on ne retourne pas au programme appelant, mais à son appelant, en indiquant l'erreur dans HRSERR.

```
E94E-E0 F0      CPX #$F0        X>=240 ?
E950-B0 05      BCS $E957       oui ----------------------------------------
E952-C0 C8      CPY #$C8        Y>=200 ?                                     I
E954-B0 01      BCS $E957       oui ----------------------------------------0
E956-60         RTS             coordonnées ok, on sort.                     I
E957-68         PLA             on dépile poids fort (>0) <-----------------
E958-8D AB 02   STA $02AB       dans HRSERR
E95B-68         PLA             et poids faible de l'adresse de retour
E95C-60         RTS             et on retourne à l'appelant de l'appelant
```

ROUTINE PAPER

```
E95D-18         CLC
E95E            BYT $24
```

ROUTINE INK

```
E95F-38         SEC
```

FIXE LA COULEUR DE FOND OU DU TEXTE

Principe:A contient la couleur, X la fenêtre ou 128 si mode HIRES et C=1 si la couleur est pour l'encre, 0 pour le fond.
Changer la couleur consiste à remplit la colonne couleur correspondante avec le code de couleur. Aucun test de validité n'étant fait, on peut utiliser ce moyen pour remplir les colonnes 0 et 1 de n'importe quel attribut.

```
E960-48         PHA             on sauve la couleur
E961-08         PHP             et C
E962-86 00      STX $00         fenêtre dans RES
E964-24 00      BIT $00         HIRES ?
E966-30 3F      BMI $E9A7       oui ----------------------------------------
E968-86 28      STX $28         TEXT, on met le numéro de fenêtre dans $28   I
E96A-90 05      BCC $E971       si C=0, c'est PAPER                          I
E96C-9D 40 02   STA $0240,X    on stocke la couleur d'encre                 I
E96F-B0 03      BCS $E974       si C=1 c'est INK                             I
E971-9D 44 02   STA $0244,X    ou la couleur de fond                        I
E974-BD 48 02   LDA $0248,X    est on en 38 colonnes ?                      I
E977-29 10      AND #$10                                                     I
E979-D0 0C      BNE $E987       mode 38 colonnes ---------------------------I
E97B-A9 0C      LDA #$0C        mode 40 colonnes, on efface l'écran        I I
E97D-20 B5 DB   JSR $DBB5       (on envoie CHR$(12))                       I I
```

```
E980-A9 1D     LDA #$1D        et on passe en 38 colonnes                    I
E982-20 B5 DB  JSR $DBB5       (on envoie CHR$(29))                          I
E985-A6 28     LDX $28         on prend X=numéro de fenêtre                  I
E987-BD 30 02  LDA $0230,X     on prend la ligne 0 de la fenêtre <-----------
E98A-20 69 CE  JSR $CE69       #40 dans RES
E98D-BD 38 02  LDA $0238,X     AY=adresse de base de la fenêtre
E990-BC 3C 02  LDY $023C,X
E993-20 89 CE  JSR $CE89       on ajoute l'adresse à RES (ligne 0 #40) dans RES
E996-BC 28 02  LDY $0228,X     on prend la première colonne de la fenêtre
E999-88        DEY             on enlève deux colonnes
E99A-88        DEY
E99B-38        SEC
E99C-BD 34 02  LDA $0234,X     on calcule le nombre de lignes
E99F-FD 30 02  SBC $0230,X     de la fenêtre
E9A2-AA        TAX             dans X
E9A3-E8        INX
E9A4-98        TYA             colonne 0 dans Y
E9A5-B0 0C     BCS $E9B3       inconditionnel ------------------------------
E9A7-A9 00     LDA #$00        <-------------------------------------------+
E9A9-A2 A0     LDX #$A0                                                    I
E9AB-85 00     STA $00         RES=$A000 , adresse HIRES                   I
E9AD-86 01     STX $01                                                     I
E9AF-A2 C8     LDX #$C8        X=200 pour 200 lignes                       I
E9B1-A9 00     LDA #$00        A=0 pour colonne de début = colonne 0       I
E9B3-28        PLP             on sort C <---------------------------------
E9B4-69 00     ADC #$00        A=A+C
E9B6-A8        TAY             dans Y
E9B7-68        PLA             on sort le code
E9B8-91 00     STA ($00),Y -->on le place dans la colonne correspondante
E9BA-48        PHA          I  on le sauve
E9BB-18        CLC          I
E9BC-A5 00     LDA $00      I  on passe 28 colonnes
E9BE-69 28     ADC #$28     I  (donc une ligne)
E9C0-85 00     STA $00      I
E9C2-90 02     BCC $E9C6    I
E9C4-E6 01     INC $01      I
E9C6-68        PLA          I  on sort le code
E9C7-CA        DEX          I  on compte X lignes
E9C8-D0 EE     BNE $E9B8    ---
E9CA-60        RTS             et on sort
```

TRACE UN CERCLE

Principe:Pour tracer une ellipsoide en général, on utilise la formule :

(X*X)/(A*A)+(Y*Y)/(B*B)=1, A et B étant respectivement la largeur et la hauteur de l'ellipse. Pour un cercle, A=B donc on écrit :

X*X+Y*Y=R*R soit encore X=SQR(R*R-Y*Y).

Pour tracer le cercle, il suffit de faire varier Y de 0 à R. On obtient des valeurs positives de X et de Y donc la quart inférieur droit du cercle. On trace les 3 autres quarts par symétries. Le problème d'un tel algorithme c'est qu'il nécessite le calcul d'une exponentiation (SQR(A)=A^0.5) et une soustraction décimale.
Son atout est de n'avoir à calculer qu'un quart des valeurs.

Les concepteurs de l'ATMOS (et à fortiori F. BROCHE) ayant jugé que cet algorithme était par trop complexe et laborieux, on préféré le calcul par suites croisées dont la formule est :

X0=0 et Xn=X(n-1)+Yn/R (n et n-1 sont les indices des termes X et Y)
Y0=R et Yn=Y(n-1)-Xn/R

Etant donnée la priorité de calcul, on calcule en fait les termes :

Xn = Xn-1 + Yn-1 / R
Yn = Yn-1 - Xn / R ce qui fait déja une petite erreur de calcul.

De plus, diviser à chaque fois par R serait long. Les programmeurs, et c'est là leur génie, ont donc pensé à deux choses fort astucieuses:

a) on divisera non pas par R, mais par la puissance de deux immédiatement supérieure à R afin de se ramener à des décalages. on devient ainsi trop précis, ce qui rattrape l'erreur passée.

b) on va coder Xn et Yn sur deux octets qui seront se et sf, respectivement les parties entières et décimale de Xn et Yn. on calcule Xn=AS par Xn=A+B/256. Ce qui revient en fait à considérer les 8 bits de B (b7b6b5b4b3b2b1b0) comme des bits de puissance négatives décroissantes (b-1b-2b-3b-4b-5b-6b-7b-8). La précision est donc inférieure a 2^-9, soit à 0,002. Ce qui est très suffisant.

Une fois ces deux conventions posées, on peut tracer le cercle très facilement. Son aspect sera de symétrie diagonales et non verticale/horizontale du fait de la quadrature exercée sur les valeurs mais bon. Pour tracer, on calcule un par un les termes des suites et si la valeur entière d'un des termes au moins change, on affiche le point. Et on continue jusqu'à ce que Xn et Yn soit revenus à leur position initiale.

Remarque:La routine est buggée, en effet si le rayon est 0, la boucle de calcul de la puissance de 2 > au rayon est infinie, idem si le rayon est 128. Il aurait suffit d'incrémenter le rayon avant le calcul...

```
9CB-A5 46     LDA $46        on sauve HRSX
9CD-48        PHA
9CE-A5 47     LDA $47        et HRSY
9D0-48        PHA
9D1-A9 AA 02  LDA #02AA      et on met le pattern dans $56
9D4-85 56     STA $56        car le tracé du cercle en tient compte
9D6-A5 47     LDA $47        on prend HRSY
9D8-38        SEC
9D9-E5 4D     SBC $4D        -rayon
9DB-A8        TAY            dans Y
9DC-A6 46     LDX $46        on prend HRSX
9DE-20 F3 E7  JSR $E7F3      et on place le premier point du cercle (X,Y-R)
9E1-A2 08     LDX #$08       X=7+1 pour calculer N tel que Rayon<2^N.
9E3-A5 4D     LDA $4D        on prend le rayon
9E5-CA        DEX            on enlève une puissance
9E6-0A        ASL            on décale le rayon à gauche
9E7-10 FC     BPL $9E5       jusqu'à ce qu'un bit se présente dans b7
9E9-86 0C     STX $0C        exposant du rayon dans $0C
9EB-A9 80     LDA #$80       A=$80 soit 0,5 en décimal
9ED-85 0E     STA $0E        dans sfX
9EF-85 10     STA $10        et sfY
9F1-0A        ASL            A=0
```

```
E9F2-85 0F     STA $0F        dans seX
E9F4-A5 4D     LDA $4D        A=Rayon
E9F6-85 11     STA $11        dans seY
E9F8-38        SEC
E9F9-66 0D     ROR $0D        on met b7 de $0D à 1 (ne pas afficherle point)
E9FB-A5 10     LDA $10        AX=sY
E9FD-A6 11     LDX $11
E9FF-20 62 EA  JSR $EA62      on calcule sY/R (en fait sY/2^N)
EA02-18        CLC
EA03-A5 0E     LDA $0E        on calcule sX=sX+sY/R
EA05-65 12     ADC $12
EA07-85 0E     STA $0E
EA09-A5 0F     LDA $0F
EA0B-85 12     STA $12
EA0D-65 13     ADC $13
EA0F-85 0F     STA $0F        la partie entière seX a bougé ?
EA11-C5 12     CMP $12
EA13-F0 0D     BEQ $EA22      non ----------------------------------------------
EA15-B0 06     BCS $EA1D      elle a augmenté ------------------------------    I
EA17-20 D9 E7  JSR $E7D9      elle à baissé, on déplace le curseur        I    I
EA1A-4C 20 EA  JMP $EA20    ---à droite                                   I    I
EA1D-20 E7 E7  JSR $E7E7    I  on déplace le curseur à gauche <------------    I
EA20-46 0D     LSR $0D      -->on indique qu'il faut afficher le point         I
EA22-A5 0E     LDA $0E        AX=sX <-------------------------------------------
EA24-A6 0F     LDX $0F
EA26-20 62 EA  JSR $EA62      on calcule sX/R (en fait sX/2^N)
EA29-38        SEC
EA2A-A5 10     LDA $10        et sY=sY-sX/R
EA2C-E5 12     SBC $12
EA2E-85 10     STA $10
EA30-A5 11     LDA $11
EA32-85 12     STA $12
EA34-E5 13     SBC $13
EA36-85 11     STA $11        seY à changé (faut-il se déplacer verticalement)?
EA38-C5 12     CMP $12
EA3A-F0 0E     BEQ $EA4A      non ----------------------------------------------
EA3C-B0 06     BCS $EA44      on est monté ---------------------------------    I
EA3E-20 C1 E7  JSR $E7C1      on est descendu, on déplace le curseur       I    I
EA41-4C 4E EA  JMP $EA4E    ---vers le bas et on affiche                   I    I
EA44-20 CD E7  JSR $E7CD    I  on déplace le curseur vers le haut <---------    I
EA47-4C 4E EA  JMP $EA4E    O--et on affiche                                    I
EA4A-24 0D     BIT $0D      I  faut-il afficher le point ? <--------------------
EA4C-30 03     BMI $EA51    I  non, on passe  ------------------------------------
EA4E-20 92 E7  JSR $E792    -->on affiche le point nouvellement calculé          I
EA51-A5 0F     LDA $0F        seX=0 ? <-------------------------------------------
EA53-D0 A3     BNE $E9F8      non, on boucle
EA55-A5 11     LDA $11        oui, seY=rayon?
EA57-C5 4D     CMP $4D
EA59-D0 9D     BNE $E9F8      non, on boucle
EA5B-68        PLA            oui, on a fait le tour
EA5C-A8        TAY            on reprend les coordonnées du curseur sauvées
EA5D-68        PLA            dans X et Y
EA5E-AA        TAX
EA5F-4C F3 E7  JMP $E7F3      et on replace le curseur
```

CALCUL LE DEPLACEMENT sX ou sY

Action:calcule dans $13,$12 la valeur de (X,A)/R, en fait (X,A)/2^N.

```
EA62-85 12     STA $12        on place la partie fractionnaire dans $12
```

```
EA64-86 13      STX $13         et la partie entière dans $13
EA66-A6 0C      LDX $0C         X=N tel que Rayon<2^N
EA68-A5 13      LDA $13         on garde le signe du résultat
EA6A-2A         ROL
EA6B-66 13      ROR $13         et on divise par 2^X
EA6D-66 12      ROR $12         dans $13,$12
EA6F-CA         DEX
EA70-D0 F6      BNE $EA68
EA72-60         RTS
```

COMMANDE FILL

Action:écrit HRS1 lignes de HRS2 codes HRS3 à partir des coordonnées courantes du curseur. Le code HRS3 peut-être n'importe quel code. S'il à son bit 6 à 1, ce sera des pixels, son bit 7 à 1 ce sera en inverse vidéo. Si son bit 6 est à 0, ce sera un code de controle (0-31) ou un autre non défini (32-63).

```
EA73-A5 4B      LDA $4B         AY=adresse de la ligne du curseur
EA75-A4 4C      LDY $4C
EA77-85 00      STA $00         dans RES
EA79-84 01      STY $01
EA7B-A6 4F      LDX $4F         X=nombre de colonnes
EA7D-A4 49      LDY $49         Y=position du curseur
EA7F-A5 51      LDA $51
EA81-91 00      STA ($00),Y     on écrit X colonnes
EA83-C8         INY
EA84-CA         DEX
EA85-D0 FA      BNE $EA81
EA87-A9 28      LDA #$28        on ajoute une ligne
EA89-A0 00      LDY #$00        à l'adresse
EA8B-20 89 CE   JSR $CE89
EA8E-C6 4D      DEC $4D         et on fait toutes les lignes
EA90-D0 E9      BNE $EA7B
EA92-60         RTS
```

COMMANDE SCHAR

Action:Affiche une chaine de caractères en HIRES par envois successifs des caractères de la chaine. Le FB est forcé à 1, dommage...

```
EA93-85 51      STA $51         on sauve l'adresse de la chaine dans HRS3
EA95-84 52      STY $52
EA97-86 4F      STX $4F         et sa longueur dans HRS2
EA99-A9 40      LDA #$40        FB=1
EA9B-85 57      STA $57         dans HRSFB
EA9D-A0 00      LDY #$00        on indexe le premier caractère
EA9F-84 50      STY $50         dans HRS2+1  <-------------------------------
EAA1-C4 4F      CPY $4F         on à fini ?                                   I
EAA3-B0 ED      BCS $EA92       oui, on sort                                  I
EAA5-B1 51      LDA ($51),Y     non, on prend un code                         I
EAA7-20 B5 EA   JSR $EAB5       on l'envoie                                   I
EAAA-A4 50      LDY $50         et on indexe le caractère suivant             I
EAAC-C8         INY                                                           I
EAAD-D0 F0      BNE $EA9F       -------------------------------------------
```

COMMANDE CHAR

Action: affiche à la position du curseur le caractère HRS1 dans la table HRS2 selon HRSFB. La table 0 représente les caractères normaux et 1 les mosaiques. Le principe est facile à suivre et la routine banale.

```
EAAF-A5 4D      LDA $4D         on prend le code dans A
EAB1-0A         ASL             on sort b7
EAB2-46 4F      LSR $4F         on met dans C le type de caractère à afficher
EAB4-6A         ROR             et dans b7 donc A>128 si caractère alterné
EAB5-48         PHA             dans la pile
EAB6-A5 46      LDA $46         est-on au dela de la colonne 234 ?
EAB8-C9 EA      CMP #$EA
EABA-90 17      BCC $EAD3       non ----------------------------------------------
EABC-A6 4A      LDX $4A         oui, X=HRSX6                                      I
EABE-A5 47      LDA $47         A=HRSY                                            I
EAC0-69 07      ADC #$07        on ajoute à HRSY                                  I
EAC2-A8         TAY             dans Y                                            I
EAC3-E9 BF      SBC #$BF        -191 (donc on complémente à 199)                  I
EAC5-90 09      BCC $EAD0       si HRSY est bon                                   I
EAC7-F0 07      BEQ $EAD0       même égal ------------------------------------    I
EAC9-C9 08      CMP #$08        est-on à 8 ?                                  I   I
EACB-D0 02      BNE $EACF       non, ok ---------------------------------     I   I
EACD-A9 00      LDA #$00        oui, alors on met HRSY=0                  I   I   I
EACF-A8         TAY             on remet A dans Y <-----------------------    I   I
EAD0-20 F3 E7   JSR $E7F3       on recalcule la position du point  <----------    I
EAD3-68         PLA             on prend le code <-------------------------------
EAD4-20 31 FF   JSR $FF31       on calcule son adresse dans RESB
EAD7-A0 00      LDY #$00        on met 0 dans RES
EAD9-84 00      STY $00
EADB-A5 49      LDA $49         on sauve HRSX6 et HRSX40 dans la pile
EADD-48         PHA
EADE-A5 4A      LDA $4A
EAE0-48         PHA
EAE1-B1 02      LDA ($02),Y     on lit la ligne courante du caractère
EAE3-0A         ASL             on elimine b7 et b6 qui n'ont pas d'utilité
EAE4-0A         ASL             <-------------------------------------------------
EAE5-F0 0C      BEQ $EAF3       si le code est 0, on passe ----------------------+---
EAE7-48         PHA             on sauve la ligne                                 I   I
EAE8-10 03      BPL $EAED       si le point est allumé                            I   I
EAEA-20 9C E7   JSR $E79C       on l'affiche sans gérer le pattern                I   I
EAED-20 D9 E7   JSR $E7D9       puis on déplace le curseur à droite               I   I
EAF0-68         PLA             on prend le code                                  I   I
EAF1-D0 F1      BNE $EAE4       si ce n'est pas 0, on boucle ---------------------+---
EAF3-20 C1 E7   JSR $E7C1       on descend le curseur d'une ligne <----------------
EAF6-68         PLA             on récupère HRSX6 et HRSX40
EAF7-85 4A      STA $4A
EAF9-68         PLA
EAFA-85 49      STA $49
EAFC-A4 00      LDY $00         on ajoute une ligne de faite
EAFE-C8         INY
EAFF-C0 08      CPY #$08
EB01-D0 D6      BNE $EAD9       et on fait les 8
EB03-A5 46      LDA $46         on ajoute 6 (5+1 car C=1)
EB05-69 05      ADC #$05        à HRSX
EB07-AA         TAX             dans X
EB08-A4 47      LDY $47         HRSY dans Y
EB0A-4C F3 E7   JMP $E7F3       et on positionne le curseur en X,Y
```

ROUTINE PLAY

Action:ouvre les canaux musicaux selon HRS1, canaux de bruits selon HRS2, avec l'enveloppe HRS3, de période HRS4*32 us.

```
EB0D-A5 4F      LDA $4F         on prend la définition d'autorisation des
EB0F-0A         ASL             canaux de bruit
EB10-0A         ASL             dans b5b4b3
EB11-0A         ASL
EB12-05 4D      ORA $4D         et les canaux musicaux dans b2b1b0
EB14-49 3F      EOR #$3F        on inverse le tout
EB16-AA         TAX             dans X
EB17-A9 07      LDA #$07        registre 7 (autorisation)
EB19-20 1A DA   JSR $DA1A       et on ouvre les canaux demandés
EB1C-06 53      ASL $53         on multiplie la periode par 2
EB1E-26 54      ROL $54
EB20-A6 53      LDX $53         on envoie la période poids faible
EB22-A9 0B      LDA #$0B
EB24-20 1A DA   JSR $DA1A
EB27-A6 54      LDX $54
EB29-A9 0C      LDA #$0C        et poids fort
EB2B-20 1A DA   JSR $DA1A
EB2E-A4 51      LDY $51         on prend le numéro de l'enveloppe demandée
EB30-BE 38 EB   LDX $EB38,Y     on lit l'enveloppe PSG correspondante
EB33-A9 0D      LDA #$0D        et on valide l'enveloppe
EB35-4C 1A DA   JMP $DA1A

EB38            BYT 00,11,04,08,10,11,12,13  enveloppes PLAY de 0 à 7
```

PERIODE DES NOTES DE L'OCTAVE 0

Remarque:Les nombres sont des périodes en 16' de us. Donc pour avoir la fréquence de la note correspondante, on fait, si n est le nombre: F=1/((n*16)*10^-6)) c'est à dire F=1000000/16n.
Attention:les notes stockées ici sont à multiplier par 2 pour obtenir les périodes de l'octave 0.

		note	fréquence	octave 3	octave 7 (maxi)
EB40	BYT $0000	blanc	I 00,000 Hertz	I 000,00 Hertz	I 0000,0 Hertz
EB42	BYT $0EEE	DO	I 16,352 "	I 261,64 "	I 4186,2 "
EB44	BYT $0E16	DO#	I 17,332 "	I 277,31 "	I 4437 "
EB46	BYT $0D4C	RE	I 18,360 "	I 293,77 "	I 4700,3 "
EB48	BYT $0C8E	RE#	I 19,446 "	I 311,13 "	I 4978,2 "
EB4A	BYT $0BD8	MI	I 20,613 "	I 329,81 "	I 5277 "
EB4C	BYT $0B2E	FA	I 21,837 "	I 349,40 "	I 5590,5 "
EB4E	BYT $0A8E	FA#	I 23,131 "	I 370,09 "	I 5921,5 "
EB50	BYT $09F6	SOL	I 24,509 "	I 392,15 "	I 6274,5 "
EB52	BYT $0966	SOL#	I 25,976 "	I 415,62 "	I 6650 "
EB54	BYT $08E0	LA	I 27,508 "	I 440,14 "	I 7042,2 "
EB56	BYT $0860	LA#	I 29,151 "	I 466,41 "	I 7462,7 "
EB58	BYT $07E8	SI	I 30,879 "	I 494,07 "	I 7905,1 "

COMMANDE MUSIC

Principe:on lit la note correspondant au troisième paramètre, puis on le divise par 2^octave pour obtenir la période juste. Puis on replace la

période trouvée et on execute SOUND. Les paramètres de MUSIC sont dans HRSx, comme en BASIC.

```
EB5A-A4 4F      LDY $4F         on prend l'octave dans X
EB5C-A5 51      LDA $51         la note dans A
EB5E-0A         ASL             *2 car 2 octets par note
EB5F-AA         TAX             dans X
EB60-BD 40 EB   LDA $EB40,X     on lit la note
EB63-85 4F      STA $4F
EB65-BD 41 EB   LDA $EB41,X     et on multiplie par 2^(octave+1)
EB68-4A         LSR             (+1 car les notes sont stockées pour octave -1)
EB69-66 4F      ROR $4F
EB6B-88         DEY
EB6C-10 FA      BPL $EB68
EB6E-85 50      STA $50         $4F-$50 contient la note
EB70-A6 53      LDX $53         on lit le volume dans X
EB72            BYT $2C         et on passe l'instruction suivante
```

COMMANDE SOUND

Principe:on envoie d'abord le volume, ou 16 si le volume est nul. Ensuite on teste si la période demandée est musicale ou faite de bruit. On ajuste ensuite les périodes musicales (16 bits) ou bruit (8 bits).
Les paramètres sont dans HRSx comme en HYPER-BASIC.

```
EB73-A6 51      LDX $51         on lit le volume dans X
EB75-8A         TXA             on met le volume dans A
EB76-D0 02      BNE $EB7A       volume <>0, on l'envoie ----------------------
EB78-A2 10      LDX #$10        on indique volume géré par l'enveloppe
EB7A-A4 4D      LDY $4D         on prend le canal
EB7C-88         DEY             -1
EB7D-98         TYA             dans A
EB7E-C9 03      CMP #$03        >3 ?
EB80-90 02      BCC $EB84       non
EB82-E9 03      SBC #$03        oui, on enlève 3
EB84-09 08      ORA #$08        et on indique registres de volume (8,9,10)
EB86-20 1A DA   JSR $DA1A       et on écrit le registre de volume
EB89-C0 03      CPY #$03        canal >=3 ?
EB8B-B0 0C      BCS $EB99       oui ------------------------------------------
EB8D-98         TYA             on envoie les fréquence dans le registre du canal
EB8E-0A         ASL             *2 car deux octets
EB8F-A8         TAY
EB90-69 01      ADC #$01        +1
EB92-A6 50      LDX $50         on envoie le poids fort
EB94-20 1A DA   JSR $DA1A
EB97-98         TYA             registre poids faible dans A
EB98            BYT $2C         et on saute l'instruction suivante
EB99-A9 06      LDA #$06        A=6 pour période de bruit blanc
EB9B-A6 4F      LDX $4F         X=poids faible de la période
EB9D-4C 1A DA   JMP $DA1A       on envoie
```

DONNEES POUR SONS PREPROGRAMMES

PING

```
EBA0            BYT $18,0,0,0,0,0,0   soit 2604 Hertz sur canal musical 1
EBA7            BYT $3E,$10,0,0       soit canal 1 musical, volume par enveloppe
EBAC            BYT 0,$0F,0           soit période 61,5 ms et enveloppe 0
```

SHOOT

```
:BAF            BYT 0,0,0,0,0,0       pas de canaux musicaux
:BB4            BYT $0F,7,$10,$10,$10soit 3 canaux en bruit, volume par enveloppe
:BB9            BYT 0,8,0             soit Periode 30 ms et enveloppe 0
```

EXPLODE

```
:BBC            BYT 0,0,0,0,0,0       pas de canaux musicaux
:BC1            BYT $1F,07            periode bruit  identique à SHOOT ! (4 bits)
:BC3            BYT $10,$10,$10       tous canaux en bruits, volume par enveloppe
:BC6            BYT 0,$18,0           période 98 ms et enveloppe 0
```

DONNEES ZAP

```
:BCA            BYT 0,0,0,0,0,0,0 soit rien pour l'instant
:BD1            BYT $3E,$0F       soit canal 1 musical volume maxi
:BD3            BYT 0,0,0,0,0,0   soit pas d'enveloppe
```

ROUTINE PING

```
:BD9-A2 A0      LDX #$A0        on indexe PING
:BDB-A0 EB      LDY #$EB
:BDD-D0 0A      BNE $EBE9       et on envoie
```

ROUTINE SHOOT

```
:BDF-A2 AE      LDX #$AE        on indexe SHOOT
:BE1-A0 EB      LDY #$EB
:BE3-D0 04      BNE $EBE9       et on envoie
```

ROUTINE EXPLODE

```
:BE5-A2 BC      LDX #$BC        on indexe EXPLODE
:BE7-A0 EB      LDY #$EB
:BE9-4C E7 D9   JMP $D9E7       et on envoie
```

ROUTINE ZAP

Principe:On augmente la période du canal 1 de 0 à 1800 us de 16 en 16 us par intervalle d'une milliseconde.

```
:BEC-A2 CA      LDX #$CA        on envoie les données de base
:BEE-A0 EB      LDY #$EB
:BF0-20 E7 D9   JSR $D9E7       au PSG
:BF3-A9 00      LDA #$00
:BF5-AA         TAX             0 dans X
:BF6-8A         TXA
:BF7-48         PHA             dans la pile
:BF8-A9 00      LDA #$00        on envoie X dans le canal 0
:BFA-20 1A DA   JSR $DA1A
:BFD-A2 00      LDX #$00        on attend près d'1 ms
:BFF-CA         DEX
:C00-D0 FD      BNE $EBFF
:C02-68         PLA
:C03-AA         TAX
:C04-E8         INX
:C05-E0 70      CPX #$70        et cela 112 fois, donc période croissante
:C07-D0 ED      BNE $EBF6
:C09-A9 08      LDA #$08
```

```
EC0B-A2 00      LDX #$00
EC0D-4C 1A DA   JMP $DA1A       et on coupe le son
```

LIT UN CODE DU BUFFER SERIE ENTREE

```
EC10-A2 0C      LDX #$0C        on indexe buffer série entrée
EC12-4C 5D DB   JMP $DB5D       on lit un code venant du minitel
```

ATTEND UN CODE DU BUFFER SERIE ENTREE

```
EC15-20 10 EC   JSR $EC10       on lit le code    <--
EC18-B0 FB      BCS $EC15       C=1 , lecture ratée -
EC1A-60         RTS
```

ECRIT UN CARACTERE DANS LE BUFFER SORTIE SERIE

```
EC1B-2C 1B EC   BIT $EC1B       N=0, écriture
EC1E-4C 79 DB   JMP $DB79       et on gère le buffer série sortie
```

ENVOIE A EN SORTIE MINITEL

Action:Si b6 de $5B est à 1, envoie A dans le buffer série sans cérémonies. Sinon, si A est un code de controle (0-31 ou 128-159), envoie d'abord un 02. Si A n'est pas un code de controle, mais a son bit 7 à 1, envoie d'abord un 01. En fait b6 de $5B est à 0 si on dialogue via le MINITEL, et 1 si on dialogue simplement via la RS232.

```
EC21-24 5B      BIT $5B         b6 de $5B ?
EC23-70 24      BVS $EC49       oui, on passe ----------------------------------------
EC25-AA         TAX             donnée dans X                                          I
EC26-30 10      BMI $EC38       >128, on saute ---------------------------------       I
EC28-C9 20      CMP #$20        code de controle ?                             I       I
EC2A-B0 1D      BCS $EC49       non ------------------------------------------+---    C
EC2C-69 20      ADC #$20        oui, on force b5 à 1                           I       I
EC2E-48         PHA             dans la pile  <--------------------------------I       I
EC2F-A9 02      LDA #$02        on envoie un 02                                I   I   I
EC31-20 49 EC   JSR $EC49                                                      I   I   I
EC34-68         PLA                                                            I   I   I
EC35-4C 49 EC   JMP $EC49       puis le code de controle                       I   I   I
EC38-C9 A0      CMP #$A0        <128+32 ? <------------------------------------+---    I
EC3A-B0 04      BCS $EC40       non --------------------------------------------+--     I
EC3C-69 C0      ADC #$C0        oui, on ramène à 0-31                          I   I   I
EC3E-B0 EE      BCS $EC2E       inconditionnel ---------------------------------   I   I
EC40-29 7F      AND #$7F        on enlève b7 <-----------------------------------------
EC42-48         PHA             dans la pile
EC43-A9 01      LDA #$01        en envoie un 01
EC45-20 49 EC   JSR $EC49
EC48-68         PLA
EC49-2C 49 EC   BIT $EC49       N=0, V=0 <-------------------------------------------
EC4C-4C 12 DB   JMP $DB12       on envoie le code dans A
```

ENVOIE A EN SORTIE SERIE AVEC CHECK

```
EC4F-86 0C      STX $0C         on sauve X et Y
EC51-84 0D      STY $0D
EC53-48         PHA             et A
EC54-24 5B      BIT $5B         b7 de $5B à 1 ?
EC56-10 06      BPL $EC5E       non ----------------------------------------------
EC58-20 21 EC   JSR $EC21       on envoie A au MINITEL                             I
```

```
EC5B-4C 61 EC   JMP $EC61    ---et on passe                                    I
EC5E-20 1B EC   JSR $EC1B    I  on envoie simplement A au buffer <-------------
EC61-68         PLA          -->on reprend le code
EC62-45 0E      EOR $0E         on inverse selon $0E
EC64-85 0E      STA $0E         et on place dans $0E pour controle (CHECK)
EC66-A6 0C      LDX $0C         on récupère X et Y
EC68-A4 0D      LDY $0D
EC6A-60         RTS
```

LIT UN CODE SUR LA RS232

Action: lit un code sur la RS232 en gérant le CTRL-C. Si le code est un code de controle, renvoie le caractère suivant avec son bit 7 à 1.
Dans le cas ou b7 ou b6 de $5B est à 1, on renvoie le code simplement.
Si b7=0, on lit le code par la $EC10
Si b7=1, on lit le code par la $C518
Ce qui revient au même, allez comprendre...

```
EC6B-86 0C      STX $0C         on sauve X et Y
EC6D-84 0D      STY $0D
EC6F-0E 7E 02   ASL $027E    -->CTRL-C pressé ?
EC72-90 03      BCC $EC77    I  non, on saute
EC74-68         PLA          I  oui, on dépile le retour à l'appelant
EC75-68         PLA          I
EC76-60         RTS          I  et on retourne à l'appelant de l'appelant
EC77-24 5B      BIT $5B      I  b7 de $5B à 1 ?
EC79-30 10      BMI $EC8B    I  oui ---------------------------------------------
EC7B-20 10 EC   JSR $EC10    I  non, on lit un code venant de la RS232          I
EC7E-B0 EF      BCS $EC6F    0--si pas de code lu, on saute au CTRL-C           I
EC80-48         PHA          I  on sauve le code  <-------------------------    I
EC81-45 0E      EOR $0E      I  on ajuste le CHECK                         I    I
EC83-85 0E      STA $0E      I                                             I    I
EC85-68         PLA          I  on prend le code                           I    I
EC86-A6 0C      LDX $0C      I  et on restaure X et Y                      I    I
EC88-A4 0D      LDY $0D      I                                             I    I
EC8A-60         RTS          I                                             I    I
EC8B-20 B4 EC   JSR $ECB4    I  on lit un code sur la RS232 <--------------+----
EC8E-B0 DF      BCS $EC6F    ---pas de code, on boucle sur CTRL-C          I
EC90-24 5B      BIT $5B         b6 de $5B à 1 ?                            I
EC92-70 EC      BVS $EC80       oui, on sort ------------------------------0
EC94-C9 20      CMP #$20        code de controle ?                         I
EC96-B0 E8      BCS $EC80       non, on sort ------------------------------0
EC98-48         PHA             on sauve le code de controle               I
EC99-20 B9 EC   JSR $ECB9       on attend un code RS232                    I
EC9C-AA         TAX             dans X                                     I
EC9D-68         PLA                                                        I
EC9E-A8         TAY             on prend le code de controle dans Y        I
EC9F-8A         TXA             et son suivant dans A                      I
ECA0-C0 01      CPY #$01        le controle est 01 ?                       I
ECA2-D0 04      BNE $ECA8    ---non                                        I
ECA4-09 80      ORA #$80     I  oui, on force b7 à 1                       I
ECA6-30 D8      BMI $EC80    I  et on saute ----------------------------------0
ECA8-C9 40      CMP #$40     -->code suivant <64 ?                         I
ECAA-B0 04      BCS $ECB0    ---non                                        I
ECAC-E9 1F      SBC #$1F     I  oui, on retire 31                          I
ECAE-B0 D0      BCS $EC80    I  inconditionnel -------------------------------0
ECB0-69 3F      ADC #$3F     -->on ajoute 63                               I
ECB2-90 CC      BCC $EC80       inconditionnel ----------------------------
```

LIT UN CODE EN ENTREE SERIE

```
ECB4-A2 0C     LDX #$0C        indexe buffer série entrée
ECB6-4C 18 C5  JMP $C518       lit le code dans le buffer
```

ATTEND UN CODE DANS LE BUFFER ENTREE SERIE

```
ECB9-20 B4 EC  JSR $ECB4       on lit le code
ECBC-B0 FB     BCS $ECB9       jusqu'à ce que la lecture soit faite
ECBE-60        RTS

ECBF-38        SEC             C=1
ECC0           BYT $24         saute le CLC
ECC1-18        CLC             C=0
ECC2-A9 80     LDA #$80        A=128, N=1
ECC4-4C 5D DB  JMP $DB5D       ferme ou ouvre l'entrée RS232

ECC7-38        SEC             C=1
ECC8           BYT $24         saute le CLC
ECC9-18        CLC             C=0
ECCA-A9 80     LDA #$80        N=1
ECCC-4C 79 DB  JMP $DB79       ferme (C=1) ou ouvre (C=0) la sortie RS232

ECCF-38        SEC             C=1
ECD0           BYT $24         saute le CLC
ECD1-18        CLC             C=0
ECD2-A9 80     LDA #$80        N=1
ECD4-4C F7 DA  JMP $DAF7       ferme ou ouvre l'entrée MINITEL

ECD7-38        SEC             C=1
ECD8           BYT $24         saute le CLC
ECD9-18        CLC             C=0
ECDA-A9 80     LDA #$80        N=1
ECDC-4C 12 DB  JMP $DB12       ferme ou ouvre la sortie MINITEL
```

CALCULE LA TAILLE D'UN FICHIER

```
ECDF-38        SEC
ECE0-AD 2F 05  LDA $052F       on calcule dans LOSALO
ECE3-ED 2D 05  SBC $052D       la différence entre les adresses
ECE6-8D 2A 05  STA $052A       de début et de fin du fichier :
ECE9-AD 30 05  LDA $0530       FISALO-DESALO
ECEC-ED 2E 05  SBC $052E
ECEF-8D 2B 05  STA $052B
ECF2-AD 2D 05  LDA $052D
ECF5-AC 2E 05  LDY $052E
ECF8-85 00     STA $00         on met dans RES l'adresse de début du fichier
ECFA-84 01     STY $01
ECFC-60        RTS
```

ENVOIE L'ENTETE SERIE D'UN FICHIER

Principe: l'entête d'un fichier série est comme suit :
- 50 fois SYN pour synchro
- 1 fois 24 pour fin de synchro
- 12 caractères du nom
- 0 fin du nom
- 7 octets pour type, début, fin et execution du programme

1 octet de controle pour ce qui precede.

```
ECFD-A2 32      LDX #$32        on envoie 50 fois l'octet de synchro
ECFF-A9 16      LDA #$16        16 (SYN)
ED01-20 4F EC   JSR $EC4F
ED04-CA         DEX
ED05-D0 F8      BNE $ECFF
ED07-A9 24      LDA #$24        puis un octet 24
ED09-20 4F EC   JSR $EC4F
ED0C-A9 00      LDA #$00
ED0E-85 0E      STA $0E
ED10-A2 00      LDX #$00
ED12-BD 18 05   LDA $0518,X     puis le nom du fichier
ED15-20 4F EC   JSR $EC4F
ED18-E8         INX
ED19-E0 0C      CPX #$0C        (12 caractères)
ED1B-D0 F5      BNE $ED12
ED1D-A9 00      LDA #$00        puis un 0 pour finir le nom
ED1F-20 4F EC   JSR $EC4F
ED22-A2 00      LDX #$00        puis type et adresses du fichier
ED24-BD 2C 05   LDA $052C,X
ED27-20 4F EC   JSR $EC4F
ED2A-E8         INX
ED2B-E0 07      CPX #$07        1+2+2+2 font 7
ED2D-D0 F5      BNE $ED24
ED2F-A5 0E      LDA $0E         puis l'octet de controle
ED31-4C 4F EC   JMP $EC4F
```

LIT L'ENTETE D'UN FICHIER

Principe:Dès qu'on est synchronisé avec l'entête et qu'on est sur qu'il s'agit bien d'un entête TELESTRAT, on lit l'entête comme on l'a sauvé puis on affiche le nom. Le caractère bizarre qui suit le nom n'est autre que le code de controle (CHECK)>

```
ED34-20 6B EC   JSR $EC6B       on lit un code RS232 <-------------------------
ED37-C9 16      CMP #$16        est-ce SYN ?                                   I
ED39-D0 F9      BNE $ED34       non, on boucle ---------------------------------0
ED3B-A2 0A      LDX #$0A        on va lire 10 SYN                              I
ED3D-20 6B EC   JSR $EC6B       on lit                                         I
ED40-C9 16      CMP #$16        SYN ?                                          I
ED42-D0 F0      BNE $ED34       non, on reprend la synchro au début ------------0
ED44-CA         DEX                                                            I
ED45-D0 F6      BNE $ED3D       oui, on en lit 10                              I
ED47-20 6B EC   JSR $EC6B       on lit un code <-------------------------------I
ED4A-C9 16      CMP #$16        fin de synchro ?                             I I
ED4C-F0 F9      BEQ $ED47       non ------------------------------------------I
ED4E-C9 24      CMP #$24        oui est-ce 24 ?                                I
ED50-D0 E2      BNE $ED34       non, la synchro est ratée, on repart à 0 -------
ED52-A9 00      LDA #$00        on met le CHECK
ED54-85 0E      STA $0E         à 0
ED56-20 6B EC   JSR $EC6B    --->on lit un code
ED59-AA         TAX          I
ED5A-F0 06      BEQ $ED62    I  0, on saute ------------------------------------
ED5C-20 B5 DB   JSR $DBB5    I  on affiche le caractère du nom                 I
ED5F-4C 56 ED   JMP $ED56    ---et on affiche tout le nom                      I
ED62-A2 00      LDX #$00        <-----------------------------------------------
```

```
ED64-20 6B EC   JSR $EC6B       on lit la définition du fichier
ED67-9D 2C 05   STA $052C,X
ED6A-E8         INX
ED6B-E0 07      CPX #$07        (7 octets)
ED6D-D0 F5      BNE $ED64
ED6F-20 6B EC   JSR $EC6B       on lit l'octet de CHECK
ED72-09 30      ORA #$30        on force b5b4 à 11
ED74-4C B5 DB   JMP $DBB5       et on l'affiche
```

COMMANDE CONSOLE

Action: transforme le TELESTRAT en terminal série. Les codes tapés au clavier sont transmis à la RS232 et les codes reçus via la RS232 sont affichés à l'écran. On sort par CTRL-C.

```
ED77-20 C1 EC   JSR $ECC1       on ouvre le buffer RS232 entrée
ED7A-20 C9 EC   JSR $ECC9       on ouvre le buffer RS232 sortie
ED7D-20 10 EC   JSR $EC10       on lit un code en entrée <-------------------------
ED80-B0 03      BCS $ED85       si pas de code on saute ------------------------- I
ED82-20 B5 DB   JSR $DBB5       on affiche le code à l'écran                   I  I
ED85-20 CF C7   JSR $C7CF       on lit un code sur le canal 0 <----------------- I
ED88-B0 F3      BCS $ED7D       pas de code, on saute ---------------------------0
ED8A-C9 03      CMP #$03        CTRL-C ?                                        I
ED8C-F0 06      BEQ $ED94    ---oui                                             I
ED8E-20 1B EC   JSR $EC1B    I  non, on écrit le code en sortie RS232           I
ED91-4C 7D ED   JMP $ED7D    I  et on boucle ----------------------------------
ED94-20 BF EC   JSR $ECBF    -->on ferme l'entrée RS232
ED97-4C C7 EC   JMP $ECC7       et la sortie RS232
```

COMMANDE SDUMP

Principe: Envoie à l'écran tous les codes lus sur l'entrée série. Les codes de controle sont affichés en hexa à l'encre rouge, les autres sous leur forme ASCII. CTRL-C arrête tout cela.

```
ED9A-20 C1 EC   JSR $ECC1       on ouvre le buffer RS232 entrée
ED9D-0E 7E 02   ASL $027E    -->on teste si CTRL-C
EDA0-B0 25      BCS $EDC7    I  oui, on sort ----------------------------------
EDA2-20 10 EC   JSR $EC10    I  on lit un code                                 I
EDA5-B0 F6      BCS $ED9D    0--pas de code                                    I
EDA7-AA         TAX          I                                                 I
EDA8-30 04      BMI $EDAE    I  b7=1, on saute ------------------------------  I
EDAA-C9 20      CMP #$20     I  code de controle ?                          I  I
EDAC-B0 13      BCS $EDC1    I  non ---------------------------------------+--I
EDAE-48         PHA          I  on sauve le code <------------------------- I  I
EDAF-A9 81      LDA #$81     I  on envoie encre rouge                       I  I
EDB1-20 B5 DB   JSR $DBB5    I                                              I  I
EDB4-68         PLA          I                                              I  I
EDB5-20 54 CE   JSR $CE54    I  on convertit le code en hexa                I  I
EDB8-20 B5 DB   JSR $DBB5    I  et on affiche                               I  I
EDBB-98         TYA          I                                              I  I
EDBC-20 B5 DB   JSR $DBB5    I                                              I  I
EDBF-A9 87      LDA #$87     I  encre blanche                               I  I
EDC1-20 B5 DB   JSR $DBB5    I  <-------------------------------------------  I
EDC4-4C 9D ED   JMP $ED9D    ---et on boucle                                   I
EDC7-4C BF EC   JMP $ECBF       on ferme le buffer RS232 entrée <--------------
```

COMMANDE SSAVE

Remarque:C=1 en entrée pour SSAVEA, 0 sinon.

```
EDCA-66 5B      ROR $5B         on met 0 dans b7 de $5B
EDCC-46 5B      LSR $5B         sans toucher à l'indicateur d'entête
EDCE-20 C9 EC   JSR $ECC9       on ouvre le buffer RS232 sortie
EDD1-20 0A EE   JSR $EE0A       on envoie le programme
EDD4-4C C7 EC   JMP $ECC7       et on ferme le buffer RS232 sortie
```

COMMANDE MSAVE

Remarque:C=1 en entrée pour MSAVEA, 0 sinon.

```
EDD7-66 5B      ROR $5B         on met 1 dans b7 de $5B
EDD9-38         SEC             sans toucher à l'indicateur d'entête
EDDA-66 5B      ROR $5B
EDDC-20 D9 EC   JSR $ECD9
EDDF-20 0A EE   JSR $EE0A
EDE2-4C D7 EC   JMP $ECD7
```

COMMANDE SLOAD

Remarque:C=1 en entrée pour SLOADA, 0 sinon.

```
EDE5-66 5B      ROR $5B         b7 de $5B à 0 pour envoi RS232
EDE7-46 5B      LSR $5B
EDE9-A9 40      LDA #$40        bug corrigé par rapport au TELEMON V2.3
EDEB-8D 0E 03   STA $030E       on interdit les IRQ par T1
EDEE-20 C1 EC   JSR $ECC1       on ouvre la sortie RS232
EDF1-20 56 EE   JSR $EE56       on envoie le programme
EDF4-A9 C0      LDA #$C0        et on rétablit les IRQ
EDF6-8D 0E 03   STA $030E
EDF9-4C BF EC   JMP $ECBF       on ferme la sortie RS232
```

COMMANDE MLOAD

```
EDFC-66 5B      ROR $5B         b7 de $5B à 1 pour envoi minitel
EDFE-38         SEC
EDFF-66 5B      ROR $5B
EE01-20 D1 EC   JSR $ECD1       on ouvre la sortie minitel
EE04-20 56 EE   JSR $EE56       on envoie le programme
EE07-4C CF EC   JMP $ECCF       et on ferme la sortie minitel
```

SAUVE UN FICHIER RS232/MINITEL

Principe:En entrée, $5B à son bit 7 selon le récepteur (0 pour RS232 et 1 pour minitel) et son bit 6 à 1 si on envoie sans entête.
La différence RS232/MINITEL est faite telle qu'il se casse 1/2 seconde (normalement :) tous les 256 octets. En fait ce temps est souvent rallongé à cause des interruptions croisées qui ralentissent la cadence de décompte des timers...

```
EE0A-24 5B      BIT $5B         entête à envoyer ?
EE0C-70 03      BVS $EE11       non ----------------------------------
EE0E-20 FD EC   JSR $ECFD       on envoie l'entête                     I
EE11-20 DF EC   JSR $ECDF       on calcule la taille du programme <----------------
EE14-A9 00      LDA #$00        on met 0
EE16-85 0E      STA $0E         dans le CHECK
EE18-AD 2A 05   LDA $052A    -->on envoie la partie hors-page (comme décalages)
```

```
:E1B-F0 12      BEQ $EE2F    I  si fini, on saute ----------------------------.
:E1D-A0 00      LDY #$00     I                                                |
:E1F-B1 00      LDA ($00),Y  I  on lit l'octet                                |
:E21-20 4F EC   JSR $EC4F    I  on l'envoie en sortie série                   |
:E24-CE 2A 05   DEC $052A    I  on décompte les octets                        |
:E27-E6 00      INC $00      I  on ajuste l'adresse de lecture                |
:E29-D0 ED      BNE $EE18    I                                                |
:E2B-E6 01      INC $01      I                                                |
:E2D-D0 E9      BNE $EE18    ---inconditionnel                                |
:E2F-AD 2B 05   LDA $052B       on envoie page par page <---------------------0
:E32-F0 1D      BEQ $EE51                                                     |
:E34-A0 00      LDY #$00                                                      |
:E36-B1 00      LDA ($00),Y     on lit le code  <-----                        |
:E38-20 4F EC   JSR $EC4F       on l'envoie           I                       |
:E3B-C8         INY             256 fois              I                       |
:E3C-D0 F8      BNE $EE36       ---------------------                         |
:E3E-CE 2B 05   DEC $052B       on indique une page de moins                  |
:E41-E6 01      INC $01         on ajoute une page à l'adresse de lecture     |
:E43-24 5B      BIT $5B         mode minitel ?                                |
:E45-10 E8      BPL $EE2F       non ------------------------------------------0
:E47-A9 30      LDA #$30        oui, on indique 48 dixièmes                   |
:E49-85 44      STA $44         dans TIMEUD                                   |
:E4B-A5 44      LDA $44                                                       |
:E4D-D0 FC      BNE $EE4B       et on attend le retour à 0 (donc 0.5 secondes)|
:E4F-F0 DE      BEQ $EE2F       et on boucle ----------------------------------0
:E51-A5 0E      LDA $0E         puis on envoie le code de controle
:E53-4C 4F EC   JMP $EC4F
```

LIT UN FICHIER RS232/MINITEL

Principe:Selon $5B (voir plus haut) on lit ou non l'entête et on charge le fichier. On ne vérifie pas que l'entête est bien chargé... dommage ! L'octet de CHECK du programme est ensuite affiché. Il suffit de le comparer avec $0E pour savoir si le programme s'est bien chargé.

```
:E56-24 5B      BIT $5B         entête ?
:E58-70 03      BVS $EE5D       non ---------------------------------------.
:E5A-20 34 ED   JSR $ED34       oui, on lit l'entête                       |
:E5D-20 DF EC   JSR $ECDF       on calcule la taille du fichier <----------|
:E60-24 5B      BIT $5B         entête ?                                   |
:E62-50 08      BVC $EE6C       oui ----------------------------------------|
:E64-A9 FF      LDA #$FF        non, on indique 64 Ko                      |
:E66-8D 2A 05   STA $052A       à lire (sans valeur puisqu'on arrête de charger
:E69-8D 2B 05   STA $052B       par CTRL-C)                                |
:E6C-A0 00      LDY #$00        0 dans CHECK <-----------------------------|
:E6E-84 0E      STY $0E         (somme de controle)                        |
:E70-AD 2A 05   LDA $052A    -->on lit les octets hors-page                |
:E73-F0 11      BEQ $EE86    I  fini ------------------------------------------.
:E75-20 6B EC   JSR $EC6B    I  on lit un code venant de la RS232               |
:E78-91 00      STA ($00),Y  I  en mémoire                                      |
:E7A-CE 2A 05   DEC $052A    I  on indique un code lu                           |
:E7D-E6 00      INC $00      I  on ajuste l'adresse mémoire                     |
:E7F-D0 EF      BNE $EE70    I                                                  |
:E81-E6 01      INC $01      I                                                  |
:E83-4C 70 EE   JMP $EE70    ---et on boucle                                    |
:E86-AD 2B 05   LDA $052B    -->puis les pages <--------------------------------|
:E89-F0 12      BEQ $EE9D    I  fini ------------------------------------------'
:E8B-A0 00      LDY #$00     I
```

```
EE8D-20 6B EC   JSR $EC6B    I  on lit un code                                  I
EE90-91 00      STA ($00),Y  I  en mémoire                                      I
EE92-C8         INY          I  et ainsi pour 256 octets                        I
EE93-D0 F8      BNE $EE8D    I                                                  I
EE95-E6 01      INC $01      I  on ajoute une page en mémoire                   I
EE97-CE 2B 05   DEC $052B    I  un page lue de plus                             I
EE9A-4C 86 EE   JMP $EE86    ---et on boucle                                    I
EE9D-20 6B EC   JSR $EC6B       on lit le CHECK <-------------------------------
EEA0-09 30      ORA #$30        force b5b4 à 11
EEA2-4C B5 DB   JMP $DBB5       et on l'affiche
```

COMMANDE RING

Action:Attend une sonnerie du téléphone. C=1 en sortie si la sonnerie était en route lors de l'appel à la routine, 0 sinon.

```
EEA5-A9 00      LDA #$00        on met 0 dans dans valeur joystick
EEA7-8D 8C 02   STA $028C       car le joystick parasite la ligne (!)
EEAA-A9 10      LDA #$10        on prend %00010000 pour test V2IFR
EEAC-2C 2D 03   BIT $032D       transition CB1 ?
EEAF-D0 32      BNE $EEE3       non, on passe
EEB1-38         SEC             oui C=1
EEB2-60         RTS             on sort
```

DETECTE SONNERIE TELEPHONIQUE

Action:attend pendant 65 ms une transition par CB1 du VIA2. Si le timer arrive à 0 sans transition, A=0 et C=1. Si une transition est detectée, alors A=1 et C=1 si la transition à eu lieu dans les 28 ms, A=1 et C=1 si la A=1 et C=0 sinon.

```
EEB3-A9 FF      LDA #$FF        on met 65536
EEB5-8D 28 03   STA $0328       dans V2T2
EEB8-8D 29 03   STA $0329
EEBB-AD 29 03   LDA $0329       on lit V2T2H  <-------------------------------
EEBE-C9 C5      CMP #$C5        =$C5 ? (donc 15 ms passées ?)                  I
EEC0-B0 F9      BCS $EEBB       au dessus ------------------------------------
EEC2-2C 20 03   BIT $0320       ?
EEC5-A9 20      LDA #$20     -->on isole l'état VIA2
EEC7-2D 2D 03   AND $032D    I  de T2
EECA-D0 13      BNE $EEDF    I  T2=0 on passe ------------------------------------
EECC-A9 10      LDA #$10     I  on isole CB1                                    I
EECE-2D 2D 03   AND $032D    I  CB1 à 1 ?                                       I
EED1-F0 F2      BEQ $EEC5    ---non                                             I
EED3-AD 29 03   LDA $0329       oui, on lit V2T2H                               I
EED6-C9 AD      CMP #$AD        =$AD (donc 13 ms passées ?)                     I
EED8-90 07      BCC $EEE1       oui, A>0, C=1 et on sort                        I
EEDA-C9 B5      CMP #$B5        non, 7 ms passées ? (oui:C=0 sinon C=1)         I
EEDC-A9 01      LDA #$01        A=1                                             I
EEDE-60         RTS                                                             I
EEDF-A9 00      LDA #$00        A=0  <-----------------------------------------
EEE1-38         SEC             C=1
EEE2-60         RTS             on sort
```

DETECTION DE SONNERIE RING (suite)

```
EEE3-78         SEI             pas d'interruptions
EEE4-A2 04      LDX #$04        on teste 4 fois la sonnerie
EEE6-20 B3 EE   JSR $EEB3
EEE9-CA         DEX
```

```
EEEA-D0 FA      BNE $EEE6
EEEC-20 B3 EE   JSR $EEB3       on teste la sonnerie
EEEF-F0 0A      BEQ $EEFB       pas de sonnerie ----------------------------------
EEF1-B0 F9      BCS $EEEC       C=1, transition rapide, X=0                       I
EEF3-E8         INX                                                               I
EEF4-4C EC EE   JMP $EEEC                                                         I
EEF7-58         CLI           -->on rétablit les IRQ                              I
EEF8-4C AA EE   JMP $EEAA     I  et on boucle sur le test CB2                     I
EEFB-E0 06      CPX #$06      I  X=6 ? <-----------------------------------------
EEFD-90 F8      BCC $EEF7     ---en dessous, on boucle
EEFF-20 B3 EE   JSR $EEB3       on lit CB1    <-----------------
EF02-B0 FB      BCS $EEFF       on boucle sur la detection -----
EF04-A0 1E      LDY #$1E        Y=30
EF06-A2 00      LDX #$00
EF08-20 B3 EE   JSR $EEB3       on détecte la sonnerie  <------------------
EF0B-90 01      BCC $EF0E       pas detectée                                      I
EF0D-E8         INX             detectée, on compte une fois de plus              I
EF0E-88         DEY                                                               I
EF0F-D0 F7      BNE $EF08       on boucle ---------------------------------------
EF11-E0 0F      CPX #$0F        on à détecté 15 fois ?
EF13-B0 E2      BCS $EEF7       plus, on repart
EF15-58         CLI             oui, on rétablit les IRQ (encore ?)
EF16-A9 0A      LDA #$0A        on attend 0,5 secondes
EF18-85 44      STA $44
EF1A-A5 44      LDA $44
EF1C-D0 FC      BNE $EF1A
EF1E-18         CLC             C=0
EF1F-60         RTS             et on sort
```

PRISE DE LIGNE

Action:retourne le modem du Minitel et prend la ligne...

```
EF20-20 D9 EC   JSR $ECD9       ouvre sortie MINITEL
EF23-A9 6F      LDA #$6F        envoie ESC/39/6F
EF25-20 30 EF   JSR $EF30       soit PRO1 OPPO (retournement)
EF28-A9 68      LDA #$68        ESC/39/68
EF2A-20 30 EF   JSR $EF30       soit PRO1 CONNEXION (prise de ligne)


EF2D-4C D7 EC   JMP $ECD7       et ferme sortie  MINITEL
```

ENVOIE UNE SEQUENCE PRO1 AU MINITEL

```
EF30-48         PHA             sauve le code xx de PRO1 xx
EF31-A9 1B      LDA #$1B        envoie ESC
EF33-20 49 EC   JSR $EC49
EF36-A9 39      LDA #$39        et 39 au minitel
EF38-20 49 EC   JSR $EC49       (ESC/39 = PRO1)
EF3B-68         PLA
EF3C-4C 49 EC   JMP $EC49       et la donnée
```

LIBERE LA LIGNE

```
EF3F-20 D9 EC   JSR $ECD9       ouvre sortie minitel
EF42-A9 67      LDA #$67        envoie PRO1 DECONNEXION
```

```
EF44-20 30 EF   JSR $EF30
EF47-4C D7 EC   JMP $ECD7       et ferme la sortie minitel
```

COMMANDE WCXFIN

Action:attend que le correspondant frappe CONNEXION/FIN. S'il ne l'a pas fait dans les 25 secondes, retourne C=1. Sinon, C=0.

```
EF4A-20 D1 EC   JSR $ECD1       ouvre l'entrée minitel
EF4D-A9 FA      LDA #$FA        attend 1 seconde
EF4F-85 44      STA $44
EF51-A5 44      LDA $44
EF53-C9 F0      CMP #$F0
EF55-D0 FA      BNE $EF51
EF57-A2 0C      LDX #$0C        X=12 (entrée série)
EF59-20 0C C5   JSR $C50C       vide le buffer minitel
EF5C-A5 44      LDA $44      -->on avait 25 secondes
EF5E-D0 05      BNE $EF65    I  le temps n'est pas écoulé ----------------------
EF60-20 CF EC   JSR $ECCF    I  25 secondes écoulées                           I
EF63-38         SEC          I  C=1, entrée minitel fermée                     I
EF64-60         RTS          I                                                 I
EF65-20 B4 EC   JSR $ECB4    I  on lit un code minitel <-----------------------
EF68-B0 F2      BCS $EF5C    O--pas de code
EF6A-C9 13      CMP #$13     I  est-ce SEP ?
EF6C-D0 EE      BNE $EF5C    O--non
EF6E-20 B9 EC   JSR $ECB9    I  oui, on lit le deuxième
EF71-C9 53      CMP #$53     I  est 53 ? (SEP/53=connexion/fin)
EF73-D0 E7      BNE $EF5C    ---non
EF75-20 CF EC   JSR $ECCF       oui, on ferme l'entrée minitel
EF78-18         CLC             et C=0
EF79-60         RTS
```

COMMANDE MOUT

```
EF7A-48         PHA             sauve la donnée
EF7B-20 D9 EC   JSR $ECD9       ouvre le minitel en sortie
EF7E-68         PLA
EF7F-20 49 EC   JSR $EC49       envoie le code
EF82-4C D7 EC   JMP $ECD7       ferme le minitel en sortie
```

COMMANDE SOUT

```
EF85-48         PHA             sauve la donnée
EF86-20 C9 EC   JSR $ECC9       ouvre la RS232 en sortie
EF89-68         PLA
EF8A-20 1B EC   JSR $EC1B       envoie la donnée
EF8D-4C C7 EC   JMP $ECC7       ferme la RS232 en sortie
```

AJOUTE 0.5 A ACC1

```
EF90-A9 E4      LDA #$E4        indexe 1/2
EF92-A0 F5      LDY #$F5
EF94-4C AF EF   JMP $EFAF       fait (AY)+ACC1
EF97-60         RTS             ben voyons ! le EF79 était trop loin ?
```

(AY)-ACC1

```
EF98-20 EC F1   JSR $F1EC       on met (AY) dans ACC2
```

ACC2-ACC1

```
EF9B-A5 65      LDA $65         on complément le signe de ACC1
EF9D-49 FF      EOR #$FF
EF9F-85 65      STA $65
EFA1-45 6D      EOR $6D         on plaque ACC2S
EFA3-85 6E      STA $6E         et on stocke le produit des signes (ACCPS)
EFA5-A5 60      LDA $60         on positionne A et P selon ACC1E
EFA7-4C B2 EF   JMP $EFB2       et on additionne
```

JUSTIFIER LA MANTISSE SELON A

```
EFAA-20 E5 F0   JSR $F0E5       on justifie
EFAD-90 3F      BCC $EFEE       inconditionnel
```

(AY)+ACC1

```
EFAF-20 EC F1   JSR $F1EC       (AY) -> ACC2
```

ACC2+ACC1

Principe: Le principe de l'addition en virgule flottante est aisé à assimiler, mais lourd à programmer. En fait, le problème est d'additionner des nombres ayant même exposant. Ainsi, on va ramener le plus petit au même exposant que le plus grand par une justification (X décalages de la mantisse, X étant la valeur absolue de la différence des exposants). Ensuite, on ajoute les mantisses et on ajuste l'exposant qui par défaut est fixé au plus grand des exposants de deux opérandes.

```
EFB2-D0 03      BNE $EFB7       ACC1<>0,on passe
EFB4-4C 77 F3   JMP $F377       ACC1=0, on sort avec ACC2 dans ACC1
EFB7-BA         TSX             on sauve SP pour retour en cas d'erreur
EFB8-86 89      STX $89         dans FLSVS
EFBA-A6 66      LDX $66         on sauve le bit d'extension
EFBC-86 7F      STX $7F         dans FLTRL
EFBE-A2 68      LDX #$68        on indexe ACC2
EFC0-A5 68      LDA $68         et on prend ACC2E (exposant)
EFC2-A8         TAY             dans Y
EFC3-F0 D2      BEQ $EF97       si ACC2=0, on sort simplement (pourquoi pas EF79?)
EFC5-38         SEC
EFC6-E5 60      SBC $60         on calcule la différence des exposants
EFC8-F0 24      BEQ $EFEE       exposants égaux ----------------------------------
EFCA-90 12      BCC $EFDE       ACC2E<ACC1E ---------------------------------     I
EFCC-84 60      STY $60         l'exposant du résultat est ACC2E              I     I
EFCE-A4 6D      LDY $6D         le signe                                      I     I
EFD0-84 65      STY $65         est ACC2S                                     I     I
EFD2-49 FF      EOR #$FF        on complémente le signe                       I     I
EFD4-69 00      ADC #$00        +1 (C=1), donc S=-S                           I     I
EFD6-A0 00      LDY #$00        0 dans l'extension                            I     I
EFD8-84 7F      STY $7F         car ACC2 n'a pas d'extension                  I     I
EFDA-A2 60      LDX #$60        on indexe ACC1 pour justification             I     I
EFDC-D0 04      BNE $EFE2   ---inconditionnel                                 I     I
EFDE-A0 00      LDY #$00    I   on indique  <-------------------------------------  I
EFE0-84 66      STY $66     I   pas d'extension                                    I
EFE2-C9 F9      CMP #$F9    -->plus de 8 décalages ?                              I
EFE4-30 C4      BMI $EFAA       oui, on décale un octet                           I
EFE6-A8         TAY             on indexe le nombre de décalages                  I
EFE7-A5 66      LDA $66         ACC1EX dans A                                     I
EFE9-56 01      LSR $01,X       on indique signe à 0                              I
```

```
EFEB-20 FC F0  JSR $F0FC      et on justifie                                  I
EFEE-24 6E     BIT $6E        on teste le produit des signes <----------------
EFF0-10 57     BPL $F049      signes identiques, on passe
EFF2-A0 60     LDY #$60       on indexe ACC1
EFF4-E0 68     CPX #$68       c'est ACC1 le plus grand ?
EFF6-F0 02     BEQ $EFFA      oui ------------------------
EFF8-A0 68     LDY #$68       non, c'est ACC2, on l'indexe I
EFFA-38        SEC            <--------------------------
EFFB-49 FF     EOR #$FF       on complémente l'extension
EFFD-65 7F     ADC $7F        +1+extension, on ajoute les extensions
EFFF-85 66     STA $66        dans ACC1EXE
F001-B9 04 00  LDA $0004,Y    soustrait octet 3 de la mantisse
F004-F5 04     SBC $04,X
F006-85 64     STA $64
F008-B9 03 00  LDA $0003,Y    octet 2
F00B-F5 03     SBC $03,X
F00D-85 63     STA $63
F00F-B9 02 00  LDA $0002,Y    octet 1
F012-F5 02     SBC $02,X
F014-85 62     STA $62
F016-B9 01 00  LDA $0001,Y    octet 0
F019-F5 01     SBC $01,X
F01B-85 61     STA $61
F01D-B0 03     BCS $F022      positif, on passe ------------------------------
F01F-20 90 F0  JSR $F090      négatif, on complémente                         I
F022-A0 00     LDY #$00       <-----------------------------------------------
F024-98        TYA            extension=0
F025-18        CLC            décalages=0
F026-A6 61     LDX $61        on prend l'octet 0 de la mantisse <-------------
F028-D0 4A     BNE $F074      pas 0, on décale bit à bit                      I
F02A-A6 62     LDX $62        sinon, on décalge octet 1 -> octet 0            I
F02C-86 61     STX $61                                                        I
F02E-A6 63     LDX $63        2 -> 1                                          I
F030-86 62     STX $62                                                        I
F032-A6 64     LDX $64        3 -> 2                                          I
F034-86 63     STX $63                                                        I
F036-A6 66     LDX $66        extension -> 3                                  I
F038-86 64     STX $64          :                                             I
F03A-84 66     STY $66        0 dans l'extension                              I
F03C-69 08     ADC #$08       on ajoute 8 aux décalages                       I
F03E-C9 28     CMP #$28       5 décalages faits ?                             I
F040-D0 E4     BNE $F026      non --------------------------------------------
F042-A9 00     LDA #$00       oui, 0 dans exposant
F044-85 60     STA $60        donc nombre nul
F046-85 65     STA $65        et 0 dans le signe
F048-60        RTS
```

ADDITION DES MANTISSES

```
F049-65 7F     ADC $7F        extension dans
F04B-85 66     STA $66        ACC1EX
F04D-A5 64     LDA $64        octets 3
F04F-65 6C     ADC $6C          :
F051-85 64     STA $64          :
F053-A5 63     LDA $63        octets 2
F055-65 6B     ADC $6B
F057-85 63     STA $63
F059-A5 62     LDA $62        octets 1
F05B-65 6A     ADC $6A
F05D-85 62     STA $62
F05F-A5 61     LDA $61        octets 0
```

```
F061-65 69      ADC $69
F063-85 61      STA $61
F065-4C 81 F0   JMP $F081       et on ajuste l'exposant
```

JUSTIFICATION BIT A BIT

```
F068-69 01      ADC #$01        on indique un décalage de plus  <----------------
F06A-06 66      ASL $66         on décale l'extension                           :
F06C-26 64      ROL $64         et la mantisse                                  :
F06E-26 63      ROL $63                                                         :
F070-26 62      ROL $62                                                         :
F072-26 61      ROL $61                                                         :
F074-10 F2      BPL $F068       si toujours pas assez, on boucle ----------------
F076-38         SEC             justification terminée
F077-E5 60      SBC $60         on ajuste l'exposant
F079-B0 C7      BCS $F042       exposant déborde, le nombre était trop petit
F07B-49 FF      EOR #$FF        on retrouve l'exposant
F07D-69 01      ADC #$01        par inversion
F07F-85 60      STA $60         dans ACC!E
F081-90 0C      BCC $F08F
F083-E6 60      INC $60         on incrémente l'exposant
F085-F0 40      BEQ $F0C7       et on sort avec OVERFLOW s'il revient à 0
F087-66 61      ROR $61         sinon, on décale la mantisse
F089-66 62      ROR $62         pour retrouver le signe
F08B-66 63      ROR $63
F08D-66 64      ROR $64
F08F-60         RTS
```

COMPLEMENTE LA MANTISSE

```
F090-A5 65      LDA $65         on complémente
F092-49 FF      EOR #$FF        le signe
F094-85 65      STA $65
F096-A5 61      LDA $61         octet 0
F098-49 FF      EOR #$FF
F09A-85 61      STA $61
F09C-A5 62      LDA $62         octet 1
F09E-49 FF      EOR #$FF
F0A0-85 62      STA $62
F0A2-A5 63      LDA $63         octet 2
F0A4-49 FF      EOR #$FF
F0A6-85 63      STA $63
F0A8-A5 64      LDA $64         octet 3
F0AA-49 FF      EOR #$FF
F0AC-85 64      STA $64
F0AE-A5 66      LDA $66         extension
F0B0-49 FF      EOR #$FF
F0B2-85 66      STA $66
F0B4-E6 66      INC $66         et on ajoute 1
F0B6-D0 0E      BNE $F0C6
F0B8-E6 64      INC $64
F0BA-D0 0A      BNE $F0C6
F0BC-E6 63      INC $63
F0BE-D0 06      BNE $F0C6
F0C0-E6 62      INC $62
F0C2-D0 02      BNE $F0C6
F0C4-E6 61      INC $61
F0C6-60         RTS
```

SORTIE PAR OVERFLOW

```
F0C7-A9 01      LDA #$01        on indexe erreur 1
F0C9-85 8B      STA $8B         dans FLERR
F0CB-A6 89      LDX $89
F0CD-9A         TXS             on restaure la pile sur l'appelant primaire
F0CE-60         RTS             et on sort
```

JUSTIFIER ACC3 A DROITE

```
F0CF-A2 6E      LDX #$6E
```

JUSTIFIER A DROITE SELON A ET X

Action: X contient l'index de l'accumulateur flottant concerné et A contient l'exposant de départ. On décale soit octet par octet (F0D1) soit bit par bit (F0F2) et cela jusqu'à ce que A vale 0.

```
F0D1-B4 04      LDY $04,X       octet 3 <-----------------------------------------
F0D3-84 66      STY $66         dans extension                                    I
F0D5-B4 03      LDY $03,X       octet 2                                           I
F0D7-94 04      STY $04,X       dans 3                                            I
F0D9-B4 02      LDY $02,X       1                                                 I
F0DB-94 03      STY $03,X       dans 2                                            I
F0DD-B4 01      LDY $01,X       0                                                 I
F0DF-94 02      STY $02,X       dans 1                                            I
F0E1-A4 67      LDY $67         code de remplissage (0 en général)                I
F0E3-94 01      STY $01,X       dans octet 0                                      I
F0E5-69 08      ADC #$08        on ajoute 8                                       I
F0E7-30 E8      BMI $F0D1       négatif                                           I
F0E9-F0 E6      BEQ $F0D1       ou nul, on continue octet par octet ---------------
F0EB-E9 08      SBC #$08        on récupère la valeur
F0ED-A8         TAY             en index
F0EE-A5 66      LDA $66         extension dans A
F0F0-B0 14      BCS $F106       si pas de décalages (Y=0) on sort -----------------
F0F2-16 01      ASL $01,X       on décale l'octet 0  <----------------------       I
F0F4-90 02      BCC $F0F8       pas de bit sorti ---------------------      I       I
F0F6-F6 01      INC $01,X       bit sorti, on ajoute 1               I      I       I
F0F8-76 01      ROR $01,X       on ajuste l'octet 0 <-----------------      I       I
F0FA-76 01      ROR $01,X       et on décale les 4                          I       I
F0FC-76 02      ROR $02,X                                                   I       I
F0FE-76 03      ROR $03,X                                                   I       I
F100-76 04      ROR $04,X                                                   I       I
F102-6A         ROR             et l'extension                              I       I
F103-C8         INY             on a fini ?                                 I       I
F104-D0 EC      BNE $F0F2       non, on boucle ------------------------------       I
F106-18         CLC             C=0 <-----------------------------------------------
F107-60         RTS
```

CONSTANTES DE BASE

```
F108            BYT $82,$13,$5D,$8D,$DE  soit 2,302585093 ou LN(10)
F10D            BYT $82,$49,$0F,$DA,$9E  soit 3,14159265  ou PI radians
F112            BYT $88,$34,$88,$00,$00  soit 180         ou PI degrés
```

COEFFICIENTS POLYNOME DE LN

```
F117            BYT $03                  pour 4 coefficients
F118            BYT $7F,$5E,$56,$CB,$79  soit 0,4342559419 soit presque 2/7LN(2)
F11D            BYT $80,$13,$9B,$0B,$64  soit 0,5765845412 # 2/5LN(2)
F122            BYT $80,$76,$38,$93,$16  soit 0,9618007592 # 2/3LN(2)
```

```
F127            BYT $82,$38,$AA,$3B,$20  soit 2.885390073  # 2/LN(2)
```

CONSTANTES POUR CALCUL LN

```
F12C            BYT $80,$35,$04,$F3,$34  soit 0,7071067812 ou SQR(2)/2
F131            BYT $81,$35,$04,$F3,$34  soit 1,414213562  ou SQR(2)
F136            BYT $80,$80,$00,$00,$00  soit -0,5
F13B            BYT $80,$31,$72,$17,$F8  soit 0,6931471806 ou LN(2)

F140-60         RTS                      pour sortie relative


F141-A9 02      LDA #$02                 on indexe erreur PARAMETRE NUL OU NEGATIF
F143-4C C9 F0   JMP $F0C9                et on sort avec erreur
```

FONCTION LN

Principe: La première des routines d'approximation de polynomes, et sans doute la plus compliquée. Le problème étant de ramener la valeur à calculer dans l'intervalle [0,1[. Pour cela on va poser les équivalences suivantes:
soit x = m*2^E alors LN(x)=LN(m)+LN(2^E)
d'oç LN(x)=LN(2)(E+ 1/LN(2) * LN(m))
Il est bien évident que 0<=m<1 donc on peut maintenant calculer le LN
Etant donné que m est plus près de 1 que de 0, on va calculer en fait LN(1-m). Plus précisement, LN(1 - SQR(2)/(m+SQR(2)/2) - 1/2).
Le cheminement jusqu'à ce résultat vous sera épargné, rassurez-vous !
Le D.L. de cette formule est :
LN(x)=2*(x+x3/3+x5/5+x7/7). Le tout étant finalement divisé par LN(2), ce qui explique que les coefficients soit 2/tLN(2).

Remarque: Les coefficients sont légèrement différents des valeurs données (sauf le premier, étant donné qu'ils résultent d'une transformation par les suites de CHEBYSHEV. (cf:chapitre sur les fonctions mathématiques).

```
F146-BA         TSX                      on sauve l'adresse de l'appelant
F147-86 89      STX $89                  dans FLSVS
F149-20 BD F3   JSR $F3BD                on prend le signe de ACC1
F14C-F0 F3      BEQ $F141                0, erreur
F14E-30 F1      BMI $F141                négatif, erreur aussi
F150-A5 60      LDA $60                  on prend l'exposant
F152-E9 7F      SBC #$7F                 -128 (C=0) donc on isole sa valeur
F154-48         PHA                      dans la pile
F155-A9 80      LDA #$80                 on place exposant +0
F157-85 60      STA $60                  dans ACC1E
F159-A9 2C      LDA #$2C                 on indexe SQR(2)/2
F15B-A0 F1      LDY #$F1
F15D-20 AF EF   JSR $EFAF                on ajoute à ACC1
F160-A9 31      LDA #$31                 on indexe SQR(2)
F162-A0 F1      LDY #$F1
F164-20 87 F2   JSR $F287                on divise SRQ(2)/ACC1
F167-A9 A5      LDA #$A5                 on indexe 1
F169-A0 F8      LDY #$F8
F16B-20 98 EF   JSR $EF98                on calcule 1-ACC1
F16E-A9 17      LDA #$17                 on indexe le polynome
F170-A0 F1      LDY #$F1
F172-20 E1 F6   JSR $F6E1                et on calcule P(ACC1)
F175-A9 36      LDA #$36                 on indexe -1/2
F177-A0 F1      LDY #$F1
```

```
F179-20 AF EF   JSR $EFAF       et on calcule ACC1-1/2
F17C-68         PLA             on prend l'exposant
F17D-20 E9 F9   JSR $F9E9       on calcule E+ACC1
F180-A9 3B      LDA #$3B
F182-A0 F1      LDY #$F1
F184-20 EC F1   JSR $F1EC       on multiplie par LN(2)
F187-F0 B7      BEQ $F140       si ACC1=0, on sort
F189-D0 05      BNE $F190       sinon, on multiplie ACC2*ACC1 soit ACC1*LN(2)
```

MULTIPLICATION ACC1*ACC2

Principe:Trivial puisque c'est le même qu'en décimal :
si xa=ma*2^Ea et xb=mb*2^Eb, on obtient pour xa*xb
xa*xb=ma*mb*2^(Ea+Eb). Il suffit donc de multiplier les mantisses en positionnant la multiplication comme en décimal et d'additionner les exposants (avec un justification éventuelle).
Le principe de la multiplication binaire consiste à faire 32 fois les opérations suivantes :
1-sortir un bit du multiplicateur
2-si ce bit est 1, ajouter le multiplicande au résultat partiel
3-décaler le multiplicande à gauche
4-si on n'a pas fait 32 tours, repartir en 1-
Cette méthode nécessite de stocker le résultat sur 32 bits puisqu'on décale 16 fois 16 bits. Les programmeurs de l'ATMOS ont contourné cette perte de mémoire en décalant le multiplicateur à gauche et le multiplicande à droite. Ainsi le résultat ne nécessite que 16 bits, mais les bits perdus à droite auraient pu ajouter au résultat; il s'ensuit donc une certaine perte de précision. Etait-il vraiment nécessaire d'économiser 8 octets ? Un petit coup d'optimisation à la BROCHE aurait permit d'obtenir des résultats plus probants.

```
F18B-F0 B3      BEQ F140        Z positionné selon ACC1E, si ACC1=0, on sort
F18D-BA         TSX             on sauve l'adresse
F18E-86 89      STX $89         de l'appelant dans FLSVS
F190-20 17 F2   JSR $F217       on additionne les exposants
F193-A9 00      LDA #$00        on met 0
F195-85 6F      STA $6F         dans ACC3 (mantisses)
F197-85 70      STA $70         l'exposant d'ACC3 est inexistant car
F199-85 71      STA $71         l'exposant de la multiplication est
F19B-85 72      STA $72         déja calculé en partie dans ACC1E
F19D-A5 66      LDA $66         on prend l'extension
F19F-20 B9 F1   JSR $F1B9       * ACC3
F1A2-A5 64      LDA $64         octet 3
F1A4-20 B9 F1   JSR $F1B9       * ACC3
F1A7-A5 63      LDA $63         octet 2
F1A9-20 B9 F1   JSR $F1B9
F1AC-A5 62      LDA $62         octet 1
F1AE-20 B9 F1   JSR $F1B9
F1B1-A5 61      LDA $61         octet 0
F1B3-20 BE F1   JSR $F1BE
F1B6-4C 01 F3   JMP $F301       et on passe ACC3 -> ACC1 avec justification
```

MULTIPLICATION PARTIELLE

```
F1B9-D0 03      BNE $F1BE       si l'octet à multiplier est nul
F1BB-4C CF F0   JMP $F0CF       on indique 8 décalages et on passe sans ajouter
F1BE-4A         LSR             pourquoi pas SEC/ROR ?
F1BF-09 80      ORA #$80        on met b7 à 1 pour compter 8 décalages
F1C1-A8         TAY             dans Y
```

```
F1C2-90 19      BCC $F1DD       si il ne faut pas additionner, on saute ----------
F1C4-18         CLC                                                             |
F1C5-A5 72      LDA $72         on additionne ACC2 à ACC3                       |
F1C7-65 6C      ADC $6C         octet 3                                         |
F1C9-85 72      STA $72                                                         |
F1CB-A5 71      LDA $71                                                         |
F1CD-65 6B      ADC $6B         octet 2                                         |
F1CF-85 71      STA $71                                                         |
F1D1-A5 70      LDA $70                                                         |
F1D3-65 6A      ADC $6A         octet 1                                         |
F1D5-85 70      STA $70                                                         |
F1D7-A5 6F      LDA $6F                                                         |
F1D9-65 69      ADC $69         octet 0                                         |
F1DB-85 6F      STA $6F                                                         |
F1DD-66 6F      ROR $6F         on décale ACC1 <--------------------------------
F1DF-66 70      ROR $70         donc le multiplicande à droite
F1E1-66 71      ROR $71
F1E3-66 72      ROR $72
F1E5-66 66      ROR $66
F1E7-98         TYA
F1E8-4A         LSR
F1E9-D0 D6      BNE $F1C1       et on fait pour les 8 bits du multiplicateur
F1EB-60         RTS
```

(AY) -> ACC2

```
F1EC-85 7D      STA $7D         on sauve l'adresse du nombre
F1EE-84 7E      STY $7E
F1F0-A0 04      LDY #$04        pour 5 octets
F1F2-B1 7D      LDA ($7D),Y     octet 3
F1F4-85 6C      STA $6C
F1F6-88         DEY
F1F7-B1 7D      LDA ($7D),Y     octet 2
F1F9-85 6B      STA $6B
F1FB-88         DEY
F1FC-B1 7D      LDA ($7D),Y     octet 1
F1FE-85 6A      STA $6A
F200-88         DEY
F201-B1 7D      LDA ($7D),Y     octet 0
F203-85 6D      STA $6D         dans le signe de ACC2
F205-45 65      EOR $65         produit signes ACC1S et ACC2S
F207-85 6E      STA $6E         dans ACCFS
F209-A5 60      LDA $6D         bit forcé à 1
F20B-09 80      ORA #$80
F20D-85 69      STA $69         dans octet 0
F20F-88         DEY
F210-B1 7D      LDA ($7D),Y     et l'exposant
F212-85 68      STA $68
F214-A5 60      LDA $60         en sortie A et P positionnés selon ACC1E
F216-60         RTS
```

SOMME DES EXPOSANTS ACC1E+ACC2E

Action: Additionne ACC1E et ACC2E dans ACC1E. Ce qui est plus compliqué qu'il n'y parait puisque d'après les conventions sur les exposants, #80=-1, #81=0 et #82=+1. d'ou +1 + +1 = #82+#82 = 4 avec dépassement, de même que -2 + -2 = #7F+#7F=#FE=+125 avec signe négatif.
En règle générale, la somme de deux exposants positifs donne un résultat positif qu'il suffit de majorer de #80 et la somme de deux exposants négatif un résultat négatif que l'on minorera de #80. Si les exposants

de signes opposés, pas de problème, on n'est sur de ne pas dépasser !

```
F217-A5 68      LDA $68         si ACC2=0
F219-F0 1C      BEQ $F237       on sort avec ACC1=0 ----------------------------
F21B-18         CLC                                                           I
F21C-65 60      ADC $60         on ajoute ACC1E                               I
F21E-90 04      BCC $F224       pas de dépassement                            I
F220-30 1A      BMI $F23C    ---dépassement et négatif, trop grand            I
F222-18         CLC           I C=0                                           I
F223            BYT $2C       I et on saute le BPL qui brancherait            I
F224-10 11      BPL $F237     I pas de dépassement et positif, nombre trop petit-0
F226-69 80      ADC #$80      I on ajoute 128                                 I
F228-85 60      STA $60       I pour récuper le bon résultat dans ACC1E       I
F22A-F0 13      BEQ $F23F    -----si 0, on sort avec ACC1>0                   I
F22C-A5 6E      LDA $6E      II on met le produit des signes                  I
F22E-85 65      STA $65      II dans ACC1S                                    I
F230-60         RTS          II                                               I
F231-A5 65      LDA $65      II on complémente le signe de ACC1 à 1           I
F233-49 FF      EOR #$FF     II s'il était négatif, N=0 et on sort            I
F235-30 05      BMI $F23C    II sinon, on indique OVERFLOW et on sort         I
F237-68         PLA          II on dépile l'adresse de retour  <--------------
F238-68         PLA          II pour retourner à l'appelant de l'appelant
F239-4C 42 F0   JMP $F042    II on sort avec ACC1=0
F23C-4C C7 F0   JMP $F0C7    I-->indique erreur OVERFLOW
F23F-4C 46 F0   JMP $F046    --->on indique ACC1 positif
```

10*ACC1 -> ACC1

Principe:Cette routine étant appelée très souvent, son optimisation est la bienvenue. Elle consiste à sauver ACC1, à ajouter 2 à son exposant, ce qui revient à multiplier ACC1 par 4, à ajouter la valeur sauvée, donc on obtient ACC1*5 et à ajouter à l'exposant pour tomber sur ACC1*10

```
F242-20 87 F3   JSR $F387       on met ACC1 dans ACC2
F245-AA         TAX             ACC1=0 ?
F246-F0 10      BEQ $F258       oui, on sort
F248-18         CLC
F249-69 02      ADC #$02        ACC1*4 (on ajoute 2 à l'exposant donc on
F24B-B0 EF      BCS $F23C       multiplie pas 2^2) si on dépasse, OVERFLOW
F24D-A2 00      LDX #$00        on met 0 dans le produit des signes
F24F-86 6E      STX $6E
F251-20 C2 EF   JSR $EFC2       +ACC2 donc *5
F254-E6 60      INC $60         et exposant +1 donc *2 donc finalement *10
F256-F0 E4      BEQ $F23C       si exposant nul, OVERFLOW
F258-60         RTS

F259            BYT $84,$20,$00,$00,$00 soit 10 en virgule flottante
```

ACC1/10 -> ACC1

```
F25E-20 87 F3   JSR $F387       on met ACC1 dans ACC2
F261-A2 00      LDX #$00        produit des signes à 0
F263-A9 59      LDA #$59        on indexe 10
F265-A0 F2      LDY #$F2
F267-86 6E      STX $6E
F269-20 23 F3   JSR $F323       on met 10 dans ACC1
F26C-4C 8A F2   JMP $F28A       et on calcule ACC2/ACC1
```

FONCTION LOG

Principe:on se sert de l'égalité LOG x = LN x / LN 10 .

```
F26F-BA        TSX             on sauve pour sortie après erreur
F270-86 89     STX $89
F272-20 49 F1  JSR $F149       on calcule LN de ACC1
F275-20 87 F3  JSR $F387       on arrondit ACC1 dans ACC2
F278-A9 08     LDA #$08        on indexe LN(10)
F27A-A0 F1     LDY #$F1
F27C-20 23 F3  JSR $F323       dans ACC1
F27F-4C 8A F2  JMP $F28A       et on divise ACC2 par ACC1
```

INDIQUE DIVISION PAR 0

```
F282-A9 03     LDA #$03        on indique erreur 3
F284-85 8B     STA $8B         dans FLERR
F286-60        RTS
```

DIVISION ACC2/ACC1

Principe: Si A1=M1*2^E1 et A2=M2*2^E2 alors A2/A1=M2/M1 * 2 (E2-E1).
Il suffit donc de diviser les mantisses et de soustraire les exposants.
Le procédé retenu par Fabrice BROCHE est (malheureusement) identique à celui retenu par Andy BROWN sur la V1.1. La routine est en fait la même alors qu'un effort de finition (dixit BROCHE !) aurait pu faire gagner de précieux octets.
Enfin, toujours est-il que la division binaire est dans son principe identique à la division décimale, c'est à dire que l'on doit essayer tous les chiffres de la base pour voir lequel convient. En binaire, il se trouve que seul 0 et 1 existent, les solutions sont donc triviales : Il suffit de soustraire le diviseur au dividende et de regarder le signe de cette différence. Positive, on ajoute 1 au résultat, négative 0. Et on décale le dividende jusqu'à ce qu'il soit nul.

```
F287-20 EC F1  JSR $F1EC       ACC1=0 ?
F28A-F0 F6     BEQ $F282       oui, division par 0
F28C-BA        TSX             non, on sauve le pointeur de pile
F28D-86 89     STX $89
F28F-20 96 F3  JSR $F396       on arrondit ACC1
F292-A9 00     LDA #$00        on veut E2-E1
F294-38        SEC             donc E2+(-E1)
F295-E5 60     SBC $60         donc on complémente ACC1E
F297-85 60     STA $60
F299-20 17 F2  JSR $F217       et on additionne les deux exposants
F29C-E6 60     INC $60         on ajoute 1 car inverse=complément à 2
F29E-F0 9C     BEQ $F23C       si le résultat est nul, il y a overflow.
F2A0-A2 FC     LDX #$FC        on indexe ACC3
F2A2-A9 01     LDA #$01        on prépare 8 décalages
F2A4-A4 69     LDY $69
F2A6-C4 61     CPY $61         on compare le diviseur au dividende
F2A8-D0 10     BNE $F2BA       s'ils sont différents, on passe ----------
F2AA-A4 6A     LDY $6A                                                  I
F2AC-C4 62     CPY $62                                                  I
F2AE-D0 0A     BNE $F2BA                                                I
```

```
F2B0-A4 6B      LDY $6B                                                 I
F2B2-C4 63      CPY $63                                                 I
F2B4-D0 04      BNE $F2BA                                               I
F2B6-A4 6C      LDY $6C                                                 I
F2B8-C4 64      CPY $64                                                 I
F2BA-08         PHP             on sauve le résultat de la comparaison <----------
F2BB-2A         ROL             on met 1 ou 0 dans le résultat
F2BC-90 0C      BCC $F2CA       si pas de débordement, on saute ------------------
F2BE-E8         INX             sinon, on sauve le résultat                     I
F2BF-95 72      STA $72,X                                                       I
F2C1-F0 05      BEQ $F2C8       s'il est nul, on passe ------------------------- I
F2C3-10 33      BPL $F2F8       si index positif, on sort                     I I
F2C5-A9 01      LDA #$01        on refait 8 décalages                           I
F2C7-2C         BYT $2C         on saute l'instruction suivante                 I
F2C8-A9 40      LDA #$40        on indexe l'extension                           I
F2CA-28         PLP             on récupère le résultat de la comparaision <-------
F2CB-B0 0E      BCS $F2DB       dividende > diviseur, on soustrait
F2CD-06 6C      ASL $6C         on décale le dividende
F2CF-26 6B      ROL $6B         à gauche
F2D1-26 6A      ROL $6A
F2D3-26 69      ROL $69
F2D5-B0 E3      BCS $F2BA       si un bit déborde, ACC2 > ACC1
F2D7-30 CB      BMI $F2A4       sinon, on compare ACC2 et ACC1
F2D9-10 DF      BPL $F2BA       si b7 de ACC2=0, alors ACC1<ACC2

F2DB-A8         TAY             on sauve le résultat provisoire
F2DC-A5 6C      LDA $6C         on soustrait ACC1 à ACC2
F2DE-E5 64      SBC $64
F2E0-85 6C      STA $6C
F2E2-A5 6B      LDA $6B
F2E4-E5 63      SBC $63
F2E6-85 6B      STA $6B
F2E8-A5 6A      LDA $6A
F2EA-E5 62      SBC $62
F2EC-85 6A      STA $6A
F2EE-A5 69      LDA $69
F2F0-E5 61      SBC $61
F2F2-85 69      STA $69
F2F4-98         TYA             on récupère le résultat et on décale le dividende
F2F5-4C CD F2   JMP $F2CD       pourquoi pas BCS ?

F2F8-0A         ASL             on récupère l'extension dans b7b6
F2F9-0A         ASL
F2FA-0A         ASL
F2FB-0A         ASL
F2FC-0A         ASL
F2FD-0A         ASL
F2FE-85 66      STA $66         on sauve dans ACC1EX
F300-28         PLP             on ajuste la pile
```

ACC3 -> ACC1

```
F301-A5 6F      LDA $6F         on transfère ACC3 dans ACC1
F303-85 61      STA $61
F305-A5 70      LDA $70
F307-85 62      STA $62
F309-A5 71      LDA $71
F30B-85 63      STA $63
F30D-A5 72      LDA $72
F30F-85 64      STA $64
```

```
F311-4C 22 F0   JMP $F022       et on (est)'e 'a marche
```

PI -> ACC1

```
F314-20 C7 F8   JSR $F8C7       quel mode de calcul d'angles ?
F317-F0 06      BEQ $F31F       radian ----------------------------------
F319-A9 12      LDA #$12        degre, on indexe 180                    :
F31B-A0 F1      LDY #$F1                                                :
F31D-D0 04      BNE $F323                                               :
F31F-A9 0D      LDA #$0D        on indexe 3.14159265 <-------------------
F321-A0 F1      LDY #$F1
```

(AY) -> ACC1

```
F323-85 7D      STA $7D         on sauve AY
F325-84 7E      STY $7E
F327-A0 04      LDY #$04
F329-B1 7D      LDA ($7D),Y     octet 3
F32B-85 64      STA $64
F32D-88         DEY
F32E-B1 7D      LDA ($7D),Y     octet 2
F330-85 63      STA $63
F332-88         DEY
F333-B1 7D      LDA ($7D),Y     octet 1
F335-85 62      STA $62
F337-88         DEY
F338-B1 7D      LDA ($7D),Y     octet 0
F33A-85 65      STA $65         on sauve le signe
F33C-09 80      ORA #$80        on force b7 à 1
F33E-85 61      STA $61         et on sauve l'octet 1
F340-88         DEY
F341-B1 7D      LDA ($7D),Y
F343-85 60      STA $60         exposant
F345-84 66      STY $66         et 0 dans l'extension
F347-60         RTS
```

ARRONDIT ACC1 DANS ACC5

```
F348-A2 73      LDX #$73        on indexe ACC5
F34A-2C         BYT $2C         et on saute
```

ARRONDIT ACC1 DANS ACC4

```
F34B-A2 78      LDX #$78        on indexe ACC4
F34D-A0 00      LDY #$00        Y=0
```

ARRONDIT ACC1 DANS XY

```
F34F-20 96 F3   JSR $F396       on arrondit ACC1
F352-86 7D      STX $7D         on sauve XY
F354-84 7E      STY $7E
F356-A0 04      LDY #$04        et on transfère comme d'habitude
F358-A5 64      LDA $64
F35A-91 7D      STA ($7D),Y     octet 3
F35C-88         DEY
F35D-A5 63      LDA $63
F35F-91 7D      STA ($7D),Y     octet 2
F361-88         DEY
F362-A5 62      LDA $62
F364-91 7D      STA ($7D),Y     octet 1
```

```
F366-88        DEY
F367-A5 65     LDA $65          octet 0
F369-09 7F     ORA #$7F
F36B-25 61     AND $61          avec signe
F36D-91 7D     STA ($7D),Y
F36F-88        DEY
F370-A5 60     LDA $60
F372-91 7D     STA ($7D),Y      et exposant
F374-84 66     STY $66
F376-60        RTS              Y=0
```

ACC2 -> ACC1

```
F377-A5 6D     LDA $6D          signe
F379-85 65     STA $65
F37B-A2 05     LDX #$05         et 5 octets de la mantisse
F37D-B5 67     LDA $67,X
F37F-95 5F     STA $5F,X
F381-CA        DEX
F382-D0 F9     BNE $F37D
F384-86 66     STX $66          et 0 dans l'extension
F386-60        RTS
```

ARRONDIT ACC1 DANS ACC2

```
F387-20 96 F3  JSR $F396        on arrondit ACC1
```

ACC1 -> ACC2

```
F38A-A2 06     LDX #$06         6 octets
F38C-B5 5F     LDA $5F,X        en transfert direct
F38E-95 67     STA $67,X
F390-CA        DEX
F391-D0 F9     BNE $F38C
F393-86 66     STX $66          et 0 dans l'extension
F395-60        RTS
```

ARRONDIT ACC1

Action:arrondit ACC1 selon le bit de l'extension , cela revient à ajouter 0,5 et à prendre la partie entière d'un nombre décimal.

```
F396-A5 60     LDA $60          ACC1=0 ?
F398-F0 FB     BEQ $F395        oui, on sort
F39A-06 66     ASL $66          non, bit d'extension ?
F39C-90 F7     BCC $F395        non, on sort
F39E-20 B8 F0  JSR $F0B8        oui, on incrémente la mantisse
F3A1-D0 F2     BNE $F395        si pas 0, on sort
F3A3-4C 83 F0  JMP $F083        si 0, on incrémente l'exposant
```

INT(ACC1)->AY

```
F3A6-A5 65     LDA $65          nombre négatif ?
F3A8-30 0E     BMI $F3B8        oui ----------------------------------------
F3AA-A5 60     LDA $60                                                      I
F3AC-C9 91     CMP #$91         nombre>65536 ?                              I
F3AE-B0 08     BCS $F3B8        oui ----------------------------------------0
F3B0-20 39 F4  JSR $F439        on convertit en entier                      I
F3B3-A5 64     LDA $64          et on sauve dans YA (pourquoi pas AY ?)     I
```

```
F3B5-A4 63      LDY $63
F3B7-60         RTS
F3B8-A9 0A      LDA #$0A        erreur 10 <----------------------------------
F3BA-4C C9 F0   JMP $F0C9       (nombre trop grand ou trop petit)
```

RETOURNE LE SIGNE DE ACC1

Action:retourne A=1 si ACC1>0, A=-1 si ACC1<0 ou A=0 si ACC1=0.

```
F3BD-A5 60      LDA $60         ACC1=0 ?
F3BF-F0 09      BEQ $F3CA       oui, on sort
F3C1-A5 65      LDA $65         non, négatif ?
F3C3-2A         ROL
F3C4-A9 FF      LDA #$FF        oui, A=255=-1
F3C6-B0 02      BCS $F3CA
F3C8-A9 01      LDA #$01        non, A=1
F3CA-60         RTS
```

SGN(ACC1)->ACC1

```
F3CB-20 BD F3   JSR $F3BD       prend signe de ACC1
F3CE-2C         BYT $2C         et saute l'instruction suivante
```

ACC1 = -1

```
F3CD-A9 FF      LDA #$FF        force signe négatif
```

A -> ACC1

```
F3D1-85 61      STA $61         on sauve A
F3D3-A9 00      LDA #$00
F3D5-85 62      STA $62         on annule le poids faible
F3D7-A2 88      LDX #$88        exposant=+7
F3D9-A5 61      LDA $61         on prend le signe
F3DB-49 FF      EOR #$FF        on l'inverse
F3DD-2A         ROL             dans C
F3DE-A9 00      LDA #$00        A=0
F3E0-85 63      STA $63         on annule la mantisse
F3E2-85 64      STA $64
F3E4-86 60      STX $60         on fixe l'exposant
F3E6-85 66      STA $66
F3E8-85 65      STA $65
F3EA-4C 1D F0   JMP $F01D       et on ajuste la mantisse selon l'exposant
```

INT(YA) -> ACC1 (non signé)

```
F3ED-85 61      STA $61         on sauve YA
F3EF-84 62      STY $62
F3F1-A2 90      LDX #$90        on fixe l'exposant à +15
F3F3-38         SEC
F3F4-B0 E8      BCS $F3DE       inconditionnel, on termine la conversion
```

FONCTION ABS

```
F3F6-46 65      LSR $65         on enlève le signe de ACC1
F3F8-60         RTS             et on sort
```

COMPARER (AY) ET ACC1

Action: retourne dans A -1,0 ou 1 selon le signe de ACC1-(AY). Cette routine, bien compliquée en apparence, nous plonge dans les délices des comparaisons en binaire signé.

```
F3F9-85 7D      STA $7D         on sauve AY
F3FB-84 7E      STY $7E
F3FD-A0 00      LDY #$00
F3FF-B1 7D      LDA ($7D),Y     on prend l'exposant
F401-C8         INY
F402-AA         TAX             dans Y
F403-F0 B8      BEQ $F3BD       si nombre nul, le resultat vient de ACC1
F405-B1 7D      LDA ($7D),Y     on prend le signe (intégrée à la mantisse)
F407-45 65      EOR $65         et on le compare à ACC1S
F409-30 B6      BMI $F3C1       les signes sont différents, on retourne selon ACC1
F40B-E4 60      CPX $60         on compare les exposants
F40D-D0 21      BNE $F430       différents, on passe (C contient le signe) -------
F40F-B1 7D      LDA ($7D),Y     on reprend la mantisse                          I
F411-09 80      ORA #$80        bit de signe à 1                                I
F413-C5 61      CMP $61         premier octet identique ?                       I
F415-D0 19      BNE $F430       non, C contient le signe ----------------------C
F417-C8         INY                                                             I
F418-B1 7D      LDA ($7D),Y     oui, octet suivant ?                            I
F41A-C5 62      CMP $62                                                         I
F41C-D0 12      BNE $F430       id.                                             I
F41E-C8         INY                                                             I
F41F-B1 7D      LDA ($7D),Y     octet 2                                         I
F421-C5 63      CMP $63                                                         I
F423-D0 0B      BNE $F430       id                                              I
F425-C8         INY                                                             I
F426-A9 7F      LDA #$7F        extension nulle ?                               I
F428-C5 66      CMP $66         oui, C=1, non C=0                               I
F42A-B1 7D      LDA ($7D),Y     on soustrait la mantisse 3                      I
F42C-E5 64      SBC $64         et l'extension d'ACC1                           I
F42E-F0 C8      BEQ $F3F8       si 0, alors nombres égaux                       I
F430-A5 65      LDA $65         on prend ACC1S <-------------------------------
F432-90 02      BCC $F436       si ACC1>(AY) on passe -------------------------
F434-49 FF      EOR #$FF        sinon, on inverse signe                         I
F436-4C C3 F3   JMP $F3C3       et on positionne A en sortie <------------------
```

ACC1 -> $64 63 62 61 en ENTIER SIGNE

Remarque: $67 n'est initialisé que si le nombre est négatif, il doit donc valoir 0 avant d'appeler la routine, sous peine de surprises !. Cet octet étant utilisé comme MENFY par la routine XMENU, il est normal que les résultats numériques soit erronnés en sortie de cette routine !

```
F439-A5 60      LDA $60         nombre nul ?
F43B-F0 4A      BEQ $F487       oui, nombre nul aussi et on sort
F43D-38         SEC             non, on retire exposant #A0
F43E-E9 A0      SBC #$A0        pour ramener à 0-31
F440-24 65      BIT $65         ACC1 positif ?
F442-10 09      BPL $F44D       oui ---------------------------------------------
F444-AA         TAX             exposant dans X                                 I
F445-A9 FF      LDA #$FF        on indique nombre négatif                       I
F447-85 67      STA $67         dans ACC1J                                      I
F449-20 96 F0   JSR $F096       on complémente la mantisse                      I
F44C-8A         TXA             et on récupère l'exposant                       I
```

```
F44D-A2 60      LDX #$60        on prépare les rotations <----------------
F44F-C9 F9      CMP #$F9        y a-t-il plus de 8 décalages à faire ?
F451-10 06      BPL $F459       non, on saute ----------------------------
F453-20 E5 F0   JSR $F0E5       oui, on fait les décalages
F456-84 67      STY $67         on remet 0
F458-60         RTS
F459-A8         TAY             exposant dans Y <-------------------------
F45A-A5 65      LDA $65         on prend le signe
F45C-29 80      AND #$80        dans b7
F45E-46 61      LSR $61         on reolace dans $61
F460-05 61      ORA $61
F462-85 61      STA $61
F464-20 FC F0   JSR $F0FC       et on décale la mantisse
F467-84 67      STY $67         et 0 dans ACC1J
F469-60         RTS
```

FONCTION INT

```
F46A-A5 60      LDA $60         on prend ACC1E
F46C-C9 A0      CMP #$A0        supérieur à 31 ?
F46E-B0 F9      BCS $F469       oui, on ne fait rien
F470-20 39 F4   JSR $F439       on convertit en entier 32 bits
F473-84 66      STY $66         extension à 0
F475-A5 65      LDA $65         on prend le signe
F477-84 65      STY $65         et on le fixe positif
F479-49 80      EOR #$80        on l'inverse
F47B-2A         ROL             dans C
F47C-A9 A0      LDA #$A0        exposant=31
F47E-85 60      STA $60
F480-A5 64      LDA $64         on prend le poids faible
F482-85 88      STA $88         dans FLINT (ACC1 est un entier)
F484-4C 1D F0   JMP $F01D       et on justifie la mantisse

F487-85 61      STA $61         A=0, on annule ACC1M
F489-85 62      STA $62
F48B-85 63      STA $63
F48D-85 64      STA $64
F48F-A8         TAY             et Y=0
F490-60         RTS
```

AX -> ACC1

```
F491-85 61      STA $61         on sauve le poids fort
F493-86 62      STX $62         et le poids faible
F495-A2 90      LDX #$90        exposant=+15
F497-38         SEC             C=1
F498-4C DE F3   JMP $F3DE       et on convertit en flottant
```

AFFICHE ACC1 EN DECIMAL

```
F49B-20 A5 F4   JSR $F4A5       convertit ACC1 en décimal
F49E-A9 00      LDA #$00        indexe BUFTRV (INUTILE ! AY l'indexe déja
F4A0-A0 01      LDY #$01        en sortie de la conversion décimale !!!)
F4A2-4C A8 C7   JMP $C7A8       et affiche le nombre
```

CONVERSION DECIMALE

Action:ACC1 est convertit en décimal ASCII dans BUFTRV ($100), terminé par 0.
En sortie AY indexe BUFTRV permettant l'affichage direct du nombre.
Principe:En deux parties. La première consiste à trouver le signe, la position d'un éventuel point décimal et l'éventuelle nécessité d'utiliser la

notatin scientifique.
La deuxième consiste à convertir véritablement le nombre ACC1. Le principe est universel, il faut soustraire des puissances de dix successives jusqu'à un resultat nul. Dans le cas de cette routine, les puissances de dix soustraites sont alternativement positives et négatives ce qui évite des comparaisons 32 bits fastidieuses. Quoiqu'il en soit, la routine est super balèze mais tant qu'elle mouline...

emarque:En sortie ACC1 est détruit.

```
F4A5-A0 00      LDY #$00        Y pointe sur le début du tampon
F4A7-A9 20      LDA #$20        A contient un espace (ce sera - si ACC1 < 0 )
F4A9-24 65      BIT $65         ACC1 positif ?
F4AB-10 02      BPL $F4AF    ---oui
F4AD-A9 2D      LDA #$2D     I  non, on prend "-"
F4AF-99 00 01   STA $0100,Y  -->dans le tampon
F4B2-85 65      STA $65         et dans ACC1S
F4B4-84 77      STY $77         on sauve le pointeur
F4B6-C8         INY             +1
F4B7-A9 30      LDA #$30        A="0"
F4B9-A6 60      LDX $60         X contient l'exposant
F4BB-D0 03      BNE $F4C0    ---Si non nul, on passe
F4BD-4C C8 F5   JMP $F5C8    I  si ACC1=0, on affiche "0" tout simolement !
F4C0-A9 00      LDA #$00     -->A=0
F4C2-E0 80      CPX #$80        exposant de l'ordre de 1 ?
F4C4-F0 02      BEQ $F4C8    ---oui
F4C6-B0 09      BCS $F4D1    I  plus grand que 1 ----------------------------------
F4C8-A9 D5      LDA #$D5     -->on indexe 1 000 000 000, qui est le plus          I
F4CA-A0 F5      LDY #$F5        grand nombre affichable directement               I
F4CC-20 84 F1   JSR $F184       on multiplie le nombre par 1e10                   I
F4CF-A9 F7      LDA #$F7        et exposant=-10                                   I
F4D1-85 74      STA $74         dans ACC4E <-------------------------------------
F4D3-A9 DA      LDA #$DA        on indexe 999 999 999
F4D5-A0 F5      LDY #$F5
F4D7-20 F9 F3   JSR $F3F9       on compare ACC1 à 1e10 - 1
F4DA-F0 1E      BEQ $F4FA       si égal, précision maximum, on saute -------------
F4DC-10 12      BPL $F4F0       si plus grand, on divise ------------------------  I
F4DE-A9 DF      LDA #$DF        si plus petit, on indexe 999 999,9             I  I
F4E0-A0 F5      LDY #$F5                                                       I  I
F4E2-20 F9 F3   JSR $F3F9       on compare de nouveau au nombre                I  I
F4E5-F0 02      BEQ $F4E9       égal, on multiplie ------------------------    I  I
F4E7-10 0E      BPL $F4F7    ---plus grand on arrête                        I  I  I
F4E9-20 42 F2   JSR $F242    I  on multiplie ACC1 par 10 <--------------------  I  I
F4EC-C6 74      DEC $74      I  et on enlève 1 à l'exposant                    I  I
F4EE-D0 EE      BNE $F4DE    I  et on continue à ajuster                       I  I
F4F0-20 5E F2   JSR $F25E    I  on divise ACC1/10 <-----------------------------  I
F4F3-E6 74      INC $74      I  on ajoute 1 à l'exposant pour compenser           I
F4F5-D0 DC      BNE $F4D3    I  inconditionnel:on continue à ajuster              I
F4F7-20 90 EF   JSR $EF90    -->on arrondit le nombre (+0.5)                      I
F4FA-20 39 F4   JSR $F439       on convertit en entier 32 bits <------------------
F4FD-A2 01      LDX #$01        on place le point décimal en 1 pour l'instant
F4FF-A5 74      LDA $74         on prend l'exposant base 10
F501-18         CLC
F502-69 0A      ADC #$0A        on ajoute 10 (1E10 étant le maximum affichable)
F504-30 09      BMI $F50F       si négatif, il faut la notation scientifique -----
F506-C9 0B      CMP #$0B        exposant > 10 ?                                   I
F508-B0 06      BCS $F510       oui, notation scientifique nécessaire ----------- I
F50A-69 FF      ADC #$FF        non, on calcule la place de la virgule         I  I
F50C-AA         TAX             dans X                                         I  I
```

```
F50D-A9 02     LDA #$C2
F50F-38        SEC
F510-E9 02     SBC #$02
F512-85 75     STA $75        A contient le deuxième octet
F514-86 74     STX $74        on sauve la position de la virgule
F516-8A        TXA            dans A
F517-F0 02     BEQ $F51B      ---si 0, on affiche tout de suite
F519-10 13     BPL $F52E      I  si positif, on n'affiche pas de virgule
F51B-A4 77     LDY $77        -->on reprend la position dans le tampon
F51D-A9 2E     LDA #$2E       A="."
F51F-C8        INY
F520-99 00 01  STA $0100,Y    on place le point décimal
F523-8A        TXA            nombre inférieur à 1 ?
F524-F0 06     BEQ $F52C      non
F526-A9 30     LDA #$30       oui,
F528-C8        INY            on place un "0"
F529-99 00 01  STA $0100,Y    devant la virgule
F52C-84 77     STY $77        on replace l'index
```

CONVERSION EN DECIMAL

```
F52E-A0 00     LDY #$00       indexe le début de la table
F530-A2 80     LDX #$80       X=0 (exposantement parlant)
F532-18        CLC            on ajoute la valeur courante dans la table
F533-A5 64     LDA $64        à la mantisse
F535-79 EC F5  ADC $F5EC,Y    octet 3
F538-85 64     STA $64
F53A-A5 63     LDA $63
F53C-79 EB F5  ADC $F5EB,Y    octet 2
F53F-85 63     STA $63
F541-A5 62     LDA $62
F543-79 EA F5  ADC $F5EA,Y    octet 1
F546-85 62     STA $62
F548-A5 61     LDA $61
F54A-79 E9 F5  ADC $F5E9,Y    octet 0
F54D-85 61     STA $61
F54F-E8        INX            +1
F550-B0 04     BCS $F556      ---si on dépasse, on saute
F552-10 DF     BPL $F533      I  partie addition, on continue
F554-30 02     BMI $F558      I  sinon, on passe
F556-30 DA     BMI $F532      -->partie soustraction, on continue
F558-8A        TXA            partie soustraction ?
F559-90 04     BCC $F55F      oui, on ne complémente pas
F55B-49 FF     EOR #$FF       on complémente à 10 en inversant
F55D-69 0A     ADC #$0A       et en ajoutant 10
F55F-69 2F     ADC #$2F       et "0"-1
F561-C8        INY            on ajoute 5 à l'index
F562-C8        INY            pour pointer sur l'élément suivant
F563-C8        INY
F564-C8        INY
F565-84 76     STY $76        dans $76
F567-A4 77     LDY $77        on lit le pointeur dans buffer d'affichage
F569-C8        INY            +1
F56A-AA        TAX            donnée dans X
F56B-29 7F     AND #$7F       on annule b7
F56D-99 00 01  STA $0100,Y    et on place le code
F570-C6 74     DEC $74        point décimal ?
F572-D0 06     BNE $F57A      non
F574-A9 2E     LDA #$2E       oui, on prend "."
F576-C8        INY
```

```
F577-99 00 01   STA $0100,Y     dans le buffer                                  I I
F57A-84 77      STY $77         on sauve l'index <------------------------------ I
F57C-A4 76      LDY $76         on reprend le pointeur de la table              I
F57E-8A         TXA             et le code                                      I
F57F-49 FF      EOR #$FF        on inverse                                      I
F581-29 80      AND #$80        et annule b7                                    I
F583-AA         TAX             dans X                                          I
F584-C0 24      CPY #$24        on a passé toute la table ?                     I
F586-D0 AB      BNE $F533       non, on boucle ---------------------------------
F588-A4 77      LDY $77         on reprend le pointeur d'affichage
F58A-B9 00 01   LDA $0100,Y     on prend le dernier code <---------------------
F58D-88         DEY             pour éliminer les zéros après la virgule        I
F58E-C9 30      CMP #$30        est-ce "0"                                      I
F590-F0 F8      BEQ $F58A       oui, on boucle ---------------------------------
F592-C9 2E      CMP #$2E        est-on sur le point décimal ?
F594-F0 01      BEQ $F597       oui, on le vire --------------------------------
F596-C8         INY             +1                                              I
F597-A9 2B      LDA #$2B        exposant positif <-----------------------------
F599-A6 75      LDX $75         on prend l'exposant base 10
F59B-F0 2E      BEQ $F5CB       si 0, on sort
F59D-10 08      BPL $F5A7       si positif, on ecrit ---------------------------
F59F-A9 00      LDA #$00                                                        I
F5A1-38         SEC                                                             I
F5A2-E5 75      SBC $75         on complémente l'exposant                       I
F5A4-AA         TAX                                                             I
F5A5-A9 2D      LDA #$2D        on affiche "-"                                  I
F5A7-99 02 01   STA $0102,Y     <----------------------------------------------
F5AA-A9 45      LDA #$45        et "E"
F5AC-99 01 01   STA $0101,Y
F5AF-8A         TXA
F5B0-A2 2F      LDX #$2F        on prépare "0"-1
F5B2-38         SEC
F5B3-E8         INX
F5B4-E9 0A      SBC #$0A        et on convertit l'exposant en décimal
F5B6-B0 FB      BCS $F5B3
F5B8-69 3A      ADC #$3A
F5BA-99 04 01   STA $0104,Y     dans le buffer
F5BD-8A         TXA
F5BE-99 03 01   STA $0103,Y     sur deux chiffres
F5C1-A9 00      LDA #$00        et on termine sur un 0
F5C3-99 05 01   STA $0105,Y
F5C6-F0 08      BEQ $F5D0
F5C8-99 00 01   STA $0100,Y     on sauve le code
F5CB-A9 00      LDA #$00
F5CD-99 01 01   STA $0101,Y     et le 0 de fin
F5D0-A9 00      LDA #$00        AY indexe BUFTRV
F5D2-A0 01      LDY #$01
F5D4-60         RTS             et on sort
```

CONSTANTES POUR CONVERSION DECIMALE

```
F5D5 BYT $9E,$6E,$6B,$28,$00  SOIT 1 000 000 000    (en flottant)
F5DA BYT $9E,$6E,$6B,$27,$FD  SOIT   999 999 999
F5DF BYT $9B,$3E,$BC,$1F,$FD  SOIT      999 999.5
F5E4 BYT $80,$00,$00,$00,$00  SOIT 0.5 (décidément ! on le voit partout !)

F5E9 BYT $FA,$0A,$1F,$00      SOIT -100 000 000 (en binaire 32 bits signés)
F5ED BYT $00,$98,$96,$80      SOIT   10 000 000
F5F1 BYT $FF,$F0,$BD,$C0      SOIT   -1 000 000
```

```
F5F5 BYT $00,$E1,$F5,$80        SOIT      100 000
F5F9 BYT $FF,$FF,$D8,$F0        SOIT      -10 000
F5FD BYT $00,$00,$03,$E8        SOIT        1 000
F601 BYT $FF,$FF,$FF,$9C        SOIT         -100
F605 BYT $00,$00,$00,$0A        SOIT           10
F609 BYT $FF,$FF,$FF,$FF        SOIT           -1

F60D-4C 42 F0   JMP $F042       on indique résultat nul
```

FONCTION SQR

```
F610-20 87 F3   JSR $F387       on arrondit ACC1 dans ACC2
F613-A9 E4      LDA #$E4        on indexe 1/2
F615-A0 F5      LDY #$F5
F617-20 23 F3   JSR $F323       on met (AY) dans ACC1
```

FONCTION ACC2^ACC1

```
Principe:X^Y=EXP(Y.LN(X)), avec les conventions Y=0 => X^Y=1 et X=0 => X^Y=0.
         Inutile de dire que l'exponentiation est une opération lente et peu
         précise, à éviter tant que possible !
Remarque:Pour des raisons de commodité, on appellera X et Y ACC2 et ACC1.
Amerie  :Si Y=0, on calcule EXP(0) au lieu de retourner simplement 1 !!!

F61A-F0 70      BEQ $F68C       si 0 on passe
F61C-8A         TSX             on sauve le pointeur de pile
F61D-86 89      STX $89
F61F-A5 68      LDA $68         X=0 ?
F621-F0 EA      BEQ $F60D       oui, résultat nul
F623-A2 80      LDX #$80        on indexe ACC6
F625-A0 00      LDY #$00
F627-20 52 F3   JSR $F352       et on sauve ACC1
F62A-A5 6D      LDA $6D         X positif ?
F62C-10 0F      BPL $F63D       ok ----------------------------------------
F62E-20 6A F4   JSR $F46A       sinon, on calcule INT(Y)                    I
F631-A9 80      LDA #$80                                                    I
F633-A0 00      LDY #$00                                                    I
F635-20 F9 F3   JSR $F3F9       on compare INT(Y) à Y                       I
F638-D0 03      BNE $F63D       différent, on passe ------------------------O
F63A-98         TYA             on indique signe positif                    I
F63B-A4 88      LDY $88         et on lit l'indicateur d'entier             I
F63D-20 79 F3   JSR $F379       X -> ACC1 <---------------------------------
F640-98         TYA             on sauve l'indicateur
F641-48         PHA
F642-20 49 F1   JSR $F149       on calcule LN(X)
F645-A9 80      LDA #$80        on indexe Y
F647-A0 00      LDY #$00
F649-20 84 F1   JSR $F184       on calcule Y*LN(X)
F64C-20 8F F6   JSR $F68F       puis EXP(Y.LN(X))
F64F-68         PLA             on sort la parité
F650-4A         LSR             positif ?
F651-90 0A      BCC $F65D       oui, on passe
```

NEGATION D'UN NOMBRE

```
F653-A5 60      LDA $60         nombre nul ?
F655-F0 06      BEQ $F65D       oui, on passe ------------
F657-A5 65      LDA $65         non, on complémente le     I
F659-49 FF      EOR #$FF        signe à 1                  I
```

```
F65B-85 65      STA $65                                  I
F65D-60         RTS             <------------------------

F65E BYT $81,$38,$AA,$3B,$29  soit 1/LN(2)
```

COEFFICIENT DU POLYNOME DE EXP(X)

```
F663 BYT $07                  pour 8 coefficients
F664 BYT $71,$34,$58,$3E,$56  soit 2.149876 E-05   (coefficient 7)
F669 BYT $74,$16,$7E,$B3,$1B  soit 0.0001435231
F66E BYT $77,$2F,$EE,$E3,$85  soit 0.0013422635
F673 BYT $7A,$1D,$84,$1C,$2A  soit 0.009614017
F678 BYT $7C,$63,$59,$58,$0A  soit 0.0555051269
F67D BYT $7E,$75,$FD,$E7,$C6  soit 0.2402263846
F682 BYT $80,$31,$72,$18,$10  soit 0.6931471862
F687 BYT $81,$00,$00,$00,$00  soit 1                (coefficient 0)
```

FONCTION EXP

```
Principe:le D.L. de EXP est EXP(X)=1+x+x^2/2+x^3/3!+..+x^n/n!
         D'après le théorème de Mc Laurin, ce D.L. n'est applicable que
         si x est voisin de 0, il faut donc ruser pour ramener notre calcul
         dans l'intervalle [0,1[ :
         EXP(X)=EXP(LN(2)/LN(2) * X)
               =(EXP(LN(2))^(X/LN(2))
               =2^(X/LN(2))
         soit Y=X/LN(2)=E+D (parties entières et décimales de Y)
         EXP(Y)=2^E.2^D or E est entier donc
                si 2^D=M*2^F, 2^D*2^E=M*2^(E+F)
                puisque  0<=D<1, on peut approche 2^D facilement.
         on calcule donc un D.L. de EXP sur X/LN(2)=Y qui est
         EXP(X)=EXP(Y.LN(2))=1+Y.LN(2)+...+(Y.LN(2))^n/n!  (OUF !)

F68C-BA         TSX             on sauve S pour sortie par erreur éventuelle
F68D-86 89      STX $89
F68F-A9 5E      LDA #$5E
F691-A0 F6      LDY #$F6        on indexe 1/LN2
F693-20 84 F1   JSR $F184       on calcule ACC1/LN(2)
F696-A5 66      LDA $66         on prend l'extension
F698-69 50      ADC #$50        pour arrondi
F69A-90 03      BCC $F69F
F69C-20 96 F3   JSR $F396       on arrondit si nécessaire
F69F-85 7F      STA $7F         et on sauve
F6A1-20 8A F3   JSR $F38A       on sauve ACC1 dans ACC2
F6A4-A5 60      LDA $60         exposant
F6A6-C9 88      CMP #$88        >8 ?
F6A8-90 03      BCC $F6AD       non, on passe
F6AA-20 31 F2   JSR $F231       oui, test d'overflow et signe de ACC1
F6AD-20 6A F4   JSR $F46A       on calcule INT(ACC1)
F6B0-A5 88      LDA $88         on prend le nombre
F6B2-18         CLC
F6B3-69 81      ADC #$81        +129 pour en faire un exposant
F6B5-F0 F3      BEQ $F6AA       si 0, overflow
F6B7-38         SEC             sinon, on ramène à +128
F6B8-E9 01      SBC #$01
F6BA-48         PHA             dans la pile
F6BB-A2 05      LDX #$05        on échange ACC1 et ACC2
F6BD-B5 68      LDA $68,X       ACC1=X/LN(2)
F6BF-B4 60      LDY $60,X       ACC2=INT(ACC1)
```

```
F6C1-95 60      STA $60,X
F6C3-94 68      STY $68,X
F6C5-CA         DEX
F6C6-10 F5      BPL $F6BD
F6C8-A5 7F      LDA $7F         on sauve aussi ACC2EX
F6CA-85 66      STA $66         dans ACC1EX
F6CC-20 9B EF   JSR $EF9B       on cacule la partie décimale ACC2-ACC1
F6CF-20 53 F6   JSR $F653       on inverse (donc ACC1=ACC1-ACC2)
F6D2-A9 63      LDA #$63
F6D4-A0 F6      LDY #$F6        on indexe le polynome EXP
F6D6-20 F7 F6   JSR $F6F7       on calcule
F6D9-A9 00      LDA #$00        on indique produit des signes positif
F6DB-85 6E      STA $6E
F6DD-68         PLA             on récupère la partie entière
F6DE-4C 19 F2   JMP $F219       on ajoute à l'exposant du résultat et on sort
```

CALCULE DE ACC1*P(ACC1*ACC1)

Principe:Certaines fonctions (comme COS et SIN) n'ont que des termes pair ou impairs. Afin de ne pas avoir à calculer tous les termes du D.L., donc avec un facteur sur deux nul, on calcule en fait le polynome du carre du nombre. On obtient ainsi un polynome à puissances paires quelle qu'elle fussent. On termine alors par multiplier le polynome par la variable afin de retrouver le polynome.
Ainsi le polynome P(x)=a0+X*(a1+x*(a2+x*(a3+x*(...+an*xn))))
devient P(x)=X*a1+X^3*a2+... dans le cas de SIN par exemple.

```
F6E1-85 85      STA $85         on sauve l'adresse du polynome
F6E3-84 86      STY $86
F6E5-20 48 F3   JSR $F348       on sauve ACC1
F6E8-A9 73      LDA #$73
F6EA-20 84 F1   JSR $F184       on calcule ACC1^2 dans ACC4
F6ED-20 FB F6   JSR $F6FB       puis P(ACC1^2)
F6F0-A9 73      LDA #$73
F6F2-A0 00      LDY #$00
F6F4-4C 84 F1   JMP $F184       et ACC1*P(ACC1^2)
```

CALCULE DE P(ACC1) DANS ACC1

Principe:comme précedemment, AY contient l'adresse des coefficients du polynome. Afin d'éviter le calcul laborieux des puissances successives de la variable, on calcule le polynome factorisé sous sa forme canonique :
P(X)=A0+X*(A1+X*(A2+X*(...+An*X))..))
Ce qui développé donne bien :
P(X)=A0+A1*X+A2*X^2+...+An*X^n

```
F6F7-85 85      STA $85         on sauve l'adresse des coefficients
F6F9-84 86      STY $86
F6FB-20 4B F3   JSR $F34B       ACC1 -> ACC4
F6FE-B1 85      LDA ($85),Y     on prend le nombre de coeffs
F700-85 87      STA $87         dans FLPO
F702-A4 85      LDY $85         on indexe la première valeur
F704-C8         INY
F705-98         TYA
F706-D0 02      BNE $F70A
F708-E6 86      INC $86
F70A-85 85      STA $85
F70C-A4 86      LDY $86         AY=adresse du premier coeff  An
```

```
:70E-20 84 F1   JSR $F184       on multiplie par la variable  <----------------
:711-A5 85      LDA $85         on indexe prochain coeff                        I
:713-A4 86      LDY $86.                                                        I
:715-18         CLC                                                             I
:716-69 05      ADC #$05                                                        I
:718-90 01      BCC $F71B                                                       I
:71A-C8         INY                                                             I
:71B-85 85      STA $85                                                         I
:71D-84 86      STY $86                                                         I
:71F-20 AF EF   JSR $EFAF       on additionne le résultat partiel au résultat   I
:722-A9 78      LDA #$78        on indexe la variable                           I
:724-A0 00      LDY #$00                                                        I
:726-C6 87      DEC $87         encore des coeffs ?                             I
:728-D0 E4      BNE $F70E       oui ------------------------------------------
:72A-60         RTS
```

VALEURS SUITE RND

```
:72B BYT $98,$35,$44,$7A,$68  soit 11879546,42
:730 BYT $68,$28,$B1,$46,$20  soit 3,927678 E-08
```

FONCTION RND

Principe: retourne dans ACC1 un nombre pseudo aléatoire calculé sur la base d'une suite récurrente Un+1=a.Un+b. Après quelques bidouilles et la mise à 0 de l'exposant, on sauve la valeur Un et on retourne dans ACC1. Si la valeur entrée dans ACC1 est négative, ACC1 devient la nouvelle U0 du générateur; si elle est nulle, on retourne la valeur courante.

```
:735-20 BD F3   JSR $BDF3       prend signe de ACC1
:738-AA         TAX             dans X
:739-30 18      BMI $F753       négatif, on change U0 ------------------------
:73B-A9 EF      LDA #$EF        on indexe la valeur courante                    I
:73D-A0 02      LDY #$02                                                        I
:73F-20 23 F3   JSR $F323       dans ACC1                                       I
:742-8A         TXA             signe dans A                                    I
:743-F0 E5      BEQ $F72A       0 ? on retourne la valeur courante              I
:745-A9 2B      LDA #$2B        on indexe la première valeur a                  I
:747-A0 F7      LDY #$F7                                                        I
:749-20 84 F1   JSR $F184       que l'on multiplie à ACC1                       I
:74C-A9 30      LDA #$30        et deuxième valeur b                            I
:74E-A0 F7      LDY #$F7                                                        I
:750-20 AF EF   JSR $EFAF       que l'on additionne à ACC1                      I
:753-A6 64      LDX $64         on permute octet 3 <---------------------------
:755-A5 61      LDA $61         et octet 0
:757-85 64      STA $64
:759-86 61      STX $61
:75B-A9 00      LDA #$00        signe positif
:75D-85 65      STA $65
:75F-A5 60      LDA $60         exposant dans extension
:761-85 66      STA $66
:763-A9 80      LDA #$80        exposant à 0
:765-85 60      STA $60
:767-20 22 F0   JSR $F022       on ajuste la mantisse
:76A-A2 EF      LDX #$EF
:76C-A0 02      LDY #$02
:76E-4C 52 F3   JMP $F352       et on envoie ACC1 dans la valeur courante
```

FONCTION RAND

Action:retourne dans ACC1 le nombre INT(RND(1)*ACC1).

```
F771-20 48 F3   JSR $F348       on arrondit ACC1 dans ACC5
F774-20 35 F7   JSR $F735       on prend un nombre RND dans ACC1
F777-A9 73      LDA #$73        AY pointe sur ACC5
F779-A0 00      LDY #$00
F77B-20 84 F1   JSR $F184       on multiplie le nombre RND par ACC5
F77E-4C 6A F4   JMP $F46A       et on prend sa valeur entière
```

FONCTIONS SIN ET COS

Principe:toujours le même, à ceci près que l'angle doit être modulo 2 PI. On commence donc par diviser le nombre par 2pi. L'angle est en fait ramené à l'intervalle [0,PI/2[étant données les propriétés de la fonction SIN Le Développement limité de SIN(X) est:

$$SIN(X) = X - \frac{X^3}{3!} + \frac{X^5}{5!} + \ldots + \frac{X^{(2n+1)}}{(2n+1)!}*(-1)^n$$

X étant ramené à 0,pi/2, donc à 0,1.57 les concepteurs on choisi de calculer le D.L. en 1. ce qui apporte une précision douteuse dès que X s'approche de 0[2PI].

Remarque:pour calculer le COS de ACC1, on calcule SIN(ACC1+PI/2).

COS

```
F781-20 B1 F8   JSR $F8B1       on convertit en radians si nécessaire
F784-A9 DC      LDA #$DC        on indexe PI/2
F786-A0 F7      LDY #$F7
F788-20 AF EF   JSR $EFAF       on ajoute PI/2 à ACC1
F78B-4C 91 F7   JMP $F791       et on calcule le SIN
```

SIN

```
F78E-20 B1 F8   JSR $F8B1       on convertit en radians si nécessaire
F791-20 87 F3   JSR $F387       on arrondit ACC1 dans ACC2
F794-A9 E1      LDA #$E1        on indexe 2*PI
F796-A0 F7      LDY #$F7
F798-A6 6D      LDX $6D         on prend le signe de ACC2 comme produit des signes
F79A-20 67 F2   JSR $F267       et on divise le nombre par 2*PI
F79D-20 87 F3   JSR $F387       on arrondit dans ACC2
F7A0-20 6A F4   JSR $F46A       et on calcule sa valeur entière
F7A3-A9 00      LDA #$00
F7A5-85 6E      STA $6E         on indique produit des signes positif
F7A7-20 9B EF   JSR $EF9B       on calcule la partie décimale ACC1-INT(ACC1)
F7AA-A9 E6      LDA #$E6
F7AC-A0 F7      LDY #$F7        on indexe 1/4
F7AE-20 98 EF   JSR $EF98       puis 1/4-ACC1 pour ramener ACC1 dans l'intervalle
F7B1-A5 65      LDA $65         on prend le signe
F7B3-48         PHA             que l'on sauve
F7B4-10 0F      BPL $F7C5       positif, on passe ----------------------------------
F7B6-20 90 EF   JSR $EF90       sinon on ajoute 1/2                               I
F7B9-A5 65      LDA $65                                                           I
F7BB-30 0B      BMI $F7C8       si négatif, ok ------------------------------------ I
F7BD-A5 8A      LDA $8A         sinon, on inverse le signe                      I  I
F7BF-49 FF      EOR #$FF                                                        I  I
F7C1-85 8A      STA $8A                                                         I  I
F7C3-24         BYT $24         et on saute le PHA                              I  I
F7C4-48         PHA                                                             I  I
```

```
F7C5-20 53 F6  JSR $F653      on inverse le signe de ACC1 <----------------+---
F7C8-A9 E6     LDA #$E6      on indexe 1/4 <-------------------------------
F7CA-A0 F7     LDY #$F7
F7CC-20 AF EF  JSR $EFAF     on ajoute à ACC1 pour être dans [-1/4,1/4]
F7CF-68        PLA           on récupère le signe
F7D0-10 03     BPL $F7D5     positif, on passe ----------------------------
F7D2-20 53 F6  JSR $F653     si négatif, on inverse le signe                 I
F7D5-A9 EB     LDA #$EB      on indexe le polynome de SIN <------------------
F7D7-A0 F7     LDY #$F7
F7D9-4C E1 F6  JMP $F6E1     et on calcule le polynome par méthode de parité
```

CONSTANTES POUR SIN ET COS

```
F7DC BYT $81,$49,$0F,$DA,$A2  soit PI/2 (1,570796327)
F7E1 BYT $83,$49,$0F,$DA,$A2  soit 2*PI (6,283185307)
F7E6 BYT $7F,$00,$00,$00,$00  soit 1/4  (0,25)
```

COEFFICIENTS DU POLYNOME SIN

```
F7EB BYT $05                  pour 6 coefficients
F7EC BYT $84,$E6,$1A,$2D,$1B  soit -14,38139067 coefficient 5
F7F1 BYT $86,$28,$07,$FB,$F8  soit 42,00779712
F7F6 BYT $87,$99,$68,$89,$01  soit -76,70417026
F7FB BYT $87,$23,$35,$DF,$E1  soit 81,60522369
F800 BYT $86,$A5,$5D,$E7,$28  soit -41.3417021
F805 BYT $83,$49,$0F,$DA,$A2  soit 2*PI          coefficient 0
```

FONCTION TAN

Principe:TAN(X)=SIN(X)/COS(X), la tangente est donc un calcul lent et peu précis
alors que c'est une fonction courante... un D.L. eut été bienvenu !

```
F80A-20 B1 F8  JSR $F8B1      on convertit en radians au besoin
F80D-20 48 F3  JSR $F348      on sauve ACC1 dans ACC5
F810-A9 00     LDA #$00
F812-85 8A     STA $8A        on indique signe positif
F814-20 91 F7  JSR $F791      on calcule SIN
F817-A2 80     LDX #$80       on indexe ACC6
F819-A0 00     LDY #$00
F81B-20 52 F3  JSR $F352      on y sauve SIN(ACC1)
F81E-A9 73     LDA #$73       on indexe ACC6 sauvé
F820-A0 00     LDY #$00
F822-20 23 F3  JSR $F323      dans ACC1
F825-A9 00     LDA #$00       signe positif
F827-85 65     STA $65
F829-A5 8A     LDA $8A        on prend le signe et
F82B-20 C4 F7  JSR $F7C4      on calcule le COS
F82E-A9 80     LDA #$80       on indexe SIN(ACC1)
F830-A0 00     LDY #$00
F832-4C 87 F2  JMP $F287      et on calcule SIN/COS
```

FONCTION ATN

Principe:Pour calculer l'arctangente, il faut d'abord ramener la variable à
l'intervalle [0,1[ce qui se fait par l'égalite :
ATN(X)=PI/2-ATN(1/X) et ATN(-X)=-ATN(X) (fonction impaire)

on calcule ensuite un DL de ATN'(X) qui est :

$$ATN'(X)=\frac{1}{1+X^2}$$

Un DL de ATN' est donc 1-x2+x4-...+(-1)^p*x^(2p)
Il suffit d'intégrer ce DL pour obtenir un DL de ATN :

ATN=x-x^3/3+x^5/5-...+(-1)^p.x^(2p+1)/(2p+1)
on trouvera donc comme coefficients du polynome des valeurs voisines
(CHEBYSHEV oblige) de 1,1/3,1/5,1/7...+/- 1/(2p+1)

```
F835-A5 65     LDA $65        on prend le signe
F837-48        PHA            on le sauve
F838-10 03     BPL $F83D   ---positif, on passe
F83A-20 53 F6  JSR $F653   I  négatif, on inverse
F83D-A5 60     LDA $60     -->on prend l'exposant
F83F-48        PHA            on le sauve
F840-C9 81     CMP #$81       supérieur à 1 ?
F842-90 07     BCC $F84B      non, on passe ----------------------------------
F844-A9 A5     LDA #$A5       on indexe 1                                     I
F846-A0 F8     LDY #$F8                                                       I
F848-20 87 F2  JSR $F287      on calcule 1/X                                  I
F84B-A9 6D     LDA #$6D       on indexe le polynome ATN <---------------------
F84D-A0 F8     LDY #$F8
F84F-20 E1 F6  JSR $F6E1      on calcule ATN(X) ou ATN(1/X) selon le cas
F852-68        PLA            le nombre était >1 ?
F853-C9 81     CMP #$81
F855-90 07     BCC $F85E      non, on passe ----------------------------------
F857-A9 DC     LDA #$DC       on indexe PI/2                                  I
F859-A0 F7     LDY #$F7                                                       I
F85B-20 98 EF  JSR $EF98      et on calcule PI/2-ATN(1/X)                     I
F85E-68        PLA            on récupère le signe de X <---------------------
F85F-10 03     BPL $F864   ---positif, ok
F861-20 53 F6  JSR $F653   I  négatif alors on renvoie -ATN(-X)
F864-20 C7 F8  JSR $F8C7   -->mode radian ?
F867-F0 03     BEQ $F86C   ---oui
F869-4C AA F8  JMP $F8AA   I  non, on convertit en degrés
F86C-60        RTS         -->on sort
```

COEFFICIENTS POLYNOME ATN

```
F86D BYT $0B                        pour 11 coefficients (quelle précision !)
F86E BYT $76,$B3,$83,$BD,$D3        soit -0,0006847939  coefficient 11
F872 BYT $79,$1E,$F4,$A6,$F5        soit  0,0048509422
F878 BYT $7B,$83,$FC,$B0,$10        soit -0,0161117018
F87D BYT $7C,$0C,$1F,$67,$CA        soit  0,034209638
F882 BYT $7C,$DE,$53,$CB,$C1        soit -0,0542791328
F887 BYT $7D,$14,$64,$70,$4C        soit  0,0724571965
F88C BYT $7D,$B7,$EA,$51,$7A        soit -0,0898023954
F891 BYT $7D,$63,$30,$88,$7E        soit  0,1109324134
F896 BYT $7E,$92,$44,$99,$3A        soit -0,1428398077
F89B BYT $7E,$4C,$CC,$91,$C7        soit  0,1999991205
F8A0 BYT $7F,$AA,$AA,$AA,$13        soit -0,3333333157
F8A5-BYT $81,$00,$00,$00,$00        soit  1                coefficient 0
```

CONVERTIT ACC1 EN DEGRES

```
F8AA-A9 BD     LDA #$BD       on indexe 180/PI
F8AC-A0 F8     LDY #$F8
```

```
F8AE-4C 84 F1  JMP $F184       et on calcule ACC1*180/PI
```

CONVERTIT ACC1 EN RADIANS SI MODE DEGRES ACTIF

```
F8B1-20 C7 F8  JSR $F8C7       mode radian actif ?
F8B4-F0 16     BEQ $F8CC       non, on sort sans convertir
```

CONVERTIT ACC1 EN RADIANS

```
F8B6-A9 C2     LDA #$C2        on indexe PI/180
F8B8-A0 F8     LDY #$F8
F8BA-4C 84 F1  JMP $F184       et on calcule ACC1*PI/180
```

CONSTANTES DE CONVERSION

```
F8BD BYT $86,$65,$2E,$E0,$D8  soit 57,29577959   ou 180/PI
F8C2 BYT $7B,$0E,$FA,$35,$19  soit  0,0174532926 ou PI/180
```

TESTE SI MODE DEGRES

```
F8C7-AD 0D 02  LDA #$20D       lit FLGTEL
F8CA-29 20     AND #$20        mode degrés ?
F8CC-60        RTS             oui, Z=0, non Z=1.
```

(AY)+ACC1 -> (AY)

```
F8CD-85 00     STA $00         on sauve AY dans RES
F8CF-84 01     STY $01
F8D1-20 AF EF  JSR $EFAF       calcule (AY)+ACC1
F8D4-A6 00     LDX $00         indexe RES
F8D6-A4 01     LDY $01
F8D8-4C 52 F3  JMP $F352       transfère ACC1 dans (RES)
```

CODE UN NOMBRE HEXA ASCII # DANS ACC1

Action:Le nombre hexadécimal à l'adresse (RES) est décodé dans ACC1.

```
F8DB-20 FC F9  JSR $F9FC   --->prend un code dans la phrase
F8DE-90 07     BCC $F8E7   I---chiffre, on passe
F8E0-C9 41     CMP #$41    II  est-ce au dessus de "A" ?
F8E2-90 31     BCC $F915   II  non, code invalide ----------------------------
F8E4-E9 37     SBC #$37    II  oui, on retire "A"-10 donc A=10, B=11 etc...     I
F8E6-2C        BYT $2C     II  et on saute                                      I
F8E7-E9 2F     SBC #$2F    I-->chiffre, on retire "0" (C=1)                     I
F8E9-C9 10     CMP #$10    I   caractère hexa valable ?                         I
F8EB-B0 28     BCS $F915   I   non ------------------------------------------O
F8ED-0A        ASL         I   oui, on mulptiplie par 16                       I
F8EE-0A        ASL         I                                                    I
F8EF-0A        ASL         I                                                    I
F8F0-0A        ASL         I                                                    I
F8F1-A2 04     LDX #$04    I   et 4 fois                                       I
F8F3-0A        ASL         I                                                    I
F8F4-26 62     ROL $62     I   on amène dans ACC1                              I
F8F6-26 61     ROL $61     I                                                    I
F8F8-B0 18     BCS $F912   I   overflow, on sort ------------------------------ I
F8FA-CA        DEX         I                                                I I
F8FB-D0 F6     BNE $F8F3   I                                                I I
F8FD-F0 DC     BEQ $F8DB   ----inconditionnel                               I I
                                                                            I I
```

CODE UN NOMBRE BINAIRE ASCII % DANS ACC1 I I

```
Action:Comme au dessus mais avec un nombre en duodécimal (binaire).

F8FF-20 FC F9   JSR $F9FC    ---on lit le code
F902-B0 11      BCS $F915    I  pas chiffre, on sort ----------
F904-C9 32      CMP #$32     I  est-ce "2" ?
F906-B0 0D      BCS $F915    I  au dessus, fin de code ---------
F908-C9 31      CMP #$31     I  est-ce 1 (C=1 si oui, 0 sinon)
F90A-26 62      ROL $62      I  on ajuste ACC1
F90C-26 61      ROL $61      I
F90E-B0 02      BCS $F912    I  si overflow, on sort ----------
F910-90 ED      BCC $F8FF    -->sinon, on continue
F912-4C C7 F0   JMP $F0C7       on indique OVERFLOW et on sort <----------
F915-A2 90      LDX #$90        exposant=+15 <-----------------------------
F917-38         SEC             C=1
F918-20 DE F3   JSR $F3DE       on justifie la mantisse de ACC1
F91B-A2 00      LDX #$00        X=0, opération OK
F91D-60         RTS             et on sort
```

CONVERTIT UN NOMBRE EN ASCII DANS ACC1

Action:le nombre à l'adresse AY est codé dans ACC1. Ce nombre (sous forme de chaine ASCII) peut être précédé de # (hexa) ou % (binaire). Le codage est plus rapide s'il est sous forme binaire, un peu moins sous forme hexa et carrément lourd sous forme décimale simple.

Principe:Aaaaah ! enfin une routine en béton ! un peu pompée sur la V1.1 mais mais suffisemment optimisée pour qu'elle devienne puissante.
Le princice est simple : pour # et % on est en entier.
Pour un nombre décimal, on code d'abord la mantisse (si X=3.32E-5, la mantisse est 3.32 et l'exposant -5). Puis selon que l'exposant est positif négatif, on divise ou on multiplie le nombre par 10 autant de fois que l'exposant est grand. l'exposant ayant été ajusté selon la position du point décimal.

Remarque:Contrairement à la V1.1, l'exposant peut être en minuscules...

```
F91E-85 00      STA $00         on sauve l'adresse de la chaine
F920-84 01      STY $01         dans RES
F922-BA         TSX             et le pointeur de pile
F923-86 89      STX $89         pour éventuelle sortie sur erreur
F925-A9 00      LDA #$00        on annule RESB (compteurs)
F927-85 02      STA $02
F929-85 03      STA $03
F92B-85 66      STA $66         et l'extension d'ACC1
F92D-A2 05      LDX #$05        ainsi que
F92F-95 60      STA $60,X       ACC1
F931-95 73      STA $73,X       et ACC4
F933-CA         DEX
F934-10 F9      BPL $F92F
F936-20 FE F9   JSR $F9FE       on lit le premier code
F939-90 16      BCC $F951       chiffre, alors c'est décimal ----------
F93B-C9 23      CMP #$23        est-ce "#"                            I
F93D-F0 9C      BEQ $F8DB       oui, on décode alors de l'HEXADECIMAL I
F93F-C9 25      CMP #$25        est-ce "%"                            I
F941-F0 BC      BEQ $F8FF       oui, on décode alors du DUODECIMAL (binaire)
F943-C9 2D      CMP #$2D        est-ce un "-" ?                       I
F945-F0 05      BEQ $F94C       oui, on place -1  dans $03 ----------  I
F947-C9 2B      CMP #$2B        non, est-ce "+" ?                  I  I
F949-D0 08      BNE $F953    ---non                                I  I
```

```
F94B-2C          BYT $2C      I  on saute l'instruction suivante          I          I
F94C-86 03       STX $03      I  on indique nombre négatif <--------------           I
F94E-20 FC F9    JSR $F9FC    I  on prend le caractère suivant <-------              <
F951-90 7A       BCC $F9CD    I  chiffre, on passe ---------------------+----------
F953-C9 2E       CMP #$2E     -->est-ce un point décimal ?               I          >
F955-F0 4F       BEQ $F9A6       oui ------------------------------------+---------- I
F957-C9 45       CMP #$45        exposant "E" ?                          I          I I
F959-F0 04       BEQ $F95F    ---oui                                     I          I I
F95B-C9 65       CMP #$65     I  non, "e" ?                              I          I I
F95D-D0 4D       BNE $F9AC    I  non, on sort ----------------------------+----------+-I
F95F-A6 02       LDX $02      -->on va traiter un exposant à 3 caractères maxi      III
F961-20 FC F9    JSR $F9FC       on lit un code                          I          III
F964-90 10       BCC $F976       chiffre, on passe ----------------------+--------- III
F966-C9 2D       CMP #$2D        est-ce "-" ?                            I        I III
F968-F0 05       BEQ $F96F       oui, on indique négatif                 I        I III
F96A-C9 2B       CMP #$2B        est-ce "+" ?                            I        I III
F96C-D0 2A       BNE $F998       non, fin du nombre ----------------------+------- I III
F96E-2C          BYT $2C         on saute l'instruction suivante         I      I I III
F96E-66 77       ROR $77         on indique exposant négatif             I      I I III
F971-20 FC F9    JSR $F9FC       on prend le code suivant                I      I I III
F974-B0 24       BCS $F99A       pas chiffre, on saute ------------------+----- I I III
F976-A5 75       LDA $75         on prend l'exposant <--------------------+---+-+-- III
F978-C9 0A       CMP #$0A        =10 ?                                   I   I I   III
F97A-90 09       BCC $F985       inférieur, on saute ---------------------I   I I   III
F97C-A9 64       LDA #$64        on prépare exposant=100                 II   I I   III
F97E-24 77       BIT $77         exposant positif ?                      II   I I   III
F980-30 11       BMI $F993       négatif, on saute ----------------------++-- I I   III
F982-4C C7 F0    JMP $F0C7       positif, OVERFLOW                       II I I I   III
F985-0A          ASL             on calcule les dizaines <---------------I I I I   III
F986-0A          ASL             *4                                      I I I I   III
F987-18          CLC             +exposant (donc *5)                     I I I I   III
F988-65 75       ADC $75                                                 I I I I   III
F98A-0A          ASL             *10                                     I I I I   III
F98B-18          CLC                                                     I I I I   III
F98C-A4 02       LDY $02                                                 I I I I   III
F98E-71 00       ADC ($00),Y     et on ajoute les unités                 I I I I   III
F990-38          SEC                                                     I I I I   III
F991-E9 30       SBC #$30        -"0" bien sur                           I I I I   III
F993-85 75       STA $75         on sauve l'exposant <-------------------+-- I I   III
F995-4C 71 F9    JMP $F971       et on continue l'exposant               I   I I   III
F998-86 02       STX $02         <---------------------------------------+---+--   III
F99A-24 77       BIT $77         exposant positif ? <--------------------+----     III
F99C-10 0E       BPL $F9AC       oui on termine ---------------------------+-------+0I
F99E-A9 00       LDA #$00        négatif                                 I         III
F9A0-38          SEC             on complémente l'exposant               I         III
F9A1-E5 75       SBC $75                                                 I         III
F9A3-4C AE F9    JMP $F9AE       et on termine ----------------------------+------   III
F9A6-66 76       ROR $76         on indique partie décimale <-------------+-----+-----II
F9A8-24 76       BIT $76                                                 I     I     II
F9AA-50 A2       BVC $F94E       deuxième virgule ------------------------     I     II
F9AC-A5 75       LDA $75         on retire à <---------------------------------+-----I
F9AE-38          SEC             l'exposant <-----------------------------------     I
F9AF-E5 74       SBC $74         le nombre de chiffres après la virgule             I
F9B1-85 75       STA $75                                                            I
F9B3-F0 12       BEQ $F9C7       si exposant nul , ok ---------------------------- I
F9B5-10 09       BPL $F9C0       exposant positif, on justifie ----------------  I I
F9B7-20 5E F2    JSR $F25E    ->exposant négatif, on divise par 10          I   I I
F9BA-E6 75       INC $75      I autant de fois qu'est l'exposant            I   I I
F9BC-D0 F9       BNE $F9B7    .--                                           I   I I
F9BE-F0 07       BEQ $F9C7       inconditionnel ------------------------------+--0 I
```

```
F9C0-20 42 F2  JSR $F242     ->positif, on multiplie par 10 <-----------------  I
F9C3-C6 75     DEC $75       I autant de fois qu'est l'exposant                  I
F9C5-D0 F9     BNE $F9C0     --                                                  I
F9C7-A5 03     LDA $03         on prend le signe <-------------------------------
F9C9-30 16     BMI $F9E1       négatif ------------------------------------------
F9CB-10 17     BPL $F9E4       positif ------------------------------------------ I
F9CD-48        PHA             on sauve le chiffre <-------------------------------+-+---
F9CE-24 76     BIT $76         partie entière ou décimale ?                      I I
F9D0-10 02     BPL $F9D4    ---entière, on saute                                 I I
F9D2-E6 74     INC $74      I  décimale, on indique un chiffre de plus           I I
F9D4-20 42 F2  JSR $F242    -->on multiplie ACC1 par 10                          I I
F9D7-68        PLA             on récupère le chiffre                            I I
F9D8-38        SEC                                                               I I
F9D9-E9 30     SBC #$30        -"0"                                              I I
F9DB-20 E9 F9  JSR $F9E9       on ajoute à ACC1                                  I I
F9DE-4C 4E F9  JMP $F94E       et on continue                                    I I
F9E1-20 53 F6  JSR $F653       signe négatif, on inverse ACC1 <------------------+--
F9E4-A2 00     LDX #$00        pas d'erreur <-------------------------------------
F9E6-4C 96 F3  JMP $F396       on sort en arrondissant ACC1 (plus précis)
```

ACC1+A -> ACC1

```
F9E9-48        PHA             on sauve A
F9EA-20 87 F3  JSR $F387       on arrondit ACC1 dans ACC2
F9ED-68        PLA             on récupère A
F9EE-20 D1 F3  JSR $F3D1       on la met dans ACC1
F9F1-A5 6D     LDA $6D         prend signe de ACC2
F9F3-45 65     EOR $65         sur signe de ACC1
F9F5-85 6E     STA $6E         dans produit des signes
F9F7-A6 60     LDX $60         on positionne Z selon ACC1
F9F9-4C B2 EF  JMP $EFB2       et on fait ACC1+ACC2
```

LIT UN CARACTERE SELON RES ET RESB

Action:retourne dans A le caractère lu dans (RES+RESB). Le caractère est passé en majuscules et en sortie, C=0 si c'est un chiffre. l'index (RESB) est incrémenté automatiquement avant la lecture.

```
F9FC-E6 02     INC $02         on incrémente l'index <--------
F9FE-A4 02     LDY $02                                       I
FA00-B1 00     LDA ($00),Y     on lit le caractère courant   I
FA02-20 F0 DF  JSR $DFF0       en majuscules                 I
FA05-C9 20     CMP #$20        espace ?                      I
FA07-F0 F3     BEQ $F9FC       oui, on ignore ----------------
FA09-C9 30     CMP #$30        <"0" ?
FA0B-90 09     BCC $FA10       oui ---------------
FA0D-C9 3A     CMP #$3A        non, chiffre ?    I
FA0F-60        RTS             non:C=1/oui:C=0   I
FA10-38        SEC             pas chiffre, C=1 <-
FA11-60        RTS
```

TESTE SI ACC1 EST UN ENTIER

Action:retourne A=1 si le nombre est un entier et A=0 sinon. Entier signifie ici entier long (entre -2^31 et 2^31 donc sur un intervalle discret de 8 Mo, soit 4 294 967 296 * 2).

```
FA12-20 87 F3  JSR $F387       ACC1 -> ACC2
FA15-A5 68     LDA $68         exposant nul ?
FA17-F0 19     BEQ $FA32       oui, nombre est donc 0 ----------
```

```
FA19-10 21      BPL $FA3C       exposant négatif donc nombre pas entier -------- I
FA1B-38         SEC                                                           I I
FA1C-A9 A1      LDA #$A1        exposant >32 ?                                I I
FA1E-E5 68      SBC $68                                                       I I
FA20-90 1A      BCC $FA3C       oui, nombre décimal ------------------------0 I
FA22-AA         TAX             non, on calcule la position de la virgule     I I
FA23-CA         DEX          -->                                              I I
FA24-F0 13      BEQ $FA39    I  virgule hors nombre : nombre entier ----------+-0
FA26-46 69      LSR $69      I  on décale le nombre                           I I
FA28-66 6A      ROR $6A      I                                                I I
FA2A-66 6B      ROR $6B      I                                                I I
FA2C-66 6C      ROR $6C      I                                                I I
FA2E-90 F3      BCC $FA23    ---rien ne sort, on continue                     I I
FA30-B0 0A      BCS $FA3C       on dépasse, le nombre est décimal ----------0 I
FA32-A2 03      LDX #$03        nombre nul                                    I I
FA34-95 69      STA $69,X       on vide ACC2                                  I I
FA36-CA         DEX                                                           I I
FA37-10 EA      BPL $FA23                                                     I I
FA39-A9 01      LDA #$01        et on indique ACC1 entier <-------------------+--
FA3B-60         RTS                                                           I
FA3C-A9 00      LDA #$00        on indique nombre décimal <-------------------
FA3E-60         RTS
```

TABLES DE CONVERSION DES CARACTERES

Principe: Ces trois tables contiennent les 3 images du clavier dans les 3 modes possibles (QWERTY, AZERTY et FRENCH). Les codes correspondent au code ASCII que doit renvoyer une touche. Pour retrouver le code d'une touche dans une table en connaissant ses ligne et colonne sur la matrice, il faut faire : code=(adresse+8*colonne+ligne).
Attention, la colonne 4 n'est pas codée donc pour les lignes entre 5 et 7, il faut ramener à 4 et 6.

```
FA3F    table clavier QWERTY
FA8F    table clavier AZERTY
FB1F    table clavier FRENCH (et bwana !)
```

CARACTERES PAR DEFAUT

Remarque: la table est codée, voir plus loin pour l'explication du décodage.

```
FB8F - FE44  table des caractères
```

CODES POUR CALCUL DES CARACTERES ALTERNES

```
FE45 BYT $00                soit %00000000
FE46 BYT $38                soit %00111000
FE47 BYT $07                soit %00000111
FE48 BYT $3F                soit %00111111
```

REPLACE LES CARACTERES PAR DEFAUT

Action: redéfinit les caractères avec leurs valeurs initiales. Les deux tables (caractères normaux et alternés) sont mises à jour grace à deux algos assez optimisés.
* Pour les caractères alternés:
un bloc (3,2 ou 3 lignes pour les étages 1,2 et 3) se définit ainsi: b=%00aabbcc. Les blocs aa,bb,cc (étages 3,2 et 1) donnent :

si 00 -> 00000000
   01 -> 00111000

```
      10 -> 00000111
      11 -> 00111111

   donc le code de numéro 33 (#21=%00100001) donnera :

   33=00111000
      00111000
      00111000
      00000000
      00000000
      00000111
      00000111
      00000111
```

* Pour les caractères normaux, les codes sont lus, b5-b0 isolés sont stockés en mémoire ET posons b7b6=aa :

si aa = 00 alors rien
01 alors on répète le code
10 alors on répète deux fois le code
11 alors on ajoute un 00000000 après le code

```
FE49-A9 B9     LDA #$B9       on indexe $B900 (alternés TEXT)
FE4B-2C 0D 02  BIT $020D      mode HIRES ?
FE4E-10 02     BPL $FE52
FE50-A9 9D     LDA #$9D       oui, on indexe $9D00 (alternés HIRES)
FE52-A0 00     LDY #$00
FE54-84 00     STY $00        on sauve l'adresse de la table
FE56-85 01     STA $01        dans RES
FE58-98        TYA            A et Y à 0
FE59-48        PHA            on sauve A <-------------------------------------
FE5A-20 B2 FE  JSR $FEB2      on calcule le caractère alterné correspondant    I
FE5D-68        PLA            on récupère A                                    I
FE5E-18        CLC                                                             I
FE5F-69 01     ADC #$01       on passe au code suivant                         I
FE61-C9 40     CMP #$40       on a fait 64 codes ?                             I
FE63-D0 F4     BNE $FE59      non ---------------------------------------------
FE65-A5 01     LDA $01        oui, on prends poids fort
FE67-E9 03     SBC #$03       -3 (retour à la base)
FE69-85 0C     STA $0C        dans TRO
FE6B-E9 04     SBC #$04       -7 (pour indexer les caractères normaux)
FE6D-85 01     STA $01        dans RES
FE6F-A9 8F     LDA #$8F       AY pointe sur ?
FE71-A0 FB     LDY #$FB
FE73-85 02     STA $02        dans RESB
FE75-84 03     STY $03
FE77-A0 00     LDY #$00       Y=0
FE79-A2 00     LDX #$00       X=0 <---------------------------------------------
FE7B-A1 02     LDA ($02,X     on lit un code                                   I
FE7D-AA        TAX            dans X                                           I
FE7E-E6 02     INC $02        on indexe le code suivant                        I
FE80-D0 02     BNE $FE84                                                       I
FE82-E6 03     INC $03                                                         I
FE84-20 9F FE  JSR $FE9F      on stocke le code                                I
FE87-8A        TXA                                                             I
FE88-29 C0     AND #$C0       on isole b7b6                                    I
FE8A-F0 ED     BEQ $FE79      si pas b7 et b6, on passe ----------------------C
FE8C-C9 C0     CMP #$C0       b7=1 et b6=1 ?                                   I
FE8E-F0 08     BEQ $FE98   ---oui                                              I
FE90-C9 40     CMP #$40    I  b7=0 et b6=1 ?                                   I
FE92-F0 06     BEQ $FE9A   I  oui ------------------------------------------- I
FE94-20 9F FE  JSR $FE9F   I  b7=1 et b6=0,répète 2 fois                   I  I
FE97-2C        BYT $2C     I  et on saute l'instruction suivante           I  I
```

```
FE9A-20 9F FE   JSR $FE9F       <------------------------------------------- I
FE9D-D0 DA      BNE $FE79       inconditionnel ------------------------------------
```

PLACE LE CODE X DANS UNE TABLE

```
FE9F-8A         TXA
FEA0-29 3F      AND #$3F        on isole b5-b0
FEA2-91 00      STA ($00),Y     on stocke X
FEA4-C8         INY             on indexe la prochaine position
FEA5-D0 0A      BNE $FEB1
FEA7-E6 01      INC $01         si on dépasse, on ajuste
FEA9-A5 01      LDA $01
FEAB-C5 0C      CMP $0C         a-t-on fini ?
FEAD-D0 02      BNE $FEB1       non, on sort
FEAF-68         PLA             oui, on retourne à l'appelant de l'appelant
FEB0-68         PLA
FEB1-60         RTS
```

PLACE UN CARACTERE ALTERNE EN MEMOIRE

```
Principe:pour placer les 8 octets de définition d'un caractère alterné A,
         un calcul est fait (voir plus haut) et l'adresse dans RES est ajustée.

FEB2-A2 03      LDX #$03        pour 3 étages
FEB4-86 02      STX $02         dans RESB
FEB6-48         PHA             on sauve le code
FEB7-29 03      AND #$03        on isole les bits de définition courants
FEB9-AA         TAX             dans X
FEBA-BD 45 FE   LDA $FE45,X     on lit le code
FEBD-91 00      STA ($00),Y     on le place
FEBF-C8         INY
FEC0-91 00      STA ($00),Y     deux fois
FEC2-C8         INY
FEC3-A6 02      LDX $02
FEC5-E0 02      CPX #$02        étage 2 ?
FEC7-F0 07      BEQ $FED0       oui, alors juste 2 lignes -------------------------
FEC9-91 00      STA ($00),Y     non, alors 3 lignes                              I
FECB-C8         INY             on déborde ?                                     I
FECC-D0 02      BNE $FED0       (on ne déborde que sur un multiple de 8 donc     I
FECE-E6 01      INC $01          sur l'étage 3). oui on ajuste l'adresse.        I
FED0-68         PLA             on récupère le code <-----------------------------
FED1-4A         LSR             on passe au deux bits suivants
FED2-4A         LSR
FED3-C6 02      DEC $02         on a fait 3 étages ?
FED5-D0 DF      BNE $FEB6       non, on boucle
FED7-60         RTS
```

DEPLACE LES CARACTERES TEXT -> HIRES

```
FED8-A0 05      LDY #$05        on indexe les codes pour transfert TEXT-HIRES
FEDA-2C         BYT $2C         et on saute l'index inverse
```

DEPLACE LES CARACTERES HIRES -> TEXT

```
FEDA-A0 0B      LDY #$0B        on indexe pour transfert HIRES-TEXT
FEDD-A2 05      LDX #$05        on indique 6 données
FEDF-B9 EB FE   LDA $FEEB,Y     on ajuste DECDEB,DECFIN et DECCIB pour décalage
FEE2-95 04      STA $04,X
FEE4-88         DEY
```

```
FEE5-CA        DEX
FEE6-10 F7     BPL $FEDF
FEE8-4C 6C CD  JMP $CD6C       et on décale
```

CODES POUR DEPLACEMENT DES TABLES DE CARACTERES

```
FEEB WRD $B400,$BB80,$9800     décalage TEXT -> HIRES
FEF1 WRD $9800,$9F80,$B400     décalage HIRES -> TEXT
```

REDEFINITION DE CARACTERES

action:redéfinit un bloc de caractères à l'adresse AY. Les caractères à
à redéfinir doivent être désignés comme suit :
code ascii caractère,8 octets de définition,ascii car.,8 octets,etc...
la fin de la table est désigné avec un 0 à la place du code ASCII.
La routine à deux entrées, FEF9 si on est sur la banque 7, FEF7 sinon.

```
FEF7-18        CLC             C=0 si les données se trouvent hors banque 7
FEF8-24        BYT $24
FEF9-38        SEC             C=1 si les données se trouvent sur la banque 7
FEFA-66 00     ROR $00         on indique par b7 de RES
FEFC-85 15     STA $15         on sauve l'adresse de la table
FEFE-84 16     STY $16
FF00-A0 00     LDY #$00        on place l'index à 0 <----------------------------
FF02-20 27 FF  JSR $FF27       on lit le code ASCII                             I
FF05-F0 1F     BEQ $FF26       0, on sort                                       I
FF07-20 31 FF  JSR $FF31       sinon, on calcule l'adresse du caractère         I
FF0A-E6 15     INC $15         on indexe les 8 octets de définition             I
FF0C-D0 02     BNE $FF10                                                        I
FF0E-E6 16     INC $16                                                          I
FF10-20 27 FF  JSR $FF27       on lit un octet de définition <---------------   I
FF13-91 02     STA ($02),Y     on place dans la table des caractères        I   I
FF15-C8        INY                                                          I   I
FF16-C0 08     CPY #$08        pour 8 codes                                 I   I
FF18-D0 F6     BNE $FF10       on boucle ------------------------------------   I
FF1A-98        TYA                                                              I
FF1B-18        CLC                                                              I
FF1C-65 15     ADC $15         on indexe le prochain code ASCII                 I
FF1E-85 15     STA $15                                                          I
FF20-90 DE     BCC $FF00                                                        I
FF22-E6 16     INC $16                                                          I
FF24-B0 DA     BCS $FF00       inconditionnel, on boucle ----------------------
FF26-60        RTS
```

LIT UN CODE POINTE PAR ($15) et Y

Action:l'octet pointé par $15-16 et Y est lu dans A. Selon b7 de RES, on force
ou non la banque 7. Cela permet de redéfinir les caractères plus vite si
on travaille sans se soucier de la banque en question.

```
FF27-24 00     BIT $00         données sur la banque 7 ?
FF29-10 03     BPL $FF2E       non, il faut gérer la banque ------------------
FF2B-B1 15     LDA ($15),Y     oui, on lit simplement                         I
FF2D-60        RTS                                                            I
FF2E-4C 11 04  JMP $0411       on lit en revenant à la banque appelant <-------
```

CALCULE L'ADRESSE D'UN CARACTERE

Principe:En sachant que #1300*8=#9800, et que l'adresse d'un code se calcule
par ADR=#9800+8*ASCII, alors il ne reste plus qu'à optimiser à fond.

L'astuce n'est pas de BROCHE mais de PETER HALFORD...

```
Action:Retourne dans RESB l'adresse du caractère de code ASCII dans A.

F31-A2 13     LDX #$13        on indexe $1300
F33-86 03     STX $03         dans $03
F35-0A        ASL             code *2
F36-26 03     ROL $03         dans adresse *2
F38-0A        ASL             *4
F39-26 03     ROL $03         dans adresse *4
F3B-0A        ASL             *8
F3C-26 03     ROL $03         dans adresse *8
F3E-85 02     STA $02         et fin dans $02
F40-2C 0D 02  BIT $020D       HIRES ?
F43-30 06     BMI $FF4B       oui, c'est bon comme ça ----------------------------
F45-A5 03     LDA $03         non, C=0                                           I
F47-69 1C     ADC #$1C        on ajoute $1C pour ramener à $B400                 I
F49-85 03     STA $03         dans RESB                                          I
F4B-60        RTS             <-------------------------------------------------
```

SELECTIONNE LE CLAVIER

```
Action:active le clavier selon A.
       valeurs possibles : 0 -> QWERTY
                           1 -> AZERTY
                           2 -> FRENCH
                           3 -> bwana (en fait un dérivé de FRENCH)
                           4 -> ACCENT SET
                           5 -> ACCENT OFF

F4C-C9 04     CMP #$04        ACCENT SET ?
F4E-F0 35     BEQ $FF85       oui ------------------------------------------------
F50-C9 05     CMP #$05        ACCENT OFF ?                                       I
F52-F0 38     BEQ $FF8C       oui ----------------------------------------------  I
F54-0A        ASL             on isole b0bi * 2                               I  I
F55-29 06     AND #$06                                                        I  I
F57-85 00     STA $00         dans RES                                        I  I
F59-AA        TAX             et X                                            I  I
F5A-AD 75 02  LDA $0275       on indique le type de clavier                   I  I
F5D-29 F9     AND #$F9        dans FLGKBD                                     I  I
F5F-05 00     ORA $00                                                         I  I
F61-8D 75 02  STA $0275       par superposition                               I  I
F64-BD 90 FF  LDA $FF90,X     on lit l'adresse de la table                    I  I
F67-BC 91 FF  LDY $FF91,X     de conversion dudit clavier                     I  I
F6A-85 2A     STA $2A         dans ADKBD                                      I  I
F6C-84 2B     STY $2B                                                         I  I
F6E-20 49 FE  JSR $FE49       on replace les caractères par défaut            I  I
F71-AD 75 02  LDA $0275                                                       I  I
F74-29 06     AND #$06        demande AZERTY ou FRENCH ?                      I  I
F76-C9 04     CMP #$04                                                        I  I
F78-D0 07     BNE $FF81                                                       I  I
F7A-A9 98     LDA #$98        FRENCH on indexe $FF98 pour accents             I  I
F7C-A0 FF     LDY #$FF                                                        I  I
F7E-4C F9 FE  JMP $FEF9                                                       I  I
F81-C9 02     CMP #$02        AZERTY on fait un ACCENT SET                    I  I
F83-D0 0A     BNE $FF8F       on sort (pourquoi pas FF4B ?)                   I  I
F85-A9 AA     LDA #$AA        on indexe $FFAA <-------------------------------+--
F87-A0 FF     LDY #$FF        (accents)                                       I
F89-4C F9 FE  JMP $FEF9       et on redéfinit                                 I
F8C-4C 49 FE  JMP $FE49       on remet les caractères par défaut <------------
F8F-60        RTS             et va-z-y !
```

```
FF90 WRD $FA3F                  TABLE QWERTY
FF92 WRD $FAAF                  TABLE AZERTY
FF94 WRD $FB1F                  TABLE FRENCH
FF96 WRD $FB1F                  TABLE bwana (donc FRENCH)
```

TABLE DES REDEFINITIONS ACCENT SET

Mystère:Théoriquement le [et le (c) devraient donner des accents inconnus...

```
FF98 BYT $5B                    a accent circonflexe (!?)
FF99 BYT $1C,$22,$1C,$02,$1E,$22,$1E,$00
FFA1 BYT $60                    u accent circonflexe (re-!?)
FFA2 BYT $1C,$22,$00,$22,$22,$26,$1A,$00
FFAA BYT $7B                    e accent aigu
FFAB BYT $04,$08,$1C,$22,$3E,$20,$1E,$00
FFB3 BYT $7D                    e accent grave
FFB4 BYT $10,$08,$1C,$22,$3E,$20,$1E,$00
FFBC BYT $40                    a accent grave
FFBD BYT $10,$08,$1C,$22,$1E,$22,$1E,$00
FFC5 BYT $5C                    c cédille
FFC6 BYT $00,$00,$1E,$20,$20,$20,$1E,$04
FFCE BYT $7C                    u accent grave
FFCF BYT $10,$08,$22,$22,$22,$26,$1A,$00
FFD7 BYT $7E                    e accent circonflexe
FFD8 BYT $1C,$22,$1C,$22,$3E,$20,$1C,$00
```

```
FFDF-00 00                      soit 28 octets pour rien... était-il vraiment la
FFE1-00 00                      peine de ralentir 40% des routines sous prétexte
FFE3-00 00                      de gain de mémoire pour téminer par cette horreur
FFE5-00 00                      sans commune mesure ?
FFE7-00 00
FFE9-00 00
FFEB-00 00
FFED-00 00
FFEF-00 00
FFF1-00 00
FFF3-00 00
FFF5-00 00
FFF7-00 00
FFF9-00 00
```

```
FFFB BYT $2F                    %00101111, soit banque 15 Ko avec ROM
FFFC WRD $C000                  soit reset à froid en $C000

FFFE WRD $02FA                  soit IRQ gérées en $02FA
```

ADRESSES COMMENTEES

Vous trouverez dans cette liste l'adresse (et un descriptif de ce qui s'y trouve) des 321 points du TELEMON commentés dans le listing.
Un découpage primaire du TELEMON a été fait afin d'alléger cette liste, ce qui permet une recherche plus rapide.

ROUTINES DE RESET

C000 routine principale du RESET
268 éteint le Minitel sur Canal 1
284 routine sans usage
29B affiche A en décimal
2A6 initialise les E/S
2E0 routine à transférer en $600
392 NMI par défaut
39B codes divers pour RESET
454 zone à tranférer en $B800

ROUTINES DE GESTION DES BUFFERS

C4EA retourne les valeurs par défaut
507 crée ou initialise un buffer
50C vide un buffer
50F teste un buffer
518 lit une donnée
51D écrit une donnée
524 routine à tranférer en $400
8AF valeurs par défaut des buffers

8BF teste l'imprimante

GESTION DES CANAUX

C6E5 ouvre une E/S sur un canal
720 ferme une E/S sur un canal
756 saute une ligne sur canal 0
75D envoie un code sur un canal
7A8 envoie une chaine sur un canal
7CF lit un code sur un canal
806 attend un code sur un canal
81C envoie une commande à une E/S
838 table des routines d'E/S

GESTION DES IRQ

C868 gestion du BRK
88C gestion normale des IRQ
8BF gere la RS232

C91E gestion timers, imprimantes et curseur
C992 Selection IRQ par T1 et T2
CA2E gestion imprimante
CA55 remet l'horloge à 0
CA63 arrête l'affichage régulier de l'horloge
CA75 active l'affichage régulier de l'horloge
CA90 affiche A en décimal deux chiffres
CAA4 table des vecteurs BRK

ROUTINES DE CALCULS SYSTEMES

CBE0 gestion de menus
CD6C décalage d'un bloc mémoire
CDDD données pour conversion décimale
CDE5 conversion décimale deux chiffres
CDEA id. 5 chiffres
CDED id. 4 chiffres
CDEF conversion décimale
CE39 conversion décimale avec affichage sur canal 0
CE69 multiplication par 40
CE83 addition
CE97 multiplication
CEDC division

GESTION MEMOIRE RAM

CF06 effacer l'écran HIRES
CF14 remplir une zone mémoire
CF45 passage en HIRES
CF75 passage en TEXT
CFA4 délai de 1/3 s
CFAF teste si tous les buffers sont vides
CFDC tables de définition du prompt

GESTION DE NOM DE FICHIER

CFF0 place un nom de fichier dans BUFNOM
D0F0 passe A en majuscules
D0FB teste si A est valide dans un nom de fichier

GESTION HIRES EMULATION VIDEOTEX

D111 initialise les tables VDT
D140 gère le curseur VDT
D178 envoie A sur l'écran VDT
D1ED poursuit une séquence VDT
D1F3 gère les codes de controle
D208 table des codes de controle
D22E gestion de séquence
D257 gestion du ESC
D2F0 gestion séquence Ci
D37F interprète les codes de controle

D442 code une donnée pour affichage VDT
D4A7 table des codes SYN et SS2
D4BC table des positions
D4C6 caractéres accentuables
D4DE codes VDT des caractères SS2/CAN
D4F1 code HRS -> VDT
D530 place une donnée VDT dans les tables
D57B stoppe l'émulation VIDEOTEX
D5BD initialise les flags VDT
D5C6 gestion de la sortie HIRES-VIDEOTEX
D5DC affichage d'un code en HIRES-VDT
D711 affiche un motif mosaique
D74D éteint le curseur VDT
D74F allume le curseur VDT
D756 affiche le curseur VDT
D759 efface le curseur VDT
D796 table de redéfinition pour VDT

GESTION DU CLAVIER

D7DF gestion du clavier
D81F conversion en ASCII
D8CF données bip clavier
D8DD gestion de FUNCT
D903 scrutation du clavier
D95C gestion de l'E/S clavier
D9A4 vide le buffer clavier
D981 initialise le clavier

GESTION DU PSG

D9E7 envoi 14 registres au PSG
DA1A envoi d'un registre au PSG
DA4F coupe le son du PSG

GESTION IMPRIMANTE

DA56 initialise l'imprimante
DA70 gère l'E/S imprimante parrallèle
DAE4 envoie un CRLF sur l'imprimante

GESTION RS-232

DAF7 gestion entrée MINITEL
DB12 gestion sortie MINITEL
DB54 initialise les valeurs RS-232
DB5D gestion entrée RS-232
DB79 gestion sortie RS-232

GESTION ECRAN MODE TEXT

DB86 gestion des fenêtres TEXT
DBB5 affiche A sur fenêtre X
DBE8 table de gestion des codes de controle
DC2B sortie de gestion des codes de controle
DC4C affichage d'un caractère
DC69 place un code sur l'écran
DCEB gestion des codes de controle
DD13 inverse l'état du curseur
DD47 déplace curseur à gauche
DD55 déplace curseur à droite
DD74 efface la ligne du curseur
DD7A efface la fin de la ligne du curseur
DD92 déplace le curseur à droite
DD9D déplace le curseur en bas
DDB8 efface la fenêtre
DDFB initialise une fenêtre
DE12 calcule l'adresse d'une ligne
DE1E éteint le curseur
DE20 allume le curseur
DE2D inverse le curseur à l'écran
DE54 scrolle une fenêtre vers le haut
DE5C scrolle une fenêtre vers le bas
DEE3 données pour fenêtres systèmes
DEFB crée une fenêtre
DF5B initialise l'affichage

GESTION DES PORTS JOYSTICK/SOURIS

DF90 lit valeur joystick
DF99 lit valeur souris
DFAB initialise les ports
DFF3 valeurs par défaut
DFFA gère joystick droit (rien)
DFF8 gère joystick gauche
E085 gestion souris
E0E1 gère boutons souris
E19D envoie ASCII selon souris
E19E envoie ASCII selon joystick

HARD-COPIES

E1B9 Hard-copy TEXT
E209 Hard-copy VIDEOTEX texte
E250 Hard-copy HIRES
E253 routine standard du HCOPY

ENTREE DE COMMANDE

E2DE place le curseur en fin de commande
E2F9 teste si prompt
E301 calcul première position post-prompt
E322 amène le curseur en fin de ligne
E34F copie la commande après le prompt
E355 copie la commande après le curseur

E3D0 affiche le contenu de BUFEDT
E435 routine d'édition de ligne de commande
E537 traite les caractères normaux
E5B9 gère les codes de controle
E62A positionne le curseur en X,Y
E648 envoie un code sur la sortie vidéo
E656 envoie un code en sortie série
E66C affiche le prompt
E680 cherche une ligne
E6B0 insère une ligne

TRAVAIL HIRES

E749 conversion ASCII -> binaire
E792 affiche curseur HIRES
E7C1 déplace curseur vers le bas
E7CD vers le haut
E7D9 vers la droite
E7E7 vers la gauche
E7F3 place le curseur en X,Y
E819 trace un rectangle en relatif
E82C trace un rectangle en absolu
E866 trace un trait en absolu
E885 trace un trait en relatif
E921 calcul la tangente d'un segment
E92F commande CURSET
E93C commande CURMOV
E942 vérifie les paramètres relatifs
E94E vérifie les paramètres absolus
E95D PAPER
E95F INK
E9CB CIRCLE
EA73 FILL
EAAF CHAR
EB0D PLAY
EB40 table des notes (octave -1)
EB5A MUSIC
EB73 SOUND
EBA0 données PING,SHOOT,EXPLODE,ZAP
EBD9 PING
EBDF SHOOT
EBE5 EXPLODE
EBEC ZAP

TRAVAIL MINITEL - RS-232

EC10 lit un code à l'entrée MINITEL
EC15 attend un code entrée MINITEL
EC1B écrit A en sortie RS
EC21 écrit A en sortie MINITEL
EC4F envoi de A avec gestion de CHECK
EC6B lit un code RS
ECB4 lit un code en entrée série
ECB9 attend un code en entrée série

ECBF-C1 ferme/ouvre buffer RS entrée
ECC7-C9 " RS sortie
ECCF-D1 " MINITEL entrée
ECD7-D9 " MINITEL sortie
ECDF calcule la taille d'un fichier
ECFD envoie l'entête d'un fichier
ED34 lit l'entête d'un fichier
ED77 CONSOLE
EDBA SDUMP
EDCA SSAVE
EDD7 MSAVE
EDE5 SLOAD
EDFC MLOAD
EE0A sauve fichier RS/MINITEL
EE56 lit fichier RS/MINITEL
EEA5 RING
EEB3 teste sonnerie
EF20 prise de ligne
EF30 envoi de séquence PRO1
EF3F UNCONNECT
EF4A WCXFIN
EF7A MOUT
EF85 SOUT

TRAVAIL MATHEMATIQUE ET NUMERIQUE

remarque:lorsque le contenant du résultat est omis, c'est qu'il s'agit de ACC1.

EF90 calcule ACC1+0,5
EF98 (AY)-ACC1
EF9B ACC2-ACC1
EFAA justifie la mantisse selon A
EFAF (AY)+ACC1
EFB2 ACC2+ACC1
F049 additionne les mantisses de ACC1 et ACC2
F068 justification bit à bit
F090 complémente la mantisse de ACC1
F0C7 sortie par OVERFLOW
F0C9 sortie flottant par erreur A
F0CF justifie ACC3 à droite
F0D1 justifie à droite
F108 constantes de base
F117 coefficients de calcul de LN
F12C constantes pour LN
F141 sortie par paramètre négatif ou nul
F146 LN
F184 AY*ACC1
F188 ACC2*ACC1
F1B9 multiplication partielle
F1EC (AY) ->ACC2
F217 addition des exposants
F242 10*ACC1
F25E ACC1/10
F26F LOG
F282 division par 0
F287 divise ACC2 par ACC1
F301 ACC3 -> ACC1
F314 PI -> ACC1 (F314, il fallait oser !)

323 (AY) -> ACC1
348 AACC1 -> ACC5
34B AACC1 -> ACC4
34F ACC1 -> (XY)
377 ACC2 -> ACC1
387 AACC1 -> ACC2
38A ACC1 -> ACC2
396 arrondit ACC1
3A6 INT(ACC1) -> AY
3BD cherche le signe de ACC1
3CB retourne le signe de ACC1 dans ACC1
3CD ACC1=-1
3D1 A -> ACC1
3ED INT(YA) -> ACC1
3F6 fonction ABS
3F9 compare (AY) et ACC1
439 convertit ACC1 en entier 32 bits
46A INT
491 AX -> ACC1
49B affiche ACC1 en décimal
4A5 conversion flottant -> décimal
5D5 constantes pour conversion
610 SQR
653 opérateur -
663 coefficients de EXP
68C EXP
6E1 calcule ACC1*P(ACC1^2)
726 suite RND
735 calcule RND(ACC1)
771 calcule RAND(ACC1)
781 COS
78E SIN
7C4 calcul COS sans mise à jour
7DC constantes pour SIN
7EB coefficients de SIN
80A TAN
835 ATN
86B coefficients ATN
8AA DEG
8B6 RAD
8BD constantes de conversion
8C7 teste si mode degré
8CD (AY)+ACC1 -> ACC1
8DB conversion ascii hexa -> ACC1
8FF conversion ascii binaire -> ACC1
91E conversion ASCII décimal
9E9 ACC1+A -> ACC1
9FC lit un code d'une chaine
A12 teste si ACC1 est un entier

GESTION DES TABLES DE CARACTERES

A3F table QWERTY
AAF table AZERTY
B1F table FRENCH
B8E table de redéfinition des caractères
E49 installe les caractères par défaut
E9F place X dans tables RES
EB2 redéfinit un caractère alterné

```
$ED8  transfert caractères TEXT -> HIRES
$EDA  transfert caractères HIRES -> TEXT
$EEB  codes pour déplacement caractères
$EF7  redéfinition de caractères
$F27  lit un code en mémoire selon $15-16
$F31  trouve l'adresse du caractère A
$F4C  redéfinit le clavier
$F98  table de définition d'accents
$FFB  vecteurs et octet de statut du TELEMON
```

LES VARIABLES DU TELEMON

Les variables listées ci-dessous, par plage d'utilisation, sont uniquement celles utilisées par le TELEMON. Ainsi, les variables TELEMATIC, TELE-ASS, HYPER-BASIC ou même STRATSED n'y figurent pas. Ces variables figurent dans l'annexe à la fin du livre.
Les variables du TELEMON, reconnues d'ailleurs par TELE-ASS, sont parfois connues sous plusieurs noms étant donnés leurs usages. Vous verrez d'ailleurs que les variables multi-usage n'ont pas spécialement été bien choisies !

Il faut de plus distinguer quatre types de variable. Ces quatre types sont : Les adresses, les drapeaux, les variables de travail et les valeurs. Les adresses doivent être modifiées à bon escient. Les variables de travail peuvent servir mais leur modification ne change rien. Quant aux flags (drapeaux) et aux valeurs, leur intérêt réside dans l'effet instantané que crée leur modification.

LA PAGE 0

TRAVAIL GENERAL

RES	$00	
RESB	$02	
DECDEB	$04	paramètres pour décalage
DECFIN	$06	
DECCIB	$08	
DECTRV	$0A	
TR0	$0C	travail général
TR1	$0D	
TR2	$0E	
TR3	$0F	
TR4	$10	
TR5	$11	
TR6	$12	
TR7	$13	
DEFAFF	$14	caractère à afficher par défaut lors de conversion décimale
	$15	adresse de lecture inter-banques
	$17	inutilisés
	$18	
	$19	travail canaux
	$1A	compteur E/S
	$1B	sauvegarde E/S
	$1C	travail E/S
	$1D	sauvegarde donnée E/S
	$1E	indicateur transition libre/liée sur RS232
	$1F	inutilisés

```
         $20

IRQSVA   $21     sauvegarde des registres en entrée d'IRQ
IRQSVX   $22
IRQSVY   $23

         $24     inutilisés
         $25
```

VARIABLES ECRAN

```
ADSCR    $26     adresse du début de la ligne courante écran
SCRNB    $28     valeur du numéro de fenêtre courante
         $29     sauvegarde SCRNB
```

VARIABLE CLAVIER

```
ADKBD    $2A     adresse table de conversion ASCII
```

VARIABLES VIDEOTEX HIRES

```
ADVDT    $2C     adresse écran HIRES
ADASC    $2E     adresse table ASCII
ADATR    $30     adresse tables de attributs

VDTPAR   $32     travail VDT:
                 b7    : attribut G0 à valider
                 b6-b4 : G0-G1 couleur de fond
                 b2    : G0 soulignage

VDTASC   $33     travail VDT:
                 b7    : délimiteur
                 b6    : G1 disjoint/non disjoint
                 b5-b0 : motif G1

VDTATR   $34     travail VDT:
                 b7    : G1 on/off
                 b6    : indic. vidéo inverse/normale en G0
                 b5    : indic. double largeur G0
                 b4    : indic. double hauteur
                 b3    : G0-G1 clignotement
                 b2-b0 : G0-G1 couleur de texte

VDTFT    $35     b7=1 si SS2
                 b6=1 si SEP

         $36     travail VIDEOTEX
         $37

VDTX     $38     coordonnées écran (G0-G1) colonne
VDTY     $39     et ligne
VDTGX    $3A     position code graphique G1: 0-1 pour la colonne
VDTGY    $3B     et 0-2 pour la ligne

FLGVDO   $3C     b7=1 si séquence en cours
                 b6=1 si ESC
                 b4=1 si US
                 b3=1 si REP
                 b1-b0 donnent le nombre de caractères de la séquence
```

LGVD1	$3D	b7=1 si curseur allumé b6=1 si mode graphique b1 : mode trace/efface graphique b0 : ??
	$3E	sauvegarde donnée VDT
	$3F	inutilisé

VARIABLES HORLOGE

ADCLK	$40	adresse d'affichage de l'horloge
TIMEUS	$42	décompteur 16 bits en seconde (utilisateur)
TIMEUD	$44	décompteur 16 bits en dixièmes de secondes (id.)

VARIABLES HIRES

HRSX	$46	position X du curseur HIRES
HRSY	$47	position Y
	$48	inutilisé
HRSX40	$49	X/6 (partie entière)
HRSX6	$4A	reste de X/6 (position du point dans le sextet écran)
ADHRS	$4B	adresse de la ligne du curseur
HRS1	$4D	paramètres pour fonction HIRES et musicales
HRS2	$4F	les 5 params sont des entiers signés (-32768 .. 32767)
HRS3	$51	
HRS4	$53	
HRS5	$55	
HRSFB	$57	b7-b6 contiennent le FB d'affichage (0,1,2 ou 3)
	$58	inutilisé

VARIABLES RS-232

RS232T	$59	b0-b3 : vitesse de transfert (de 75 à 19200 bauds) b4=1 si horloge interne b6-b5 : mode de transmission 00 = 8 bits 01 = 7 10 = 6 11 = 5 bits b7=1 si un seul stop
RS232C	$5A	b0-b3 : doivent valoir 0 b4=1 si écho SCR demandé (SDUMP) b5=1 si parité requise b7-b6 : mode de parité : 00 parité impaire en emission/réception 01 id. mais parité paire 10 parité émise, réception non testée 11 SPACE émis, réception non testée
	$5B	indicateur RS b7=1 si mode MINITEL, 0 si RS-232 b6=1 si entête lors de transferts, 0 sinon

TRAVAIL SYSTEME

SCEDEB	$5C	servent à l'insertion et à la
SCEFIN	$5E	recherche de ligne dans une zone mémoire
	$5F	inutilisé

GESTION DE MENUS

:N	$60	variable de travail
:NDDX	$61	coordonnées du premier choix. Colonne
:NDDY	$62	et ligne
:NDFY	$63	et dernière ligne de la fenêtre de choix
:NX	$64	travail choix
:NLX	$65	longueur de la barre
:NDY	$66	position absolue de la fenêtre (colonne)
:NFY	$67	(ligne) = ACC1J, d'où erreurs de calcul après XMENU
_GMEN	$68	drapeau de travail
)MEN	$69	table des choix

VARIABLES FLOTTANT

CC1E	$60	exposant de l'accumulateur 1
CC1M	$61	mantisse (32 bits)
CC1S	$65	signe
CC1EX	$66	extension (b7)
CC1J	$67	indicateur pour conversion décimale (doit valoir 0)
CC2E	$68	accumulateur 2
CC2M	$69	
CC2S	$6D	
CCPS	$6E	produit des signes ACC1S EOR ACC2S
CC3	$6F	accu 3
CC4E	$73	accu 4
CC4M	$74	
CC5E	$78	accu 5
CC5M	$79	
LTR0	$7D	drapeaux de travail
LTR1	$7E	
CC6	$80	accu 6
LPOLP	$85	adresse du polynome de travail
LPOO	$87	nombre de valeur du polynome
LINT	$88	b7=1 si ACC1 entier
LSVS	$89	sauvegarde de SP pendant travail flottant
LSGN	$8A	signe de ACC1
LERR	$8B	1 si sortie par overflow, 3 si division par 0.

PAGE 1

BUFTRV	$100	travail numérique et alphanumérique (seulement 16 octets, le reste étant la pile 6502)

PAGE 2

BANQUES

BANKST	$200	valeur des octets $FFF8 de chaque banque

VARIABLES INITIALISATION

TABDRV	$208	table d'activation d'un lecteur. le lecteur x (0..3) est activé si (TABDRV+x)<>0, alors TABDRV+x contient le nombre de pistes, pour formatage, et b7=1 si la disquette est double face.
DRVDEF	$20C	numéro du lecteur par défaut
FLGTEL	$20D	drapeau système: b7=1 si HIRES b6=1 si MINITEL en terminal vidéo b5=1 si mode degré b2=1 si BONJOUR.COM détecté b1=1 si imprimante parrallèle détectée b0=1 si STRATSED absent en mémoire
XOROM	$20E	taille mémoire trouvée en ROM
XORAM	$20F	et en RAM.

VARIABLES IRQ

TIMED	$210	horloge, dixièmes
TIMES	$211	secondes
TIMEM	$212	minutes
TIMEH	$213	heures
FLGCLK	$214	b7=1 si il faut afficher l'horloge toutes les secondes
	$215	flag compteur horloge

VARIABLES ECRAN

	$216	drapeau gestion curseur
	$217	drapeau état curseur
WDSCRL	$218	adresse de chaque fenêtre (octet faible)
WDSCRH	$21C	(octet fort)
WCRX	$220	position du curseur dans la fenêtre (colonne)
WCRY	$224	(ligne)
WCRDX	$228	coordonnées de la fenêtre (colonne gauche)
WCRFX	$22C	(ligne haut)
WCRDY	$230	(colonne droite)
WCRFY	$234	(ligne bas)
WCRBAL	$238	adresse de base de la fenêtre (poids faible)
WCRBAH	$23C	(poids fort)
WCRCT	$240	couleur de texte
WCRCF	$244	couleur de fond
FLGSCR	$248	drapeau curseur: b7=1 si il faut afficher le curseur b6=1 si curseur fixe b5=1 si vidéo inverse b4=1 si 38 colonnes, 0 si 40 colonnes b3=1 si prochain code ESC xx b2=1 si séquence US en cours b1=1 si double hauteur b0 compteur US
CURSCR	$24C	caractère sous le curseur
JPRVEC	$24E	erreur, ce vecteur ne vecte rien !
	$250	vecteur hard-copy HIRES
	$251	4 octets inutilisés

:CRTXT	$256	définition fenêtre TEXT
:CRHIR	$25C	définition fenêtre TEXT en HIRES
:CRTRA	$262	définition fenêre TRACE

VARIABLES CLAVIER

:BDCOL	$268	contient l'image des 8 colonnes après scrutation clavier
	$270	drapeau (0 si aucune touche pressée à la dernière scrutation)
	$271	contient le code de la dernière touche pressée
:BDVRR	$272	nombre pour répétition clavier
:BDVRL	$273	diviseur répétition
	$274	inutilisé

:LGKBD	$275	drapeau clavier b7=1 si majuscules b6=1 si son clavier b5=1 si ESC = CTRL T b2-b1 : 00 si clavier QWERTY 01 AZERTY 10 FRENCH b0=1 si gérer les touches de fonction
:BDFCT	$276	adresse de gestion des touches de fonction
:BDSHT	$278	contient la touche pressée en addition: b7=1 si CTRL pressé b6=1 si FUNCT b5=1 si joystick gauche b4 inutilisé b3=1 si souris b0=1 si SHIFT
:BDKEY	$279	valeur ASCII de la touche décodée
	$27A	4 octets inutilisés
:BDCTC	$27E	b7=1 si CTRL-C pressé

VARIABLE VIDEOTEX

/DTPIL	$282	4 octets de valeurs pour les séquences PROtocole

VARIABLES IMPRIMANTE

.PRX	$286	colonne actuelle sur imprimante
.PRY	$287	ligne actuelle (convention)
.PRFX	$288	largeur d'impression
.PRFY	$289	longueur d'une page (convention)
:LGLPR	$28A	drapeau imprimante: b7=1 si imprimante prête à traiter une donnée b6=1 si pas de CRLF en fin de ligne b2=1 si RS232 au lieu de CENTRONICS b1=1 si écho SCR -> LPR b0=1 si pas de LF après CR
PRSY	$28B	position devant déclencher un saut de page (convention)

VARIABLES JOYSTICK/SOURIS

:LGJCK	$28C	b6=1 si joystick ~~droit~~ gauche ~~(non géré)~~ b0=1 si souris / joystick droit
:CGVAL	$28D	valeur joystick : 000HBFGD (haut,bas,feu,gauche,droite) ↑ gauche

JCDVAL	$28E	valeur souris : HB FG GD FC FD (FG,FC,FD sont les trois feu)
	$28F	délai T2 de base (16 bits)
	$291	travail joystick
	$292	travail souris
	$293	travail joystick
	$294	travail souris
	$295	boutons souris
	$296	tampon boutons souris
	$297	répétition joystick
	$298	nombre avant répétition
	$299	répétition souris
	$29A	compteur avant répétition
	$29B	compteur bouton souris
	$29C	répétition bouton souris
TCKTAB	$29D	valeurs renvoyées par joystick et souris : défaut : 11,10,32,8,9,3,3 soit haut,bas,espace,gauche,droite et CTRL-C,CTRL-C
	$2A4	diviseur déplacement souris
	$2A5	compteur déplacement souris
	$2A6	compteurs clavier
	$2A7	
	$2A8	
	$2A9	inutilisé

VARIABLES TRAITEMENT HIRES

HRSPAT	$2AA	motif de tracé
HRSERR	$2AB	indication d'erreur dans les paramètres
	$2AC	inutilisé
	$2AD	inutilisé

VARIABLES ET TABLES DES E/S

IOTAB0	$2AE	contient les numéros des E/S ouvertes sur canal 0 (4 maxi)
IOTAB1	$2B2	id. canal 1
IOTAB2	$2B6	2
IOTAB3	$2BA	3
ADIOB	$2BE	ces 48 octets (2*24) permet de gérer 24 ressources d'entrée et de sortie. Les adresses contenues dans cette table sont celles des routines de gestion desdites E/S. Elles sont utilisées lors de demande d'ouverture/fermeture,lecture/ecriture.
FLGRST	$2EE	b7=1 si DEL+RESET lors de RESET

VECTEURS ET ADRESSES TELEMON

VNMI	$2F4	execution RESET simple (banque et adresse)
	$2F7	adresse pour execution routine d'E/S
VIRQ	$2FA	execution IRQ (JMP $406 par défaut)
VAPLIC	$2FD	execution APLIC (banque,adresse)

	$406	execution IRQ normale
	$409	gestion des buffers
EXBNK	$40C	exécution routine inter-banques
	$40F	buffer banques IRQ
	$410	buffer banques BRK
VEXBNK	$414	JMP xxxx vecteur pour EXBNK
BNKCIB	$417	numéro de la banque pour VEXBNK
	$418	pointeur inter-banques

VARIABLES DIVERSES

STRATSED ET EDITEUR

DRIVE	$500	numéro du lecteur (0-3) pour accès disque
BUFNOM	$517	drive (1),nom (9) et extension (3) à la suite
EXTDEF	$55D	extension par défaut (COM à la base)
BUFEDT	$590	buffer d'édition

VIDEOTEX

BVDTAS	$9000	buffer attributs VDT (1 Ko)
BVDTAT	$9400	buffer couleur VDT (1 Ko)
VDTDES	$9C00	travail et conversion G2
TABDBH	$9C80	buffer double hauteur VIDEOTEX

VECTEURS DU TELEMON

Sont appelés vecteurs du TELEMON les routines que l'on peut utiliser sans se soucier de la version du TELEMON sur laquelle tournera l'application. Quoique fort banale et courante (CF. MS-DOS, CP/M 80, etc...) l'utilisation des IRQ pour appeler les routines systèmes font de la vectorisation du TELEMON un must du genre.

En d'autre termes, appeler une routine du TELEMON se fera par BRK $xx et non JMP $xxyy. Ce qui fait gagner un octet, perdre un paquet de cycles machine et surtout rend linéaire l'espace mémoire (banques) du TELESTRAT.

Ainsi, avant d'appeler une routine du TELESTRAT par BRK $xx (adressage page zero), il faudra positionner registres et/ou variables mémoire comme indiqué dans la définition. Il est en outre inutile de se trouver sur la banque 7.

Les vecteurs sont rangés ici par ordre numérique, ce qui ne correspond pas toujours à quelque chose. Toutefois on distingue quelques groupes importants.

LA GESTION DES CANAUX

```
nom         : XOPx
numero(s)   : 00,01,02,03
adresse(s): C6E5,C6E8,C6EB,C6EE
entrée      : A
sortie      : X,C

action      : ouvre l'E/S A sur le canal x (0-3). En sortie, C=1 si l'E/S était
              déja ouverte. Z=1 si canal saturé et C=0,Z=0 si E/S ouverte.


nom         : XCLx
numero(s)   : 04,05,06,07
adresse(s): C720,C723,C726,C729
entrée      : A
sortie      : N

action      : ferme, si elle était ouverte, l'E/S A sur le canal considéré. Si A
              contient 0, ferme toutes les E/S sur le canal. En sortie Y=255 ,N=1.


nom         : XRDx
numero(s)   : 08,09,0A,0B
adresse(s): C7CF,C7D2,C7D5,C7D8
entrée      : rien
sortie      : A,C

action      : lit une donnée du canal x dans A. C=1 et A=0 si aucun code n'a été
              lu, C=0 et A=code si donnée lue. X et Y sont inchangés.
              A noter que les E/S étant scrutées dans leur ordre d'ouverture, elle
              sont prioritaires pour la lecture.
```

```
nom         : XRDWx
numero(s)   : 0C,0D,0E,0F
adresse(s): C806,C809,C80C,C80F
entrée      : rien
sortie      : A,C

action      : attend un code sur le canal x. En sortie, X et Y sont inchangés et
              C=1, A contient le code lu.


nom         : XWRx
numero(s)   : 10,11,12,13
adresse(s): C75D,C762,C767,C76C
entrée      : A
sortie      : rien

action      : écrit A sur le canal x. A,X et Y sont conservés.


nom         : XWSTRx
numero(s)   : 14,15,16,17
adresse(s): C7A8,C7AB,C7AE,C7B1
entrée      : A,Y
sortie      : A,Z

action      : envoie sur le canal x la chaine de codes située en AY et terminée
              par un 0. en sortie A=0 et Z=1.
```

ROUTINES SYSTEME DIVERSES

```
nom         : XDECAL
numero(s)   : 18
adresse(s): CD6C
entrée      : DECDEB,DECFIN,DECCIB
sortie      : C,N,Z

action      : décale un bloc mémoire de DECDEB à DECFIN vers DECCIB si possible.
              En sortie, A,X et Y sont intacts, Z et N selon A et C=0 si le
              décalage n'a pas eu lieu (C=1 si ok).


nom         : XTEXT
numero(s)   : 19
adresse(s): CF75
entrée      : rien
sortie      : P

action      : passe en mode TEXT. En sortie, P est représentatif de FLGTEL.
```

```
nom         : XHIRES
numero(s)   : 1A
adresse(s): CF45
entrée      : rien
sortie      : P

action      : passe en HIRES, P est selon FLGTEL. Si on est déja en mode HIRES,
              l'écran est simplement effacé.



nom         : XEFFHI
numero(s)   : 1B
adresse(s): CF06
entrée      : rien
sortie      : X,Z

action      : efface l'écran HIRES. En sortie, X=0 et Z=1.



nom         : XFILLM
numero(s)   : 1C
adresse(s): CF14
entrée      : RES,X,Y,A
sortie      : X,Z

action      : remplit la zone de RES à YX avec le code A. En sortie, X=0,Z=1.
              si on déborde au dela de $C000, comme pour toutes les modifs mémoire
              du TELEMON, il ne se passe rien.



nom         : ZADCHA
numero(s)   : 1D
adresse(s): FF31
entrée      : A
sortie      : RESB,C

action      : calcule dans RESB l'adresse de définition du code ASCII A. C=0.



nom         : XTSTLP
numero(s)   : 1E
adresse(s): C6BF
entrée      : rien
sortie      : Z

action      : teste dans FLGTEL si l'imprimante est détectée. en sortie,
              Z=0 si l'imprimante a été détectée.
```

nom : XMINMA
numero(s) : 1F
adresse(s): D0F0
entrée : A
sortie : A

action : si A est entre "a" et "z" , A passe entre "A" et "Z"...

CALCULS SYSTEMES

nom : XMUL40
numero(s) : 20
adresse(s): CE69
entrée : A
sortie : A,Y,RES

action : multiplie A par 40 dans AY et RES. P est selon Y.

nom : XMULT
numero(s) : 21
adresse(s): CE97
entrée : AY,RES
sortie : TR0-1-2-3,Z

action : multiplie AY par RES dans TR0-1-2-3, Z=1 en sortie.

nom : XADRES
numero(s) : 22
adresse(s): CE89
entrée : AY,RES
sortie : AY,RES,Z

action : additionne RES et AY dans RES et AY. Z selon A résultant.

nom : XDIVIS
numero(s) : 23
adresse(s): CEDE
entrée : RES,AY
sortie : RES,RESB,Z

action : divise (très rapidement !) RES par AY. Le résultat est dans RES, le reste dans RESB et Z=1,X=0.

DIVERS SYSTEME

nom : XNOMFI
numero(s) : 24
adresse(s): CFFD
entrée : X,AY

```
sortie     : BUFNOM,X,C

action     : place le nom de fichier de longueur X à l'adresse AY dans BUFNOM.
             En sortie, C=1 si le nom contenait des jokers (?,*). N=1 si le nom
             de fichier est invalide, le pourquoi est dans X :
             X=128 si un des codes est invalide
             X=129 si le drive spécifié dans le nom est incorrect
             X=130 si le nom est trop long
             X=131 si il y a plusieurs jokers dans le nom (*)
             X=132 si l'extension est invalide.

             Si N=0, opération bien déroulé, cependant :
             X=0 si le nom est de longueur nulle
             X=1 si le nom n'a qu'une lettre.



nom        : XCRLF
numero(s)  : 25
adresse(s): C756
entrée     : rien
sortie     : comme pour LDA #$0A/BRK XWRO

action     : saute une ligne sur le canal 0 (envoie CR,LF = 0/D 0/A ou 13,11)



nom        : XDECAY
numero(s)  : 26
adresse(s): E749
entrée     : AY
sortie     : AY,RESB,X

action     : convertit au maximum la chaine à l'adresse AY en un nombre dans AY
             et RESB. X contient le nombre de codes décodés.



nom        : XBINDX
numero(s)  : 28
adresse(s): CDEF
entrée     : AY,X,DEFAFF
sortie     : TR4-5-6

action     : convertit le nombre AY, de longueur X+2 en mettant les zeros de
             début de nombre (espace en général). Le nombre est convertit en
             ASCII à l'adresse (TR5-6). Sa longueur est dans TR4.



nom        : XDECIM
numero(s)  : 29
adresse(s): CE39
entrée     : comme XBINX
sortie     : Z,Y

action     : convertit et affiche le nombre AY... Z=1 et Y=longueur du nombre
             affiché en sortie.
```

nom : XHEXA
numero(s) : 2A
adresse(s): CE54
entrée : A
sortie : AY,C

action : convertit A en hexadécimal ASCII dans AY. C=0 en sortie.
aucun test de validité n'est fait.

nom : XA1AFF
numero(s) : 2B
adresse(s): F49B
entrée : ACC1
sortie : Z=1

action : affiche ACC1 en décimal sur le canal 0.

ROUTINES EDITEUR

nom : XMENU
numero(s) : 2C
adresse(s): CBE0
entrée : voir action
sortie : X,A

action : gère un menu dont les choix, séparés par 0, sont à l'adresse ADMEN.
le dernier choix est clos par deux 0.MENDDY et X contiennent le
numéro des lignes de bord de la fenêtre, MENDDX contient la colonne
où doit débuter la fenêtre. MENLX contient la largeur de la barre
en vidéo inverse, A contient le numéro du premier choix à afficher
et Y la ligne dans la fenêtre où il faut afficher la barre.
En sortie, X contient le numéro du choix (0 - n-1) et A :
13 si sortie RETURN
27 si ESC
32 si barre d'espace
(le CTRL-C n'est pas géré)

nom : XEDT
numero(s) : 2D
adresse(s): E435
entrée : A,X
sortie : A,RES,Y,X,BUFEDT

action : lit au clavier un ligne dans BUFEDT. en entrée A contient la
longueur maxi acceptée (<110) et X le nombre d'espaces à insérer
avant de positionne le curseur.
En sortie, la ligne est dans BUFEDT terminée par 0, le numéro de
ligne est dans RES, le nombre d'espaces bidons en début de ligne
dans Y et le début réel de la ligne (sans espaces ni numéro de
ligne) dans X. A contient le mode de sortie :
13 : RETURN
03 : CTRL-C

10 : flèche bas
11 : flèche haut

nom : XINSER
numero(s) : 2E
adresse(s): E6B0
entrée : voir action
sortie : voir action

action : insère dans le listing d'adresses extrêmes SCEDEB,SCEFIN la ligne numéro RES, de longueur A, contenu à l'adresse TR0-1.
Si A=0, on efface la ligne de numéro RES.
En sortie, SCEFIN est ajusté à la nouvelle fin du listing, SCEDEB contient l'adresse de la ligne et TR3-4 contient (signé) la variation de taille du listing.

nom : XSCELG
numero(s) : 2F
adresse(s): E680
entrée : SCEDEB,RES
sortie : C,RESB

action : recherche dans le listing à l'adresse SCEDEB la ligne de numéro RES. Si elle est trouvée, C=1 et adresse dans RESB. C=0 sinon.

ROUTINES VIDEO

nom : aucun
numero(s) : 30
adresse(s): D140
entrée : rien
sortie : rien

action : controle le déplacement du curseur VIDEOTEX.
Cette routine n'est pas utilisable seule et sa vectorisation semble bien étrange.

nom : aucun
numero(s) : 31
adresse(s): D711
entrée : rien
sortie : X,Z

action : affiche selon VDTX,VDTY,VDTGX et VDTGY un motif VIDEOTEX G2 (mosaique). En sortie, X=0 et Z=1.

nom : XEDTIN
numero(s) : 32
adresse(s): E537
entrée : A

```
sortie     : X

action     : envoie le code A dans BUFEDT. En sortie, X est conservé et Z,N sont
             positionnés selon X.



nom        : XECRPR
numero(s)  : 33
adresse(s): E66C
entrée     : rien
sortie     : comme après LDA #127 / BRK XWRO

action     : affiche un prompt sur la sortie vidéo courante (minitel et/ou écran



nom        : XCOSCR
numero(s)  : 34
adresse(s): DE1E
entrée     : rien
sortie     : rien

action     : affiche le curseur dans la fenêtre 0.



nom        : XCSSCR
numero(s)  : 35
adresse(s): DE20
entrée     : rien
sortie     : rien

action     : éteint le curseur dans la fenêtre 0.



nom        : XSCRSE
numero(s)  : 36
adresse(s): DEFB
entrée     : X,AY
sortie     : P

action     : crée la fenêtre X selon les adresses stockées en AY.
             dans l'ordre :
             colonne gauche (0-39) -> 1 octet
             colonne droite (0-39) -> 1 octet
             ligne haut     (0-27) -> 1 octet
             ligne bas      (0-27) -> 1 octet
             adresse de base       -> 2 octets ($BB80 en général).

             cette fonction très puissante permet de créer une fenêtre de
             n'importe quelle taille n'importe où en mémoire. Par exemple :
             0,80,0,255,$4000 crée une fenêtre de 80 colonnes,255 lignes qui
             débute en #4000...



nom        : XSCROH
numero(s)  : 37
```

adresse(s): DE54
entrée : X,Y
sortie : N=1

action : scrolle vers le BAS (!) la fenêtre courante (selon $28) de la ligne X à la ligne Y. Si X=Y, attendez-vous à des surprises !

nom : XSCROB
numero(s) : 38
adresse(s): DE5C
entrée : X,Y
sortie : N=1

action : scrolle vers le haut cette fois ci.

nom : XSCRNE
numero(s) : 39
adresse(s): FEF7
entrée : AY
sortie : Z=1

action : redéfinit des caractères selon la tables à l'adresse AY : code ascii,8 nombres de def.,code suivant,8 nbres,...,0

nom : aucun
numero(s) : 3A
adresse(s): D442
entrée : A
sortie : $37,A

action : gère une donnée videotex (code simple, début de séquence, poursuite de séquence,etc..). Reportez-vous à la routine pour la liste exacte des codes de sortie de la routine.

nom : aucun
numero(s) : 3B
adresse(s): D4F1
entrée : A
sortie : Y,X

action : code A dans le sens HIRES VDT -> VIDEOTEX. En sortie, Y,A,X contiennent : 0,code,0 si code simple
sequence,code1,0 si séquence simple
sequence,code1,code2 si séquence double

GESTION DE L'HORLOGE

nom : aucun
numero(s) : 3C

adresse(s): CA55
entrée : rien
sortie : A,X, N=0 Z=0

action : remet l'horloge à 0. A=1,X=255 et Y inchangé.

nom : XCLCL
numero(s) : 3D
adresse(s): CA65
entrée : rien
sortie : N=0

action : stoppe l'affichage de l'horloge. LSR $214 est 10 fois plus rapide (et oui !) mais perd un octet... au choix.

nom : XWRCLK
numero(s) : 3E
adresse(s): CA69
entrée : AY
sortie : rien

action : indique qu'il faut afficher l'horloge en AY. A,Y,X et P sont inchangés. A noter qu'on peut envoyer l'horloge n'importe ou en RAM

ROUTINES SONORES

nom : XSONPS
numéro(s) : 40
adresse(s): D9E9
entrée : X,Y
sortie : P

action : envoie les 14 données situées à l'adresse XY dans les registres R0-R13 du PSG AY3-8912. En sortie, P est inchangé.

nom : XEPSG
numéro(s) : 41
adresse(s): DA1A
entrée : A,X
sortie : A,Y,P

action : envoie X dans le registre A du PSG. En sortie, A,Y et P sont inchangés.

nom : XOUPS
numéro(s) : 42
adresse(s): DDD8
entrée : rien
sortie : Y,A

action : émet un OUPS. En sortie, Y=0 , A=7 et Z=0,N=0.

nom : XPLAY
numéro(s) : 43
adresse(s): EB0D
entrée : HRS1,HRS2,HRS3,HRS4
sortie : A,Y,P

action : ouvre les canaux musicaux (HRS1), bruits (HRS2) avec enveloppe HRS3 et periode HRS4*32 us.
Pour ouvrir un canal on met 1 dans le tercet b2b1b0 de poids faible du HRS demandé. par exemple HRS1=00000000 00000101 ouvre les canaux 1 et 3 en musical. En sortie, Y contient le numéro de l'enveloppe, A contient 13 et Z=0,N=0.

nom : XSOUND
numéro(s) : 44
adresse(s): EB73
entrée : HRS1,HRS2,HRS3
sortie : A,Y,P

action : envoie un signal de période HRS2*32 us sur le canal HRS1 selon le volume HRS3. Si HRS3 est à 0, le volume est géré par l'enveloppe demandée par le dernier XPLAY effectué.
En sortie, N=0,Z=0 et A et Y sont quelconques.

nom : XMUSIC
numéro(s) : 45
adresse(s): EB5A
entrée : HRS1,HRS2,HRS3,HRS4
sortie : A,Y,P

action : joue la note HRS2 à l'octave HRS3 sur le canal HRS1 avec le volume HRS4. Si HRS4=0 le volume est géré par l'enveloppe demandé dans le dernier PLAY. A,Y et P sont comme après un XSOUND.

nom : XZAP
numéro(s) : 46
adresse(s): EBEC
entrée : rien
sortie : A,Y,P

action : émet un ZAP. En sortie, A=8,Y=$EB et Z=1,N=0.

nom : XSHOOT
numéro(s) : 47
adresse(s): EBDF
entrée : rien
sortie : A,Y,P

action : émet un SHOOT. A inchangé, Y=$EB et Z=0,N=0.

GESTION IMPRIMANTE

nom : XLPRBI
numéro(s) : 48
adresse(s): DA72
entrée : A
sortie : C

action : envoie A sur l'imprimante (ou tout autre appareil branché sur la sortie parallèle CENTRONICS). En sortie C=0.

nom : XLPCRL
numéro(s) : 49
adresse(s): DAE4
entrée : rien
sortie : A,C

action : envoie un CR,LF (retour et saut de ligne) sur la sortie CENTRONICS. En sortie, C=0 et A est conservé.

nom : XHCSCR
numéro(s) : 4A
adresse(s): E1B9
entrée : $28
sortie : A,C

action : hard-copy de la fenêtre TEXT $28. En sortie, A=0 et C=1.

nom : XHCVDT
numéro(s) : 4B
adresse(s): E209
entrée : rien
sortie : rien

action : envoie sur imprimante le contenu ASCII de la page HIRES VIDEOTEX. En sortie, les registres sont come apres XLPCRL.

nom : XHCHRS
numéro(s) : 4C
adresse(s): E250
entrée : rien
sortie : X

action : envoie l'écran HIRES sur imprimante. X=0 en sortie.
la routine est buggée, donc inutilisable en interne.
Toutefois, on peut detourner le vecteur d'execution en plaçant en $250-1 l'adresse de sa propre routine de Hard-copy HIRES.

GESTION CLAVIER

nom : XALLKB
numéro(s) : 50
adresse(s): D903
entrée : rien

sortie : Y,Z,KBDCOL

action : scrute le clavier. En sortie, KBDCOL (8 octets) contient l'image des 8 colonnes du clavier. Z=0 si une touche a été pressée et Y le numéro de sa colonne.

nom : XKBDAS
numéro(s) : 51
adresse(s): D81F
entrée : Y,Z,KBDCOL
sortie : buffer clavier

action : selon Y,Z et KBDCOL positionnés en sortie de XALLKB sont envoyés dans le buffer clavier deux codes : le code ASCII de la touche et son code SHIFT.
%CFxxxxxS : C=1 si CTRL
F=1 si FUNCT
S=1 si SHIFT

nom : XGOKBD
numéro(s) : 52
adresse(s): FF4C
entrée : A
sortie : rien

action : change le type de clavier selon la valeur de A :
00 QWERTY
01 AZERTY
02 FRENCH
03 rien
04 ACCENT SET
05 ACCENT OFF

GESTION DES BUFFERS

nom : XECRBU
numéro(s) : 54
adresse(s): C51D
entrée : X,A,Y
sortie : C

action : écrit la donnée A ou AY dans le buffer X. C=1 si buffer plein.

nom : XLISBU
numéro(s) : 55
adresse(s): C518
entrée : X
sortie : A,Y,C,V

action : Lit dans A ou AY une donnée du buffer X. C=1 si buffer vide.

nom : XTSTBU
numéro(s) : 56
adresse(s): C50F

entrée : X
sortie : C

action : met C à 1 si le buffer X est vide.

nom : XVIDBU
numéro(s) : 57
adresse(s): C50C
entrée : X
sortie : rien

action : vide le buffer X

nom : XINIBU
numéro(s) : 58
adresse(s): C507
entrée : RES,AY,X
sortie : A,Y,Z

action : initialise le buffer X de RES à AY exclut. En sortie, A=0,Y=255 et Z=1.

nom : XDEFBU
numéro(s) : 59
adresse(s): C4EA
entrée : X
sortie : comme XINIBU

action : rend au buffer X ses valeurs par défaut.

nom : XBUSY
numéro(s) : 5A
adresse(s): CFB1
entrée : rien
sortie : C

action : teste si un buffer au moins est non vide. S'ils sont tous vides, C=1.

GESTION DE LA SORTIE SERIE

nom : XSDUMP
numéro(s) : 5C
adresse(s): ED9A
entrée : rien
sortie : rien

action : macro-commande équivalente au BASIC SDUMP.
Tous les codes lus en entrée sur la RS232-C sont envoyés à l'écran, les codes de controle sont en hexa rouge, les autres en ASCII.
CTRL-C pour finir.

```
nom         : XCONSO
numéro(s)   : 5D
adresse(s): ED77
entrée      : rien
sortie      : rien

action      : transforme le TELESTRAT en terminal série. Les codes tapés au
              clavier sont envoyés sur la RS232 et les codes venant de la RS
              sont envoyés à l'écran comme SDUMP.


nom         : XSLOAD
numéro(s)   : 5E
adresse(s): EDE5
entrée      : C
sortie      : DESALO,FISALO,EXSALO

action      : charge un fichier via la RS232. Si C=1, on charge sans entête à
              l'adresse DESALO et on coupe par CTRL-C. En sortie, DESALO,FISALO,
              EXSALO contiennent les adresse debut,fin,execution du fichier.


nom         : XSSAVE
numéro(s)   : 5F
adresse(s): EDCA
entrée      : DESALO,FISALO,EXSALO,C
sortie      : rien

action      : sauve un fichier, debut,fin,execution dans DESALO,FISALO,EXSALO, via
              la RS232. Si on sauve sans entête, C=1.


nom         : XMLOAD
numéro(s)   : 60
adresse(s): EDFC
entrée      : comme XSLOAD
sortie      : comme XSLOAD

action      : comme XSLOAD mais les fichiers sont lus via le Minitel.


nom         : XMSAVE
numéro(s)   : 61
adresse(s): EDD7
entrée      : comme XSSAVE
sortie      : comme XSSAVE

action      : comme XSSAVE mais le fichier est envoyé au Minitel.


nom         : XRING
numéro(s)   : 62
adresse(s): EEA5
entrée      : rien
sortie      : rien

action      : attend un appel téléphonique.
```

```
nom          : XWCXFI
numéro(s)    : 63
adresse(s)   : EF4A
entrée       : rien
sortie       : C

action       : attend que les codes CONNEXION/FIN Teletel soient reçus.
               Si la touche CX/FIN est pressée dans les 25 secondes, C=0.


nom          : XLIGNE
numéro(s)    : 64
adresse(s)   : EF20
entrée       : rien
sortie       : rien

action       : prend la ligne. Envoie les séquences PRO1/OPPO (retournement) et
               PRO1/CONNEXION (prise de ligne).


nom          : XDECON
numéro(s)    : 65
adresse(s)   : EF3F
entrée       : rien
sortie       : rien

action       : coupe la ligne.


nom          : XMOUT
numéro(s)    : 66
adresse(s)   : EF7A
entrée       : A
sortie       : rien

action       : envoie A au Minitel.


nom          : SOUT
numéro(s)    : 67
adresse(s)   : EF85
entrée       : A
sortie       : rien
action       : envoie A sur la RS232-C.
```

TRAVAIL EN VIRGULE FLOTTANTE

LES OPERATEURS

```
nom          : XA1DEC
numéro(s)    : 68
adresse(s)   : F4A5
entrée       : rien
sortie       : AY,BUFTRV
```

```
action    : convertit ACC1 dans BUFTRV ($0100). En sortie, AY contient BUFTRV.


nom       : XDECA1
numéro(s) : 69
adresse(s): F91E
entrée    : AY
sortie    : ACC1

action    : convertit dans ACC1 le nombre en ASCII en AY.
            La forme binaire est décodée très rapidement, écrivez vos
            nombre en binaire le plus souvent possible.


nom       : XA1PA2
numéro(s) : 6A
adresse(s): EFB2
entrée    : ACC1,ACC2
sortie    : FLERR,ACC1

action    : ACC1+ACC2 -> ACC1. Si overflow, FLERR=1.


nom       : XA2NA1
numéro(s) : 6B
adresse(s): EF9B
entrée    : ACC1,ACC2
sortie    : FLERR,ACC1

action    : ACC2-ACC1 -> ACC1


nom       : XA1MA2
numéro(s) : 6C
adresse(s): F18B
entrée    : ACC1,ACC2,Z
sortie    : ACC1,FLERR

action    : ACC1*ACC2 -> ACC1. Z doit refléter ACC1E.


nom       : XA2DA1
numéro(s) : 6D
adresse(s): F28A
entrée    : ACC1,ACC2,Z
sortie    : ACC1,FLERR

action    : ACC2/ACC1 -> ACC1. Z selon ACC1E en entrée. FLERR=3 si division par
            zéro.


nom       : XA2EA1
numéro(s) : 6E
adresse(s): F61A
entrée    : ACC1,ACC2,Z
sortie    : ACC1,FLERR

action    : calcule ACC2^ACC1 dans ACC1. Z selon ACC1E en entrée.
```

```
nom         : XNA1
numéro(s)   : 6F
adresse(s): F653
entrée      : ACC1
sortie      : ACC1

action      : calcule -ACC1 dans ACC1.
```

LES FONCTIONS

```
nom         : XSIN
numéro(s)   : 70
adresse(s): F78E
entrée      : ACC1
sortie      : ACC1

action      : calcule dans ACC1 le sinus de ACC1


nom         : XCOS
numéro(s)   : 71
adresse(s): F781
entrée      : ACC1
sortie      : ACC1

action      : calcule dans ACC1 le cosinus (sinus X+PI/2) de ACC1. Le cosinus
              est plus long à calculer que le sinus car il nécessite une somme.


nom         : XTAN
numéro(s)   : 72
adresse(s): F80A
entrée      : ACC1
sortie      : ACC1,FLERR

action      : calcule TAN(ACC1) par sinus et cosinus... très long.
              FLERR=3 si ACC1=PI/2 modulo PI.


nom         : XATN
numéro(s)   : 73
adresse(s): F835
entrée      : ACC1
sortie      : ACC1

action      : calcule l'arctangente de ACC1.


nom         : XEXP
numéro(s)   : 74
adresse(s): F68C
entrée      : ACC1
sortie      : ACC1,FLERR

action      : calcule l'exponentielle e de ACC1. FLERR=1 si overflow.
```

```
nom         : XLN
numéro(s)   : 75
adresse(s): F146
entrée      : ACC1
sortie      : ACC1,FLERR

action      : calcul le logarithme e de ACC1. FLERR=2 si ACC1 est négatif ou nul.


nom         : XLOG
numéro(s)   : 76
adresse(s): F26F
entrée      : comme XLN
sortie      : comme XLN

action      : comme XLN, mais calcule le Logarithme décimal


nom         : XRND
numéro(s)   : 77
adresse(s): F735
entrée      : ACC1
sortie      : ACC1

action      : si ACC1 positif, lit un nombre aléatoire.
              si ACC1 négatif, ABS(ACC1) devient la nouvelle racine du générateur
              aléatoire en ($2EF).


nom         : XSQR
numéro(s)   : 78
adresse(s): F610
entrée      : ACC1
sortie      : ACC1

action      : calcule la racine carrée de ACC1


nom         : XRAD
numéro(s)   : 79
adresse(s): F8B6
entrée      : ACC1
sortie      : ACC1

action      : convertit ACC1 de degrés en radians.


nom         : XDEG
numéro(s)   : 7A
adresse(s): F8AA
entrée      : ACC1
sortie      : ACC1


action      : convertit ACC1 de radians en degrés.
```

```
nom         : XINT
numéro(s)   : 7B
adresse(s): F46A
entrée      : ACC1
sortie      : ACC1

action      : calcule la partie entière de ACC1.


nom         : XPI
numéro(s)   : 7C
adresse(s): F314 (il fallait oser !)
entrée      : rien
sortie      : ACC1

action      : ACC1=PI, soit 3,14159265


nom         : XRAND
numéro(s)   : 7D
adresse(s): F771
entrée      : ACC1
sortie      : ACC1

action      : retourne un entier aléatoire entre 0 et INT(ACC1).
```

TRANSFERTS FLOTTANTS <-> MEMOIRE

```
nom         : XA1A2
numéro(s)   : 7E
adresse(s): F387
entrée      : ACC1
sortie      : ACC2,X

action      : ACC1 -> ACC2 et X=0.


nom         : XA2A1
numéro(s)   : 7F
adresse(s): F377
entrée      : ACC2
sortie      : ACC1,X

action      : ACC2 -> ACC1 et X=0.


nom         : XIYAA1
numéro(s)   : 80
adresse(s): F3ED
entrée      : YA
sortie      : ACC1

action      : transfère YA (attention ! pas AY) dans ACC1.
              le résultat est un entier.
```

```
nom         : XAYA1
numéro(s)   : 81
adresse(s): F323
entrée      : AY
sortie      : ACC1

action      : transfère dans ACC1 le flottant en (AY), le résultat est un réel.


nom         : XA1IAY
numéro(s)   : 82
adresse(s): F3A6
entrée      : ACC1
sortie      : AY,FLERR

action      : la valeur entière de ACC1 est stockée dans AY.
              FLERR=10 si ACC1 est > 65535


nom         : XA1XY
numéro(s)   : 83
adresse(s): F352
entrée      : XY
sortie      : Y=0

action      : transfère ACC1 en (XY), le résultat est un réel.


nom         : XAA1
numéro(s)   : 84
adresse(s): F396
entrée      : ACC1
sortie      : ACC1

action      : arrondit ACC1 selon ACC1EX.


nom         : XADNXT
numéro(s)   : 85
adresse(s): F8CD
entrée      : AY,ACC1
sortie      : ACC1,FLERR

action      : (AY)+ACC1 -> ACC1, FLERR=1 si overflow.


nom         : XINTEG
numéro(s)   : 86
adresse(s): FA12
entrée      : ACC1
sortie      : A

action      : teste si ACC1 a une partie décimale nulle. A=1 si oui.
```

FONCTIONS HIRES

```
nom         : aucun
numéro(s)   : 88
adresse(s): E7E7
entrée      : rien
sortie      : HRSX40,HRSX6

action      : ajuste HRSX40,HRSX6 pour un décalage du curseur HIRES vers la gauch
              à utiliser avec parcimonie...


nom         : aucun
numéro(s)   : 89
adresse(s): E7D9
entrée      : rien
sortie      : HRSX40,HRSX6

action      : id. mais vers la droite


nom         : aucun
numéro(s)   : 8A
adresse(s): E7C1
entrée      : rien
sortie      : HRSX40,HRSX6

action      : id. mais vers le bas


nom         : aucun
numéro(s)   : 8B
adresse(s): E7CD
entrée      : rien
sortie      : HRSX40,HRSX6

action      : id. mais vers le haut


nom         : XHRSSE
numéro(s)   : 8C
adresse(s): E792
entrée      : rien
sortie      : rien

action      : affiche le curseur selon HRSFB à sa position courante.


nom         : XDRAWA
numéro(s)   : 8D
adresse(s): E866
entrée      : HRSx
sortie      : rien

action      : trace un trait de HRS1,HRS2 à HRS3,HRS4 selon HRSFB.
              les coordonnées sont sur deux octets signés.


nom         : XDRAWR
numéro(s)   : 8E
adresse(s): E885
entrée      : HRSx
```

```
sortie     : Z

action     : trace un trait de la position du curseur, d'u déplacement horizontal
             HRS1 et vertical HRS2, selon HRSFB.


nom        : XCIRCL
numéro(s)  : 8F
adresse(s): E9CB
entrée     : HRS1
sortie     : rien

action     : trace un cercle de rayon HRS1 avec pour centre la position courante
             du curseur. Un rayon de 0 plante la machine...


nom        : XCURSE
numéro(s)  : 90
adresse(s): E92F
entrée     : HRSx
sortie     : HRSERR

action     : place le curseur en HRS1,HRSà si possible. Sinon, HRSERR > 0.


nom        : XCURMO
numéro(s)  : 91
adresse(s): E93C
entrée     : HRSx
sortie     : HRSERR

action     : déplace le curseur de HRS1 pixels horizontaux et HRS2 verticaux.
             HRSERR>0 si le curseur sort de l'écran.


nom        : XPAPER
numéro(s)  : 92
adresse(s): E95D
entrée     : A,X
sortie     : A,X

action     : remplit la colonne 0 du code A dans la fenêtre X (ou 128 si HIRES).
             En sortie, A inchangé et X=0.


nom        : XINK
numéro(s)  : 93
adresse(s): E95F
entrée     : A,X
sortie     : A,X

action     : remplit la colonne 1, comme pour PAPER


nom        : XBOX
numéro(s)  : 94
adresse(s): E819
entrée     : HRSx
sortie     : HRSERR,X=255
```

action : trace, à partir du curseur, un rectangle de longueur HRS1 et de largeur HRS2 (ou l'inverse... on va pas chipoter.).

nom : XABOX
numéro(s) : 95
adresse(s): E82C
entrée : HRSx
sortie : HRSERR,X=255

action : trace un rectangle de HRS1,HRS2 à HRS3,HRS4.

nom : XFILL
numéro(s) : 96
adresse(s): EA73
entrée : HRSx
sortie : Z

action : écrit HRS1 lignes de HRS2 codes HRS3 à partir du curseur. Z=1 en sortie.

nom : XCHAR
numéro(s) : 97
adresse(s): EAAF
entrée : HRS1,HRS2
sortie : rien

action : affiche le code ASCII HRS1 de la table HRS2 sur l'écran HIRES. Le curseur est déplacé de 8 pixels vers la droite.

nom : XSCHAR
numéro(s) : 98
adresse(s): EA93
entrée : A,Y,X
sortie : rien

action : affiche la chaine en (AY) de longeur X à l'écran HIRES. Le FB ne peut être modifié.

nom : XEXPLO
numéro(s) : 9C
adresse(s): EBE5
entrée : rien
sortie : X,Y

action : émet un EXPLODE. En sortie, XY pointe toujours sur la table EXPLODE de telle sorte que pour émettre de nouveau un EXPLODE, il suffit d'appeler XSONPS.

nom : XSHOOT
numéro(s) : 9D
adresse(s): F78E
entrée : rien
sortie : X,Y

action : émet un SHOOT, voir XEXPLO..

POST-SCRIPTUM

Si vous désirez me faire part de votre avis, ou commander d'autres exemplaires de cet ouvrage, écrivez à l'adresse ci-dessous, en prenant soin de joindre un enveloppe timbrée si une réponse manuscrite est attendue :

Guillaume MEISTER
6, Rue Nicolas François GILLET
57050 METZ