ATMOS F83 V 2.0

MINI-MANUEL UTILISATEUR

Michel ZUPAN Février 87

AVERTISSEMENT

F83 est le standard le plus élaboré du langage FORTH développé à Berkeley par Mike PERRY et Henry LAXEN selon les normes Forth-83 de la communauté internationale Forth.

La présente version pour Oric-Atmos s'est voulue aussi proche que possible de celles disponibles sous CP/M ou MS-DOS avec un grand nombre de particularités qui en font un outil très supérieur aux noyaux passe-partout du " domaine public " proposés pour des machines plus performantes sur le plan hardware.

Plus qu'un langage, le FORTH est un concept qui permet aux programmeurs chevronnés de développer des applications de très haut niveau, compactes, rapides, modulaires, portables mieux que tout autre langage évolué. Un grand nombre de logiciels professionnels des plus utilisés ont été conçus en FORTH.

Toutefois le Forth présente comme tout langage le défaut de ses qualités. Etant un langage très proche système, son apprentissage et sa maîtrise complète représentent un investissement intellectuel important. Ainsi s'explique sa relative faible diffusion face à des langages plus faciles comme le BASIC, le PASCAL, le C ou le LISP.

Pour bien faire, notre Atmos-F83 ne devrait pas être livré sans un manuel complet détaillant l'organisation matérielle et logicielle de la machine, la programmation du 6502 et de ses périphériques, les concepts fondamentaux du Forth, son implantation, le glossaire exhaustif et commenté de tous les mots de tous les vocabulaires.

Un tel manuel dépasserait les 1000 pages ce qui est hors de proportion avec la modestie de l' ATMOS (et de mes moyens !).

Nous vous recommandons vivement la lecture des ouvrages traitant du Forth, de l'Atmos ou du 6502 en général.

Les programmeurs déjà habitués au FIG-Forth risquent d'être momentanément décontenancés par les nouveautés du F83. Un manuel de référence spécifique est indispensable. L'auteur du F83 pour Atmos a participé à l'élaboration du seul manuel en Français du F83.

Il reste que c'est au clavier de votre Atmos-F83 que vous ferez le plus de découvertes. Explorez, cherchez, testez, comprenez : le F83 dispose de tous les outils pour vous aider. Et si vous rencontrez quelques difficultés , n'hésitez pas à me contacter : les oric-forthiens forment une petite famille...

Courage donc, et que le FORTH soit avec vous !

Michel ZUPAN

FAISONS CONNAISSANCE

La présente version F83 est exclusivement configurée pour ATMOS avec lecteur DISCORIC exploité par SEDORIC.
Les matériels ORIC-1, TELESTRAT, drive JASMIN, sont incompatibles ainsi que les autres systèmes d'exploitation.

MISE EN ROUTE

Placez la disquette F83 dans le drive.
Bootez le contrôleur.
Vous êtes désormais sous F83.

Le système affiche un message comportant la version du logiciel, sa date d'enregistrement (vous pourrez les modifier) ainsi que le nom du fichier ouvert par défaut. Ce fichier est le premier fichier .FTH que le Forth a pu trouver sur le disque.

Le système est en attente d'une commande. Tapez plusieurs fois la touche RETURN. le message OK signifie que le Forth a fini son travail sans erreur (pour l'instant aucun !) et qu'il attend une nouvelle commande.

PLONGEONS DANS LE DICTIONNAIRE

L'unité lexicale du Forth est le mot. Les mots sont compilés dans le dictionnaire. Ils sont regroupés par vocabulaires indépendants pour une meilleure exploitation du dictionnaire.

Tapons ORDER (suivi de <return>, nous ne le répéterons plus) :
Nous lisons Context: FORTH ONLY
Current: FORTH

FORTH est le vocabulaire de base du langage. ONLY est un vocabulaire minimal servant surtout à gérer les autres vocabulaires.

CONTEXT est une suite de vocabulaires dits de contexte c.a.d de recherche des mots dans le dictionnaire. Si vous entrez un mot par exemple TOTO, il sera actuellement cherché dans FORTH puis dans ONLY et comme il ne s'y trouve pas vous aurez droit à un petit message d'erreur.

CURRENT indique le vocabulaire courant, celui où seront placés les nouveaux mots que vous allez définir.

Il existe d'autres vocabulaires dans votre Forth. Tapez VOCS . Vous avez la liste des vocabulaires qui composent actuellement le dictionnaire.

Tapons maintenant WORDS

Quelle liste ! Nous voyons défiler tous les mots du premier vocabulaire de contexte, en l'occurence FORTH soi-même.

Tous les mots de listages (WORDS LIST INDEX CAT DIR DESAS SEE etc) ont leur action suspendue par l'appui d'une touche (utilisez la barre d'espace) puis reprise par un nouvel appui sauf si le deuxième appui est <return> qui lui stoppe définitivement le listage. Essayez encore avec WORDS.

Entrons ALSO EDITOR
et voyons comment nous avons modifié le contexte : ORDER
Si nous exécutons à nouveau WORDS , nous listons les mots de EDITOR , le vocabulaire de l'éditeur Forth.

Mais revenons plutôt au Forth : ONLY FORTH

PREMIERS CALCULS

Entrons 2 3 + .
en veillant à laisser au moins un espace entre chaque mot forth : ce sont les séparateurs naturels des mots.

Nous lisons 5 , résultat de 2+3 : Forth sait calculer !

Voici ce que nous avons commandé :
1) nous avons placé 2 au sommet de la "pile"
2) nous avons empilé encore 3
3) nous avons exécuté l'addition par + entre les deux nombres avec résultat sur la pile.
4) le point . a effectué la sortie du sommet de la pile vers l'affichage à l'écran.

Nous venons de saisir deux principes fondamentaux du Forth : les mots agissent sur une pile de paramètres et les calculs s'effectuent sur cette pile selon la logique polonaise inverse (comme sur les calculatrices Hewlett-Packard)

OBSERVONS LA PILE

Tapons plusieurs nombres par exemple 4 9 -3 157
puis .S plusieurs fois : .S affiche la pile sans la modifier et il va nous servir à tester l'action des manipulateurs de la pile que sont des mots comme DUP DROP SWAP OVER ROT -ROT NIP TUCK 2DROP etc...
De même observons avec quelques nombres sur la pile, au besoin en en ajoutant ce que font des opérateurs tels que + - * / MOD /MOD 1+ 2* etc...

En Forth, tous les mots (et il y en a !) travaillent avec une pile.

CREONS UN MOT

Ecrivons la ligne suivante :

```
: POLICE 127 32 DO I EMIT LOOP ;
```

Entre : et ; nous venons de définir le nouveau mot POLICE qui affiche à l'écran la police des caractères ASCII disponibles. Nous avons utilisé une structure DO...LOOP de boucle indicée [32,127[pour envoyer (EMIT) chaque caractère ASCII de l'indice I.

Exécutons POLICE pour vérifier son bon fonctionnement (admirons au passage la rapidité du Forth). Exécutons WORDS : le mot POLICE fait partie de notre dictionnaire ; c'est ça la magie du Forth.

Essayez
```
ACCENT  ( caractères accentués )
POLICE
QWERTY  ( clavier anglais )
POLICE
```

VISITE DANS LES ECRANS

Exécutez les mots qui suivent :

.FILE : le système affiche le fichier courant ouvert.

CAT : nous voyons les premières lignes de chaque écran de ce fichier.

1 LIST : nous listons le premier écran du fichier courant.

Tiens, nous retrouvons la définition de notre mot POLICE qui a déjà été compilé.

EDITONS

Ecrivons 1 EDIT : nous allons éditer l'écran 1.
Avec les flèches déplaçons-nous sur la ligne définissant POLICE et détruisons-la en faisant un <CTRL-X>.
Intercalons une ligne avec un <CTRL-Y> et écrivons dans cette ligne un message à afficher :

```
.( CECI EST UN ESSAI DE L'EDITEUR PLEIN-ECRAN DU FORTH )
```

Notez le scrolling dans un écran qui fait 16 lignes de 64 caractères

COMPILONS

Sortons de l'éditeur par la touche <ESC> et compilons notre écran : 1 LOAD
Un programme peut être écrit au clavier ou exécuté à partir des écrans de mémoire du Forth.

Nous venons de faire un petit tour rapide dans les principales activités de notre FORTH.

CONTENU DE LA DISQUETTE V2.0

F83KER.COM
noyau du F83 sans éditeur ni tools ni touches de fonctions.

F83DEV.COM
vocabulaire FORTH + touches de fonctions + EDITOR + SCREENS + TOOLS (désassembleur et décompilateur) : c'est un ensemble pratique pour le développement. Ce fichier COM est celui en autostart au booting de la disquette.

EDITOR.FTH
vocabulaire EDITOR de l'éditeur de ligne.

SCREENS.FTH
vocabulaire SCREENS d'édition plein-écran avec application à EDITOR.

FUNCT.FTH
installation des touches de fonctions.

DESAS.FTH
désassembleur 6502.

SEE.FTH
décompilateur forth.

ASM.FTH
macro-assembleur 6502

HRS.FTH
extensions de fonctions graphiques et sonores propres à l'ORIC. Ce fichier d'extensions est encore à developper.

PRINTER.FTH
extension au vocabulaire PRINTER.

DEBUG.FTH
débogueur (mode pas-à-pas) des mots forth. Un écran d'aide avec le mode d'emploi est affiché en fin de compilation.

WORK.FTH
votre premier fichier de travail...

F83. TOUCHES DE CONTROLES

Le F83 admet en standard l'exécution directe de commandes de contrôles au clavier y compris en cours de saisie.

Ces commandes sont obtenues par la touche CTRL (plus les flèches doublant les CTRL-H à K) et sont vectorisées dans une table pointée par la variable CTRL.

Dans la table CTRL1 se trouvent les commandes contrôles de base :

- gestion de l'écran, du curseur, du clavier, imprimante
- validations ou annulations des entrées.
- passages décimal/hexadécimal
- contrôles des tampons pour changements de fichiers ou disque.

Il est possible selon les besoins de modifier ou d'ajouter des commandes. Soit par exemple le mot :

```
: PING 7 EMIT ;
```

il suffit de faire

```
' PING CTRL1 CONTROL G 2* + !
```

pour que CTRL-G exécute chaque fois PING .

Voir en annexe la liste de CTRL1

Il existe une autre table totalement différente pour les contrôles de l'éditeur plein-écran SCREENS.

Notez que par rapport au F83 CP/M ou MS-DOS les tables sont nommées CTRL au lieu de CC pour éviter la confusion avec le nombre hexa CC, que les commandes disponibles sont beaucoup plus nombreuses et que la vectorisation est directe sans obligation d'un DROP sur la valeur ASCII de la touche de contrôle.

LISTE DES COMMANDES DE CONTROLE

```
CTRL-A  :  A-IN     saisie du caractère sous curseur
CTRL-B  :  HOME     curseur début d'écran
CTRL-C  :  C-IN     warm boot et DEFAULT pour changement disque
CTRL-D  :  DECIMAL  base décimale ( 'D' affiché dans status )
CTRL-E  :  HEX      base hexadécimale ( 'H' affiché dans status )
CTRL-F  :  KCLK     bascule clic clavier
CTRL-G  :  NOOP     ( libre )
CTRL-H  :  BS-IN    ou flèche gauche
CTRL-I  :  FW-IN    ou flèche droite
CTRL-J  :  DOWN-IN  ou flèche bas
CTRL-K  :  UP-IN    ou flèche haut = mouvements curseur
CTRL-L  :  CLS      effacement de l'écran
CTRL-M  :  CR-IN    ou touche return de fin de saisie
CTRL-N  :  CLL-IN   effacement de la ligne
CTRL-O  :  NOOP     ( mais inhibition clavier prioritaire sur ORIC )
CTRL-P  :  P-IN     bascule écho imprimante ( 'P' dans ligne status )
CTRL-Q  :  CURSOR   bascule curseur visible
CTRL-R  :  NOOP
CTRL-S  :  SAVE-BUFFERS  sauve les blocs immédiatement
CTRL-T  :  CAP-IN   bascule minuscules-majuscules clavier
CTRL-U  :  NOOP
CTRL-V  :  EMPTY-BUFFERS vide les tampons immédiatement
CTRL-W  :  NOOP
CTRL-X  :  X-IN     reprise de saisie depuis le début
CTRL-Y  :  NOOP
CTRL-Z  :  NOOP
CTRL-[  :  NOOP   ( ou touche escape )
CTRL-\  :  NOOP
CTRL-]  :  NOOP
```

TOUCHES DE FONCTION

Les touches de fonction ne font pas partie du standard du F83. Il eut été dommage de ne pas utiliser celles de l' ATMOS. L'usage proposé n'est qu'indicatif et peut être facilement rempllacé pour d'autres applications spécifiques.

A une touche de fonction peut être assigné un mot du dictionnaire ou une chaine quelconque de caractères.

L'appui de la touche de fonction (FUNCT+touche) est alors l'équivalent d'une saisie rapide du mot ou de la chaine assignée.

L'appui simultané de SHIFT-droite et de la touche de fonction annule la saisie en cours et saisit le mot ou la chaîne avec validation immédiate (comme suivi d'un return).

Notez les différences avec les touches de contrôles qui ne passent pas par la saisie d'un mot ou d'une chaîne.

Le codage des touches de fonction se réfère à la matrice du clavier ORIC. Voir par exemple le manuel du SEDORIC.

Les assignations de touches de fonction se font par ASSIGN (code--) pour un mot et ASSIGN" (code--) pour une chaîne. Exemple :

HEX BA ASSIGN CAT assigne à FUNCT-C le mot CAT (catalogue)

HEX 98 ASSIGN" CR VERSION TYPE" assigne à FUNCT-V la chaine permettant d'afficher la version du logiciel.

Les chaines assignées (pas les mots seuls) sont compilées dans le dictionnaire sans en-tête : attention donc à FORGET qui peut obliger à les ré-assigner si on agit en deça de l'assignation.

La liste des assignations proposées à l'initialisation est volontairement limitée pour vous permettre de vous faire le clavier que vous désirez :

```
FUNCT-1  :  ONLY FORTH DEFINITIONS DECIMAL
FUNCT-2  :  ORDER .S
FUNCT-3  :  CR .FILE SCR ?
FUNCT-4  :  HERE H.
FUNCT-A  :  ALSO
FUNCT-D  :  DEFINITIONS
FUNCT-E  :  EMPTY-BUFFERS
FUNCT-F  :  FORTH
FUNCT-L  :  LIST
FUNCT-O  :  ONLY
FUNCT-P  :  PREVIOUS
FUNCT-S  :  SAVE-BUFFERS
FUNCT-T  :  TYPE
FUNCT-V  :  VOCABULARY
FUNCT-W  :  WORDS
FUNCT-=  :  .CTRL .FUNCT
```

L' EDITEUR DU F83

Le Forth n'a jamais imposé un standard d'éditeur : les habitudes des programmeurs et les caractéristiques de chaque machine sont trop diverses.

Tout Forth possède un éditeur, le plus souvent un éditeur de lignes. Le F83 estime souhaitable l'incorporation d'un éditeur plein-écran et propose dans les versions du domaine public un éditeur de ligne de type " Starting Forth " ainsi que les primitives d'un éditeur plein-écran portable à divers terminaux.

Notre éditeur est lui aussi double : d'une part un éditeur de ligne pour lequel par commodité d'habitudes nous avons gardé la syntaxe du WFR-79 (décrit dans le livre " le Concept Forth " de P.COURTOIS - EDITEST). D'autre part un éditeur plein-écran dédié ORIC.

SCREENS est un vocabulaire d'édition plein-écran visualisant sur la fenêtre de l'écran Oric des textes de n'importe quelle dimension. Il a été conçu en premier lieu pour un traitement de texte 80 colonnes ou plus.

SCREENS peut être utilisé par EDITOR en configuration habituelle 16 lignes de 64 colonnes ou encore en 32 lignes de 32 colonnes par écran Forth.

Les mots SCREENS-ON et SCREENS-OFF sélectionne l'éditeur avec ou sans SCREENS.

Les deux éditeurs sont très complémentaires et peuvent mixer leurs fonctions. C'est vrai notamment des fonctions de recherche de mots ou de transferts de lignes de EDITOR qui restent accessibles en SCREENS-ON.

Exemple : si vous êtes en édition plein-écran et que vous faites <ESC> pour sortir de SCREENS puis
F TRUC
vous retournez aussitôt en SCREENS avec le curseur positionné au mot ' TRUC ' .

1) les fonctions générales d'édition

n EDIT passe en édition de l'écran n selon le mode actif
ED reprend l'édition de l'écran courant
NS ED passe à l'édition de l'écran suivant
PS ED passe à l'écran précédent
DONE quitte EDITOR et sauve au besoin avec mise à jour

2) Les commandes EDITOR en ligne

n CLEAR	efface l'écran n
n1 n2 CLEAN	efface les écrans n1 à n2
n NEW	attend l'entrée de nouvelles lignes depuis n
n UNDER	attend l'entrée de lignes intercalées sous n
L	liste l'écran courant
n P <texte>	place un nouveau texte dans ligne n
TOP	curseur en début d'écran
n T	affiche ligne n , curseur en début de ligne
n M	déplace le curseur de n cases (positif ou négatif)
C <texte>	inserre texte à la position du curseur
F <texte>	cherche texte, curseur en fin de texte
B	curseur en début de texte trouvé
N	cherche nouvelle occurence du texte dans PAD
X <texte>	supprime texte dans la ligne courante
TILL <texte>	supprime depuis le curseur jusqu'à texte compris
n H	copie une ligne dans le PAD
n D	détruit la ligne n après l'avoir copiée dans PAD
n R	place la ligne dans le PAD à la ligne n
n I	inserre la ligne du PAD à la ligne n
n E	efface (espaces) la ligne n
n S	décale la ligne n et les suivantes vers le bas
NS	passe à l'écran suivant
PS	passe à l'écran précédent (le PAD n'est pas modifié : transferts posibles)
FRESH	restitue l'écran courant depuis le disque (sert à annuler une session d'édition)
n1 n2 COPY	copie l'écran n1 dans l'écran n2 Il faut SAVE-BUFFERS ou FLUSH pour valider la copie ce qui permet de changer de fichier ou de disque
STAMP	Estampille la ligne zéro : votre cachet est affiché et demandé pour une mise à jour . Faire return dès qu'il est conforme

3) Les commandes EDITOR plein-écran sous SCREENS

a) se déplacer

les 4 flèches : déplacement du curseur dans le texte
SHIFT-gauche + les flèches : scrolling du texte dans l'écran selon ses dimensions affichables

RETURN : passage à la ligne suivante
CTRL-D : curseur en début de ligne puis début d'écran
CTRL-F : curseur en fin de texte de ligne
CTRL-N : curseur au mot suivant
CTRL-B : curseur au mot précédent

b) écrire

toute entrée de caractères en mode insertion sur ligne

c) effacer

DEL : effacement caractère arrière
CTRL-] : effacement caractère avant
CTRL-X : destruction de toute la ligne
(en répétition sert à effacer tout l'écran)
CTRL-L : effacement du reste de la ligne
CTRL-G : effacement tout un mot

d) couper-coller

CRTL-Y : coupe la ligne
(en début de ligne sert à inserrer une ligne)
CTRL-U : recolle la ligne
(la ligne suivante n'est supprimée que si elle est vide)

e) copier

CTRL-A : prend dans le PAD le reste d'une ligne
CTRL-Z : fixe la fin du bloc dans le PAD
(inutile si on veut copier toute une ligne)
CTRL-C : copie le bloc fixé entre CTRL-A et CTRL-Z à la position du curseur

f) terminer

ESC : sort de SCREENS mais reste dans EDITOR

g) fonction spéciale de SCREENS pour EDITOR

CTRL-P : bascule 32*32 / 16*64
ceci dans le but de faciliter la lecture verticale d'un écran en scrollant bas-haut plutôt que droite-gauche

Je vous conseille de vous familiariser avec cet éditeur plein-écran : vous découvrirez ses qualités très vastes et ses trucs (combinaisons de commandes, mixages, ajouts).

LES MOTS DU SEDORIC

Le F83 sait gérer les fichiers du DOS SEDORIC . Ceci permet une utilisation très aisée du langage et une ouverture nouvelle du FORTH à son environnement. Certains mots sont compilés dans le vocabulaire SEDORIC, d'autres d'usage général sont dans le vocabulaire FORTH.

1) Sous vocabulaire SEDORIC

EXT TYPE affiche l'extension des fichiers par défaut et qu'il n'est pas nécessaire de préciser dans un nom de fichier.
EXT: TXT affecte .TXT comme l'extension des noms de fichiers.

RDM-FILE donne au FORTH un gestion directe de la disquette secteur par secteur comme dans les anciens FIG. Le formatage de base est celui du T-FORTH dont on peut récupérer les écrans.Les disques en accès direct ne peuvent être utilisés pour d'autres programmes que ceux en FORTH.

SED-FILE permet au FORTH de gérer la disquette par fichiers SEDORIC : c'est le mode usuel d'utilisation du F83.

8 CREATE-FILE ESSAI crée un nouveau fichier ESSAI.ext sur la disquette comportant 8 blocs ou écrans. Ce fichier est rempli avec les 8 premiers Ko du langage aussi pour l'utiliser en FORTH faut-il l'initialiser avec rien dedans :

```
OPEN ESSAI    ouvre le fichier ESSAI nouvellement créé
ALSO EDITOR   prend l'éditeur pour modifier les écrans
1 8 CLEAN     efface tous les écrans de ESSAI.ext
```

Le fichier ESSAI est maintenant prêt à recevoir vos programmes FORTH. Ce fichier est manipulable comme tout fichier du SEDORIC. Il a une structure de type code-machine.

2) Sous FORTH

Nous avons déjà vu la commande la plus usuelle :

OPEN nom-de-fichier

qui ouvre un fichier et le rend courant. Il faut se souvenir que l'ouverture d'un fichier ne vide pas les tampons, ceci essentiellement pour faciliter les transferts et les copies d'un fichier à un autre. Aussi n'oubliez pas de faire un EMPTY-BUFFERS (CTRL-V ou FUNCT-E) ou encore un FLUSH si nécessaire quand vous changez de fichier.

TAKE nom-de-fichier quant-à lui charge (1 LOAD) un fichier secondaire (un package si vous voulez) mais une fois la compilation de ce fichier achevée retourne au fichier courant à l'endroit laissé. Exemple TAKE DEBUG compile très simplement le débogueur. Attention : DEBUG ne doit pas contenir un autre TAKE car cette version ne peut travailler que sur deux fichiers en même temps.

CAT affiche l'index (les premières lignes des écrans) de tout le fichier courant. C'est l'équivalent de 1 CAPACITY @ INDEX
DIR affiche le directory de la disquette.

INDEX DES MOTS FORTH

Le quick-index qui suit est l'index des mots du vocabulaire FORTH dans le fichier F83DEV.COM. Ce vocabulaire est plus limité dans le Kernel ou au contraire plus étendu si sont compilées des extensions.

LISTE DES ABREVIATIONS DU QUICK-INDEX :

(...--...)	action sur la pile
<...>	structure devant faire suite dans le flot d'entrée
Im.	mot immédiat
DW	mot d'exécution vectorisée (defered word)
o	octet
n	nombre 16 bits
u	nombre 16 bits non signé
d	nombre 32 bits sur deux cellules 16 bits
ud	nombre double (32 bits) non signé
ad	adresse 16 bits
l	longueur d'une chaine ou zone mémoire
car	caractère codé ASCII
f	flag booléen 0 ou -1
/	sépare deux actions possibles sur la pile
*	mot dangereux : à ne pas utiliser tel quel sans de sérieuses précautions de compilation (pour avertis)

```
!           (n,ad--) met n à l'adresse ad
!CSP        (--) sauve profondeur de pile pour compilation
"           <texte"> (--ad,l) Im. compile une chaine
#           (d1--d2) convertit un digit dans chaine numerique
#>          (d--ad,l) termine une conversion numerique-alpha
#BUFFERS    (--8) constante nombre de tampons de blocs : 8
#LINE       (--ad) variable nombre de lignes émises
#OUT        (--ad) variable nombre de caractères émis
#S          (d--0,0) conversion d'un double dans tous digits
#TIB        (--ad) variable nb de car. dans le tampon d'entrée
#VOCS       (--5) constante nombre de vocabulaires CONTEXT
'           <mot> (--cfa) donne le cfa d'un mot
(           <texte)> (--) début de commentaires
(")         (--ad,l) * primitive exécution de "
(+LOOP)     (n--) * primitive exécution de +LOOP
(.")        (--) * primitive exécution de ."
(.)         (n--ad,l) conversion nombre-chaine
(;CODE)     (--) * primitive exécution ;CODE
(;USES)     (--) * primitive exécution ;USES
(?DO)       (n1,n2--) * primitive exécution ?DO
(?ERROR)    (ad,l,f--) vecteur d'affichage de chaine d'erreur
(?KEY)      (--o) vecteur de saisie touche ?KEY
(ABORT")    (f--) * primitive exécution ABORT"
(ABORT)     (--) vecteur standard de ABORT
(CHAR)      (ad,n,car--ad,n+1) vecteur standard CHAR
(COLD)      (--) * partie LM d'initialisation système
(CR)        (--) CR sur écran : vecteur standard de CR
(D.)        (d--ad,l) conversion double-chaine
(DO)        (n1,n2--) * primitive exécution DO
(EMIT)      (o--) envoi car. écran : vecteur standard EMIT
(FIND)      (ad,lfal--cfa,f) primitive recherche mot dans vocabulaire
(FORGET)    (ad--) primitive de FORGET
(IS)        (cfa--) * primitive exécution IS en compilation
(KEY)       (--o) attente car. clavier : vecteur std KEY
(LOAD)      (n--) vecteur standard de LOAD
(LOOP)      (--) * primitive exécution LOOP
(NTOP)      (--ad) variable sommet dictionnaire
(NUMBER)    (ad--d) vecteur standard NUMBER
(NUMBER?)   (ad--d,f) primitive conversion alpha-numérique
(OF)        (n1,n2--) * primitive exécution OF
(SOURCE)    (--ad,l) vecteur standard SOURCE
(TERMINAL)  (--) vecteur standard TERMINAL
(U.)        (u--ad,l) conversion nombre non signé - chaine
(UAREA)     (--ad) variable zone des variables d'usager
(UD.)       (ud--ad,l) conversion double non signé - chaine
*           (n1,n2--n3) multiplication
*/MOD       (n1,n2,n3--n4,n5) reste et quotient règle de trois
*D          (n1,n2--d) multiplication simples - double
+           (n1,n2--n3) addition
+!          (n,ad--) ajoute n au contenu de ad
+LOOP       (n--) Im. compile fin de boucle incrémentée DO..+LOOP
,           (n--) compile un nb. 16 bits
,"          (--) primitive de compilation de chaine
-           (n1,n2--n3) soustraction
-->         (--) passe l'interprétation à l'écran suivant Im.
-ROT        (n1,n2,n3--n3,n1,n2) rotation inverse
-TRAILING   (ad,l1--ad,l2) supprime espaces en fin d'une chaine
.           (n--) affiche un nombre 16 bits
."          <texte"> (--) Im. compile l'affichage d'une chaine
```

```
."          <texte>) (--) Im. affiche un message en exécution
.CTRL       (--) affiche la liste des commandes de contrôles
.FILE       (--) affiche le nom du fichier courant DW
.FUNCT      (--) affiche les assignations des touches de fonctions
.R          (n,l--) affiche un nombre formaté à droite dans champ l
.S          (--) affiche le contenu de la pile
/           (n1,n2--n3) division entière plancher
/*          (n1,n2,n3--n4) règle de trois ( resul.intermédiaire double )
/MOD        (n1,n2--n3,n4) reste et quotient division entière
/STRING     (ad,l,l'--ad2,l2) troncature à droite d'une chaine
0           (--0) constante 0
0<          (n--f) teste si n<0
0<>         (n--f) teste si n<>0
0=          (n--f) teste si n=0
0>          (n--f) teste si n>0
1           (--1) constante 1
1+          (n--n+1) incrémente
1-          (n--n-1) décrémente
2           (--2) constante 2
2*          (n--n*2) multiplie par 2
2+          (n--n+2) ajoute 2
2-          (n--n-2) soustrait 2
2/          (n--n/2) divise par 2
2DROP       (d--) supprime deux cellules de la pile
2DUP        (d--d,d) duplique deux cellules de la pile
2OVER       (d1,d2--d1,d2,d1) copie l'avant-dernier couple de cellules
2SWAP       (d1,d2--d2,d1) permute deux couples de cellules
3           (--3) constante 3
3DUP        (n1,n2,n3--n1,n2,n3,n1,n2,n3) duplique trois cellules
:           <mot> (--) début de définition Forth "deux-points"
;           (--) Im. fin de définition "deux-points"
;CODE       (--) Im. compile passage définition en code
;USES       (--) Im. compile passage vers une ad de code
<           (n1,n2--f) teste si n1<n2
<#          (--) initialise conversion numérique-alpha
<>          (n1,n2--f) teste si n1<>n2
<MARK       (--ad) marque le début d'un branchement arrière
<RESOLVE    (ad--) compile un branchement arrière absolu
=           (n1,n2--f) teste si n1=n2
>           (n1,n2--f) teste si n1>n2
>BODY       (cfa--pfa) conversion CFA en PFA
>BUFFERS    (--ad) adresse des pointeurs de tampons
>END        (--ad) adresse fin des pointeurs de tampons
>IN         (--ad) variable pointeur d'interprétation dans la source
>LINK       (cfa--lfa) conversion CFA en LFA
>MARK       (--ad) compile un branchement avant absolu
>NAME       (cfa--nfa) conversion CFA en NFA
>R          (n--) empile sur pile de retour
>RESOLVE    (ad--) ajuste un branchement avant absolu
>SIZE       (--74) constante taille des pointeurs de tampons : 74
?           (ad--) affiche le contenu de ad
?BRANCH     (f--) * primitive de branchement conditionnel
?COMP       (--) vérifie mode compilation
?CSP        (--) vérifie profondeur de pile en fin de compilation
?DNEGATE    (d,n--d') change signe d'un double si n négatif
?DO         (n1,n2--) Im. compile début de boucle conditionnée n1<>n2
?DUP        (n--n/n,n) duplique une cellule si non nulle
?ENOUGH     (n--) vérifie au moins n cellules sur la pile
?ERROR      (ad,l,f--) DW affiche chaine et ABORT si flag d'erreur
?EXEC       (--) vérifie mode exécution
```

```
?KEY        (--o) DW saisit au vol car. clavier, zéro si pas de touche
?LEAVE      (f--) sortie conditionnelle de boucle DO..LOOP
?LOADING    (--) vérifie mode chargement d'écrans
?MISSING    (f--) erreur d'un mot introuvable
?NEGATE     (n1,n2--n1') inverse signe de n1 si n2 négatif
?PAIRS      (n1,n2--) vérifie la parité des structures de compilation
?STACK      (--) vérifie la conformité de la pile
?STOP       (--f) suspend l'exécution et teste si fin demandée
@           (ad--n) délivre le contenu de ad
A-IN        (ad,n1--ad,n2) saisie d'un car. sous curseur
ABORT       (--) DW ré-initialisation piles et interpréteur
ABORT"      <texte"> (f--) Im. compile un test d'erreur avec message
ABS         (n--n') valeur absolue
ABSENT?     (n--ad,f) cherche dans les tampons si un bloc est absent
ACCENT      (--) caractères accentués
AGAIN       (--) Im. fin de boucle infinie BEGIN..AGAIN
ALLOT       (n--) réserve n octets dans le dictionnaire
4 ALSO      (--) gère CONTEXT pour ajouter un vocabulaire
AND         (n1,n2--n3) "et" logique
ASCII       <lettre> (--o) Im. donne le code ASCII d'un car.
ASSEMBLER   (--) DW vocabulaire assembleur
ASSIGN      <mot> (code--) assigne un mot à une touche de fonction
ASSIGN"     <texte"> (code--) assigne une chaine à une touche de fonction
AT          (n1,n2--) positionne le curseur d'écran
AVOC        (--ad) variable de sauvegarde d'un vocabulaire
AZERTY      (--) clavier français
B/BUF       (--1024) constante bytes par tampon ou bloc : 1024
BASE        (--ad) variable base de numération
BASIC .     (--) sortie du F83, retour au BASIC
BEGIN       (--) Im. compile début de boucle infinie ou indéfinie
BETWEEN     (n1,n2,n3--f) teste si n2=< n1 =< n3
BL          (--32) constante code espace
BLANK       (ad,l--) remplit d'espaces une chaine ou zone mémoire
BLK         (--ad) variable bloc courant ( zéro si TIB )
BLOCK       (n--ad) délivre l'adresse d'un bloc ( lecture au besoin )
BODY>       (pfa--cfa) conversion PFA en CFA
BOOT        (--) DW procédure de mise en route du système
BOUNDS      (ad,l--ad,ad') début et fin d'une chaine
BRANCH      (--) * primitive de branchement inconditionnel
BS-IN       (--) déplacement curseur arrière
BUFFER      (n--ad) donne adresse d'un bloc ( pas de lecture )
BUFFER#     (n--ad) donne adresse d'un pointeur de tampon
BYE         (--) verrouille pointeurs, donne les adresses et sort du F83
C!          (o,ad--) met un octet à ad
C,          (o--) compile un octet
C-IN        (--) ctrl-C changement de disque warm-boot
C/L         (--64) constante nombre de car. par ligne : 64
C@          (ad--o) lit un octet à ad
CALL        (ad--) exécute une routine LM
CAP-IN      (--) bascule minuscules-majuscules du clavier
CAPACITY    (--ad) variable capacité d'écrans du fichier courant
CAPS        (--ad) variable booléenne d'interprétation des minuscules
CAPS-COMP   (ad1,ad2,l--f) comparaison de chaines minus=majuscules
CASE        (--) Im. compile début structure CASE..OF..ENDOF..ENDCASE
4 CAT       (--) affiche l'index du fichier courant
CHAR        (ad,n,car--ad,n') DW mise en place d'un car dans un tampon
CLIT        (--) * primitive d'un litéral 8 bits
CLL-IN      (--) effacement de la ligne au curseur
CLS         (--) effacement écran
CMOVE       (ad1,ad2,l--) transfert de l octets de ad1 à ad2 par le bas
```

```
CMOVE>      (ad1,ad2,l--) transfert de l octets de ad1 à ad2 par le haut
CODE        <mot> (--) définition d'un mot en code machine
COLD        (--) * lancement du système
COMP        (ad1,ad2,l--f) comparaison de chaines minus<>majuscules
COMPARE     (ad1,ad2,l--f) comparaison de chaines, minuscules selon CAPS
COMPILE     <mot> (--) compile le cfa d'un mot
CONSTANT    <mot> (n--) définition d'une constante
3 CONTEXT   (--ad) pile des vocabulaires de contexte
CONTROL     <lettre> (--o) empile ou compile le code contrôle ( A -> 1 )
CONVERT     (ud1,ad1--ud2,ad2) primitive de conversion alpha-double nombre
COUNT       (ad1--ad2,l) conversion chaine implicite en chaine explicite
CR          (--) DW envoi d'un retour-chariot
CR-IN       (ad,n--ad,n') fin de saisie par touche 'return'
CRASH       (--) vecteur d'initialisation d'un mot vectorisé
CREATE      <mot> (--) crée un en-tête de mot, cfa type variable
CRESET      (o,ad--) met à 0 les bits d'ad selon bits 1 de o
CSET        (o,ad--) met à 1 les bits d'ad selon bits 1 de o
CSP         (--ad) variable trace de profondeur de pile en compilation
CTOGGLE     (o,ad--) inverse les bits d'ad selon bits 1 de o
CTRL        (--ad) variable vecteur de la table des contrôles
CTRL1       (--ad) table des contrôles de base
3 CURRENT   (--ad) pointeur du vocabulaire courant
CURSOR      (--) bascule curseur visible
D+          (d1,d2--d3) addition de doubles
D.          (d--) affichage d'un double
D.R         (d,l--) affichage d'un double formaté à droite dans champ l
DABS        (d--d') valeur absolue double
DECIMAL     (--) base courante décimale
DEFAULT     (--) DW ouvre un fichier par défaut
DEFER       <mot> (--) définition d'un mot vectorisé ( ou différé )
DEFINED     <mot> (--cfa,f) cherche le mot qui suit dans CONTEXT
DEFINITIONS (--) rend courant le premier vocabulaire de CONTEXT
DEL-IN      (ad,l--ad,l') annulation du dernier car. saisi
DELETE      (ad1,l1,l2--) supprime les l2 premiers car d'une chaine
DEPTH       (--n) donne la profondeur de la pile en cellules
4 DESAS     (ad--) désassemble une zone mémoire
DIGIT       (car,base--n,f) primitive de conversion d'un car en nombre
4 DIR       (--) directory des fichiers du disque courant
DISCARD     (--) annule la mise à jour du bloc courant
DISK-ERR?   (f--) erreur R/W du lecteur disque
DLITERAL    (d--) compile un litéral double
DNEGATE     (d--d') inverse le signe d'un double
DO          (n1,n2--) Im. compile début d'une boucle définie DO..LOOP/+LOOP
DOES>       (--) Im. début compilation partie exécution dans définisseur
DONE?       (n--f) teste la fin de SOURCE ou changement de STATE
DOUBLE?     (--f) teste si denière conversion alpha-double
DOWN-IN     (--) déplacement curseur bas
DP          (--ad) variable pointeur du dictionnaire
DPL         (--ad) variable point décimal d'un double
DROP        (n--) supprime une cellule sur la pile
DUMP        (ad--) affiche une "vidange-mémoire"
DUP         (n--n,n) duplique une cellule sur la pile
EDIT        (n--) DW lance l'édition de l'écran n qui devient écran courant
ELSE        (--) Im. compile une alternative dans IF..ELSE..THEN
EMIT        (n--) DW envoie un car au terminal de sortie
EMPTY-BUFFERS (--) vide les tampons et initialise pointeurs de tampons
END?        (--ad) variable fin d'interprétation
ENDCASE     (--) Im. fin de structure CASE..OF..ENDOF..ENDCASE
ENDOF       (--) Im. fin de choix dans structure CASE..OF..ENDOF..ENDCASE
ERASE       (ad,l--) rempli avec octet 0 une chaine ou zone mémoire
```

```
EXECUTE   (cfa--) exécute un mot
EXIT      (--) quitte un niveau d'interprétation interne
EXPECT    (ad,l--) entrée d'une chaine au terminal ; exécute les ctrls
FALSE     (--0) constante booléenne faux : 0
FENCE     (--ad) variable fin du dictionnaire protégé
FILE      (--ad) variable fichier courant
FILL      (ad,l,o--) remplit une zone mémoire avec octet o
FIND      (ad,l--cfa,f) cherche une chaine dans CONTEXT
FIRST     (--ad) constante début des tampons : hexa 7800
FLD       (--ad) variable champ d'en-tête (non utilisée)
FLIP      (n1--n2) échange octets haut et bas d'une cellule 16 bits
FLUSH     (--) sauve les blocs mis à jour et vide les tampons
FONCTION  (--) traitement des touches de fonction
FORBID    <mot> (--) interdit un mot
FORGET    <mot> (--) oublie un mot et tous ceux qui le suivent
FORTH     (--) vocabulaire FORTH
FOUND     (--ad) variable de travail de SEARCH
FUNCT     (--ad) table des assignations des touches de fonction
FW-IN     (--) déplacement curseur avant
H.        (n--) affiche n en hexa non signé sans changer BASE
HASH      (ad1,ad2--n) détermine le hash-code d'un mot dans vocabulaire
HEADER    <mot> (--) crée un en-tête de mot dans le dictionnaire sans cfa
HELLO     (--) vecteur standard de BOOT
HERE      (--ad) adresse courante du dictionnaire
HEX       (--) base courante hexadécimale
HIDE      (--) occulte la dernière création dans le dictionnaire
HLD       (--ad) variable de travail conversion numérique-alpha
HOLD      (car--) place un car dans une conversion numérique-alpha
HOME      (--) curseur en début d'écran
I         (--n) indice de boucle DO..LOOP
ID.       (nfa--) affiche le nom d'un mot
IF        (f--) Im. compile un branchement conditionnel
IMMEDIATE (--) rend immédiate la dernière définition
INDEX     (n1,n2--) affiche l'index des écrans n1 à n2 du fichier courant
INK       (n--) fixe l'encre des caractères
INSERT    (ad1,l1,ad2,l2--) inserre chaine 1 en début de chaine 2
INTERPRET (--) lance l'interpréteur externe de SOURCE
IS        <mot> (cfa--) vectorise un mot d'exécution vectorisée
KCLK      (--) bascule clic clavier
KEY       (--o) DW attend un car au terminal d'entrée
L>NAME    (lfa--nfa) conversion LFA en LFA
LARGEST   (ad1,n1--ad2,n2) primitive de WORDS plus grande ad d'une série
LAST      (--ad) variable du NFA du dernier mot défini
LATEST?   (n--n/a,f) teste si un bloc est le plus recemment référencé
LEAVE     (--) sortie inconditionnelle d'une boucle DO..LOOP
LIMIT     (--ad) constante fin des tampons
LINE      (n--ad,l) adresse et longueur de la ligne n de l'écran courant
LINK>     (lfa--cfa) conversion LFA en CFA
LIST      (n--) liste l'écran n qui devient écran courant
LIT       (--) * primitive de litéral 16 bits
LITERAL   (n--) Im. compile un litéral 8 ou 16 bits
LOAD      (n--) DW charge et interprète l'écran n du fichier courant
LOCK      (--) verrouille le dictionnaire, ajuste les pointeurs initiaux
LOOP      (--) Im. compile fin de boucle définie DO/?DO..LOOP
M/MOD     (d1,n1--n2,n3) reste et quotient division double/simple
MAX       (n1,n2--n3) maximum de n1 ou n2
MIN       (n1,n2--n3) minimum de n1 ou n2
MISSING   (--) fait de la place dans les tampons pour nouveau bloc
MOD       (n1,n2--n3) reste de la division entière
MOVE      (ad1,ad2,l--) transfert de l octets de ad1 à ad2 sans overlap
```

```
MU/MOD      (d1,n1--n2,d2) reste et quotient division double/simple
N>LINK      (nfa--lfa) conversion NFA en LFA
NAME>       (nfa--cfa) conversion NFA en CFA
NEGATE      (n1--n2) inverse le signe
NIP         (n1,n2--n2) supprime l'avant-dernière cellule
NOOP        (--) mot sans action
NOT         (f--f') inverse un booléen
NUMBER      (ad--d) DW convertit une chaine en nombre double
NUMBER?     (ad--d,f) primitive de (NUMBER) conversion chaine-double
OF          (n1,n2--n1) Im. compile choix dans structure CASE..OF...
OFF         (ad--) place zéro à ad
OFFSET      (--ad) variable réserve de bloc
ON          (ad--) place -1 (vrai) à ad
ONLY        (--) vocabulaire ONLY seul vocabulaire de CONTEXT
OPEN        <nom-fichier> (--) ouvre un fichier et le rend courant
OR          (n1,n2--n3) "ou" logique
OVER        (n1,n2--n1,n2,n1) copie l'avant-dernière cellule
PAD         (--ad) adresse du 'PAD' ou bloc-note tampon temporaire
PAPER       (n--) couleur du fond écran
PARSE       (car--ad,l) isole du flot d'entrée une chaine jusqu'à car
PARSE-WORD  (car--ad,l) isole du flot d'entrée une chaine séparée par car
PICK        (n1--n2) copie la n1-ième avant-dernière cellule de la pile
PLACE       (ad1,l,ad2--) place en ad2 chaine explicite depuis ch.implicite
PREVIOUS    (--) supprime le premier vocabulaire de CONTEXT
PRIOR       (--ad) variable travail de FIND
QUERY       (--) attend une entrée dans le TIB
QUIT        (--) ré-initialise l'interpréteur externe
QWERTY      (--) clavier anglais
R           (--n) synonyme de R@
R#          (--ad) variable pointeur dans l'écran courant
R0          (--ad) constante adresse de la pile de retour : hexa 1FF
R0RP!       (--) * initialise la pile de retour
R>          (--n) dépile la pile de retour
R@          (--n) copie une cellule de la pile de retour
RAM/ROM     (--) * bascule RAM-overlay - ROM
RD#         (piste,secteur,ad--f) primitive lecture de secteur
RDBLOCK     (ad--) lecture de bloc en accès direct sans sedoric
READ-BLOCK  (ad--) DW lecture d'un bloc
REPEAT      (--) Im. compile fin de boucle indéfinie BEGIN..WHILE..REPEAT
REPLACE     (ad1,l1,ad2,l2--) surcharge chaine 1 dans la chaine 2
RESERVE     (--150) constante de secteurs réservés en accès direct
REVEAL      (--) valide la dernière définition
ROT         (n1,n2,n3--n2,n3,n1) rotation sur trois cellules
RP!         (n--) * force le pointeur de la pile de retour
RP@         (--n) donne le niveau du pointeur de la pile de retour
RUN         (--) interprète la SOURCE en multiligne
S/T         (--16) constante secteurs par piste en accès direct
S0          (--ad) constante adresse de la pile : hexa 9D
S>D         (n--d) conversion nombre 16 - 32 bits
SAVE-BUFFERS (--) sauve sur disque les blocs mis à jour
SCAN        (ad1,l1,car--ad2,l2) cherche car dans chaine : donne ch.restante
SCR         (--ad) variable écran courant
SEAL        (--) isole le premier vocabulaire dans CONTEXT
SEARCH      (ad1,l1,ad2,l2--n,f) cherche chaine 1 dans chaine 2
SEE         <mot> (--) DW décompile un mot t
SIGN        (n--) inserre le signe dans une conversion numérique-alpha
SKIP        (ad1,l1,car--ad2,l2) saute d'une chaine les car de début
SOURCE      (--ad,l) DW adresse et longueur de la source à interpréter
SP!         (ad--) * force le pointeur de pile
SP@         (--n) donne le niveau du pointeur de pile
```

```
SPACE       (--) envoie un espace
SPACES      (n--) envoie n espaces
SPAN        (--ad) variable nombre de car saisis
STATE       (--ad) variable état système exécution ou compilation
STATUS      (--) DW message de fin d'exécution ( simple CR en vecteur std )
SWAP        (n1,n2--n2,n1) permutation de deux cellules
TAKE        <nom-fichier> charge un fichier-package et revient au fic. crt.
TERMINAL    (--) DW configure le terminal
THEN        (--) Im. compile une fin de condition IF..( ELSE )..THEN
THRU        (n1,n2--) LOAD les écrans n1 à n2
TIB         (--ad) constante adresse du Terminal Input Buffer : hexa 100
TRAVERSE    (ad1,sens--ad2) traverse un NFA cadré par bits de poids fort
TRIM        (ad1,ad2--) primitive de (FORGET)
TRUE        (-- -1) constante booléenne vrai : -1
TUCK        (n1,n2--n2,n1,n2) copie une cellule sous les deux premières
TYPE        (ad,l--) envoie une chaine au terminal de sortie
U*D         (u1,u2--d) synonyme de UM*
U.          (u--) affiche un 16 bits non signé
U.R         (u,l--) affiche un non-signé formaté à droite dans champ l
U2/         (u1--u2) division par deux non signée
U<          (u1,u2--f) teste si u1<u2
U>          (u1,u2--f) teste si u1>u2
UD.         (ud--) affiche un double non signé
UM*         (u1,u2--d) multiplication mixte simples non signés - double
UM/MOD      (d1,n1--n2,n3) reste et quotient simples division mixte
UNNEST      (--) * primitive de fin de définition deux-points
UNTIL       (f--) Im. compile fin de boucle indéfinie BEGIN..UNTIL
UP-IN       (--) déplacement curseur haut
UPDATE.     (--) marque l'écran courant comme mis à jour
UPPER       (ad,l--) force une chaine en majuscules
USER        <mot> (n--) définition d'une variable usager
VARIABLE    <mot> (--) définition d'une variable
VERSION     (--ad,l) chaine de la version en cours
VOC-LINK    (--ad) variable de lien des vocabulaires
VOCABULARY  <mot> (--) définition d'un vocabulaire
WARM        (--) ré-initialisation à chaud
WARNING     (--ad) variable booléenne de mise en garde des doublons
WHERE       (ad,n--) DW localise une erreur
WHILE       (f--) Im. compile condition dans boucle BEGIN..WHILE..REPEAT
WIDTH       (--ad) variable longueur maximale d'un NFA
WITHIN      (n1,n2,n3--f) teste si n2 =< n1 < n3
WORD        (car--ad) isole du flot d'entrée un mot séparé par car
WORDS       (--) liste les mots du premier vocabulaire de CONTEXT
WR#         (piste,secteur,ad--f) primitive d'écriture d'un secteur
WRBLOCK     (ad--) écriture d'un bloc en accès direct sans sedoric
WRITE-BLOCK (ad--) DW écriture d'un bloc
X-IN        (ad,n--ad,0) annulation de saisie
XOR         (n1,n2--n3) "ou exclusif" logique
[           (--) Im. passage en exécution
['] <mot> (--cfa) Im. compile en littéral le CFA d'un mot
[COMPILE] <mot> Im. force la compilation d'un mot immédiat
\           <texte> Im. passe l'interpréteur à la ligne suivante
]           (--) passage en compilation
```

VOCABULAIRE SEDORIC

```
(.FILE)     (--) affiche nom du fichier crt : vecteur de .FILE
(DEFAULT)   (--) ouvre le premier fichier .ext : vecteur DEFAULT
(FROM)      (--) transfert FCB - FCB1
(SAVE-FILE) (ad1,ad2--) primitive de sauvegarde sur disque
(TO)        (--) transfert FCB1 - FCB
?FILE       <nom-fichier> (--) cherche un fichier et le rend courant
?RANGE      (n--n) vérifie qu'un numéro de bloc est compatible
?SED        (--) vérifie le mode sedoric
B/FCB       (--20) constante dimension d'un FCB : 20 octets
CREATE-FILE <nom-fichier> (n--) crée un fichier de n blocs
EXT         (--ad,3) chaine de l'extension par défaut
EXT:        <ext> (--) change l'extension par défaut
FCB         (--ad) ad du FCB ( File Control Buffer ) courant
FCB!        (fcb--) remplit un FCB
FCB1        (--ad) ad du FCB temporaire
FILE!       (--) rend courant le fichier pointé par sedoric
FILE-READ   (ad--) vecteur de READ-BLOCK en sedoric
FILE-WRITE  (ad--) vecteur de WRITE-BLOCK en sedoric
FILENAME    <nom-fichier> (--) prend un nom ds bufnom du sedoric
NAME!       (ad,1--) charge bufnom sedoric avec chaine, ajoute l'ext.
OPEN-FILE   (--) ouvre le fichier courant
RDM-FILE    (--) configure en accès direct de secteurs
RECORD#     (n--piste,secteur) adresse physique d'un enregistrement sed
SEARCH      (--f) cherche un fichier sur le disque
SED-FILE    (--) configure en sedoric
```

Le fichier SEDORIC ajoute des fonctions annexes à la gestion du DOS avec notamment copies de fichiers, rename, delete, et l'utilisation en multi-fichiers.

VOCABULAIRE PRINTER

Il ne contient que les mots fondamentaux de gestion de l'imprimante

```
(P-CR)      (--) envoie un CR sur imprimante et écran
(PCR)       (--) envoie un CR sur imprimante
(PEMIT)     (car--) envoie un car sur imprimante et écran
(PRINT)     (car--) envoie un car sur imprimante
P-IN        (--) bascule écho imprimante ( exécuté par CTRL-P )
POFF        (--) écho imprimante OFF
PON         (--) écho imprimante ON
```

Le fichier PRINTER ajoute des fonctions annexes au vocabulaire PRINTER telles que driver français/anglais des caractères, types et polices de caractères, espacements, soulignage, gras, listing des écrans en simple ou double, formatages. Ces mots doivent être adaptés à votre imprimante.

VOCABULAIRE TOOLS

Ce vocabulaire sert de fourre-tout pour des mots qui encombreraient le glossaire FORTH usuel. Il contient les mots servant au désassembleur et au décompilateur qui n'apparaissent ainsi dans FORTH que sous DESAS et SEE. Il serait fastidieux d'en faire l'index . Sachez toutefois que vous pouvez y trouver des mots utiles notamment dans la gestion de tableaux. Le vocabulaire TOOLS est créé la première fois dans DESAS : il faut le définir avant de compiler une application qui l'utilise si le désassembleur n'est pas compilé.

PRECISIONS DIVERSES

Le système est au départ en CAPS ON : l'interpréteur comprend aussi bien les minuscules que les majuscules.

Pour sauver une application dans un fichier COM faites BYE , lisez les adresses xxxx et yyyy de début et fin et sauvez votre application par

```
SAVE "MYFORTH.COM" , A#xxxx , E#yyyy , AUTO
```

Pour revenir au FORTH après une escapade en BASIC (par exemple pour utiliser des commandes du DOS) faites CALL #504.

En cas de plantage, le bouton RESET retourne au FORTH mais les tampons sont vidés et un fichier est ouvert pas défaut. Il en est de méme sur une instruction BRK.

F83 : PRINCIPALES MODIFICATIONS PAR RAPPORT AUX ANCIENS FORTH

Les vocabulaires sont indépendants. Le concept ONLY ALSO permet de chaîner jusqu'à cinq vocabulaires de recherche.

CONTEXT est une pile de un à cinq vocabulaires.

Les vocabulaires ne sont pas immédiats.

WORDS remplace VLIST et ne liste qu'un seul vocabulaire.

DEFER et IS introduisent le concept des mots vectorisés notamment utilisés pour les entrées-sorties.

La mémoire de masse est gérée en fichiers compatibles avec le DOS hôte et non plus en accès direct de blocs.

Les blocs FORTH sont de même taille que les écrans. Les tampons de blocs sont gérés dynamiquement par des pointeurs de tampons dans une zone séparée des tampons. Ceux-ci n'ont plus de zéros séparateurs entre blocs.

La compilation des branchements est en adressage mémoire absolu au lieu d'un adressage relatif au pointeur d'interprétation.

La stucture DO...LOOP est améliorée. La primitive (DO) empile l'adresse de fin de boucle sur la pile de retour ce qui permet à ?DO , LEAVE , ?LEAVE de sortir directement d'une boucle en cours d'exécution.

Le chaînage des mots dans un vocabulaire est direct de link à link et non plus imbriqué link-name-link-name.

Les recherches dans un vocabulaire sont encore accélérées par le tissage de chaque vocabulaire en quatre brins : un mot est chainé dans un des quatre brins par hash-code sur la première lettre de son nom.

' (tick) renvoie le cfa d'un mot et non son pfa. C'est un mot non immédiat. Le litéral d'un cfa est compilable par ['] .

La division entière répond aux principes mathématiques de la division plancher.

VARIABLE n'est pas à précéder d'une valeur d'initialisation; celle-ci est toujours zéro.

Les booléens sont 0 (FALSE faux) et -1 (TRUE vrai) .

CREATE crée un en-tête sans pfa. Le cfa par défaut est celui d'une variable.

Les mots de définitions utilisent une structure CREATE..DOES> optimisée avec renvoi direct du code exécutif depuis le cfa du mot défini vers la partie exécution du mot définisseur.

L'interprétation utilise des mots séparés pour l'exécution ou la compilation. Cette séparation détaille les processus généraux pour permettre des méta-processus ultérieurs.

Les stuctures de contrôle de compilation utilisent des primitives de couplages. (<MARK <RESOLVE >MARK >RESOLVE).

La gestion du clavier admet l'exécution immédiate de codes de contrôle. Ces commandes sont vectorisées dans une ou des tables.

Le sommet de la pile est indicé à partir de zéro (PICK).

Les conversions entre les différents champs de l'en-tête d'un mot ont des noms non ambigus d'une désignation explicite (>BODY >NAME N>LINK BODY>). Le lfa est compilé avant le nfa lequel précède le cfa.

." ne s'utilise qu'en compilation. Il faut utiliser .(en exécution.

La compilation des chaines est généralisée (S" ,").

POINT DE VIEW

Le F83 du CP/M ou MS-DOS offre les view-fields et les écrans SHADOW qui gardent dans la définition d'un mot le site de définition d'un mot et doublent les écrans de définitions avec des écrans de commentaires. Malgré son côté attrayant pour voir le source d'un mot ce dispositif est trop rigide, encombre le dictionnaire, divise par deux la mémoire de masse disponible, fait double emploi avec des structures plus souples (décompilateur, écrans d'aide), oblige à de lourdes procédures de compression-décompression des fichiers etc...

Pour toutes ces raisons ayant engagé une polémique et pour la raison que l'ATMOS est une petite machine, nous n'avons pas compilé de view-fields. Ceci n'a aucune incidence sur l'utilisation et la compatibilité du F83. Quant aux écrans SHADOW, on peut toujours en proposer et les lister en vis-à-vis. C'est ce qui a été fait avec ASSEMBLER.

EPILOGUE A LA VERSION 2.0

Votre F83 est un prodigieux outil de développement. Si vous êtes un vrai forthien, vous conviendrez que sa mise au point a été un travail de "forthené". Mon but n'a été que d'aider à la diffusion du FORTH. Toutefois mon expérience avec le T-FORTH, FIG que j'avais diffusé en freeware n'a pas été très encourageante : un seul correspondant m'a fait l'amitié d'un " retour d'ascenseur " !

Si vous désirez la version 3.0 déjà largement avancée, il vous faudra participer un peu en développant vous-même une amélioration, un programme ou une application si modeste soit-elle. Soyez vous aussi des forthiens actifs !

Au menu de la version 3.0 :

- TASKER le traitement multi-tâche
- SEDORIC les extensions DOS et fichiers
- TEX le traitement de texte sous SCREENS
- BASILE2 la base de données
- MFLOAT calculs flottants multi-précision
- FORTHLOG système expert de JEDI transposé F83
- META le métacompilateur
- MACROS la bibliothèque de macros et le méta-assembleur
- CALC le tableur
- DAO
- ...

Bien Forth à vous !

Michel ZUPAN