

Note pour les propriétaires de versions antérieures de DOSPLUS

Certaines fonctions du système ont subi des modifications importantes, qui nécessitent des explications. Ce sont en particulier :

- (1) Les méthodes de configuration du système.
- (2) La préparation d'un disque maître en double densité.

Pour tous les propriétaires de DOSPLUS 3.5, même si ils ont eu des versions antérieures de DOSPLUS il est très important de lire avec attention la description de l'ordre SYSTEM. Elle fournit beaucoup de renseignements sur la façon d'ajuster DOSPLUS 3.5 à leurs besoins. Jointes à ce qui suit, il devrait être possible de configurer ce système de toute manière souhaitée.

Méthodes de configuration du système

Les versions antérieures de DOSPLUS utilisaient un principe de 'peripherals internes', c'est à dire que les pilotes (drivers) étaient inclus dans le système opératoire de DOSPLUS. Ce qui obligeait à écrire une nouvelle version de ce système chaque fois que l'on voulait contrôler un nouveau périphérique (imprimante, disque dur, etc...), c'était devenu infernal.

Maintenant, DOSPLUS utilise un principe de 'peripherals externes', c'est à dire que les pilotes nécessaires au fonctionnement des divers périphériques ne font plus partie du système, mais lui sont extérieurs, et ne lui sont rattachés que sur un choix de l'utilisateur. À défaut, DOSPLUS utilisera les pilotes de la ROM, qui sont donc les PILOTES NORMAUX de DOSPLUS 3.5.

Cette méthode est bien plus souple et nous permet de fournir le support d'une variété bien plus importante de périphériques, sans avoir à apporter de modifications majeures au reste du système. Elle lève aussi certaines restrictions sur la mémoire, qui était encombrée par de nombreux pilotes toujours présents.

Elle a aussi pour conséquence qu'il faut provoquer la mise en service de certaines fonctions évoluées, alors qu'elles étaient disponibles en permanence. L'ordre ASSIGN est là pour ce faire.

Pilote d'imprimante : Sur Modèle I comme sur Modèle III, DOSPLUS utilise le pilote de la ROM, à moins que l'on ne provoque la mise en service du pilote PR/DVR, fourni sur le disque, qui permet la mise en page automatique, le contrôle d'imprimante série (à condition d'avoir le matériel adapté), la sortie avec tampon disque (SPOOL), etc... La préparation du système est plus longue, mais nous permet de préparer des pilotes spéciaux pour des imprimantes non standard. Nous verrons plus tard comment les installer en permanence.

Pilotes clavier et vidéo : Sur Modèle I, les habitués seront surpris de ne pas retrouver les touches à répétition, les minuscules (si leur machine en est équipée) à la première mise en route. Ces pilotes existent toujours, mais il faut demander leur activation (KI/DVR et DD/DVR). Il est important de remarquer que RIEN ne manque, mais qu'il faut provoquer les mises en route.

Beaucoup d'utilisateurs des versions précédentes verront qu'ils ne peuvent pas configurer le système avant que les pilotes de DOSPLUS ne soient installés. Les pilotes normaux sont ceux de la ROM.

Tout ceci n'est pas limité aux pilotes ci-dessus. Nous pouvons fournir (en supplément) des pilotes de disques durs, de lecteur 8 pouces, etc...

Configuration permanente de DOSPLUS

Après cet exposé des principes, nous allons voir comment rendre permanente la configuration élaborée.

Dans les systèmes précédents, il fallait utiliser plusieurs ordres pour configurer DOSPLUS: CONFIG, RS232, FORMS, et d'autres. Il y avait pour chacun d'eux une façon, souvent SAVE, de le figer.

Un des changements apportés par la version 3.5 est qu'il ne suffit plus de sauvegarder quelques paramètres, mais aussi les pilotes eux-mêmes, sinon il faudrait les recharger à chaque lancement.

Si par exemple vous avez installé le pilote d'imprimante pour commander une imprimante série, en utilisant ASSIGN pour installer le pilote PR/DVR avec les paramètres voulus. Vous avez aussi utilisé FORMS pour faire la mise en page de votre impression. Tout marche bien, jusqu'à la prochaine initialisation du système, où il faudrait refaire tout ce travail.

C'est pourquoi nous avons fourni la possibilité d'écrire des 'fichiers de configuration', à ne pas confondre avec les ordres de configuration. Ils ne font qu'enregistrer la configuration établie au moyen des ordres.

Toute l'information est enregistrée sur un fichier disque, comme un programme en langage machine. Il suffit ensuite d'exécuter ce fichier comme n'importe quel autre programme, pour retrouver la configuration initiale.

Vous pouvez bien sûr mettre ce fichier dans une ligne AUTO pour retrouver cette configuration dès l'initialisation. En utilisant l'AUTO invisible, on peut même le faire discrètement, et comme on peut mettre plusieurs commandes dans une instruction AUTO, le fichier de configuration peut y être inclus parmi d'autres ordres.

Enfin, cette technique peut être utilisée pour contrôler des disques durs. Une fois le lecteur initialisé, vous pouvez lui avoir, si vous le désirez, transféré le contrôle du système. Puis vous faites un fichier contenant cette configuration. Après l'initialisation par le disque souple, l'exécution de ce fichier assurera la reconfiguration instantanée pour le disque dur.

Vous pouvez, bien sûr, faire autant de fichiers de configuration que vous le désirez, chacun différent des autres, et changer de configuration simplement par l'appel de l'un d'eux. Bien que cela ne soit pas nécessaire, il est bon d'avoir sur le disque, au moins un fichier avec une configuration minimum, qui peut servir de point de départ pour la confection des autres.

A noter que si des articles tels que TRAF, JCL, ou des filtres sont actifs, ils seront aussi enregistrés dans le fichier de configuration, ainsi que le CONFIG en vigueur et les noms de périphériques; en somme tout est sauvegardé.

Cette traduction a été faite par: Claude Pilverdier
60, grande rue
89550 Hery

Tel: (86) 47 73 07

COPYRIGHT (c) 1983 par PENTASONIC
5 rue Maurice Bourdet
75016 PARIS

Confection d' un disque systeme maitre double densite pour Modele I

Tout ce qui suit suppose que vous avez le materiel necessaire pour travailler en double densite, c' est a dire qu' un doubleur de densite est installe dans l' interface d' expansion.

Creation du disque :

La premiere et la plus importante chose a se rappeler est de faire une copie de securite des deux disques Modele I avant toute chose, et de ne travailler qu' avec les doubles.

DOSPLUS 3.5 est fourni sur deux disques simple densite. C' est pour qu' ils puissent etre lus et charges sur n' importe que systeme. Il n' est plus necessaire d' avoir une version double densite differente. Il faut toutefois suivre une procedure simple pour obtenir un disque maitre en double densite.

Note: Les proprietaires du doubleur de densite RADIO-SHACK, catalogue 26-1143, doivent installer une modification avant de commencer. Elle est decrite plus loin.

La premiere chose a faire est de formater le disque: utiliser le programme FORMAT, et repondre 'D' quand il demande quelle densite utiliser (NdT: cela ne peut se faire qu' avec 2 lecteurs). Puis utiliser l' ordre SYSGEN pour copier le systeme sur le disque fraichement formate.

Quand c' est termine, utiliser l' ordre COPY (voir COPY dans la librairie) pour recopier tous les fichiers du disque simple densite qui est dans le lecteur systeme sur le disque double densite. Quand le premier disque original est recopie en entier, mettre le deuxieme a sa place, et repeter la copie. Le contenu des deux disques sera alors sur un seul disque double densite.

Une fois cette operation terminee, le disque maitre double densite est pret, rangez le soigneusement et faites vos recopies d' apres lui.

Exemple :

Placer le disque simple densite etiquete 'A' dans le lecteur systeme, initialiser la machine, et placer un disque vierge dans le lecteur 1. Taper l' ordre

FORMAT :1

Quand il demande 'Simple or Double', taper 'D', repondre aux autres questions suivant le materiel dont vous disposez (nombre de cylindres, nombre de faces, etc.). Quand le formatage est termine, oter le disque marque 'A' ou 'SDEN' et le remplacer par l' autre, marque 'B', ou sans le tampon 'SDEN'. Puis taper l' ordre

SYSGEN :1

SYSGEN va copier les modules systeme sur le disque du lecteur 1. Quand c' est fini, ce disque ne contiendra que les fichiers systeme. Pour copier les utilitaires et les autres fichiers sur ce disque commander

COPY :0 :1, ECHO, SPW="PASSWORD", INV, NEW

Ce qui copiera tous les fichiers simple densite sur le nouveau disque double densite, en supposant que le mot de passe du disque est encore PASSWORD. Si vous l' avez modifie, modifiez la commande en consequence. Quand c' est termine, echangez le disque du lecteur 0 avec celui qui est marque 'A' ou 'SDEN', et retapez le meme ordre, ou '/' et 'ENTER'.

Quand c' est termine, vous etes en possession d' un disque systeme double densite, qui vous servira a faire tous vos BACKUP ulterieurs. Vous feriez de meme pour faire un disque maitre 80

pistes ou double face.

Ce qui suit est destine a tous les utilisateurs de Modele I, equipes du doubleur de densite de Radio-Shack, et concerne une modification a appliquer avant que DOSPLUS puisse fonctionner avec.

NOTICE A TOUS LES UTILISATEURS DE MODELE I.

Si vous avez un doubleur de densite Radio-Shack, ref. cat. 26-1143, vous devez modifier votre DOSPLUS 3.5 pour qu' il puisse fonctionner correctement. Pour faire cette modification, entrez la commande suivante sous DOSPLUS :

DO RS

Cet ordre va lancer une suite d' operations qui modifiera DOSPLUS dans le lecteur 0 dans le sens souhaite.

Il est a noter que cette modification est deconseillee pour les autres doubleurs de densite, dont le fonctionnement pourrait etre perturbe.

Exemple :

Placer une copie du disque marque 'A' ou 'SDEN' dans le lecteur 0. Sous DOSPLUS entrer l' ordre :

DO RS

Ceci provoquera la modification du disque pour le fonctionnement avec le doubleur Radio-Shack. Quand c' est termine, presser le bouton 'RESET', et recommencer les operations precedentes pour faire un disque maitre double densite.

Si pour une raison quelconque, vous desirez que le disque marque 'B', ou sans la marque 'SDEN' fonctionne aussi en double densite, vous devrez d' abord copier le fichier 'RS/TXT' du disque 'B' au disque 'A' avec le disque 'B' en lecteur 0, et redonner l'ordre 'DO RS'.

NOTE : Vous devez reinitialiser apres avoir execute la modification. C' est vital pour le fonctionnement correct du systeme. Ensuite, en constituant le disque maitre, il NE FAUT PAS reinitialiser au changement de disques, pour que la modification reste en memoire jusqu' a la fin du travail.

Quelques indications.

Pauses pendant DIR, FREE, CAT:

Il est possible de suspendre l' affichage provoque par ces ordres. Cela se fait en pressant la barre d' espace. la sortie s' arrete jusqu' a ce que l' on presse une touche quelconque.

Cete methode fonctionne avec la plupart des ordres DOS, comme LIST, MAP, etc. Mais BASIC utilise toujours <shift> à.

Recopie d' ecran.

Si vous avez installé le pilote de remplacement KI/DVR, vous pouvez à tout moment obtenir l'impression du contenu de l'écran en faisant 'control *' ('control' = <shift>+flèche en bas). Ceci s'applique au Modèle III comme au Modèle I.

LABEL/CMD

Comme nous le verrons dans la section de ce manuel consacrée au BASIC, le BASIC étendu de DOSPLUS 3.5 permet l'adressage par étiquettes. En bref, ceci permet de remplacer les numéros de ligne habituellement utilisés pour les branchements (GOTO, GOSUB, etc.) par des étiquettes alphanumériques. Par exemple GOTO 2155 peut être remplacé par GOTO CALCULPAYS. Ces noms sont affectés en utilisant l'ordre NAME suivi du nom, au début de la ligne voulue. Par exemple le programme BASIC suivant :

```
10 print"(1) Saisie d' un nouvel article"
20 print"(2) Affichage de l' article actuel"
30 print"(3) Suppression d' un article perime"
40 print
50 input" Choix ";x
60 on x goto 1000, 2000, 3000
70 goto 10
1000 ....
*
2000 ....
*
3000 ....
*
```

pourrait être écrit avec les étiquettes :

```
10 print"(1) Saisie d' un nouvel article"
20 print"(2) Affichage de l' article actuel"
30 print"(3) Suppression d' un article perime"
40 print
50 input" Choix ";x
60 on x goto saisie, affichage, suppression
1000 name saisie
*
2000 name affichage
*
300 name suppression
*
```

Ce qui est bien agréable pour écrire un programme en BASIC. Malheureusement, la plupart des interpréteurs BASIC ne comprennent pas ce langage, et les programmes utilisant ce mode d'adressage écrits sous DOSPLUS 3.5 ne pourront pas fonctionner avec d'autres BASIC (y compris T-BASIC). Le programme LABEL/CMD est un decodeur d'étiquettes, c'est à dire qu'il remplace dans le deuxième exemple les étiquettes par les numéros de ligne correspondants, et les supprime, pour faire de ce programme un programme ordinaire.

Pour faire un programme normal à partir d'un programme avec étiquettes, taper :

```
LABEL nomfich1 nomfich2
```

où nomfich1 est le nom du fichier à convertir, et nomfich2 celui du nouveau programme en basic normal.

Si des noms de fichiers ne sont pas fournis au lancement de LABEL, celui-ci les demandera.

La transformation s'opère en deux étapes. La première consiste à examiner tout le programme pour construire une table de toutes les étiquettes et des numéros de ligne associés. La deuxième enlève toutes références aux étiquettes, qui sont remplacées par les numéros de ligne correspondants. Si une référence est faite à une étiquette non définie, celle-ci ne sera pas mo-

différent dans le texte du programme, et le message :

```
resolving line f:xxx undefined label : xxxxxxxx  
(transformation de la ligne xxx: étiquette inconnue : xxxxxxxx)
```

sera affiché, indiquant le numéro de la ligne où cette étiquette a été mentionnée, ainsi que son nom.

Après la fin du programme LABEL, le programme objet, décodé, peut être utilisé normalement.

Mot de passe du disque:

Le mot de passe de tous les disques fournis avec DOSPLUS 3.5 par Micro-Systems Software est "PASSWORD".

Il peut être demandé pour l'exécution de certaines fonctions globales telles que COPY, KILL, ATTRIB, PROT. Ce mot de passe peut aussi être substitué à tout moment au mot de passe individuel d'un fichier.

NOTE: Ceci devrait attirer votre attention sur l'importance du mot de passe qui est attribué au formatage d'un disque. S'il n'est pas fourni, c'est que le mot de passe de ce disque est vide, (tout autre mot sera considéré comme invalide). Si on veut protéger un fichier sur ce disque, il faut affecter un mot de passe non vide au disque. Si cela n'a pas été fait au formatage, utiliser PROT.

Mots de passe de fichiers.

Beaucoup des fichiers utilitaires du système de DOSPLUS sont protégés par mot de passe, pour vous éviter de les effacer accidentellement. Pour copier isolément un de ces fichiers, le mot de passe est le même que l'extension. Par exemple, le mot de passe pour SYS0/SYS est SYS, et pour BACKUP/CMD, ce sera CMD.

Introduction

Bienvenue au DOSPLUS 3.5. Ce Systeme d' Exploitation de Disques unique ete concu en pensant qu' un programme n' a pas besoin d' engendrer la confusion pour etre puissant. Pour ceux d' entre vous qui connaissent nos precedents systemes, DOSPLUS 3.5 sera une surprise. C' est peut-etre actuellement le systeme le plus independant des peripheriques qui ait jamais ete ecrit pour les TRS-80 Modeles I et III. Ce manuel est fait pour vous aider a connaitre DOSPLUS 3.5 et est ecrit (nous l' esperons) d' une facon agreable et claire. Il comprend les sections suivantes :

Le manuel operatoire : Doit vous permettre de comprendre les principes de fonctionnement du systeme et d' en saisir les differentes parties. Il concerne le nouvel utilisateur, definit aussi la syntaxe des ordres ainsi que le mode d' emploi de DOSPLUS.

La table des commandes (library) : Cette partie couvre les ordres systeme, c' est a dire ceux qui sont executes par le systeme lui-meme. Vous pourrez eliminer les fichiers de votre choix, mais en premiere approximation, un systeme minimum les possedera.

Les utilitaires de DOSPLUS : couvre les programmes utilitaires, qui sont destines a ameliorer et etendre les possibilites du systeme. Il est facile de les supprimer en cas de besoin, pour que l' utilisateur configure son systeme suivant ses besoins et la configuration de son materiel.

Le manuel BASIC disque : DOSPLUS 3.5 a ses propres BASIC disque. Un BASIC disque est un programme qui renforce les possibilites de celui qui est inclus dans la memoire morte du TRS-80. DOSPLUS en a deux : BASIC et TBASIC. Cette section decrira chacun d' eux avec ses particularites.

Le manuel technique DOSPLUS 3.5 : Cette derniere partie du manuel contient toutes les adresses de la RAM et de la ROM qui sont importantes pour ceux qui veulent des informations sur le SED. Elle comprend aussi une documentation sur les differents sous-programmes du systeme et leur fonctionnement. La lecture de cette partie du manuel est obligatoire pour le programmeur en langage machine qui veut interfacer ses programmes avec le systeme.

Nous esperons que DOSPLUS vous plaira et que vous aurez plaisir a l' utiliser. Il y a des differences entre DOSPLUS et les autres systemes, y compris ses versions anterieures. C' est pourquoi nous vous recommandons fortement la lecture du manuel avant de commencer a l' utiliser.

Syntaxe generale du manuel.

Ce document utilise une syntaxe assez uniforme pour que nous la decrivions ici. Le premier element est la maniere generale d' entrer les ordres.

Trois termes seront employes dans ce manuel en reponse aux demandes d' information du systeme (invites) : 'presser', 'taper' et 'entrer'.

'Presser' concerne une seule touche, indiquee dans le texte. 'taper' peut signifier soit la frappe d' une seule touche, soit de plusieurs; dans les deux cas, il vous sera indique ce que vous devez taper, caractere par caractere. 'entrer' est utilise quand l' utilisateur doit fournir une reponse basee sur un groupe de parametres valable pour cet ordre et les remplir suivant son choix. Nous vous demanderons d' entrer quelquechose quand ce sera a vous de faire le choix de ce qu' il faut taper. Le texte a 'entrer' doit toujours etre termine par 'ENTER'.

Par simplification, nous avons adopte certaines conventions de notation :

Lettres majuscules : Le mot doit etre tape tel-quel.

Minuscules : l' information a taper doit etre choisie par l' utilisateur parmi une liste de valeurs compatibles avec cet ordre.

<..> : indique que l' information encadree est facultative et peut etre entree ou non suivant la situation et le voeu de l' utilisateur.

(..) : indique la zone des parametres. Cette zone est composee des parametres qui modifieront l' action de l' ordre pour satisfaire les besoins de l' utilisateur. Les parentheses ne sont plus une exigence de DOSPLUS. La zone des parametres est maintenant definie autant par la position que par les separateurs.

H : indique que la valeur qui suit est hexadecimale. Elle est utilisee pour entrer fournir au systeme des donnees sous forme hexadecimale.

La forme generale de chaque commande est la commande elle-meme, suivie de la zone d' entree/-sortie, puis de la zone des parametres. La zone des parametres est facultative, ainsi que certaines parties de la zone des entree/sorties; elles peuvent etre ajoutees a la discretion de l' operateur.

Premiere utilisation

Si c' est la premiere fois que vous utilisez DOSPLUS 3.5, il est tres important d' en faire avant toute chose une copie de travail, pour ne pas risquer d' abimer l' original. Pour ce faire, prevoyez deux disques vierges, et effectuez pour chacun d' eux un BACKUP (voir BACKUP dans la section UTILITAIRES).

Generalites

Dans la section suivante de ce manuel, nous allons voir :

- (1) designation des lecteurs, fichiers et peripheriques
- (2) Entree des ordres.

Pour ceux qui sont interesses par la personnalisation de leur systeme, et ceux qui viennent d'un systeme precedent, nous exposons la nouvelle maniere de fixer la configuration du systeme dans le chapitre SYSTEM.

Designation des fichiers, lecteurs et peripheriques

Designation des fichiers

Nous allons couvrir ces articles dans l'ordre ci-dessus. D'abord les designations de fichiers: le seul moyen de copier des informations de facon permanente et de les retrouver par la suite est de les placer dans un 'fichier'. Un fichier est un groupe de donnees organisees enregistrees sur un disque. Ce peut etre un fichier de programme ou un fichier de donnees, mais toute donnee enregistree sur disque est enregistree dans un fichier, qui peut garder des donnees sur disque jusqu'a ce que vous soyez pret a les recuperer. Elles sont retrouvees par le nom du fichier dans lequel elles ont ete rangees, que ce nom ait ete donne a la creation du fichier, ou apres un changement de nom. De ce point de vue, les disques ne sont pas autre chose que des classeurs electroniques.

Une designation de fichier (un fichier pour abreger) est ecrit dans le format general suivant :

Nom/extension.mot de passe;d

'Nom' est une suite de 1 a 8 caracteres utilise pour definir le fichier auquel on accede. Il peut contenir tout caractere alphanumerique, ou n'importe quel caractere special fichier (qui seront decrits plus loin).

'/ext' est une extension facultative. Elle consiste en 1 a 3 caracteres qui precisent davantage le fichier dont il est question. Deux fichiers ayant le meme nom, mais des extensions differentes sont des fichiers differents. Les caracteres utilisables sont les memes que pour le nom.

'Mot de passe' est un mot de passe facultatif, compose de 1 a 8 caracteres. Il est utilise en liaison avec le niveau de protection que vous fixez avec l'ordre ATTRIB, pour controler les acces a votre fichier (par exemple pour eviter qu'un utilisateur ne le detruise par maladresse). Les caracteres utilisables sont les memes que ci-dessus.

';d' est un numero de lecteur facultatif, designant le lecteur sur lequel ce fichier est ou doit etre range. Nous parlerons plus tard de ces designations de lecteurs, mais si vous en fournissez un, il doit exister dans le systeme.

Caracteres autorises dans les noms de fichiers : Ce domaine differe des versions precedentes de DOSPLUS, et de tous les autres Systemes d'Exploitation Disques pour modele I et III. Tous les noms standards sont acceptes, mais nous avons ajoute des caracteres speciaux et relache les contraintes sur les noms de fichiers.

Sous DOSPLUS les noms de fichiers peuvent contenir les caracteres suivants :

- (1) Les lettres majuscules et minuscules (elles ne sont pas distinguees).
- (2) Les chiffres 0-9.
- (3) Les caracteres speciaux : \$, %, &, +, < et >.

de plus ces caracteres peuvent etre employes n'importe ou dans la designation du fichier. Contrairement aux regles anterieures.

ATTENTION : Bien que ceci donne davantage de souplesse a DOSPLUS cela peut provoquer des incompatibilites avec d'autres systemes. Ne pas l'oublier si vous faites des programmes que vous voulez transferer sur d'autres systemes.

Aucun espace n'est admis dans la designation des fichiers, DOSPLUS termine la lecture du nom de fichier au premier espace ou caractere illegal qu'il rencontre. Par exemple, RIEN BON/-DON serait pris pour RIEN.

Chaque partie du nom de fichier, en dehors du nom lui-meme, a un caractere particulier qui indique au systeme quelle partie de la designation arrive. Pour l'extension, c'est '//', pour le mot de passe, c'est '.', et pour le numero de lecteur, c'est ':'. Ces caracteres sont obligatoires, si l'on utilise la portion correspondant du nom de fichier, leur absence provoquerait une erreur.

Il n'est pas non plus possible de mettre un caractere 03 (fin de texte) ou 13 (retour chariot) dans un nom de fichier, car ils signalent tous les deux au SED la fin de la ligne d'ordres. Pour ecrire plusieurs ordres sur la meme ligne, il faut les separer par ';', qui signale un retour chariot implicite, mais n'arrete pas l'exploration de la ligne d'ordres. Pour plus de details, voir la section 'Caracteristiques incorporees', vers la fin de la section Operation.

Autres exemples et details sur la designation des fichiers: dans tout le systeme, et donc dans ce manuel, nous allons rencontrer deux types de fichiers: les fichiers programmes, et les fichiers de donnees. Le type de fichier n'est connu que de l'utilisateur qui l'a cree, et dans la majorite des cas, DOSPLUS n'a aucun moyen de le savoir. Il n'y a qu'une exception: les programmes en langage machine Z-80, qui sont executes directement si ils possedent l'extension '/cmd'. Si ce ne sont pas des programme Z-80, ils causeront une erreur, (voir 'LOAD').

Ainsi que nous venons de le dire, tous les fichiers ont une designation, par exemple :

PRIXVENT/HT1.FINANCES:1

Ce nom a quatre parties: le nom, l'extension, le mot de passe et le n° de lecteur. Il est correct.

PRIXACHAT ne le serait pas (trop long), ni TARIF?, (comporte un caractere interdit).

La deuxieme partie est l'extension. Elle est generalement utilisee pour indiquer le type d'information qui est rangee dans le fichier, et elle est separee du nom par une barre de fraction '//'. Voici quelques exemples d'extensions que vous pourrez trouver sur le disque DOSPLUS :

ASM fichier source assembleur.

BAS programme basic.

CIM programme image memoire (langage machine), pas necessairement executable.

CMD Code objet Z-80 executable. Souvent appele 'fichier commande'.

DAT fichier de donnees.

DVR fichier pilote. C' est un pilote de peripherique, qui permet de faire fonctionner differents materiels sous DOSPLUS. Ils peuvent etre mis en place par la commande ASSIGN.

FLT fichier filtre, contient les donnees necessaires pour la manipulation de caracteres entre les differents peripheriques.

PAT fichier de modification, contient les donnees necessaires pour que le programme PATCH modifie un fichier.

SYS fichier systeme. Fait vraiment partie du SED DOSPLUS.

TXT fichier texte code en ASCII. Est aussi utilise par la commande DO par default.

Les extensions sont facultatives, et vous pouvez utiliser celles de votre choix. Celles-ci ne sont que des exemples, et parfois des valeurs par default utilisees par des ordres DOSPLUS.

Bien que l' extension ne soit pas exiguee, elle est souvent utile pour preciser le contenu d' un fichier, par exemple :

PAYE/JAN

PAYE/FEV

PAYE/MAR

sont des fichiers differents. Seuls les programmes en code objet Z-80 doivent avoir l' extension /CMD pour etre executables directement au niveau commande du SED (si l' extension n' est pas precisee).

La troisieme partie du nom est le mot de passe (.FINANCES). Un mot de passe peut etre affecte a chaque fichier pour en controler l' acces. La description de l' ordre ATTRIB donne d' autres explications sur la protection des fichiers. Le mot de passe a une longueur de 1 a 8 caracteres. Quand vous affectez un mot de passe a un fichier, faites bien attention a pouvoir le retrouver (c' est le cle du coffre).

Le quatrieme element de la designation est la designation du lecteur. Il informe simplement DOSPLUS que l' on veut qu' il recherche ou cree le fichier sur le lecteur indique, sinon il le chercherait sur tous les lecteurs , reels ou fictifs. Pour l' instant il suffit de savoir que DOSPLUS est livre en standard avec les NOs de 1 a 7, qui peuvent etre affectes a des lecteurs materiels ou fictifs, et rebaptises. La designation de lecteur est egalement facultative.

De ces quatre elements, deux seulement individualisent un nom de fichier : le nom et l' extension. Deux fichiers ne peuvent avoir ces deux elements en commun et etre distingues par le mot de passe, tout essai dans ce sens se solderait par une erreur. Quand au NO de lecteur, il ne permet pas de distinguer deux fichiers du meme nom qui seraient sur deux disques installes successivement sur le meme lecteur.

En resume, rappelez-vous qu' un nom de fichier comporte necessairement le nom, et facultativement le reste.

Designation des peripheriques et des lecteurs.

DOSPLUS est prévu pour 8 peripheriques orientes caracteres (ils traitent l'information un caractere a la fois), et 8 peripheriques orientes fichiers, (il traitent un fichier) lecteurs de disques. En voici la liste :

Peripherique	nom par defaut	Classe
CLAVIER	KI	Entree
VIDEO	DO	Sortie
IMPRIMANTE	PR	Sortie
PORT SERIE	RS	Entree/sortie
DEFIN P.UTIL.	U1	Defini par utilisateur
id	U2	id id

Lecteurs

1er lecteur	0	Entree/sortie
2eme id	1	id
3 id	2	id
4 id	3	id
5 id	4	id
6 id	5	id
7 id	6	id
8 id	7	id

Un nom de peripherique est un groupe de deux caracteres qui lui sont affectes. Pour l'appeler a lui, il faut utiliser son nom.

Le premier groupe est compose de peripheriques orientes caractere, c'est-a-dire que les transactions entre eux et le systeme se font caractere par caractere. Pour le second groupe, des lecteurs de disque, les transactions se font par fichier, c'est a dire qu'ils traitent un fichier a la fois.

Ceci ne veut pas dire qu'un fichier ne peut pas fonctionner oriente caractere, il le peut. Il y a des fichiers speciaux qui peuvent fonctionner de cette facon, comme des canaux, mais un lecteur ne peut pas, c'est pourquoi les 8 derniers pilotes fonctionnent par fichier.

Vous pouvez donner a vos peripheriques le nom de votre choix, mais par souci de standardisation, nous vous suggerons d'utiliser leurs noms de defaut, qui est utilise dans ce document. Pour rebaptiser un pilote, utiliser l'ordre RENAME (voir plus loin), a ne pas confondre rebaptiser un lecteur avec le changement de l'ordre dans lequel les disques sont explores, qui est modifiable par CONFIG.

RESTRICTIONS :

Il n'est pas possible de donner le meme nom a deux pilotes differents. Pour inverser les noms de deux pilotes, il faut passer temporairement par un troisieme.

APPEL DES PILOTES :

On appelle un pilote par sa designation si c'est un pilote systeme, par le NO de son lecteur si c'est un pilote de lecteur. Ces noms comptent deux parties :

- (1) L'indicateur de type

(2) Le nom du peripherique.

L'indicateur de type est un seul caractere qui indique si l'on appelle un pilote systeme ou un pilote lecteur. Le premier est 'à', le deuxieme est 'l'; àKI est le clavier, li est le lecteur N° 1.

Le nom de peripherique est un groupe de deux caracteres non reserves, utilise pour référencer le pilote.

Chaque fois que l'on veut s'adresser a un pilote, on doit utiliser son nom, d'où l'importance de se rappeler ce nom si on le change. Pour voir la liste des noms en vigueur, utiliser la possibilite d'affichage des ordres JOIN ou FORCE.

Dans la plupart des cas, DOSPLUS permet d'utiliser les peripheriques systeme au lieu des fichiers, c'est une partie de ce que nous appelons l'indépendance des peripheriques.

RESUME DU CONTROLE DES PERIPHERIQUES SOUS DOSPLUS

Le principe de la commande des peripheriques sous DOSPLUS est en fait simple : les donnees n'ont que deux facons d'aller de A à B, à l'interieur du systeme :

- (1) octet par octet
- (2) fichier par fichier

Les deux modes ne sont pas les memes, et il vaut mieux se le rappeler pour eviter des ennuis.

Pour definir un trajet pour les donnees, on peut designer l'une des trois choses suivantes :

- (1) un nom de peripherique
- (2) un nom de fichier.
- (3) un nom de lecteur.

Les options 1 et 2 peuvent operer dans le mode caractere, les options 2 et 3 peuvent fonctionner en mode fichier. Le nom de fichier a la particularite de pouvoir fonctionner en mode caractere ou en mode fichier.

Explication detaillee de la ligne de commande

La ligne de commande est le moyen de communiquer avec DOSPLUS. Quand vous etes en mode commande du SED (vous rappelez-vous ce terme ?), vous pouvez entrer plusieurs ordres a DOSPLUS sur la meme ligne (jusqu' a 64 caracteres). Cette ligne de texte est la ligne de commande, chaque ordre comprend quatre zones : (1) l' ordre lui-meme, (2) le champ d' entree/sortie, (3) la zone des parametres et (4) la zone des parametres (facultative). Examinons tour a tour chacune d' elles.

L' ordre : c' est l' ordre DOSPLUS proprement dit, inscrit dans la table des ordres (library). Il appellera la partie du systeme que vous voulez utiliser, il doit etre la premiere donnee de la ligne (les blancs initiaux sont ignores), et doit etre suivi d' un terminateur ou un separateur, sinon DOSPLUS supposera que vous avez entre un nom de programme. Un terminateur est un retour chariot, entre en tapant <enter> apres l' ordre (ou un ; si il y a plusieurs ordres sur la ligne), par exemple LIB <enter>. Un separateur, o' un autre cote, pourrait etre un espace introduit apres le nom de l' ordre et avant les donnees suivantes.

Le champ d' entree/sortie : c' est l' espace suivant immediatement l' ordre. Il definit la direction du mouvement et quels fichiers ou peripheriques seront concernes. La zone d' E/S est elle-meme partagee en trois parties, chacune etant separee de ses voisins et delimiteurs par un espace. Ce sont : La zone source, la zone destination, et la zone du masque. Elles sont indiquees par les mots FROM, TO et USING, dans l' ordre. Ces mot delimiteurs peuvent etre omis, sauf si l' on veut changer l' ordre des zones. Par exemple :

```
COPY FROM TEST/CMD:0 TO TEST/CMD:1
```

aura le meme effet que :

```
COPY TEST/CMD:0 TEST/CMD:1
```

ou que :

```
COPY TO TEST/CMD:1 FROM TEST/CMD:0
```

mais est different de :

```
COPY TEST/CMD:1 TEST/CMD:0
```

La zone du masque est un champ qui contient une designation de fichier comprenant des caracteres de masquage. Il est utilise pour faire l' effet d' une commande globale sur plusieurs fichiers, ou sur tous. Les trois caracteres de masquage sont '*', '?', et '~'. Un point d' interrogation indique que le caractere qui est a cette position est a negliger. Un asterisque termine la portion du masque ou il reside et complete le champ avec des '~'. Par exemple :

```
T??T/B??
```

acceptera les fichiers TEST/EAS et TOUT/BAI, car le deuxieme et le troisieme caracteres ont ete remplaces par des '~'.

```
T*/*
```

accepterait tous les nom commençant par un T, sans distinction d' extension. Si l' on ne veut pas preciser de nom de fichier, il suffit de mettre un asterisque dans sa zone. Le point d' exclamation indique que les deux zones sont a remplir de points d' interrogation :

```
COPY !:0 !1
```

Provoquerait la recopie de tous les fichiers du disque 0 sur le disque 1, c'est un code qui peut être très utile. DOSPLUS reconnaît un masquage quand, (1) USING précède le masque, (2) le masque est dans sa zone réservée, (3) il contient des caractères de masquage.

La zone paramètres : Elle permet d'indiquer certains choix particuliers qui modifient l'effet de l'ordre. Il n'est pas nécessaire de la remplir si l'on ne veut utiliser que les valeurs par défaut des paramètres, ou si l'on veut ajouter des commentaires. La zone des paramètres est séparée de la zone des E/S par soit une virgule, soit une parenthèse gauche (ouverture). Dans la zone paramètres, ceux-ci doivent être séparés les uns des autres par un séparateur. Dans la zone E/S, un espace était nécessaire car une virgule aurait indiqué le début de la zone paramètres, mais à l'intérieur de celle-ci, on peut utiliser soit la virgule, soit l'espace. Pour ajouter des commentaires, il faut fermer la zone par une parenthèse droite pour indiquer à DOSPLUS que ce qui suit est un commentaire, et non la suite de la ligne d'ordres.

Dans la zone des paramètres, vous allez entrer des paramètres suivis par des expressions. Celles-ci indiqueront l'action que le paramètre va avoir sur l'ordre. Ce peut être l'une des trois choses suivantes :

(1) Une chaîne de caractères, comme pour un mot de passe ou le nom d'un disque, ou toute autre entrée qui demande une chaîne que le système doit utiliser, cette chaîne doit être mise entre guillemets.

(2) Une valeur, pour fournir à l'ordre une valeur numérique du paramètre, comme le nombre de ligne par page pour l'imprimante. Ces valeurs peuvent être fournies dans toute base numérique, à condition de la faire suivre de l'indicateur de base correct. Elles n'ont pas à être mises entre guillemets.

(3) Une alternative, qui sert à indiquer un état positif ou négatif du paramètre (actif ou non) les termes utilisés sont pour oui, YES ou ON indifféremment, comme NO ou OFF pour NON ou négatif. Ils peuvent être abrégés en 'Y' et 'N' et ne doivent pas être entre guillemets.

NOTE: Pour séparer le nom d'un paramètre de sa valeur, utiliser '='.

La zone commentaires permet de placer un commentaire facultatif à la fin d'un ordre exécutable. C'est utile pour les fichiers BUILD et DD, car cela permet de renseigner l'ordre en cours d'exécution. Par exemple, une ligne pourrait dire "CREATE ESSAI/DON (LRL=54) - Création d'un fichier d'index", pour permettre à l'utilisateur de savoir ce que fait la commande en cours.

Voici, sur un exemple, comment l'interprète des ordres examine une ligne d'ordres, soit la commande :

DIR :0 TO APR (ALPHA) - Impression d'une table alphabétique des fichiers.

DOSPLUS va lire la ligne de gauche à droite. Noter que la syntaxe de cette ligne est correcte. La zone E/S est séparée de l'ordre par un espace, et les autres zones sont également séparées les unes des autres par des espaces. La zone des paramètres commence par une parenthèse gauche, et finit par une parenthèse droite, qui commence la zone des commentaires.

DOSPLUS va lire l'ordre DIR, qui lui indique qu'il va falloir fournir une table de fichiers (directory). Les premiers caractères de la zone E/S n'étant pas un mot délimiteur (FROM, TO ou USING), le système suppose que nous utilisons la séquence d'ordre définie par défaut et prend :0 comme zone source. Il trouve le délimiteur TO, et sait ainsi que APR est la zone destination, qui dans ce cas est à sa position par défaut, et TO n'était pas nécessaire. Toutefois, en utilisant TO, nous nous affranchissons des positions par défaut. Cette phrase peut être placée n'importe où dans la ligne de commande, et, si le délimiteur est présent, sera identi-

fiee comme la zone destination.

Puis DOSPLUS trouve une parenthese gauche, qui lui indique que la zone E/S est terminee, et qu' il entre dans celle des parametres. A droite de la parenthese, le systeme trouve ALPHA, qui indique que nous voulons la liste dans l' ordre alphabetique. L' element suivant est la parenthese droite, qui indique la fin de la zone parametres, et qu' il ne faut pas tenir compte de ce qui suit.

Definition de termes -

Ce qui suit est une liste de termes employes par DOSPLUS accompagnes de leurs definitions et de leur traduction. Cela ne pretend pas etre un glossaire du systeme, mais simplement une liste de termes techniques employes frequemment. Pour les comprendre clairement, il peut etre utile pour le lecteur novice de lire les pages precedentes, et peut etre certains des articles suivants du manuel. Pour des lecteurs plus avertis, cela peut etre un aide-memoire utile pour la terminologie.

Terme	Definition
Nom de fichier	Reference a un fichier disque precis ne peut pas contenir de caracteres de masquage, mais peut contenir un designateur de lecteur optionnel.
Desig. de lecteur	deux points ':' suivi par un nom de lecteur de 1 ou 2 caracteres, designe un lecteur particulier. Ne peut etre utilise que pour des E/S de fichier. N' est pas un peripherique oriente caractere.
Design. periph.	un signe '@' suivi d' un nom de peripherique a 1 ou 2 caracteres. Designe l' une des 8 pilotes systeme. Ne peut etre utilise que pour des E/S par caractere et peut etre specifie pour demander un canal d' E/S.
Canal	Un canal est une voie orientee caractere d' E/S. Quand un canal est demande, cela indique que le mouvement des donnees se fera par caractere. Un canal peut etre un fichier ou un peripherique, mais pas un lecteur, sauf si celui-ci fait partie d' un nom de fichier.
Masque	Nom de fichier qui contient des caracteres de masquage, utilise pour effectuer des commandes globales sur des groupes de fichiers. Ne peut pas etre utilise comme canal, ne consiste que d' un nom et d' une extension, et peut etre utilise en conjonction avec un canal.
Parametre	zone de controle facultatif, qui permet de preciser COMMENT on veut qu' une instruction soit executee. Peut etre un choix (oui ou non), une valeur (taille d' un tampon, longueur de bloc, etc.). Si le parametre est un choix, sa simple mention l' active generalement ('Y' par default).
Separateur	Mot ou caractere utilise pour separer des delimitateurs, des canaux, des parametres, etc., dans la zone d' E/S, le separateur DOIT etre un espace; dans la zone parametres, ce peut etre un espace ou une virgule. Si vous mettez une virgule dans la zone d' E/S, DOSPLUS terminera celle-ci et commencera a lire des parametres. Les separateurs ne sont pas facultatifs, car les differentes parties de la zone doivent etre separees pour etre evaluees correctement.
Delimiteur	Designa une zone. C' est FROM, TO ou USING. Il indique une direction dans la zone d' E/S. Ces mots ne peuvent etre utilises comme noms de fichiers; ils doivent etre encadres par des separateurs et leur usage est facultatif. En fait vous ne vous en servirez que si vous voulez modifier l' ordre des differentes parties de la zone d' E/S.

Tout au long de ce manuel, nous allons utiliser ces termes. Comprenez combien ils peuvent être nouveaux pour vous, nous vous suggérons de relire cette section chaque fois que vous aurez un doute.

Liste des ordres de la librairie

Voici la liste des ordres de la librairie de DOSPLUS 3.5. Pour lancer un ordre, entrer son nom, suivi des parametres necessaires:

Ordre	Description	Page
APPEND	mettre deux peripheriques/fichiers l' un derriere l' autre	2/2
ASSIGN	installe un programme pilote	2/4
ATTRIB	modifie les attributs d' un fichier	2/6
AUTO	prepare une execution automatique d' ordres	2/9
BOOT	initialise le systeme	2/10
BREAK	controle la touche BREAK	2/10
BUILD	ecrit un fichier texte	2/11
CAT	affiche le catalogue des fichiers d' un disque	2/12
CLEAR	nettoie la memoire utilisateur ou un fichier	2/13
CLOCK	controle l' affichage de l' heure	2/14
CLS	efface l' ecran	2/14
CONFIG	Modifie la configuration du systeme	2/15
COPY	copie un periph/fichier sur un periph/fichier	2/21
CREATE	definit et alloue un fichier	2-24/1
DATE	met la date a jour ou l' affiche	2-24/3
DEBUG	active le moniteur systeme	2-24/3
DIR	affiche une liste detaillee des fichiers d' un disque	2/24
DO	execute une suite d'ordres	2/27
DUMP	enregistre sur disque le contenu d' une zone memoire	2/29
ERROR	affiche un message d' erreur detaille	2/30
FILTER	active un fichier filtre	2/31
FORCE	redirige l' E/S	2/33
FORMS	Modifie les parametres de l' imprimante	2/35
FREE	affiche l' espace libre d' un disque	2/37
I	Initialise un lecteur de disque	2/38
JOIN	relie deux peripheriques logiques	2/38
KILL	elimine un fichier ou un peripherique	2/41
LIE	affiche la presente liste	2/43
LOAD	charge un fichier disque en memoire	2/45
PAUSE	suspend l' execution	2/47
PROT	modifie l' etat de protection d' un disque	2/47
RENAME	rebaptise un fichier ou un peripherique	2/49
RESET	restaure un peripherique a son pilote par default	2/50
RS232	modifie les reglages du port serie	2/51
SCREEN	envoie le contenu de l' ecran a un peripherique	2/53
SYSTEM	adapte votre systeme d' exploitation	2/54
TIME	affiche l' heure ou la met a l' heure	2/57
VERIFY	lance la verification automatique des disques	2/58

APPEND

Cet ordre est utilise pour ajouter un fichier ou un peripherique a la suite d'un au- tre.

APPEND 'FROM' Periph1/fichier1 /TO/ periph2/fichier2 (param=choix,...)

Periph1/fichier1 est le systeme SOURCE, c' est celui qui va etre ajoute.

Periph2/fichier2 est le systeme destination, auquel le premier va etre ajoute.

Param=choix est le choix facultatif.

Les parametres sont :

(C)MD= Y/N ajoute au fichier destination en format execution directe (par ex un fichier /CMD)

(S)TRIP= Y/N recule d' un octet a partir de le fin du premier fichier.

L' ordre APPEND peut etre employe pour concatener facilement deux fichiers, en evitant d' avoir a ouvrir les deux, se positionner a la fin du premier, lire la source, etc.

La plupart des fichiers donnees laissent le soin d' entretenir leur fin de fichier au SED, dans ces cas le directory l' indique. C' est le cas des fichiers ecrits par BASIC et des programmes BASIC eux-memes, mais certains programmes marquent eux-memes leurs fin de fichier, comme SCRIPSIT, qui la signale par un octet a 0. Il est donc necessaire de supprimer cet octet, sinon il ne serait pas possible de charger la deuxieme partie du nouveau fichier, d' ou l' interet du parametre STRIP.

APPEND peut aussi etre utilise comme une sorte de MERGE dynamique. on peut ajouter deux programmes BASIC, enregistres en ASCII (SAVE"...",A), et charger le programme resultant. Les lignes ajoutees vont remplacer les anciennes si elles ont le meme numero.

APPEND a aussi la possibilite d' ajouter au fichier destination un fichier au format machine. Ce format est le format utilise pour ranger sur disque un programme en langage machine de telle facon qu' il puisse etre recharge en memoire aux bonnes adresses. Ce type de fichier contient des 'marqueurs de bloc' qui informent le SED du type de donnees qu' ils contiennent et de la place exacte ou ils doivent etre charges en memoire. Au rangement d' un fichier de donnees, DOSPLUS ne regarde pas le fichier, mais copie simplement ce qu' il trouve a l' adresse qui lui a ete indiquee. Au contraire, au chargement d' un fichier au format machine, DOSPLUS examine le fichier pour savoir ou il veut etre charge. Chaque bloc identifiant son contenu, DOSPLUS neglige les commentaires et ce qui y ressemble.

De plus, les quatre derniers octets d' un tel fichier sont ce que l' on appelle l' adresse de transfert, c' est a dire l' adresse a laquelle commencera l' execution du programme. Quand cette adresse est trouvee, l' execution commence immediatement, et il ne serait donc pas possible de concatener deux programmes en langage machine, car le premier commencerait a etre execute avant que le second soit charge. Pour eviter ce probleme, DOSPLUS, pour APPEND va ecraser ces quatre derniers octets, et l' adresse de transfert du dernier module sera utilisee.

Ajouter un peripherique a un fichier est essentiellement la meme chose que la copie de ce periph. sur le fichier (voir COPY), sauf que si vous ajoutez un periph a un fichier, celui-ci sera prolonge au lieu d' etre remplace.

Faites tres attention pour ajouter des peripheriques. Comme tout systeme ayant la meme souplesse, cela peut etre mal utilise et coincer la machine. Etudiez soigneusement votre logique

avant de le faire. Il faut toujours ajouter depuis un peripherique d'entree vers un peripherique de sortie. Si vous n'etes pas sur de ce que fait un peripherique particulier, utilisez FORCE ou JOIN pour vous renseigner.

Exemples :

```
APPEND FROM DONNEFI1 TO DONNEFI2  
APPEND DONNEFI1 DONNEFI2
```

Cet ordre ajoutera le contenu du fichier DONNEFI1 a la suite du fichier DONNEFI2.

```
APPEND NOUVEAU/BAS TO ANCIEN/BAS
```

Ajouterait le programme NOUVEAU au programme ANCIEN.

```
APPEND PATCH/CMD PROGRAM/CMD (CMD)  
APPEND PATCH/CMD PROGRAM/CMD (C)
```

Cet ordre va ajouter le module en langage machine PATCH/CMD a la suite du programme machine PROGRAM/CMD. A l'execution de ce dernier depuis le SED, les instructions du programme PATCH/CMD vont se fondre dans le programme et le modifier. C'est un moyen TRES efficace de modifier des programmes. Il suffit d'ecrire le module de modification et de l'ajouter au programme a modifier.

```
APPEND àKI DOCUFICH/TXT:1
```

Ajouterait tout ce qui serait tape au clavier a la suite du fichier DOCUFICH/TXT du lecteur 1. Ca permettrait d'ajouter de nouvelles instructions a la suite d'un fichier BUILD, par exemple (notez que BUILD a une facon superieure de le faire).

```
APPEND TO SERIAL/DAT:0 FROM àRS (STRIF)
```

```
APPEND TO SERIAL/DAT:0 FROM àRS,S
```

Cet ordre ouvrirait le fichier SERIAL/DAT sur le lecteur 0, pointerait la fin du fichier, reculerait d'un caractere pour supprimer un octet eventuel de fin de fichier, et ajouterait tout ce qui viendra de l'interface serie (a condition bien sur que celle-ci soit installee et activee).

En resume, APPEND n'affecte jamais le fichier ou peripherique source.

De plus, a moins que l'on ne prenne l'option /CMD, le complement sera toujours sauve sous forme de donnees. Les complements en langage machine doivent etre ajoutes avec l'option CMD. Il n'y a pas d'alternative.

ASSIGN

Cet ordre permet de mettre en place un programme pilote pour tout peripherique ou lecteur.

La syntaxe est :

ASSIGN (FROM) periph/lecteur TO nomfich

ASSIGN (FROM) periph1/lecteur1 (TO) periph2/lect2.

periph est le nom du periph ou lecteur pour qui on veut installer le pilote
(par ex. àKI, àDO, !2 etc.)

nomfich est le nom du programme pilote.

periph1/lect1 est le nom d'un quelconque peripherique ou lecteur dont les informations seront prise pour periph2/lect2.

L' instruction ASSIGN permet d' installer des pilotes non standards pour n' importe quel peripherique ou lecteur du systeme. Cela vous permet d' utiliser du materiel different de celui que fournit RADIO-SHACK (disques durs, 8 pouces, etc.)

Un pilote est simplement un programme en langage machine qui controle les E/S d' un peripherique particulier. Suivant la nature du materiel, le pilote peut aussi recevoir certains signaux de controle du peripherique pour en controler le fonctionnement, comme dans le cas d' un pilote d' imprimante serie. Ce pilote peut suspendre la sortie quand le peripherique signale 'pas pret', ou 'manque de papier'. D' autres pilotes peuvent controler des entrees comme le clavier, ou l' interface serie. Ils peuvent traiter toute donnee recue des ces peripheriques. On peut donc sans se tromper considerer un programme 'pilote' comme une interface entre un equipement materiel et le SED DOSPLUS.

L' ordre ASSIGN peut prendre deux formes differentes. L' une transfere le pilote du disque a la memoire, l' autre affecte simplement deux peripheriques au meme pilote. L' avantage de la seconde methode est de ne pas encombrer la memoire avec une deuxieme copie du pilote. Celui-ci doit toutefois y avoir ete mis au moins une fois pour que le peripherique puisse fonctionner.

DOSPLUS est livre en standard avec des pilotes pour les principaux peripheriques (àKI, àDO, àPR et les lecteurs de disques). En general, ces pilotes conviennent a une grande variete de cas, toutefois il peut arriver que l' un d' eux ne convienne pas a un equipement particulier, par exemple une imprimante qui utilise des symboles non standards, ou pour faire des E/S sur le port serie, ou pour un disque dur. Dans ces cas, il faut installer un pilote particulier pour le peripherique considere. Il en est fourni avec le systeme, d' autres peuvent etre ecrits pour atteindre ce but. La section technique de ce manuel fournit des indications sur la facon de faire.

Un programme pilote est range sur disque comme n' importe quel fichier, est appele par l' ordre ASSIGN en cas de besoin. Si vous ASSIGNEZ un pilote, vous devez fournir la designation du peripherique et celle du programme pilote. Les masques ne sont pas tolères, et chaque pilote doit etre installe independamment des autres. Il n' y a pas d' extension par defaut, mais nous suggerons "/DVR".

Si vous utilisez la premiere forme de ASSIGN, lecture du pilote sur disque, ce pilote sera charge en haut de memoire et se protegera en ajustant le pointeur du sommet de memoire. C' est en tout cas ce que font ceux que nous fournissons, et nous ne pouvons prejuger des autres. Les pilotes sont aussi totalement relogeables, c' est a dire qu' ils n' ont pas de place assignee, et peuvent etre charges dans n' importe quel ordre.

L' installation d' un pilote par ASSIGN annule tous les FORCE ou JOIN anterieurs, mais les filtres deja etablis restent actifs. Tous les pilotes actifs au moment ou la configuration est sauvegardee (voir SYSTEM) seront sauvegardes avec et restaures quand le fichier correspondant

sera execute.

Si d' autres peripheriques doivent utiliser le meme pilote, ce serait un gaspillage de memoire que de recharger ce pilote a chaque fois, en utilisant la premiere forme de ASSIGN. Pour l' eviter, utilisez la deuxieme forme, qui fait que seuls quelques octets sont ajoutes, pour tenir compte du nouveau peripherique.

Par exemple, si vous ASSIGNiez le lecteur 4 au pilote DISQDUR/DVR (Nous supposons que le lecteur 4 est un disque dur), et que vous vouliez utiliser 5 6 et 7egalement comme disques durs, il vous suffirait d' ASSIGNer chacun d' eux au lecteur 4 (ASSIGN :5 :4, ASSIGN :6 :4, etc.), ce qui les soumettrait au pilote deja installe pour le 4.

NOTE: Quand un pilote a ete affecte a plusieurs lecteurs de cette facon, tous les lecteurs partageront les memes reglages de CONFIG que le premier, y compris le NO du lecteur physique (voir CONFIG).

L' ASSIGNation d' un lecteur a lui-meme (ex ASSIGN :4 :4) reinstalle le pilote present defini pour ce periph/lect. dans le cas de periph caracteres, tout JOIN ou FORCE sera annule. Si ils ont ete elimines par KILL (voir KILL), ils seront reactives. Si ce lecteur ou periph n' a jamais eu de pilote affecte, une erreur sera signalee.

Chaque pilote fourni par Micro-Systems pour DOSPLUS est livre avec des instructions d' installation et d' utilisation.

Passage de parametres avec l' ordre ASSIGN

Certains pilotes peuvent necessiter la fourniture de parametres au lancement. Ce peut etre des designations de fichiers, ou autre chose. Ces parametres sont fournis comme pour n' importe quel ordre de la librairie. Exemple :

ASSIGN &KI KI/DVR FRASES/DAT

Permettrait d' entrer avec chaque touche un troncon de texte, range dans le fichier frases/dat apres avoir ete prepare par BUILD par exemple.

ATTRIB

Cet ordre permet de definir les modalites de protection d' un fichier.

La syntaxe est :

```
ATTRIB nomfich (param=exp...)
ATTRIB <using> masque (param=exp...)
```

nomfich est le nom du fichier dont on veut modifier les attributs.

masque est le masque qui designe un fichier ou un groupe de fichiers.

(param=exp...) est l' attribut que l' on veut modifier, et sa nouvelle valeur.

Les parametres d' ATTRIB sont :

(P)W= chaine	mot de passe du disque, necessaire pour les operations avec masque.
(A)CC= chaine	nouveau mot d' acces.
(U)PD= chaine	Nouveau mot de modification.
(P)ROT= valeur	Nouveau degre de protection.
(L)RL= valeur	Nouvelle longueur de bloc.
(I)NV= choix	Nouvel etat de visibilite.
(K)EEP= choix	Nouvel etat longueur max.
(M)OD= choix	Nouvel indicateur de modification.

L' ordre ATTRIB donne le controle total de la protection d' un fichier disque. Il est destine a modifier les autorisations d' acces a un fichier, mettre ou oter certains indicateurs que DOS- PLUS entretient pour chaque fichier, ou changer le mot de passe d' un fichier.

L' ordre ATTRIB opere dans deux modes, standard et global, dans le mode standard, on precise le nom du fichier apres l' ordre lui-meme, y compris les extensions, mots de passe et NO de le- cteur eventuels, puis la liste des parametres a modifier. Dans le mode global, on fournit le masque apres l' ordre, suivi de la liste des parametres. S' il y a au moins un fichier protege dans la liste, il faut fournir le mot du disque avec le parametre PW.

L' usage de ATTRIB couvre deux domaines, le premier concerne la protection, le deuxieme dif- ferents indicateurs.

Controle de la protection d' un fichier.

Un niveau de protection est inutile sans mot de passe. Si l' on n' en fournit pas, cela revient a en mettre un compose de 8 blancs. C' est la valeur par defaut, et il est le seul a pouvoir etre utilise.

Le mot d' acces fonctionne suivant le niveau de protection choisi. S' il en a ete affecte un, l' utilisateur doit le fournir pour acceder au fichier. Quand il l' a fourni, il ne peut acceder au fichier que jusqu' au niveau accorde. Toutefois, la connaissance du mot de mise a jour permet toute action sur le fichier. Ce dernier est donc a garder a l' abri des indiscre- tions, plus que les mots d' acces.

Il faut aussi se rappeler que sous DOSPLUS le mot de passe du disque peut toujours etre u- tilise a la place d' un mot de passe de fichier. Il est donc egalement a proteger.

Il peut aussi être modifié en utilisant l' instruction PROT.

Les niveaux de protection possibles sont les suivants :

Numero	Niveau
0	pas de protection.
1	kill, rename, écriture, lecture, execution.
2	rename, écriture, lecture, execution.
3	inutilisé à présent.
4	lecture, écriture, execution.
5	lecture, execution.
6	execution seule.
7	système, pas à la disposition de l' usager.

Le niveau 1 donne accès total au fichier.

Le niveau 2 permet tout sauf la destruction.

Le niveau 4 permet d' écrire dans le fichier, ou de le lire, mais pas de le rebaptiser ou le détruire.

Le niveau 5 ne permet que la lecture ou l' execution.

Le niveau 6 ne permet que l' execution. Si c' est un programme BASIC, il n' est soumis qu' à RUN.

Le niveau 7 n' est mis que par nous, pour éviter des destructions intempestives de programmes système.

Ces niveaux ne sont liés qu' au mot d' accès. Le mot de mise à jour donne accès total au fichier.

Modification des indicateurs et conditions.

L' autre fonction d' ATTRIB est la modification de certains indicateurs d' état : Longueur de bloc (LRL), Visibilité (INV), Modification (MOD) ou maintien de la longueur (KEEP).

LRL n' est là que pour indiquer quelle était la longueur de bloc initiale du fichier. Sous DOS- PLUS un fichier peut être ouvert avec une longueur de bloc quelconque à chaque fois. Toutefois, certains programmes exigent une longueur précise, et en basic il est souvent commode de pouvoir lire la longueur de bloc normale d' un fichier.

Pour l' utiliser mettre LRL= (1 à 256), le masque peut permettre de traiter plusieurs fichiers.

Pour rendre un fichier invisible, il suffit de mettre (INV=Y), et (INV=N) dans le cas contraire. Il peut être intéressant de rendre un fichier invisible, pour éviter d' encombrer l' écran à l' affichage du directory, ou par discrétion.

Quand un fichier est affecté de l' option (KEEP=Y), l' espace qui lui est attribué ne sera jamais diminué par DOSPLUS, même s' il raccourcit. Cela peut être intéressant pour des fichiers dont la longueur varie beaucoup, car autrement, ils pourraient se trouver coupés en plusieurs tronçons, plus lents à lire.

L' indicateur MOD=N supprime l' indicateur que DOSPLUS met à un fichier pour indiquer que l' on y a écrit depuis qu' il a été mis sur le disque. Il peut permettre de ne faire la copie de sécurité que des fichiers qui ont été modifiés.

Exemple :

ATTRIB paye/dat:2 (upd="moi",acc="tous",prot=5,inv)

ATTRIB paye/dat:2,u="moi",acc="tous",p=5,i

Auront le mem resultat : affecter au fichier PAYE/DAT du lecteur 1 le mot de modif. 'moi', le mot d' acces "tous", la protection ne permettra a ceux qui connaissent "tous" que de lire le fichier pour faire leur travail.

AUTO

Permet de faire executer une ligne d'ordres des l' initialisation.

=====

AUTO lect <ligne d'ordres>

lect est une designation de lecteur normale.

ligne d'ordres est une ligne d'ordres identique a ce que l' on taperait au clavier, mais rangee dans un fichier /txt, ecrite par exemple par l' ordre BUILD, ou un programme de traitement de textes, ou un fichier basic.

Parametres :

! AUTO invisible
x AUTO incassable

=====

Les ordres executes par AUTO peuvent etre des ordres de la librairie, un nom de programme, ou tout ordre que vous pourriez entrer vous-meme.

Il existe trois modes pour AUTO, le normal, qui affiche les instructions avant de les executer, l' invisible qui ne les affiche pas, mais les deux peuvent etre interdits en appuyant sur <break> a l' initialisation. Le deuxieme est precrit par ! apres l' ordre.

Le troisieme mode ne peut pas etre evite et se deroule obligatoirement. il peut etre combine avec le deuxieme, car il affiche les instructions.

Exemple :

AUTO !DIR

afficherait la table des matieres.

Conclusion :

L' un des aspects les plus utiles de AUTO est peut etre d' executer un 'fichier de configuration'. A l' initialisation, ces fichiers sont constitues par SYSTEM.

Si vous voulez mettre plusieurs ordres sur la meme ligne, separez les par des ';' (sans dépasser 63 caracteres au total).

AUTO

Permet de faire exécuter une ligne d'ordres dès l'initialisation.

AUTO lect <ligne d'ordres>

lect est une désignation de lecteur normale.

ligne d'ordres est une ligne d'ordres identique à ce que l'on taperait au clavier, mais rangée dans un fichier /txt, écrite par exemple par l'ordre BUILD, ou un programme de traitement de textes, ou un fichier basic.

Paramètres :

- ! AUTO invisible
 - x AUTO incassable
-

Les ordres exécutés par AUTO peuvent être des ordres de la librairie, un nom de programme, ou tout ordre que vous pourriez entrer vous-même.

Il existe trois modes pour AUTO, le normal, qui affiche les instructions avant de les exécuter, l'invisible qui ne les affiche pas, mais les deux peuvent être interdits en appuyant sur <break> à l'initialisation. Le deuxième est précrit par ! après l'ordre.

Le troisième mode ne peut pas être évité et se déroule obligatoirement. Il peut être combiné avec le deuxième, car il affiche les instructions.

Exemple :

AUTO !DIR

afficherait la table des matières.

Conclusion :

L'un des aspects les plus utiles de AUTO est peut-être d'exécuter un 'fichier de configuration'. À l'initialisation, ces fichiers sont constitués par SYSTEM.

Si vous voulez mettre plusieurs ordres sur la même ligne, séparez les par des ';' (sans dépasser 63 caractères au total).

BOOT

Permet de reinitialiser le systeme par programme.

=====

La syntaxe est :

BOOT

Il n' y a pas de parametres.

=====

Cet ordre est utile pour reinitialiser le systeme sous controle d' un programme. Le disque doit etre en place dans le lecteur 0, sous peine d' une erreur. Si la date et l heure sont actifs, ils vous seront demandes avant que AUTO n' execute les taches prevues. Si l' on veut le paraly- ser, il faut presser <enter> avant d' arriver a ces invites.

BREAK

Permet d' activer ou desactiver la touche BREAK.

=====

La syntaxe est :

BREAK <param>

param est un choix facultatif.

ON pour activer (default), OFF pour desactiver

=====

La commande BREAK permet de manipuler la touche BREAK. Il peut etre utile d' empecher un utili- sateur d' interrompre un programme, pour eviter des troubles dans son execution par exemple.

Il faut toutefois utiliser cet ordre avec precaution, car la touche BREAK est inutilisable pour tout le monde, jusqu' a sa reactivation, ou la reinitialisation du systeme.

BUILD

Permet de creer un fichier en ASCII sur disque, ou d'envoyer des caracteres ASCII a n'importe quel peripherique.

La syntaxe est :

BUILD periph/fich (param=choix)

periph/fich indique le support destinataire.

(param=choix) est un parametre facultatif qui modifie l'action de l'ordre :
APPEND=choix qui permet d'ajouter le texte a un fichier existant. (abrege : A)

Cet ordre permet de fournir des informations en ASCII a un peripherique ou un fichier. L'usage le plus frequent est pour faire des fichiers textes pour les divers ordres de DOSPLUS.

Cet ordre est l'un des plus utiles de la librairie, car il permet d'ecrire des listes d'instructions a executer a la mise en route, des listes de modifications a utiliser avec le programme PATCH, des listes de filtrage (Voir FILTER), etc...sans avoir a mettre en route un programme de traitement de texte, qui pourrait aussi faire ce travail, a condition de sauver le texte en ASCII.

BUILD affecte par default l'extension '/TXT', comme 'DO'. Au lancement (par ex. BUILD I- NIT:0) il affiche :

Enter text (63 chars/line) 'Entrez le texte (63 caracteres/ligne).

Vous tapez alors votre texte, puis quand la ligne est finie, pressez <enter>, vous passez a la ligne suivante. La pression sur <BREAK> provoque l'enregistrement sur disque, la ligne ou se trouve le curseur, si elle est inachevee, ne sera pas terminee par un retour chariot, et donc probablement pas executee.

Il est possible d'utiliser l'extension APPEND ou A pour ajouter du texte au bout d'un fichier existant.

Exemple :

```
BUILD INIT:0 <ENTER>
Enter text (63 chars/line)
FORMS,W=60,L=60 <ENTER>
.L' imprimante est reglee.
CAT :1 */BAS <ENTER>
BASIC MENU/BAS:1-F:3-M:65000 <ENTER>
<BREAK>
```

Ceci construirait un fichier appele INIT/TXT sur le lecteur 0, qui pourrait etre utilise par l'instruction 'DO INIT, et executerait les instructions qu'il contient : 60 lignes de 60 caracteres par page pour l'imprimante, affichage du catalogue des programmes /BAS du disque 1, et chargement du programme 'MENU/BAS' du lecteur 1, avec trois fichiers simultanes, et une reservation d'espace au dessus de l'adresse 65000.

Il est possible d'ecrire des lignes de commentaires, en les faisant commencer par '.,.

CAT

Cet ordre provoque l'affichage du catalogue d'un disque.

La syntaxe est :

CAT <FROM> lect <TO> fichier/periph <USING> masque (param=expr...)

De ces designations ont la signification habituelle.

Les parametres sont :

(S)YSTEM = choix affiche aussi les fichiers systeme
(I)NVIS = choix id invisibles
(K)ILL = choix id les noms de fichiers detruits non remplaces.

L'affichage des noms des fichiers se fait sur 4 colonnes. La sortie se fait par default sur l'ecran, mais peut se faire sur un autre peripherique (utiliser àPR ou àRS par ex.).

L'option masque permet de trier les fichiers à afficher, suivant le masque choisi.

L'usage de ces deux options necessite la designation du lecteur, ou de TO et USING explicitement.

CLEAR

Permet de remplir un fichier ou une zone memoire avec une paire d'octets choisie par l'utilisateur.

La syntaxe est :

CLEAR nomfich (param=expr,...)

Nomfich est facultatif et n'a pas a etre utilise pour une zone memoire.

Les parametres peuvent etre :

(S)TART = expr. Adresse debut de zone memoire.

(E)ND = expr. Adresse fin de zone memoire.

(D)ATA = expr. Donnee a ecrire.

Les deux modes de fonctionnement (memoire ou fichier) s'excluent mutuellement, et certains parametres ne fonctionnent que dans le mode ad-hoc.

Le parametre START permet de definir l'adresse memoire ou commence le nettoyage. Par defaut il est de 5C00H (23552), valeur en dessous de laquelle il n'est pas possible de descendre.

Le parametre DATA permet de definir 2 octets qui seront ecris alternativement dans l'espace defini, si vous n'en fournissez qu'un, il sera recopie dans le deuxieme.

Le parametre END vous permet d'indiquer la fin de la zone memoire a nettoyer. Sa valeur par defaut est le haut de la memoire non reservee (MEMORY), il n'est pas possible d'aller au dela.

Exemples :

CLEAR

Remplira toute la memoire utilisateur entre 5C00H et MEMORY avec des 0.

CLEAR (6000H,7000H,data=6CAAH)

remplira la zone 6000H a 7000H avec les octets 6CH et AAH alternativement.

CLEAR,6000H,7000H,d=6CAAH

aura le meme effet.

CLEAR paye/jul:1,D=65

remplirait le fichier paye/jul du lecteur 1 avec des 'A'.

Ne pas oublier que si vous envoyez le dernier ordre, le fichier est perdu.

CLOCK

Permet de faire afficher l'heure de la pendule interne.

Syntaxe :

CLOCK choix

choix est une alternative (ON/OFF), mettant EN ou HORS service cette pendule.

La mise à l'heure se fait par l'ordre TIME. A la mise en route, DOSPLUS essaie de lire l'heure en mémoire, si l'invite de mise à l'heure a été supprimée à la mise en route. Si cette lecture donne un résultat vraisemblable, elle sera prise, sinon l'heure commencera à 0.

Quand l'heure passe 23:59:59, elle est remise à 0 et la date avance d'un jour.

CLS

Vide l'écran et repasse en vidéo normale.

Syntaxe : CLS

CONFIG

Permet de configurer DOSPLUS pour l'utilisation de lecteurs non standards.

Syntaxe :

CONFIG lecteur (param=exp...)

lecteur est le numero du lecteur qui doit etre configure.

(param=exp...) est le parametre facultatif.

Les parametres sont :

Disques souples :

(W)P= choix active la protection ecriture logicielle.
(M)D= choix indique le delai de mise en vitesse moteur.
HL= choix delai de chargement des tetes.
(S)TEP=valeur temps piste a piste des tetes.
(SK)IP= choix mode double saut de piste.
(SIZ)E= valeur indique le diametre physique du disque.
(SID)ES=choix indique le nombre de surfaces actives.
(P)D= valeur indique le lecteur materiel concerne par cette designation de lecteur.

Disques durs :

(SIZ)E= valeur indique la taille de disque.
(SID)ES=valeur indique le nombre de surfaces actives.
(W)P= choix protection ecriture logiciel.
(S)TEP=valeur temps piste a piste.
(H)O= valeur indique le decalage de N0 de tete.
(C)O= valeur decalage de N0 de cylindre.
(T)S= valeur Nombre de secteur par piste.
(P)D= valeur N0 du lecteur physique concerne par le designation.

L'ordre CONFIG vous permet de configurer DOSPLUS pour fonctionner correctement avec toutes sortes de lecteurs, et d'en regler tous les parametres. C'est le pilote qui interprete chaque parametre et agit en consequence.

Utiliser CONFIG peut etre aussi simple ou difficile que vous le desirez. Si vous utilisez le materiel standard de RADIO-SHACK, vous n'en avez pas besoin, sauf pour modifier l'ordre d'exploration des lecteurs, ou modifier l'ordre habituel des choses d'une maniere quelconque.

Nous allons decrire le fonctionnement des cet ordre, d'abord pour les disques souples, puis pour les durs.

Disques souples.

Le premier pas est d'afficher les réglages présents. Il suffit de taper :
CONFIG <enter>

Vous devriez voir apparaître quelque chose comme :

```
$00 :0 Floppy,Dden,Size=5,Sides=1,Step=3,PD=0,MD
$01 :1 Floppy,Dden,Size=5,Sides=1,Step=3,PD=0,MD
$02 :2 Floppy,Dden,Size=5,Sides=1,Step=3,PD=0,MD
$03 :3 Floppy,Dden,Size=5,Sides=1,Step=3,PD=0,MD
$04 :4 NIL
$05 :5 NIL
$06 :6 NIL
$07 :7 NIL
```

Qui constitue le réglage standard de DOSPLUS 3.5 par défaut, sauf dans certains cas particuliers.

Le premier élément est le NO du périphérique lecteur. Vous pouvez voir qu'il y en a 8, numérotés de 0 à 7. Ils peuvent être définis à volonté, tant qu'il n'y a pas plus de 4 lecteurs de disques souples et 4 de disques durs. Plusieurs désignations peuvent concerner le même lecteur physique.

Le deuxième élément de la ligne est le NO du lecteur physique. Il n'a aucun lien avec la façon dont les lecteurs sont explorés, qui est celui des désignations, ni avec aucun élément du fonctionnement des lecteurs. Ceux-ci peuvent d'ailleurs être modifiés par RENAME. La seule restriction est que l'on ne peut avoir deux lecteurs avec la même désignation (voir RENAME et Désignation des fichiers et périphériques).

paramètres disques souples

Floppy

Support souple. Ce paramètre indique que le lecteur auquel correspond cette ligne de configuration est actuellement défini comme un disque souple. (ndt : bien que cela puisse sembler curieux, on peut avoir un disque dur signalé dans la liste de CONFIG sans avoir réellement de disque dur). Il est contrôlé par le programme pilote, et ne peut être modifié par l'utilisateur que si il change le pilote installé pour ce périphérique. Ce peut être réalisé par l'ordre ASSIGN).

Dden ou Sden

Densité d'enregistrement. Indique si le disque considéré est en double ou simple densité. Ce paramètre n'est pas modifiable. DOSPLUS 3.5 reconnaît la densité d'enregistrement d'un disque et en tient compte. Ce paramètre est là pour votre information.

Size

Taille physique du disque. Ce paramètre affiche et vous permet de configurer la dimension physique du disque utilisé (5 ou 8 pouces). À noter qu'il faut un pilote spécial pour utiliser les lecteurs 8 pouces.

Ce qui veut dire que pour utiliser un disque 8 pouces, ou tout autre disque non standard, il faut d'abord ASSIGNER le pilote convenable, puis utiliser CONFIG pour régler les paramètres du lecteur. Comme on utilise un disque 8 pouces, on mettrait : Size=8.

Exemple

```
CONFIG :2 (Size=8)
```

CONFIG :AA (size=5)
CONFIG :3,SIZ=8

Sides

Nombre de faces. Ce parametre vous permet de configurer DOSPLUS 3.5 pour utiliser des lecteurs double face. La creation d'un disque double face est en realite realisee par l'utilitaire FORMAT. Quand il demande les informations sur le disque, l'une des questions est :

'Single ou Double sided?' (Simple ou Double densite?)

Si vous repondez par 'D', FORMAT fera un disque double face (materiel permettant). A chaque fois, FORMAT initialise une table de donnees sur ce disque qui permet a DOSPLUS de reconnaitre le type du disque. C'est la table de controle du disque ou DCT. L'un des elements de cette table concerne le nombre de faces du disque, et est exploite au premier acces de DOSPLUS a ce disque. Cependant si vous le changez, il faut que DOSPLUS en soit informe, ce qui peut se faire de deux manieres :

Soit utiliser l'ordre I qui initialise le lecteur, pour qu'au prochain acces DOSPLUS sache qu'il doit relire la DCT car quelquechose a change. Soit inserer ce parametre dans CONFIG.

Il faut donc faire quelquechose chaque fois que l'on change de type de disque pour en informer le systeme.

Exemple :

CONFIG :2 (sides=2)
CONFIG AA,SIZ=1

Step

Vitesse de la tete. Ce parametre peut etre modifie. Il affiche le temps de deplacement des tetes que DOSPLUS utilise pour les different lecteurs du systeme. La valeur n'est pas un temps exact, mais une valeur relative que DOSPLUS interprete. La correspondance est :

Valeur	temps de piste a piste
0	6 millisecondes
1	12 dd
2	20 dd
3	30 dd(Double densite)
4	40 dd (Simple dd)

DOSPLUS regle ce temps a deux endroits. Le premier, controle par SYSTEM est la valeur par defaut. Elle est prise a l'initialisation. Toutefois, dans votre systeme, certains lecteurs ne peuvent pas aller aussi vite que d'autres. Il est donc pour celles-ci utiliser la deuxieme methode et ajuster cette vitesse pour chaque lecteur.

Quand vous employez CONFIG pour ajuster cette vitesse individuellement, il est commode de l'enregistrer dans un fichier de configuration (voir SYSTEM), et d'executer ce fichier a l'initialisation.

PD

Lecteur physique. Ce parametre affiche le lecteur physique reel auquel s'adresse un pilote de disque particulier. Vous pouvez le modifier pour indiquer a quel lecteur materiel un pilote doit s'adresser. plusieurs pilotes peuvent s'adresser a un meme lecteur materiel.

Ce parametre peut etre utilise pour modifier l'ordre de priorite des lecteurs, si l'ordre naturel de 0 a 7 ne vous convient pas. Il suffit de donner a PD la valeur souhaitee dans la liste des parametres CONFIG.

Exemple :

CONFIG :2 (PD=1)

CONFIG :1 (PD=2)

WP

Protection ecriture par logiciel. Ce parametre n' apparait pas dans notre exemple, car il n' etait pas actif, sinon il apparaitrait immediatement apres le NO physique. Son interet est de pouvoir etre mis et ote plus facilement qu' une protection materielle, tout en ayant le meme effet.

MD

Motor on delay (delai de mise en route moteur). Est destine a configurer DOSPLUS pour les lecteurs qui ne lancent le moteur des lecteurs que quand c' est necessaire, ou pour les lecteurs qui gardent les moteurs en route en permanence. Ces derniers sont principalement les 8 pouces, pour lesquels il est inutile de perdre du temps a chaque selection, alors que les autres sont des 5 pouces. Ce parametre est actif par default.

HL

Head Load delay (delai de chargement de tete). Ce parametre est destine a tenir compte des lecteurs qui ne gardent pas la tete de lect/ecrit. chargee (au contact du disque) en permanence, des 8 pouces en general. Certains lecteurs gardent les tetes chargees en permanence, ce qui est frequent pour les 5 pouces. Parmi les premiers, certains chargent au lancement des moteurs, et comme il est deja prevu un delai, il est inutile d' en ajouter un autre. Par contre d' autres ne la chargent qu' a la selection, et la il faut prevoir un delai pour que la tete ait le temps de se stabiliser avant de lire. La valeur par default est OFF.

Skip

Saut de piste double. Ce parametre permet de lire un disque 40 pistes dans un lecteur 80 pistes. A noter qu' il ne faut pas ecrire sur ce disque dans ce lecteur, la largeur de la tete etant plus faible, la piste porterait au milieu la nouvelle inscription, l' ancienne resterait sur les bords, et le disque serait inutilisable dans un lecteur 40 (ou 35) pistes. Ceci inclut l' utilisation de ce disque pour un 'BACKUP' ou 'COPY', DOSPLUS mettant a jour certaines donnees sur le disque origine, il deviendrait inutilisable, a moins d' etre protege en ecriture. A l' utilisation de SKIP, l' absence de parametre equivaut a un 'Y' (oui). Exemple :

CONFIG :2 (SKIP=Y)

CONFIG :2 (SLIP=N)

CONFIG :2,SK=Y

CONFIG :2,SK

Si nous devons regler tous les parametres possibles pour l' un des lecteurs de la liste ci-dessus, leur etat aurait l' allure ci-dessous :

```
$00 :0 Floppy,Dden,Size=5,Sides=1,Step=3,PD=0,MD
$01 :1 Floppy,Dden,Size=5,Sides=1,Step=3,PD=1,MD
$02 :2 Floppy,Dden,Size=5,Sides=1,Step=3,PD=2,WP,MD,Skip
$03 :0 Floppy,Dden,Size=5,Sides=1,Step=3,PD=3,MD
$04 :4 NIL
$05 :5 NIL
$06 :6 NIL
$07 :7 NIL
```

Le lecteur 2 a toutes les options prises.

Disques durs.

La premiere chose a faire pour CONFIGurer des disques durs est d' ASSIGNer au lecteur considere le pilote convenable. Des renseignements plus detaillés sont fournis dans la section Pilotes et Filtres. En ce qui concerne cette partie du manuel, nous supposerons que le pilote est installé, et ne nous occuperons que des parametres (ceux qui sont differents des disques souples).

Hard

Support rigide. Indique que le lecteur logique est considere comme un disque dur. Son role n' est que d' information.

Fix ou Rem

Disques fixes ou amovibles. Indique si le pilote installé peut travailler avec des disques fixes seulement ou aussi avec des disque amovibles. Dependand du pilote, il n' est la que pour votre information.

Size (Taille)

Diametres des disques. Permet de configurer DOSPLUS suivant la taille des disques. Certains pilotes peuvent fonctionner avec differents lecteurs d' un meme fabricant en modifiant simplement les réglages de ce lecteur. Les diametres sont 5 ou 8 pouces, le premier étant de loin le plus frequent.

Sides (faces)

Nombre de surfaces. Permet d' indiquer au pilote le nombre de surfaces a gerer. C' est exactement le double du nombre de disques. Il existe ici quelques restrictions, car on ne peut avoir plus de 256 secteurs dans un cylindre. Pour determiner le nombre de secteurs d' un cylindre, multipliez le nombre de surfaces par le nombre de secteurs par piste. Par exemple, si vous avez une taille de piste (TS) de 32 secteurs, le nombre de disques maximum que peut supporter le systeme est de 4 ($4 \times 2 \times 32 = 256$). Dans la plupart des cas vous ne serez pas concernes, et il vous suffira de fixer le nombre de faces egal au double du nombre de disques.

Step (pas)

temps de saut de piste. permet d' indiquer ce temps pour ce lecteur. c' est une valeur relative de 0 a 255, qui n'a aucun lien direct avec sa valeur reelle, et depend du pilote utilise (voir sa notice).

PD

Lecteur physique. Permet de definir le lecteur physique concerne par une designation de lecteur logique. Ce parametre permet entre autres choses de diviser un lecteur physique (souple ou dur) en plusieurs lecteurs logiques.

CO

Decalage de cylindre. Ce parametre est une partie de ce qui permet de definir quelle zone du disque dur une designation de lecteur va concerner. Si par exemple vous voulez partager en deux un lecteur de 230 cylindres, vous alloqueriez a chaque lecteur logique 115 cylindres. Pour que chaque designation de lecteur ait une zone distincte, il faudrait utiliser le parametre CO pour dire a l' une d' elles qu' elle commence au cylindre 115.

HO

Decalage de tete. Permet de partager un lecteur dur suivant les numeros de surfaces, au lieu des numeros de cylindre. Il peut etre employe en combinaison avec le precedent, et permet le cas echeant de contourner le probleme d' un trop grand nombre de surfaces, ou de secteurs par cylindre. Ex CONFIG :4 (HO=3).

TS

Taille de disque. Permet d'informer DOSPLUS du nombre de secteurs par piste (pas par cylindre). Ce parametre est a prendre dans la documentation du lecteur. Voici un exemple concernant un lecteur de 203 cylindres, 3 disques, 10 Megaoctets. Il est partage en 3 volumes en utilisant CO. Le systeme comporte aussi trois disques souples, et deux lecteurs logiques sont inutilises.

```
$00 :4 Hard,Fix,Size=5,Sides=6,Step=6,PD=0,CO=0,H0=0,TS=32
$01 :5 Hard,Fix,Size=5,Sides=6,Step=6,PD=0,CO=115,H0=0,TS=32
$02 :6 Hard,Fix,Size=5,Sides=6,Step=6,PD=0,CO=172,H0=0,TS=32
$03 :2 Floppy,Dden,Size=5,Sides=1,Step=0,PD=2,MD
$04 :1 Floppy,Sden,Size=5,Sides=1,Step=0,PD=1,MD
$05 :0 Floppy,Dden,Size=5,Sides=1,Step=0,PD=0,MD
$06 :A NIL
$07 :B NIL
```

La sequence complete a executer serait :

Utiliser ASSIGN pour mettre en place le pilote et activer ce peripherique.

Utiliser CONFIG pour regler les parametres pour chaque volume.

Utiliser le formatteur de disques durs fourni avec le pilote de disques durs pour formater le disque dur. Le mode d'emploi est fourni avec le pilote particulier.

Utiliser SYSGEN pour y recopier les fichier SYSteme.

Copier tous les fichiers utilisateurs sur le disque dur.

Utiliser ASSIGN pour enregistrer les operations que vous venez d'executer sur le lecteur systeme, ce qui passerait le controle au disque dur. Vous pouvez ensuite rebaptiser les lecteurs a votre guise.

Peut etre ce modele vous sera-t-il de quelque utilite.

Conclusion.

Ne pas oublier qu'aucune modification faite par CONFIG n'est permanente, pour qu'elle le devienne, il faut utiliser SYSTEM pour ecrire un fichier, quand tous les parametres sont ce que vous voulez. par la suite, l'execution de ce fichier permettra de reconstruire cette configuration.

Ne soyez pas intimide par CONFIG et les disques durs. Avec chaque pilote que fournit Micro-Systems Software, sera fourni en principe un fichier JCL qui installera votre systeme sous differentes configurations standard. Vous n'aurez a transpirer que si vous definissez une configuration non standard.

COPY

Permet de copier des données d' un endroit à l' autre du système.

Syntaxe :

- 1.COPY <FROM> fich/periph <TO> fich/periph (param=expr...)
- 2.COPY <FROM> fich <TO> fich/periph (param=expr...)
- 3.COPY <FROM> lecteur <TO> lecteur <USING> masque (param=expr...)

Les parametres sont :

- (D)PW="chaîne" mot de passe du disque destination.
- (E)CHO=choix Affiche le nom des fichiers au fur et à mesure.
- (I)NVIS=choix copie aussi les fichiers invisibles.
- (M)OD=choix Copie basée sur l' indicateur de modif.
- (Q)VER=choix Questionne avant d' écraser le fichier du même nom.
- (P)ROMPT=choix Demande le changement de disques (copie à un seul lecteur).
- (Q)UERY=choix Questionne avant de copier.
- (SP)W="chaîne" Mot de passe du disque source.
- (T)INY=choix Copie secteur par secteur.
- (N)EW=choix Ne copie que les fichiers qui n' existent pas sur le disque destination.
- (O)LD=choix Ne copie que les fichiers déjà existant sur le disque destination.

Pour bien comprendre le fonctionnement de COPY, il faut savoir que les périphériques au sens large, c' est à dire y compris les disques, et les fichiers qu' ils contiennent, se divisent en deux catégories, suivant qu' ils émettent ou reçoivent des données : Le clavier est un périphérique émetteur, et ne peut rien recevoir, inversement l' écran ne peut fournir de données. Il est donc possible de définir la catégorie des périphériques émetteurs, et celle des périphériques récepteurs. L' imprimante et la visualisation sont dans celle-ci, alors que le clavier est dans celle-là.

Certains périphériques peuvent appartenir aux deux : un fichier, ou un lecteur, ou l' interface série (RS232) peuvent aussi bien émettre que recevoir.

Dans l' expression de l' ordre, nous trouverons toujours dans l' ordre d' écriture, après COPY, un nom de périphérique émetteur, un nom de récepteur, suivis éventuellement d' un ou de plusieurs paramètres destinés à indiquer une certaine façon de faire le travail.

L' ordre COPY fonctionne sous trois modes :

- (1) D' un périph/fichier à un autre, octet par octet.
- (2) D' un fichier à un autre, fichier par fichier.
- (3) D' un disque à l' autre, fichier par fichier.

l' encadré Syntaxe donne des exemples de chacun de ces trois modes. Nous allons maintenant les examiner plus en détail.

Mode 1

Dans ce mode appelé aussi "copie périphérique", nous copions :

- * D' un périphérique dans un fichier.
- * D' un périphérique à un autre.
- * D' un fichier à un périphérique.

Il y a donc au moins un périphérique en cause dans ce mode. C' est celui qui permet par exemple de

transmettre des données entre ordinateurs, avec :

COPY àRS recep/TXT
ou
COPY envoi/txt àRS

Mode 2

Dans ce mode, appelé aussi 'copie par fichier', nous copions :

- * D' un fichier a un autre, avec changement de nom.
- * D' un fichier a un autre, sans changement de nom.

Par exemple :

COPY essai:0 to essai:1
ou
COPY essai:0 :1

La première rédaction est évidemment obligatoire pour changer le nom du fichier, le deuxième n' étant pas obligatoirement identique au premier.

Note technique : Dans ce mode, nous mettons en jeu le ' grand tampon par défaut, c'est-à-dire que DOSPLUS utilise toute la mémoire disponible pour faire la copie. Si nous voulons conserver un programme qui y réside, il nous faudra préciser l' option TINY, ou la copie n' utilise que la place d' un secteur, dans la zone du SED, mais prend plus de temps.

Pour copier un fichier dans une autre zone du disque, il faudra changer le nom, le même disque ne pouvant accepter deux fichiers du même nom.

Si la copie doit se faire avec un seul lecteur, utilisez le paramètre PROMPT, qui vous demandera l' échange des disques en temps utile.

Mode 3

Dans ce mode, aussi appelé 'copie par masque', nous copions entre deux lecteurs, en utilisant un masque. Il est donc préférable ou nécessaire, si la copie met en jeu un périphérique, ou un seul fichier, utiliser un autre mode.

Un bon exemple d' utilisation du mode 3 serait de recopier tous les fichiers d' un disque sur un autre (les fichiers système ne peuvent en aucun cas être copiés par cet ordre, voir SYSGEN). La rédaction pourrait être l' une des suivantes :

```
COPY !:0 :1
(tout copier d' un lecteur sur l' autre)
COPY :0 :1 !
(copier du lecteur 0 au lecteur 1 en utilisant tout)
COPY :0 :1 <using> */*
(copier du lecteur 0 au lecteur 1 en utilisant tout)
COPY USING ! :0 TO :1 FROM :0
(copier en utilisant tout, sur le disque 0 du disque 1)
COPY to :0 !:0
(copier sur le disque 1 tout le disque 0)
COPY :0 :1
(copier du disque 0 sur le disque 1)
```

Qui auraient toutes le même effet. La phrase entre parenthèses sous chaque instruction est là pour essayer d' expliquer le fonctionnement de l' ordre.

C' est au cours de ces copies avec masque que la plupart des parametres vont entrer en jeu, C' est ce que nous allons voir maintenant.

Une copie avec masque ne copiera pas les fichiers invisibles sans y etre expressément invite. Ceci devient particulierement important lorsque l' on veut preparer un disque non standard avec SYSGEN et COPY.

De meme les noms des fichiers copies ne sont affiches que si le parametre 'ECHO' le demande.

L' utilisation de MOD evite la copie de fichiers qui n' ont pas ete modifies. Cela peut faire gagner beaucoup de temps pour la mise a jour d' un disque de fichiers de donnees. L' indicateur de modification est mis par le systeme chaque fois que l' on modifie un fichier, c' est-a-dire quand l' on y ecrit. Il permet donc de distinguer les fichiers qui ont ete modifies depuis qu' ils ont ete ecrits sur le disque.

Le parametre KILL provoque la destruction du fichier copie sur le disque source apres la copie.

QUERY oblige le systeme a demander la permission de copier chaque fichier.

OVER oblige le systeme a demander la permission avant la copie, seulement si le fichier existe deja. Il est souvent prudent de l' utiliser.

Si l'un de ces deux derniers parametres est actif, avant de copier chaque fichier correspondant au masque, son nom sera affiche, suivi de 'Copy ?'. La reponse 'Y' et <enter> signifiera l' acceptation, toute autre provoquera le saut au suivant.

Lors d' une copie avec masque, il est frequent que certains fichiers soient proteges par un mot de passe. Il n' est pas possible de copier un tel fichier sans un mot de passe, mais il n' est pas possible non plus de specifier tous les mots de passe au depart. Aussi le mot de passe du disque en tient-il lieu. C' est la raison d' etre des parametres SPW et DPW. Ce sera necessaire pour reproduire le disque d' origine, apres SYSGEN, en copiant les utilitaires.

NEW et OLD sont aussi destines a la copie par masque, ils fournissent un moyen d' effectuer celle-ci en fonction du contenu de chaque disque. NEW peut donc etre utilise pour s'assurer que le disque destination contient tous les fichiers du disque source. OLD peut etre utilise pour effectuer une mise a jour du second disque.

Conclusion : Nous vous suggerons d' utiliser QUERY, ECHO et OVER pour des copies qui concernent un grand nombre de fichiers, surtout si le masque risque d' etre trop general.

Normalement, COPY ne copie jamais les fichiers systeme. Comme toutefois certains d' entre vous peuvent en avoir ecrit, pour leur permettre de les copier, nous avons prevu un parametre supplementaire, SYSTEM, qui permet de le faire, mais pas de copier les autres.

Certaines combinaisons de parametres peuvent ne pas fonctionner: NEW et OVER ne demanderont jamais l' autorisation d' ecrire : si le fichier existe, il ne sera pas copie, pourquoi donc poser la question ? De meme QUERY et ECHO ne feront pas poser deux fois la meme question.

CREATE

crée des fichiers disque et réserve leur espace

La syntaxe est:

CREATE nomfich (param=exp)

nomfich est une désignation standard de fichier

'param=exp' est un paramètre facultatif indiquant quelle action complémentaire vous attendez.

Les paramètres sont:

(D)ata=valeur remplit le fichier de données (1 ou 2 octets)
 (G)RANS=valeur nombre de granules à réserver
 (K)EEP=choix indique fichier non raccourcissable
 (K)ILO=choix nombre de koctets à réserver
 (L)RL=valeur longueur de bloc
 (S)IZE=valeur nombre de blocs
 (V)ERIFY=choix vérifie l'espace disque après la création

L'utilisation de cet ordre permet de créer un fichier et de lui réserver l'espace souhaité, ce qui est différent du fonctionnement habituel du système, où l'espace n'est alloué à un fichier qu'au fur et à mesure des besoins.

En créant ainsi un fichier, vous pouvez aussi éviter que l'espace qui lui est alloué ne diminue (paramètre KEEP), si à un moment quelconque il n'est pas assez long pour remplir l'espace qui lui est alloué.

Il n'est pas possible de créer un fichier sous un nom qui existe déjà, au moins sur le disque indiqué. Si le disque n'a pas la place d'accueillir votre fichier, vous aurez droit à un message d'erreur, qui vous indiquera qu'il vaut mieux essayer un autre disque.

Si vous ne réservez pas d'espace, vous aurez créé un fichier de longueur nulle, dont l'intérêt n'est pas évident (sauf pour réserver une place dans le directory).

L'intérêt de cet ordre est d'accroître sensiblement la vitesse d'accès au fichier, car la réservation d'espace se fait dans les meilleures conditions.

Reservation de l'espace:

Elle peut se faire par l'un des trois paramètres suivants:

SIZE, qui est lié au nombre et à la longueur des blocs.

KILO, qui s'exprime directement en koctets.

GRANS, qui est le nombre de granules à réserver (un granule est l'unité d'affectation d'espace disque de DOSPLUS, il est égal à un certain nombre de secteurs, qui varie suivant le type de disque utilisé et la densité d'enregistrement (voir section technique, note sur la GAT), c'est une façon d'accélérer le fonctionnement du système, en évitant d'avoir à procéder à une affectation d'espace chaque fois qu'un fichier s'allonge de 256 octets.

L'utilisation de LRL n'est pas compromettante, car DOSPLUS permet de définir la longueur d'enregistrement à chaque ouverture d'un fichier, mais peut dans certains cas faciliter les calculs de longueur.

Verification des donnees:

Pour verifier qu' il n' y a pas de defaults dans l' espace alloue au disque, vous disposez de deux parametres: DATA et VERIFY. Le premier fait remplir votre fichier de la donnee indiquee, le deuxieme relit secteur par secteur et verifie que ce qu' il lit est conforme a ce qu' il a ecrit.

Exemples:

```
CREATE monfich/dst:2 (LRL=128,SIZE=100)
CREATE monfich/dst:1,L=028,S=100
```

feraient creer un fichier sur le lecteur 2, son nom serait 'monfich/dst', sa longueur serait de 128*100=12800 octets.

```
CREATE monfich/dst:2,KILO=13,DATA=6CAA,K,V
```

creerait un fichier de longueur voisine, la zone qui lui serait affectee serait remplie avec les octets 6C et AA alternativement, et verifie.

DATE

permet d'afficher la date, ou de la mettre à jour

Syntaxe:

DATE

DATE MM/JJ/AA ou MM/DD/AAAA

La première écriture fait afficher la date dans le format indiqué à la ligne suivante, la deuxième permet de la mettre à jour (les '/' peuvent être remplacés par d'autres séparateurs (, ; - etc).

DOSPLUS accepte les dates de 1900 à 1999. MicroSystems Software offrira une modification gratuite en l'an 2000 aux propriétaires de DOSPLUS 3.5.

DEBUG

contrôle le débogueur (debugger)

La syntaxe est:

DEBUG (ON) ou (OFF)

DEBUG est un puissant moniteur sur disque. Il permet d'examiner toutes les adresses mémoire, de modifier n'importe quelle adresse (en mémoire vive) ou registre du Z-80. Son usage nécessite quelques connaissances en langage machine Z-80.

À l'activation, aucune modification du fonctionnement de la machine n'apparaît tant qu'une erreur ne se produit pas, que l'on ne charge pas un programme ou que l'on n'appuie pas sur <BREAK>. Dans ces cas, il se met en route et affiche son écran.

Vous disposez alors des ordres suivants:

A	affichage en ASCII
C	pas à pas avec saut des sous-routines
Daaaa	affichage de la zone mémoire comprenant l'adresse 'aaaa' (hexa)
Gaaaa,bbbb,cccc	saut à l'adresse 'aaaa', avec arrêts en 'bbbb' et 'cccc' (facultatifs)
H	affichage en hexa
I	déroulement pas à pas d'un programme (ne fonctionne qu'en ME).
Maaaa<espace>	modification de mémoire à partir de l'adresse 'aaaa', l'espace fait avancer le curseur, <ENTER> enregistre les modifications et retourne à DOSPLUS.
O	retour à DOSPLUS (DEBUG encore armé, comme ci-dessus)
Rpr aaaa	modification d'une paire de registre (le blanc est obligatoire)
S	passage en affichage plein écran
U	lance la mise à jour dynamique de l'affichage
X	passage en mode affichage des registres (qui est le mode initial)
;	affiche la page suivante
-	" précédente

Voici un exemple d'affichage de DEBUG (mode registres):

```

AF = 4044 -Z---P--
BC = 02C3: DA CC 06 D7 CA 97 19 FE 2F 28 4F CD 93 02 CD 35
DE = 3CFD: 20 20 20 20 20 20 20 20 20 32 38 34 30 3A 20 20
HL = 3D3D: 20 20 20 20 20 20 20 20 20 32 38 35 30 3A 20 20
AF' = 0514 ---H-P--
BC' = 0000:
      F3AF C3 74 06 C3 00 40 C3 00 40 E1 E9 C3 9F 06
DE' = 7C74: 75 B5 39 C3 3A 33 00 FC 00 00 00 00 FF F8 7F 02
HL' = 6056: 05 08 7E 3C E6 1F 77 08 FD E1 C9 21 36 40 01 01
IX = 401D: 02 66 FD 16 3E 20 8C 4F 02 B6 FB 43 11 00 B5 52
IY = 44A2: 40 79 46 10 00 00 00 00 00 2A 01 C0 02 12 12 06
SP = 41DC: 0D 02 C0 3F B7 FD D5 FD 42 00 1C 40 DD 03 1D 40
PC = 055E: ED B0 C1 EB 18 19 7D E6 C0 6F E5 11 40 00 19 7C
      2840: 2B 2A C5 7E 23 E5 CD EF 28 E1 4E 23 46 CD 5A 28
      2850: E5 6F CD CE 29 D1 C9 CD BF 28 21 D3 46 E5 77 25
      2860: 73 23 72 E1 C9 2B 06 22 50 E5 0E FF 23 7E 0C B7
      2870: 28 06 BA 26 03 B8 20 F4 FE 22 CD 7B 1D E3 23 EE

```

Le format general est le suivant:

Les paires de registres sont indiquees le long du bord gauche de l' ecran, les registres normaux d' abord, les autres ensuite, enfin les registres speciaux (IX, IY, SP, PC).

Le registre AF contient les indicateurs systeme, la lettre de chacun affichee indique qu' il est a 1 (ou MIS).

Les autres registres affichent leur contenu, suivi de celui des 16 octets de la memoire commençant a l' adresse egale au contenu du registre double (apres SP est affiche le contenu de la pile).

Les quatre dernieres lignes affichent le contenu de la memoire, chaque ligne commence par son adresse, suivi de 16 octets. Ce sont elles que l' on peut modifier par Maaaa.

Dans le cas de l' affichage pleine page, vous pouvez lire 16 ligne analogues, la premiere adresse est toujours terminee par '00'.

DIR

Affiche la table detaillee du contenu d' un disque (Directory)

Syntaxe :

DIR <from> lecteur <to> fich/periph <using> masque <ENTER>

Les parametres sont :

(S) SYSTEM=choix Affiche aussi les fichiers systeme.
 (I) INVIS=choix Affiche aussi les fichiers invisibles
 (K) KILL=choix Affiche aussi les fichiers invisibles

DIR affiche le Directory d' un disque, c' est a dire le maximum d' informations sur chaque fichier : Nom et extension, protections, taille, longueur de bloc, nombre de secteurs, nombre de blocs, volume en Koctets et date de la derniere mise a jour (si vous entrez regulierement la date).

Comme CAT, DIR fonctionne de deux facons : standard ou globale. Dans la premiere, vous lisez les noms de tous les fichiers d' un disque, suivant les parametres utilises, dans l' autre, vous lisez les noms de tous les fichiers du systeme repondant au masque, quelque soit le lecteur les contenant.

En demandant le directory, vous pouvez aussi preciser l' organe de sortie (visu par default). Ce peut etre un fichier ou un peripherique.

La redaction la plus simple est :

DIR

qui affichera le directory du lecteur 0, fichiers visibles.

La suivante sera :

DIR :1

qui affichera la meme chose, pour le lecteur 1.

L' affichage du directory des fichiers.

Il devrait avoir l' aspect suivant :

Drive: 0	SYST3.50	- Space:	105/128	66.0k		
Filespec	Attrib	LRL	#Secs	#Recs	Space	Updated
LSCRIP/CMD	0.....	256	45	45	12.0k	00/00/80
SCRIPSIT/LC	0.....	256	42	42	10.5k	00/00/80
INTROS/SCR	0.....+	256	43	43	10.5k	00/00/80
INIT/CFG	0.....+	256	7	7	3.0k	00/00/80

La premiere ligne est un resume succinct du disque :

- * Le nom du lecteur
- * Le nom du disque.
- * L' etat des noms de fichiers (libres/total)
- * La place libre en Kilooctets.

Le nom du lecteur (1 dans notre exemple) est la designation courante du lecteur etudie (c' est celui qui est precede de ':' dans la designation, qui peut ne pas etre le numero physique).

Le nom du disque est celui qui est affecte au disque par FORMAT.

L' etat des noms de fichiers permet de savoir combien de fichiers supplementaires vous pouvez mettre sur ce disque (DOSPLUS n' en permet qu' un maximum de 256 y compris les extensions, quel que soit l' espace disponible sur le disque). Si le premier nombre est nul, le disque est plein (les fichiers existants peuvent toutefois etre allonges, a condition de ne pas necessiter de segments supplementaires).

L' espace disponible vous donne l' espace disponible.

La ligne suivante est l' en-tete du tableau. Les titres des colonnes sont :

- * Nom de fichier.
- * Attributs de protection.
- * Longueur d' enregistrement logique (LRL).
- * Nombre de secteurs materiels (fSects).
- * Nombre d' enregistrements (fRecs).
- * Taille totale en Octets (Space).
- * Date de la derniere mise a jour (Updated).

Puis viennent les lignes du directory proprement dites. Si le disque est vide, ou si aucun fichier ne repond au masque, il n' y en aura aucune, mais il y aura autant de lignes que de fichiers a montrer. Si vous ne voyez pas un fichier que vous recherchez, essayez encore avec les parametres INVIS, KILL OU SYSTEM. Examinons chacun de ces articles un par un :

Le nom de fichier comprend le nom et l' extension.

Les attributs de protection : cette colonne contient 7 elements de grande importance :

- (1) Le degre de protection chiffre de 0 a 7, correspond au degre de protection (voir ATTRIB).
- (2) L' indicateur de mot de passe, P signale qu' un mot de passe est affecte a ce fichier.
- (3) L' indicateur systeme : S indique que le fichier est un fichier systeme.
- (4) L' indicateur KILL, K indique que le fichier a ete detruit, mais est eventuellement recuperable par RESTORE.
- (5) L' indicateur I indique que le fichier est invisible. Si vous avez utilise le parametre INVIS, c' est le seul moyen de savoir si un fichier est visible ou non.
- (6) L' indicateur de KEEP. Si ! apparait, le fichier ne pourra jamais etre raccourci (attribut mis par CREATE ou ATTRIB).
- (7) L' indicateur de modification. + indique que le fichier a ete modifie depuis qu' il est sur le disque. Il vous indique en principe que ce fichier contient des donnees uniques, et devrait etre recopie sur un autre disque.

La longueur de bloc est celle qui a ete affectee a la creation du fichier. Sa valeur est de 1 a 256. dans la plupart des cas, elle n' a qu' un interet documentaire. Si elle est inferieure a 256, le nombre d' enregistrements sera superieur a celui des secteurs.

Le nombre de secteurs est le nombre de secteurs physiques occupes par le fichier.

Le nombre de blocs est le nombre de blocs de longueur LRL constituant le fichier.

La taille totale en kilo octets est une autre expression de la taille du fichier.

Manuel utilisateur DOSPLUS 3.50

La date de la dernière mise à jour est celle de la dernière écriture sur ce fichier. Si elle est 00/00/80, cela veut dire que la date n'était pas mise lors de cet événement.

DO

Permet de lancer l'execution d' un fichier d' ordres.

Syntaxe :

DO nomfich (param=exp...)

Les parametres sont :

(B) BREAK=choix active ou desactiv la touche BREAK.

(H) HIGH=valeur regle le haut de memoire avant de commencer.

L' ordre DO permet de prendre dans un fichier des sequences de caracteres exactement comme si ils etaient tapes au clavier. Ce peut etre tres utile lorsque l' initialisation du systeme necessite qu' un certain nombre d' ordres soient executes a chaque mise en route. Ces ordres peuvent etre des ordres de la librairie, des noms de programmes d' applications que vous voulez executer, ou tout ce que vous pourriez entrer au niveau commande du SED.

Quand DO arrive a la fin de la liste, il rend la main a DOSPLUS. L' invite de DOSPLUS restera affichee en meme temps que le curseur reaparaitra. D' autres ordres peuvent etre tapes des ce moment.

Le parametre BREAK permet d' interrompre le fonctionnement de la touche <break> pendant l' execution du fichier, et de le restaurer apres. Sinon il est possible d' interrompre le déroulement de ce fichier quand il s' arrete sur une instruction PAUSE.

Le parametre HIGH permet de proteger la memoire au-dela de l' adresse designee, et ainsi controler la zone ou DO va resider. Il occupe 288 octets en memoire, sous le haut de memoire (256 pour un tampon, et 32 pour un DCB). Si vous avez des programmes en haut de memoire qui ne se protègent pas eux-memes, c' est un moyen de les proteger, sinon DO risque des les ecraser. Ceci n' est pas le cas des pilotes de DOSPLUS.

Vous avez deux moyens d' ajuster le haut de memoire: par le parametre HIGH de SYSTEM, ou celui de DO. Il peut etre fourni en decimal, supposition par defaut, ou en hexadecimal, si l' adresse est suivie de 'H'. Cet pointeur n' est pas modifie apres l' execution de DO.

Applications de DO

DO peut etre utile dans de nombreux domaines, mais les plus frequentes utilisations sont sans doute :

- (1) initialisation de programmes d' application.
- (2) Jeux d' instructions souvent utilisees.
- (3) Installation de corrections dans des programmes.
- (4) Fonctionnement automatique de programmes.

La premiere application est sans doute la plus utile. Pour lancer de nombreux programmes d'application, surtout en basic, il est souvent necessaire de regler certains ordres de la librairie, ou de definir des interfaces avec le systeme AVANT de lancer le programme proprement dit. DO vous permet d' entrer tous ces ordres et d' enchaîner avec le programme sans intervention de l' operateur, ce qui rend l' emploi du systeme plus agreable pour l' utilisateur, car il n' a pas a se soucier de la syntaxe des ordres.

La facon la plus commode de lancer cette execution est de mettre l' intruction DO dans une commande AUTO. Si par exemple nous ecrivons un fichier pour ajuster FORMS et lancer notre programme basic, nous placerions les instructions suivantes dans le fichier PAYE/TXT :

FORMS,P=66,L=60,W=40
BASIC PAYE-F-5

puis nous taperions l' instruction :

AUTO DO PAYE

A chaque mise en route du systeme (avec ce disque), l' instruction DO PAYE serait affichee et les ordres qu' elle contient seraient executes, apres etre affichees (voir AUTO).

L' usage de lignes de commentaires et de PAUSE peut permettre des echanges de disques, par exemple pour mettre a jour une copie de securite apres un travail.

DUMP

Permet de copier sur disque une zone memoire.

Syntaxe :

DUMP nomfich (param=exp...)

Les parametres sont :

- (D) DATA=choix indique que le fichier n' est pas un programme. Le fichier constitue ne sera pas un fichier programme.
- (E) END=valeur Adresse de fin de la zone.
- (R) RELO=valeur Adresse de rechargement.
- (S) START=valeur Adresse du debut de zone.
- (T) TRA=valeur Adresse de transfert.

DUMP est utilise pour copier sur disque une partie quelconque de la zone memoire comprise entre les pointeurs de debut et de fin de memoire utilisateur. Son contenu peut etre n' importe quoi: programme langage machine, donnees, etc.

Une fois le fichier sur disque, dans le cas des programmes en langage machine, vous pouvez soit en lancer l' execution directement en tapant son nom au niveau SED, soit le charger simplement par LOAD, et le lancer apres soit avec le parametre RUN de LOAD, soit par DEBUG, (voir ces instructions).

DUMP permet ainsi d' entrer en memoire des programmes en langage machine avec le mode modification de DEBUG, et de les sauver sur disque.

A la copie sur disque, il faut preciser si il s' agit de donnees ou d' un programme, car ces deux fichiers sont enregistres dans des formats differents. Pour un programme le CPU doit connaitre l' adresse de chargement (load address) et l' adresse de lancement (Transfer address). Elles peuvent toutes deux etre modifiees. Par exemple, votre programme pourrait avoir ete ecrit entre 8000H et B000h, mais pour etre execute, il devrait etre entre 6000H et 9000H; c' est l' occasion d' utiliser RELO, en lui donnant la valeur 6000H. A partir de la, a chaque chargement, il debuterait en 6000H.

Certains programmes ne sont pas lances a leur premiere adresse, mais a une autre, ce qui explique la presence de TRA, qui permet de la preciser pour les lancements.

Si aucun de ces deux parametres n' est fourni, DUMP le rechargera dans la zone ou il l' a lu, et le considerera comme un fichier de donnees, et retournera au SED.

Exemples :

DUMP essai (S=7000H,E=7250H,D)

enregistrera comme donnees cette zone dans ce fichier.

DUMP essai (S=9000H,E=9500H,R=6500H,T=665FH)

enregistrera un programme qui sera mis a partir de 6500H, et sera lance a 665FH.

ERROR

Permet d'obtenir un message d'erreur detaille (en anglais) pour chaque code d'erreur ou pour la derniere erreur, suivant la syntaxe employeee.

=====

Syntaxe :

ERROR valeur (valeur est le NO d'erreur dont on veut lire le texte).

=====

DOSPLUS fournit des messages d'erreur detailles au lieu de numeros, mais par mesure de compatibilite et pour permettre de s'y referer, cet ordre traduit le codes en clair (!).

ERROR a aussi la capacite rare de se rappeler la derniere erreur survenue. Il suffit de taper ERROR, et elle sera reaffichee.

FILTER

Permet de constituer un 'filtre' pour tout peripherique oriente caractere.

Syntaxe :

FILTER <from> periph <to> nomfich (param)

nomfich est le nom du fichier contenant le filtre.

Le parametre est

(Y)ES	charge et active le filtre, ou engage	un filtre deja
charge.		
(N)O	Charge un filtre sans l'activer, peut	etre utilise
pour desactiver un	filtre.	
(M)AP	affiche les codes du filtre.	

Un filtre est une table de donnees qui permet de transformer un caractere en un autre. Par exemple un programme a ete ecrit pour un imprimante qui fait remonter le papier d'une ligne a la reception du code 14. Pour le faire fonctionner avec une autre qui emploie le code 255, il suffirait de constituer un fichier filtre indiquant de remplacer 15 par 255 (voir FILTERS), et de l'activer.

DOSPLUS est livre en standard avec plusieurs filtres, qui sont expliques dans la section specialisee de ce manuel. Il ne faut pas les confondre avec les pilotes, qui sont des programmes.

Un bon exemple de filtre est le clavier DVORAK, qui redefinit les touches du clavier. Le peripherique a filtrer est dans ce cas le clavier, nous voulons modifier les donnees avant que le systeme ne les evalue. Nous pourrions simplement renommer les touches conformement a ce type de clavier; cela n'accomplirait que la moitie du travail. Par exemple la touche "Q" aurait l'air d'un "D", mais afficherait quand meme un "Q". Il faut donc un filtre.

Le peripherique a filtrer est dans ce cas le clavier. Nous voulons modifier les donnees avant que le systeme ne les evalue. Si nous transformions les "Q" en "D" (c'est cela la filtrage) avant qu'ils ne soient affiches, nous verrions bien un "D", mais DOSPLUS y verrait toujours un "Q", c'est pourquoi il faut les transformer des qu'ils quittent le clavier et avant qu'ils n'atteignent son pilote, ce que nous appelons filtrer le clavier.

Dans ce filtre que nous installons sur le clavier, nous instruons le systeme de convertir tous les "Q" en "D", ce qui donne le resultat suivant : vous tapez la lettre etiquetee "D", qui est en realite "Q". Cette derniere lettre passe dans le filtre qui la convertit en "D", et la transmet au pilote clavier, d'ou elle part dans le systeme comme "D".

Pour votre commodite, un filtre pour clavier DVORAK est fourni avec DOSPLUS. Si vous aviez un tel clavier, il vous suffirait de taper :

FILTERKI DVORAK

Et tout irait tout seul. Quand vous etablissez un fichier de configuration quelconque par l'ordre SYSTEM, tous les filtres actifs y sont sauves, et reinstalles automatiquement quand cette configuration est appelee.

Ecriture d' un filtre

Certain autres systemes vous permettent quelque filtrage, mais presque toujours, ils necessitent un usage plus ou moins pousse de l' assembleur. Pas DOSPLUS, Les filtres n' y sont rien de plus que des fichiers texte en ASCII.

Il faut d' abord etablir la liste des caracteres que vous voulez traduire. Ce peut etre en hexadecimal, ou un caractere entre guillemets. Pour traduire par exemple la lettre "A", on peut utiliser soit '41H', soit "A". Puis il faut mettre le signe '=' pour indiquer que la nouvelle valeur suit, et enfin la nouvelle valeur suivant la meme syntaxe que l' ancienne. Les instructions auraient alors cet aspect :

```
41=61    (traduit "A" en "a")  
"A"="a"
```

Il est possible de meler les deux types d' expression.

Une utilisation possible d' un filtre est d' eliminer certain caracteres qui font faire des choses bizarres a l' imprimante.

Exemples :

```
FILTER FROM aDD TO AFFICH/FLT  
FILTER aDD AFFICH
```

Sont deux expressions equivalentes.

```
FILTER aDD (MAP)
```

Afficherait tout filtre actif pour l' affichage. S' il n' y en a pas, on affichera 'No function' (pas en fonction). S' il y en a un, le filtre sera affiche.

```
FILTER aDD (NO)
```

desactivera le filtre actif pour l' affichage, mais ne le detruira pas, il pourra etre reengage par :

```
FILTER aDD (Y)
```

```
FILTER aDD AFFICH (NO)
```

Provoquera le chargement du filtre en memoire pour l' affichage, sans l' activer. Il pourra l' etre plus tard.

Pour oter un filtre d' un peripherique, il faut soit charger un fichier de configuration ou il n' est pas; soit reinitialiser le systeme, (OFF) desengage le filtre, mais ne l' efface pas de la memoire, et sa place n' est pas recuperable, sauf si on remplace ce filtre par un autre pour le meme peripherique.

FORCE

Permet de remplacer un peripherique par un autre.

Syntaxe :

FORCE

FORCE <FROM> designperiph <TO> periph/fichier

L'ordre FORCE fournit a l'utilisateur la possibilite de modifier le cheminement des E/S du systeme, ce qui donne un maximum de souplesse avec un minimum d'effort : la sortie imprimante peut etre redirigee sur un fichier disque, en vue d'une impression a une periode plus favorable, la sortie video peut de meme etre dirigee sur l'imprimante en vue d'une etude a tete reposee, etc...

Il n'y a pas de valeurs par defaut, Si un peripherique est designe, un autre peripherique ou un fichier doit etre designe, si un peripherique ou un fichier sont designes, un fichier doit etre designe. Si FORCE est entre sans designation de fichier ou de canal, il affichera la situation presente, qui aura l'aspect suivant :

```
$00 à KI ← 4DCAH
$01 à DO ↔ 0473H
$02 à PR → 03C2H
$03 à RS ↔ NIL
$04 à U1 - NIL
$05 à U2 - NIL
```

Le sens du mouvement des donnees mis en jeu par l'ordre FORCE doit etre le meme pour les deux peripheriques : les peripheriques emetteurs ne peuvent remplacer que des peripheriques emetteurs. Pour deplacer des donnees entre peripheriques de nature differente, utiliser COPY.

Si l'on FORCE un peripherique sur lui-meme, il est remis dans son etat normal. RESET peut aussi etre utilise pour annuler tous les ordres de redirection (voir RESET).

Restrictions

- (1) Seuls les peripheriques 0-5 peuvent etre le peripherique origine. Ce sont KI, DO, PR et RS, ainsi que les peripheriques definissables par l'utilisateur U1 et U2. (Ils peuvent etre rebaptises, et c'est leur nom actuel qui doit etre utilise. Il n'est pas permis d'utiliser des designations de lecteurs, sauf avec des noms de fichiers.
- (2) Un peripherique emetteur ne doit etre remplace que par un autre peripherique emetteur, ou par un peripherique capable d'emettre et de recevoir, et les peripheriques recepteurs uniquement par des peripheriques recepteurs, ou emetteurs/recepteurs. Bien qu'il soit permis de remplacer un peripherique recepteur par un peripherique emetteur et reciproquement, le resultat n'est pas toujours previsible.
- (3) L'ordre dans lequel les peripheriques sont remplaces est important : rappelez-vous que vous remplacez le premier peripherique par le deuxieme.
- (4) Quand vous remplacez un peripherique par un fichier, rappelez-vous que ce dernier restera ouvert jusqu'a ce que le peripherique soit remis dans son etat initial et FORCE annule. Si la machine est reinitialisee sans que ce soit fait, le fichier peut etre inutilisable.

Exemples :

FORCE àDD àPR

enverra sur l'imprimante tout ce qui doit aller sur l'écran.

FORCE àPR àPR

annulera l'ordre précédent.

FORMS

Permet de definir certains parametres de l'imprimante.

Syntaxe :

FORMS

FORMS (param=exp,...)

Les parametres sont :

(P)PAGE = valeur	Nombre de lignes possibles par page.
(L)lines =valeur	Nombre de lignes a ecrire par page. Le pilote de la ROM ne le fait pas.
(W)IDTH =valeur	Nombre de caracteres par ligne.
(T)OP	envoie un saut de page a l'imprimante pour l'initialiser.
(C) =valeur	Envoie immediatement un caractere ou deux (a preciser) a l'imprimante.

L'ordre FORMS vous donne le controle d'un certain nombre de parametres qui permettront au pilote de l'imprimante de tenir compte des caracteristiques du papier utilise, ou de certaines caracteristiques de l'imprimante elle-meme.

Certaines de ces fonctions ne seront utilisables que si vous activez le pilote de rechange fourni avec DOSPLUS. Le pilote en ROM ne tient compte de LINES ni sur le modele I ni sur le Modele III, ni de WIDTH sur le Modele I. Consultez la notice du pilote de rechange pour de plus amples details.

PAGE vous permet de definir le nombre de lignes qu'il est possible d'imprimer sur votre papier. Le papier standard a une longueur de page de 11 pouces (27,94 cm). Si votre imprimante ecrit 6 lignes par pouce, vous indiquerez PAGE=66. Si votre papier a des pages de 12 pouces (30,48 cm), vous indiquerez PAGE=72. Pour le meme papier et une imprimante qui ecrit 8 lignes/pouce, indiquez PAGE=96. (Sur le Modele III, avec le pilote standard, il faut indiquer le nombre +1).

LINES permet de fixer le nombre de lignes que l'on veut effectivement ecrire avant de faire un saut de page. Le pilote laissera en blanc un nombre de lignes egal a la difference entre PAGE et LINES, ce qui facilite la mise en page.

WIDTH permet d'informer le pilote du nombre de caracteres qu'il peut ecrire sur une ligne avant de passer a la suivante. Ce parametre est tres utile si l'on utilise une imprimante grande largeur (132 colonnes) avec du papier de largeur normale (80 colonnes), ou si l'on a une imprimante qui permet differentes largeurs de caracteres. Sur le Modele III, ce parametre doit etre entre augmente de 2 avec le pilote resident.

Un point a surveiller est le mode graphique de certaines imprimantes, qui peut necessiter l'envoi d'un grand nombre d'octets pour configurer l'imprimante, ou la metre dans l'un des modes speciaux. Il peut arriver que cela amene a envoyer plus de caracteres qu'il n'y en a sur une ligne. Le pilote standard du Modele I ne gerant pas ce parametre, il n'y aura pas de conflit, mais sur le Modele III, Le defaut de WIDTH est 255. Quand ce nombre est atteint, on passe a la ligne suivante. Pour eviter ce probleme sur ce modele avec le pilote de rechange, entrez WIDTH=0, et il ne s'occupera plus du nombre de caracteres.

TOP envoie un saut de page a l'imprimante, ce qui permet d'avancer le papier sans tou-

cher a l'imprimante (et de remettre a 0 le compteur de lignes). Il n'y a pas de parametre.

CODE permet d'envoyer immediatement un ou deux octets a l'imprimante. Il peut etre utilise avec plusieurs objectifs :

- (1) Envoyer un saut de page a une imprimante qui ne reconnait pas 0CH.
- (2) Envoyer un saut de ligne.
- (3) Initialiser un mode special (double largeur, soulignage, etc...)
- (4) Etre utilise dans un fichier DD pour envoyer a l'imprimante une suite de codes propres a la mettre dans une configuration determinee, convenant au programme qui va etre execute.

Ce ne sont que quelques exemples de ce que l'on peut faire avec ce parametre, car on peut envoyer n'importe quel code.

FREE

Permet l'affichage de l'espace libre sur le disque et dans le directory de tous les disques installes.

Syntaxe :

FREE

FREE <from> lecteur <to> periph/fichier

Il n'y a pas de parametre.

Sans designation de lecteur, FREE lit le directory de chaque disque installe et determine l'espace qui y est libre. (disque installe comprend les lecteurs logiques, qui ont pu etre crees en partageant un lecteur dur par exemple. La place libre est affichee en kilooctets, ainsi que le nombre de places dans la table des noms de fichiers.

Avec un nom de lecteur, FREE ne lit que lui, et affiche une carte du disque. Cette carte montre les granules formattes et indique ceux qui sont occupes (x), libres (.) ou interdits (L). Les granules du directory sont designes par 'D'. L'affichage se fait par cylindre, le numero du cylindre etant affiche en tete de ligne.

Exemples :

Drive: 0 TRS - Space: 066/128 0.0k

```
000-006: xxx xxx xxx xxx xxx xxx xxx
007-013: xxx xxx xxx xxx xxx xxx xxx
014-020: xxx xxx xxx xxx xxx xxx DDD
021-027: xxx xxx xxx xxx xxx xxx xxx
028-034: xxx xxx xxx xxx xxx xxx xxx
035-039: xxx xxx xxx xxx xxx
```

Drive: 0 SYST3.5D - Space: 105/128 66.0k

```
000-006: xxx ... .. ... .. xxx
007-013: xxx xxx xxx xxx ... ..
014-020: ... ... ... ... xxx xxx DDD
021-027: xxx xLx xxx xxx xxx xodL xxx
028-034: xxx xxx xxx L.L L. .LL .LL
035-039: LL. LLL LL. L.L LLL
```

La carte peut bien sur etre envoyee sur un autre peripherique que l'ecran, par exemple FREE &PR en provoquerait l'impression.

I

Permet d'initialiser un disque.

Syntaxe :

I lecteur (param=choix)

Le parametre est :

(M)OUNT=choix initialise le disque immediatement.

L'ordre I permet d'indiquer a DOSPLUS d'initialiser un disque, ou tous, c'est a dire de lire les informations, enregistrees dessus, qui decrivent ses caracteristiques au systeme, ainsi que le directory. Si donc un disque est initialise correctement, vous etes sur de deux choses :

- (1) DOSPLUS sait maintenant exactement comment ce disque est formatte.
- (2) Ce disque est lisible par le systeme, sauf erreurs locales.

A l'envoi de cet ordre, le systeme est invite a initialiser un disque au prochain acces. Si aucune indication de lecteur n'est fournie, il va executer le travail pour tous les lecteurs, autrement il ne le fera que pour celui qui est indique.

Le parametre MOUNT n'est necessaire que si vous utilisez des lecteurs double face et envisagez d'utiliser des disques tantot double face, tantot simple face. Il est inutile si tous les disques ont le meme format.

Ce parametre provoque l'initialisation du lecteur des l'envoi de l'ordre, sans attendre le prochain acces.

Il est egalement possible d'ajuster le nombre de faces d'un disque (pour le SED) avec CONFIG, mais I fait la meme chose pour moins d'effort de l'utilisateur, et verifie que le disque est lisible.

JOIN

Permet de lier deux peripheriques dans le systeme DOSPLUS.

Syntaxe :

JOIN <FROM> nomperiph <TO> periph/fichier.

L'ordre JOIN permet des E/S simultanees par deux peripheriques du systeme. Si par exemple vous voulez que tout ce qui est affiche sur l'ecran soit aussi imprime, vous n'avez qu'a donner l'ordre JOIN àDD àPR. A partir de la, tout ce qui est affiche est aussi imprime.

Il est aussi possible de JOINDre un peripherique emetteur a un fichier, tout ce qu'il enverra sera copie dans un fichier.

Il n'y a pas de valeurs par defaut pour 'nomperiph' ni pour 'periph/fichier', les deux doivent etre fournis. Si un peripherique est designe, il faut designer aussi un peripherique ou un fichier. Si aucun des deux n'est fourni, JOIN affichera l'etat des branchements (comme FORCE), s'il en existe.

Les peripheriques JOINTs doivent etre de meme categorie, emetteur ou recepteur. JOINDre un peripherique a lui-meme supprime les liaisons anterieures eventuelles. On peut aussi utiliser RESET.

Restrictions

- (1) Seuls les peripheriques 0-5 peuvent etre le peripherique origine. Ce sont KI, DO, PR et RS, ainsi que les peripheriques definissables par l'utilisateur, U1 et U2. (Ils peuvent etre rebaptises, et c'est leur nom actuel qui doit etre utilise. Il n'est pas permis d'utiliser des designations de lecteurs, sauf avec des noms de fichiers.
- (2) Un peripherique emetteur ne doit etre lie qu'a un autre peripherique emetteur, ou par un peripherique capable d'emettre et de recevoir, et les peripheriques recepteur uniquement a des peripheriques recepteurs, ou emetteurs/recepteurs. Bien qu'il soit permis de lier un peripherique recepteur a un peripherique emetteur et reciproquement, le resultat n'est pas toujours previsible.
- (3) L'ordre dans lequel les peripheriques sont lies est important : rappelez-vous que vous liez le premier peripherique au le deuxieme, c'est-a-dire que si vous liez àDD A àPR, ce qui apparaitra sur l'ecran, mais ce qui va sur l'imprimante n'ira pas sur l'ecran.
- (4) Quand vous liez un peripherique a un fichier, rappelez-vous que ce dernier restera ouvert jusqu'a ce que le peripherique soit remis dans son etat initial et JOIN annule. Si la machine est reinitialisee sans que ce soit fait, le fichier peut etre inutilisable.

Exemples :

JOIN àDD TO ECRAN/TXT

Enverra tout ce qui apparaîtra sur l'écran dans le fichier ECRAN.TXT, et pourra être examiné plus tard.

JOIN ÀRS ÀKI
JOIN ÀDO ÀRS

Par ces deux commandes, vous liez d'une part le périphérique de communications au clavier, et l'écran au même périphérique (ÀRS est capable d'émission et de réception). Si la liaison série est établie correctement, vous pouvez faire contrôler votre TRS-80 par un terminal distant, vous avez créé un 'serveur'.

JOIN ÀPR ÀPR

Annule la liaison en cours pour l'imprimante.

KILL

Permet de detruire un fichier ou un groupe de fichiers, et de neutraliser des peripheriques ou des lecteurs de disques.

Syntaxe :

```
KILL nomfich (param=expr)
KILL <from> lecteur <using> masque (param=expr)
KILL periph
KILL lecteur
```

Les parametres sont :

(I)NV=choix Indique si le masque englobe les fichiers invisibles.
 (E)CHO=choix Pour une destruction avec masque, indique au SED d'afficher les noms des fichiers avant de les detruire.
 (S)YS=choix Indique si le masque englobe les fichiers systeme.
 (Q)UERY=choix Pour une destruction avec masque, indique au SED de demander la permission de detruire chaque fichier.
 (P)W=choix Fournit le mot de passe du disque, en remplacement des mots de passe des fichiers.

L'ordre KILL permet d'eliminer du systeme des articles inutiles (fichiers, lecteurs ou peripheriques). Vous avez sans doute remarque qu'il utilise les memes parametres que COPY, sauf en ce qui concerne le mot de passe. Ils fonctionnent de la meme facon. Comme COPY, KILL opere sous deux modes : standard et global.

Le mode standard est appele en faisant suivre l'ordre KILL d'un nom de fichier, auquel cas ce fichier sera detruit, le mode global en le faisant suivre d'un masque, et tous les fichiers correspondant au masque seront detruits.

Si KILL est appele avec un nom de fichier sans NO de lecteur, le systeme va effectuer une recherche globale (sur tous les lecteurs actifs) jusqu'a ce qu'il trouve un fichier portant ce nom, puis il le detruira. Si un NO de lecteur a ete fourni, la recherche sera limitee a ce lecteur.

Si KILL est entre avec un masque sans NO de lecteur, le lecteur courant sera explore et purge, sinon ce sera le lecteur designe. Dans ce cas il faut fournir le mot de passe du disque, qui remplace les mots de passe individuels des fichiers.

ECHO permet de voir les noms des fichiers detruits, et eventuellement de recuperer ceux qui auraient ete detruits involontairement avec RESTORE.

QUERY permet d'autoriser ou interdire chaque destruction de fichier, mais oblige a participer a toute l'operation. Pour obtenir une destruction taper 'Y'<enter>. <enter> seul est un refus.

INV et SYS permettent d'inclure les fichiers invisibles et systeme dans le masque, ce qui est rare.

PW est necessaire pour une destruction globale.

La destruction de peripheriques doit etre faite avec precaution, car on risque de se trouver dans une situation sans autre issue qu' une reinitialisation generale du systeme : que faire d' autre apres un KILL @KI, qui vous enleve votre clavier. Les peripheriques elimines peuvent etre reactives en les remettant sur eux-memes (ASSIGN @PR @PR).

Exemple :

KILL */vie:4 (QUERY,PW="super")

Provoquera la destruction de tous les fichiers ayant l' extension /VIE du lecteur 4.

KILL */*:x1,Q,P="doi"

Provoquera la destruction apres accord de l' operateur de tous les fichiers du lecteur nomme x1 (tous les noms de fichiers seront acceptes par le masque */* qui peut etre remplace par !, et les fichiers invisibles et systeme sont acceptes par le masque, a cause de INV et SYS). Cela reviendrait a avoir un disque fraichement formate.

KILL PAYE?/DAT:1,Q,PW="JOUR"

Ne detruirait que les fichiers tels que PAYE1/DAT, PAYEF/DAT du lecteur 1, et avec permission.

LIB

Envoie la liste des ordres de la librairie au peripherique designe.

Syntaxe :

LIB <to> periph/fichier.

Il n' y a pas de parametre.

DOSPLUS fait la difference entre les ordres de la librairie et les programmes. Les ordres de la librairie font partie du systeme. Ce sont leurs noms qui sont fournis par LIB. Si un programme a le meme nom qu' un ordre de la librairie, ce dernier sera execute en priorite. Le systeme est etendu par addition de programmes qui remplissent des fonctions que ne remplissent pas les ordres de la librairie. Ces programmes ne faisant pas partie du systeme peuvent etre elimines sans autre consequence que la perte de leur service. Il n' en va pas de meme pour les ordres de la librairie, qu' il est delicat de supprimer.

Exemple :

LIB @PR

LIST

Envoie la liste des donnees provenant d'un fichier ou d'un peripherique, sur un autre fichier ou peripherique.

Syntaxe :

LIST <from> periph1/fichier1 <to> periph2/fich2 (param=exp...)

Le parametre est (C)TL=choix. Il determine si les codes de controle (ASCII 00H - 1FH) sont envoyes tels quels ou transformes en points (".").

L'ordre LIST est utilise normalement pour envoyer la liste d'un fichier disque a un organe de sortie tel que l'imprimante, mais peut etre utilise pour lister des donnees venant d'un organe d'entree comme le clavier ou le port serie. Les codes non imprimables sont remplaces par des ".", sauf les codes de controle ASCII, dans le cas ou l'on a pris l'option CTL.

Une pression sur la barre d'espace permet de suspendre l'affichage, <break> l'arrete.

LOAD

Permet de charger en memoire un fichier disque.

Syntaxe :

LOAD <from> fichier (param=exp...)

Les parametres sont :

(P)PROMPT=choix Determine si l' utilisateur sera invite a changer de disque ou non.
(R)RUN=choix Indique si le fichier doit etre execute ou non des la fin du chargement.
(S)START=adresse Determine la premiere adresse a charger pour un fichier langage machine.
(T)TRA=adresse Indique l' adresse de lancement.

LOAD lit un fichier sur disque et le charge en memoire. Si c' est un programme, c' est a dire qu' il est en code machine executable, et a l' extension /CMD, l' adresse de chargement viendra du fichier lui-meme, sinon c' est un fichier 'image memoire' et il faut employer START.

Tout fichier qui ne contient pas de codes de chargement est un fichier 'image memoire'. Les programmes executables contiennent des codes speciaux qui disent au systeme ou les ranger en memoire et ou les lancer. Un fichier 'image memoire' ne contient pas ces codes, ce peut etre des instructions binaires, du texte ASCII, ou des donnees. Nous lui donnons generalement l' extension /CIM. Si un nom de fichier est donne sans extension, LOAD prendra /CIM si START est utilise, /CMD dans la cas contraire.

Le parametre PROMPT permet de charger des programmes ne venant pas d' un disque systeme, en utilisant le lecteur systeme. Vous etes invite a mettre le disque convenable au bon moment. Presser <enter> fera progresser le travail. Si le programme doit etre execute, vous serez alors invite a remettre le disque systeme avant de le lancer.

Le parametre RUN indique au systeme que vous voulez que le programme soit lance apres son chargement. L' usage combine de ce parametre et de PROMPT permet d' executer des programmes en langage machine enregistres sur un disque donnees avec un seul lecteur.

Le parametre TRA indique au systeme le point d' entree d' un fichier image-memoire. Il peut etre utilise pour negliger le point d' entree normal d' un fichier /CMD.

Exemples :

LOAD essai/cmd:1

Ne provoquera que le chargement du fichier essai/cmd, l' adresse de chargement est trouvee dans le fichier lui-meme.

LOAD essai/cmd:1 (run)

ferait la meme chose, puis lancerait l' execution du programme. Cela revient au meme que taper 'essai' au niveau de commande de DOSPLUS.

LOAD essai/cim,S=7C00H,T=7D1EH

charge le fichier ESSAI/CIM a partir de l' adresse 7C00H, et indique une adresse de lancement en 7D1EH.

PAUSE

Suspend l'execution jusqu' a une pression sur une touche.

=====

Syntaxe :

PAUSE <message>

message est un texte facultatif

=====

L'ordre PAUSE fournit un moyen commode de suspendre l'execution de DOSPLUS pour donner a l'operateur une chance d'executer un travail quelconque. Il est generalement utilise dans un fichier DO. A l'execution, le mot PAUSE, suivi du message eventuel est affiche, et l'execution est arretee jusqu' a ce que l'on appuie sur une touche. Si l'on appuie sur <break>, la suite de l'execution est abandonnee, sauf si cette touche a ete neutralisee. A noter que toute la commande doit tenir sur une seule ligne.

Exemples :

Soit un fichier DO compose des lignes suivantes :

```
CLOCK ON
PAUSE Mettez SVP le disque PAYE/CAD.
LOAD PAYE/CIM (S=5500H)
PAYE
```

A l'execution de ce fichier, l'affichage de l'heure serait lance, puis l'instruction PAUSE serait executee, affichant :

PAUSE Mettez SVP le disque PAYE/CAD.

Et l'execution serait suspendue, donnant ainsi le temps a l'operateur (trice) de rechercher son disque et de le mettre en place, puis de continuer en pressant <enter>. Si la touche <break> est neutralisee, l'execution serait relancee par toute autre touche que <break>.

PROT

Permet de modifier des informations sur le disque.

=====

Syntaxe :

PROT lecteur (param=expr...)

Les parametres sont:

- (P)W=chaîne Fournit le mot de passe actuel du disque au systeme.
- (M)PW=chaîne Fournit le nouveau mot de passe si on veut le changer.
- (N)AME=chaîne indique le nouveau nom du disque.
- (D)ATE=chaîne indique la nouvelle date pour le disque. Peut etre entre en format libre.
- (L)OCK=choix determine si le nouveau mot de passe du disque doit etre affecte ou ote de tous les fichiers.

- (A)CC=choix si LOCK=Y le mot de passe du disque sera affecte a tous les fichiers pour l' acces, si ACC=N, tous les mots d' acces seront otes.
- (U)PD=choix Si LOCK=Y le mot de passe mise a jour du disque sera affecte a tous les fichiers, si LOCK=N, tous les mots de mise a jour seront supprimes.
- (C)LEAN=choix indique si les places inutilisees du directory doivent etre remplies de 0.
-

L' ordre PROT permet de modifier les attributs de protection qui avaient ete affectes a un disque par FORMAT ou BACKUP, ou une utilisation anterieure. Ces attributs comprennent le nom du disque, la date et le mot de passe du disque. De plus, PROT vous permet d' affecter le mot de passe du disque a tous les fichiers qu' il supporte, ou inversement d' enlever les mots de passe de tous les fichiers utilisateur (les fichiers systeme ne seront pas concernees).

Le parametre PW est une chaine de 8 caracteres valides (voir ATTRIB), entre guillemets ou apostrophes, il n' est pas fait de difference entre majuscules et minuscules.

Le parametre MPW affecte un nouveau mot de passe du disque. Les regles sont les memes que ci-dessus.

Le parametre NAME change le nom du disque, il est soumis aux memes regles.

Le parametre DATE permet de changer la date du disque. Elle est normalement affectee par FORMAT ou BACKUP, mais vous avez ainsi un moyen de la changer. En fait c' est une chaine de 8 caracteres alphanumeriques.

Le parametre LOCK affecte l' etat de protection des fichiers du disque. LOCK=Y affecte le mot de passe du disque a tous les fichiers comme mot d' acces et de mise a jour, a moins que l' on n' ait precise ACC ou UPD. Inversement, PROT=N ote a tous les fichiers les mots de passe d' acces et de modif, sauf comme ci-dessus.

Les parametres ACC et UPD utilises en liaison avec LOCK restreignent les modifications de mots de passe. Par exemple si ACC=NO est utilise avec LOCK=Y, seul le mot de passe de modif serait celui du disque, le mot d' acces ne serait pas touche. Ce serait l' inverse si UPD=N avait ete utilise.

CLEAN a pour effet de determiner si les places libres du directory doivent ou non etre remplies de 0. Certaines d' entre elles peuvent appartenir a des fichiers detruits, si elles sont remplies de 0, toute possibilite de recuperer ces fichiers est perdu.

Exemple :

```
PROT :AA (PW="SECRET",MPW="SDEC")
```

Changerait le mot de passe du disque du lecteur AA de "Secret" a "sdec". A noter que l' evaluation des caracteres independemment de leur categorie (majuscule ou minuscule) aurait donne le meme resultat avec "secret" et "sdec".

```
PROT :5 (lock=on,upd=n)
```

effacerait les mots de passe d' acces des fichiers utilisateur.

RENAME

Permet de rebaptiser des peripheriques, des lecteurs de disques et des fichiers.

Syntaxe :

RENAME <from> periph1/fichier1 <to> periph2/fichier2

Il n' y a pas de parametre.

Sous DOSPLUS l' utilisateur peut rebaptiser tout peripherique, fichier ou lecteur. Les noms doivent etre conformes a la section operations de ce manuel. En bref, un nom de peripherique est un @ suivi de un ou deux caracteres valables; un nom de lecteur de disques est ':' suivi de un ou deux des memes caracteres; un nom de fichier consiste en un nom de un a huit caracteres, une extension composee de '.' et de un a trois caracteres. Les mots de passe des fichiers ne peuvent pas etre changes par RENAME, mais s' ils existent, ceux-ci doivent etre fournis pour que le travail soit execute correctement.

Il n' est pas permis de donner le meme nom a deux peripheriques ou fichiers. Le masque ne fonctionne pas avec RENAME.

Le peripherique logique et/ou les noms de lecteurs sont inscrits dans la table des peripheriques du systeme. Un peripherique particulier doit donc etre designe par son nouveau nom, jusqu' a un nouveau changement de nom. Les nouveaux noms de fichiers remplacent les anciens dans le directory du disque. Si le meme nom existe sur plusieurs disques, il ne sera change que sur le premier si l' on ne precise pas le NO de lecteur.

Exemples :

RENAME @KI @CV

Change le nom du clavier de @KI a CV.

RENAME :0 :OR

Change le nom du premier lecteur de :0 a :OR

RESET

Relie un peripherique ou un lecteur a son pilote par defaut.

Syntaxe :

```
RESET
RESET <from> periph/lecteur
```

Il n' y a pas de parametre.

L' ordre RESET est utilise pour annuler tout ordre JOIN ou FORCE effectif pour un peripherique et relie celui-ci a son defaut. Un pilote par defaut peut etre soit la valeur a l' initialisation, soit une valeur ASSIGNEe. RESET relie toujours a la derniere valeur utilisee.

Si un peripherique ou un lecteur sont designes, l' action n' affectera que le peripherique/lecteur designe, sinon elle sera globale. Les filtres restent actifs; les noms logiques des peripheriques ou lecteurs ne sont pas modifies. Si un peripherique etait lie a un fichier, celui-ci sera ferme.

Exemple :

```
RESET @PR.
```

RS232

Permet d'afficher les réglages en service pour l'interface serie, ou des les modifier.

Syntaxe :

RS232

RS232 (param=expr...)

Les parametres sont :

(B)AUD=valeur	regle la vitesse en bauds
(W)ORD=valeur	regle la longueur du mot
(S)TOPS=valeur	nombre de bits stop.
(P)ARITY=valeur	nombre de bits de parite.
(E)VEN=choix	parite paire.
(O)DD=choix	parite impaire.
(D)TR=choix	ligne DTR (machine prete)
(R)TS=choix	Ligne RTS (autor. emission)
(B)REAK=choix	etat du break RS232.

L'ordre RS232 sert au controle du peripherique de communications serie (RS232C), qui est generalement utilise pour les communications avec d'autre ordinateurs, ou avec une imprimante serie. Il vous permet de connaitre et de modifier les parametres qui en controlent le fonctionnement.

Il n'y a aucun pilote standard pour ce peripherique dans le Modele I; ceux qui sont fournis avec le Modele III sont si peu efficaces que DOSPLUS les elimine a la mise en route. C'est pourquoi vous pouvez utiliser cet ordre pour fixer tout parametre de votre choix, mais apres avoir installe le pilote que nous vous fournissons. Vous trouverez de plus amples informations a son sujet dans la section pilotes et filtres.

Pour lire les parametres actifs, il suffit d'entrer RS232. Il seront alors affiches.

Le parametre BAUD controle la vitesse de transmission. Les valeurs autorisees sont :50, 75, 110, 134, 150, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 7200, 9600 et 19200. Les plus utilisees sont 300 et 600 bauds.

Le parametre WORD permet de choisir la longueur de mot a utiliser, c'est a dire le nombre de bits qui le composent. Les valeurs possibles sont 5, 6, 7 et 8. Les deux dernieres valeurs sont les plus frequemment utilisees : 7 permet de transmettre tous les codes ASCII, 8 permet de passer n'importe quel octet.

Le parametre STOP permet de fixer le nombre de bits de fin de mot. En communications asynchrone, chaque mot de donnee est encadre par des bits de debut et de fin, qui ont pour but de synchroniser le debut des elements de donnees. Le bit de debut est normalement un seul bit a 0, apres la reception des donnees, il y a habituellement un ou deux bits de fin a 1. La reception du mot suivant commence par un autre bit de debut, a 0.

Les valeurs possibles sont 1 ou 2. Il vous suffit de connaitre les besoins de votre cor-

respondant et de prendre les memes parametres.

Le parametre PARITY permet d'activer ou neutraliser la procedure de controle d'erreurs de parite. Il s'agit d'un bit supplementaire ajoute a chaque mot qui permet de savoir si le nombre de bit a 1 du mot est correct. Ce parametre est soit actif (ON), soit inactif (OFF). Il est aussi a regler suivant les besoins du correspondant.

Le parametre EVEN (actif ou ON) decide que le nombre de bits a 1 doit etre pair, Le parametre ODD decide que ce nombre doit etre impair. Si PARITY est OFF, ces deux parametres n'ont pas de sens. Dans le cas contraire EVEN=Y equivaut a ODD=N et reciproquement.

Le parametre DTR vous permet d'activer ou neutraliser le signal DTR. Beaucoup d'equipements, comme les modems, l'utilisent pour savoir si vous (votre TRS 80) est pret a communiquer, il s'agit d'un signal logique, non d'information transmise.

Le parametre RTS vous permet d'activer ou non le signal RTS. C'est aussi un signal utilise par des peripheriques (modems par ex.) pour savoir si vous etes pret a recevoir des donnees. Vous pouvez aussi le laisser actif.

Le parametre BREAK permet d'activer la condition BREAK du RS232. En general, vous pouvez le laisser coupe, car s'il est actif, vous ne pouvez ni emettre, ni recevoir.

Exemples :

```
RS232 (BAUD=300,WORD=7,STOPS=1,PARITY,EVEN,DTR,RTS,BREAK=N)
RS232,B=300,W=7,S=1,P,E,D,R,BR=N
```

Reglera l'interface RS232 pour une vitesse de 300 bauds, un mot de 7 bits, la parite sera paire, les lignes RTS et DTR actives,break inactif.

```
RS232,B=1200
```

Changerait la vitesse pour 1200 bauds, sans modifier les autres parametres.

SCREEN

Permet de copier l'ecran sur un fichier ou vers un autre peripherique.

Syntaxe :

SCREEN <to> periph/fichier.

Il n'y a pas de parametre.

L'ordre SCREEN permet d'envoyer ce qui est sur l'ecran au moment ou il est donne au fichier ou peripherique designe (QFR par default).

C' est un moyen commode de conserver des affichages. L'ordre peut etre inclus dans des programmes basic ou lance depuis des programmes binaires pour garder trace des entrees de l'operateur.

Exemple :

SCREEN

enverrait le contenu de l'ecran a l'imprimante. L'execution du programme en cours sera suspendue pendant la duree de cette operation.

SCREEN to ecran/vid

enverrait le contenu de l'ecran dans le fichier ECRAN/VID, qui serait cree s'il n'existe pas, sinon son contenu precedent serait ecrase.

SYSTEM

Permet de configurer certaines parties de DOSPLUS suivant vos besoins.

Syntaxe :

SYSTEM
SYSTEM (param=expr...)
SYSTEM fichier

Les parametres sont :

(T)IME=choix	active ou desactive la question TIME
(D)ATE=choix	id id id DATE
(L)OGO=choix	id id l' affichage du logo
(B)LINK=choix	id id le clignotement du curseur
(C)APS=choix	inverse le mode caractere (maj. ou minuscules)
(C)URSOR=valeur	permet de choisir le caractere du curseur
(H)IGH=valeur	fixe l' adresse du haut de memoire.
(S)TEP=valeur	fixe la vitesse par defaut des tetes
(S)AVE=choix	rend permanentes certaines modifications
(P)ORT=valeur	adresse du port vers lequel on fait une sortie a la mise en route.
(M)ODE=valeur	octet a envoyer a ce port.

L' ordre SYSTEM permet de modifier certains parametres de fonctionnement de DOSPLUS, mais son aspect le plus important est peut-etre la constitution de 'fichiers de configuration'. Ce sont des programmes binaires executables au niveau commande de DOSPLUS, dans un programme JCL ou dans n' importe quel cas ou il est possible d' executer un tel programme. Ces fichiers retablissent l' exacte configuration qui etait en service au moment de leur ecriture.

SYSTEM peut prendre trois formes. D' abord, vous pouvez entrer SYSTEM seul, et lire les pointeurs de memoire actuels. Deuxiemement, vous pouvez entrer SYSTEM suivi d' une liste de parametres a modifier. Enfin vous pouvez entrer SYSTEM suivi d' un nom de fichier, ou il ecrira le fichier de configuration.

Mode 1

Tapez simplement :

SYSTEM

et pressez enter. DOSPLUS affichera trois valeurs : L' adresse du debut de memoire utilisateur, l' adresse du haut de la memoire utilisateur et l' adresse du sommet physique de la memoire. Ces valeurs sont affichees en hexadecimal.

Mode 2

Ce mode permet de modifier les parametres de SYSTEM. Certains prendront effet tout de suite, d' autres ne prendront effet qu' a la prochaine reinitialisation. Certain vont configurer le disque automatiquement de facon permanente, pour d' autres il vous faudra preciser un autre parametre

pour les rendre permanents.

Prennent effet immediat et sont permanents :

TIME si vous mettez TIME=N, l'heure ne vous sera plus demandee systematiquement. Vous pourrez toujours la fixer avec l'ordre TIME

DATE Si DATE est inactif, la date ne vous sera plus demandee. Il vous sera toujours possible de la mettre avec l'ordre DATE (voir cet ordre)

LOGO si ce parametre est =N, le logo DOSPLUS ne sera plus affiche

PORT permet d'envoyer un octet choisi sur un port determine. De nombreuses accessoires étrangers a RADIO-SHACK demandent qu'un nombre soit emis par un certain port pour fonctionner. DOSPLUS peut le faire automatiquement.

MODE definit l'octet a envoyer

Prennent effet immediatement, mais ne deviennent permanents qu'apres SAVE :

BLINK permet d'activer ou desactiver la fonction clignotement du curseur. Le Modele III supporte les deux modes de fonctionnement, mais pour que BLINK fonctionne sur Modele I, il faut d'abord activer le pilote clavier DOSPLUS.

CAPS permet de changer entre majuscules et minuscules (comme <shift> 0), (si le pilote clavier est installe sur Modele I). A chaque SAVE, la valeur actuelle devient la valeur par defaut.

CURSOR permet de changer le caractere curseur a tout moment, en donnant a ce parametre la valeur du caractere choisi, en decimal ou hexa. Il necessite aussi le pilote clavier de DOSPLUS sur modele I.

HIGH permet de modifier la limite superieure de la memoire utilisateur. DOSPLUS utilise le fond de memoire pour certains pilotes et filtres. Cette zone ne devrait etre modifiee par l'utilisateur sous aucun pretexte. Les programmes auxiliaires de DOSPLUS qui l'utilisent se protègent eux-memes, ainsi que devraient faire tous ceux qui utilisent cette zone. Tous les programmes devraient aussi la respecter (c'est le cas des notres). Pour permettre quand meme de proteger la zone d'un programme contre les interventions des autres, nous vous avons mis ce parametre.

Parametres ne devenant effectifs qu'a la prochaine initialisation du systeme :

STEP regle la valeur par defaut du parametre de passage des tetes d'une piste a une autre. La valeur d'origine est la plus lente possible, pour que DOSPLUS fonctionne avec n'importe quel lecteur. Toutefois les lecteurs de RADIO-SHACK, comme la plupart des autres peuvent fonctionner a des vitesses plus elevees : les lecteurs Modele I peuvent tous fonctionner a 20 ms, ceux du modele III a 6ms. Pour ne pas avoir a construire un fichier de configuration rien que pour ca, Nous l'avons ajoute.

Ce parametre ne regle que la valeur generale. Si vous avez un lecteur qui opere a une vitesse differente, il vous faudra utiliser CONFIG. Ce parametre peut prendre les valeurs suivantes :

Valeur	pas
0	6ms
1	12ms
2	20ms
3	30 ms (double densite)
	40 ms (simple densite)

Note technique : pour des lecteurs 8 pouces, la valeur de ce pas sera la moitie de celle qui est donnee ci-dessus. Limitez vous aux valeurs de cette table, d'autres valeurs pourraient provoquer un mauvais fonctionnement du controleur de disques. Ces valeurs sont relatives.

Le parametre STEP devient permanent immediatement, mais n'entrera en service qu' a la prochaine reinitialisation.

Mode 3

C' est la forme de SYSTEM qui permet la constitution des fichiers de configuration.

Pour comprendre les methodes de confection des fichiers de configuration, il est res important de comprendre ce que sont ces fichiers et pourquoi les utiliser. Dans tout le systeme DOSPLUS, on trouve des ordres tels que CONFIG, FORMS et RS232 qui permettent d' alterer des parametres affectant le fonctionnement du systeme, certains pilotes sont necessaires pour permettre l' accomplissement de certaines fonctions; avant d' etre utilise, le langage de controle du travail (JCL) doit etre charge. Pour toutes ces applications, toutes ces zones retournent dans leur etat initial a chaque reinitialisation du systeme. La raison d' etre des fichiers de configuration est de restaurer les zones memoires correspondantes dans l' etat ou elles etaient a leur ecriture.

Supposons que nous ayons active les pilotes de rechange pour le clavier, l' imprimante, la video et l' interface serie; que nous ayons prepare FORMS pour un certain format de papier, que nous ayons defini le lecteur NO 1 comme un double face, d' une vitesse de tetes de 6ms et modifie les parametres par defaut du RS232. En bref nous avons adapte DOSPLUS a un travail particulier et a une machine particuliere.

Nous voulons maintenant preserver tout cela, car c' est long a faire. Il nous suffit d' entrer :

```
SYSTEM init:0
```

pour que DOSPLUS ecrive toutes nos donnees, nos pilotes, etc dans le fichier init/cfg:0 (cfg est l' extension par defaut).

Par la suite, pour remettre en service cette configuration, il nous suffira d' entrer :

```
INIT/CFG:0
```

ou de le faire lancer par un fichier DO, ou par un autre moyen.

Rien n' empeche d' avoir plusieurs fichiers de configuration, correspondant a des cas differents, et de choisir celle qui convient au travail que nous voulons executer a la mise en route.

TIME

Permet la mise a l'heure de l'horloge interne et la lecture de l'heure.

Syntaxe :

TIME Affichage de l'heure.
TIME hh:mm:ss Mise a l'heure

L'heure est affichee sous le format HEURE: MINUTES: SECONDES.
Pour la mise a l'heure, le format peut etre le meme, mais vous pouvez utiliser un format libre, ou les separateurs peuvent etre tout caractere non numerique. Cela donne davantage de souplesse.

L'heure fournie par cette horloge est sur 24 heures.

Exemples :

TIME 03:05:00
TIME 3.5
TIME 3-5/0

Toutes ces formes metront l'horloge a 3 heures 5 minutes (du matin).

VERIFY

Provoque la verification systematique de toute ecriture sur disque.

Syntaxe :

VERIFY (param)

param est l' etat desire de la commande (ON=active, OFF, ou N pour inactive).

Cet ordre provoque la relecture automatique apres ecriture de toute E/S disque. Il ralentit quelque peu la vitesse de ces operations, mais peut etre souhaitable pour des donnees importantes.

Quand VERIFY est actif, meme des utilitaires comme CONVERT, ou des ordres de la librairie comme COPY verifient ce qu' ils ecrivent. Tout programme utilisateur qui utilise les sous-programmes standards d' E/S de fichiers sera oblige d' en faire autant.

Exemples :

VERIFY
VERIFY YES
VERIFY Y
VERIFY ON

mettront VERIFY en service.

VERIFY OFF
VERIFY N

le mettront hors service.

Programmes utilitaires

<u>Nom du programme</u>	<u>Page</u>
BACKUP	3/2
CONVERT	3/4
DIRCHECK	3/6
DISKDUMP	3/8
DISKZAP	3/11
FORMAT	3/16
HELP	3/17
MAP	3/18
PATCH	3/19
RESTORE	3/21
SYSGEN	3/22
TAPE	3/23
TRAP	3/25
 CODIR	 3/25

BACKUP

Ce programme copie toutes les données d'un disque sur un autre. Il ne fait que des 'photocopies'. Pour reproduire un disque par une méthode de 'copie par fichiers', utilisez l'ordre COPY de la librairie. Cet ordre servira aussi pour copier le disque dur.

=====

BACKUP <from> :lect1, <to> :lect2 (param=expr)

Lect1 est le lecteur origine
Lect2 est le lecteur destination.

Les paramètres admis sont les suivants :

(D)ATE=chaîne permet d'entrer la date directement depuis sa ligne d'ordres, si vous savez que la date du système n'est pas à jour, et ne voulez pas être questionné.

(U)SE=chaîne Vous permet d'obliger à écraser tout ce qui peut exister sur le disque SANS avoir à donner votre avis, 'Y' provoque la poursuite de la recopie, 'N' la fait abandonner, 'F' provoque un formatage systématique.

=====

Cet utilitaire permet d'effectuer une copie d'un disque. Il est de bonne pratique d'effectuer de fréquentes copies de vos disques de données.

Avec le BACKUP de DOSPLUS, il n'est pas nécessaire de formater un disque au préalable. Si le disque est vierge, BACKUP le formatera automatiquement. Même dans le cas contraire, vous avez la possibilité de le formater a priori avant de l'utiliser.

Si vous n'avez pas indiqué les lecteurs source et destination, BACKUP vous les demandera; il ne vous demandera que celui qui manque si vous n'en avez indiqué qu'un.

BACKUP peut aussi être lancé par un fichier DO. Cela permet d'utiliser BACKUP comme une option d'un menu dans un programme BASIC et de retourner au menu. La procédure est la suivante :

- (1) Le premier ordre du fichier DO doit vous faire quitter BASIC et retourner au SED.
- (2) Exécuter le BACKUP
- (3) faire recharger le BASIC par le fichier DO et relancer votre menu.

Elle est simplifiée par la possibilité de fournir toutes les indications sur la ligne d'ordres de DOSPLUS, c'est vraiment une opération 'mains dans les poches'. L'opérateur n'a même pas à répondre à une remarque 'Diskette contains data' (le disque contient des données).

Questions

Source drivespec ? (lecteur source)
répondre par la désignation du lecteur qui contient le disque à reproduire.
Destination drivespec ? (lecteur destination)

répondre par la désignation du lecteur qui contient le disque destination.

Pour ces deux réponses, il suffit de fournir la désignation du lecteur (un ou deux caractères, sans ':'). Vous pouvez indiquer le même lecteur pour faire un BACKUP avec un seul lec-

teur.

Backup date (MM/DD/YY) ? (date de la copie <Mois/Jour/Annee>)

Cette question ne vous sera posee que si la date du systeme n' est pas valide, et si vous ne l' avez pas fournie sur la ligne d' ordres. Vous avez trois reponses possibles :

- (1) <break> et abandonner.
- (2) <Enter>, la date sera 00/00/80.
- (3) entrer huit caracteres ASCII de votre choix.

Si le disque n' est pas vierge, et si vous n' avez pas precise le parametre USE, vous verrez la question :

Diskette contains data, Use or not ?
(le disque contient des donnees, Utiliser ou non ?)

La reponse peut etre :

- (1) <break> et abandonner.
- (2) Entrer 'Y' ou 'U' et tenter d' utiliser le format existant.
- (3) Entrer 'F', et provoquer un formatage a priori.

Quand BACKUP aura des reponses a toutes ces questions, il executera son travail. Le disque destination aura les memes nom et mot de passe que l' original. La date sera soit la date courante, soit ce que vous aurez fourni.

Lecteur unique ou lecteurs multiples

Si les lecteurs source et desination sont differents, l' operation se fera seule. Dans le cas contraire, l' operateur sera invite de temps a autre a intervertir les disques source et destination (attention a ne pas se tromper).

Exemples :

BACKUP

Lancera le programme, qui demandera toutes les informations, sauf peut etre la date.

BACKUP :0 :1,Date="sept 24",use="y"

lancera le programme, qui n' aura besoin d' aucune intervention exterieure.

Il faut noter que BACKUP ne fonctionne pas entre disques de formats differents: les deux disques doivent etre de meme densite, nombre de faces, et nombre de pistes. En cours de travail, BACKUP ne recopie que les granules alloues, et saute les autres, ce qui peut faire paraitre son fonctionnement quelque peu erratique.

Note importante: Si BACKUP vous pose la question 'Diskette contains data, Use or not', il ne faut pas changer le disque source. En effet cela provoquerait l' inscription de donnees erronees qui par la suite degraderaient les donnees. Par contre vous pouvez changer le disque destination.

De toute evidence, si vous lancez BACKUP avec toutes les reponses aux questions sur la ligne d' ordres, vous avez interet a ce que les disques soient montes et prêts, surtout si vous avez pris l' option 'USE=Y'.

CONVERT

Cet utilitaire a plusieurs usages :

- (1) Permettre a DOSPLUS 3.5 de copier des fichiers du TRSDOS double densite sur des disques compatibles DOSPLUS 3.5
- (2) Permettre d' afficher le catalogue des disques TRSDOS double densite.
- (3) rendre les disques Modele I simple densite lisibles par le DOSPLUS 3.5 du Modele III.

```
CONVERT <from> !lects <to> !lectd <using> masque (param=expr)
CONVERT <from> !lect (param=expr)
```

Les parametres sont :

CAT=option Indique a CONVERT d' afficher le catalogue d' un disque Modele III, double densite.

DIR=option Comme ci-dessus

(E)CHO=option Provoque l' affichage des fichiers copie sur un support compatible DOSPLUS 3.5.

(I)NVIS=option Inclut les fichiers invisibles dans le masque.

(S)YSTEM=option Inclut les fichiers systeme dans le masque.

(O)VER=option Provoque une demande d' accord si le fichier existe deja sur le disque destination

(Q)UERY=option Demande accord avant toute copie.

V12 Informe CONVERT que le disque Mod III est une version 1.2 ou anterieure

V13 Informe CONVERT dd dd 1.3

(M)OD=option Pour DOSPLUS Modele I seulement, indique si le disque a utiliser est un disque double densite Modele I ou Modele III..

La premiere application de cette liste, copier des fichiers du TRSDOS double densite sur des disques DOSPLUS 3.5, fonctionne sur les deux modeles. Le Modele I simple densite ne peut pas l' executer, car les disques sont illisibles sur une machine simple densite.

L' utilisation de CONVERT pour copier des fichiers du TRSDOS double densite demande deux lecteurs : un pour contenir le disque double densite, l' autre pour contenir le disque destination compatible DOSPLUS. Il faut bien sur avoir en permanence un disque systeme DOSPLUS dans lecteur 0.

La syntaxe generale est :

```
CONVERT <from> !srce <to> !dest <using> masque (param=expr)
```

ou srce est le lecteur contenant le disque TRSDOS, !dest le lecteur contenant le disque DOSPLUS.

Exemples :

```
CONVERT !1 !0,*/*cmd copiera tous les fichiers du disque TRSDOS double densite du lecteur 1 au lecteur 0.
```

```
CONVERT !1 !0,e copiera tous les fichiers,visibles et non systeme, en affichant leurs noms.
```

Le parametre MOD1 n' existe que sur le CONVERT du Modele 1. Les usagers du Modele III n' ont pas a s' en soucier. Le CONVERT du Modele 1 peut CONVERTir des fichiers en TRSDOS double densite des deux modeles. Il suppose normalement que l' operation se fait sur un TRS-80 Modele 1 double densite, au lieu d' un modele III. C' est pourquoi, pour convertir les fichiers pour un TRS-80 modele 1 double densite, on pourrait utiliser l' ordre :

CONVERT :2 :1,e

Mais pour faire cette conversion pour un Modele III, il faudrait ecrire :

CONVERT :2 :1,e,mod1=N

'Mod1=n' indiquant qu' il s' agit d' un disque Modele III.

Au moment de la redaction de ce manuel, il existe deux types generaux de TRSDOS pour le Modele III: la version 1.2 et anterieure, et la version 1.3. L' emploi de CONVERT exige que le programme soit informe du type de TRSDOS auquel il va avoir affaire. La version 1.3 est la valeur par default. Donc l' ordre :

CONVERT AR!:3 :0

copiera les fichiers dont le nom commence par AR sur le disque TRSDOS 1.3 du lecteur 3. Si c' etait un disque TRSDOS 1.2 ou 1.1 il faudrait entrer :

CONVERT AR!:3 :1,V12

Les utilisateurs de Modele I doivent mettre le parametre MOD1=N pour lire les disquettes Modele III, ex :

CONVERT :1,CAT,MOD1=N

Permettrait de lire le catalogue d' une disquette Modele III, TRSDOS 1.3, sur un Modele I.

Le troisieme objectif majeur de CONVERT est de rendre les disques simple densite ecrits par le Modele I utilisables sur le Modele III. La piste de leur directory est dans un format inutilisable par le Mod III. CONVERT Peut etre utilise pour remedier a ce probleme. La syntaxe est alors :

CONVERT :lect. (lect. est le lecteur portant la disquette) a modifier.

Cette forme de CONVERT ne necessite qu' un seul lecteur, l' operateur sera invite a echanger les disques systeme et objet quand ce sera necessaire. Elle ne copie pas les fichiers d' un disque sur un autre, mais modifie le disque objet.

Note : apres cette operation, il peut arriver que certains systemes d' exploitation du Mod I (comme TRSDOS 2.3) ne puissent plus lire correctement le disque objet a cause de modifications du directory. DOSPLUS Mod I les lira, comme la plupart des SED actuellement disponibles pour le Mod I. S' il est necessaire de reutiliser un tel disque avec un SED qui ne peut pas le lire, il est possible d' annuler les modifications effectuees, A CONDITION D' AVOIR UN MODELE I DOUBLE DENSITE. Il suffit de reutiliser la meme syntaxe que ci-dessus :

CONVERT :dest.

Le directory sera remis dans son etat initial, et n' importe quel SED modele I pourra lire le disque normalement.

DIRCHECK

Cet utilitaire a pour but de verifier l' integrite du directory d' un disque, et au choix, de reparer certains types de dommages.

DIRCHECK <from> :lect <to> fichier/periph (param=expr)

Les parametres sont :

- (C)YLO Prescrit d' ignorer une erreur 'extraneous or unassigned gran' (granules externes ou non affectees) sur les fichiers d' initialisation du Modele I double densite (BOOT/SYS).
- (F)ILES Prescrit de reparer les erreurs eventuelles sur les titres du directory.
- (G)AT Prescrit de reparer la table d' allocation des granules (Granule Allocation Table), si necessaire.
- (H)IT Prescrit de reparer la table des index de rangement (Hash Index Table)

Cet utilitaire peut etre utilise pour verifier automatiquement le directory d' un disque, et signaler les erreurs qui s' y trouvent. L' ordre le plus simple est :

DIRCHECK :lect

le directory va etre lu et une liste (eventuelle) d' erreurs sera affichee. A la fin, Vous verrez 'DIRCHECK complete, xxx total errors' (DIRCHECK termine, nombre total d' erreurs xxx).

La liste des erreurs peut etre dirigee vers tout peripherique oriente caractere, ou un fichier, en indiquant son nom.

DIRCHECK est egalement capable de reparer certaines erreurs du directory. Les parametres FILES, GAT et HIT sont utilises pour informer DIRCHECK quelle(s) partie du directory devraient etre reparees.

Il est bon de noter que, comme n' importe quel 'reparateur de directory', DIRCHECK ne peut pas reparer les erreurs suivantes :

Erreur	Remede possible
Locked gran assigned to file (Affectation de granule interdit)	Detruire le fichier en cause.
Granule multiply assigned (Granule affecte plusieurs fois)	id
Granule assigned past cylinder count (Granule affecte au dela du nombre de cylindres.)	id
BOOT/SYS not found (fichier BOOT/SYS introuvable)	Restaurer le fichier BOOT/SYS sur le disque
BOOT/SYS not assigned space (Pas de place affectee a BOOT/SYS)	id
DIR/SYS not found (DIR/SYS introuvable)	restaurer DIR/SYS sur le disque.
DIR/SYS not assigned space	

(pas de place affectee a BOOT/SYS)

id

Bien que DIRCHECK soit capable de reparer bien des erreurs, il peut arriver que certains fichiers dont l'information etait erronee dans le directory en soient affectes.

Il est un autre parametre de DIRCHECK, CYLO, qui sert pour les disques double-densite du Modele I, lesquelles ont le cylindre 0 en simple densite. Sur ces disques, le fichier BOOT/SYS n'occupe pas autant de granules qu'il est indique dans la table d'allocation, ce qui resulterait en une erreur 'extraneous or unassigned gran', alors que le disque est parfaitement normal. Ce parametre evite cette fausse indication d'erreur, pour ce seul cylindre.

NOTE :

DIRCHECK va prendre le nombre de surfaces conforme a la situation CONFIG en service pour le lecteur approprie si le parametre GAT est actif. C'est a dire que si le lecteur 1 est CONFIGURE pour SIDES=2 au moment d'un DIRCHECK,GAT, DIRCHECK va ecrire dans le GAT les informations indiquant que le disque est a double face.

Ceci peut etre utilise avantageusement pour convertir des disques double-face style DOSPLUS 3.4/4.0 au format DOSPLUS 3.5.

DISKDUMP

Utilitaire en langage machine destine a afficher et modifier des secteurs d' un disque.

DISKDUMP nomfich. <enter>

Si il n' est pas donne de nom de fichier, DISKDUMP le demandera apres avoir affiche une invite en forme d' asterisque.

Entrez alors le nom du fichier, y compris l' extension et le mot de passe eventuels. DISK-DUMP va alors afficher le premier secteur du fichier. Vous disposez des commandes suivantes:

;	avance d' un secteur
-	recule dd
+	avance au dernier secteur.
=	recule premier
F	cherche un valeur hexadecimale.
G	va au secteur indique.
M	Entre le mode modification.
P	recherche une adresse (dans un programme binaire)
@ xx	remplit xx octets avec 00, a partir de la position du curseur.
@ xxyy	remplit xx octets avec l' octet yy , id

L'affichage d' un secteur aura l' aspect suivant :

```

AF = 4044 -Z---P--
BC = 014C: 30 D2 4A 1E 16 FF 14 D6 03 30 FB C6 03 4F F1 87
DE = 3E74: 20 20 2E 20 20 2E 20 20 2E 20 20 20 53 50 20 20
HL = 3EB4: 20 20 2E 20 20 2E 20 20 40 20 20 20 50 43 20 20
AF' = 0514 ---H-P--
BC' = 2500: C1 CA FE 27 FE C5 CA 9D 41 FE C8 CA C9 27 FE C7
DE' = 7C74: 75 B5 39 C3 3A 33 00 FC 00 00 00 00 FF F6 7F 02
HL' = 6056: 04 08 7E 3C E6 1F 77 08 FD E1 C9 21 36 40 01 01
IX = 401D: 02 66 FD 16 3E 00 8C 4F 02 B6 FB 43 0D 00 84 52
IY = 43CB: 08 FF FF 80 20 00 00 42 00 00 08 00 00 05 00 FF
SP = 41D2: 0A 02 C0 3F B7 FD D5 FD 42 00 1C 40 DD 03 1D 40
PC = 055E: ED B0 C1 EB 18 19 7D E6 C0 6F E5 11 40 00 19 7C
      2940: 2A D8 40 EB DF 28 DA 01 3F 29 C5 AF B6 23 5E 23
      2950: 56 23 C8 44 4D 2A D6 40 DF 60 69 D8 E1 E3 DF E3
      2960: E5 60 69 D0 C1 F1 F1 E5 D5 C5 C9 D1 E1 7D B4 C6
      2970: 2B 46 2B 4E E5 2B 6E 26 00 09 50 59 2B 44 4D 2A
00 00: . . S Y S I 4 . 5 . * * *
00 10: * * * N O T I C
20: E * * * * *
30: T h i s p r o g r a m i s
40: c o p y r i g h t e d b y
50: * . * M i c r o - S y s t e
60: m s S o f t w a r e , I n c
70: . , * . * 1 9 8 3 .
80: A l l r i g h t s r e s e r
90: v e d . * * * *
A0: * * * N O T I C
B0: E * * * * *
C0: . . R , R , R , S w U / . > S 2
D0: J S 2 5 S 2 S . . . C ^ R .
E0: C S . C j S . . C _ S . C . S
F0: . ! S , y D . W , ^ . . .

```

Le nombre a deux chiffres dans le coin en haut a gauche de l' ecran indique le numero physique du lecteur sur lequel se trouve le fichier (pas sa designation).

Juste au-dessous se trouve le numero du bloc physique affiche.

Legerement decale a droite est une colonne de nombres de deux chiffres hexadecimaux. Ils indiquent chacun le numero relatif de l' octet de la premiere colonne de la ligne sur laquelle ils se trouvent.

A droite de cette colonne, est l' information contenue dans le secteur. Elle est affichee en hexa par defaut, en huit groupes de deux octets chacun. Cet affichage hexa peut etre transforme en affichage ASCII en pressant la touche <clear>.

Le secteur precedent peut etre affiche en pressant "-", le suivant par ";". Un ordre qui aurait pour effet de depasser les limites du fichier sera ignore.

Le premier secteur peut etre affiche par "=", le dernier par "+".

DISKDUMP peut afficher n' importe quel secteur, par l' ordre 'G', suivi du numero choisi et de <enter>.

L' ordre 'F' permet de trouver une valeur hexadecimale quelconque dans un secteur. Elle doit etre exprimee en hexadecimal si l' affichage est en hexa, en ASCII dans l' autre cas. Un pave blanc sera affiche a la place trouvee. Pour quitter cet affichage, presser n' importe quelle touche.

Les secteurs peuvent etre modifies en utilisant le mode 'modification', ou l' on entre en pressant la touche 'M' en mode affichage. Un pave blanc apparait en haut a gauche de l' affichage. Il peut etre deplace avec les quatre fleches. La valaure de l' octet qu' il masque est affichee dans la colonne de gauche. Pour modifier cet octet, entrer sa nouvelle valeur dans le mode correspondant a l' affichage. Pour enregistrer la (ou les) modification(s), presser <enter>. <break> les annulera.

L' ordre 'P' peut etre tres pratique pour modifier des programmes binaires (load module format), car ,suivi de l' adresse cherchee, il la trouve, et fait passer DISKDUMP en mode modif. Si il ne peut la trouver, il annoncera 'Invalid data' (donnee invalide), et retournera a l' in-

vite nom de fichier.

L'ordre @ suivi de deux octets en hexa permet de remplir le nombre d'octets egal au premier avec le second. Si ce dernier est omis, il sera egal a 0.

DISKZAP

Cet utilitaire est l'editeur de secteurs. Il peut etre utilise pour afficher, modifier, copier ou verifier des secteurs, comme pour formater des pistes.

DISKZAP

Il n'y a pas de parametre.

Apres son lancement, DISKZAP affiche son menu

```
*  Set      (reglages)
   Fill     (remplissage)
   Copy     (Copie)
   Print    (Impression)
   Verify   (Verification)
   Format    (formatage)
   Display  (affichage et modification)
```

C'est le menu principal, qui donne la liste des options secondaires et vous permet de choisir. Les ordres sont lances en plaçant le curseur en face l'option choisie et en pressant <enter>. Pour chaque lecteur, DISKZAP a certaines valeurs par default :

```
35 (Mod I) ou 40 (Mod III) cylindres.
10 (Mod I) ou 18 (Mod III) secteurs sur la piste 0.
Simple (Mod I) ou double (Mod III) densite sur piste 0
10 (Mod I) ou 18 (Mod III) secteurs sur autres pistes.
Simple (Mod I) ou Double (Mod III) densite id.
```

Toutes ces valeurs peuvent etre changees par l'option SET. L'asterisque qui apparait a gauche des termes du menu est le curseur de controle, qui se deplace avec les fleches verticales. Pour quitter DISKZAP, presser 'Q'.

SET (modification des parametres du disque)

DISKZAP est lance avec le curseur sur cette option. Si les parametres par default conviennent, vous pouvez passer a une autre (NDT pour se faire la main avec ce programme, il vaut mieux utiliser un disque qui peut etre 'plante' sans inconvenient). Pour selectionner une option, placer le curseur en face, et <enter>. Pour SET la premiere question sera:

Drivespec ? (lecteur)

repondre par la designation du lecteur choisi. La question suivante sera:

Cylinder count ? (nombre de cylindres)

repondre par le nombre de cylindres reel (celui du formatage). Puis:

Surface count ? (nombre de surfaces)

La reponse est 1 ou 2 pour les disques souples, ou plus pour les durs. Puis:

CYL 0 sec/trk ? (nombre de cylindres par pistes du cylindre 0)

La valeur par default est correcte pour les disque souples 5 pouces (10 pour Mod I, 18 pour Mod III). Si le disque etudie a un nombre different, le fournir. Puis:

CYL 0 density ? (densite du cylindre 0)

repondre par S pour simple ou D pour double. Il est a noter que sur Modele I, les disques systeme DOSPLUS 3.5 ont la piste 0 en simple densite. C'est necessite par le fait que le chargeur initial, dans la ROM, ne connait que ca. Les disques de donnees sont toutefois formats en double densite.

Apres la definition de la piste 0, vous allez pouvoir definir les memes parametres pour le reste du disque :

Sectors /track ? (secteurs par piste)

Track density ? (densite des pistes).

Vous y repondez comme aux precedentes. Si vous ne voulez pas modifier tous les parametres, vous pouvez interrompre la sequence par <break>. Les reponses fournies sont deja enregistrees. Vous pouvez laisser un parametre inchangé en pressant <enter> au lieu de repondre.

FILL (remplissage des secteurs avec l'octet de son choix)

Permet d'effacer completement le contenu d'un secteur. Les questions sont dans l'ordre :

Drivespec ?

Cylinder ?

Sector ? (secteur)

sector count ? (nombre de secteurs)

Fill data ? (donnee de remplissage)

Repondre a ces questions, en hexadecimal, sauf pour le nombre de secteurs. Exemple :

Drivespec ? 1

Cylinder ? 25

Sector ? 0E

Sector count ? 18

Fill data ? E5.

Une fois lance, DISKZAP va effectuer le travail en affichant les numeros des secteurs qu'il remplit.

Copy (copie de secteurs)

Cette fonction permet de copier des secteurs d'un disque sur un autre, ou a un autre endroit du meme disque. Au lancement de cet ordre, une serie de questions va etre proposee a votre choix :

Drivespec ?

Cylinder

Sector ?

ces deux questions concernent le premier ou seul secteur source.

Drivespec ?

Cylinder

Sector ?

Pour le premier secteur destination.

Sector count ?

Car on peut bien sur copier plusieurs secteurs en une fois, a condition qu'il s'agisse de secteurs consecutifs, au depart comme a l'arrivee.

Si vous desirez copier un disque entier du Modele III, il faudrait fournir les reponses suivantes (le source est le 0, le 1 est destination) :

Drivespec ? 0
Cylinder ? 0
Sector ? 0
Drivespec ? 1
Cylinder ? 0
Sector ? 0
Sector count ? 720

Des cette derniere reponse entree, DISKZAP va commencer la copie, en cherchant a charger le maximum de secteurs a la fois, avant de les copier (cela signifie qu' il va ecraser le reste de la memoire utilisateur). Les numeros des secteurs source et destination sont affiches a mesure de l' avance du travail.

Si DISKZAP rencontre une erreur pendant ce travail, il s' arrete et affiche l' erreur, et demande :

Continue Y or N ?

Si vous repondez Y, il copiera ce qu' il pourra du secteur en cause, et continuera, vous permettant de sauver le maximum, vous permettant ensuite de travailler sans etre ennuye par des secteurs defectueux.

Print (impression de secteurs choisis)

Cet ordre permet d' imprimer le contenu des secteurs choisis (contigus). Les questions seront :

Drivespec ?
Cylinder ?
Sector ?
Sector count ?

Les secteurs sont affiches en meme temps qu' ils sont imprimes. DISKZAP ne verifie pas l' etat de l' imprimante, et attendra le cas echeant qu' elle soit prete (blocage).

Verify (lecture et verification de secteurs choisis)

Permet de verifier les secteurs de votre choix. Il affiche les messages voulus en cas d' erreur, <enter> le fait continuer. Les questions sont:

Drivespec ?
Cylinder ?
Sector ?
Sector count ?
Ignore Data AM ?

Par cette question, DISKZAP vous demande si vous desirez ou non qu' il signale les secteurs du directory, qui ont une marque d' adressage speciale. Si vous repondez 'N', chaque fois qu' il en rencontrera, il affichera :

AM/WRITE FAULT (erreur de marqueur de donnée)

Vous pouvez interrompre la vérification d'un disque en pressant <enter>, qui vous renvoie au menu principal.

Format (formatage d'une ou plusieurs pistes)

Permet de reformatter une ou plusieurs pistes, mais n'est pas destiné à remplacer FORMAT, qui effectue d'autres travaux que le simple formatage. Les questions sont :

Drivespec ?
Cylinder ?
Sector ?
Sector count ?
Interleave factor (facteur d'entrelacement)

Ce facteur permet de choisir l'ordre dans lequel les secteurs sont numérotés sur le disque (ce n'est pas la suite physique), et peut avoir un effet important sur les temps d'accès. Ses valeurs normales sont 2 pour des disques simple densité, et 3 en double densité (5 pouces).

Display (affichage et modification de secteurs)

C'est probablement l'option la plus utilisée et le cœur de l'éditeur de disques. DISKZAP utilise un éditeur plein écran, avec déplacements du curseur dans tous les sens.

Les questions posées sont :

Drivespec ?
Cylinder ?
Sector ?

Après la réponse à la dernière, vous verrez un affichage du genre :

```

00 00: 5F00 0000 0042 4F4F 5420 2020 2053 5953 _...BOOT  SYS
14 10: DD23 DD23 0600 0000 FFFF FFFF FFFF FFFF .#. #.....
02 20: 5E02 3880 0053 5953 3134 2020 2053 5953 ^.;.SYS14  SYS
30: DD23 DD23 0700 1941 FFFF FFFF FFFF FFFF .#. #...A.....
40: 0000 0000 0000 0000 0000 0000 0000 0000 .....
50: 0000 0000 0000 0000 0000 0000 0000 0000 .....
60: 0000 0000 0000 0000 0000 0000 0000 0000 .....
70: 0000 0000 0000 0000 0000 0000 0000 0000 .....
80: 0000 0000 0000 0000 0000 0000 0000 0000 .....
90: 0000 0000 0000 0000 0000 0000 0000 0000 .....
A0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
B0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
C0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
D0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
E0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
F0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
```

En haut à gauche, apparaissent trois nombres: le numéro de lecteur, le numéro du cylindre et le numéro du secteur. Le reste est identique à celui de DISKDUMP. Le mode Modification/correction fonctionne également de la même façon. DISKZAP n'accepte que des valeurs hexadéci-

males en entree, sauf pour les nombres de secteurs et de cylindres.

Si DISKZAP rencontre une erreur de lecture en mode affichage, il va s'arreter et montrer l'erreur. Il vous demandera aussi si vous voulez continuer. Si vous repondez 'Y', il affichera ce qu'il aura pu lire du secteur en cause (si il n'a RIEN lu, ce sera probablement l'affichage precedent). Vous pourrez alors passer en mode correction et faire toutes les corrections possibles avant de le reecrire. Le secteur sera recopie sur le disque, y compris toutes les modifications que vous aurez faites. Il sera ensuite purge des erreurs techniques de lecture, mais sans que cela prejuge de la correction de son contenu. (NDT : il est souvent efficace, pour corriger un secteur, de le lire, et si le contenu semble correct, de passer en mode correction, puis de taper <enter> sans rien changer).

FORMAT

Permet de preparer un disque a recevoir des donnees.

FORMAT :lec (param=expr...)

Note : ces parametres peuvent etre fournis des a ligne d'ordre de DOSPLUS 3.5, Ceux qui ne seraient pas fournis seront demandes par le programme.

Les parametres possibles sont :

- (D)ATE="chaîne" Chaîne de caracteres.
- (P)W="chaîne" Mot de passe choisi pour le disque.
- (N)AME="chaîne" Nom du disque.
- (C)YL=valeur nombre de cylindres auquel vous voulez formater le disque (le materiel doit en etre capable).
- (S)IDES=valeur Nombre de surfaces a formater (1 ou 2).
- (DE)N="chaîne" Densite d'ecriture (Simple ou Double).
- (U)SE="chaîne" Permet d'eviter la question 'Diskette contains data, Use or Not' (ce disque contient des donnees, a Utiliser ou Non). A utiliser avec precaution, car ce qui serait dessus serait irremediablement perdu.

Ces parametres ont les valeurs suivantes par default, elles sont donnees par <enter> en reponse a une question :

Date	00/00/80
PW	vide
NAME	vide
CYLS	40
SIDES	1
DEN	S pour Mod I, D pour Mod III
USE	Non.

Des qu'il a reçu toutes les reponses, FORMAT entame son travail. Il verifie tous les secteurs, et s'il en trouve qui sont defectueux, il verrouille le granule correspondant, pour eviter que les fichiers ne les utilisent, puis ecrit les fichiers BOOT/SYS et DIR/SYS, avant de retourner a DOSPLUS 3.5. Si le formatage se fait avec un seul lecteur, ne pas oublier de remettre le disque systeme avat de repondre a 'Insert system disk'.

HELP

Sert d'aide memoire pour la syntaxe des ordres de la librairie et quelque utilitaires tres utilises.

HELP

HELP <from> ordre <to> fichier/periph.

'ordre' est un ordre de la librairie, ou le nom d'un utilitaire decrit dans l'utilitaire PATCH.

Le programme HELP fournit des informations sur les ordres suivants :

APPEND	ASSIGN	ATTRIB	AUTO	BOOT
BREAK	BUILD	CAT	CLEAR	CLOCK
CLS	CONFIG	COPY	CREATE	DATE
DEBUG	DIR	DO	DUMP	ERROR
FILTER	FORCE	FORMS	FREE	I
JOIN	KILL	LIB	LIST	LOAD
PAUSE	PROT	RENAME	RESET	RS232
SCREEN	SYSTEM	TIME	VERIFY	
BACKUP	CONVERT	DIRCHECK	DISKDUMP	FORMAT
MAP	PATCH	RESTORE	SYSGEN	

Pour afficher l'aide disponible pour l'une de ces commandes, taper HELP suivi de son nom, par ex :

HELP COPY

fera afficher :

```

COPY <FROM> fs/@ds <TO> fs/@ds (param=exp,...)
COPY <FROM> fs <TO> :dr (param=exp,...)
COPY <FROM> :dr <TO> :dr <USING> wildmask (param=exp,...)

DPW='string'  ECHO=switch  INVIS=switch  KILL=switch
MOD=switch    OVER=switch  PROMPT=switch  QUERY=switch
SPW='string'   TINY=switch  NEW=switch   OLD=switch
    
```

Si vous demandez les renseignements concernant un ordre ou un programme étrangers a cette liste, HELP affichera la liste des noms valides.

HELP peut envoyer ses resultats a n'importe quel periph ou fichier. Par exemple :

HELP attrib @pr

Provoque l'impression des informations sur ATTRIB.

MAP

Fournit une liste, par cylindre et secteur, des zones allouees aux fichiers d' un disque.

MAP <from> :lect <to> fich/periph <using> masque (param=exp)

Les parametres sont :

(S)YSTEM inclut les fichiers systeme.
 (I)NVIS id invisibles.
 (H)EX fournit les valeurs en hexadecimal.

MAP permet de connaitre l' affectation de l' espace du disque par fichier. Pour afficher tous les fichiers, entrer :

MAP :lect

lect, etant le numero du lecteur choisi. L' affichage aura l' allure suivante :

```
DOS:3.50 01/26/83
RS/DVR      001,006 - 001,011
TRAP/CMD    001,012 - 001,017
             002,000 - 002,003
FILE/DVR    003,000 - 003,005
DO/DVR      000,006 - 000,011
KI/DVR      000,012 - 000,017
PR/DVR      001,000 - 000,005
```

Chaque nom de fichier est suivi d' un groupe de nombres. Le premier couple indique les numeros du premier cylindre et du premier secteur du fichier; le deuxieme indique ceux du dernier cylindre et du dernier secteur, du fichier ou de son premier troncon. Dans ce deuxieme cas, une autre ligne (ou plusieurs) sera occupee par un autre jeu de nombres fournissant les memes indications pour la premiere extension du fichier (comme TRAP/CMD dans l' exemple ci-dessus).

MAP peut fournir les valeurs en decimal (par default), ou en hexa, si le parametre HEXA est precise. Il peut aussi fournir ses renseignements a un fichier/peripherique quelconque, et accepte les masques, comme COPY ou KILL.

PATCH

Permet de modifier a des fichiers programmes. Il peut utiliser de fichiers de modification, ou etre utilise directement depuis le clavier.

```
PATCH fichier1 fichier2
PATCH fichier1
PATCH fichier1 nommodif (KILL)
```

nommodif estle nom de la modification a enlever.
(K)ILL=option sert a indiquer a PATCH que la modification est a enlever, et non a installer.

La premiere forme est utilisee quand on dispose d' un fichier des modifications a installer (fichier2).

Un fichier de modifications consiste en une ou plusieurs lignes de modifications (63 car. maxi). Une ligne a le format suivant :

A=xxxxH,F=xxxxxxxx,C=xxxxxxxx

Les trois parametres A,C et F peuvent etre places dans n' importe quel ordre, mais doivent tous apparaitre, meme vides.

'A' sert a fournir l' adresse ou la modification doit etre installee, en decimal (defaut ou suffixe D), ou hexa (suffixe H), ou octal (suffixe O), ou binaire (suffixe B). Si l' adresse indiquee par 'A' ne peut etre trouvee, PATCH affiche 'Address not found' (adresse introuvable) et retourne a DOSPLUS.

Le parametre 'F' designe une chaine de caracteres ou de valeurs hexadecimales que PATCH va chercher a l' adresse indiquee avant d' installer la modification. S' il trouve ces octets il continue le travail, sinon il affiche 'String not found' (chaine NON trouvee) et retourne a DOSPLUS. Cette chaine peut etre vide, il n' a pas de controle dans ce cas. Exemple :

A=5481H,F=CD8955,C=000000

ou, si le contenu de l' adresse ne nous interesse pas :

A=5481H,F=,C=000000

Si vous fournissez une chaine hexadecimale, l' espace entre les octets est facultatif. Si vous voulez fournir une chaine de caracteres a 'F', il faut l' encadrer de guillemets ou d' apostrophes (" ou ').

Le parametre 'C' sert a fournir les octets de remplacement. Ils peuvent etre fournis sous les memes formes que pour 'F'. Exemple :

A=655FH,F='bonjour',C="Au revoir"

Il est possible d' inserer des lignes de commentaires, en les faisant preceder d' un point.

Les fichiers PATCH peuvent etre ecrits avec BUILD, ou un programme de traitement de textes (a condition qu' il enregistre son produit en ASCII).

La deuxieme forme permet d' installer la modification a partir du clavier, en entrant des lignes ecrites comme les precedentes. Il faut les taper, puis pour les installer, presser

<break>.

Quand il installe des modifications dans un programme, PATCH affecte un nom a la liste qu'elles constituent. Si elles ont ete prises dans un fichier, le fichier de PATCH prendra ce nom, sinon, ce sera *NONAME*. Ces noms sont importants pour la troisieme fonction de PATCH: annuler une modification, grace au parametre KILL. Il est evident que pour ce faire, il lui faut la liste des modifications installees, ainsi que celles des octets remplaces, qu'il range dans le meme fichier. Si vous lancez l'ordre :

PATCH fichier fichpatch (KILL)

et qu'il ne trouve pas le fichier fichpatch, il renverra le message 'patch not found'.

RESTORE

Permet de recuperer des fichiers detruits par KILL.

RESTORE nomfichier

Il faut savoir que RESTORE e peut pas recuperer n'importe quel fichier : KILL ne fait que modifier un octet dans le DIRECTORY, et liberer la place qu' il occupait. Pour que RESTORE fonctionne, il faut que cette place ni celle occupee par ses extensions, n' ait absolument pas ete modifiee. Dans chacun de ces cas, RESTORE afficherait 'Disk space has been reallocated' (l' espace sur le disque a ete reffecte) et abandonnerait.

Pour faire sa recuperation, RESTORE explore tous les lecteurs, a moins qu' il ne lui en ait ete impose un, et recuperera le premier exemplaire qu' il trouvera. Si un fichier a ete cree et detruit plusieurs fois sur le meme disque, il peut avoir plusieurs emplacements d' etiquette dans le DIRECTORY, et la version recuperee ne sera pas forcement la bonne. Pour recuperer celle-ci, la seule methode est de recuperer celles qui viennent, de les rebaptiser avant de les detruire, et de recommencer jusqu' a obtenir satisfaction.

SYSGEN

Permet de placer DOSPLUS 3.5 sur tout support compatible : disques durs, ou disque souples double face.

SYSGEN :lect nomfichier (param=exp)

Lect est le nom du lecteur portant le disque a traiter
nomfichier est le nom d' un eventuel programme de lancement (bootstrap) a placer sur le disque.

Le parametre est : (X)FER= valeur, et sert a indiquer une adresse ou passer le controle apres l' initialisation. Il est utilise conjointement a un programme de lancement.

SYSGEN permet donc de preparer des disques de caracteristiques quelconques, compatibles avec DOSPLUS 3.5, : disques 8 pouces, disques double face et disques durs. Il est necessaire que le lecteur a traiter soit CONFIGURE correctement, et que le support soit tel qu' il sort du formatage (rien ne doit avoir ete ecrit dessus, sauf DIS/SYS et BOOT/SYS).

La forme la plus simple est :

SYSGEN :lect

Qui place tous les fichiers systeme sur le nouveau disque, qui pourra alors etre utilise comme disque systeme.

Le nom de fichier facultatif permet a l' utilisateur de placer un programme d' initialisation special sur le cylindre 0, secteur 0 du disque prepare, et des son chargement, le programme serait lance a l' adresse fournie par XFER.

TAPE

Transfere un programme systeme de bande sur disque, un fichier objet sur une bande, ou les deplace, puis les reecrit sur le support que choisit l'operateur.

TAPE

Il n'y a pas de parametres.

Une fois charge, TAPE affiche son en-tete, et questionne l'operateur avec un curseur 'X'. Tout ordre valable pour TAPE peut alors etre entre. Ce sont :

B	ajuste la vitesse bande (Mod III seulement)
H	Affiche l'aide memoire.
I	Reinitialise le programme.
L	charge un fichier disque.
M	decrit la zone du programme.
O	decale le programme.
Q	retourne a DOSPLUS.
R	lit une bande systeme.
S	sauve un fichier disque.
W	ecrit un programme sur bande.

'B' permet de choisir la vitesse cassette. La syntaxe est 'B L' pour la vitesse lente (500 bauds) ou 'B H' pour la vitesse rapide (1500 bauds), pour Mod III seulement.

'H' affiche la liste des ordres valables.

'I' reinitialise l'utilitaire TAPE. La syntaxe est 'I', sans parametres. Si vous ne l'utilisez pas, le deuxieme fichier sera charge a la suite du precedent, ce qui signifie qu'il est possible de concatener des programmes. Ils doivent d'ailleurs satisfaire a un certain nombre de criteres: etre relogeables, et ecrits pour cela. La procedure serait la suivante: charger le premier fichier, le decaler pour degager la place, charger le deuxieme, decaler l'ensemble si necessaire, et recopier le total sous un seul nom.

'L' charge un fichier disque. La syntaxe est 'L nomfich'. Il ne fait rien d'autre que charger le programme en memoire.

'M' fournit les adresses de lancement et de chargement de tous les fichiers en memoire. La syntaxe est 'M' ou 'M P'. Le parametre '-P', facultatif, permet l'impression du resultat. Le format est 'adresse-adresse', chaque segment etant affiche separement deux fois. La colonne de gauche concerne le segment original et ne change pas. Celle de droite concerne le segment decale, et reflete l'effte du decalage en vigueur.

'O' va decaler tous les programmes et segments en memoire de la quantite specifiee. La syntaxe est 'O mnnn', ou 'nnnn' est le decalage que l'on veut imposer aux modules. Ce n'est pas la nouvelle adresse. Par exemple si un programme loge en 8C00H est decale de 1000H (O 1000), il residera en 9C00H. TAPE n'utilise que des valeurs hexadecimales. Si vous tapez un nombre de plus de quatre chiffres, seuls les 4 derniers seront pris en compte. Dans 99% des cas un decalage de 1000H va tres bien.

'R' lit une bande systeme. La syntaxe est 'R nomfich' (nomfich est le nom du programme sur bande). Tout ce que fait cette commande est de charger le programme en memoire.

'S' copie sur disque le programme en memoire. La syntaxe est 'S nomfich', ou 'S nomfich-A'. Nomfich est un nom de fichier standard, '-A' est un suffixe facultatif, qui indique a TAPE d'ajouter au programme un module qui inhibe DOSPLUS 3.5 au chargement

du programme. Si par exemple vous devez charger un programme en 4E00H, il faut le sau-
ver avec l'option -A. Le module rebrancherait les vecteurs d' E/S sur la ROM, repla-
cerait le programme correctement, et lui passerait le controle. C' est particuliere-
ment utile pour des jeux sur cassette et autres programmes qu' il n' est pas possible
de reloger quand on les copie sur disque. Le suffixe -A sera souvent necessaire dans
ces cas.

'W' est l' ordre qui ecrit une bande systeme. La syntaxe est 'W nomfich' ou 'W nom-
fich-A'. Le fonctionnement est le meme que ci-dessus.

TAPE peut etre utilise pour reecrire un fichier sur le meme support ou il a ete lu, apres
avoir ete decale. Il est de la sorte plus qu' un simple utilitaire casset- te-disque et
vice-versa.. Il peut par exemple etre utilise pour savoir ou un programme est charge en memoire,
quelle est son adresse de lancement.

TRAP

Permet d'intercepter des erreurs d'E/S disque, pour que l'opérateur choisisse l'action à entreprendre en cas d'erreur.

=====

TRAP sans parametre.

=====

Quand il est actif, TRAP intercepte beaucoup d'erreurs disques avant que elles ne soient signalées au programme qui a demandé l'accès disque. En cas d'erreur, TRAP permet l'affichage habituel, mais affiche aussi l'invite suivante :

Abort, Continue, Retry, Ignore?(Abandonne, Continue, Reessayez, Ignore)

Il suffit de taper la première lettre du mot choisi pour obtenir l'exécution. Il peut être utile, car si il est demandé, le programme appelant ne sera pas informé de l'erreur, et pourra peut être accéder à des portions de fichiers inaccessibles autrement.

TRAP, une fois lancé, reste actif jusqu'à la prochaine initialisation du système, ou au chargement d'un fichier de configuration qui ne le contient pas.

CO-DIR 2.1 - Directory controle par le curseur, version 2.1
Brand

Copyright (C) 1982 par Hunt K.

Picotrin technology Inc.
3531 San Castle Bld.
Lantana, Fl. 33462

TABLE DES MATIERES

Chapitre 1

1.1 INTRODUCTION.....	3/28
1.2 NOTATIONS.....	3/28
1.3 SONORISATION.....	3/28
1.4 MOT DE PASSE.....	3/28
3/29	

Chapitre 2

2.1 MISE EN PLACE DU SYSTEME.....	3/29
2.2 PROGRAMMES.....	3/29
2.3 UTILISATEURS MONOLECTEURS.....	3/29
2.4 UTILISATEURS A PLUSIEURS LECTEURS.....	3/29
2.5 CONFIGURATION DU SYSTEME.....	3/29
2.6 TRANSFERT DE CO-DIR 2.1 SUR VOTRE DISQUE.....	3/29
2.7 REBAPTEME DE CO-DIR 2.1.....	3/29
3/29	

Chapitre 3

3.1 OPERATION.....	3/29
3.2 UTILISATION DE CO-DIR 2.1.....	3/29
3.3 CONTROLE DU CURSEUR.....	3/30
3.4 TAMPON DIRECTORY DE CO-DIR 2.1.....	3/30
3.5 TAMPON PLEIN.....	3/30
3.6 ARRET DE L' HORLOGE.....	3/30

Chapitre 4

4.1	GENERALITES SUR LES ORDRES.....	3/32
4.2	ORDRES.....	3/33
4.3	DESCRIPTION DES ORDRES.....	3/33
4.4	<ENTER> - EXECUTION D' UN FICHIER.....	3/33
4.5	<A> - DIRECTORY DE CO-DIR AVEC LES OPTIONS A,I.....	3/33
4.6	<C> - COPIE D' UN FICHIER SUR LE LECTEUR DESIGNE....	3/33
4.7	NUMERO DE LECTEUR PAR DEFAUT.....	3/34
4.8	EXEMPLE DE COPIE.....	3/34
4.9	<D> - DIRECTORY DE CO-DIR.....	3/34
4.10	<F> - FREE (LIBRES).....	3/34
4.11	<G> - AFFICHAGE DES ZONES AFFECTEES.....	3/35
4.12	<H> - HELP (AIDE MEMOIRE).....	3/35
4.13	<J> - JOB CONTROL (CONTROLE DU TRAVAIL).....	3/35
4.14	<K> - KILL (DESTRUCTION) D' UN FICHIER.....	3/35
4.15	EXEMPLE.....	3/36
4.16	<L> - LISTAGE D' UN FICHIER.....	3/36
4.17	<M> - MODIFICATION D' UN FICHIER.....	3/36
4.18	<O> - CHARGEMENT D' UN FICHIER.....	3/36
4.19	<P> - IMPRESSION D' UN FICHIER.....	3/36
4.20	<R> - REBAPTEME D' UN FICHIER.....	3/36
4.21	<S> - RECHERCHE D' UN FICHIER, OU D' UNE EXTENSION.....	3/36
4.22	RECHERCHE PAR NOMS.....	3/36
4.23	RECHERCHE PAR EXTENSION.....	3/36
4.24	TAMPON DE RECHERCHE.....	3/37
4.25	<Z> - MISE A 0 D' UN FICHIER.....	3/37
4.26	MOTS DE PASSE.....	3/37

Chapitre 5

5.1	TRUCS ET COMMENTAIRES.....	3/37
5.2	DERNIERE PAGE.....	3/38
5.3	VITESSE DE CHARGEMENT.....	3/38
5.4	APERCU TECHNIQUE.....	3/38

1.1 INTRODUCTION :

CO-DIR 2.1 - directory oriente curseur Version 2.1

CO-DIR 2.1 est un programme utilitaire qui vous permet de lancer en tapant une ou deux touches (pas besoin de taper de noms de fichiers) la plupart des ordres de DOSPLUS concernant un fichier sur disque. CO-DIR 2.1 vous permet de traiter les fichiers, comme ils sont affichés, à l'aide d'une paire de curseurs clignotants. Pour utiliser CO-DIR 2.1, tapez simplement CODIR au lieu de DIR ou CAT.

Les commandes SED connues de CO-DIR 2.1 et ses commandes propres sont : Copy, Directory, Free, Help, Do (fichier DO), Kill, List, Load, Modify, Print, Rename, Search, et Zero. Chacune d'elles est décrite dans la documentation.

CO-DIR n'est pas un programme d'utilisation accidentelle. Vous l'utiliserez à chaque initialisation, parce que la plupart des fonctions de DOSPLUS sont exécutables à partir de CO-DIR plus vite et plus facilement.

Toutes les pages que vous voyez pendant le listage du directory resident en mémoire, ce qui permet de les faire defiler comme un traitement de texte. Presser <shift> <fleche haut> déplace le curseur en haut à gauche de l'écran, même s'il faut changer la page qui est sur l'écran.

Toutes les options, sauf <enter> affichent une question (réponse 'Y' ou 'N', ou NO de lecteur) sur la ligne d'ordres (la dernière en bas de l'écran), par ex. 'Copy to Which drive' (copie sur quel lecteur).

1.2 Notations du manuel :

Toutes les majuscules ou les mots entre < et > représentent des touches uniques du clavier. Si il y en a deux ensemble, il faut les presser simultanément. Par ex. :

<shift><fleche en haut>

1.3 Sonorisation :

Pour tirer parti de la sonorisation facultative de CO-DIR, il faut être équipé d'un MINI-AMPLIFICATEUR (Radio-Shack 277-108 ou équivalent), branche à la grosse prise grise (AUX) du cordon de magnétophone. Cette sonorisation émet des sons graves si la commande a été exécutée, des sons aigus signalent soit une commande mal entrée, soit que CO-DIR n'a pas pu exécuter l'ordre. Des sons sont aussi émis pour KILL, PRINT ET ZERO.

NOTE: il faut lire la section tampon de ce document avant de se lancer dans CO-DIR.

1.4 MOT DE PASSE :

Si un fichier est protégé par un mot de passe, il est possible d'y accéder en maintenant la touche <shift> en choisissant votre option, par ex. si vous pressez <shift><L>, CO-DIR demandera le mot de passe à la ligne d'ordres, puis l'ajoutera au nom du fichier avant que vous ne le listiez.

2.1 PREPARATION DU SYSTEME.

2.2 programmes : CO-DIR 2.1 contient 2 fichiers: CODIR/CMD, CODIR2/MOD.

2.3 UTILISATEURS MONOLECTEURS :

Si vous n'avez qu'un lecteur, vous devrez recopier CODIR/CMD (3 granules), et CODIR2/MOD (2 granules).

2.4 UTILISATEURS A PLUSIEURS LECTEURS :

vous pouvez laisser le programme sur n'importe quel disque pour l'ajuster a votre systeme.

2.5 CONFIGURATION DU SYSTEME :

Lancer le programme basic CODIR2/MOD et repondre aux questions, ce qui vous permettra entre autres d'activer ou desactiver la plupart des commandes. Vous aurez ainsi une version de CO-DIR/CMD adaptee a votre systeme et a vos besoins.

2.6 TRANSFERT DE CO-DIR 2.1 SUR VOTRE DISQUE SYSTEME :

Bien qu'il soit possible d'utiliser CO-DIR 2.1 quelque soit le disque sur lequel il est copie, il vaut mieux l'avoir sur le disque systeme.

Le seul programme a transferer est CO-DIR/CMD, CODIR2/MOD est rarement utilise.

2.7 REBAPTEME DE CO-DIR 2.1 :

CO-DIR vous est fourni sous le nom CO-DIR/MOD. Vous pouvez lui donner n'importe quel nom (en gardant l'extension /CMD), comme D/CMD, SALUT/CMD, etc...

Si vous le rebaptisez, essayez de lui donner un nom commençant par D, car la command <D>i-rectory de CO-DIR est activee par 'D'. Cela vous evitera d'avoir a presser 2 touches pour la meme fonction : Si votre nom etait 'C/CMD' par exemple, vous pourriez taper <C> quand CO-DIR est actif, croyant lire un Directory, et en fait lancer une copie.

Note : Il ne faut pas modifier la notice de Copyright affichee au chargement, cela entrainerait des difficultes de fonctionnement, et il vous faudrait reinitialiser la machine, CO-DIR ne peut etre utilise si cette ligne est modifiee.

Vous devez etre maintenant pret a utiliser CO-DIR 2.1.

3.1 OPERATION :

NOTE SPECIALE

CO-DIR 2.1 est prevu pour n'etre utilise qu'a partir du DOSPLUS. N'essayez pas de l'utiliser a partir du BASIC.

NOTE SPECIALE

Normalement, pendant l'execution d'un ordre CAT, l'invite DOSPLUS apparait quand tous les noms de fichiers ont ete affichees, ou quand vous avez presse <BREAK>.

3.2 UTILISATION DE CO-DIR 2.1 :

Pour utiliser CO-DIR 2.1, il suffit de taper CODIR (ou le nom que vous lui avez donne) au lieu de CAT ou DIR, avec les memes parametres :

```
CODIR (I)
CODIR :2
```

Si vous tapez CODIR au lieu de DIR ou CAT, un Directory va etre affiche, normal sauf qu'un petit bloc graphique va apparaitre en haut a gauche de l'ecran, et y rester jusqu'a la fin de

l' affichage. De courts bips seront emis par la sortie son, si elle est active, a l' affichage de chaque ligne. Le petit bloc graphique montre que le tampon de CO-DIR 2.1 est actif, et que toutes les informations sont rangees dans un tampon memoire, pour vous permettre d' explorer par la suite le tableau entier, meme s' il ne tient pas sur l' ecran.

Une fois la lecture terminee, u lieu de l' invite normale de DOSPLUS apparaitra sur la derniere ligne de l' ecran le message :

CO-DIR 2.1 - Cursor oriented DIR. (C) 1982 by Hunt K. Brand.

Deux curseurs vont clignoter sur le premier nom de la liste. CO-DIR est pret a recevoir des ordres.

Pour quitter CO-DIR et retourner a DOSPLUS, taper <break> ou <clear> pendant que les curseurs clignotent sur un nom de fichier. S' il n' y avait pas de noms, CO-DIR retournerait au SED.

3.3 Controle du curseur :

Une fois CO-DIR actif, vous pouvez deplacer la paire de curseurs en utilisant les fleches. <shift><fleche haut> ramene les curseurs sur le premier nom de la premiere page. L' ordre SEARCH (recherche) permet de trouver un fichier particulier.

Les fichiers detruits en utilisant l' option (K)ILL ne peuvent plus etre traites, et les curseurs ne s' arreteront normalement pas dessus.

Pour les y obliger, presser la barre d' espace en meme temps que les fleches. C' est utile aussi pour mettre les curseurs sur un nom contigu a un fichier detruit ou un fichier systeme.

3.4 TAMPON DIRECTORY :

Quand CO-DIR est active, un tampon memoire est ouvert, et le reste jusqu' a la fin, manuelle ou normale du listage du directory. Un petit bloc graphique apparait dans le coin superieur droit de l' ecran et des bips sonores sont emis vers la sortie son. chaque nom de fichier est range dans ce tampon pour un usage ulterieur.

A la fin du listage, une tonalite basse et breve indique que le tampon est ferme et le bloc disparaît.

Une procedure ordinaire consiste a tenir <enter> enfoncee jusqu' a la fin du listage (il est toujours possible de retourner lire un nom n' importe ou). Le tampon va se fermer et CO-DIR attendra que vous relachiez <enter> pour passer a l' etape suivante. A partir du moment ou les curseurs clignotent, <enter> fonctionnera comme indique dans le chapitre ORDRES.

Le tampon peut etre ferme manuellement en pressant <break>, ce qui arrete aussi le listage du directory. Le DOS peut continuer son listage, jusqu' en bas de page eventuellement, mais le contenu du tampon sera limite a ce qu' il etait a ce moment la. Cela ne risque pas de provoquer de difficultes.

3.5 TAMPON PLEIN :

Le tampon est limite a la plus haute adresse disponible indiquee par HIMEM . Si le nombre de noms est tel que cette capacite soit depassee, le tampon se fermera automatiquement, limitant ainsi la liste. Comme vous etiez probablement a la recherche d' un fichier particulier, et voulez continuer la recherche jusqu' a l' avoir trouve, meme apres la fermeture du tampon, CO-DIR va proceder comme suit :

Après la fin du listage du directory, CO-DIR va transferer le dernier groupe de noms (que vous avez vu sur l' ecran après l' arret) au sommet du tampon, ce qui va probablement causer la perte de quelques noms déjà enregistrés, mais va permettre de traiter les noms de la dernière page, ainsi que la plupart de ceux qui avaient été rangés avant la fermeture du tampon.

3.6 ARRET DE L' HORLOGE :
 Le tampon étant constamment mis à jour (écriture des noms, modification de noms sur l' écran, etc...), l' horloge devrait être arrêtée. Si elle ne l' est pas, CO-DIR la sauvera dans le tampon chaque fois que vous faites défiler la liste sur l' écran, écrasant au passage quelques parties de noms de fichiers !!! Il vaut donc mieux s'assurer que l' heure n' est pas affichée avant d' entrer CO-DIR.

4.1 RESUME DES ORDRES :

TOUCHE	FONCTION
<ENTER>	EXECUTION du fichier - si pas /CMD suppose programme basic, charge basic et essai de lancer.
<ESPACE>& <ENTER>	Force CO-DIR a executer un fichier comme si il avait l' extension /CMD
<BREAK>ou <CLEAR>	Sortie de CO-DIR (les curseurs doivent etre sur un nom de fichier).
<A>	lit le directory du lecteur designe, avec les options A et I.
<C>	Copie le fichier sur le lecteur designe.
<D>	Lit le directory du lecteur designe, <shift><D> ajoute les fichiers invisibles.
<F>	FREE demande le numero du lecteur.
<H>	Affiche l' aide memoire (<enter> A C D F G H J K L M O P R S Z.
<J>	Controle du travail. Active un fichier DO comme si l' on tapait DO nomfich.
<K>	KILL detruit le fichier
<L>	Liste sur l' ecran (qui est restaure apres).
<M>	MODIFY (modifie) un fichier. Utilise DISKDUMP. CO-DIR ajoute automatiquement le nom du fichier a traiter.
<O>	charge le fichier, charge les fichiers /CIM ou /CMD en memoire.
<P>	Imprime le fichier.
<R>	rebaptise le fichier. le nouveau nom sera tape sur la ligne d' ordres.
<S>	SEARCH (cherche) un nom ou une extension.
<V>	VIEW (montre) la notice de copyright et de licence, restaure l' ecran.
<Z>	Execute CLEAR sur le fichier.

4.2 ORDRES

Il existe 14 ordres affichés par l'option <H>ELP (qui est l'un d'entre eux). Les ordres de sortie et d'exécution seront décrits d'abord, puis les autres ordres seront décrits dans l'ordre alphabétique de la touche qui l'active.

Pour effectuer un travail quelconque sur un fichier, il faut d'abord placer les curseurs sur son nom.

4.3 DESCRIPTION DES ORDRES:

La ligne "réponse attendue" sous le nom de chaque ordre dans ce chapitre indique ce que CODIR attend comme réponse quand vous avez pressé cette touche de commande. La ligne "sortie" indique la touche qui peut permettre l'annulation de cet ordre.

Tous les ordres, sauf <ENTER> - exécute, <J>OB CONTROL et <L>OAD, permettent à CODIR de rester actif, de sorte que vous pouvez traiter plusieurs fichiers sans avoir à le recharger pour chaque fonction.

4.4 <ENTER> EXECUTION D'UN FICHIER

Réponse attendue : aucune.

Sortie : rien

Pour exécuter un programme (ce ne doit pas être un fichier de données), il suffit de presser <ENTER>. Si le fichier n'a pas été éliminé (marqué par 'X', voir KILL), et n'est pas un fichier /SYS, l'écran se vide, le nom de ce programme est affiché en haut à gauche de l'écran, CODIR se désactive et ce programme est chargé et lancé.

Si le programme n'a pas l'extension /CMD, CODIR suppose que c'est un programme BASIC, et chargera BASIC avec trois fichiers (ou le nombre qui aura été fixé avec CODIR21/MOD) avant de lancer ce programme.

Pour forcer un programme qui n'a pas l'extension /CMD à être lancé comme tel, il suffit de presser la barre d'espace en même temps que <ENTER>.

Si un fichier a un mot de passe, pressez <SHIFT> <ENTER> pour pouvoir entrer le mot de passe. CODIR l'ajoutera au nom du fichier.

4.5 <A> - DIRECTORY CO-DIR AVEC LES OPTIONS A, I.

Réponse attendue : <ENTER> ou NO lecteur.

Sortie : toute touche sauf <espace>

L'action sur <A> provoquera l'affichage du texte suivant sur la ligne d'ordres:

CODIR directory of which drive (d)

(Directory CODIR de quel lecteur (d))

taper le NO voulu, ou <ENTER> pour utiliser le Numéro par défaut (voir <C>). L'écran se vide, et le directory demande s'affiche, avec les fichiers invisibles et l'affectation (A,I). Si le disque est vide, retour au SED.

4.6 - <C> COPY FICHIER SUR LECTEUR DESIGNÉ

Réponse attendue : <ENTER> ou un numéro de lecteur autre que le numéro présent (celui du directory affiché), car on ne peut copier sur le même lecteur (il n'y a pas de changement de nom).

Sortie : toute touche, sauf <espace> ou un numero de lecteur.

Un petit bloc graphique apparait a cote du nom du fichier copie, le message suivant s'affiche sur la ligne d'ordres :

Copy to which drive (d)? (copie sur quel lecteur)

Presser le numero choisi, ou <enter> pour utiliser le N0 par default. Si vous avez designe un lecteur actif, CODIR va commencer a copier le fichier designe sur ce lecteur, avec le meme nom. Le petit bloc graphique a cote du nom du fichier sera remplace par le numero du lecteur destination et la ligne d'ordre affichera l'ordre en cours d'execution. Si la copie reussit, un son grave sera emis.

4.7 NUMERO DE LECTEUR PAR DEFAULT.

Quand vous choisissez une option qui necessite un numero de lecteur en reponse, la derniere partie de la ligne d'ordres sera :

.... which drive (d)? (.... quel lecteur (d)?)

Ces options sont :

<C>OPY, <D>IRECTORY, <A>, <K>ILL et <F>REE.

Il faut remplacer le '(d)' par un numero de lecteur correct, c'est a dire au plus egal au nombre qui a ete fixe avec CODIR21/MOD. Presser <enter> revient a prendre ce nombre 'd' comme reponse. Presser un autre nombre fera de ce dernier le nouveau default.

Presser un numero invalide fera emettre un son aigu pour indiquer l'erreur et annuler l'ordre.

4.8 EXEMPLE DE COPIE

Si vous avez affiche le directory du lecteur 0 et voulez copier le fichier juill/pay sur le lecteur 1, suivez la procedure suivante: placez les curseurs sur JUILL/PAY, pressez <C> et repondez '1' a la question sur le lecteur.

4.9 <D> - DIRECTORY CO-DIR

<SHIFT> <D> - DIRECTORY CO-DIR AVEC FICHIERS INVISIBLES

Reponse attendue : <enter> ou N0 lecteur

Sortie : toute touche sauf <espace>

Pressez D pour un directory normal, <shift> <D> pour voir les fichiers invisibles. Le message suivant apparaitra:

CO-DIR Directory of which drive (d)?

Pressez le numero desire (les espaces ne sont pas pris en compte), ou <enter> pour utiliser le numero par default.

4.10 <F>REE

Affichage de la carte des granules libres d'un lecteur

Reponse attendue : <enter> ou N0 lecteur

Sortie : toute touche sauf <espace>

Le message suivant va s' afficher:

Free map of which drive (d)?
(carte d' occupation de quel lecteur(d)?

La designation d' un lecteur legal fera afficher le resultat et emettre une tonalite grave.
Le message suivant s' affichera :

Press <BREAK> or <CLEAR> for DOS, any other key for CO-DIR.
(Pressez <break> ou <clear> pour le SED, toute autre pour CO-DIR).

4.11 <G> - AFFECTATION DES GRANULES, AFFICHAGE DES TAILLES DES FICHIERS

Reponse attendue : <Y>
Sortie : tout autre touche.

Cet ordre affiche la taille d' un fichier en Koctets, ce qui peut etre utile avant de copier un fichier sur un autre disque.

4.12 <H>ELP, AFFICHAGE DES TOUCHES D' ORDRES.

Reponse attendue : neant
Sortie : neant

En pressant 'H' l' affichage suivant apparait:

Choix: <SHIFT> = Password:<ENTER> A C D F G H J K L M D P R S V Z

Dans tous les cas, tonalite basse, les curseurs continuent a clignoter a la meme place.

Note: Les options neutralisees avec CODIR21/MOD n' apparaissent pas.

4.13 <J> - ACTIVATION DE FICHIER DO

Reponse attendue : <Y>
Sortie : toute autre touche.

Cet ordre permet d' activer un fichier DO. Le message 'Do (Activate) file?' (Activation du fichier DO?) apparait sur la ligne d' ordres.

Repondre <Y> fait vider l' ecran et afficher en haut a gauche: "Do nomfich"

ou 'nomfich' est le nom du fichier active. CODIR se desactive et le fichier est execute.

4.14 <K> - ELIMINATION D' UN FICHIER

Reponse attendue : <Y>
Sortie : toute autre touche

A l' envoi de cet ordre, une tonalite aigue est emise, (comme avec <P> et <Z> pour attirer l' attention sur une manoeuvre delicate).

Un bloc graphique apparait en haut a droite du fichier concerne. Presser <Y> entraine l' execution de l'ordre, une autre touche le fait abandonner.

Après une elimination, une asterisque est mise a la place de ce bloc pour indiquer que ce fichier n' existe plus.

4.15 EXEMPLE DE 'KILL'

si dans le directory du lecteur 1, qui est affiche, vous voyez un fichier 'paye/fin' que vous voulez eliminer, placez y les curseurs, pressez <K>, puis <Y>. CODIR va essayer d' eliminer ce fichier de ce lecteur. Si tout va bien, une tonalite grave sera emise et une asterisque sera placee a droite de son nom, les curseurs descendront automatiquement si possible sur le nom suivant. Il ne sera plus possible de prendre aucune option sur ce fichier, et les curseurs le sauteront si vous les deplacez dans cette zone.

4.16 <L> - LISTAGE D' UN FICHIER

Reponse attendue : <Y>
Sortie : toute autre touche

CODIR utilise la fonction LIST de la librairie, pas du BASIC. Le message 'List file?' (listage du fichier?) est mis sur la ligne d' ordres. <Y> provoque l' effacement de l' ecran, l' affichage de 'List nomfich' en haut a gauche, puis la liste du fichier demande.

4.17 <M> MODIFICATION DE FICHIER

Reponse attendue : <Y>
Sortie : tout autre touche.

Cette option utilise DISKUMP/CMD, qui doit donc exister sur l' un des lecteurs.

4.18 <O> CHARGEMENT D' UN FICHIER

Reponse attendue : <Y>
Sortie : tout autre touche.

Activation de la commande LOAD de la librairie, apres desactivation de CODIR.

4.19 <R> - REBAPTEME D' UN FICHIER.

reponse attendue: nouveau nom.
sortie: <BREAK>, <CLEAR>, <SHIFT fleche gauche>, ou <ENTER> si rien n' a ete tape.

Apres avoir presse <R>, le message suivant apparait sur la ligne d' ordres :

Rename to /

Les / representent deux traits verticaux qui delimitent la longueur du nom a fournir.

Seuls les caracteres valides sont acceptes, les autres provoquent l' emission du signal sonore d' erreur.

Presser <ENTER> provoque l' echange de l' ancien nom par le nouveau dans la liste et le rebaptisme du fichier. Si aucune erreur n' a ete detectee, le signal 'tache accomplie' est emis, et un 'R' vient se mettre a droite du nom, pour signaler que ce fichier a ete rebaptise.

4.21 <S> - RECHERCHE (A PARTIR DE LA POSITION DES CURSEURS) D' UN NOM OU D' UN EXTENSION DANS LE DIRECTORY AFFICHE.

Reponse attendue:<ENTER> ou chaine a rechercher. Pour faire la recherche dans le directory entier, pressez d' abord <SHIFT fleche en haut>.

Sortie <BREAK>, <CLEAR>, <SHIFT FLECHE GAUCHE>

Cet ordre permet de rechercher un fichier ou une extension particuliere.

La ligne d' ordres affiche le message:

Search for /
(recherche de)

Si elle est trouvee, les curseurs cligotent sur son nom.

Vous pouvez pratiquer deux sortes de recherche :

Par nom (tout ou partie)
Par extension , (tout ou partie).

Pour les recherches par fraction de mot, remplacez les caracteres non fournis par des '?'.
Un recherche par extension est indiquee par '/' devant le mot a trouver. Exemple :

?F???? <ENTER>

trouvera le premier fichier dont le nom comporte un 'F' comme deuxieme caractere.

4.24 TAMPON DE RECHERCHE

CO-DIR se rappelle l' objet de votre derniere recherche, qui reste dans le tampon de recherche. Les curseurs s' etant arretes sur le premier nom trouve, un nouveau <S> et <ENTER> prolongera la recherche jusqu'au suivant, et ainsi de suite jusqu' au dernier.

4.25 <Z> - VIDAGE D' UN FICHIER (MISE A ZERO).

Reponse attendue <Y>
Sortie toute autre touche.

Cet ordre execute l' ordre de la librairie <CLEAR> sur le fichier designe, avec '0'. Il doit donc etre utilise avec grande prudence, car le contenu du fichier traite devient absolument irrecoverable.

La ligne d' ordres affiche :

Clear (Zero !) file????

Si la reponse est <Y> l' ordre est execute, et un 'Z' vient a droite de son nom.

4.26 MOTS DE PASSE :

Si vous avez besoin d' un mot de passe pour acceder a un fichier, il suffit de maintenir pressee <SHIFT> pendant que vous pressez la touche de l' ordre choisi. Le message suivant apparait sur la ligne de commande :

Password :/

Entrez le mot de passe. L' invite normale apparaitra ensuite sur la ligne de commande. Si vous activez la fonction, ce mot sera ajoute au nom du fichier, sinon il sera oublie.

5.1 TRUCS ET COMMENTAIRES GENERAUX

5.1 DERNIERE PAGE :

Pour deplacer les curseurs sur la derniere page du tampon directory, utiliser l'option <S>, suivie de '<' ou '>' et <ENTER>. Ces caracteres ne risquant pas d'etre trouves, les curseurs clignoteront sur le dernier nom de la derniere page.

5.2 VITESSE DE CHARGEMENT

Si un fichier est ecrit sur des pistes consecutives, il peut etre charge un peu plus vite. C'est vrai pour CO-DIR aussi. C'est pourquoi, si vous voulez qu'il se charge vite, il faudra le charger quand le disque est presque vide, car c'est la qu'il a le plus de chances d'etre compact.

5.3 APERCU TECHNIQUE :

CO-DIR est charge en 8000H, le tampon directory commence en 9000H et utilise un minimum de 1 kilooctets.

Table des matieres

Section	Page
I. Job Control language (JCL)	
(Langage de commande).....	4/2
Mise en route.....	4/2
La file d' attente du clavier.....	4/2
II. Structure du programme JCL).....	4/3
Ordres JCL.....	4/3
Variables.....	4/3
"" speciales.....	4/4
Etiquettes JCL.....	4/5
Remarques JCL.....	4/5
III. Le jeu d' instructions du JCL	4/6
/DOS.....	4/6
/EXIT (sortie).....	4/6
/READ (lit).....	4/6
/TYPE (tape).....	4/7
/PRINT (imprime).....	4/8
/IF (si).....	4/8
/GOTO (va a).....	4/9
/QUEUE (file d' attente).....	4/9
/LOAD.....	4/10
/PURGE.....	4/11
/JUMP (saute a).....	4/11
/RESUME (reprends).....	4/12
/CANCEL (annule).....	4/12
/RUN (execute).....	4/12
/DEBUG.....	4/13
/VOFF (arrete visu).....	4/13
/OPTION.....	4/14
DSP.....	4/14
JCL.....	4/14
QUEUE.....	4/14
QBYTE (inverseur queue).....	4/14
DQ.....	4/15
CHR.....	4/15

I. JCL (Langage de commande du travail).

DOSPLUS contient un langage auxiliaire appele JCL (Job Control Language), qui est un langage special destine a controler le fonctionnement du systeme d'exploitation de la machine ou des programmes d'application. JCL peut permettre l'execution de travaux complexes sans intervention de l'operateur, et peut constituer pour l'utilisateur la base d'une introduction aisee a des programmes plus complexes. Il peut meme etre employe pour ecrire des miniprogrammes utilitaires a utiliser au niveau commande du systeme.

Lancement de JCL :

Il doit etre installe dans DOSPLUS 3.5. Son lancement utilise la syntaxe generale suivante :

JCL (PROC=XX,QUEUE=XX)

Le parametre facultatif PROC sert a determiner la taille de la zone des procedures. C'est la zone memoire qui contient le texte du programme JCL. Sa valeur par default est de 768 octets. Sa valeur maxi est de 4096 octets.

Le parametre facultatif QUEUE controle la taille de la file d'attente du clavier (voir plus loin). Sa valeur par default est de 256 octets, son maximum est de 4096.

Pour executer une procedure JCL, la syntaxe est :

EX procjcl <exp1> <exp2> <exp3>

procjcl est le nom de la procedure a executer (l'extension /JCL est supposee). Le nombre de parametres a taper ensuite est libre, Il peuvent, si on le desire etre lus par la procedure.

La file d'attente du clavier (QUEUE)

est le coeur de JCL, C'est une zone memoire entretenue par le systeme JCL, qui possede des instructions pour y placer des donnees et les y retrouver. Son but est de remplacer les caracteres tapes au clavier par ceux qu'elle contient. Cela veut dire que tout programme qui attendrait des donnees venant du clavier, les trouvera dans la queue a la place. Quand elle est vide, elle est remplacee par l'entree au clavier. Supposons par exemple que la queue contienne les donnees suivantes :

```
BASIC <enter>
LOAD"CALC/BAS" <enter>
LLIST <enter>
```

Si la machine est au niveau SED, elle attend une ligne de donnees du clavier. Comme il y a des donnees dans la queue, JCL va les fournir en priorite. DOSPLUS va recevoir les caracteres 'BASIC', suivis d'un retour chariot et donc charger BASIC. Puis Basic va attendre des caracteres du clavier. La queue n'etant pas vide, JCL va lui envoyer 'LOAD"CALC/BAS" <enter>'. Il va donc charger ce programme, et attendre une nouvelle entree clavier. JCL peut encore la lui fournir, c'est 'LLIST <enter>'. Cet ordre va etre execute, puis BASIC va encore attendre une entree au clavier, mais comme la queue est vide, JCL ne la fournira plus, et il faudra la taper, c'est a dire pratiquement reprendre la main.

Cet exemple illustre le fonctionnement de la queue. Comme JCL permet a l'utilisateur de decider du contenu de cette queue, il peut affranchir l'operateur de la frappe de sequences invariables, et comme de plus il a des capacites de decision, il peut travailler avec un operateur, et modifier son action en fonction des reponses de celui-ci.

II. - Structure des programmes JCL.

Comme tout langage, JCL possède des composants et une structure propres. Tous les programmes JCL sont composés d'une ou plusieurs instructions. Une instruction peut être un ordre JCL, une affectation de variable, ou une remarque.

Les fichiers de programme JCL consistent en lignes, numérotées ou non. L'éditeur de programmes BASIC peut être utilisé pour écrire des programmes JCL (utiliser la syntaxe 'save"sss", A'). La première ligne d'un programme JCL doit contenir le nom du programme JCL, qui doit être le même que celui du fichier. Par exemple le programme appelé EFFACE/JCL pourrait contenir les lignes suivantes :

```
EFFACE
/TYPE ENTER DRIVE NUMBER
/READ $D
/DOS KILL */TXT:$D,E
/EXIT
```

Si la première ligne ne correspond pas au nom du programme, JCL signalera une erreur. Cette ligne d'identification de la procédure peut être utilisée pour lire les valeurs de toutes les variables qui sont présentes sur la ligne de commande qui passe le contrôle à JCL. Par exemple si la commande :

```
EX PURGE AVR MAI JUI JUL AOU
```

est exécutée, Un programme JCL peut lire les variables suivant le nom du programme JCL par un READ implicite en ajoutant des noms de variables dans la ligne d'identification de la procédure :

```
PURGE $FIC1 $FIC2 $FIC3 $FIC4 $FIC5
```

Si JCL exécute cette procédure, les valeurs \$FIC1 à \$FIC5 contiendront les valeurs introduites sur la ligne de commande.

Il est important de noter que les mots de commande JCL doivent être suivis d'un espace, et les opérateurs JCL doivent être précédés et suivis d'un espace.

Ordres JCL

Le langage de contrôle des ordres JCL de DOSPLUS 3.5 comprend 17 ordres (ils seront décrits dans la section III). Ils informent JCL de l'action à entreprendre, qui peut modifier le contenu de la file du clavier, fournir ou recevoir des données au/du monde extérieur, changer le déroulement du programme, etc. Tous les ordres doivent être précédés d'un '//'. Beaucoup d'ordres peuvent accepter un argument, d'autres en exigent un, qui doit suivre le nom de l'ordre suivi d'un espace.

Variables JCL

JCL permet l'usage de variables. Elles doivent toutes être précédées du signe '\$', et ce sont toutes des chaînes de caractères. Leurs noms peuvent être composés de n'importe quelle combinaison de lettres et de chiffres. Ces variables peuvent contenir de 0 à 8 caractères chacune.

Avant d'exécuter une ligne de programme, JCL explore la ligne pour repérer les variables éventuelles. Quand il en trouve une, il l'ôte de la ligne et la remplace par sa valeur. Par e-

xemple, supposons que la variable \$NOMFICH ait la valeur 'AGENDA'. Les instructions JCL :

```
/DOS LIST $NOMFICH
```

Auraient pour effet de faire lire par JCL :

```
/DOS LIST AGENDA
```

Les valeurs peuvent etre donnees aux variables soit par l'ordre /READ, soit par affectation par l'operateur '='. Par exemple la ligne JCL :

```
$DSCMD = RSZ32
```

Affecterait la valeur 'RSZ32' a la variable \$DSCMD.

Les variables peuvent etre additionnees l'une a l'autre, ou plutot concatenees, simplement en juxtaposant les deux noms. Examinons le programme suivant :

```
PRGESSAI
$A = NOM
$B = FICH
$C = $A$B
```

La valeur de \$C serait mise a 'NOMFICH'. Variables et litteraux peuvent etre combines de la meme facon :

```
PRGESSAI
$A = FICH
$B = MAI
$C = $A/$B
```

Donnerait a \$C la valeur 'FICH/MAI'.

En nommant des variables, JCL n'est concerne que par les caracteres significatifs du nom : en ce qui concerne JCL, 'nom' et 'nomfich' sont identiques, car le premier est contenu dans le second. Pour identifier des variables differentes, il faut eviter que l'un d'eux soit inclus dans l'un des autres, par ex. :

FICH et FICH1

sont identiques, alors que :

FICH1 et FICH2

sont differents.

Variables speciales :

JCL connait deux variables dont la valeur est affectee par le systeme lui-meme. Il peut etre tres avantageux de les utiliser dans des procedures.

La premiere est \$ERR. Sa valeur est fournie par le systeme au retour d'un ordre de la librairie ou d'un autre programme. Si une erreur s'est produite, \$ERR contiendra le code de l'erreur en decimal, sinon "00". \$ERR est utilise dans des procedures JCL pour detecter et pieger des erreurs.

L' autre variable speciale est \$LEVEL. Des procedures JCL peuvent s' imbriquer (en appeler d' autres, qui en appellent d' autres, et ainsi de suite). La variable \$LEVEL indique quel est le niveau de procedure en cours d' execution. Le niveau le plus haut est 1. Si une procedure JCL est lancee, elle aura le niveau 1. Si elle en lance une autre, celle-ci sera de niveau 2, etc. Au fur et a mesure que des procedures se terminent, le niveau remonte. Il peut y avoir jusqu' a 8 procedures imbriquees.

Etiquettes JCL.

JCL permet l' usage d' etiquettes pour identifier des blocs de texte. Le format est :

-etiquette.

Un nom d' etiquette est toujours precede d' un tiret '-'. Il peut avoir de 1 a 8 caracteres, toute combinaison de lettres et de chiffres est acceptee.

Quand il affecte des etiquettes, JCL ne s' occupe que de la partie significative (voir plus haut).

Remarques JCL :

Le systeme JCL permet l' usage de remarques ou de commentaires dans un programme. Il ne sont pas executes, mais permettent d' eclairer le programme. Une remarque commence par un point. Tous les caracteres suivants seront ignores par JCL. Par ex. :

```
jcless  
.c' est un programme de demonstration.  
.Suppression de tous les fichiers d' un disque.  
/TYPE Pressez <ENTER> pour eliminer  
.tous les fichiers texte.
```

Les lignes commençant par un point ne décrivent que l' objectif du programme, ou en renseignent des parties .

III - Le jeu d' instructions de JCL.

Ce langage contient 17 ordres. Chacun de ceux-ci lance une action determinee , qui resulte dans un changement d' etat du programme, une execution, une E/S avec le monde exterieur, etc. Chacun de ces ordres est detaille plus loin.

/DOS

Format general : <DOS> expression

L' ordre /DOS est l' un des plus usites. Il sert a permettre a JCL de passer une ligne d' ordres a l' interpreteur d' ordres de DOSPLUS. Par exemple la commande :

/DOS DIR :1

ferait afficher par JCL le directory du disque 1. Des variables peuvent etre utilisees avec l' ordre /DOS, comme avec n' importe quel autre. Si nous supposons que la variable \$LECTEUR a la valeur '13', l' ordre JCL :

/DOS FREE \$LECTEUR

serait etudiee comme 'DIR :13', et afficherait le directory du lecteur 3.

L' ordre /DOS peut etre utilise pour executer aussi des programmes au niveau SED. Exemple :

/DOS BACKUP :0 :1,USE=Y,DATE="01/01/83"

Provoquerait la recopie du lecteur 0 au lecteur 1.

/EXIT

Format general : /EXIT

ordre de terminaison d' une procedure JCL. Le controle est rendu a la procedure du niveau immediatement superieur, ou au SED si l' on etait deja au niveau 1.

/READ

Format general : /READ \$var1 \$var2 \$var3 ...

Ordre de demander une entree au clavier ou a la queue de clavier. Les donnees sont placees dans des variables JCL. Quand l' ordre /READ est execute, JCL trouve une ligne dans la queue du clavier, ou attend qu' une ligne soit tapee au clavier. Quand il examine une ligne, /READ considere l' espace comme un delimiteur; c' est a dire qu' il affecte des valeurs a des variables, il le considere comme un caractere de fin de message. Par exemple :

/READ \$A1 \$A2 \$A3

si la ligne a lire est :

DONNEES :0 :3

Les variables prendront les valeurs :

```
$A1= DONNEES
$A2= :0
$A3= :3
```

De nombreux autres caracteres sont aussi consideres comme des terminateurs: '/', ':', ',', '.', Si dans l' exemple precedent la ligne de donnees avait ete :

```
DONNEES,:0,:3
```

Le resultat aurait ete le meme.

Il faut remarquer que chaque fois qu' un ordre /READ est execute, une ligne entiere de donnees est consommee (jusqu' au retour/chariot). Si donc on execute l' ordre :

```
/READ $VAR1 $VAR2
```

sur une ligne comme :

```
NOMFICH1 NOMFICH2 NOMFICH3 :1
```

Les deux variables prendront les valeurs NOMFICH1 ET NOMFICH2, et les reste des donnees sera perdu. Le /READ suivant commencera sur la ligne suivante.

Si par contre un ordre /READ cherche a lire plus de donnees que la ligne n' en contient, les premieres variables seront remplies, mais les dernieres resteront vides . Par exemple si la ligne :

```
/READ $A1 $A2
```

cherche a lire la ligne de donnees :

```
POINTU
```

\$A1 prendra la valeur POINTU, mais \$A2 restera vide. Il n' y aura pas de tentative de lire la ligne suivante.

```
=====
/TYPE
```

Format general : /TYPE expression

Cet ordre ecrit sur l' ecran. Par exemple la ligne :

TYPE Pressez <ENTER> quand vous etes pret

Affichera ce texte sur l' ecran du TRS-80. Il peut bien sur etre utilise avec des variables. Par exemple examinons la procedure :

```
/TYPE Entrez le nom du fichier:
.Lecture du nom au clavier
/READ $NOMFICH
.Utilise l' utilitaire CRUNCH sur ce fichier
/DOS CRUNCH $NOMFICH/BAS $NOMFICH/CRN
.Affichage d' un message pour l' operateur
/TYPE Fichier : $NOMFICH traite.
/EXIT
```

C' est une procedure simple qui va afficher le message 'Entrez le nom du fichier', invitant l' operateur a fournir un nom, qui sera place dans la variable \$NOMFICH. Quand il l' aura fait, le fichier designe (avec l' extension /BAS) sera traite par l' utilitaire CRUNCH, et le fichier resultant sera baptise du meme nom, mais avec l' extension /CRN.

L' ordre /TYPE peut etre utilise sans variable ni texte, pour fournir une ligne blanche sur l' ecran.

```
/PRINT
```

Format general : -/PRINT <expression>

Fonctionne comme /TYPE, mais ecrit sur l' imprimante au lieu de l' ecran.

```
/IF
```

Format general : /IF <expr1> <relation> <exp2> /ORDRE JCL

permet de faire des comparaisons logiques entre variables JCL, et de prendre des decisions en fonction du resultat. L' ordre /IF reconnait six operateurs de relation :EQ (egalite), NE (inegalite), GE (superieur), LE (inferieur ou egal), et LT (inferieur). Ces operateurs peuvent etre utilises pour comparer deux expressions JCL qui peuvent etre composees de variables JCL et/ou de constantes. Examinons la procedure :

```
LISTER
```

```
.Ce programme JCL produit un listage de
.fichiers ASCII sur l' imprimante ou l' ecran
/TYPE Listage de fichiers
/TYPE
/TYPE Entrez le nom du fichier
/READ $NOMFICH
/TYPE Listage sur ecran ou imprimante (E/I)
/READ $SORTIE
/IF .$SORTIE EQ .V /DOS LIST $NOMFICH
/IF .$SORTIE EQ .P /DOS LIST $NOMFICH TO @PR
/EXIT
```

Dans ce programme, la variable \$SORTIE sert a determiner si le listage d' un fichier va etre fait sur l' ecran ou l' imprimante. Quand une condition logique est vraie, l' instruction JCL qui est sur la meme ligne est executee, dans le cas contraire, JCL passe a la ligne suivante. Le simple programme suivant illustre le fonctionnement de l' ordre /IF:

```
IFPROG
```

```
/TYPE DIRECTORY OU CATALOGUE (D/C)?
/READ $DIRCAT
/IF .$DIRCAT EQ .D /DOS DIR :0
/IF .$DIRCAT EQ .C /DOS CAT :0
/EXIT
```

Dans cet exemple, si la condition \$DIRCAT = D est vraie, JCL execute l' instruction /DOS DIR :0. Si \$DIRCAT n' est pas egal a D, et est egal a C, JCL execute l' instruction /DOS CAT :0. Si \$DIRCAT n' est egal ni a D, ni a C, rien n' est fait.

Remarquez l'utilisation du point dans les instructions /IF ci-dessus. C' est une protection contre la possibilite qu' une variable soit vide dans l' ordre /IF. Cette precaution est a prendre chaque fois que cela risque de se produire. En effet, JCL remplace la variable par sa valeur avant d' executer l' ordre. Si dans notre exemple \$DIRCAT etait vide, cela donnerait :

```
/IF EQ D /EXIT
```

qui n' aurait aucun sens. En plaçant un caractere supplementaire des deux cotes de l' operateur de relation, on evite ce genre d' ennui.

/GOTO

Format general /GOTO -label

Sert a modifier le deroulement normal du programme. Avec cet ordre, l' execution d' un programme peut etre deroutee vers toute etiquette dans une procedure JCL. /GOTO est souvent utilise en liaison avec /IF. Par exemple :

```
GOTOPROG
-BOUCLE
/TYPE Entrez le Numero de lecteur :
/READ $LECTEUR
/IF , $LECTEUR EQ , /GOTO -FIN
-BOUCLE1
/DOS CAT : $LECTEUR
/IF $ERR$ NE 00 /GOTO -ERREUR
/GOTO -BOUCLE
-ERREUR
/TYPE
/TYPE L' erreur codee $ERR s' est produite.
/TYPE Abandon ou Nouvel essai (A/N) :
/READ $REPONS
/IF , $REPONS EQ , R /GOTO -BOUCLE1
/TYPE Operation abandonnee
/EXIT
-FIN
/TYPE Procedure terminee
/EXIT
```

Dans ce programme, /GOTO est utilise en liaison avec /IF, pour executer une serie complexe de commandes suivant les conditions rencontrees.

/QUEUE (file d' attente)

Format general QUEUE <expression>

Fournit un moyen de placer des donnees dans la file d' attente du clavier. Par exemple pour placer les donnees 'BASIC -F:3' dans la queue, ecrivez l' ordre :

```
/QUEUE BASIC -F:3
```

Des variables peuvent etre utilisees avec /QUEUE, comme dans la routine ci-dessous:


```

/TYPE Entrez le nom du fichier a lister
/READ $NOMFICH $EXT $LECTEUR
  .creation des delimitateurs necessaires
/IF , $EXT NE , $SLASH = /
/IF , $LECTEUR NE , $DEPOIN = :
  .chargement de la queue avec des ordres BASIC
/QUEUE LOAD"$NOMFICH$SLASH$EXT$DEPOIN$LECTEUR"
/QUEUE LPRINT"$NOMFICH - Liste du programme"
/QUEUE LPRINT
  .reglage de FORMS
/DOS FORMS,P=66,L=60,W=80
/DOS BASIC
/EXIT

```

Ce programme utilise l'ordre /QUEUE pour mettre des ordres BASIC dans la queue du clavier. Dans l'exemple, cela sert a ecrire des listes de programmes avec titre.

/QLOAD

Format general /QLOAD nomfich

Cet ordre permet de garnir la queue du clavier avec des donnees rangees dans un fichier disque. Par exemple un fichier contenant des noms de fichiers peut avoir ete ecrit avec l'ordre BUILD sous DOSPLUS (ou avec un editeur de textes). Une procedure JCL telle que la suivante peut charger les noms des fichiers dans la queue du clavier et remplir quelque fonction utile, telle que mettre l'indicateur INV avec l'ordre DOSPLUS ATTRIB :

```

INVIS $NOMFICH
-ACQUINOM
/IF , $NOMFICH NE , /GOTO -FICHOK
/TYPE Entrez le nom du fichier
/READ $NOMFICH
/GOTO -ACQUINOM
-FICHOK
/QLOAD $NOMFICH
/QUEUE ???
-RENDINV
/READ $FICH$EXT$LECT
/IF $FICH EQ ??? /GOTO FIN
$SLASH =
$DEPOIN =
/IF , $EXT NE , $SLASH = /
/IF , $LECT NE , $DEPOIN = :
/DOS ATTRIB $FICH$SLASH$EXT$DEPOIN$LECT,INV
/GOTO -RENDINV
-FIN
/TYPE Procedure terminee
/EXIT

```

L'ordre /QLOAD suppose une extension /TXT pour tous les fichiers sauf indication contraire.

/PURGE

Format general

/PURGE

L'ordre **/PURGE** a pour unique role de vider, ou purger la queue du clavier. Toute donnee placee dans cette queue avant **/PURGE** est detruite. L'ordre **/PURGE** est utilise quand une procedure JCL a place dans la queue des donnees qui, pour une raison ou une autre, doivent en etre enlevees (comme une fin de procedure prematuree et anormale).

/JUMP

Format general

/QUEUE /JUMP -etiquette

L'ordre **/JUMP** n'est pas un ordre au meme sens que **/GOTO**, **/TYPE** ou **/IF**. C'est plutot une instruction qui place un caractere special dans la queue du clavier. Quand ce caractere special est trouve par un autre programme, JCL interrompt l'execution du programme, et passe la main a une etiquette fixee par l'utilisateur dans la procedure JCL. En voici une illustration :

```
JUMPROG
/TYPE entree en BASIC . . .
/TYPE
/QUEUE /JUMP -CHARPROG
/DOS BASIC
/EXIT
.Interception de BASIC ici
-CHARPROG
/TYPE entrez le programme a editer :
/READ $NOMFICH
/QUEUE LOAD"$NOMFICH"
/QUEUE CMD"SR","PRINT","LPRINT"
/RESUME
```

Ce programme va charger l'interpreteur BASIC disque, Comme l'ordre **/JUMP** est charge dans la queue, la prochaine fois que BASIC essaiera de lire un caractere du pilote clavier, il va recevoir l'ordre **/JUMP**, faisant prendre le controle a JCL. Dans ce cas le controle est passe a l'etiquette **-CHARPROG**, qui remplace tous les ordres **PRINT** par des ordres **LPRINT**.

/RESUME

Format general

/RESUME

Cet ordre est utilise pour rendre la main a un programme interrompu par l'ordre /JUMP. Apres que tous le traitement JCL voulu a ete execute, l'execution de /RESUME va faire rendre la main au programme interrompu, au point ou JCL est intervenu. L'exemple ci-dessus en est une illustration.

/CANCEL

Format general

/CANCEL

Le but de /CANCEL est de permettre a une procedure JCL, entree par /JUMP de ne pas retourner au programme intercepte. Quand /CANCEL est execute, JCL retourne a sa procedure qui etait en cours, a la ligne suivant l'ordre ou a ete lu l'ordre /JUMP dans la queue. Par exemple :

```
PATCH
/TYPE Entrez le nom du programme a modifier
/READ $NF $LT
/IF . $LT NE . $DLIM = :
, on met les nouvelles donnees dans la queue
/DOS PATCH $NF/CMD$DLIM$LT
/TYPE Procedure terminee
/EXIT
-ENCORE
/TYPE Corrections obligatoires installees
/TYPE En avez vous d' autres (O/N)?
/READ $QUINON
/IF . $QUINON EQ .N /CANCEL
/IF . $QUINON EQ .O /GOTO -ENCORE
/TYPE le controle est passe a PATCH
/TYPE Pressez <BREAK> quand toutes les modifs
/TYPE sont installees
/TYPE
/RESUME
```

/RUN

Format general

/RUN nomfich <exp1> <exp2> <exp3>

L'ordre /RUN est utilise dans une procedure JCL pour executer un autre programme JCL. Quand une autre procedure JCL est executee avec l'ordre /RUN, l'etat de la procedure JCL en cours est sauve, y compris toutes les variables, et la variable speciale \$LEVEL est incrementee de 1. Comme chaque procedure JCL a un jeu de variables totalement independant (meme si elles ont le meme nom), le seul moyen qu'elles ont de communiquer entre elles est par la queue du clavier. Des parametres facultatifs peuvent etre passes a un programme JCL en les incorporant dans l'ordre /RUN. Ces variables sont lues dans la ligne d'identification de la procedure, comme nous l'avons deja vu.

Le programme suivant, qui ouvre cinq fichiers sur un lecteur choisi par l'utilisateur illustre l'utilisation de l'ordre /RUN :

```
RECURSV $DR
/IF $DR. NE . /GOTO -OUVRE
/TYPE %% il manque le NO de lecteur %%
/EXIT
-OUVRE
/TYPE ouverture du fichier ESSAI$LEVEL/DAT:$DR
/DOS CREATE TEST$LEVEL/DAT:$DR
/IF $ERR EQ 00 /GOTO -PADERR
/TYPE %% L' erreur $ERR est signalee %%
/TYPE %% Procedure abandonnee %%
/EXIT
-PADERR
/IF $LEVEL NE 5 /RUN RECURSV $DR
/EXIT
```

/DEBUG

Format general /DEBUG

Lance le moniteur DEBUG de DOSPLUS, qui prend le controle de la machine.

/VOFF

Format general /VOFF

Supprime l' habituel affichage des donnees lues dans la queue. Si /VOFF est place dans la queue avant les donnees a lire, leur affichage sera supprime.

/OPTION

Format general

/OPTION <param>/choix <param>/choix

Cet ordre en contient en realite 6. Il permet a un programme JCL de <1> activer ou desactiver la visualisation, <2> activer ou desactiver l'ordre JCL TRACE, <3> ouvrir ou fermer la queue, <4> autoriser ou non les demandes de caracteres isolés de la queue, <5> orienter toutes les donnees envoyees a l'ecran vers la queue ou <6> reconnaitre ou ignorer les caracteres speciaux.

Chaque sous commande de /OPTION peut etre activee ou desactivee en precisant un choix apres son nom. Par exemple :

```
/OPTION QUEUE/Y
/OPTION JCL/N
/OPTION DSP/N
```

Le choix, comme ci-dessus consiste en un seul caractere, 'Y' pour oui, 'N' pour non, separe de la sous-commande par un '/'. Plusieurs sous-commandes peuvent etre precisees avec un seul ordre /OPTION:

```
/OPTION DSP/N QUEUE/N DO/Y
```

DSP
—

Controle l'affichage des donnees.

JCL
—

Oblige JCL a afficher la trace de son activite au cours de l'execution de ses procedures. Il affiche chaque instruction avant de l'executer. C'est un moyen bien commode pour mettre au point les programmes JCL.

QUEUE
—

Active ou desactive la queue du clavier. Si elle est desactivee, toute demande au clavier doit etre satisfaite par le clavier lui-meme.

QBYTE
—

De nombreux programmes demandent a l'operateur de presser une seule touche pour accomplir certaines fonctions. Par exemple le programme BASIC ci-dessous :

```
10 PRINT" PRESSER (A) POUR ABANDONNER, (C) POUR CONTINUER"
20 A$=INKEY$:IFA$=""THEN20
30 IF A$="A"GOTO1000
40 IF A$="C"GOTO2000
50 GOTO 20
```

Il attend que l'operateur presse "A" ou "C" pour faire quelque chose. La sous-commande QBYTE autorise ou non la queue a repondre a de telles demandes de caractere isole. Les demandes pour des lignes completes de donnees ne sont pas affectees. L'etat normal de la queue est 'ON'

(accepte).

/DQ

Copie tout ce qui est envoye a l'ecran dans la queue. Par exemple examinez le programme suivant :

QCAT

/OPTION DQ/Y

/DOS CAT :1

/OPTION DQ/N

/TYPE Le catalogue est range dans la queue

Ce programme va placer une copie du catalogue du lecteur 1 dans la queue du clavier. Les donnees y seront placees meme si l'option DSP/N a ete executee.

CHR

L'ordre CHR sert a determiner comment JCL traite les "caracteres speciaux" lus dans la queue. Les caracteres speciaux sont tous les caracteres qui ne sont pas des lettres ou des chiffres. Si l'ordre /OPTION CHR/N est execute, les caracteres speciaux trouves dans la queue sont traites comme des delimitateurs, c'est a dire qu'ils terminent toute donnee lue, et ils sont sautes. Normalement, les delimitateurs peuvent etre lus dans une variable JCL, a l'exception de ceux que nous avons mentionnes plus haut (/!,.).

BASIC Disque
Table des matieres

Section	Page
Introduction	4/3
Lancement de BASIC (allocation de fichiers)	4/3
Note sur les E/S cassettes Modele I	4/5
Instructions et fonctions communes	
&H (conversion hexadecimale)	4/6
DEF FN	4/7
DEFUSR	4/7
INSTR	4/8
LINE INPUT	4/8
MID\$=	4/9
=MID\$	4/9
USR	4/10
Fonctions liees aux disques	4/12
Manipulation des fichiers	4/13
KILL	4/13
LOAD	4/13
MERGE	4/15
RUN	4/15
SAVE	4/16
Acces aux fichiers	
Initialisation des fichiers	4/17
OPEN	4/18
CLOSE	4/20
INPUT#	4/21
LINE INPUT#	4/23
PRINT#	4/24
FIELD	4/25
GET	4/26
PUT	4/27
LSET	4/27
RSET	4/27
MKI\$	4/27
MKS\$	4/27
MKD\$	4/27
CVI	4/27
CVS	4/27
CVD	4/27
EOF	4/28
LOF	4/28
LOC	4/29
Codes erreur du BASIC	4/30
Fonctions CMD	4/31
BASIC Disque etendu	4/32
CMD	4/33
DI	4/33
DU	4/33
Bref	4/34
RENUM	4/34
TAB	4/36
TRON	4/36
REF	4/36

CMD"M" (lecture dynamique des variables simples)	4/37
SR (recherche et remplacement globaux)	4/37
CMD"O" (routine de tri de tableaux)	4/38
INPUT@ (saisie écran contrôlée)	4/39
Adressage par étiquettes (branchement indirect)	4/40
Détail des indicateurs d'erreur	4/40

Introduction

Le BASIC Disque est simplement un jeu de complements du BASIC Niveau II residant en ROM, qui reside sur disque. Il contient des instructions qui permettent des E/S sur des fichiers disques pour enregistrer des donnees, et vous permettra de lancer des programmes BASIC enregistres sur disque. De plus BASIC Disque incorpore de nombreuses instructions que ne possede pas la ROM. Il est livre sous deux formes : BASIC et TBASIC.

Ce manuel documente les particularites de ces deux BASIC disque. En ce qui concerne les fonctions du BASIC Niveau II, referez vous au manuel de votre machine. Notre manuel est divise en deux sections principales. La premiere traite des fonctions standard du BASIC Disque qui apparaissent dans les deux versions. La seconde traite des fonctions etendues qui n'existent que dans le BASIC disque etendu. La premiere est de plus subdivisee en groupes d'ordres : les ordres generaux, les ordres de manipulation des fichiers programmes et les ordres de manipulation des fichiers de donnees.

Le disque contient plusieurs fichiers necessaires a BASIC et aux fonctions qui lui sont associees. La liste ci-dessous detaille leurs noms et leur usage. Celles dont on n'a pas besoin peuvent etre eliminees.

<u>Nom du fichier</u>	<u>Fonction</u>
BASIC/CMD	Le programme du BASIC etendu
TBASIC/CMD	Le id id reduit
ERROR/DVL	Programme annexe utilise par BASIC etendu pour emettre ses messages d'erreur detaillies. Si ce programme est supprime du disque, les deux BASIC emettront les memes messages d'erreur abreges.
RENUM/CMD	Renumerotage des lignes.
REF/CMD	programme de references croisees.
SR/CMD	recherche et remplacement globaux.
O/CMD	utilitaire de tri de tableaux.

Le fonctionnement des quatre derniers sera decrit dans la section dediee aux extensions.

Lancement du BASIC

Pour executer le BASIC Disque, entrez sous SED, TBASIC ou BASIC suivant la version que vous choisissez. Les differences sont :

BASIC	fonctions etendues et commande du SED. Il utilise une structure de programmes annexes (overlays) pour les ameliorations et les messages d'erreur.
TBASIC	Econome en memoire, vous en laisse le plus possible. Ses ordres sont compatibles avec la plupart des programmes. Une fois charge, il reside completement en memoire et n'utilise pas d'annexes.

Il faut indiquer, en lançant BASIC, le nombre de fichiers (superieur a 1) que vous envisagez d'utiliser simultanement et la quantite de memoire eventuelle que vous voulez proteger de BASIC. Il existe une methode de modifier le nombre de fichiers par defaut, mais nous verrons cela plus tard. La livraison DOSPLUS des deux versions a un defaut de 1 tampon fichier reserve.

Si vous prevoyez de n'ouvrir aucun fichier de donnees, vous pouvez indiquer 0 fichier au lancement de BASIC ou TBASIC.

La syntaxe generale est :

BASIC nomprog-F:nn-M:nnnn

nomprog, facultatif, est le nom du programme BASIC que vous voulez eventuellement lancer en meme temps que BASIC.

-F:nn est le nombre de tampons fichiers que vous pourrez utiliser en meme temps. Vous ne pourrez pas utiliser un numero de tampon plus eleve pour un OPEN.

-M:nnnn est l'adresse memoire la plus elevee que vous accordez a BASIC, pour proteger des routines en langage machine que vous pourriez logger en haut de memoire. Elle ne peut exceder l'adresse haute (HIMEM).

Voici quelques exemples, pour lesquels nous utiliserons BASIC, mais TBASIC pourrait etre utilise a la place.

BASIC

chargerait BASIC avec un fichier, sans memoire protegee.

BASIC *

recharge BASIC (au cas ou vous seriez retourne sous DOSPLUS et n'avez pas modifie le contenu de la memoire utilisateur, avec votre programme intact. Les tampons fichiers et la memoire protegee seront les memes qu'a la sortie de BASIC. A l'arrivee, BASIC va commencer a lister le programme pour indiquer que le retour etait reussi. Ce listage peut etre arrete par <break>.

BASIC nomprog

lance BASIC et lance le programme 'nomprog'.

BASIC -f:3

lance BASIC avec trois tampons fichiers

BASIC -m:60000

lance BASIC en reservant la memoire au-dela de l'adresse 60000.

Note : Une fois BASIC lance, il n'est plus possible de modifier la protection memoire ou le nombre de tampons fichiers sans relancer BASIC a partir du SED.

Au lancement, BASIC affiche son en-tete :

DOSPLUS - Extended Z80 Disk BASIC - Ver 1.7
(c) 1983, Micro-Systems Software Inc.

READY
>

ou

DOSPLUS - Tiny Disk BASIC - Ver 1.7
(c) 1983, Micro-Systems Software Inc.

READY

>

Suivant la version demandee.

Une fois sous BASIC, vous pouvez retourner sous SED en entrant "CMD", par souci de compatibilite, CMD"S" est accepte. Si vous avez quitte BASIC par erreur, vous pouvez y retourner immediatement en entrant BASIC *, sans dommage pour votre programme ni vos donnees en memoire. Ce ne sera pas possible si vous avez execute une operation quelconque qui modifie le contenu de la memoire, aussi il est prudent de sauver votre programme sur disque avant de quitter BASIC.

Note aux cochers de Modele I :

Pour que BASIC ou TBASIC fonctionnent correctement avec les cassettes, il faut interdire les interruptions avant toute operation sur cassette, et les autoriser des qu'ele est terminee. Ceci vaut aussi pour la copie et la lecture des programmes.

Vous realisez cette operation par l'ordre 'CMD"T"', qui interdit les interruptions, et 'CMD"R"' qui les autorise. Ces ordres peuvent etre donnees en direct, ou dans un programme. Ils sont inutiles sur le Modele III.

Fonction communes

&H (constante hexadecimale) -

Permet d'utiliser directement les valeurs hexadecimales, ce qui est souvent commode quand il s'agit d'adresses memoire. &H est mis en prefixe de la valeur qui suit immediatement.

La forme generale de l'ordre est :

&Hdddd

dddd est une valeur hexa de 1 a 4 chiffres.

La constante represente toujours un entier signe, un nombre superieur a &H7FFF sera donc interprete comme une quantite negative. Par exemple :

Nbre hexa	Valeur decim.
&H1	1
&H2	2
&H5200	20992
&H7FFF	32767
&H8000	-32768
&H8001	-32767
&H8002	-32766
&HFFFE	-2
&HFFFF	-1

Il n'est pas possible d'entrer des valeurs hexa en reponse a un "INPUT", ni de les inclure dans des lignes de DATA.

EXEMPLE

PRINT &H5200 affichera 20992 sur l'ecran.

POKE &H3C00,65 ecrira ula valeur 65 decimale au debut de la ram video.

A=PEEK(&H37E8) ecrira l'etat de l'imprimante dans la variable A.

DEF FN (definition de fonction) -

Cette instruction permet de definir des fonctions implicites. Quand c' est fait, il suffit de les appeler par leur nom pour qu' elles soient executees.

La forme generale de l' ordre est :

DEF FN var1(var2)=expres.

"var1" est le nom de la fonction (nom de variable).

"var2" est utilise pour decrire le traitement.

"expres." est la formule qui manipule "var2".

Quand une fonction est definie, on peut l' appeler par "var1" prefixe de "FN".

Le type de variable indique par "var1" definit le type de variable qui sera fourni comme resultat, "var2" definit le type de variable a fournir a la fonction, numerique ou chaine de caracteres. Ces noms de variables sont factices et peuvent a l' appel etre remplaces par d' autres noms de variables du meme type, exemple :

DEF FN TV*(T)=TC!*FAZ*100

retournera TOUJOURS une variable double precision, meme si les variables utilisees sont des variables entieres. Quand la fonction est definie, il suffit pour l' utiliser d' ecrire :

V*=FN TV*(TC!)

V* contiendra le resultat de (TC!*FAZ*100). Le nom de la variable utilise n' a pas d'importance, tant qu' il n' y a pas de remplacement de chaines et de nombres. PARTC! etait remplace par RZ, avec la meme valeur numerique, le resultat serait strictement le meme>.

Un argument au moins doit figurer dans la definition de la fonction, meme s' il est factice. Mais on peut en utiliser plusieurs,et melanger les types, tant que l' operation demandee est reguliere.

DEFUSR: definition de l' adresse d' entree dans une routine en langage machine.

DEFUSRn=adresse

"n" est le numero de la routine, de 0 a 9 (0 par defaut).

"adresse" est l' adresse de lancement de la routine, elle peut etre exprimee en decimal, ou en hexadecimal (&H,...).

EXEMPLE : DEFUSR1=&HFF00

definira le point d' entree de la routine NO 1 comme FF00 hexa (decim 65280). quand le programme appelle USR1, le controle sera transmis a la routine commençant a cette adresse.

Pour charger une routine en RAM, on peut soit utiliser des lignes de DATA et les ecrire en memoire par des POKE, ou utiliser un editeur/assembleur pour faire un fichier objet, qui sera ensuite charge en RAM par une instruction 'CMD"LOAD nomfich"'.

Dans tous les cas il faut s' assurer que la zone memoire ou la routine reside a ete protegee au lancement de BASIC, sinon BASIC la detruira.

INSTR: fonction de recherche de chaines de caracteres.

La forme generale de l'ordre est :

INSTR(n,str1,str2)

"n" est le numero du premier caractere ou la recherche doit commencer.

"str1" est le nom de la chaine dans laquelle est faite la recherche.

"str2" est le nom de la chaine a rechercher.

INSTR recherche une chaine objet dans une chaine cible a partir d'une position determinee. Vous pouvez negliger la position de depart, le default sera 1. Si la chaine cherchee se trouve plusieurs fois dans la chaine exploree, seule la premiere sera affichee. La chaine objet doit etre entierement contenue dans la chaine cible, sinon le resultat sera nul.

EXEMPLE: (A\$="bonjour bonjour",B\$="nj")

Chaine cherchee	resultat
INSTR(A\$,"ON")	0
INSTR(A\$,"on")	2
INSTR(7,A\$,"our")	13
INSTR(3,"123123","12")	4

Notez que dans le premier exemple, la reponse est 0, car INSTR distingue majuscules et minuscules.

=====

LINEINPUT: Permet d'entrer une ligne de texte complete par le clavier. Il fonctionne comme INPUT, mais seul <enter> sert de delimitateur, il accepte donc tous les caracteres

La forme generale de l'ordre est :

LINEINPUT "invite";var\$

"invite" est un texte a afficher (facultatif).

"var\$" est le nom de la variable chaine dans laquelle on veut mettre la reponse.

Les differences avec INPUT sont :

- * quand l'ordinateur attend une entree, il n'y a pas de point d'interrogation affiche.
- * chaque instruction LINEINPUT ne peut donner de valeur qu' a une variable.
- * les virgules et guillemets sont acceptes comme parties de la chaine.
- * les blancs en tete de chaine ne sont plus ignores, ils font partie de la variable var\$.
- * le seul moyen d'arreter une entree est <enter>.

LINEINPUT est utile quand on veut entrer des chaines de caracteres sans avoir a s'inquieter de l'entree accidentelle de terminateurs. Il est egalement utile aussi si l'on veut pouvoir entrer du texte contenant des virgules, guillemets ou des blancs en tete

EXEMPLE: LINEINPUT "Nom prenom ?";a\$

affichera l'invite "Nom prenom ?" et attendra la reponse, qui pourra comporter tous les signes de ponctuation.

MID\$= (remplacement d' une partie de chaine)

Cette instruction permet de remplacer une partie quelconque d' une chaine par une chaine designee, donnant une puissante capacite d' edition

La forme generale de l' ordre est :

```
MID$(var$,pos,long)=rep$
var$ est le nom de la chaine a editer
pos est le NO du premier caractere a remplacer
long est le nombre de caracteres a remplacer
rep$ est la chaine de remplacement
```

Note : la longueur de "var\$" n' est jamais changee. Si rep\$ est plus long que var\$, les caracteres supplementaires seront ignores. Cependant, si vous indiquez un nombre de caracteres a remplacer plus grand que la longueur de rep\$, celui-ci remplacera la longueur que vous aurez indiquee (il ne remplacera pas plus de caracteres qu' il n' y en a dans rep\$).

Le parametre 'long' peut etre omis, Dans ce cas il est remplace par la longueur de rep\$. Il est a remarquer que c' est la seule fonction BASIC qui soit a gauche du signe '='.

EXEMPLES :

```
(A$="abcdefg")
```

Expression	A\$ resultant
MID\$(A\$,3,4)="12345"	ab1234g
MID\$(A\$,1,2)=""	abcdefg
MID\$(A\$,5)="12345"	abcd123
MID\$(A\$,1,5)="01"	abcd01g
MID\$(A\$,1,3)="xxx"	xxxdefg

=MID\$ (reproduction d' une portion de chaine)

Cette fonction permet de dupliquer une portion d' une chaine sans l' alterer.

La forme generale de l' ordre est :

```
nouv$=MID$(orig$,pos,long)
```

nouv\$	est la chaine que nous formons par cette operation.
orig\$	est la chaine contenant la chaine a reproduire
'pos'	est le numero du premier caractere a reproduire.
'long'	est le nombre de caracteres a reproduire.

Ceci montre l' autre fonction de MID\$, qui peut aussi apparaitre a droite du signe '='. Par exemple vous pourriez ecrire :

```
B$=MID$(A$,2,3)
```

Qui definirait B\$ comme trois caracteres de A\$, pris a partir du deuxieme. De cette maniere, MID\$ devient un moyen efficace pour interroger chaque element d' une chaine, determiner s' il doit etre modifie, et le mettre a jour si necessaire. =MID\$ est un outil extremement puissant et utile.

USRn (appel d' un sous-programme en langage-machine),

Cette instruction permet de passer le controle de BASIC a un sous-programme place en RAM.

La forme generale de l' ordre est :

USRn(exp)

"n" est le numero du programme usr (de 0 a 9, 0 par default).

"exp" est un nombre entier (de -32768 a 32767) transmis comme argument au sous-programme.

Quand un appel USR est rencontre dans le programme, le controle est passe a l' adresse precisee dans l' instruction DEFUSRn correspondante. Cette adresse indique le point d' entree dans le programme binaire.

Note : Si vous essayez d' appeler une sous-routine USR avant de l' avoir definie, l' erreur "illegal function call" (appel illegal de fonction) sera signalee.

On peut echanger un argument directement entre BASIC et ce programme, d' autres peuvent etre passes par des POKE et PEEK.

EXEMPLE

A=USR1(X)

passera la valeur de X a la routine USR NO 1, qui devra avoir ete definie avant, et retournera une valeur dans A si l' on fait le saut convenable, sinon a prendra la valeur de X.

Pour passer des arguments :

ecrire par POKE leurs valeurs dans la zone protegee de la routine, a un endroit ou elle viendra les chercher. Elle pourra ranger un resultat dans la meme zone, ou BASIC pourra les lire par PEEK(). C' est le seul moyen de passer plus d' un argument a une routine USR.

Notes techniques

A l' entree dans voter routine USR, le BASIC Disque de DOSPLUS aura prepare pour vous les informations suivantes:

A= type de variable
HL=> accumulateur virgule flottante
DE=> adresse de la chaine

Le registre A contiendra une valeur qui indique le type de variable passe comme argument :

Valeur	Type
2	Entier
3	Chaine de caracteres
4	Simple precision
8	Double precision

La paire de registres HL pointe vers l' accumulateur a virgule flottante, qui, dans le cas de variables numeriques, contiendra la valeur courante.

La paire DE pointe vers l' adresse de la chaine quand un variable type 3 est presente. Donc, si A=3, DE vous dit ou trouver la chaine.

A l' adresse pointee par DE, vous trouverez trois octets d' information : le premier indique la longueur de la chaine, les deux autres l' adresse reelle du premier octet de la chaine

dans le format (LSB/MSB) (poids faible/poids fort) (voir VARPTR)

Sous-programmes de la ROM.

Deux sous-programmes de la ROM peuvent être utiles pour échanger des variables entre la routine USR et votre programme. Ce sont:

CALL 0A7FH	qui met l'argument de USR dans le registre HL. Ce CALL devrait être la première instruction de la routine, si l'on veut utiliser des arguments fournis par BASIC.
JP 0A9AH	renvoie l'entier range dans HL à la variable de sortie de USR. Si on ne cherche pas de résultat, on peut remplacer ce JP par un RET.

Fonctions liees aux disques -

La section suivante couvrira les fonctions directement attachees aux entrees/sorties disques.

Il y a en realite deux parties : l' une est la manipulation des fichiers, l' autre est l' acces aux fichiers. La premiere traite du fichier comme un tout, la seconde traite de l' acces a un enregistrement particulier. Nous allons les couvrir l' une apres l' autre.

Commandes de manipulation de fichiers

KILL	elimine d' un disque un fichier de programme ou de donnees.
LOAD	charge un programme BASIC du disque.
MERGE	fusionne un programme BASIC en format ASCII avec un programme resident en memoire.
RUN	charge et lance un programme BASIC ecrit sur le disque.
SAVE	ecrit sur disque un programme resident en memoire.

Commandes et fonctions d' acces aux fichiers

Instructions

OPEN	ouvre un fichier pour y acceder
CLOSE	ferme l' acces au fichier
INPUT#	lit sur le disque en mode sequentiel.
LINEINPUT#	lit une ligne de donnees sur le disque en mode sequentiel.
PRINT#	ecrit sur le disque en mode sequentiel.
GET	lit sur disque en mode aleatoire.
PUT	ecrit ""
FIELD	nomme et dimensionne des portions d' un tampon d' acces aleatoire.
LSET	place une valeur dans un tampon d' acces defini, en justifiant a gauche (blancs a droite pour completer)
RSET	place une valeur dans un tampon d' acces, en ajoutant les blancs a gauche.

Fonctions

CVD	restaure un nombre double precision sous forme numerique, apres sa lecture dans un fichier.
CVS	memme operation pour un nombre simple precision
CVI	idem pour un entier.
EOF	signale la fin de fichier.
LOF	indique le nombre d' enregistrement logiques dans un fichier.
MKD#	convertit un nombre double precision en une chaine de 8 caracteres pour son enregistrement sur disque.
MKS#	idem pour nombre simple precision (4 caracteres)
MKI#	idem pour entier (2 caracteres).

KILL (elimine un programme ou un fichier de donnees d' un disque)

La forme generale de l' ordre est :

KILL "nomfich"

Fonctionne de la meme facon que l' instruction homonyme de DOSPLUS, sauf que vous ne pouvez pas l' utiliser pour eliminer un peripherique. Si le lecteur n' est pas precise, une recherche generale est entreprise, et le premier fichier rencontre est detruit.

EXEMPLE :

KILL "page/bas

detruira le programme page/bas sur le premier disque ou il sera trouve.

Note : Il ne faut pas appliquer KILL a un fichier ouvert. Bien que DOSPLUS n' y fasse pas d' objection et que le disque ne soit pas detruit, ce n' est pas de la bonne technique de programmation, et a la longue des ennuis peuvent se produire.

=====

LOAD -

Cette instruction charge un programme BASIC en memoire, qu' il soit ecrit en format standard ou ASCII.

La forme generale de l' ordre est :

LOAD "nomfich",option

nomfich est le nom du fichier que vous voulez charger.

Vos options sont :

- "R" indique de lancer le programme des son chargement. Les fichiers ouverts ne sont pas fermes.
- "V" conserve les variables et les resultats des FIELD.

Si un programme est charge sans option, BASIC remettra toutes les variables a 0 et fermera les fichiers. "R" a pour effet de laisser les fichiers ouverts et de lancer l' execution du programme. Mais detruit les variables et les champs de FIELD. "V" conserve les variables, mais efface le programme et ferme les fichiers.

Les deux options peuvent etre employees simultanement (c' est un chainage de programmes).

Si vous tentez de charger un fichier qui n' est pas un programme BASIC, vous aurez l' erreur 'Direct statement in file' (ligne trop longue).

Le chargement d' un programme enregistre en ASCII sera plus lent que celui d' un programme enregistre normalement, car chaque octet est interprete, comme s' il etait entre au clavier (a la vitesse pres). Et si une ligne depasse 240 caracteres, l' erreur 'direct statement in file' (ligne trop longue) sera affichee. Tenter de charger un fichier ASCII qui n' est pas un programme BASIC provoquera la meme erreur, ou l' erreur 'Attempt to use a non program file as a program' (essai d' utiliser comme programme un fichier qui n' en est pas un)

Les lignes des programmes BASIC s' allongent quand ils sont enregistres en ASCII, car les symboles BASIC sont remplaces par leurs expressions en toutes lettres (comme sur l' ecran).

EXEMPLE

LOAD"paye/bas:2",R,V

chainera le programme PAYE/BAS du lecteur N° 2 avec le programme resident en memoire, sans fermer les fichiers ni alterer les variables.

Note du traducteur : si on veut chainer des programmes, il faut prendre soin de faire en sorte que toutes les variables chaines soient "calculees", c'est a dire obtenues par une operation (+"" par exemple), sinon le BASIC les croit toujours au meme endroit dans le nouveau programme que dans l'ancien, ce qui est rarement le cas, et bien sur ne trouve pas ce qu'il cherche.

MERGE (fusion d' un programme du disque avec un programme resident)
permet la fusionnement d' un programme enregistre en ASCII avec un programme present en memoire.
La forme generale de l' ordre est :

MERGE"nomfich",option

'option' est un choix qui permet de ne pas interrompre l' execution apres le
chargement du nouveau programme.

MERGE a le meme effet que LOAD, sauf que le programme resident ne sera pas efface. Les li-
gnes du nouveau programme ayant le meme numero que des ligne de l' ancien remplaceront celles-ci
(comme si on les entrait au clavier).

C' est une methode pratique pour assembler des modules de programmes mis au point separement.
Avec l' option "R", les fichiers seront fermes et les variables effacees, la seule action de "R"
est de permettre au programme de reprendre.

Les programmes a appeler par MERGE doivent avoir ete sauves en ASCII.

RUN (charge et lance un programme du disque)

La forme generale de l' ordre est :

RUN"nomfich",option

Les deux premieres options sont les memes que pour LOAD.

'NO' permet de faire commencer le programme au numero de ligne indique.

Si le numero de ligne n' existe pas, le programme commencera au prochain, S' il n' y en a plus,
retour au BASIC.

SAVE (enregistre un programme sur disque)

La forme generale de l'ordre est :

SAVE"nomfich",A

provoque l'enregistrement du programme resident en memoire sous le nom "nomfich".

'A' est une option qui permet l'enregistrement sous format ASCII, sinon il est enregistre sous forme compressee (comme en memoire).

Avant d'executer l'ecriture, DOSPLUS recherche sur tous les lecteurs, ou sur le lecteur indique si ce nom de fichier existe, si il n'existe pas, il le creera sur le premier lecteur disponible (risque d'erreur 'Disk full' ou disque plein), ou sur le lecteur designe.

Sauver un programme en ASCII permet de faire certaines choses : MERGE, listage a partir du mode commande du SED, edition avec un editeur de textes ou un autre programme BASIC. Pour reconnaître un tel enregistrement, on peut lui mettre l'extension "/ASC".

Mais la methode la plus courante est de le sauver sous la forme compressee, c'est a dire sans l'extension A, ce qui permet un gain de place et de vitesse a l'ecriture comme a la lecture.

Apres execution, BASIC retourne en mode commande, toutefois SAVE peut etre execute a partir d'un programme BASIC et continuer, ce qui est utile pour des programmes qui s'automodifient.

Acces aux fichiers

Cette section est consacree aux instructions d' acces aux fichiers, c' est-a-dire, comme nous l' avons mentionne plus haut, celles qui permettent d' acceder a leur contenu, alors que le maneiement des fichiers st considere le fichier comme une entite.

Initialisation des fichiers

Avant de faire tourner un programme, le nombre de fichiers a reserver doit avoir ete determine et fixe. Il faut ensuite le fournir a l' entree sous BASIC. Ce sera le NO de tampon le plus eleve a pouvoir etre accepte.

Chaque tampon se voit affecter un numero de 1 a 15. Il sera repere par ce numero a l' ouverture des fichiers. Si l' on a entre BASIC en utilisant la syntaxe :

BASIC -F:3

on dispose de trois tampons numerotes 1,2,3.

Un tampon fichier est une sorte de reservoir a donnees. Toutes les donnees echangees avec le disque doivent transiter par l' un de ces fichiers. Le fait que l' on fait "PUT" sur un enregistrement ne signifie pas qu' il a ete ecrit physiquement sur le disque. Pour acceder a un fichier, il faut dire au BASIC quel tampon utiliser, quel type d' E/S vous voulez faire, et quelle est la longueur de l' enregistrement logique, ou bloc.

Toutes ces choses sont traitees a l' ouverture et a la fermeture des fichiers.

NDT : contrairement aux TRSDOS, (au moins jusqu' au 2.3), DOSPLUS accepte les longueurs d' enregistrements logiques variables : Un fichier peut etre ouvert avec une longueur de 1 a 256, cette longueur peut etre differente a chaque ouverture , mais attention a la recherche des donnees, dont la position sur le disque ne change pas.

OPEN -

Cette instruction permet d'affecter un numero de tampon a un fichier et de l'initialiser pour l'entree/sortie.

La forme generale de l'ordre est :

OPEN"mode",numtamp,"nomfich",lrl

"mode" est le type d'entree/sortie choisi. Seul le premier caractere compte, si c'est une lettre, elle devra etre mise entre guillemets

"numtamp" est le numero du tampon (1-15), que l'on veut affecter a ce fichier. Ce peut etre une variable, un numero ne peut designer qu'un tampon a la fois.

"nomfich" est le nom du fichier.

"lrl" est la longueur de l'enregistrement logique (LRL) du fichier. C'est un nombre de 1 a 256 (256 est represente par 0, ou par omission de ce parametre).

Ceci initialisera un tampon d'entree/sortie disque. Voici les details :

"mode" est le mode d'accès au fichier. Il y en quatre : "R" est le mode d'accès aleatoire.

"I"	""	sequentiel (écriture)
"Q"	""	" (lecture)
"E"	""	" (écriture extension)

Si un fichier ouvert pour l'accès aleatoire (R) n'existe pas, BASIC le cree. S'il existe, l'ouverture ne detruit aucune donnée. Le fichier pourrait etre ferme immediatement sans subir aucune modification.

Si un fichier ouvert en lecture sequentielle (I) n'existe pas, BASIC enverra un message d'erreur. On ne peut pas lire un fichier qui n'existe pas, ni lire des données apres la fin du fichier (il n'y en a pas). Si l'on essaie de lire des données dans un fichier ouvert en accès aleatoire, a un endroit ou il n'y a pas d'enregistrement, le tampon sera rempli de 0. Si l'on tente la meme chose avec un fichier sequentiel, une erreur 'Input past end' (lecture au dela de la fin) sera retournée.

L'ouverture d'un fichier sequentiel en écriture ('O' ou sortie) revient toujours a ouvrir un nouveau fichier, eventuellement sur les "ruines" de celui qui avait le meme nom avant (si un fichier du nom utilise existait avant l'ouverture, tout revient au meme que s'il etait d'abord detruit puis ouvert). Meme s'il est referme immediatement, tout son contenu est perdu. La mise a jour du DIRECTORY est fait comme s'il venait d'etre ouvert.

L'ouverture en extension sequentielle (E) a le meme effet que 'O', sauf que l'écriture se fera a la suite des enregistrements existants, au lieu de recommencer au debut. Il n'y aura donc pas de perte de données.

"lrl" est la longueur d'enregistrement logique du fichier auquel on veut acceder. Un enregistrement physique est defini comme un secteur de disque (256 octets). Il est affiche dans la colonne (#PHY) du DIRECTORY. Un enregistrement logique est defini comme la longueur des enregistrements definis par l'utilisateur. Elle est constante pendant toute l'ouverture du fichier, bien qu'elle puisse etre differente d'une ouverture a l'autre. Celle qui figure dans la colonne (LRL) du DIRECTORY est celle qui a ete utilisee lors de la creation du fichier, si c'etait en aleatoire. En accès sequentiel, LRL vaut toujours 256, le decoupage etant fait par le systeme.

On ne peut pas donner de LRL plus grand que 256 octets, mais elle peut etre aussi petite qu'un octet. L'aspect technique de la relation entre le nombre d'enregistrements logiques et le nombre d'enregistrements physiques est explique dans la section technique de ce manuel. Les

enregistrements logiques peuvent être et sont en fait souvent à cheval sur des secteurs et des pistes.

Si l'on utilise un LRL de 64, il tient 4 enregistrements logiques dans un enregistrement physique. Ceci veut dire qu'il faudra écrire 4 enregistrements, par l'instruction PUT, avant de remplir le tampon disque, que celui-ci soit automatiquement transféré sur le disque et libéré pour une autre opération d'entrée/sortie. En effet DOSPLUS n'écrit sur le disque que quand c'est nécessaire. À moins que le tampon ne soit plein ou que l'on ne ferme le fichier, DOSPLUS n'écrit qu'en mémoire, accroissant ainsi fortement la vitesse d'entrée/sortie.

EXEMPLES -

OPEN"O",1,"agenda/don"

cherchera à ouvrir en sortie séquentielle (écriture), le fichier "agenda/don" sur le premier lecteur non protégé en écriture s'il ne le trouve pas ailleurs. S'il le trouve, son contenu sera perdu. Le tampon N° 1 sera affecté pour l'accès.

OPEN"I",2,"coinfo/don:2"

essaiera d'ouvrir pour lecture séquentielle le fichier "coinfo/don", sur le lecteur N° 2. S'il ne le trouve pas, un message d'erreur sera affiché. Le tampon N° 2 est affecté à cet accès.

OPEN"R",1,"basedone/asc":3",24

ouvrira pour accès aléatoire (ou accès direct), le fichier "basedone/asc" sur le lecteur N° 3. Si ce fichier n'existe pas, il sera créé, dans le cas contraire, son contenu ne sera pas modifié avant la première instruction PUT. La longueur d'un enregistrement logique est de 24 octets et le tampon N° 1 est affecté à cette opération.

OPEN"E",1,"invdonn:3"

ouvrira le fichier "invdonn" sur le lecteur N° 3 en séquentiel pour écriture, mais au lieu d'en détruire le contenu, les écritures se feront à la suite, pour allonger le fichier; le tampon N° 1 sera utilisé.

Note : quand un fichier est ouvert, il est désigné par le N° du tampon qui lui a été affecté.

CLOSE -

Cette instruction permet de fermer un tampon précis, un groupe de tampons, ou tous les tampons d'un coup.

La forme générale de l'ordre est :

CLOSE n0 tamp, n0 tamp, n0 tamp...

"n0 tamp" est le NO de tampon que l'on veut fermer. On peut en designer jusqu'à 15. Si aucun NO n'est fourni, ils seront tous fermés.

Si l'on tente de fermer un fichier qui n'a pas été ouvert, l'instruction n'aura simplement aucun effet.

EXEMPLES -

CLOSE 1,2,3

fermera les tampons 1,2 et 3, qui pourront être réaffectés à d'autres fichiers par d'autres instructions OPEN.

Note : Il ne faut en aucun cas enlever un disque sur lequel il y a des fichiers ouverts. Ce serait désastreux pour le fonctionnement du système : le tampon en cours n'a pas été copié sur le disque et la fiche DIRECTORY n'est pas mise à jour avant la fermeture du fichier.

Les instructions suivantes provoqueront la fermeture des fichiers :

NEW, RUN, MERGE, EDIT, CLEAR, ajout ou suppression de lignes de programmes, et certaines instructions CMD".,."

INPUT# -

lecture sequentielle d' un fichier disque.
La forme generale de l' ordre est :

INPUT# numtamp,var,var,var...

"numtamp" est le numero du tampon du fichier

"var" est la variable ou l' on veut mettre l' information lue dans le fichier. Le nombre n' en est limite que par la longueur de la ligne BASIC.

Cette instruction fait lire des donnees d' un fichier qui a ete ouvert en lecture sequentielle. A l' ouverture du fichier, un pointeur est ajuste au debut du fichier, et avance a chaque lecture (comme pour les lectures de DATA). Si on veut recommencer la lecture, il faut fermer et rouvrir le fichier.

Le format des donnees sur le disque n' a pas d' importance, ni la longueur de chaque element. La lecture continue jusqu' au premier caractere terminateur, ou a la fin du fichier. Il faut toutefois eviter de faire lire des chaines alphanumeriques dans des variables numeriques.

Pour lire les donnees avec succes, il faut connaitre la forme qu' elles vont prendre. En lecture disque, BASIC ignore les blancs en tete, il suppose que le premier caractere non blanc est le debut de l' element a lire. Les terminateurs dependent de la nature des donnees (numeriques ou alphanumeriques).

Terminateurs numeriques:

Fin de fichier

255 eme caractere

,

retour chariot (<enter> ou chr\$(13))

retour chariot/saut de ligne (chr\$(10))

blanc/retour chariot

Terminateurs de chaines entre guillemets:

fin de fichier

255 eme caractere

" (guillemet)

"/blanc/, (guillemet suivi d' un espace et d' une virgule)

"/blanc/retour chariot

Terminateurs de chaines non entre guillemets:

fin de fichier

255 eme caractere

,

retour chariot

Si le premier caractere des donnees est un guillemet ("), BASIC considere que c' est une chaine de caracteres, toutes les donnees seront lues, y compris les retours chariot et le reste, jusqu' au guillemet suivant.

Dans le cas contraire, la chaine est traitee comme une chaine libre, les donnees sont lues jusqu' au premier terminateur. Les guillemets sont traitees comme des donnees.

A la lecture de donnees numeriques sur disque, BASIC les evalue comme la fonction VAL. Si le premier caractere lu n' est pas numerique, la variable vaudra 0.

Notes techniques :

Pour utiliser une virgule sur disque comme terminateur, ce doit être un littéral, c'est à dire qu'elle doit être mise entre guillemets ("").

Si un retour chariot est précédé d'un saut de ligne, ni l'un ni l'autre ne sera pris comme terminateur.

Un essai de lecture après que le pointeur interne a atteint la fin du fichier resultera en l'affichage du message d'erreur 'input past end' (lecture après la fin).

Quand BASIC rencontre un caractère de terminaison, il regarde plus loin, en essayant de lire le plus grand nombre possible de terminateurs, pour s'assurer que le pointeur indique bien le début de la prochaine variable.

Si un retour chariot est suivi d'un saut de ligne, le saut de ligne ne sera pas considéré comme faisant partie du terminateur, mais sera le premier caractère de la lecture suivante. Ceci diffère du basic disque du TRSDOS. DOSPLUS n'ajoute pas de saut de ligne après chaque retour chariot en entrée/sortie séquentielle.

Le BASIC disque lit toujours un fichier par blocs de 256 octets. Les lecteurs ne tournent pas nécessairement à chaque lecture. Le tampon peut contenir assez de données pour plusieurs variables.

Les disques ne tourneront pas à la fermeture d'un fichier ouvert en lecture séquentielle. La lecture seule n'entraîne aucune mise à jour du DIRECTORY.

EXEMPLE -

INPUT#1,A

essaiera de lire la variable numérique 'A' dans le fichier affecté au tampon N° 1.

INPUT#2,A\$,B#,CX,D\$

essaiera de lire d'abord une chaîne (A\$), puis un nombre double précision (B#), un entier (CX) et une autre chaîne (D\$) dans le fichier affecté au tampon N° 2.

LINEINPUT # -

lit, dans un fichier en sequentiel, une ligne de texte entiere.

La forme generale de l' ordre est :

LINEINPUT# numtamp,var

"numtamp" est le numero de tampon .

"var" est le nom de la variable ou sera stockee la chaine.

Cette fonction a les memes rapports avec INPUT# que LINEINPUT avec INPUT. Elle lit une ligne de donnees alphanumeriques qu' elle met dans la variable indiquee. Ce peut etre pratique pour lire du texte d' une variete mal connue, dont on ignore les terminateurs habituels.

LINEINPUT lira tout depuis le premier caractere du fichier jusqu' a :

- * un retour chariot non precede par un saut de ligne.
- * la fin du fichier.
- * le 255 eme caractere (inclusivement).

les autres caracteres terminateurs seront simplement inclus dans la chaine.

Si les donnees sont un programme BASIC enregistre en ASCII, Chaque LINEINPUT lira une ligne de programme. Si la variable utilisee fait partie d' un tableau, on peut lire tout le programme en memoire, comme des donnees.

EXEMPLE -

LINE INPUT# 1,D\$

lira dans le fichier correspondant au tampon NO 1, et mettra le fruit de sa lecture dans la variable D\$.

PRINT # -

Cette instruction écrit dans un fichier séquentiel.

La forme générale de l'ordre est :

```
PRINT# numtamp,USING format$,var,del,var...
```

"numtamp" est le numéro de tampon affecté au fichier ouvert pour écriture séquentielle.

"USING format\$", facultatif, est une chaîne de formatage utilisée pour l'écriture. Elle fonctionne de la même façon que le PRINTUSING de la ROM BASIC.

"var" est la variable que l'on veut écrire.

"del" est le délimiteur qui doit être placé entre chaque variable à écrire. Ce devrait être une virgule.

Cette fonction écrit les données séquentiellement dans un fichier. Le fonctionnement est exactement le même qu'en lecture : un pointeur interne est placé initialement au début, puis avance à chaque PRINT#, d'où la destruction de l'information antérieures, sauf si le fichier a été ouvert par "E".

PRINT# écrit sur le disque exactement comme PRINT sur l'écran. En fait c'est la même routine pour les deux, seul le périphérique est changé. Pour le voir, il suffit d'ouvrir l'écran pour une sortie séquentielle (c'est possible car l'écran est traité comme un périphérique quelconque : OPEN"O",1,"*DO"). Quand on y fait un PRINT#, on voit que le fonctionnement est exactement le même que pour un PRINT normal.

PRINT# n'effectue aucune compression de données avant d'écrire sur le disque, tout est écrit en format ASCII, y compris les nombres. Ce qui fait que la ponctuation est très importante, les points-virgules et les virgules y ont le même effet que sur l'écran (sauf entre guillemets).

Par exemple, si l'on fait un "PRINT#1, A\$,B\$,C\$", les variables A\$,B\$,C\$ seront espacées sur le disque comme elles le seraient sur l'écran, c'est à dire avec le même nombre de blancs. Si l'on fait un "PRINT#1,A\$;B\$;C\$", les variables vont se suivre sans espace de séparation.

Ce qui serait bon avec des valeurs numériques, car le blanc final est un délimiteur, mais pour des données alphanumériques, il faut mettre un délimiteur explicite sur le disque, on entrerait quelque chose comme :

```
PRINT A$;" ";B$
```

qui mettrait une virgule entre chaque variable, et tout irait bien, sauf dans le cas d'un LINEINPUT#, pour lequel il faut un retour chariot entre chaque variable. Il faudrait alors faire :

```
PRINT#1,A$;CHR$(13);B$
```

qui écrira un retour chariot entre chaque élément et satisfait le LINEINPUT#. Rappel : Le premier caractère écrit dans une donnée chaîne de caractères est le délimiteur, qui définira si la chaîne doit être considérée comme entre guillemets ou non à la lecture par INPUT#, ou LINEINPUT#. La fonction "CHR\$(" peut être utilisée pour inclure tout code de contrôle souhaité dans le texte.

A cause de la similitude entre PRINT et PRINT#, l'option USING est également utilisable, elle fonctionnera exactement de la même façon que sur l'écran.

FIELD

Cette instruction permet d'organiser le tampon d'un fichier a acces direct, defini par des variables, pour que l'E/S puisse commencer.

La forme generale de l'ordre est :

```
FIELD numtamp1,num2 AS var1$,num3 AS var2$ ....
```

"numtamp1" est le numero du tampon defini au moment d'OPEN.

"num2" definit la longueur du champ (field) de la variable "var1\$".

"num3" definit la longueur du champ de la variable "var2\$".

On peut decouper autant de champs que necessaire, tant que la somme des longueurs des champs ne depasse pas la longueur de l'enregistrement logique. Mais auparavant il faut que le fichier ait ete ouvert avec l'instruction 'OPEN"R". Ce qui lui affecte le tampon designe pendant le FIELD. Une fois le decoupage effectue, les donnees sont pretes a circuler entre le disque et la memoire par GET et PUT.

Ainsi qu'il a ete indique dans la presentation de OPEN, un tampon de fichier en acces direct peut avoir une longueur atteignant 256 octets, mais pour etre utilisable, il doit etre partage en variables. Toutes ces variables sont toujours des chaines de caracteres, les nombres y sont compactes en caracteres et reconvertis en nombres par des fonctions dont nous parlerons plus loin dans cette section.

L'instruction peut etre utilisee autant de fois que l'on veut. Une instruction FIELD ne modifie pas le contenu du tampon, elle change seulement la facon d'y acceder. Deux tampons et plus peuvent donner acces au meme fichier. Il est aussi possible d'utiliser une boucle FOR-NEXT pour faire le decoupage d'un tampon, particulierement dans le cas d'un tableau.

Les noms de variables affectes par une instruction FIELD n'apparaissent pas dans la zone habituelle des variables chaines, et n'occupent donc pas d'espace supplementaire. Les pointeurs de ces chaines sont diriges vers le tampon du fichier, et il n'est pas necessaire de tenir compte des variables utilisees dans une instruction FIELD pour calculer l'espace a reserver aux chaines.

Cependant, si, hors d'une instruction FIELD, un nom de variable est utilise a gauche d'un signe "=", elle sort du champ, et va dans la zone normale. Par exemple si apres :

```
FIELD 1, 23 AS A$
```

on fait : A\$=B\$, la variable champ est annulee et A\$ devient une variable ordinaire.

GET: Lit un enregistrement sur disque.

La forme generale de l' ordre est :
Lecture d' un enregistrement du disque
GET num1,num2

"num1" est le numero du tampon fichier.

"num2" est le numero de l' enregistrement logique recherche. Si il est omis, c' est le prochain enregistrement qui sera lu.

Cette instruction lira des donnees du disque dans un tampon de fichier, a condition que le fichier ait ete au préalable ouvert et organise.

Quand BASIC rencontre une instruction FIELD, il va lire l' enregistrement designe sur le disque, si il n' en est pas designe, il lit l' enregistrement "courant", qui est l' enregistrement dont le numero suit immediatement le dernier numero au quel on a accede. Ce qui peut etre pratique quand on sait que l' on va lire tous les enregistrements en suivant. On peut utiliser une boucle FOR-NEXT jusqu' a la fin du fichier.

EXEMPLE :

```
GET 1,1  
GET 6
```

Si l' on cherche a lire un enregistrement au dela de la fin du fichier, BASIC renvoie un tampon plein de zeros, mais aucune erreur n' est signalee. On peut eviter cela en verifiant la fin du fichier avec la fonction LOF.

PUT: Ecrit un enregistrement sur disque.

La forme generale de l' ordre est :

PUT num1,num2

Ces deux parametres sont utilises comme dans GET.
Cette instruction transfere des donnees du tampon fichier sur le disque, a condition que le fichier ait ete ouvert et organise (OPEN et FIELD), et que les donnees y aient ete mises par LSET ou RSET.

L' enregistrement courant est le meme que pour GET.

EXEMPLE:

PUT 1,2

Si l' enregistrement designe est au dela de la fin du fichier, celui-ci sera prolonge et cet enregistrement en deviendra le dernier. Si l' on fait un PUT1,500, la machine l' écrira, ainsi que les 499 qui le precedent. Cela peut etre utile pour reserver de l' espace pour un fichier, mais cela peut aussi consommer l' espace disque "a la vitesse de la lumiere". Comme pour GET, le numero 0 et les nombres negatifs sont refuses.

LSET et RSET: Placement des donnees dans les fichiers directs.

La forme generale de l'ordre est :

LSET/RSET var\$=exp\$

"var\$" est un nom de variable defini dans FIELD.

"exp\$" est l'expression d'evaluation.

Ces deux instructions placent des donnees alphanumeriques dans le tampon defini par FIELD avant de les ecrire sur disque. Les nombres doivent etre deja convertis en chaines.

LSET completera la chaine constituee avec des blancs a gauche, RSET la complete a droite. Si la chaine a ecrire est plus longue que necessaire, elle est dans tous les cas tronquee a droite.

MKx\$ et CVx: La premiere instruction transforme un nombre en chaine de caracteres, la deuxieme effectue l'operation inverse.

A\$=MKx\$(expre) N=CVx(expre\$)

x est l'une des trois lettres "I", "S" ou "D" correspondant aux trois types de variables numeriques : entieres, simple precision, double precision. L'instruction MKx convertit un nombre en une chaine de caracteres dont la longueur depend de la lettre x: 2 pour "I", 4 pour "S" et 8 pour "D". CVx effectue la transformation inverse sur des chaines de longueur correspondante.

"expre" et "expre\$" sont les valeurs a convertir, nombre ou chaine, elles peuvent etre fournies sous forme de constantes, de variables ou d'expressions calculees (123, a1\$, 6*a ou right\$(a\$,4) par exemple).

L'operation suivant la conversion en chaine est normalement la mise en tampon par LSET ou RSET, suivie de l'ecriture par PUT.

La conversion de chaine en nombre est normalement precedee de la lecture par GET.

Note : Dans le cas de variables "chaines de caracteres", aucune conversion n'est necessaire, celle-ci ayant justement pour but de transformer en chaines les objets qui n'en sont pas.

EOF

Detecteur de fin de fichier :

La forme generale de l' ordre est :

EOF(N0tam),

ou "numtam" est le N0 du tampon alloue au fichier que l' on interroge.

Cette instruction verifie si l' on se trouve avant la fin de fichier (un 0 est retourne), ou apres (on recupere -1). Cela permet d' eviter les erreurs 'Input past end' (lecture apres la fin).

EOF(x) renvoyant une reponse logique (vrai=-1 ou faux=0), il n' est pas necessaire pour faire l' essai d' ecrire : IF EOF(1)=-1,... . Mais il suffit d' ecrire : IF EOF(1)... .

Cela pourrait etre utilise dans un programme comme celui-ci :

```
10 OPEN"I",1,"PAYE/JAN:1"
20 IF EOF(1) THEN CLOSE: GOTO100
30 INPUT#1, A$
40 B$(I)=A$:I=I+1:GOTO 20
100 REM : SUITE DU PROGRAMME...
```

LOF: Fournit le numero d' enregistrement le plus eleve.

La forme generale de l' ordre est:

LOF(N0tam)

Donnera le numero du dernier enregistrement sur le fichier passant par la tampon N0 1. Cette instruction fonctionne avec les fichiers sequentiels comme avec les fichiers a acces direct.

Toutefois, pour un fichier a acces direct, elle fournit le nombre d' enregistrements LOGIQUES, qui sera egal au nombre de secteurs si la longueur d' enregistrement est de 256, mais lui sera superieur dans tous les autres cas. En sequentiel toutefois, BASIC n' a aucun moyen de savoir comment les donnees sont reparties sur le disque, et renvoie pour cela le nombre de secteurs.

Il arrive souvent en acces direct, que l' on souhaite regarder chaque enregistrement du fichier. Un programme qui le fait pourrait avoir l' aspect suivant :

```
10 OPEN"R",1,"NOM/INX:2",12
20 FIELD 1,10 AS NOM$, 2 AS RANG$
30 FOR I=1 TO LOF(1)
40 GET 1,I
50 IF CLE$=NOM$ THEN GET 2, CUI(RANG$):GOTO100
60 NEXT I
70 PRINT " INTROUVABLE !": CLOSE: GOTO1000
100 REM : SUITE DU PROGRAMME .
```

Il permettrait d' examiner tous les enregistrements.

LOC

Fournit le NO du dernier enregistrement auquel on a accede.

La forme generale de l'ordre est :

LOC(numtamp)

On peut aussi utiliser PRINT LOC(1), ou x=LOC(1) .

EXEMPLE -

PUT 1, LOC(1)

Ecrirait dans le dernier enregistrement traite.
Cette instruction n'est precise que pour l'accès direct.

Codes d' erreur du BASIC disque

Les codes d' erreur du BASIC disque de DOSPLUS sont, pour des raisons de compatibilite, les memes que ceux des autres SED. En voici la liste :

Code	Message d' erreur	
0	0	NEXT without FOR (NEXT sans FOR)
2	2	Syntax error (Erreur de syntaxe)
3	4	RETURN without GOSUB (RETURN sans GOSUB)
4	6	Out of DATA (Manque de DATA)
5	8	Illegal function call (Appel de fonction illegal)
6	10	Overflow (Depassement)
7	12	Out of memory (manque de memoire)
8	14	Undefined line number (NO de ligne in inconnu)
9	16	Subscript out of range (Indice hors limites)
10	18	Redimensionned array (tableau redimensionne)
11	20	Division by zero (Division par zero)
12	22	Illegal direct (Direct illegal)
13	24	Type mismatch (desaccord de types)
14	26	Out of string space (manque d' espace chaines)
15	28	String too long (Chaine trop longue)
16	30	String formula too complex (Formule trop complexe)
17	32	Can't continue (impossible de continuer)
18	34	no RESUME (pas de RESUME)
19	36	RESUME without error (RESUME sans erreur)
20	38	Unprintable error (Erreur inconnue)
21	40	Missing operand (operande manquant)
22	42	Bad file data (Mauvaises donnees de fichier)
23	44	DISK BASIC feature (reserve au BASIC DISQUE)
24	46	Undefined User Function (Fonction utilisateur non definie.)
51	100	Field overflow (depassement de champ)
52	102	Internal error (erreur interne)
53	104	Bad file number (mauvais NO de fichier)
54	106	File not found (fichier non trouve)
55	108	Bad file mode (Mauvais type de fichier)
56	110	File already open (fichier deja ouvert)
58	114	Disk I/O error (Erreur d' E/S disque)
59	116	File already exists (ce fichier existe deja)
62	122	Disk full (disque plein)
63	124	Input past end (lecture apres la fin)
64	126	Bad record number (numero de bloc incorrect)
65	128	Bad file name (nom de fichier incorrect)
66	130	Mode mismatch (desaccord de modes)
67	132	Direct statement in file (le fichier n' est pas un programme)
68	134	Too many files (fichiers trop nombreux ou NO de fichier trop grand)
69	136	Disk write protected (disque protege en ecriture)
70	138	File access DENIED (Acces au fichier refuse)

CMD

Permet d'executer certaines fonctions speciales sous BASIC.

La forme generale de l'ordre est :

CMD"choix"

choix est un caractere unique qui determine la fonction a engager.

A ne pas confondre avec la fonction CMD du BASIC Etendu, qui vous permettra d'utiliser des fonctions du SED depuis BASIC et d'y retourner. Nous sommes pour le moment uniquement interesses par les fonctions communes a BASIC et TBASIC.

Ordres et choix

CMD

Retour a DOSPLUS

CMD"D"

Lance le DEBUG

CMD"E"

Affiche le dernier message d'erreur de DOSPLUS. De nombreuses erreurs differentiees par DOSPLUS causent l'affichage du meme message d'erreur par BASIC. Cet ordre vous permet d'interroger la liste d'erreurs de DOSPLUS, qui est plus etendue que celle de BASIC, pour clarifier les choses.

CMD"R"

Autorise les interruptions. Sur Modele I, devrait etre actionne apres chaque operation cassette.

CMD"S"

Retour a DOSPLUS

CMD"T"

Interdit les interruptions. Sur Modele I, doit etre actionne avant chaque operation cassette, qui sera suivie d'un CMD"R". Sur Modele III, ces operation sont executees automatiquement par la routine cassette.

Instructions propres au BASIC disque etendu

Cette section couvre les instructions propres au BASIC disque etendu. Alors que la section precedente couvrait les instructions BASIC disque communes BASIC et a TBASIC, mais les commandes documentees ci-dessous exigent le BASIC Disque etendu.

Instruction	Fonction
CMD".....	Commande du DOS depuis BASIC.
DI.....	Deplace une ligne de programme BASIC
DU.....	dedouble ""
Bref.....	Abreviations de commandes d' edition.
RENUM.....	renumerotation de texte de programme BASIC.
TAB.....	Extension de la fonction TAB
TRON.....	"" TRON
REF.....	Donne les references des variables, NO de lignes et mots-cles BASIC.
CMD"M".....	Affichage dynamique des variables.
SR.....	Edition globale du texte BASIC
CMD"O".....	tri de tableaux BASIC
INPUT@.....	Saisie ecran controlee (Chaines)
LABELS.....	Adressage indirect par etiquettes
Messages d' erreur.....	Affichage detaille des messages d' erreur

CMD" (utilisation d' ordres de DOSPLUS sous BASIC)

Forme generale de l' ordre :

CMD"commande DOSPLUS"

"commande DOSPLUS" est toute commande valide de la LIBRARY et des utilitaires.

La commande CMD a ete amelioree pour permettre l' execution de tous les ordres DOSPLUS a partir du BASIC, et d' y retourner apres leur execution, le programme et les variables etant intacts, ce qui permet la reprise du programme. Cela peut meme etre fait par le programme. "Commande DOSPLUS" peut etre une constante OU une variable chaine de BASIC.

EXEMPLE: CMD"DIR"
equivaut a
CMD\$

si D\$="DIR", Il est donc possible d' inclure la commande DOSPLUS dans une chaine BASIC et de faire un controle d' erreur avant de la lancer.

CMD"CLEAR "+FS\$+" (DATA=ESH)"

En supposant que FS\$="ESSAI/DAT", cette instruction rempra le fichier ESSAI/DAT avec l' octet ESH. Remarquez l' emploi d' une variable chaine comme ordre SED. Remarquez aussi le "+" entre la variable et les lettres. Vous devrez l' utiliser pour combiner les deux.

CMD"FREE %2 to @PR

Enverrait la carte des granules libres du disque 2 sur l' imprimante.

CMD"LOAD SOUSPRO/DEJ:3"

Chargerait le programme "SOUSPRO/DEJ:3" et rendrait la main au BASIC.

CMD"JOIN @DO @PR"

Reproduirait la sortie video sur l' imprimante.

Note : Certains utilitaires et ordres de DOSPLUS peuvent utiliser de la memoire hors de la zone normale des annexes (overlays), comme DIR et CAT avec l' option A, ce qui pourrait troubler le programme BASIC present. Mefiez vous donc quand vous utilisez ces options.

DI (deplacer une ligne de programme BASIC)

Forme generale de l' ordre :

DI aln,nnl
"aln" est l' ancien numero de ligne
"nnl" est le nouveau numero de ligne

Donnera a la ligne dont le numero etait "aln", le numero "nnl", les branchements ne suivent pas.

DU (dupliquer une ligne de programme BASIC)

Forme generale de l'ordre :

DU anl,nnl

reproduira la ligne "anl" en "nnl". La ligne subsistera avec les deux numeros. Les branchements ne suivent pas non plus.

Bref (abreviations de commandes)

Plusieurs commandes d'edition ont ete ajoutees pour faciliter l'edition des programmes BASIC. Certaines sont accessibles par une commande abregee. En addition aux commandes classiques d'edition du TRS-80 on dispose maintenant des commandes suivantes:

;	(point-virgule)	- affiche la premiere ligne du programme
shift fleche haut		- liste la premiere ligne du programme.
/ (slash)		- liste la derniere ""
fleche bas		- liste la ligne suivante
fleche haut		- "" precedente
L		- abreviation de LIST (l10-30)
D		- "" DELETE (d10-30)
E		- EDIT
G		- abreviation de GOTO
A		- AUTO
N		- "" NAME
R ou R"		- RUN ou RUN"
L"		- LOAD"
S"		- SAVE"
K"		- KILL"
.	(point)	- liste la ligne courante
,		- edite ""

Ces abreviations ne sont valables qu' en BASIC, leur usage en TBASIC provoquerait des erreurs.

Celles qui utilisent des caracteres non alphbetiques ne peuvent etre utilisees qu' en tete de ligne, en mode commande, pas dans un programme.

Notes techniques : Un caractere efface empeche les commandes abregees de fonctionner. Il faut taper <enter> pour redonner sa fraicheur a la ligne de commande.

Les abreviations alphabetiques peuvent etre utilisees n' importe ou dans une ligne de programme, BASIC les developpera a leur valeur normale quand il les rencontrera.

De plus, l' utilisation de fleches peut provoquer des listages incorrects de lignes plus longues que 240 caracteres. Utilisez alors la forme normale.

RENUM (renumerotation de lignes d' un programme BASIC)

Forme generale de l' ordre :

CMD"RENUM",nnl,inc,anl,lfn

"nnl" est le nouveau numero de la premiere ligne a renumeroter, 10 par default.

"int" est l' intervalle de renumerotation, 10 par default.

"anl" est le numero de la premiere ligne a renumeroter, la premiere du programme par default.

"lfn" est le numero auquel on veut arreter la renumerotation. Fin du programme par default. Si elle existe, elle est renumerotee.

Cet utilitaire de renumerotation change les numeros de toutes les lignes et les references correspondantes du programme comprises dans la fourchette definie par "anl, lfn". Il verifie que tous les branchements se font a des lignes existantes. En cas d' erreur, il abandonne, en affichant les numeros de ligne inexistantes. Le programme a renumeroter doit etre en memoire.

Ce renumeroteur ne deplace pas les blocs de lignes.

Note : on peut aussi remplacer la liste des parametres par "!", limitant ainsi l' action de RENUM a un controle d' erreurs.

EXEMPLES: CMD"RENUM", !

recherchera les erreurs eventuelles.

CMD"RENUM",10,5

renumerotera le programme entier, en commençant au numero 10 et avec un pas de 5.

CMD"RENUM",100,10,80,150

renumerotera la partie du programme commençant a la ligne 80, jusqu' a la ligne 150 (ou la ligne dont le numero est le plus proche par default si cette ligne n' existe pas), le nouveau bloc commencera au numero 100 avec un intervalle de 100.

Pour deplacer un bloc, sauver le programme sur disque, si ce n' est deja fait. effacer tout le programme sauf le bloc considere, renumeroter celui-ci et le sauver sous un autre nom avec le suffixe ,A. Recharger ensuite l' ancien programme, en oter le bloc modifie, faire un MERGE avec le bloc renumerote (apres lui avoir eventuellement fait de la place). Enfin mettre a jour les branchements concernes.

Il est possible de ne pas mentionner l' un ou l' autre des parametres en laissant sa place vide entre les virgules:

CMD"renum",10,,5,20

La place du pas est vide, celui-ci sera pris egal a 10.

Les bornes du bloc a renumeroter peuvent ne pas exister, elles seront remplacees par les numeros de ligne les plus proches, a l' interieur de la fourchette.

TAB (TAB pour l'imprimante).

Forme generale de l'ordre :

LPRINT TAB(pos)

pos est la valeur desiree

Permet d'utiliser cet ordre avec des numeros de colonne plus grands que ce que permet la ROM BASIC. Si vous l'utilisez sur l'ecran avec PRINT, il fonctionnera comme d'habitude avec la ROM BASIC.

Toutefois, si on l'utilise avec un LPRINT, La valeur du TAB peut dépasser 64.

TRON - (fonction d'execution pas a pas).

Forme generale de l'ordre :

TRON

Il n'y a pas de parametre.

Est entre comme en BASIC niveau II. Toutefois on progresse maintenant dans le programme instruction par instruction, en pressant <enter> a chaque pas. <break> fera sortir du programme. Les instructions sont affichees au fur et a mesure, precedees de ":". En TBASIC, TRON fonctionne comme d'habitude.

RUN

en mode TRON fera afficher ":RUN" sur l'ecran.

REF (references du programme BASIC).

Forme generale de l'ordre :

CMD"REF",par,par

'par' peut etre l'un quelconque des parametres valides pour cette commande.
Ces parametres sont :

param.	fonctions
S=	une variable, NO ligne ou mot cle.
V	toutes les variables
L	Tous les numeros de ligne.
K	tous les mot-cles
P	Impression

REF permet de choisir le parametre a etudier : NOs de ligne (L), variables (V), Mots-cles (K).

CMD"REF",K,V,L

les appellerait tous les trois.

Pour appeler une seule variable on utilise la syntaxe "S=":

```
CMD"REF",S=A
```

afficherait les numeros de toutes les lignes ou la variable 'A' apparait.
Il peut devenir aussi precis que vous :

```
CMD"REF",S=A$(
```

Affichera les numeros des lignes ou apparait l' une des variables du tableau A\$(.
La meme syntaxe s' applique pour un numero de ligne ou un mot-cle isole.

CMD"M" (Liste dynamique des variables)

Permet de voir les valeurs des variables presentement allouees.

Forme generale de l' ordre :

```
CMD"M"
```

Vous pouvez susopendre le listage par shift @, l'arreter par <break>.

Si aucune variable n' est utilisee, rien ne sera affiche. Ce n' est pas un utilitaire de reference. Pour imprimer, entrer 'CMD "M",P'.

SR (edition globale de texte BASIC).

Permet de chercher et afficher tout caractere litteral ou toute expression en ASCII qui existe dans un texte BASIC.

Forme generale de l' ordre :

```
CMD"SR",expc,expr,lnd-lnf
```

'expc' est l' expression cherchee. Ce peut etre une chaine quelconque de caracteres ne contenant pas de ''',

'expr' est l' expression eventuelle de remplacement, egalement chaine de caracteres.

'lnd' est la ligne de debut

'lnf' est la ligne de fin

C' est un tres bon outil de programmeur. Il va chercher et afficher toutes lignes ou se trouve l' expression recherchee, ou si un remplacement a ete demande, il affiche les lignes corrigees.

Exemple :

```
CMD"SR","TEST"
```

recherchera dans tout le programme la chaine "TEST", et affichera toutes les lignes ou figure cette expression, c' est le mode recherche.

```
CMD"SR","TEST31","essai31",200-300
```

Recherchera la chaine "TEST31" a partir de la ligne 200, la remplacera par "essai31" chaque fois qu' il la trouvera et affichera la ligne nouvelle. Il continuera jusqu' a la derniere ligne

avant celle qui a le numero 300.(si celle-ci n' existe pas, il ira jusqu' au bout du programme).

=====

CMD"O" (tri de tableaux BASIC).

Permet de trier des tableaux de tous types, en ordre ascendant ou descendant.

Forme generale de l' ordre :

CMD"O",exp,+ ou AN(ed)+KA-KA,TA,TA

'exp' est une expression qui indique le nombre d' elements a trier (entier)
 '+ or -' indique si le tableau cle doit etre trie en ordre ascendant ou descendant. L' ordre ascendant sera pris par default .
 'AN(db)' est le principal element du tableau cle,db est le premier element du tableau a partir duquel le tri doit commencer.
 '+KA' indique le tableau a trier suivant, "+" indique l' ordre ascendant.
 '-KA' indique le tableau a trier suivant, "-" indique ordre descendant.
 ',TA' premier tableau accessoire.
 ',TA' tableau accessoire suivant.

Un tableau cle est un tableau que CMD"O" prendra en compte pour le tri. Un tableau accessoire est par contre un tableau qui n' est la qu' en accompagnement. Quand CMD"O" trouve deux elements du tableau cle a permuter, il permutera aussi les elements correspondants de tous les autres tableaux cles et des tableaux accessoires.

Tous les elements cles doivent etre definis avant de definir les elements accessoires. Les premiers doivent etre prefixes d' un "+" ou d' un "-", sauf eventuellement l' element principal, pour lequel "+" est la valeur par default. N' utilisez pas de virgules entre l' element cle principal, qui est lui-meme separe du nombre d' elements par une virgule pour des raisons de compatibilite avec TRSDOS, et le dernier des elements cles. En effet la premiere virgule indiquant a CMD"O" le debut des elements accessoires, ce signe ne doit pas apparaitre dans la liste des elements cles, principaux ou secondaires.

EXEMPLE -

CMD"O",100,A\$(1)+B\$-C\$,D\$,E\$,F\$

Cet ordre indique a CMD"O" de trier 100 elements d' un tableau de chaines, en commençant par le N° 1 dans le tableau A\$(. Si il en trouve deux identiques, il tentera d' affiner le tri avec les elements B\$. Si il y a la aussi identite, il va poursuivre sa tentative avec les elements C\$, Mais la, le tri se fera en ordre inverse. Chaque fois qu' un element est permute, tous les elements correspondants le sont egalement.

B\$,C\$,E\$,F\$ representent les elements des tableaux B\$(x),C\$(x),D\$(x),E\$(x),F\$(x). Des "elements correspondants" sont des elements de meme rang. Par exemple les elements correspondants de A\$(2) dans l' exemple ci-dessus sont B\$(2),C\$(2),D\$(2),E\$(2) et F\$(2).

Ce tri fonctionne aussi avec des variables numeriques des trois types.

Exemple d' application

L' exemple de programme suivant va creer un index trie pour une liste d' envois :

5 CLEAR 2000:CLS

```

10 OPEN"R",1,"POSTE/DAT",52
20 FIELD1,10 AS FACTICE$,20 AS NOM$
30 EF=LOF(1):DIMA$(EF),RNZ(EF)
40 FORI=1TOEF
50 GET1,1
60 A$(I)=NOM$:RNZ(I)=LOC(1):NEXTI
70 CLOSE
80 CMD"O",EF,A$(1),RNZ
90 OPEN"R",1,"POSTE/INX",2
100 FIELD1,2ASNR$
110FORI=1TOEF
120 LSET NR$=MKI$(RNZ(I)):PUT1,I:NEXT I
130 CLOSE

```

Après l' avoir fait executer, il suffit pour avoir une liste alphabétique, d' ouvrir le fichier "poste/inx", qui contient les numeros des enregistrements et "poste/dat" qui contient les enregistrements. On lit chacun des numeros dans l' ordre ou ils sont écrits, puis l' enregistrement vers lequel il pointe, ce qui donne la liste alphabétique.

INPUT@ (saisie controlee a l' ecran) -

Permet de saisir des chaines de caracteres en en choisissant la position et le format.

Forme generale de l' ordre :

```

INPUT@<pos>,"invite",lt,it;var$

```

'<pos>' est la position d' ecran ou l' on veut faire la saisie (calculee comme pour PRINT@). L' invite va commencer a s' y ecrire, s' il y en a une.
 'invite' est le texte de l' invite eventuelle. Facultatif, ce parametre doit être une constante chaines (pas une variable).
 ',lt' indique la longueur maximum du texte (champ) que l' on attend.
 ',it' indicateur du type d' entree. doit etre '\$' pour une chaine alphanumerique, '#' pour des chiffres seulement. S' il est suivi d' un asterisque, le retour se fera des que le champ sera plein. Ce peut etre une constante ou une variable.
 ';var\$' est le nom de la variable destinee a recevoir l' entree. C' est obligatoirement une variable chaine, et elle est obligatoirement separee de 'it' par ';

Cette instruction sert a remplacer les INKEY\$ fatigants qu' il est souvent necessaire d' employer, et qui, par suite des manipulations de chaines qu' ils entraînent et qui sont lentes en BASIC, ralentissent considerablement la reponse du clavier.

Bien que INPUT@ fasse peu de controle d' erreur par lui-meme, il vous donne la possibilite d' en faire d' aussi sophistiques que vous le desirez.

EXEMPLES: INPUT@523," Entrer le nom :",20,"\$";NA\$

affichera la chaine d' invite en commençant a la position d' ecran 523 et affichera un champ de 20 tirets. Elle attendra <enter> ou <clear> pour finir et mettre la reponse dans la variable NA\$.

INPUT@645,4,"x";SI\$

n' affichera pas d' invite car il n' en a pas ete fournie, mais affichera un champ de 4 caracteres a la position 645. Le retour se fera des que le 4 eme caractere (ou <ENTER>) sera

tape, si vous tapez autre chose qu' un chiffre, il fera le mort et ne prendra rien.

Le retour pour un champ plein est surtout utile quand on attend une seule touche en évitant l' action permanente sur <Enter>.

=====

LABELS (branchements indirects dans un programme BASIC).

Cette fonction vous permet d' utiliser l' adressage indirect en BASIC. Pour la réaliser, nous avons remplacé la fonction NAME du TRSDOS par une autre de notre cru.

Forme générale de l' ordre :

```
NAME etiquette
GOTO etiquette
GOSUB etiquette
```

NAME affecte l' etiquette designée à la ligne où apparaît la déclaration de l' etiquette. Vous utilisez ensuite cette etiquette exactement comme un numéro de ligne quand vous utilisez les ordres GOTO et GOSUB.

Les restrictions sont les suivantes :

Une etiquette ne doit pas contenir de mots	reserves.
Elle ne doit pas avoir plus de 240 caracteres de	long.

Nous avons aussi modifié RENUM pour qu' il ne saute pas quand il trouve une etiquette en renumérotant un programme.

Le mot cle NAME doit être le premier élément d' une ligne, par exemple :

```
10 NAME ESSAI:FOR A=1 TO 10
20 Suite du programme...
100 GOTO ESSAI
```

Le mot NAME est le premier élément de la ligne. Dans le cas contraire, il sera considéré comme une remarque, et toute référence qui y serait faite résulterait en une erreur.

Exemple :

```
10 NAME PRINTEMPS
```

est une mauvaise etiquette (le mot cle PRINT fait partie de l' etiquette).

=====

Messages d' erreur détaillés.

Cette fonction de DOSPLUS n' est pas un ordre au sens habituel, mais plutôt une manière de fonctionner qui mérite d' être présentée.

Quand une erreur arrive sous BASIC étendu, le message d' erreur est présenté avec l' instruction mise en cause, désignée par une flèche.

Par exemple :

```
10 FOR I=1 TO 10
20 PRINT "C' EST UN ESSAI",
30 X=C:NEXT J
```

Dans cet exemple, il y a une erreur NEXT sans FOR a la ligne 30. Notre boucle est controlee par la variable I, et nous ecrivons un ordre NEXT J.

L' affichage sera quelquechose comme :

```
NEXT without FOR in 30
-> :NEXT J
```

La fleche pointe toujours sur l' instruction qui contient l' erreur, quelle que soit la longueur de la ligne, mais n' indique pas l' element defectueux, qu' il vous appartient de trouver.

Pilotes et Filtres

	<u>Nom de la Section</u>	<u>Page N°</u>
Pilotes	KI/DVR	6/2
	DO/DVR	6/4
	PR/DVR	6/5
	RS/DVR	6/7
	FILE/DVR	6/8
Filtres	DVORAK/FLT	6/9
	EPSON/FLT	6/9

--- KI/DVR ---

C' est le pilote clavier de DOSPLUS, qui offre des possibilites comme un anti-rebonds le delai avant repetition et la repetition des touches reglables, les Macrotouches, et le filtrage des caracteres.

```
=====
ASSIGN <FROM> @KI <TO> KI/DVR <USING> nomfich (param=exp)
```

nomfich est le nom de votre eventuel fichier de definition des Macrotouches.

Les parametres admis sont :

(D)B=valeur Delai d' anti rebonds

RD=valeur Delai avant la repetition des touches.

RR=valeur Cadence de repetition

Le pilote clavier de DOSPLUS 3.5 offre de nombreuses possibilites, il supporte la traduction de caracteres en utilisant des tables installees par l' ordre FILTER de DOSPLUS.

Le programme KI/DVR donne aussi acces a tous les codes ASCII par le clavier. Le clavier standard du TRS-80 ne permet pas l' entree de nombreux caracteres speciaux du code ASCII, mais ils peuvent etre entres avec KI/DVR. Les code de controle habituels, control-A a controle Z peuvent etre entres en pressant <shift fleche/bas> en mee temps que la touche alphabetique correspondante. Par exemple, pour generer control-R, pressez <shift fleche/bas R>

Les codes ASCII absents du clavier sont generes de la meme maniere, c' est a dire en pressant <shift fleche/bas> et une autre touche. La table suivante donne la correspondance entre cette touche et le code genere. :

Touche	caractere
-----	-----
Controle-6	accent circonflexe
Controle-7	equivalent
Controle-8	crochet ouvert
Controle-9	trait vertical
Controle-	souligne
Controle-;	DEL,7FH
Controle-/	le meme caractere inverse.
Controle-,	Accolade ouverte
Controle-.	Accolade fermee

Note du traducteur : L' imprimante dont je dispose ne connaissant pas ces caracteres, je ne peux qu' essayer de les decrire.

Quand vous installez le programme KI/DVR avec l' ordre ASSIGN, vous pouvez specifier certaines valeurs. La premiere, DB, sert a determiner le delai utilise pour eviter les caracteres multiples dus aux rebonds. Normalement, la valeur par defaut convient bien, mais vous pouvez la modifier suivant vos desirs. A noter que la valeur de DB n' est pas directement proportionnelle a la valeur de l' anti-rebonds, mais celle-ci est plutot proportionnelle au produit des deux octets du delai. Cette relation existe aussi pour les deux autres parametres.

Le parametre RD permet a l' operateur de definir le temps qui passe avant le demarrage de la fonction de repetition. La valeur par defaut convient dans la plupart des cas, mais peut etre

modifiée a volonte.

Le parametre RR permet d'ajuster la vitesse de repetition des touches. La valeur par defaut convient dans la plupart des cas, mais peut etre modifiée a volonte.

Le pilote clavier de DOSPLUS a une capacite de 'Macrotouches'. Les Macrotouches sont des touches programmables, qui peuvent contenir un nombre quelconque de touches elementaires. Par exemple, nous pourrions definir la touche D de telle sorte qu'elle contienne les touches "DIR :1 (S,I,A)". Puis, si nous pressons <CLEAR> et D, nous pourrions demander ce catalogue en pressant <CLEAR A>.

Avant que ce systeme puisse fonctionner, il faut bien sur ecrire le fichier de definition des touches, en utilisant par exemple BUILD. Un fichier Macrotouches contient une ou plusieurs lignes de definition de touches. Chacune a le format suivant :

x=chaine

ou x est n'importe quelle touche, et "chaine" la suite de caracteres a programmer. Pour revenir a l'exemple ci-dessus, nous entrerions la ligne suivante :

D=DIR :1 (S,I,A)/. (/.= Controle-7)

Le caractere Controle-7 est utilise par KI/DVR pour representer un retour ou <enter>. A chaque fois que l'on veut inclure un retour chariot dans une definition de touche, il faut utiliser ce caractere.

A noter qu'un fichier de definition de Macrotouches peut contenir des lignes de commentaires (nouveau ?).

Pour utiliser une Macro touche, pressez <clear> suivi de la touche correspondant au texte choisi. Chaque fois qu'un programme reclame un caractere au clavier, il va aller le chercher dans cette Macro touche, jusqu'a epuisement du stock. Il prendra alors ses informations directement au clavier. Pour utiliser la touche <clear> normalement, pressez la deux fois.

Des Macro touches peuvent etre liees en utilisant le Controle-7. Prenons par exemple les deux definitions suivantes :

B=BASIC/LOAD"MENU/BAS"Controle-/L

L=CMD"FORMS P=66,W=80,L=30Controle-7LLISTControle-7

Utiliser la macro touche L fera executer l'ordre FORMS depuis BASIC et imprimer le listage du programme BASIC resident en memoire. Utiliser la Macro touche B fera charger le BASIC depuis le SED et Charger le programme MENU/BAS. Le caractere L dans cette definition indiquera au clavier d'enchaîner avec la Macro touche L, c'est a dire de faire comme si l'on avait presse <clear> L.

Il faut etre prudent pour relier des Macro touches ensemble, et en particulier, eviter de faire des boucles, d'ou il ne serait possible de sortir qu'avec le bouton RESET.

DO/DVR

C' est le programme pilote de la visualisation de DOSPLUS. Il offre la conversion de caracteres sur le Modele I et le Modele III, et permet l' usage d' caractere special, ainsi que le support des minuscules sur le Modele I.

=====

```
ASSIGN <FROM> @DO <TO> DO/DVR
```

Les parametres admis sont :

(U)CASE (majuscules) = choix Active ou desactive le support minuscules (Modele I)

(B)LINK=valeur Regle la vitesse de clignotement du curseur (Modele I).

=====

Le pilote video de DOSPLUS supporte la conversion de caracteres avec des tables installees par l' ordre FILTER de DOSPLUS. Sur le Modele I, il donne aussi la possibilite d' utiliser pour le curseur un autre caractere que le souligne habituel. Tout cela est presque automatique, et l' utilisateur n' a besoin que d' executer l' ordre ASSIGN vu ci-dessus.

Le parametre UCASE, si mentionne dans la ligne d' ordres, provoquera la conversion par DO-/DVR de toutes les minuscules en majuscules avant leur affichage.

Le parametre BLINK donne un moyen de regler la vitesse de clignotement du curseur. Il peut etre mis a toute valeur de la 15, 1 etant le plus rapide et 15 le plus lent.

PR/DVR

Pilote d'imprimante de DOSPLUS, qui permet le controle des imprimantes serie comme des imprimantes paralleles, le SPOOLING (bobinage, ou tampon d'imprimante), la mise en page automatique, l'indentation et autres.

=====

ASSIGN <FROM> @PR <TO> PR/DVR (param=exp)

Les parametres autorises sont :

(L)F=choix	Declenche l'envoi de suts de ligne apres les retour/chariot
(N)ULL=choix	fait ignorer par l'imprimante les lignes vides.
(S)POOL=valeur	Taille du tampon facultatif.
(M)AX=valeur	Nombre de tentatives pour envoyer un caractere pendant une interruption du spooler.
(I)NDENT=valeur	Nombre d'espaces dont le texte va etre decale a droite.
(X)LATE=choix	provoque la conversion des caracteres de saut de page.
(S)ERIAL=choix	Indique une imprimante serie.
(D)ELAY=valeur	Indique le nombre de caracteres nuls apres un LF ou un CR.
(C)TS=choix	parametre facultatif de coordination
(DS)R=choix	idem
CD=choix	idem
(R)I=choix	idem
(INV)ERT	coordonne avec une imprimante serie dont les signaux sont inverses.

=====

Le pilote d'imprimante de DOSPLUS offre une souplesse et une versatilite sans pareilles. A cote du support de la puissante structure de filtrage de DOSPLUS, il offre le tampon d'imprimante (SPOOLING), l'indentation, le support des imprimantes serie et parallele, et d'autres caracteristiques sophistiquees.

De nombreuses imprimantes necessitent l'envoi d'un caractere saut de ligne (LF) apres chaque retour chariot (CR). Si le parametre LF est actif, le pilote enverra ces sauts de ligne, il est inactif par default.

Le parametre NULL sert a dire au pilote d'ignorer les lignes vides, celles qui ne consistent qu'en un retour chariot, quand il est actif. Il est inactif par default.

Le parametre SPOOL peut etre utilise pour fixer la taille en octets du tampon d'imprimante. Si aucune valeur n'est donnee pour SPOOL, il ne fonctionne pas. Autrement, le pilote va reserver en memoire la place de ce tampon, toute sortie vers l'imprimante transitera par ce tampon, ce qui, du fait que les donnees vont plus vite en memoire que sur le papier, permet d'etaler les ralentissements du programme causes par l'imprimante, car les donnees vont ensuite vers celle-ci a son rythme, meme si la machine ne lui envoie plus rien, et jusqu'a ce que le tampon soit vide.

Le parametre MAX permet de determiner la frequence d'envoi des donnees vers l'imprimante. Sur le Modele I, le 'SPOOLER' essaie d'envoyer des donnees 40 fois par seconde, 30 fois sur le Modele III. Le parametre MAX determine combien d'essais d'envoi de caractere a l'imprimante a chacune de ces 40 ou 30 fois par seconde. D'une facon generale, plus il en fera, plus vite le texte sera imprime, mais plus il fera d'essais, moins l'efficacite globale du systeme sera

bonne.

Le parametre INDENT permet de deplacer la marge gauche de l' imprimante, d' un nombre quelconque d' espaces. Par exemple, un INDENT=10 ferait commencer chaque ligne 10 caracteres apres la marge normale de l' imprimante. Le default est 0.

Le parametre XLATE determine si des caracteres de saut de ligne et de tabulation verticale sont traduits par le pilote en nombre de sauts de ligne equivalents, ou s' ils sont envoyes tels-quels. Des imprimantes 'intelligentes' qui sont capables d' executer le saut de page peuvent fonctionner avec XLATE inactif, les autres en ont besoin. Il est actif par default.

Le pilote d' imprimante de DOSPLUS supporte les imprimantes serie comme les imprimantes paralleles. Par default, il est adapte aux premieres, mais si l' on utilise le parametre SERIAL, il s' adaptera aux autres. Dans ce cas, toute la sortie imprimante est dirige vers le port RS-232. Les parametres de ce port (bauds, longueur de mots, parite etc) doivent etre fixes avec l' ordre de la librairie RS232.

Si vous utilisez une imprimante serie, vous pouvez fixer le parametre DELAY pour faire envoyer un nombre quelconque (0 a 255) de caracteres nuls apres les LF et les CR. Beaucoup d' imprimantes demandent un certain nombre de caracteres nuls pour permettre a leur mecanisme d' executer ces operations de longue duree. Fixer le parametre DELAY a la valeur recommandee pour l' imprimante permet de l' utiliser correctement.

Quatre parametres ,CTS, DSR, CD et RI servent a indiquer a l' imprimante serie lesquelles de ces lignes doivent etre hautes ou basses avant l' envoi d' un caractere. Comme de nombreuses imprimantes signalent qu' elles sont pretes a recevoir en modifiant l' etat de certaines de ces lignes, ces parametres permettent d' indiquer au pilote celles qu' il doit surveiller. Par exemple si des donnees doivent etre envoyees a une certaine imprimante quand CTR et DSR sont haut, l' affectation du pilote pourrait etre le suivant :

```
ASSIGN @PR PR/DVR,SERIAL,CTS,DSR
```

Le parametre INVERT est utilise en liaison avec les quatre parametres de coordination pour indiquer si les lignes concernees doivent etre en etat haut ou bas avant l' envoi de caractere. Normalement, le TRS-80 opere en logique inverse, mais certaines imprimantes peuvent differer.

RS/DVR

C'est le pilote RS232 de DOSPLUS. Comme tous les pilotes de DOSPLUS, il effectue la conversion de caracteres d'apres des tables installees par l'ordre FILTER.

=====

```
ASSIGN <FROM> @RS <TO> RS/DVR
```

Il n'y a pas de parametre.

=====

Notez que DOSPLUS n'a normalement aucun programme pilote d'interface serie avant que l'ordre ASSIGN ci-dessus ne soit execute, ou qu'un fichier /CFG contenant le pilote ne soit execute, avant de pouvoir executer aucune operation d'E/S utilisant l'interface serie.

FILE/DVR

Permet d'utiliser un fichier resident sur un lecteur de se comporter comme un lecteur a part entiere, et particulierement utile avec les disques durs

```
=====
ASSIGN <FROM> :lect <TO> FILE/DVR <USING> nomfich (param=exp)
```

:lec	designation de lecteur	
nomfich	fichier contenant un 'filedisk' ou disque	fichier'

Les parametres admis sont :

(S)IZE=valeur Informe FILE/DVR d'avoir a initialiser un fichier de la
taille indiquee

(I)NST=choix Informe FILE/DVR de s'installer dans le systeme d'exploit-
tation

Le programme FILE/DVR est installe dans DOSPLUS comme un pilote de lecteur de disques, et sert a permettre a un fichier resident sur un lecteur de faire comme s' il etait lui-meme un lecteur. Essentiellement, FILE/DVR permet a DOSPLUS d'avoir des lecteurs dans ses lecteurs.

Par exemple si l'ordre :

```
ASSIGN :7 FILE/DVR FDISQ:1
```

etait execute, le lecteur 7 existerait sous la forme d'un fichier appele FDISQ (de tels fichiers sont appeles FILEDISKS, ou disques-fichiers), resident sur le lecteur 1. Le lecteur 7 n'est donc pas un lecteur physique, ce n'est qu'un fichier jouant au lecteur. (ndt mais c'est parfois bien pratique).

Le parametre INST sert a indiquer au pilote de s'installer dans le systeme. FILE/DVR differe sur ce point des autres pilotes qui s'installent d'autorite, mais, pour des raisons que nous verrons plus loin, il peut arriver que nous voulions ASSIGNer un NO de lecteur a FILE/DVR, sans vouloir l'installer lui-meme. Si le parametre INST est indique, FILE/DVR va s'installer dans le systeme. Ce parametre DOIT etre utilise si FILE/DVR n'est pas encore installe. Par exemple, si nous voulons affecter au lecteur 3 le fichier Bibliot/un du lecteur 4, nous entrerons :

```
ASSIGN :3 FILE/DVR BIBLIOT/UN:1 (INST)
```

Cet instruction provoquerait l'installation de FILE/DVR et toutes les sorties destinees au lecteur 3 iraient dans le disque-fichier appele BIBLIOT/UN du lecteur 4.

Les fichiers disque que nous denomons disques-fichiers sont crees par FILE/DVR en utilisant le parametre SIZE. Ce parametre indique que a) nous voulons creer un nouveau disque-fichier, b) indiquer la taille que nous souhaitons lui donner. La valeur fournie par le parametre SIZE est la taille desiree en secteurs. La creation et l'affectation peuvent etre executees en une seule operation. Par exemple :

```
ASSIGN :2 FILE/DVR MONFICH:4,INST,SIZE=50
```

Notez le parametre INST, qui indique que le pilote n'etait pas encore installe.

Une fois le lecteur affecté à un disque-fichier, il va fonctionner presque exactement comme un lecteur normal (sauf pour des fonctions disque pures comme FORMAT et BACKUP), mais vous pourrez en lire le catalogue, ou le directory, utiliser DIRCHECK, etc.

Une fois le pilote FILE/DVR installé, d'autres lecteurs peuvent être affectés à d'autres disques-fichiers. Par exemple nous pourrions envoyer l'ordre suivant :

```
ASSIGN :5 FILE/DVR COMME:3
```

ou ne figure pas le paramètre INST, puisque FILE/DVR est déjà en place.

L'utilité principale de FILE/DVR est pour les disques durs, quoiqu'il puisse rendre service avec des disques souples. Il facilite l'utilisation optimisée de l'espace disque, en accroissant l'espace du directory. Chaque disque-fichier peut contenir jusqu'à 48 noms de fichiers, alors que le disque-fichier lui-même n'en occupe lui-même qu'un sur le directory du disque où il réside (à moins qu'il ne soit segmenté).

Ndt: Une des raisons de cette efficacité est que sur ces disques-fichiers, l'espace n'est pas affecté par granules, mais par secteurs. C'est donc un bon endroit pour ranger de petits programmes, au lieu d'occuper 6 secteurs pour un 200 octets.

DVORAK/FLT

C'est un filtre destiné au périphérique Clavier, @KI

```
=====
```

```
FILTER <FROM> @KI <TO> DVORAK/FLT
```

```
=====
```

Ce filtre redéfinit le clavier du TRS-80 pour le rendre conforme au très efficace clavier DVORAK (NDT ???).

EPSON/FLT

Filtre destiné à l'imprimante, @PR.

```
=====
```

```
FILTER <FROM> @PR <TO> EPSON/FLT
```

```
=====
```

Ce filtre est utilisé avec les imprimantes EPSON MX-80, pour traduire les codes de 128 à 191 (graphiques TRS-80) en codes de 140 à 223, que l'imprimante EPSON utilise pour imprimer les caractères graphiques du TRS-80.

SECTION TECHNIQUE

Cette partie du manuel de DOSPLUS 3.5 contient des renseignements sur les routines internes du systeme, les tables de donnees, et les formats des disques. Ces renseignements seront utiles aux programmeurs en langage machine qui veulent ecrire des logiciels sous DOSPLUS 3.5.

Ce manuel est divise en plusieurs sections:

Section	Objet	Page
I.	Subdivision des disques durs	T/2
II.	Fichiers systeme	T/3
III.	Structure du Directory	T/4
	Organisation de la GAT	T/4
	" " HIT	T/6
	des etiquettes des fichiers	T/6
IV.	Donnees du secteur 2 de BOOT/SYS	T/10
V.	Table des DCB & organisation des DCB	T/10
	DCB de @KI	T/12
	DCB de @DO	T/12
	DCB de @PR	T/13
	DCB d' initialisation de RS232	T/13
VI.	Structure des FCB	T/13
VII.	Organisation des DCT	T/15
VIII.	Points d' entree du systeme	T/18
	routines de manipulation des fichiers	T/19
	" de controle systeme	T/24
	" de chainage des interruptions	T/26
	" systeme diverses	T/28
	vecteurs systeme internes	T/34
	routines ROM utiles	T/37
IX.	Adresses memoire importantes	T/38
X.	Ecritures de pilotes pour DOSPLUS 3.5	T/41
	Pilotes de disques	T/41
	Pilotes de peripheriques	T/42
XI.	Codes d' erreur de DOSPLUS 3.5	T/43

I. - Subdivision des disques durs

Avec DOSPLUS 3.5 il est possible de subdiviser un seul disque dur en plusieurs volumes, chacun d'eux pouvant alors être considéré comme un lecteur indépendant. Cette possibilité a plusieurs utilisations importantes:

1. Elle rend possible l'utilisation de disques portant plus de 200 cylindres (maximum normalement accepté par DOSPLUS 3.5), en permettant de les partager en volumes dont le nombre de cylindres est au plus égal à 200.
2. Elle permet à l'utilisateur de diviser un seul grand disque en plusieurs disques plus petits, chacun pouvant avoir une fonction déterminée, par exemple 'programmes en cours', 'comptabilité', 'paye', 'impôts'.
3. L'utilisateur peut choisir de favoriser le temps d'accès ou l'utilisation de la place, en divisant le disque par décalage de tête ou de cylindre.

Avant d'entrer plus avant dans la présentation des effets et des avantages de la division des disques durs, nous allons rappeler brièvement le fonctionnement des disques durs, décrit avec l'ordre CONFIG de la librairie.

Une unité de disques durs comprend plusieurs disques magnétiques tournant à grande vitesse. Chacun d'eux a deux faces. Pendant qu'il tourne, une tête de lecture/écriture 'vole' au-dessus de chaque face, portée par un bras mobile. Cette tête peut être placée au-dessus de l'une des pistes circulaires (il y en a habituellement de 150 à 400). Toutes les pistes ayant le même numéro forment un cylindre. Par exemple, un lecteur peut comporter 3 disques, soit 6 faces. Si chacune de ces faces porte 306 pistes, le lecteur comprend 306 cylindres. Le premier cylindre est le cylindre 0, à tout moment, toutes les pistes du même cylindre sont survolées par la tête correspondante.

Une façon de diviser le disque est d'affecter à un volume, par exemple les cylindres de 0 à 101, au suivant les cylindres de 102 à 203, et au troisième le reste; c'est la division par décalage des cylindres (le nombre de cylindres peut varier d'un volume à l'autre). Un autre serait d'affecter les têtes 1 et 2 au volume N° 1, les têtes 3 et 4 au N° 2, et les têtes 5 et 6 au N° 3; c'est la division par décalage des têtes.

DOSPLUS 3.5 ne pouvant gérer plus de 200 cylindres par lecteur, un moyen de lui faire gérer notre lecteur de 306 pistes serait d'utiliser la première méthode, (la limitation à 200 pistes vient de raisons de compatibilité avec le TRSDOS, et ne joue que sur les 'designations de lecteur'). D'autre part cette méthode favorise la vitesse d'accès, car pour une même position des têtes, on peut accéder à une plus grande quantité d'informations, et ce qui est long est justement le déplacement de ces têtes, bien qu'il soit plus rapide pour un disque dur que pour des disques souples. Par contre la deuxième améliore l'utilisation de l'espace disque, car chaque cylindre contenant moins de secteurs, les granules seront plus petits, et un granule à peine occupé correspondra à moins de places vides que dans le premier cas.

Un autre avantage de cette subdivision a déjà été mentionné: c'est de pouvoir affecter chaque volume à des tâches différentes, comme on peut spécialiser des disques souples dans des travaux différents (cela ne joue pas sur la sauvegarde des enregistrements).

En conclusion, la subdivision des disques durs est un compromis entre des contraintes éventuelles (capacité), et des souhaits (vitesse et occupation) entre lesquels il faut faire un choix.

II. - Fichiers systeme.

Le systeme DOSPLUS 3.5 est construit sur un schema de remplacements de programmes (overlays): seule une partie du systeme complet reside en memoire a un moment donne, en dehors d' un noyau permanent, il ne charge que la routine necessaire pour le travail a executer. Ce qui permet une capacite plus grande de manoeuvres plus sophistiquees que s' il devait etre charge en permanence et en totalite en memoire.

DOSPLUS 3.5 comprend plusieurs fichiers SYSTEME, qui sont divises en trois groupes:

1. La plaque tournante du systeme : SYS0. C' est le programme qui est responsable du chargement et du lancement des autres programmes, des echanges de base avec les disques et autres peripheriques, il remplit aussi d' autres fonctions qui doivent rester en MEV. SYS0 doit etre present sur tout disque systeme; il reside de 4000H a 40FFH.
2. Le groupe des programmes de zone basse, SYS1-SYS8. Ces programmes remplissent des fonctions essentielles au systeme, telles que l' evaluation des ordres, l' ouverture et la fermeture des fichiers et des peripheriques, les messages d' erreurs, et d' autres taches de niveau moyen. Ils occupent (un a la fois) la region de 4E00H a 51FFH. Si un de ces programmes etait ote du systeme, il pourrait en resulter un fonction erratique.
3. Le groupe de zone haute, SYS9 a SYS16. Ces fichiers contiennent les ordres de la Librairie, comme DIR, CONFIG, DO etc. Ils resident dans la zone de 5200H a 5BFFH. En cas de besoin, des modules peuvent etre enleves de ce groupe, mais SANS garantie du fonctionnement du systeme. Si un module est enleve, les ordres librairie qu' il contient ne sont plus utilisables, et, au cas ou l' un de ces ordres serait appele, la reaction du systeme est imprevisible.

La table suivante detaille les fonctions de chacun des modules:

Fichier	Fonction(s)
Groupe bas	
SYS1	Traduit les ordres, routines d' evaluation des noms de fichiers.
SYS2	OPEN@, INIT@, generation du hashcode et des pieges (trapdoor)
SYS3	CLOSE@, KILL@.
SYS4	Systeme des granules, affectation de l' espace disque.
SYS5	Detection et affichage d' erreur.
SYS6	moniteur DEBUG
SYS7	EVAL@, WILD@ (masques)
SYS8	RANDIR@, CAT@, FILPTR@, SORT@.
groupe haut	
SYS9	CAT, DIR, FREE.
SYS10	APPEND, COPY, LIST.
SYS11	ASSIGN, FILTER, FORCE, JOIN, RESET.
SYS12	AUTO, BREAK, CLOCK, DATE, DEBUG, ERROR, I, LIB, PAUSE, SCREEN, TIME, VERIFY.
SYS13	DO, FORMS, RS232.
SYS14	ATTRIB, KILL, PROT.,
SYS15	BUILD, CLEAR, CREATE, DUMP, LOAD, RENAME.
SYS16	CONFIG, SYSTEM.

III. - Structure du directory

Chaque disque DOSPLUS 3.5 contient, que ce soit un disque donnees ou un disque systeme, contient un fichier appele DIR/SYS, le directory du disque. Le directory contient des informations comme les noms, emplacements, longueur, protection et autres attributs importants pour chacun des fichiers presents sur ce disque. Il est compose de trois tables:

1. La table d'allocation des granules, ou GAT. La GAT contient des informations sur l'espace libre et alloue, en meme temps que d'autres donnees du meme ordre.
2. La table des index de HASH, ou HIT. Cette table contient les codes de HASH (ils sont calcules par le systeme a partir du nom du fichier et d'autres ingredients), et sert au systeme a localiser un fichier dans le directory.
3. La table des etiquettes des fichiers. Cette table, qui occupe habituellement plusieurs secteurs, contient les informations de chaque fichier, comme son nom et son extension, le mot de passe code, le niveau de protection, sa longueur, la date de sa derniere modification, etc.

Organisation du GAT

La table d'allocation des granules, ou GAT, occupe le secteur 0 du fichier DIS/SYS. Outre l'occupation des granules, la GAT contient des informations que nous allons detailler plus loin.

Avant de nous lancer dans l'explication detaillee de la GAT, definissons le terme 'granule'. Le granule est l' unite d'espace disque elementaire pour le systeme. Il ne peut affecter a un fichier un espace plus petit. Sa taille reelle, en secteurs, depend du disque considere. Pour les disques souples, la table suivante est appliquee:

Type de disque	Taille du granule (en sect/gran)
5'' SDEN SF/DF	5
5'' DDEN SF/DF	6
8'' SDEN SF/DF	8
8'' DDEN SF	6
8'' DDEN DF	10

S = Simple, DEN = DENSite, F = Face

Sur les disques durs, la taille des granules est calculee par le systeme, a partir du nombre de secteurs par piste et du nombre de faces entrees par l'ordre CONFIG, pour donner la plus petite taille de granule possible (en respectant les deux contraintes que le nombre de secteurs par cylindre doit etre un diviseur de la taille du granule, et qu'il n'y ait pas plus de 8 granules par cylindre).

Quand DOSPLUS 3.5 affecte de l'espace disque a un fichier, il le fait en nombre de granules. Par exemple, si nous devons creer un fichier de 1286 octets sur un disque 5 pouces DDEN (6 secteurs/granule), DOSPLUS 3.5 affectera un granule entier (1536 octets) a ce fichier. De cette facon, un fichier peut etre allonge sans lui affecter continuellement des secteurs. Le but de la GAT est tenir trace des granules affectes et de ceux qui sont libres.

La table d'allocation fournit une table de tous les granules d'un disque. De l'octet 00H a l'octet 5FH (pour un disque souple), chaque octet de la table correspond au cylindre du disque qui porte le meme numero. Chaque bit de l'octet sert a indiquer (toujours dans l'ordre), si le granule correspondant est libre ou non (0=libre).

En examinant la GAT de la figure 1, nous voyons que l'octet 02H contient la valeur F8H. Converti en binaire, cela donne :

F8H = 1111 1000

Un bit a 1 representant un granule alloue, ou plutot inutilisable, nous voyons que ce cylindre a 3 granules libres (en DDEN/SF, il y a 3 granules par cylindre), il est donc vide.

Un autre exemple est l'octet 20H, qui vaut F9H, soit 1111 1001 en binaire. Il reste donc deux granules libres, les autres etant occupees ou indisponibles pour d'autres raisons (les granules inexistantes ou interdits sont indiquees comme occupees).

Sur les disques durs, la GAT est plus grande, et s'etend de 00H a C7H. La table des granules interdits (voir ci-dessous) n'existe pas, et les cylindres interdits sont simplement catalogues comme occupees.

Les octets 60H a BFH constituent ce que l'on appelle la table des granules interdits. Sa structure est la meme que la table d'allocation, mais son but est different. Pendant le formatage d'un disque, il peut arriver qu'un defaut soit trouve dans une zone limitee du disque. Au lieu de la rejeter, DOSPLUS 3.5 va simplement l'interdire, ce qui se fera en mettant a 1 le bit correspondant de cette table, les bits des granules inexistantes y sont aussi.

Apres la table des granules interdits, l'octet CBH vaut le numero de version de DOSPLUS 3.5 qui a formatte le disque, range comme une valeur BCD.

L'octet CCH indique le nombre de cylindres formats sur le disque, moins 35, qui est en est le minimum.

Le bit 5 de l'octet CDH indique si le disque a ete formatte en SF ou en DF (il est alors a 1). Les autres bits sont reserves pour un usage ulterieur.

Le code (ou piege) du mot de passe du disque est range dans les octets CEH et CFH. Les octets D0H a D7H portent le nom du disque, DBH a DFH la date du formatage; les deux sont justifiees a gauche.

Le reste du secteur, octets E0H a FFH portent l'ordre AUTO a executer a l'initialisation. C'est une chaine de caracteres terminee par un retour chariot (00H). Si ce dernier est seul, il n'y a pas de commande AUTO.

Note: Il est possible de placer un ordre AUTO dans la GAT d'un disque de donnees, soit par l'ordre AUTO, soit autrement, mais l'ordre AUTO execute est toujours celui du disque systeme. Toutefois, une telle commande serait preservee, et pourrait devenir active si le disque est SYSGENE.

```

00 00: F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
14 10: F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
00 20: F9FE F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
30: F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
40: F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
50: F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
60: F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
70: F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
80: F9FE F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
90: F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
A0: F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
B0: F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 F8F8 .....
C0: F8F8 F8F8 F8F8 F8F8 F8F8 FF35 0500 9642 .....5..B
D0: 5359 5354 332E 3544 3038 2F31 312F 3833 SYST3.5008/11/83
E0: 494E 4954 2F43 4647 0020 2020 2020 2020 INIT/CFG.
F0: 2020 2020 2020 2020 2020 2020 2020 2020

```

GAT

Figure 1

```

00 00: A2C4 2E2F 2C2D 2A2B 2829 2627 27A7 26A6 "...,-*+()&'&&
14 10: 0000 0000 0000 0000 0000 0000 0000 0000 .....
01 20: 25A5 24EA 0000 0000 0000 0000 0000 0000 %Z$.
30: 0000 0000 0000 0000 0000 0000 0000 0000 .....
40: 0000 0000 0000 0000 0000 0000 0000 0000 .....
50: 0000 0000 0000 0000 0000 0000 0000 0000 .....
60: 0000 0000 0000 0000 0000 0000 0000 0000 .....
70: 0000 0000 0000 0000 0000 0000 0000 0000 .....
80: 0000 0000 C560 0000 0000 0000 0026 0000 .....&..
90: 0000 0000 0000 0000 0000 0000 0000 0000 .....
A0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
B0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
C0: 0000 0000 0000 0000 0000 0000 5600 0000 .....V...
D0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
E0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
F0: 0000 0000 0000 0000 0000 0000 0000 0000 .....

```

HIT

Organisation de la HIT (Hash Index Table)

Note du traducteur: Le HASH CODE est une technique de rangement qui calcule un emplacement a partir d'un element propre a l'objet range (nom d'un fichier).

La table HIT occupe le secteur 1 de DIR/SYS. Elle sert a DOSPLUS 3.5 a localiser les fichiers dans le directory. A la creation d'un fichier, le systeme genere un code HASH de 1 octet a partir du nom de fichier et de l'extension. Ce code a une valeur de 01H a FFH (00H sert a indiquer une place vide). Ce code est ensuite range dans la HIT, d'ou decoule la place de l'etiquette principale de ce fichier. Par la suite, si DOSPLUS 3.5 recherche ce fichier, il recalcule le HASH CODE, et en cherche un qui lui soit egal dans la HIT, puis s'il l'a trouve, il en deduit la position de l'etiquette du fichier, qu'il lit et compare a celle qu'il cherche (le meme hashcode peut venir de noms differents). Si les noms ne correspondent pas, la recherche recommence dans la HIT, pour un autre code identique.

La position du HASH CODE dans la HIT est appelee le numero de fichier logique (LFN). De ce numero on deduit le numero de secteur et le rang de l'etiquette du fichier de la facon suivante : soit un LFN de 4BH, en binaire il vaut :

0100 1011

Les bits 0 a 4 indiquent le numero du secteur -2, les bits 5 a 7 indiquent le numero d'emplacement d'etiquette occupe. Dans notre exemple, le NO de secteur serait 13, le numero d'etiquette 2.

Organisation des etiquettes de fichiers

Les etiquettes de fichiers sont rangees dans DIR/SYS, a partir du secteur 2 et jusqu'a la fin du fichier. Chaque secteur contient 8 etiquettes de 32 octets, commençant aux octets 00, 20H, 40H, 60H, 80H, A0H, C0H et E0H. Il existe deux types d'etiquette: la principale (FPDE) et l'auxiliaire ou d'extension (FXDE). La raison d'etre de cette derniere est qu'il peut arriver que l'etiquette principale ne soit pas assez grande pour contenir toutes les etiquettes des extensions.

Octet	Description (FPDE)	Description (FXDE)
Debut+00H Indic		Indic
Bit:	7: 0=FPDE, 1=FXDE	Bit 7: - id
	6: 0=Fic. Utilis. 1=Fich SYS	6: - inutilise
	5: reserve	5: - ""
	4: 0=fich. KILLE, 1=Fic. actif	4: - la meme chose
	3: 0=fich Vis. 1=Inv	3: - inutil.
	Bits 0-2 Niveau de protection 0 a 7	Bits 0-2 inutil.
Debut+01H Indic.		Bit: 7 Pointeur de chainage inverse
Bit	7: 0=compressible 1=Incompress.	
	6: 0=fichier non modif., 1= Modif	
	5: reserve	
	4: reserve	
	Bits 0-3: Mois	
Debut+02H Date		Inutil.
Bits 3-7: Jour		
Bits 0-2: Annee-1980		
Debut+03H Octet fin de fichier		Inutilise
Debut+04H Longueur d'enregistrement logique (0=256)		Inutil.
Debut+05H Nomfich. 8 carac. Justif. a Gauc.		Inutil.

Debut+00H Extension 3 caract just. a G.	Inutil.
Debut+10H Code Mot acces, 2 octets	Inutil.
Debut+12H Code Mise a J, 2 oct.	Inutil.
Debut+14H Numero dernier bloc 2 oct.	Inutil.
Debut+16H Descripteur segments 8 oct.	Idem.
Debut+1EH terminateur desc seg/Lien LFN 2 oct.	Idem.

Debut+03H

Cet octet contient le niveau de protection et plusieurs fanions. Le bit 7 a 1, indique que l'etiquette est une extension. Le FXDE n' utilise pas les octets 02H-15H, mais l'octet 01H contient un numero logique de fichier (LFN) qui pointe vers l'etiquette precedente (FPDE ou FXDE).

Le bit 6 sert a differentier les fichiers utilisateurs et les fichiers systeme. Normalement, seuls les fichiers DOSPLUS 3.5/SYS peuvent l'avoir a 1. Il doit etre a 0 pour tous les autres.

Le bit 4 sert a indiquer si cette etiquette appartient a un fichier valide. s'il est a 0, le fichier a ete elimine par KILL, c'est ce qui permet de le recuperer par RESTORE. Ces etiquettes sont susceptibles d'etre reutilisees.

Le bit 6 signale l'etat de visibilite du fichier.

Les bits 0-2 contiennent le niveau de protection du fichier (0 a 7).

Debut+01H

Dans un FPDE, cet octet contient deux fanions et une partie de la date de la derniere mise a jour du fichier. Le bit 7 reflete l'etat compressible/incompressible du fichier (voir ATTRIB).

Le bit 6 indique si un fichier a ete modifie depuis la derniere mise a 0 de ce fanion (BACKUP, ATTRIB (MOD=N), etc).

Les bits 0-3 contiennent le mois de la derniere mise a jour. Leur valeur va normalement de 1 a 12.

Ainsi qu'il a deja ete indique, cet octet est aussi utilise dans un FXDE comme pointeur LFN inverse, c'est a dire qu'il contiendrait le numero de fichier logique de l'etiquette directory qui a conduit a cet FXDE.

Debut+02H

Cet octet contient le reste de la date de modification: les bit 3-7 contiennent le jour du mois, les bits 0-2 l'annee, commençant en 1980.

Debut+03H

La valeur de cet octet de fin de fichier (EOF) indique combien d'octets du fichier sont ranges dans son dernier bloc. Par exemple, un fichier d'une longueur de bloc de 37, avec 65 blocs est long de 2405 octets. Il remplirait 9 secteurs de 256 octets et partie d'un dixieme secteur. Cet octet nous dit combien d'octets de ce secteur sont utilises (101 dans notre exemple), et l'octet EOF vaudrait 64H (compte -1). 0 represente 256.

Debut+04H

Contient la longueur de bloc logique (LRL) utilise a la creation du fichier. 0 represente 256.

Debut+05H

C' est le debut du nom du fichier, ecrit de gauche a droite et complete avec des blancs a droite.

Debut+0DH

C' est le debut de l' extension (3 caract.) comme ci-dessus.

Debut+10H

Code du mot de passe d' acces (2 octets).

Debut+12H

Code du mot de mise a jour (2 octets).

Debut+14H

Ces 2 octets contiennent le nombre total de secteurs occupes totalement ou partiellement par le fichier.

Debut+16H

Les octets de Debut+16H a Debut+1DH constituent la liste des descripteurs de segments. C' est un groupe de 4 paires d' octets qui decrivent la position de divers segments ou morceaux du fichier. Le premier octet de chaque paire contient le numero du cylindre ou commence le segment, le deuxieme contient deux informations : les bits 5 a 7 indiquent le numero du granule du cylindre ou commence le segment, les autres, le nombre de granules contigus qui composent le segment moins 1 (un compte de 0 indique 1 granule)

Si le premier octet d' une paire est FFH, le fichier n' a plus de segments. Si aucun des quatre descripteurs ne contient ce FFH, les deux octets en Debut+1EH et Debut+1FH doivent contenir soit le FFH de fin, soit un lien avec un FXDE.

Debut+1EH

Le but de ces deux derniers octets est de (a) indiquer la fin de la liste des descripteurs du fichier, ou (b) fournir un pointeur sur une etiquette auxiliaire qui contient une autre liste de descripteurs de segments. Si Debut+1EH contient FFH, il n' y plus de segments a decrire, si c' est FEH, l' octet suivant contient un numero de fichier logique pointant vers un FXDE qui contient d' autre descripteurs de segments.

IV. - Organisation de BOOT/SYS

Tous les disques DOSPLUS 3.5, donnees ou systeme, contiennent un fichier BOOT/SYS, qui occupe les quelques premiers secteurs du disque. Sur les disques donnees, BOOT/SYS contient les informations permettant de trouver le directory. Les disques systemes et les disques durs y ont davantage de donnees, que nous analyserons apres avoir etudie sa raison d'etre.

Dans le cas de disques systeme, la premiere et plus importante responsabilite de BOOT/SYS ou BOOTSTRAP (chargement initial) est de fournir un petit programme qui charge le fichier SYS0-/SYS de DOSPLUS 3.5. A l'initialisation d'une machine, une routine de la ROM lit un secteur du cylindre 0 du lecteur systeme (secteur 0 sur Mod I, 1 sur Mod III) et le charge en memoire en 4200H pour le Mod I, ou 4300H pour le Mod III. Ce secteur doit contenir un code Z-80 executable de 256 octets au plus. Les disques de donnees ne servant pas a l'initialisation ne contiennent pas ce programme.

Apres le chargement de ce secteur, la ROM passe la main au programme charge, qui doit alors executer differentes taches:

- (a) lire le secteur 2 du BOOTSTRAP pour obtenir l'information importante et volatile du DCT sur le disque systeme. Cette information est tres importante pour la lecture de la suite.
- (b) Lire tout programme pilote systeme des secteurs 3-x en memoire, pour execution apres chargement et initialisation de SYS0 (facultatif).
- (c) Localiser et charger SYS0, puis lui passer la main.

Le programme d'initialisation n'est qu'une partie du contenu de BOOT/SYS. L'octet 2 du secteur 0 contient une valeur qui indique le numero du cylindre du directory. Par exemple, pour un disque dont le directory est sur le cylindre 20, on lirait 14H. Sur d'anciennes versions de DOSPLUS, le bit 7 aurait indique la densite. Cette convention n'est plus appliquee, mais la compatibilite avec les vieux disques est maintenue en negligeant ce bit 7.

Le secteur 2 de BOOT/SYS contient les informations suivantes : les octets de 00h a 08h sont la copie de la partie variable de la DCT (table de controle du lecteur), voir la section VII de ce document. Ces renseignements n'existent pas sur les disques de donnees.

L'octet 10H du secteur 2 contient divers fanions. Les bit 4 a 7 sont reserves pour un usage ulterieur. Le bit 3 invite l'operateur a entrer l'heure a l'initialisation (1=oui), le bit 2 fait la meme chose pour la date, et le bit 1 determine si le logo DOSPLUS 3.5 doit ou non s'afficher.

Le bit 0 de 10H indique la presence d'un pilote systeme de rechange. A 1 il indique qu'il faut charger un pilote de disque special en MEV pendant l'initialisation. Le systeme utilise ce bit en liaison avec les octets 11H et 12H, qui designent l'adresse de lancement du pilote precite.

L'octet 13H controle le clignotement du curseur, le 14H le caractere du curseur. Le 15H indique le type de caracteres a afficher apres l'initialisation, le 16H la valeur par defaut de la vitesse des tetes des lecteurs

Les octets 17H et 18H envoient une valeur definie par l'utilisateur a un port defini par le meme (port en 17, valeur en 18). ->

V. - Table et organisation de DCB (bloc de controle de peripherique)

Blocs de controle de peripheriques

Un bloc de controle de peripherique, ou DCB, est une zone de MEV qui contient des donnees de controle de la circulation des donnees vers et depuis des peripheriques orientes caracteres. DOSPLUS 3.5 en contient 6: @KI, @DD, @PR, @RS et les deux utilisateurs U1 et U2.

Les DCB peuvent etre de longueur variable, mais ils partagent une structure commune, decrite ci-dessous :

Adresse	Donne
DCB+00H	fanions type de DCB Bit 7: DCB etendu. 0=normal, 1= etendu Bit 6: Filtre. 0=pas de filtre Bit 5: Join. 0= pas de JOIN Bit 4: Force. 0=pas de Force Bit 3: NIL. 0=actif, 1=NIL Bit 2: Ctl. 0= pas de controle d' E/S Bit 1: type d' entree. 0= pas d' entree
DCB+01H	Adresse pilote sur 2 octets

Le premier octet de tous les DCB contient 8 fanions qui decrivent l' etat present du peripherique. Le bit 7 indique si l' une de trois conditions est remplie: le peripherique est JOINt, il est FORCE, ou inexistant (NIL).

Le bit 6 indique si une table de filtre est installee pour ce peripherique et active. C' est au pilote d' effectuer les conversions correspondantes.

Le bit 5 indique si le peripherique est JOINt, le bit 4 s' il est FORCE. Dans l' un ou l' autre cas, si l' un de ces bits est a 1, le bit 7 doit y etre aussi.

Le bit 3 indique si le peripherique est mis a NIL (rien). Un peripherique NIL ne recoit ni n' emet de caracteres. Si ce bit est a 1, le bit 7 doit y etre aussi, sinon le NIL n' aura pas d' effet.

Les bits 0 a 2 indiquent le type d' E/S que le pilote peut executer. Le bit 2 indique qu' il peut recevoir ou emettre des donnees de CONTROLE, c' est a dire des operations de transfert de donnees qui ne sont pas dans la classification d' entree ou de sortie. Ce sont typiquement des operations de reglage ou de lecture d' etat de peripheriques.

Le bit 1 indique un pilote qui peut recevoir des donnees pour les envoyer a un peripherique, comme par exemple un pilote d' imprimante.

Le bit 0 indique que le pilote peut recevoir des donnees d' un peripherique et les transmettre au programme appelant, comme par ex. un pilote de clavier.

Les deuxieme et troisieme octets d' un DCB contiennent l' adresse du pilote. Pour passer des donnees a un peripherique, ou en recevoir, un programme pointe une paire de registres du Z-80 sur l' adresse du DCB appropriee. Un appel est alors fait a un sous-programme appele 'Entree/Sortie de caractere', ou CIO. Cette routine prepare l' entree ou la sortie avec le peripherique et passe la main a l' adresse du pilote, contenue dans le DCB.

Pendant un JOIN ou un FORCE actifs, le contenu de DCB+01H et DCB+02H est remplace par un pointeur sur le DCB destination (dans le cas de FORCE), ou par l' adresse de la position Debut+04H de la DCBTBL du peripherique destination dans la DCBTBL (dans le cas de JOIN). L' ancienne adresse et le contenu initial de DCB+00H sont conserves dans la table du DCB (DCBTBL) decrite

ci-dessous.

La table du DCB

DOSPLUS 3.5 tient à jour une table qui référence les DCB des six périphériques système orientés caractère. Elle est composée de six paquets de 11 octets. Chacun a la structure suivante:

Octet	Donne
0,1	adresse du DCB
2,3	adresse de la table de filtrage
4,5	adresse du pilote destination
6,7	Adresse du pilote (si JOIN ou FORCE)
8	type de DCB (si JOIN ou FORCE)
9,10	nom pilote (2 caract.)

Les octets 0 & 1 contiennent un pointeur sur le bloc reel de controle. Si il valent FFFFH, le pilote n' a pas de DCB defini.

Les octets 2 & 3 pointent sur toute table de filtrage qui peut resider en MEV pour ce peripherique. Voir la section X de ce manuel 'Ecriture de pilotes pour DOSPLUS 3.5' pour d' autres details.

Les octets 4 & 5 servent pendant un JOIN a indiquer l' adresse du DCB du peripherique JOINt. Par exemple, si le pilote @KI est JOINt au pilote @RS, ces octets dans la DCBTBL de @KI contiendraient un pointeur vers le DCB de @RS.

Les octets 6 & 7 servent pendant JOIN et FORCE a ranger l' adresse du pilote. Cette adresse est deplacee du DCB pendant JOIN ou FORCE quand une adresse Debut+04H de DCBTBL de pilote origine est installee (pour un JOIN), ou quand l' adresse du pilote destination est placee (pour FORCE).

Les deux derniers octets de chaque DCBTBL contiennent le nom a deux caracteres (justifies a gauche et completes d' un blanc le cas echeant) du pilote.

DOSPLUS 3.5 entretient six pilotes orientes caracteres. Deux d' entre eux, @U1 & @U2 sont a definir par l' utilisateur et n' ont pas de zone DCB predefinie dans le systeme. @RS n' a pas d' adresse predefinie dans ls SED. Les trois pilotes restant ont tous leurs adresses reservees dans la MEV systeme. L' utilisateur peut deplacer et redefinir les DCB. L' information presentee ci-dessous se refere aux adresses standards de ces DCB et a la signification de leur contenu.

Nom du pilote: @KI
Adresse du DCB 4015H

octet	Signification
DCB+00H	Type de DCB (1 octet)
DCB+01H	Adresse pilote (2 oct)
DCB+03H	reserve (1 oct)
DCB+04H	indicateur blocage majuscules, 0= pas de maj (1 oct).
DCB+05H	Compteur de clignotement curseur (1 oct)
DCB+06H	Etat clignotement curseur (1 oct), 0= non,
DCB+07H	Interrupt clignot. curs. 0= oui (1 oct)

Nom du pilote: @DO

Adresse du DCB 401DH

octet Signification

DCB+00H Type de DCB (1 octet)
DCB+01H Adresse pilote (2 oct)
DCB+03H position curseur (2 octets)
DCB+05H Allumage curseur 0=eteint (1 oct)
DCB+06H Caract curseur (1 oct)
DCB+07H Selection carct/tabul 0= tab. (1 oct), Mod III seulement

Nom du pilote: @PR
Adresse du DCB 4025H

octet Signification

DCB+00H Type de DCB (1 octet)
DCB+01H Adresse pilote (2 oct)
DCB+03H Lignes par page (1 oct)
DCB+04H Lignes imprimees (1 oct)
DCB+05H Nombre de caracteres (1 oct)
DCB+06H Largeur maxi (1 oct)
DCB+07H Longueur page phys en lignes (1 oct)

En supplement, il existe sur la version Modele-III de DOSPLUS 3.5 un DCB d'initialisation du RS232, qui est arrange ainsi:

Adresse du DCB 41F5H

octet Signification

DCB+00H Type de DCB (1 octet)
DCB+01H Adresse pilote (2 oct)
DCB+03H Code vitesse en bauds - 4 bits pds fort= vit emiss, 4 bit pds faible=vit rec;
DCB+04H Code configuration de l' UART (1 octet)
DCB+05H Select Attente 0= pas d'attente (1 octet)

VI. - Structure des FCB (File Control Block = bloc de controle des fichiers)

L' une des plus importantes fonctions d' un systeme d' exploitation de disques est la manipulation de fichiers de donnees sur disques. Sous DOSPLUS 3.5 on accede aux fichiers par les diverses routines de manipulation de fichiers, decrites dans la section VIII de ce manuel. Chaque fois qu' une routine systeme execute un travail sur un fichier, il s' y refere par une partie de MEV appelee bloc de controle de fichier (FCB). De ce point de vue, un FCB fonctionne comme un DCB; il sert a controler le flux de donnees avec le fichier. En fait, DCB et FCB peuvent etre echanges pour les fonctions d' E/S de caracteres (voir GET@ et PUT@, section VIII).

Tous les FCB ont une structure commune. Avant qu' un fichier soit ouvert ou initialise pour l' E/S, le FCB est une zone memoire de 32 octets qui devrait contenir le nom du fichier, l' extension s' il y a lieu, le mot de passe (s' il ..), et la designation de lecteur (s' ..). pour designer le fichier. Cette designation devrait etre terminee par un caractere retour/chariot (0DH) ou ETX (03H). Apres OPEN ou INIT, le contenu du FCB est le suivant:

octet Signification

FCB+00H type de FCB, 80H (1 octet)

FCB+01H	fanions (1 octet) bit 7: (blocks de longueur variable) bit 6: Acces aleatoire bit 5: tampon = NRN bit 4: Tampon mis a jour bit 3: reserve bit 0-2: code d' acces
FCB+02H	Fanions (1 oct) bit 7: fichier non compressible bit 6: fanion modif bit 0-5: reserves
FCB+03H	Adresse du tampon d' E/S du fichier (2 oct)
FCB+05H	ecart du prochain bloc (1 oct)
FCB+06H	NO pilote (1 oct)
FCB+07H	NO logique de fichier (1 oct)
FCB+08H	octet fin de fichier (1 oct)
FCB+09H	Longueur de bloc (1 oct)
FCB+0AH	Numero du prochain bloc (2 oct)
FCB+0CH	Numero du dernier bloc (2 oct)
FCB+0EH	liste du descripteur de segment (17 oct)

L'octet place en FCB+00H est appele l' octet de type de FCB, et sert a distinguer un FCB d' un DCB. Pour un FCB, il vaut toujours 80H. /Les octets de type de DCB ne peuvent jamais avoir cette valeur.

FCB+01H contient plusieurs fanions indiquant l' etat du fichier. Le bit 7 indique des 'blocs variables', c'est-a-dire qu' un bloc partage un secteur avec d'autres blocs (ndt: nous traduisons par 'bloc' le 'logical record'). Quand ce bit 7 vaut 1, cela signifie que le SED effectue automatiquement l' assemblage de plusieurs blocs dans un secteur, et l' operation inverse, le desassemblage. Il arrive que des programmes utilisateur forcent ce bit a 0 pour obliger le systeme a ecrire un enregistrement physique entier sur le disque en travaillant avec des enregistrements de moins de 256 octets.

Le bit 6 indique le mode d' acces a un fichier. Quand on fait OPEN ou INIT pour la premiere fois sur un fichier, ce fanion est a 0, indiquant acces sequentiel. Il est mis a 1 quand une operation aleatoire est faite dessus (routine @POSN). Si on ecrit sur un fichier et que ce fanion est a 0, le systeme va raccourcir ce fichier pour l' ajuster au dernier octet ecrit.

Si le tampon d' E/S de ce fichier, au moment d' un OPEN ou d' un INIT contient le bloc suivant du fichier, le bit 5 sera a 1. Dans le cas contraire, il sera a 0 pour indiquer que le systeme doit effectuer un READ (lecture) pour acceder a ce bloc.

Le bit 4 indique que le contenu du tampon d' E/S a ete mis a jour ou modifie depuis le dernier READ ou WRITE.

Les bits 0-2 contiennent le code d' acces sous lequel le fichier a ete ouvert. Par exemple imaginez qu' un fichier appele TEST/CMD, qui a "PW" pour mot de passe de mise a jour et "TEST" pour mot d' acces, un niveau de protection de 5. Si ce fichier est ouvert comme "TEST/CMD.PW", le code d' acces dans le FCB contiendra 0, indiquant acces libre. Si le fichier est ouvert comme "TEST/CMD.TEST", le code d' acces contiendra 5, traduisant le niveau d' acces autorise par le mot d' acces.

FCB+01H contient deux autres fanions importants: le bit 7 indique au systeme de ne pas raccourcir le fichier au moment de CLOSE, meme si la longueur du fichier a diminue. Le bit 6 est a 1 si on a modifie le fichier (en y ecrivant), il reste a 0 dans l' autre cas.

FCB+03 et FCB+04H pointent sur un tampon de 256 octets fixe au moment de OPEN ou INIT, toute donnée transitant entre la machine et le fichier doit y passer.

FCB+005H est un octet qui contient l'ecart entre le prochain bloc et le secteur, si les deux commencent au même endroit, il sera nul.

FCB+06H contient le numéro du lecteur où réside le fichier.

FCB+07H est un octet égal au numéro logique de fichier (LFN) de l'étiquette principale du fichier dans le directory.

FCB+08H contient l'octet de fin de fichier (EOF), qui indique combien d'octets le fichier occupe dans le dernier secteur, 0 veut dire que tout le secteur est occupé.

FCB+09H indique la longueur de bloc, 0 équivaut à 256 octets.

FCB+0AH et FCB+0BH contiennent le numéro du prochain secteur (NRN).

FCB+0CH et FCB+0DH contiennent le nombre de secteurs PLEINS du fichier. Les secteurs partiellement remplis n'entrent pas dans ce compte.

Les octets FCB+0EH à FCB+1FH contiennent une liste de description des segments utilisée pour retrouver les segments de fichier sur le disque.

VII - Organisation des DCT

DOSPLUS 3.5 entretient huit tables de contrôle des lecteurs (DCT). Chacune d'elles contient toute l'information relative à un lecteur. Ces DCT pouvant être logés n'importe où en MEV, DOSPLUS 3.5 fournit une table, appelée DCTTBL qui contient les adresses de tous les DCT. DCTTBL consyste en 8 paquets de 4 octets arrangés comme suit :

octet	Signification
Debut+00H	Pointeur sur DCT (2 oct)
Debut+02H	Nom du lecteur (2 oct)

Les deux premiers octets pointent sur la position réelle du DCT. S'ils valent FFFFH, ce lecteur n'a pas de DCT affecté.

Les deux autres octets contiennent le nom du lecteur à deux caractères. Si ce nom n'en a qu'un, il est complété par un blanc.

Le DCT proprement dit est une zone de MEV de 20 octets qui contient toute l'information nécessaire au système et à ses pilotes pour utiliser le lecteur. Ce qui suit détaille la structure des DCT utilisés par les pilotes de DOSPLUS 3.5. L'utilisateur individuel peut écrire d'autres DCT avec la structure de son choix, mais nous suggérons l'emploi de cet arrangement standard, car l'ordre CONFIG de la librairie suppose que la structure ci-dessous est utilisée :

Information permanente du DCT

Octet	Signification
DCT+00H	type du DCT (1 oct)
DCT+01H	adresse pilote (2 oct)
DCT+03H	indicateurs (1 oct)
	bit 7: select 5''/8'', 0=5''
	bit 6: Prot écrit. 0=pas de prot.
	bit 5: select souple/dur. 0=souple
	bit 4: délai moteur. 0=pas de délai

bit 3: delai tete. 0=pas de delai
 bit 2: Saut (skip). 0= pas de saut
 bit 1: select fixe/amovib. 0=fixe
 bit 0: initial disque. 0=non

DCT+04H code vitesse tete (1 oct)
 DCT+05H ecart tete (1 oct)
 DCT+06H ecart cylindres. (2 oct)
 DCT+08H ecart secteur. (1 oct)
 DCT+09H position tete (1 oct)
 DCT+0AH n0 physique lect (1 oct)

Information perissable du DCT

DCT+0BH fanions (1 oct)
 bit 7: select simpl/double densite. 0=simple
 bit 6: protection directory. 0=pas de prot.
 bits 0-5 reserves

DCT+0CH Nombre de surfaces (1 oct)
 DCT+0DH Secteurs par piste (1oct)
 DCT+0EH Longueur directory (1 oct)
 DCT+0FH Sect/gran. (1 oct)
 DCT+10H gran/cylindre (1 oct)
 DCT+11H secteur/cylindre. (1 oct)
 DCT+12H place du directory. (1 oct)
 DCT+13H Nombre de cylindres (1 oct)

Comme le montre la table, le DCT est partage en deux parties, appelees perissable et non perissable. Les donnees non perissables comprennent l'adresse du pilote et la plupart des caracteristiques physiques du lecteur. Les donnees perissables contiennent l'information propre a un disque particulier, comme la place du directory, la densite, le nombre de faces, etc. Cette information perissable est sujette a modification automatique a l'initialisation d'un disque.

Donnees non perissables

DCT+00 est appele l'octet de type de DCT. C'est analogue aux octets de type des DCB et FCB deja vus, et sert aussi a identifier un DCT et a le distinguer d'un FCB ou d'un DCB. Sa valeur normale pour un lecteur actif est 40H. Si le lecteur est NIL, le bit 3 sera aussi a 1, donnant une valeur de 48H.

DCT+01H et DCT+02H contiennent l'adresse du programme pilote du lecteur.

DCT+03H contient 8 fanions d'etat du lecteur. Le bit 7 indique si le lecteur est un 5 ou 8 pouces (0=5).

Le bit 6 sert d'indicateur de protection ecriture. Il est mis a 1 par l'option WP de CONFIG. C'est au pilote de l'honorer.

Le bit 5 indique si le DCT decrit un disque dur ou souple (1=dur).

Les bit 4 et 3 indiquent s'il y a un delai pour le moteur et la tete (dans l'ordre), (1=oui).

Quand le bit 2 est a 1, le pilote sait qu'il doit faire sauter a la tete une piste sur deux pour chercher un cylindre quelconque. Ceci permet a DOSPLUS 3.5 de lire un disque 40 pistes dans un lecteur 80 pistes.

Le bit 1 a 1 indique au pilote que le disque dur est amovible, sinon il est fixe.

Le bit 0 a 1 indique au pilote de relire l'information du disque au prochain acces lect-/ecrit. (log).

DCT+04H contient le code de vitesse tete du lecteur.

DCT+05H contient un octet qui indique le ecart de tete pour le partage en volumes fixe par CONFIG.

DCT+06H et DCT+07H contiennent le ecart de cylindres pour le partage en volumes fixe par CONFIG.

DCT+08 contient un ecart de secteur d' un octet. C' est simplement le numero du premier secteur d' une piste.

DCT+09H contient sur 1 octet le numero de cylindre sur lequel se trouve la tete (ou les tetes).

DCT+0AH est le numreo physique du lecteur.

Donnees perissables.

DCT+0bH contient des indicateurs de la nature du disque en place dans le lecteur. Le bit 7 indique la densite d' ecriture (1= DDEN).

Le bit 6 indique si le directory du disque est protege, c' est-a-dire ecrit avec une marque d' adresse speciale. C' est generalement le cas des disques souples, et rarement le cas des disques durs. Ce bit informe donc le pilote du type de marqueurs d' adresse qu' il doit attendre en lisant un disque.

DCT+0CH est un octet qui indique le nombre de surfaces du disque.

DCT+0DH est un octet qui contient le nombre de secteurs par piste du disque.

DCT+0EH (1 octet) indique la longueur du directory.

DCT+0FH est la taille des granules, en secteurs/granule.

DCT+10 est le nombre de granules par cylindre.

DCT+11H est le nombre de secteurs par cylindre.

DCT+12H contient le ecart entre le premier cylindre du volume (DCT+06H et DCT+07H) et le cylindre du directory.

DCT+13H contient le nombre total de cylindres du volume.

VIII - Point d'entree dans le systeme

DOSPLUS 3.5 fournit a l'utilisateur de nombreuses fonctions systeme utilisables avantageusement dans des programmes d'application. Elles vont etre decrites dans les pages suivantes.

Nom de la routine	Adresse		Numero de Page
	Mod I	Mod III	
ABORT@	4030H	4030H	T/24
BKSP@	4445H	4445H	T/22
CAT@	445AH	4419H	T/26
CKEOF@	4451H	4457H	T/23
CLOSE@	4428H	4428H	T/20
CMD@	4400H	4400H	T/25
CMNDI@	4405H	4405H	T/25
DEBUG@	440DH	440DH	T/25
DISKIO@	4485H	4488H	T/34
DVID@	444EH	4451H	T/28
DSF@	0033H	0033H	T/37
DSPLY@	4467H	4467H	T/29
ERROR@	4409H	4409H	T/25
EVAL@	4479H	4479H	T/30
EXIT@	4020H	4020H	T/24
FEXT@	4473H	444BH	T/23
FILPTR@	4454H	428DH	T/24
FSPEC@	441CH	441CH	T/19
GET@	0013H	0013H	T/37
GTDAT@	4470H	4470H	T/30
GTTIME@	446DH	446DH	T/30
INIT@	4420H	4420H	T/19
KBD@	002BH	002BH	T/37
KEY@	0049H	0049H	T/38
KEYIN@	0040H	0040H	T/38
KILL@	442CH	442CH	T/20
LOAD@	4430H	4430H	T/20
LOCDCB%	4488H	44A0H	T/36
LOCDCI%	448BH	44A3H	T/36
LOCDEV@	447FH	447FH	T/32
MULT@	444BH	444EH	T/28
OPEN@	4424H	4424H	T/19
PARAM@	4476H	4454H	T/28
PEOF@	444BH	444BH	T/23
POSN@	4442H	4442H	T/22
PRINT@	446AH	446AH	T/29
PRT@	003BH	003BH	T/38
PUT@	001BH	001BH	T/37
RANDIR@	4457H	4290H	T/24
READ@	4436H	4436H	T/21
RES@	4413H	4416H	T/35
REW@	443FH	443FH	T/22
RUN@	4433H	4433H	T/21
SET@	4410H	4413H	T/27
SORT@	4482H	4482H	T/33
VER@	443CH	443CH	T/21
WILD@	447CH	447CH	T/32
WRITE@	4439H	4439H	T/21

Routines de manipulation de fichiers

Toutes les routines de manipulation de fichiers demandent que la paire de registres DE pointe sur un DCB ou un FCB valide de D>DOSPLUS 3.5 (voir les sections V et VI de ce manuel). Notez que beaucoup de routines de manipulation de fichiers (FSPEC@, OPEN@, CLOSE@, KILL@) sont utiles pour les pilotes a caracteres et ceux a fichiers.

FSPEC@

A>Adresse: 441CH (Mod I & III)

FSPEC@ sert a deplacer une designation de peripherique ou de fichier d' une zone MEV (habituellement le tampon d' ordres) vers un FCB ou un DCB. FSPEC@ deplace la designation caractere par caractere jusqu' a ce qu' il rencontre soit un terminateur (" ", ",", ":", retour chariot, etc), soit un caractere invalide. Dans ce dernier cas FSPEC@ termine avec une erreur. En tete, il ignore les blancs, les virgules ou les mot delimitateurs FROM, TO & USING. Notez que FSPEC@ ne renvoie pas un code, mais simplement une indication d' erreur.

ENTREE: DE=> FCB ou DCB destinataire.
HL=> zone memoire contenant la designation.

SORTIE: EC altere.
DE=> designation du fichier/peripherique, terminee par ETX (ASCII 03)
HL=> zone suivante du tampon
Indicateurs: Z=pas d' erreur, NZ= erreur
Si NZ A= caractere errone.

INIT@

Adresse: 4420H (Mod I & III)

INIT@ sert a ouvrir les fichiers et peripheriques pour l' E/S et peut aussi creer de nouveaux fichiers sur un disque. Avant d' appeler INIT@ le registre DE doit pointer sur un DCB ou FCB de 32 octets contenant le nom du fichier ou du peripherique a ouvrir, ou creer et ouvrir si c' est un nouveau fichier. Si un fichier subit INIT@ la paire HL doit pointer sur un tapon d' E/S de 256 octets, et le registre B contenir la longueur de blocs (LRL) d' ouverture du fichier (0 pour 256). Au retour d' INIT@ le registre F contien l' etat d' erreur, et A le code d' erreur eventuel.

ENTREE: B = longueur de bloc
DE=> FCB ou DCB
HL=> Tampon E/S de 256 octets.

SORTIE: A code d' erreur
Indicateurs: Z=pas d' erreur, NZ= erreur
C= fichier cree.

Z>OPEN@

Adresse: 4424H (Mod I & III)

OPEN@ fonctionne d' une maniere semblable a INIT@, a l' exception qu' OPEN@ ne cree pas de nouveau fichier. Si le fichier a ouvrir n' existe pas, OPEN@ revient avec une erreur 'File not found'.

ENTREE: B = longueur de bloc.
 DE=> FCB ou DCB
 HL=> tampon de 256 octets

SORTIE: A = code d'erreur.
 Indicateurs: Z=pas d' erreur, NZ= erreur

CLOSE@
 Adresse: 4428H (Mod I & III)

Cette routine sert a terminer toute E/S sur un fichier/peripherique. Quand CLOSE@ est appele, toutes les donnees restant dans le tampon d' E/S du disque sont copiees sur le fichier, et la longueur de celui-ci est mise a jour si necessaire. CLOSE@ devrait toujours etre utilise apres une ecriture sur un fichier, mais n' est pas utile en cas de lecture exclusive.

Apres un appel de CLOSE@, le FCB ou le DCB contiendra encore le nom du fichier ou du peripherique, et dans le premier cas, le numero du lecteur ou le fichier reside. Le mot de passe n' est pas restaure dans le FCB.

ENTREE: DE=> FCB ou DCB ouvert

SORTIE: A = code d'erreur.
 Indicateurs: Z=pas d' erreur, NZ= erreur

KILL@
 Adresse: 442CH (Mod I & III)

Cette routine sert a eliminer un fichier d' un disque, liberant l' espace qu' il occupe. L' etiquette de ce fichier n' est pas otee du directory par KILL@, et ce fichier peut donc etre recupere plus tard avec RESTORE. Notez que KILL@ doit etre utilise avec un FCB OUVERT, et non ferme. Apres execution de KILL@, le contenu du FCB sera detruit. KILL@ ne peut etre utilise pour eliminer des peripheriques systeme.

ENTREE: DE=> FCB ouvert

SORTIE: A = code d'erreur.
 Indicateurs: Z=pas d' erreur, NZ= erreur

LOAD@
 Adresse: 4430H (Mod I & III)

Cette routine sert a charger un fichier objet en MEV. Le fichier doit etre copie sur disque au format 'fichier a charger'. A l' entree dans LOAD@, DE doit pointer sur un FCB contenant le nom du fichier a charger. A la sortie de LOAD@, HL contiendra l' adresse de transfert du fichier a charger.

ENTREE: DE=> FCB non ouvert

SORTIE: A = code d'erreur.
 HL= Adresse de lancement du programme
 Indicateurs: Z=pas d' erreur, NZ= erreur

RUN@

Adresse: 4433H (Mod I & III)

RUN@ travaille comme LOAD@, mais apres le chargement du fichier en memoire, il en lance l'execution a l'adresse de transfert.

ENTREE: DE=> FCB non ouvert

SORTIE: En cas de succes, Le controle est passe au programme juste charge.
Sinon la routine retourne au SED via ABORTE@.

READ@

Adresse: 4436H (Mod I & III)

Cette routine sert a lire un bloc d'un fichier. DE pointe sur le bloc ouvert de controle du fichier, et HL vers un tampon utilisateur. Ce tampon n'est pas necessaire si le fichier est ouvert avec un LRL de 256. Dans ce cas le bloc est mis dans le tampon d'E/S defini par le programme utilisateur a l'ouverture. Si LRL est inferieur a 256, l'utilisateur doit designer un tampon dont la longueur soit egale a LRL pour contenir le bloc lu sur le disque.

ENTREE: DE=> FCB ouvert

Si LRL < 256

HL=> tampon utilisateur

Si LRL=256

HL est neglige, les donnees sont mises dans le tampon d'E/S

SORTIE: A = code d'erreur.

Indicateurs: Z=pas d'erreur, NZ= erreur

WRITE@

Adresse: 4439H (Mod I & III)

Sert a ecrire un seul bloc dans un fichier. Si OPEN@ ou INIT@ a ete fait sur le fichier avec un LRL de 256, les donnees du tampon d'E/S, designe par la meme routine, sont ecrites sur le fichier. si le LRL est <256, HL pointe sur un tampon utilisateur qui contient les donnees a ecrire sur le fichier. Comme d'habitude, DE pointe sur le FCB ouvert.

ENTREE: DE=> FCB ouvert

si LRL=256

HL=> tampon de 256 octets

si LRL < 256

HL est neglige.

SORTIE: A = code d'erreur.

Indicateurs: Z=pas d'erreur, NZ= erreur

VER@

Adresse: 443CH (Mod I & III)

La routine VER@ fonctionne exactement comme WRITE@ ci-dessus. De plus, VER@ relit l'enregistrement apres l'écriture pour verifier que l'écriture est reussie. Si l'enregistrement ne peut etre lu correctement, une erreur est signalee.

ENTREE: DE=> FCB ouvert
 si LRL=256
 HL=> tampon de 256 octets
 si LRL<256
 HL est neglige.

SORTIE: A = code d'erreur.
 Indicateurs: Z=pas d' erreur, NZ= erreur

REW@
 Adresse: 443fH (Mod I & III)

REW@ sert a diriger le pointeur du prochain bloc du fichier sur le premier bloc. Avant d'ajuster le contenu du FCB, REW@ ecrit toutes donnees restant dans le tampon d'E/S sur le disque.

ENTREE: DE=> FCB ouvert

SORTIE: A = code d'erreur.
 Indicateurs: Z=pas d' erreur, NZ= erreur

POS@
 Adresse: 4442H (Mod I & III)

Cette routine permet a l'utilisateur de diriger le pointeur du prochain bloc sur tout bloc desire. Bien sur, POS@ ecrira toute donnee restant dans le tampon des blocs du fichier avant d'aller vers un nouveau bloc. Notez que cette routine est le seul moyen de choisir le mode d'accès aleatoire pour un fichier.

ENTREE: BC = Numero de bloc.
 DE=> FCB ouvert.

SORTIE: A = code d'erreur.
 Indicateurs: Z=pas d' erreur, NZ= erreur

BKSP@
 Adresse: 4445H (Mod I & III)

BKSP@ sert a diriger le pointeur du prochain bloc d'un fichier sur le bloc precedent, apres avoir ecrit toute donnee restant dans le tampon du fichier. Par exemple, si le numero de bloc courant est 89, BKSP@ fera pointer le bloc 88. Notez que BKSP@ ne lit pas le bloc, mais ajuste simplement l'information du FCB pour que le prochain bloc lu ou ecrit soit le precedent (88 dans notre exemple). Si l'on tente BKSP@ a partir du bloc 0, le nouveau numero de fichier sera FFFFH, ou 65535, et tres probablement, une erreur de fin de fichier sera signalee.

ENTREE: DE=> FCB ouvert

SORTIE: A = code d'erreur.

Indicateurs: Z=pas d' erreur, NZ= erreur

PEOF@

Adresse: 4448H (Mod I & III)

PEOF@ dirige le pointeur du prochain bloc sur le bloc suivant le dernier bloc du fichier. Cela sert si on le desire, a allonger un fichier sans le lire entierement en MEV. Notez que PEOF@ renvoie NORMALEMENT le code d' erreur 1CH (fin de fichier). Il ne faut pas tenir compte de cette erreur, mais les autres doivent etre traitees normalement. Comme toutes les autres routines de positionnement, PEOF@ enregistre toute donnee residuelle avant d' aller vers un autre bloc.

ENTREE: DE=> FCB ouvert

SORTIE: A = code d'erreur,
Indicateurs: Z=pas d' erreur, NZ= erreur

FEXT@

Adresse: 4473H (Mod I) & 444BH (Mod III)

FEXT@ ajoute une extension a un nom de fichier. A l' entree, DE doit pointer sur un FCB non ouvert contenant un nom de fichier valide pour DOSPLUS (extension, mot de passe et designation de lecteur sont facultatifs), et HL doit pointer sur une extension de 3 octets. Si celle-ci est plus courte, elle doit etre terminee par un ASCII 03 (ETX) ou 0DH (RC). A l' execution, FEXT@ ajoutera l' extension au nom de fichier, si celui-ci n' en comporte pas, sinon ne fera rien.

Notez que l' adresse de cette routine est differente pour les versions de DOSPLUS pour les Modeles I & III. Le point d' entree du Modele I est bon pour les deux modeles, mais l' inverse est FAUX.

ENTREE: DE=> FCB non ouvert
HL=> tampon de l' extension

SORTIE: A = code d'erreur,
BC est altere
Indicateurs: Z=pas d' erreur, NZ= erreur

CKEOF@

Adresse: 4451H (Mod I) & 4457H (Mod III)

CKEOF@ fournit un moyen de verifier la fin du fichier. Si c' est le cas, CKEOF@ retourne un fanion dans le registre des fanions qui indique que le bloc actuel est le dernier du fichier.

Notez que si CKEOF@ voit que le fichier est a sa fin, il retournera un code d' erreur dans le registre A: 1CH ou 1DH, qui tous deux indiquent que l' on est au bout du fichier. Notez egalement que l' adresse est differente sur les Modeles I & III.

ENTREE: DE=> FCB ouvert

SORTIE: A = code d'erreur,
Indicateurs: Z=pas d' erreur, NZ= erreur

Routines de controle du systeme.

Ce groupe de points d'entree dans le systeme permet a l'utilisateur de rentrer d'un programme en cours a la ligne de commande du SED, de passer des ordres au systeme, afficher de messages d'erreur, entrer le moniteur DEBUG, et afficher le directory d'un disque.

EXIT@

Adresse 4020H (Modeles I & III)

Un saut a cette routine renvoie la machine au niveau commande de DOSPLUS. Notez que l'utilisateur n'a pas besoin de preserver la pile, car le systeme la reinitialise a l'entree de EXIT@.

ABORT@

Adresse 4030H (Modeles I & III)

fonctionne d'une maniere analogue a EXIT@, mais si ABORT@ est utilise pour sortir d'un programme sur une erreur DOSPLUS, ABORT@ appellera DEBUG si il est actif. S'il n'y a pas d'erreur, ou si DEBUG est inerte, ABORT@ aura le meme effet que EXIT@. Les programmes utilisateurs ne devraient sortir par ce vecteur qu'en cas de fin anormale de leur programme.

FILPTR@

Adresse 4020H (Modele I) & 4280H (Modele III)

FILPTR@ sert a obtenir deux informations sur un fichier: Le lecteur sur lequel il reside et le numero d'logique de fichier de son etiquette du directory.

L'adresse de cette routine n'est pas la meme sur le Modele I et sur le Modele III.

Entree: DE=> FCB ouvert

Sortie: A= Code d'erreur
 B= Numero du pilote lecteur (0 a 7)
 C= numero logique de fichier
 Indicateurs: Z= pas d'erreur, NZ= pas d'erreur.

RAMDIR@

Adresse 4457H (Modele I) & 4290H (Modele III)

La routine RAMDIR@ permet a un programme utilisateur d'examiner le contenu d'une ou plusieurs etiquettes du directory, ou de determiner l'occupation de l'espace d'un disque.

L'adresse de RAMDIR@ est differente sur DOSPLUS Modele I et Modele III.

Entree: B= Numero de lecteur (0 a 7)
 C= selecteur de fonction (0, 1-254,255)
 HL=> tampon MEV

Si C=0 Toutes les etiquettes principales seront lues dans le tampon pointe par HL. Il

doit avoir une taille suffisante pour contenir toutes les étiquettes principales actives sur le disque. Sa taille maximum peut être calculée avec la formule:

$$\text{Volutamp} = ((\text{dirsiz}-2) \times 8 \times 22) + 1$$

ou Volutamp est la taille du directory d'un disque en secteurs.

Chaque étiquette du directory est écrite dans le tampon suivant le format :

Debut+00H: 15 caractères pour nom, extension et désignation de lecteur. La chaîne est justifiée à gauche, et complétée avec des blancs à droite.

Debut+0FH: Niveau de protection (0-6)

Debut+10H: octet de fin de fichier

Debut+11H: Longueur de bloc

Debut+12H: numéro du dernier bloc (2 octets, LSB, MSB)

debut+13H: inutilisé

Si C=1 à 254: Une seule étiquette du directory est lue dans un tampon de 22 octets pointé par HL. La structure est la même que ci-dessus.

Si C=255: L'information sur l'espace libre (4 octets) est rangée dans un tampon pointé par HL. Les deux premiers octets représentent le nombre de Koctets alloués aux fichiers sur le disque, les deux suivants sont le nombre de Koctets libres sur le disque. Les deux valeurs sont rangées dans le format LSB, MSB.

Sortie: A= code d'erreur
Indicateurs: Z=pas d'erreur, NZ=erreur.

CMD@

Adresse 4400H (Modele I & III)

Un saut à CMD@ provoque une sortie vers DOSPLUS, comme EXIT@.

CMNDI@

Adresse 4405H (Modele I & III)

Cette routine est l'interpréteur d'ordres de DOSPLUS. L'utilisateur peut envoyer un ordre valide (pointé par HL) à CMNDI@, et DOSPLUS l'exécutera, puis retournera au programme appelant. En cas d'erreur, DOSPLUS retournera au niveau de commande via ABORT@.

Entree: HL=> tampon d'ordre, termine par ASCII 0DH.

ERROR@

Adresse 4409H (Modele I & III)

Cette routine affiche les messages d'erreur de DOSPLUS. Le numéro de l'erreur à afficher est passé à ERROR@ dans les bits 0-5 du registre A. Le bit 6 contrôle l'affichage d'un message d'erreur accentué (encadré de deux paires d'étoiles * *). Le message normal est affiché si le bit 6 est à 1. Le bit 7 détermine si la routine ERROR@ retourne au programme appelant (1), ou va au niveau commande du SED via ABORT@.

Entree: A= code erreur

DEBUG@

Adresse 440DH (Modele I & III)

Lance le moniteur DEBUG, sans tenir compte de son etat actif ou inerte.

CAT@

Adresse 445AH (Modele I) & 4419H (Modele III)

Affiche le directory d' un disque. Le numero du lecteur dont on veut afficher le directory peut etre fourni de deux facons: (a) Un numero de pilote code en ASCII "0" a "7" peut etre range a l' adresse 442BH (Mod I) ou 4271H (Mod III), ou (b) le numero en binaire peut etre charge dans le registre C. La methode (a) est compatible avec TRSDOS, la methode (b) est plus naturelle.

Entree: C= numero binaire du pilote (0-7)
 OU
 (4242B)= valeur ASCII du numero du pilote du disque, "0"- "7" (Mod I)
 OU
 (4271B)= valeur ASCII du numero du pilote du disque, "0"- "7" (Mod III)

Routines de chainage des interruptions

DOSPLUS 3.5 fournit deux routines utilisables pour ajouter et supprimer des routines utilisateur d' interruption dans la chaine d' interruptions. La chaine est une serie de pointeurs sur 2 octets entretenue par DOSPLUS. Chaque pointeur de la chaine est appele une 'case d' interruption'. Ces pointeurs servent a indiquer des 'blocs de travail', qui consistent en un pointeur vers la routine de traitement proprement dite et toutes informations necessitees par la routine d' interruption.

A l' entree dans la routine des interruptions, le registre IX pointera vers le premier octet du bloc de travail. Par exemple, examinons la routine ci-dessous:

```

LD      A,0      ;CASE D' INTERRUPTION #0, 133.33 MS
LD      DE,INTTSK ;BLOC DE TRAVAIL
CALL    SET@     ;INSERE LE TRAVAIL DANS LA CHAINE
;
;
INTTSK  DEFW      TRAV      ;POINTEUR SUR BLOC TRAVAIL
COMPT  DEFW      0         ;MOT COMPTEUR
;
TRAV    LD        L,(IX+2)   ;LIT LE LSB (=poids faible) DU COMPTEUR
LD      H,(IX+3)   ;LIT LE MSB (poids fort)
INC     HL         ;INCREMENTE LE COMPTE
LD      DE,2250    ;LE COMPTE EST A 2250 ?
PUSH    HL         ;SAUVE LE COMPTE
OR      A          ;VIDE LE CARRY
SBC     HL,DE      ;COMPARE HL & DE
POP     HL         ;RESTAURE LE COMPTE
JR      NZ,TRAV1    ;COMPTE < 2250
CALL    SIFFLE     ;FAIT UN BRUIT
LD      HL,0       ;RECOMMENCE A COMPTE
TRAV1   LD        (IX+2),L   ;RANGE LE COMPTEUR (LSB)
LD      (IX+3),H    ;RANGE LE MSB

```

RET

C'est une routine futile qui fera un bruit (genere par la routine imaginaire SIFFLE) toutes les cinq minutes sur Modele III (2250*133,33 ms). Remarquez comment le travail sur l'interruption peut beneficier du fait que IX pointe sur le bloc de travail. Il est habituel de placer toutes les variables utilisees par le traitement de l'interruption dans le bloc de travail, pour simplifier le travail de traitement.

DOSPLUS fournit en realite deux chaines d'interruptions: une rapide et une lente. Sur TRS-80, la pendule (horloge en temps reel ou RTC), genere des interruptions plusieurs fois par seconde (40 sur Modele I & 30 sur Modele III). La chaine rapide d'interruptions est parcourue toutes les 25 millisecondes sur Modele I, et toutes les 33,33 ms sur Modele III. Il y a deux cases d'interruptions rapides sur Modele I (numeros 4 & 5), et 4 sur Modele III (numeros 4-7).

La chaine d'interruptions lentes est parcourue toutes les 100 ms sur Modele I, et toutes les 133,33 sur Modele III. Les deux modeles ont 4 cases d'interruption lentes, numerotees de 0 a 3.

La table ci-dessous contient des informations sur l'utilisation des cases d'interruption par DOSPLUS:

Case NO Modele I		Modele III	
0	LIBRE	100ms	LIBRE 133,33ms
1	clignotement curseur de @DD	100ms	LIBRE 133,33
2	Affichage heure	100ms	LIBRE 133,33
3	mise a jour de 1' heure	100ms	LIBRE 133,33ms
4	LIBRE	100ms	LIBRE 133,33ms
5	SPOOLER de @PR	25ms	LIBRE 33,33ms
6	S/O		SPOOLER de PR 33,33ms
7	S/O		TRAVAIL DE LA ROM 33,33ms

SET@

Adresse 4410H (Modele I) & 4413H (Modele III)

SET@ insere un pointeur dans la chaine des interruptions de DOSPLUS. Ce pointeur doit pointer sur une case d'interruption, qui a son tour doit pointer sur la routine de traitement proprement dite. Avant l'entree dans cette derniere, tous les registres ont ete sauves sur la pile (Note: sous DOSPLUS Modele I, le registre IX n'est pas sauve sur la pile), et le programme utilisateur n'a qu'a executer un RET quand il a fini son travail.

Vous avez remarque que le point d'entree n'est pas le meme sur les deux TRS-80.

Entree: A = numero de case d'interrup.
DE=> bloc de travail interruptions

Sortie: BC est modifie

RESET@

Adresse 4413H (Modele I) & 4416H (Modele III)

Retire une tâche de la chaîne des interruptions.

Les points d'entrée diffèrent sur les deux modèles.

Entrée: A= numéro de la case contenant la tâche à retirer

Sortie: BC est change.

Routines diverses

DOSPLUS 3.5 fournit de nombreuses routines utiles qu'il n'est pas commode de classer dans une catégorie particulière, sinon celle-ci. Vous y trouverez le lecteur de paramètres et évaluateur d'ordres, l'évaluateur de masques, la routine de tri, des routines arithmétiques, et bien d'autres.

MULT@

Adresse 444BH (Modèle I) & 444EH (Modèle III)

Cette routine exécute une multiplication de 16 bits par 8.

Les adresses diffèrent sur les deux modèles.

Entrée: A= Multiplicateur

HL= Multiplicande

Sortie: A= LSB

L= NSB

H= MSB

DIVD@

Adresse 444EH (Modèle I) & 4451H (Modèle III)

Exécute une division de 16 bits par 8.

Les adresses diffèrent sur les deux modèles.

Entrée: A= diviseur

HL= Dividende

Sortie: A= reste

HL= quotient

PARAM@

Adresse 4476H (Modèle I) & 4454H (Modèle III)

Cette routine explore une zone de paramètres pour chercher leurs noms et leurs valeurs. Par exemple la ligne d'ordres:

DUMP TEST/CMD (START=7000H,END=71FFH,TRA=7120H)

contient 3 paramètres dans sa zone de paramètres, START, END, et TRA, dont les valeurs sont

respectivement 7000H, 71FFH, et 7120H. Le travail de PARAM@ est de trouver ces parametres et de leur attribuer leur valeur.

PARAM@ demande un BLOC DE PARAMETRES. Cela consiste en une serie d'elements, comprenant chacun un nom de parametre de 6 caracteres, justifie a gauche et complete par des blanc, et un pointeur de deux octets, qui indique a PARAM@ ou logger la valeur du parametre. Le bloc de parametres est termine par un 00H. Un bloc de parametres typique est illustre ci-dessous:

```
PRMBLK DEFH      'START'    ;PARAMETRE START
        DEFH      SRTVAL    ;POINTEUR
        DEFH      'END'     ;PARAMETRE END
        DEFH      ENDVAL    ;POINTEUR
        DEFH      'TRA'     ;PARAM TRANSFERT
        DEFH      TRAVAL    ;POINTEUR
        DEFB      0         ;FIN DU BLOC
```

C' est un bloc de parametres qui pourrait etre utilise pour l' exemple ci-dessus. Dans ce cas, la valeur de START (7000H) serait rangee en memoire a l' adresse etiquetee SRTVAL, END (71FFH) irait a l' adresse nommee ENDVAL, et TRA (7120H) en TRAVAL.

Si PARAM@ rencontre dans la zone des parametres un mot qui ne se trouve pas dans sa liste, il revient avec un etat NZ pour indiquer qu' une erreur s' est produite, et le caractere qui l' a causee sera dans le registre A. PARAM@ ne renvoie pas de code d' erreur.

PARAM@ peut accepter des valeurs numeriques ou des valeurs logiques. Une valeur numerique est une valeur comme 'data=32'. Une valeur logique est une valeur du genre "oui=vrai" ou "non=faux", comme 'parity=NO'. Si le parametre est donne sans valeur, comme dans "DIR :3 (SYS)", il prend la valeur 'vrai'. Une valeur logique 'vrai' est represente par 'FFFFH' dans la valeur du parametre, une valeur de 'faux' par '0000'.

PARAM@ ignore les virgules et les blancs qui precedent la zone des parametres.

Les adresses different sur les deux modeles. L' adresse du modele I est valable sur le modele III, mais l' l' inverse ne' est pas vrai.

Entree: DE=> bloc de parametres
HL=> texte de la zone des parametres

Sortie: BC est change
Indicateurs: Z=pas d' erreur, NZ= erreur

DSPLY@

Adresse 4467H (Modele I & III)

permet a un programme utilisateur d' afficher un bloc de texte sur l' ecran (soumis a FORCE et a JOIN). Cette routine reconnait deux caracteres de fin : ASCII 03H (ETX), qui arrete l' affichage du bloc et laisse le curseur a sa derniere position, et ASCII 0DH (RC), qui est affiche et termine l' affichage du bloc.

Entree: HL=> texte a afficher, termine par 03H ou 0DH

PRINT@

Adresse 446AH (Modele I & III)

Ecrit un bloc de texte sur l'imprimante (@PR), soumis a FORCE et a JOIN. Le bloc de texte doit etre termine par ETX, ou RC qui fait commencer une nouvelle ligne a l'imprimante.

Entree: HL=> texte a imprimer, termine par 03H ou 00H

GTIME@

Adresse 446DH (Modele I & III)

Fabrique une chaine de caracteres representant l'heure du systeme, dans une zone memoire definie par l'utilisateur, et sous le format HH:MM:SS

Entree: HL=> tampon ASCII de 8 octets.

GTDATE@

Adresse 4470H (Modele I & III)

Fabrique une chaine ASCII representant la date du systeme dans une zone definie par l'utilisateur sous le format MM/DD/AA.

Entree: HL=> tampon ASCII de 8 octets.

EVAL@

Adresse 4479H (Modele I & III)

C' est l'evalateur d'ordres de DOSPLUS. Cette routine explore la ligne d'ordres en cherchant la source <FROM>, la destination <TO>, le masque <USING>, et les zones de parametres, plaçant la valeur de chacun dans des regions de la memoire designees par l'utilisateur.

Normalement, EVAL@ designe les zones dans l'ordre FROM, TO, USING, en explorant la ligne de gauche a droite. C' est pourquoi la ligne:

COPY :1 :0 /TXT (ECHO)

serait evaluee avec ":1" comme zone source, ":0" comme zone destination, et "/TXT" comme zone masque. La zone des parametres est toujours signalee par une virgule ou une parenthese gauche. L'ordre des zones sur la ligne d'ordres peut etre modifie en utilisant les delimitateurs FROM, TO et USING, ou les caracteres de masquage. Par exemple la ligne:

COPY TO :0 USING /TXT FROM :1 (ECHO)

serait evaluee exactement comme le premier exemple, parce que les delimitateurs FROM, TO et USING indiquent quelle zone est quoi. De meme toute zone contenant des caracteres de masquage est supposee etre la zone du masque ou de USING. la ligne:

COPY !/TXT:1 :0 (ECHO)

est evaluee comme les deux exemples precedents. Quand EVAL@ rencontre "!/TXT", il le place immediatement dans la zone du masque, parce qu'il contient un caractere de masquage. EVAL@ continue ensuite dans l'ordre normal, plaçant ":1" dans la zone source, et ":0" dans la zone destination.

Pour executer sa tache, EVAL@ a besoin d'un bloc de donnees qui lui indique ou placer les donnees des differentes zones. Ce bloc est appele le bloc d'evaluation, c' est une zone memoire

de 9 octets contenant un indicateur d' un octet et 4 pointeurs de 2 octets arranges ainsi:

EVALBLK+00H	octet de fanions
	Bit 3: zone des parametres pleine
	Bit 2: zone du masque pleine (USING)
	Bit 1: Zone destination pleine (TO)
	Bit 0: Zone source pleine (FROM)
EVALBLK+01H	pointeur sur DCB source
EVALBLK+03H	pointeur sur DCB destination
EVALBLK+05H	pointeur sur DCB masque
EVALBLK+07H	pointeur bloc de parametres

Après l' execution de EVAL@, l' octet de fanions indique les zones qui ont ete detectees sur la ligne d' ordres et placees dans les DCB appropriees, ou les valeurs de parametres dans le cas de parametres.

Les DCB source, destination et masque sont des zones memoire de 33 octets qui consistent en un octet de fanions et d' un DCB de 32 octets. L' octet de fanion contient les informations suivantes:

Bit 7:	designation de peripherique dans la zone
Bit 6:	designation de fichier dans la zone
Bit 5:	la designation de fichier contient des caracteres de masque
Bit 4:	reserve
Bit 3:	la zone peripherique contient une designation de lecteur
bits 0-2:	Numero de peripherique

Après l' execution de EVAL@, les contenus des zones sont places dans leurs DCB respectifs, et l' octet de fanions peut servir a connaitre le type d' information qui est contenu dans chaque DCB. Le bit 7 a 1 indique une designation de peripherique, qui peut etre le nom d' un peripherique oriente caractere, comme @PR, ou d' un lecteur comme !1. Si c' est un nom de lecteur, il peut etre contenu dans un nom de fichier comme FICHER/DAT!2.

Le bit 6 indique qu' il s' agit d' une designation de fichier, et le bit 5 que cette designation contient des caracteres de masque.

Le bit 3 est a 1 si la designation de peripherique signalee par le bit 7 appartient a un pilote de lecteur.

Les bits 0-2 contiennent le numero logique de n' importe quel peripherique contenu dans le DCB. Ce numero peut etre utilise en liaison avec les routines LOCDCB% et LOCDCB% decrites ailleurs dans ce manuel.

Le bloc de parametres utilise par EVAL@ est legerement different de celui utilise par PARAM@. Le bloc de EVAL@ contient un ou plusieurs parametres, chacun d' eux, de 9 octets, etant constitue ainsi:

PRMBLK+00H	nom du parametre (6 octets)
PRMBLK+06H	pointeur (2 octets)
PRMBLK+08H	indicateur de type (1 octet)

Les 8 premiers octets sont identiques a ceux utilises par PARAM@. Les six premiers contenant le nom du parametre et les deux suivants un pointeur vers le DCB. Le bloc de EVAL@ a un octet supplementaire, qui indique a EVAL@ le type de valeurs acceptable pour chaque parametre.

Cet octet est appele l' indicateur de type et contient trois fanions :

Bit 7:	Chaine
--------	--------

Bit 6: Valeur numerique
 Bit 5: Valeur logique

Les valeurs de chaine doivent etre encadrees de guillemets ou d' apostrophes, et l' adresse de debut de la chaine est placee a l' adresse designee pour la valeur du parametre dans le bloc parametre. Les valeurs numeriques peuvent etre donnees en decimal ou hexadecimal, et peuvent couvrir la fourchette 0-65535 decimal ou 0000H-FFFFH. La valeur est placee a l' adresse de la valeur du parametre. Les valeurs logiques peuvent etre indiquees comme "YES" ou "ON" (vrai) ou comme "NO" ou "FALSE" (faux). Si un parametre est donne sans valeur logique, celle-ci sera supposee VRAIE. Un VRAI est represente a l' adresse de la valeur par FFFFH, et un FAUX par 0000H.

Si une ligne de commande essaie d' introduire un parametre d' un type non prevu dans l' octet de specification, EVAL@ retournera une erreur.

Notez que si l' on ne veut pas utiliser un bloc de parametres avec EVAL@, le pointeur de bloc de parametres dans le bloc d' evaluation DOIT pointer sur un octet a 00H.

Entree HL=> ligne de commande, terminee par ASCII 03H ou 0DH
 IX=> bloc d' evaluation

Sortie A= code erreur
 IX=> fanions d' evaluation
 Indicateurs: Z=pas d' erreur, NZ= erreur

Note: si une erreur se produit, HL pointera sur l' octet qui l' a causee.

=====

WILD@

Adresse 447CH (Modele I & III)

WILD@ sert a comparer une designation de fichier a un masque et retourne un indicateur d' etat qui indique si le nom correspond au masque. WILD@ en realite remplit deux fonctions: la premiere est d' etablir le masque a utiliser pour les comparaisons a venir, la seconde est d' effectuer la comparaison proprement dite. Le nom de fichier ou le masque doivent etre correctement termines par un ASCII 03H ou 0DH.

Entree: B= selecteur de fonction
 si B=0, comparaison du nom et du masque
 si B=1, etablissement d' un nouveau masque
 HL=> nom de fichier a comparer, ou masque a etablir

Sortie: Indicateurs: Si B=0, Z=nomfich correspond au masque, NZ= pas de correspondance
 Si B=1, Z= masque etabli, NZ= masque invalide

=====

LOCDEV@

Adresse 402DH (Modele I & III)

Si on lui donne une designation de peripherique, cette routine retourne un numero de peripherique. Comme tous les peripheriques peuvent etre rebaptises a volonte, cette routine est tres pratique pour trouver quel peripherique correspond a quel nom. Le numero de peripherique peut alors etre utilise pour localiser les DCB ou DCT appartenant a ce peripherique en utilisant les routines LOCDCB% et LOCDCZ%. Le numero de peripherique est retourne dans le registre A, bits 0-2. Le bit 3 du numero de peripherique vaut 1 si le peripherique est un lecteur de disques, il vaut 0 dans le cas contraire.

Entree: DE=> nom de preipherique (2 caracteres precedes de '@' ou '!')

Sortie: A= Bit 3: indicateur de lecteur
 Bit 0-2: numero de peripherique
 Indicateurs: Z= peripherique trouve, NZ= non trouve.

SORT0

Adresse 447FH (Modele I & III)

sert a trier un bloc memoire compose d' un nombre quelconque de mots d' une longueur definie par l' utilisateur. Tous les mots de la liste a trier doivent etre de la meme longueur. La cle de tri peut commencer n' importe ou dans le mot, et etre d' une longueur quelconque, sans depasser celle des mots a trier.

Entree: B= decalage entre les premiers caracteres de la cle et du mot
 C= longueur de la liste des mots
 DE=> 1 eroctet du dernier mot
 H= selecteur de tri: H=0: tri ascendant, H=1: tri descendant
 L= longueur de la cle
 IX=> premier octet du premier mot de la liste

Sortie: A= code d' erreur
 Indic. Z= pas d' erreur, NZ= erreur

Vecteurs internes du systeme

Cette categorie englobe trois routines tres utiles: le systeme d' E/S disque, et deux routines utilisees pour localiser les DCT et DCB du systeme.

DISKIO@

Adresse: 4485H (Modele I) & 4488H (Modele III)

C' est le systeme d' E/S disque de DOSPLUS 3.5. Grace a lui l' utilisateur peut acceder a n' importe quel cylindre et n' importe quel secteur de n' importe quel disque. La routine DISKIO@ a dix fonctions de base:

Code de la fonction	Nom de la fonction	Description de la fonction
0	DCHECK	Verifie l' etat du lecteur
1	DHOME	Met la tete en piste 0 et initialise le lecteur.
2	DSEEK	Place la tete sur un cylindre
3	DREAD	Lit un secteur
4	DVERF	verifie un secteur
5	DWRITE	ecrit un secteur
6	SREAD	lit un secteur du directory
7	SWRITE	ecrit "" ""
8	DWRITA	ecrit un secteur systeme
9	DFORMT	formate une piste ou un cylindre

Toutes les fonctions de DISKIO@ ont besoin des memes informations et fournissent les memes indications en sortie:

Entree: A= code de la fonction
 C= numero du pilote lecteur (0-7)
 D= numero du cylindre
 E= numero logique du secteur
 HL=> tampon d' E/S disque

Sortie: A= code d' erreur
 Indicateurs: Z= pas d' erreur, NZ= erreur

Pour toutes les fonctions de DISKIO@, le code de fonction est charge dans le registre A.

Le registre C doit contenir le numero du pilote lecteur (fourni par les routines LOCDEV% et EVAL@, ou la position du bloc DCTBEL de ce lecteur).

Le registre D contient le numero de cylindre. Dans le cas de disques durs fractionnes, ce numero de cylindre est le numero du cylindre DANS LE VOLUME, pas le numero physique.

Le registre E contient le numero du secteur, compte a partir du premier secteur du cylindre. Le pilote du lecteur doit ajouter le decalage des secteurs range dans la DCT du lecteur pour obtenir le numero physique du secteur.

La paire HL doit pointer sur un tampon d' E/S disque de 256 octets. Toute donnee a ecrire sur le disque doit etre rangee auparavant dans ce tampon, toute donnee lue sur le disque y apparaitra.

DCHECK

Cette fonction sert à déterminer l'état de préparation du lecteur. Pour cette fonction, seul le code de fonction et le numéro du pilote lecteur doivent être fournis. Au retour de DCHECK0, l'indicateur d'état sera à Z si le lecteur est prêt, sinon à NZ. L'indicateur C sera aussi à 1 si le disque est protégé en écriture.

DHOME

Cette fonction met la tête de lecture/écriture sur le cylindre de numéro logique 0. DHOME ne nécessite que les deux mêmes informations que DCHECK.

DSEEK

sert à placer la tête de lecture/écriture sur le cylindre dont le numéro logique est indiqué. Le code de fonction, le numéro de pilote, et bien sûr le numéro de cylindre doivent être fournis à l'entrée dans DSEEK.

DREAD

lit le secteur indiqué sur le disque. Les données lues sont rangées dans le tampon pointé par HL. Toutes les données prévues au début doivent être fournies.

DVERF

Cette fonction est similaire à DREAD ci-dessus en ce que elle lit un secteur du disque dans un tampon pointé par HL. La différence est dans le fait qu'en cas d'erreur, DVERF ne tente pas de relire le secteur, mais abandonne immédiatement et signale l'erreur.

DWRITE

écrit sur le disque les données rangées dans le tampon de 256 octets pointé par HL.

SREAD

lit un seul secteur du cylindre du directory. Seuls ce code de fonction, le numéro de pilote et le numéro de secteur doivent être fournis à SREAD, qui localisera automatiquement le cylindre du directory. Les données lues seront placées dans le tampon d'E/S disque pointé par HL.

SREAD remplit une autre fonction importante sous DOSPLUS 3.5. Elle initialise la portion perissable de la DCT du lecteur si l'indicateur 'log' (DCT+03H, bit 0) est à 1. Sur les disques souples, cela signifie lire la DCT avec les données rangées dans le pilote lui-même, et lire la GAT pour déterminer si le disque est à simple ou double face. Le nombre de surfaces est mis à jour avec cette information.

Sur les disques durs, l'initialisation du DCT par SREAD est plus simple. SREAD n'a qu'à lire l'information perissable de la DCT logée dans BOOT/SYS (cylindre 0, secteur 2), et la ranger dans la DCT.

SWRITE

cette fonction est utilisée pour écrire un secteur dans le directory du disque. Il suffit de donner les mêmes informations que pour SREAD, pour les mêmes raisons.

DWRITEA

fait la même chose que DWRITE, à la différence suivante: DWRITEA écrit un secteur VERROUILLÉ ou PROTÉGÉ, réservé au directory. C'est d'une utilisation courante par les programmes

de formatage pour ecrire le directory d' un disque.

DIFORMT

permet de formater n' importe quel cylindre ou n' importe quelle piste sur un lecteur. Le choix entre une piste isolee ou un cylindre complet est l' affaire du pilote. Le pilote standard de disques souples fourni avec DOSPLUS 3.5 formate une seule piste.

LODCCB%

Adresse: 4488H (Modele I) & 44A0H (Modele III)

Cette routine peut etre utilisee pour localiser le bloc de controle de peripherique de n' importe quel peripherique oriente caractere. Quand on lui donne le numero de pilote, LOCCCB% retourne son adresse. C' est tres utile car DOSPLUS 3.5 permet aux programmes utilisateurs de reloger les DCB n' importe ou en MEV, et en fait certains peripheriques n' ont pas d' adresse tant qu' un pilote ne leur a pas ete ASSIGNE. Si LOCCCB% est utilise pour un peripherique non defini, elle renvoie FFFFH.

Notez que cette routine n' a pas la meme adresse sur Modele I et sur Modele II.

Entree: A= Numero de peripherique

Sortie: DE=> bloc de controle du peripherique.

LOCCDCT%

Adresse: 448BH (Modele I) & 44A3H (Modele III)

Fait pour les lecteurs de disques ce que fait LOCCCB% pour les peripheriques orientes caracteres.

Entree: A= Numero du pilote de lecteur

Sortie: DE=> table de controle du lecteur

Routines utiles de la ROM

En supplement aux points d' entree dans DOSPLUS 3.5, la ROM des TRS-80 contient des routines tres pratiques, nous en presentons quelques unes:

GET@

Adresse: 0013H (Modele I & III)

C' est une partie du systeme d' E/S caractere (CIO) de la ROM du TRS-80. La routine GET@ sert a lire un seul caractere, dans n' importe quel peripherique ou fichier du systeme. A la lecture dans un peripherique, il ne faudrait tenir compte d' aucun code d' erreur retourne par GET@.

Entree: DE=> DCB ou FCB ouvert

Sortie: A= caractere du peripherique ou fichier
Indicateurs: inconnu si peripherique (depend du pilote)
Si fichier: Z= pas d' erreur, NZ= erreur, code dans A

PUT@

Adresse: 001BH (Modele I & III)

c' est la partie sortie caractere du systeme d' E/S caracteres de la ROM. Il permet d' envoyer un octet a n' importe que peripherique ou fichier. Ne pas tenir compte des indications d' erreur.

Entree: A= caractere a envoyer
DE=> DCB ou FCB ouvert

sortie: Indicateurs: Inconnus si peripheriques (dependent du pilote)
si fichier: Z= pas d' erreur, NZ=erreur, code dans A

KBD@

Adresse: 002BH (Modele I & III)

va chercher un caractere dans le pilote clavier. Typiquement, le pilote retourne l' etat dans l' indicateur Z, bien que cela depende du pilote de clavier en service. Ceux de la ROM et de DOSPLUS 3.5 retournent NZ si un caractere a ete trouve, Z autrement.

Sortie: A= touche enfoncee
DE modifie

DSP@

Adresse: 0033H (Modele I & III)

Envoie un seul caractere au pilote d' affichage.

Entree: A caractere a afficher

Sortie: DE modifie

PRTE

Adresse: 003BH (Modele I & III)

Envoi d' un caractere a l' imprimante.

Entree: A= caractere a envoyer

Sortie: DE modifie

KEYIN

Adresse: 0040H (Modele I & III)

Accepte une ligne d' entree tapée au clavier, terminee par pression sur <ENTER> ou <BREAK>.

Entree: B= longueur maximum de la ligne
 HL=> tampon d' entree

Sortie: B= nombre de caracteres recus, moins le terminateur
 C= longueur maxi de la ligne
 DE est modifie
 Indicateurs: Z si rien reçu
 C si <BREAK> presse

KEY

Adresse: 0049H (Modele I & III)

fonctionne comme KBD, mais ne retourne au programme appelant que quand une touche a ete pressee.

Sortie: A= touche pressee
 DE modifie

IX. - Adresses importantes de la memoire.

Plusieurs zones de donnees DOSPLUS 3.5 peuvent etre utiles a ceux qui ecrivent des programmes destines a etre utilises sous DOSPLUS. Notez que les adresses marquees d' un asterisque (*) peuvent changer dans de futures versions de DOSPLUS.

BREAK*

Adresse: 4312H (Modele I) & 42ADH (Modele III)

Cette adresse contient le vecteur de la touche BREAK, ou l' adresse ou est passe le controle du programme quand cette touche est enfoncee. C' est une zone de 3 octets qui contient normalement un RET, mais l' utilisateur peut y mettre un vecteur vers toute routine de son choix.

DATE\$

Adresse: 4044H (Modele I) & 42A1H (Modele III)

Adresse ou est rangee de la date sur 3 octets, dans l'ordre annee, jour, mois.

* DDOCB\$

Adresse: 435A (Modele I) & 429AH (Modele III)

contient sur 2 octets l'adresse du bloc de 288 octets (un FCB de 32 octets et un tampon d'E/S de 256 oct. affecte par le systeme a l'usage des fichiers DO. Le bit 6 de SFLAG\$, (plus loin) indique si un bloc DO a ete reserve.

HIMEM\$

Adresse: 4049H (Modele I) & 4411H (Modele III)

2 oct. pointent a tout moment sur la plus haute adresse libre de la MEV utilisateur. Cette valeur change de temps a autre quand divers pilotes, filtres et programmes sont charges en memoire haute. C'est la responsabilite de tout programme operant sous DOSPLUS de respecter cette valeur, c'est-a-dire qu'aucun programme ne doit utiliser une adresse superieure a celle contenue dans HIMEM\$.

INPBUF\$

Adresse: 4318H (Modele I) & 4225H (Modele III)

C'est la zone du tampon de commande de DOSPLUS. Tous les ordres entres au niveau commande de DOSPLUS sont places dans ce tampon de 64 caracteres.

* OVLYH\$

Adresse: 4308H (Modele I) & 429EH (Modele III)

Cet octet contient une valeur qui indique lequel des fichiers du groupe des auxiliaires (overlays) hauts est presentement en memoire. Les quatre bits de poids fort contiennent le numero du systeme -1. Les programmes utilisateurs utilisent cet octet principalement pour forcer le systeme a recharger un fichier systeme en chargeant FFH dans OVLYH\$.

* OVLYL\$

Adresse: 430AH (Modele I) & 429DH (Modele III)

Cet octet contient une valeur qui indique lequel des fichiers du groupe des auxiliaires (overlays) bas est presentement en memoire. Les quatre bits de poids fort contiennent le numero du systeme -1. Les programmes utilisateurs utilisent cet octet principalement pour forcer le systeme a recharger un fichier systeme en chargeant FFH dans OVLYL\$.

* SFLAG1\$

Adresse: 4302H (Modele I) & 4297H (Modele III)

Cet octet contient huit fanions qui refeletent l'etat actuel du systeme:

Bit	Signification
7	reserve
6	bloc DD alloue
5	indicateur DD actif
4	reserve
3	indicateur VERIFY
2	"" DEBUG
1	"" acces RUN
0	"" touche BREAK neutralisee

Le bit 6 indique qu' un bloc de 288 octets a ete reserve en MEV a l' usage d' un ordre DD.

Le bit 5 indique qu' un fichier DD est actuellement actif, pour informer le pilote du clavier d' aller chercher les donnees dans le fichier DD.

Le bit 3 indique au systeme d' effectuer une verification de lecture apres ecriture de toutes les donnees ecrites sur disque.

Le bit 2 active DEBUG.

Le bit 1 est mis a 1 par le systeme pendant l' execution d' une routine RUN@, pour eviter des erreurs 'File acces denied due to password protection' en essayant d' executer un fichier dont le niveau de protection est 'execution seule'.

Le bit 0 neutralise la touche BREAK, de sorte qu' aucun code de touche n' est envoye quand cette touche est pressee.

=====

TIME\$

Adresse: 4041H (Modele I) & 4217H (Modele III)

zone de 3 octets ou est rangee l' heure du systeme dans l' ordre heures, minutes, secondes.

=====

=====

OSVER\$

Adresse: 403EH (Modele I seulement)

cet octet contient le numero de version du SED, en format BCD. Les 4 bits de poids fort contiennent le numero de version, les 4 bits de poids faible le numero d' edition.

=====

X. - Ecriture de pilotes pour DOSPLUS 3.5.

DOSPLUS 3.5 permet à l'utilisateur l'écriture de pilotes de périphériques, programmes qui assurent l'interface entre le système et un équipement extérieur, comme des lecteurs de disques, une imprimante, un lecteur de cartes, etc. Cette section du manuel contient les informations nécessaires à l'écriture de ces pilotes.

L'écriture de tels programmes est une tâche simple pour le programmeur familier avec l'assembleur du Z-80. DOSPLUS remplit lui-même une certaine partie du travail, fournissant des tables de filtrage, une capacité importante de redirection (FORCE et JOIN) à l'intérieur des routines d'E/S du système, une puissante structure de DCT et DCB, etc.

Tous les pilotes sont installés en MEV par l'ordre ASSIGN, dont la forme générale est :

ASSIGN <FROM> @ds/ldr <TO> nomfich (param=expr, param=expr,...)

ou "ds" est la désignation de l'un des six périphériques orientés caractères (@KI, @DO, @PR, @RS, @UI et @UZ) ou l'un des 8 pilotes de disques (de :0 à :7), et 'nomfich' le nom d'un programme pilote pour ce périphérique particulier. La commande ASSIGN charge le pilote en MEV et lui passe la main. A ce moment, les registres suivants contiennent des informations importantes et utiles :

Entree:	BC=>	adresse minimum de la MEV utilisateur
	DE=>	"" maximum "" ""
	HL=>	première zone de DOSPLUS suivant le nom du pilote
	IX=>	le DCB/DCT pour le périphérique
	IY=>	la zone de DCBTBL/DCTTBL pour ce périphérique

C'est ensuite la responsabilité de ce programme de se reloger en haut de mémoire et d'ajuster HIMEM\$ pour se protéger. Il doit aussi mettre son point d'entrée dans le DCB/DCT du périphérique à DCB/DCT+01H et DCB/DCT+02H, et aussi initialiser dans le DCB/DCT toute donnée dont il peut avoir besoin.

Si un périphérique n'a pas de DCB/DCT défini au moment de ASSIGN, DOSPLUS va lui en créer un de 20 octets en haut de mémoire, dont il mettra l'adresse automatiquement dans DCBTBL ou DCTTBL. Cette adresse sera aussi enregistrée dans le registre IX.

Si le DCB/DCT d'un périphérique n'existe pas encore, le programme peut le reloger en MEV utilisateur à une adresse quelconque, en modifiant simplement le pointeur dans le DCB/DCT en IY+00 et IY+02 à l'entrée dans le programme.

Le pilote peut recevoir des paramètres de la ligne de commande de DOSPLUS par l'utilisation de FSPEC et PARAM ou de EVAL.

Pilotes de disques

les pilotes de disques de DOSPLUS 3.5 doivent supporter les dix fonctions décrites dans la routine système DISKIO\$. A partir des informations fournies, le pilote doit remplir la fonction demandée par le programme appelant. Tous les registres nécessaires peuvent être utilisés sans précautions particulières, le système se chargeant de leur sauvegarde et de leur restauration aux moments opportuns. En cas d'erreur, le pilote doit retourner l'indicateur Z à 0 et le code d'erreur dans A, et retourner NZ dans le cas contraire.

Un exemple de pilote de disque est reproduit plus loin (mais non traduit). C'est un bon modèle de structure et de fonctionnement de pilote.

Pilotes de peripheriques orientes caracteres

Les pilotes pour les six peripherie orientes caracteres sont bien plus simples que ceux des disques. Ces pilotes n'ont a supporter qu'une, deux ou trois des fonctions elementaires d'E/S caractere remplies par CIO: entree, sortie, E/S de controle.

Quand CIO passe le controle au pilote, les registres du Z-80 contiennent les donnees suivantes:

```
Entree:      B=      type d' E/S
              bit 2: controle
              bit 1: sortie
              bit 0: entree
              C=      caractere a emettre
              IX=>    bloc de controle du peripherique
Indicateurs: Z= ecriture, NZ= lecture
```

L'usage des registres est libre (voir plus haut).

Notez que dans le cas d'une entree de caractere, le bon usage voudrait que l'on renvoie un indicateur pour faire savoir au programme appelant si l'on a reçu un caractere ou non. generalement, l'indicateur Z est mis a 1 si l'on n'a rien reçu, a 0 dans le cas contraire. En effet beaucoup de programmes supposent que le pilote renvoie un indicateur utilisable.

Les pilotes d'E/S de caracteres sont charges du filtrage, ou conversion des valeurs de caracteres. La DCBTBL contient pour chaque pilote un pointeur sur une table de filtrage, et le DCB lui-meme contient dans l'octet de type un bit qui indique si un filtrage doit etre execute. Si ce bit (bit 6 de DCB+00H) est a 1, un filtrage devrait etre execute sur tous les octets echanges par ce pilote.

La table de filtrage, pointee dans la DCBTBL par les octets Entree+02H et Entree+03H est de longueur variable. Son premier octet indique sa longueur en octets, 0 indiquant 256, puis viennent les elements de la table, de deux octets chacun, le premier est le caractere a filtrer, le deuxieme sa nouvelle valeur. Les caracteres ne figurant pas dans la table devraient ressortir inchanges.

Un exemple de pilote RS232 (en Version Originale) est joint a ce manuel pour donner un exemple de structure et de fonctionnement de pilote oriente caractere.

Codes d' erreur de DOSPLUS 3.5.

Vous avez ci-dessous la liste des codes d' erreur et des messages correspondant de DOSPLUS 3.5:

Dec	Hex	Message	
0	00	pas d' erreur	
1	01	CRC error during header read	erreur CRC en lecture en-tete
2	02	Seek error during read	erreur de piste en lecture
3	03	Lost data during read	perte des donnees en lecture
4	04	CRC error during read	erreur CRC en lecture
5	05	Data record not found during read	donnees non trouvees en lecture
6	06	Attempted to read locked/deleted data	essai de lecture de donnees interdites/effacees
7	07	Attempted to read system data record	essai de lire un enregistrement systeme
8	08	Drive not available	lecteur indisponible
9	09	CRC error during header write	erreur CRC en ecriture d' en-tete
10	0A	Seek error during write	erreur de piste en ecriture
11	0B	Lost data during write	donnees perdues en ecriture
12	0C	CRC error during write	erreur CRC en ecriture
13	0D	data record not found during write	enregistrement non trouve en ecriture
14	0E	Write fault on disk drive	defaut d' ecriture sur le lecteur
15	0F	Write protected disk	disque protege en ecriture
16	10	Illogical file number	numero de fichier illogique
17	11	Directory read error	erreur de lecture du directory
18	12	Directory write error	erreur d' ecriture du directory
19	13	Improper filename	nom de fichier impropre
20	14	GAT read error	erreur de lecture de la GAT
21	15	GAT write error	erreur d' ecriture de la GAT
22	16	HIT read error	erreur de lecture de la HIT
23	17	HIT write error	erreur d' ecriture de la HIT
24	18	File not in directory	fichier inconnu au directory
25	19	File access denied due to password protection	
		Acces a ce fichier refuse (protection par mot de passe)	
26	1A	Directory space full	Directory complet
27	1B	Disk space full	disque plein
28	1C	Attempted to read past EOF	essai de lire apres la fin du fichier
29	1D	Attempted to read outside file limits	essai de lire hors du fichier
30	1E	Directory full can't extend file	Directory complet, extension impossible
31	1F	Program not found	programme inconnu
32	20	Improper drive number	numero de lecteur incorrect
33	21	No memory space available	pas de place memoire disponible
34	22	Attempted to use a non program file as a program	
		Essai d' utiliser comme programme un fichier qui n' en est pas un	
35	23	Memory fault during program load	
		defaut de memoire pendant le chargement du programme	
36	24	Attempted to load read only memory	essai de charger la memoire morte
37	25	Illegal access attempted to protected file	tentative de viol d' un fichier protege
38	26	I/O attempted to unopen file	essai d'E/S sur un fichier non ouvert
39	27	Device in use	peripherique en service
40	28	Protected system device	peripherique systeme protege
41	29	Device not available	peripherique indisponible
42	2A	No device space available	
		pas de place pour un peripherique (ou son pilote)	
43	2B	Illegal devspec	designation de peripherique illegale

Manuel utilisateur DOSPLUS 3.50

44	2C	Illegal filespec	designation de fichier illegale
45	2D	Invalid data provided	fourniture de donnees invalides
46	2E	Invalid parameter	parametre invalide
47	2F	I/O field not found	zone d' E/S introuvable
48	30	Terminated	termine
49	31	File already exists	ce fichier existe deja
50	32	Device already exists	ce peripherique (ou son pilote) existe deja