RANDOS v1

Texte tapé par Fabrice Francès

Mode d'emploi du système d'exploitation de micro disque RANDOS.

Nous nous bornons ici à indiquer les différences avec le DOS décrit dans le manuel du micro disques Oric.
RANDOS offre un certain nombre de possibilités supplémentaires et élargit l'utilisation de certains fonctions du DOS.
Citons tout de suite les répertoires arborescents. Supposons que trois personnes : Luc, Jean et Paul utilisent le même disque. Chacun créera sa propre liste de programme. Au départ, le répertoire contiendra LUC.DIR, JEAN.DIR et PAUL.DIR.
L'appel de LUC.DIR affichera les programmes de Luc seulement. C'est plus commode. On peut aussi dans une gestion distinguer des chapitres et des sous-chapitres.

!BACKUP fonctionne comme avec le DOS. Pour recopier des fichiers d'un disque DOS à un disque RANDOS ont dispose sur le disque d'un programme spécial TRANS.
!BACKUP avec RANDOS ne copie que les zones qui contiennent des fichiers. De la sorte, la recopie se fait le plus rapidement possible.

!MAKE est utilisé pour créer un sous-répertoire. Le nom de ce sous-répertoire sera affiché avec le suffixe .DIR et doit être différent des noms des autres fichiers. Si vous choisissez une extension autre que DIR elle sera ignorée et automatiquement remplacée par .DIR
!MAKE"JEUX" créera un sous-répertoire JEUX.DIR. Pour entrer dans ce sous-répertoire il faut utiliser !CHANGE.
!MAKE ,S indique que le sous-répertoire en service doit être la racine : c'est celui qui servira au début, lors de la mise en route du système.

!CHANGE sert à changer de répertoire. Si l'on n'indique rien après CHANGE, on obtient le répertoire "racine". Si le sous-répertoire appelé n'existe pas dans le répertoire actuel, un message d'erreur sera généré. Inutile de donner de suffixe, il sera ignoré de toute façon et remplacé par .DIR.
!FORMAT amène le répertoire en service à devenir la racine.

!DIR affiche tous les fichiers du répertoire en service. Si plus d'un écran est nécessaire, un appui sur une touche affiche la page suivante. Escape permet de sortir. Le lecteur utilisé est celui choisi par défaut.
!DIR 1 affiche le répertoire du lecteur n° 1.
!DIR"*.DIR" affiche tous les sous répertoires.
!DIR"1-*.BAS" affiche tous les fichiers BASIC du lecteur n° 1, extraits du répertoire en service.
!DIR"XX.SUF" affiche ce seul fichier.
!DIR,A donne un répertoire détaillé.

!DEL permet d'effacer des fichiers, mais ne peut pas effacer les répertoires sauf s'ils viennentt d'être créés et sont vides. Il faut préalablement annuler la protection par !PROT"XX.DIR",N avant d'effacer le répertoire par !DEL"XX.DIR"

!COPY fonctionne comme avec le DOS. Une nouveauté : l'option ; qui rend le nom du fichier invisible à l'écran lors de l'affichage du répertoire. Une autre option est ,B. Lorsqu'on copie un fichier avec cette option, cela provoque l'inscription de ,B dans la liste des options de ce fichiers. Si on modifie ce fichier, cette indication est effacée. De la sorte, une mise à jour des sauvegardes est facilitée. Il suffit d'appeler les programmes d'un autre type et ne seront recopiés que ceux qui n'ont pas l'option ,B dans leur description. L'utilisation des caractères "carte blanche" est bien sûr intéressante ici.

!DRV
!DRV 1 indique que le lecteur enregistreur numéro 1 est choisi comme unité prioritaire.
!DRV ? vous renvoie le numéro de l'unité prioritaire.
Si vous choisissez une unité qui n'existe pas, l'appareil va le chercher désespérément ...

!FORMAT
!FORMAT 0 (Ne pas oublier l'espace entre FORMAT et 0)
Lorsque vous envoyez cette commande, il vous est demandé :
SYSTEM DISC ? Entrez Y si vous voulez que le RANDOS figure sur le disque que vous initialisez. C'est le cas le plus fréquent. Sinon, taper N.
Si vous avez répondu Y, vous voyez s'afficher :
READING SYSTEM...
READING DOS PROGRAMS.....
Mettez alors le disque a initialiser dans le lecteur que vous avez indiqué au début et appuyez sur RETURN.
On vous demande alors :
No OF TRACKS ? (nombre de pistes) Répondez 40 ou 80 suivi de RETURN.
DISC NAME ? (nom du disque) Indiquez le en 6 caractères au plus puis RETURN.
DUAL SIDED ? (double face) Tapez Y (oui) ou N (non)
L'initialisation commence alors.
Nota: il n'est pas possible de sous-entendre le numéro de l'unité (donc indiquer !FORMAT 0 ou !FORMAT 1, etc.)
Cette commande détruit le contenu de la mémoire vive et clôt tous les fichiers restés ouverts.

!LOAD est analogue à celle du DOS.
Elle clôt les fichiers restés ouverts.
Les fichiers dont le suffixe est .BAS ou .COM peuvent être rappelés par ! suivi de leur nom sans le suffixe. Ils démarrent automatiquement.

!PROT propose une nouvelle option : ,A autorisant l'allonge de ce fichier.

!REN ne peut pas être utilisé pour les noms de répertoire.

!SAVE fonctionne comme pour le DOS.
On dispose en plus de l'option ,O qui autorise la sauvegarde d'un fichier alors que le même existe déjà sur le disque. Dans ce cas, le nouveau remplace l'ancien. Il est écrit à sa place.

!SYS fonctionne de façon analogue à celle du DOS.

!BUILD s'utilise pour créer un fichier texte sur le disque à partir de caractères entrés au clavier. Après avoir entré !BUILD"ESSAI.TXT", les données entrées au clavier iront dans ce fichier dès que vous aurez tapé CTRL-C. Si vous tapez sur RETURN, un retour chariot et un saut de ligne seront générés pour faciliter la mise en page.

!TYPE rappelle à l'écran les fichiers texte créés par !BUILD ou par un programme qui crée des fichiers texte. La barre d'espace s'utilise comme lors de l'affichage d'un listing BASIC et CTRL-C pour interrompre le processus. Des programmes BASIC ne peuvent pas être ainsi rappelés, cela tient à la façon dont ils sont stockés sur le disque.

Messages d'erreur

1277 ou #4FD
0 erreur avec le DOS obligeant une interruption du BASIC avec affichage d'erreur.
1 erreur avec DOS ne provoquant pas d'arrêt du programme, un numéro d'erreur est envoyé en 1279 (#4FF).

Normalement, lorsqu'une erreur se produit, un message d'erreur s'affiche. On peut supprimer l'affichage par POKE 1277,1 et le rétablir par POKE 1277,0. On peut utiliser le nombre contenu en 1279 pour des sous-programmes de traitement d'erreur. Voici la liste des numéros d'erreur et leur sens.

0 Pas d'erreur.
1 Fichier non trouvé

2 Fin de commande erronée
3 Numéro du lecteur oublié
4 Numéro de lecteur non valable
5 Noms de fichiers non valable
6 Erreur sur le disque
7 Attribut non valable
8 "Carte blanche" non autorisée
9 Fichier déjà créé.
10 Place insuffisante sur le disque
11 Fichier déjà ouvert
12 Répertoire déjà en service
13 Il manque l'adresse de fin
14 Adresse de début supérieure à l'adresse de fin
15 Oubli de 'TO'
16 Les fichiers ne sont pas sur le même disque
17 Tableau non dimensionné
18 Lecteurs distincts
19 Disques de types différents
20 Mauvais disque. Formatage impossible.
21 Fichier déjà sauvegardé
22 Syntaxe
23 Le nom de fichier manque
24 Numéro de fichier en service
25 Confusion de type
26 Disque protégé contre l'écriture
27 Disque changé, impossible d'écrire
28 Numéro de fichiers non en service
29 Fin de fichier
30 Répertoire non trouvé
31 Répertoire existant
32 Fichier ouvert en lecture seulement
33 Tableau trop grand
34 Numéro d'enregistrement trop grand
35 Numéro de fichier non valable
36 Trop grand nombre de fichiers ouverts
37 Réservé
38 Fin d'enregistrement
39 Réservé
40 Réservé
41 Le fichier utilisé
42 Mémoire insuffisante

Les fichiers de données.

Avec le système RANDOS on dispose de fichiers à accès séquentiel, à accès direct, possibilité d'accès octet par octet, sauvegardes et rappels de données en tableaux.

Fichiers, enregistrements, champs.
Un fichier contient un certain nombre d'enregistrements, chaque enregistrement étant découpé en parties appelées champs. Les champs sont usuellement désignés par des noms de variables NOM$, TEL$, ADR$...
un disque peut comporter un nombre quelconque de fichiers. On ne pourra toutefois en exploiter que 8 simultanément. Chaque fichier peut comporter jusqu'à 32768 enregistrements de chacun 65536 octets aux plus. Tout cela dans la limite de la place disponible sur le disque. Dans la pratique, un fichier contient quelques centaines d'enregistrements de chacun quelques centaines d'octets au plus.

Description d'un enregistrement.
Voici un exemple de champs d'un fichier d'adresses.
RANDOS doit connaître la longueur de chaque champ. Il faut donc prévoir cinq octets ici puisqu'il y a cinq champs. Chaque enregistrement occupera donc 175 octets. On peut enregistrer mille personnes (de 0 à 999). Supposons que notre fichier comporte 300 noms, il faut réserver sur le disque un espace de 300 x 175 égal 52500 octets.

Pour gagner de la place.
Quand le RANDOS est lancé, une partie vient s'installer en mémoire vive. Certaines commandes restent sur le disque et seront appelées selon les besoins en OVERLAY. Voici le contenu des 9 systèmes OVERLAY.

SYSTEM.OV0 Messages d'erreur.
SYSTEM.OV1 - GET, OPEN, PUT, SET, CREATE, EXTEND, FILES
SYSTEM.OV2 - DIR, SAVE, OLD, DRV
SYSTEM.OV3 - FORMAT
SYSTEM.OV4 - BACKUP
SYSTEM.OV5 - STORE, RECALL, CHANGE, MAKE
SYSTEM.OV6 - DEL, PROT, REN, BUILD, TYPE
SYSTEM.OV7 - routines pour accès octet par octet RNDBYTE, WRNBYTE, SETRAN
SYSTEM.OV8 - COPY

On peut supprimer d'un disque celles qui ne servent pas.
1. Les déprotéger. 2. Les effacer.
Attention à OV7 appelé par divers autres.

Comme RANDOS initialise le micro disque en y marquant des secteurs de 512 octets à raison de 9 secteurs par piste, pour un disque simple face comportant 40 pistes la place disponible est 184000 octets environ.

Écriture d'un fichier séquentiel.

```
5 !DEL"ADR.DTA"
10 !FILES 1
20 !OPEN 1,"ADR.DTA",W
30 INPUT N$
32 IF N$="F" THEN 70
35 INPUT NOM$,ADR$,CPV$,TEL$
40 PRINT
50 !PUT 1,N$,NOM$,ADR$,CPV$,TEL$
60 GOTO 30
70 !CLOSE 1
```

En ligne 10, on précise qu'on ouvre un seul fichier. Une partie de la mémoire vive (usuellement 512 octets) est réservé : c'est un tampon. L'instruction !FILES 1 doit précéder toutes les commandes concernant les fichiers. En ligne 20, on ouvre le fichier "ADR.DTA" en écriture (W=Write)
En ligne 30-35 on saisit les données au clavier. La procédure est ici simplifiée au maximum. En ligne 50, on demande l'écriture sur le disque. On est envoyé en ligne 70 pour fermer le fichier par introduction de "F".

Lecture des fichiers séquentiels.

```
100 !FILES 1
110 !OPEN 1,"ADR.DTA",R
130 INPUT K$
140 POKE 1277,1
150 !GET 1,N$,NOM$,ADR$,CPV$,TEL$
160 IF PEEK(1279)=29 THEN PRINT "PAS TROUVE":PRINT:GOTO 200
170 IF K$<>N$ THEN 150
```

```
180 PRINT N$,NOM$,ADR$,CPV$,TEL$
190 POKE1277,0
200 !CLOSE 1
```

En ligne 110, on ouvre le fichier "ADR.DTA" en lecture (R=Read)
Les lignes 130, 160 et 190 détectent la fin de fichier et agissent en garde fou. La ligne 150 lit les enregistrements. La ligne 180 procède à l'affichage quant la bonne fiche a été trouvée ce que vérifie la ligne 170. Enfin la ligne 200 ferme le fichier après utilisation. L'utilisation d'un fichier séquentiel oblige la lecture de tous les enregistrements jusqu'à celui demandé. Un fichier à accès direct est bien plus rapide à l'emploi, mais un peu plus délicat à créer.

Initialisation d'un fichier à accès direct.

On commence par réserver sur le disque la place nécessaire à la totalité du fichier. C'est la commande !CREATE qui inscrit des caractères nuls (cde ASCII 0) à l'emplacement des fichiers assurant ainsi l'effacement d'éventuelles données rémanentes.

```
!CREATE"ADR.DTA",175,300
```

Les nombres 175 et 300 correspondent à l'exemple choisi. (voir description d'un enregistrement).

Écriture d'un fichier à accès direct.

```
10 !FILES 1
20 !OPEN 1,"ADR.DTA",D,175
30 INPUT N$
32 IF N$="F" THEN 70
35 INPUT NOM$,ADR$,CPV$,TEL$
40 !SET 1,VAL(N$)
50 !PUT 1,N$,NOM$,ADR$,CPV$,TEL$
60 GOTO 30
70 !CLOSE 1
```

Remarquez comment, en ligne 40, on demande à la tête d'écriture de se placer pour la fiche VAL(N$).

Lecture d'un fichier à accès direct

```
100 !FILES 1
110 !OPEN 1,"ADR.DTA",D,175
130 INPUT K$: K=VAL(K$)
135 IF K<0 OR K>300 THEN 130
140 !SET 1,K
150 !GET 1,N$,NOM$,ADR$,CPV$,TEL$
160 IF N$="" THEN PRINT "PAS TROUVE":PRINT:GOTO 200
180 PRINT N$,NOM$,ADR$,CPV$,TEL$
200 !CLOSE 1
```

La ligne 140 place à la tête de lecture là où il faut pour lire la fiche numéro K. Si la variable N$ est vide, c'est que la fiche n'existe pas.

Extension d'un fichier à accès direct.

Un fichier ancien peut être augmenté. Vous aviez prévu 300 noms et maintenant il vous en faut 350.

```
!EXTEND"ADR.DTA",175,50
```

Cette commande va inscrire 175 x 50 octets nuls à la fin du fichier ADRESSES. Si, physiquement, la place n'est pas disponible immédiatement après le fichier, le système RANDOS va chercher de la place disponible : cela ne change rien à vos yeux, vous ne vous en rendrez peut-être même pas compte.

Travail octet par octet.
Parfois l'accès à un octet particulier peut être fort utile. Par exemple, pour modifier un texte. Voici un exemple où les espaces sont supprimés. On lit le fichier texte "ADR.DTA" et on le recopie dans CMPACT.TXT, octet par octet, en ne recopiant pas les espaces (ligne 70).

```
10 !FILES 2
20 !OPEN 1,"ADR.DTA",RB
30 !OPEN 2,"CMPACT.TXT",WB
40 POKE 1277,0
50 !GET 1,A
60 IF PEEK(1279)=29 THEN PRINT"FIN": GOTO 90
70 IF A<>ASC(" ") THEN !PUT 2,A
80 GOTO 50
90 POKE 1277,0
100 !CLOSE
```

Il s'agit de fichiers séquentiels. On peut aussi traiter des fichiers à accès direct, dans ce cas écrire DB, mais c'est délicat car la longueur des champs fait partie des données inscrites sur le disque. Il faut veiller à ne pas confondre ces octets avec d'autres ...

Le passage de variables d'un programme à un autre.

Vous pouvez désirer lancer un nouveau programme à partir d'un autre. C'est facile avec LOAD et SAVE en utilisant l'option AUTO. Cependant toutes les variables du premier programme sont perdues lors du chargement du deuxième programme. Il peut se faire que les variables du premier soit nécessaire pour le second. Comment ce passage de variables est-il possible avec le RANDOS ?
STORE et RECALL ont une option M qui sauvegarde et rappelle un tableau en utilisant une zone de la mémoire vive au lieu d'utiliser le disque. Cette commodité peut-être utilisée pour le passage de variables d'un programme un autre :
- sauver le second programme avec l'option AUTO
- dans le premier programme, mettre les variables en tableau. Le réserver en mémoire vive par STORE avec option M et appeler le deuxième programme par LOAD.
- vous aurez écrit dans le second programme un RECALL avec l'option M en n'oubliant pas de dimensionner correctement le tableau. Les variables sont alors disponibles.
Cette technique est très puissante, la seule restriction est que le tableau ne soit pas trop grand, l'espace disponible en RAM doit le suffire.

Commandes de fichier du RANDOS.

!CLOSE pour fermer les fichiers.
!CLOSE [<n>]
<n> est un nombre de 1 à 8 correspondant à celui utilisé avec OPEN. Si aucun numéro n'est indiqué, tous les fichiers sont fermés.
À noter :
1. si un fichier a été ouvert et non fermé, un message d'erreur est généré lors d'une tentative d'ouverture. Il est d'usage d'ouvrir les fichiers au début de programme et de les clore à la fin.
2. un fichier resté ouvert peut être endommagé.
3. il est usuel de clore chaque fichier séparément pour mieux déceler les erreurs.

!CREATE réserve de l'espace sur le disque pour un fichier à accès direct et inscrit son nom dans le répertoire en service.
À noter :
1. Cette commande doit précéder OPEN contenant les options D ou DB. Le nom de fichier doit être différent de ceux existants.
2. En cas de besoin, utiliser EXTEND pour augmenter le nombre de fiches.

!EXTEND sert à augmenter le nombre de fiches d'un fichier à accès direct.
!EXTEND <nom du fichier>,<longueur>,<nombre de fiches à ajouter>

!FILES sert à ouvrir des zones tampons pour chaque fichier
!FILES <nombre de fichiers>
Le nombre de fichier peut aller de 1 à 8

A noter:
1. Ce nombre correspond aux fichiers que l'on ouvrira par OPEN et fermera par CLOSE
2. Cette commande doit précéder les autres (sauf HIMEM)
3. !FILES 1 est facultatif, un tampon étant toujours disponible avec le RANDOS
4. Cette commande clôt tous les fichiers qui seraient restés ouverts

!GET sert à lire les données d'un fichier et à l'affecter à la variable choisie
!GET <n>,<liste de variables>
<n> est un nombre de 1 à 8 correspondant à celui utilisé avec OPEN
<liste de variables> un variable ou plusieurs séparées par des virgules

A noter:
1. La lecture se fait en séquence et les variables sont nourries dans l'ordre
2. Si une variable n'est pas conforme aux exigences du BASIC un message d'erreur est affiché
3. Les variables sont du type chaîne mais pour le travail octet par octet elles sont du type réel

!OPEN ouvre un fichier pour permettre la lecture ou l'écriture
!OPEN <n>,<nom de fichier>, <option 1>[,<option 2>]
<n> est un nombre de 1 à 8. On peut disposer d'un maximum de 8 fichiers accessibles simultanément. Chaque fichier a son propre numéro, c'est celui de du tampon par où transiteront les données.
<option 1>
R pour la lecture
W pour l'écriture
D pour lire ou écrire dans un fichier à accès direct
A pour écrire à la fin d'un fichier séquentiel
Pour l'accès octet par octet, on utilise respectivement RB, WB, DB, AB
<option 2> longueur d'une fiche, en octets, si l'option 1 est D ou DB

A noter:
1. si vous utilisez R, D ou A en option, il faut que le fichier existe dans le répertoire en service.
2. si vous utilisez l'option W, au contraire, le fichier ne doit pas exister, il sera créé
3. cette commande doit précéder GET et PUT

!PUT sert à écrire des données dans un fichier
!PUT <n>,<liste de variables>
<n> est un nombre de 1 à 8 correspondant à celui utilisé avec OPEN
<liste de variables> une variable ou plusieurs séparées par des virgules
Mêmes remarques que pour GET

!RECALL et **!STORE**
même utilisation qu'avec le DOS v1.1 ou V1.0
En plus, option M expliquée dans le passage de variables

!SET utilisé pour placer la tête de lecture/écriture au bon endroit pour lire ou écrire dans un fichier à accès direct
!SET <n>,<numéro de la fiche>
<n> est un nombre de 1 à 8 conforme à celui utilisé dans OPEN
<numéro de la fiche> que l'on veut lire ou écrire

A noter:
1. un message d'erreur sera affiché si l'on cherche à utiliser SET sans avoir choisi l'option D ou DB avec OPEN
2. une erreur sera affichée si l'on cherche à lire une fiche dont le numéro n'est pas compatible avec les déclarations faites par CREATE ou EXTEND

Remarque générale
Pour FILES, OPEN, PUT, GET, OPEN, CLOSE, SET il faut un espace avant d'écrire le numéro qui suit.

Ce document est provisoire et destiné à aider ceux n'ont pas une connaissance suffisante de l'anglais
!HELP 1 2 3 ... 30 affiche à l'écran tous les exemples.