Examen des outils du DOS 4.0

Bruno PETAZZONI

XMA2EMS.SYS : ce *device driver* permet d'exploiter une mémoire EMS. Mettre dans le fichier CONFIG.SYS la ligne :

```
DEVICE = XMA2EMS.SYS {FRAME=adr} {Pn=adr} {/X:n}
```

Les éléments entre accolades sont optionnels. `FRAME=` permet de définir la zone de mémoire conventionnelle utilisée comme cadre, en général la valeur de adr sera D000. Si l'on ne dispose pas de 64 K consécutifs, on utilisera un ou plusieurs paramètres `Pn=adr` où n est un numéro de page (entre 0 et 3) et adr une adresse de segment (par exemple `P0=C000` et `P1=C400`). De plus, `P254` doit être précisé si l'on veut que FASTOPEN exploite la mémoire EMS ; et `P255` doit être précisé si l'on veut que BUFFERS et/ou VDISK exploitent cette mémoire. `/X:n` permet de préciser la taille (en pages de 16 Ko) que l'on veut exploiter, à défaut toute la mémoire EMS sera utilisée. Note pour ceux qui veulent briller en société : on ne doit pas lire *XMA-deux-EMS* mais *XMA-two-EMS* en le prononçant *XMA-to-EMS* ; exactement comme dans *EXE2BIN* : le *2* ne signifie pas *deux* mais *vers*. Astuce (de la même eau : T42 se lit *tea for two*).

XMAEM.SYS (ou EMM386.SYS) : ce *device driver* permet, sur un ordinateur équipé d'un 386, de simuler l'EMS dans de la mémoire étendue. Mettre dans le fichier CONFIG.SYS la ligne :

```
DEVICE = XMAEM.SYS {n}
```

Le paramètre optionnel n précise comme pour XMA2EMS le nombre de pages de 16 Ko à réserver (par défaut, la totalité). Cette déclaration doit être placée avant celle de XMA2EMS.

BUFFERS : commande de configuration bien connue, qui admet désormais le paramètre `/X`, permettant de loger les buffers dans l'EMS, à condition d'avoir précisé `P255` dans la déclaration de XMA2EMS.

VDISK.SYS : ce *device driver* (maintenant baptisé RAMDRIVE.SYS) permet de créer un disque virtuel. L'option `/E` permet

de le loger dans la mémoire étendue, l'option /X (ou /A sur certains systèmes, consulter votre documentation), de le loger dans l'EMS. Ne pas oublier dans ce dernier cas de préciser P255 dans la déclaration de XMA2EMS, et /X dans la déclaration de BUFFERS.

SMARTDRV.SYS : encore un *device driver*. Il permet d'exploiter tout ou partie de la mémoire étendue ou EMS, comme une mémoire *cache*, pour accélérer l'accès aux secteurs des disques *durs*. Placer dans CONFIG.SYS la ligne :

```
DEVICE = SMARTDRV.SYS {n} {/A}
```

où n est la taille du cache en Ko (384 par défaut). Le paramètre /A permet de loger le cache en mémoire EMS, à condition d'avoir au préalable déclaré XMA2EMS.SYS.

FASTOPEN : ce programme permet d'accélérer notablement l'accès aux fichiers, en mémorisant les noms des derniers fichiers auxquels on a accédé, ainsi que les informations permettant de localiser les blocs d'espace disque qui leur sont alloués. On évite ainsi, lors de chaque ouverture, de relire répertoire(s) et table d'allocation (d'où le nom de cet outil). L'option /X permet de loger ces informations dans l'EMS, à condition que l'on ait précisé P254 dans la déclaration de XMA2EMS.

Pour fixer les idées, voici un exemple de fichier de configuration. On suppose que l'ordinateur est un AT, disposant d'un Mo de mémoire étendue. On en utilise 512 Ko pour loger un disque virtuel, 384 Ko pour loger une mémoire cache. Le reste suffit largement pour 40 buffers et FASTOPEN :

```
DEVICE = XMA2EMS.SYS FRAME=D000 P254=D000 P255=D000 /X
DEVICE = VDISK.SYS 512 128 64 /X
DEVICE = SMARTDRV.SYS 384 /A
BUFFERS = 40 /X
```

On mettra dans AUTOEXEC.BAT la ligne suivante :

```
FASTOPEN C:=(100,500)
```

qui mémorisera les chemins d'accès aux cent derniers fichiers, et les indications d'emplacement pour 500 blocs.

DRIVPARM ET DRIVER.SYS

Encore un point sur lequel les documentations Microsoft manquent de clarté. Quand aux bouquins de provenances diverses, ils sont carrément consternants.

DRIVPARM est une commande de configuration, qui doit donc apparaitre dans CONFIG.SYS, et sert à *modifier* les paramètres d'une unité. La syntaxe en est :

```
DRIVPARM = /C /D:d /F:f /H:h /N /S:s /T:t
```

Seul le paramètre `/D:d` est obligatoire, il spécifie le numéro physique de l'unité que l'on veut paramétrer ; d vaut 0 pour le premier lecteur de disquettes, 1 pour le deuxième, etc, 128 pour le premier disque fixe, etc. `/C` indique que l'unité est équipée d'un dispositif de détection d'ouverture de la porte, `/N` que le support n'est pas amovible. `/H`, `/S` et `/T` spécifient respectivement le nombre de têtes (faces pour une disquette), secteurs par piste, et pistes par face. `/F` précise le type de l'unité. Par défaut, `DRIVPARM = /D:0` indique que le lecteur de disquettes numéro 0 (que le DOS considèrera comme A:) est un 3,5 pouces de 720 Ko.

DRIVER.SYS est un *device driver* qui doit donc être déclaré dans CONFIG.SYS par une commande DEVICE. Syntaxe :

```
DEVICE = DRIVER.SYS /C /D:d /F:f /H:h /N /S:s /T:t
```

Les paramètres ont la même signification que pour DRIVPARM (c'est heureux). Chaque déclaration de DRIVER.SYS (il peut y en avoir plusieurs) ajoute une nouvelle unité DOS, il faut donc au besoin changer LASTDRIVE. Alors que DRIVPARM ne prend pas de place en mémoire, DRIVER.SYS s'insère dans la liste chaînée des *device drivers* et va donc consommer un peu de mémoire.

Examinons deux cas de figure intéressants. Tout d'abord, un PC bête, avec un lecteur 5,25 pouces, un disque fixe, et un lecteur (externe ou pas) 3,5 pouces 720 Ko. Mettez dans CONFIG.SYS la commande `DRIVPARM=/D:1`, après avoir vérifié deux détails matériels :

* les switches du PC doivent être configurés pour indiquer deux lecteurs de disquettes
* sur la carte électronique du lecteur 3,5 pouces, le cavalier de sélection doit être sur DS1 (si la numérotation commence à DS0, ce qui est le cas en général)

Si vous ne mettez pas DRIVPARM=/D:1 vous pourrez utiliser le lecteur 3,5 pouces, mais seules les 40 premières pistes seront exploitées, puisque le DOS considèrera, par défaut, qu'il s'agit d'un lecteur 5,25 pouces 360 Ko.

Deuxième cas de figure. Un AT avec un lecteur 5,25 pouces et un lecteur 3,5 pouces. Le premier est vu comme A: (et c'est vraisemblablement un lecteur 1,2 Mo), le deuxième est vu comme B:. La copie de disquette (sans changer de format!) est faisable, mais la copie de fichiers ne l'est plus : mettez donc dans CONFIG.SYS la déclaration :

```
DEVICE = DRIVER.SYS /D:0 /F:1
```

Désormais, vous disposez d'une unité C: (ou D: si la machine est équipée d'un disque fixe). Vous pouvez donc copier des fichiers de A: vers C:, le DOS se chargera de vous demander de changer de disquette au moment opportun. Un conseil : au lieu de COPY, utilisez donc XCOPY : il est beaucoup plus rapide, et comme il lit plusieurs fichiers d'un coup, vous aurez moins de manipulations de disquettes.

Vous pouvez bien entendu mettre aussi :

```
DEVICE = DRIVER.SYS /D:1 /F:2
```

pour gérer une unité D: (ou E: s'il y a un disque fixe) qui sera logée sur le lecteur 3,5 pouces.

Une expérience que j'ai réalisée lors d'un stage : récupérez un lecteur 5,25 pouces 80 pistes (comme ceux qui équipaient les 9020 et le Sil'Z 16, il doit bien en traîner un quelque part). Placez le cavalier sur DS1 (comme rappelé plus haut). Ouvrez un PC à deux lecteurs de disquettes, déconnectez et démontez le deuxième lecteur et remplacez-le par le 80 pistes. Préparez une disquette système avec, dans CONFIG.SYS, la commande :

```
DRIVPARM = /D:1 /F:0 /T:80
```

Amorcez avec cette disquette et formatez une disquette dans le lecteur B:. Surprise, vous avez 720 Ko. A priori, il est préférable d'utiliser dans ce lecteur des disquettes certifiées 96 tpi au lieu des 48 tpi habituelles (disquettes qui étaient qualifiées en leur temps de *quad density*). Evidemment, la manip fonctionne aussi sur un PC ne disposant que d'un lecteur: il faut simplement penser à régler les switches sur la carte mère. Seul problème : je ne sais pas comment faire pour lire et

écrire des disquettes 360 Ko sur un tel lecteur ; c'est certainement possible, puisque le Sil'Z 16 le faisait parfaitement.

Bruno PETAZZONI