MÉMOIRE DILUÉE...

Bruno PÉTAZZONI

Une incroyable confusion régnant à ce sujet, il est temps de mettre au clair les notions de mémoire étendue, EMS, LIM 4.0 et autres babioles.

Le lecteur est supposé maîtriser les concepts d'*adresse*, de *segment* et d'*offset* sur la famille Intel 86, ainsi que le mécanisme de conversion d'une adresse *logique* en adresse *physique* (on trouvera un bref rappel sur ces points en fin d'article).

Pour commencer, un examen des possibilités d'adressage de la famille Intel '86.

Les 86 et 88 (qui activent les compatibles PC/XT) possèdent vingt lignes d'adresse, et sont donc capables d'adresser *directement* 1 Mo. Les 286 et 386SX possèdent 24 lignes, l'espace *physique* adressable passe à 16 Mo ; enfin, les 386 et 486 possèdent 32 lignes d'adresse, ce qui porte l'espace adressable à 4 Go. Il faut noter que, du 286 au 486, l'espace *virtuel* adressable est nettement plus vaste : 1 Go par tâche sur le 286, 64000 Go par tâche sur les 386 et 486...

Dans l'architecture PC/XT, les 640 Ko allant de 00000 à 9FFFF peuvent être occupés par de la RAM, que nous qualifierons de mémoire *conventionnelle contiguë*. Les 128 Ko suivants, de A0000 à BFFFF ont été réservés pour les mémoires d'écran. Enfin, de C0000 à FFFFF, on trouvera alternativement des mémoires mortes (le BIOS tout à la fin, des extensions du BIOS pour gérer le disque dur ou la carte EGA/VGA vers le milieu) et des trous.

Dis pépé, c'est quoi EMS ?

L'idée de base des cartes EMS, promues par l'association Lotus-Intel-Microsoft (d'où le nom de LIM), consiste à réserver une zone de 64 Ko dans l'un de ces trous ; à découper cette zone en quatre *fenêtres* de 16 Ko chacune ; à découper la mémoire présente sur la carte en *pages*, de 16 Ko également. Une électronique située sur la carte permet d'appliquer chacune des fenêtres sur n'importe quelle page. Bien entendu, la carte

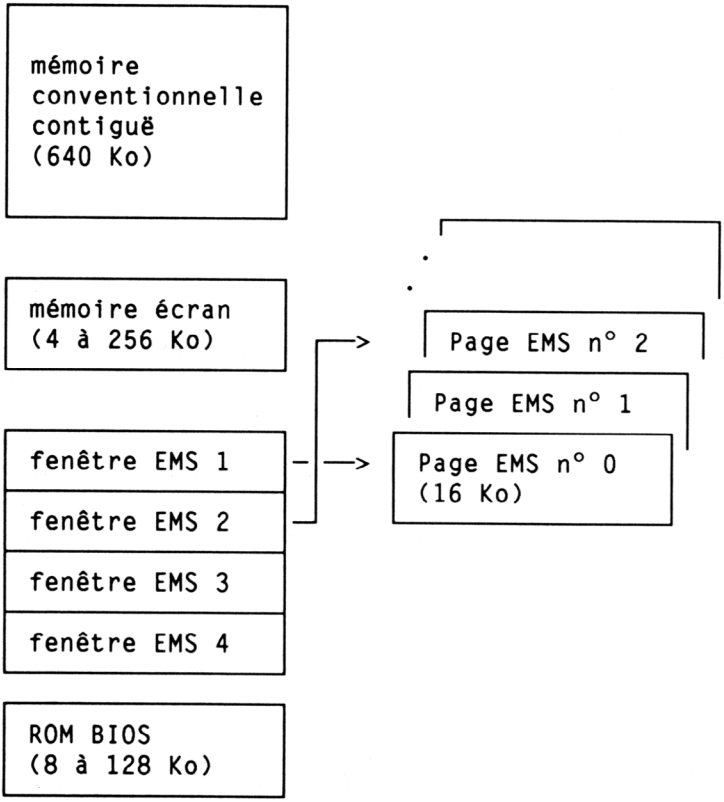
peut contenir beaucoup plus de 64 Ko : jusqu'à 32 Mo dans la version 4.0 de LIM.

On retrouve un mécanisme bien connu, celui de la *commutation de banque* (bank switching), souvent implanté sur des machines 8 bits comme l'Apple][, ou, pour le matériel E.N., sur le Sil'Z II (signalons aux ahuris de service dans certaines revues qu'il ne s'agit pas exactement de *pagination* ; le terme *mémoire paginée*, employé en particulier dans les documentations Microsoft est tout juste adéquat). Dans EMS, le **E** signifie *expanded*, ce qui se traduit assez mal en français par *expansé* (on a tendance à penser à du polystyrène), en tout cas ce n'est pas de la mémoire *étendue* (cf. la revue de la presse nulle plus loin) ; on peut parler de *mémoire d'expansion*, mais ce n'est pas très gracieux.

Une remarque à propos de LIM : on voit régulièrement dans la presse des plumitifs de bas étage parler de la *norme* LIM. Hélas pour eux, LIM est un *standard* industriel, mais pas une norme : aucun document émanant d'un organisme de normalisation ne décrit LIM, et personne n'est *obligé* de fabriquer des cartes ajoutant de la mémoire selon les prescriptions du trio LIM, même si Microsoft a un peu trop tendance à se poser en pondeur de normes à tout va.

Les premières versions d'EMS n'offraient l'accès qu'à travers une fenêtre. Depuis EEMS (Enhanced EMS, promue par un autre triumvirat contre nature, unissant AST, Quadram et l'inénarrable Ashton-Tate), et LIM 4.0 (fulgurante réponse du premier trio au second), on dispose de quatre fenêtre, et on peut les appliquer à n'importe quel endroit de l'espace adressable.

Un bon dessin valant mieux qu'un long discours, voici en gros ce qu'il y a en mémoire :



Dans cet exemple, la fenêtre EMS numéro 1 est appliquée sur la page EMS numéro 0, et la fenêtre EMS numéro 2 sur la page EMS numéro 3. Les deux flèches matérialisent cette association, que les américains appellent *mapping*. On pourrait traduire par *topographie*.

Pour le programmeur, l'accès à l'EMS se fait par le biais de l'INT 67H, modulo un pilote EMM.SYS (**E**xpanded **M**emory **M**anager), livré avec la carte, et qui met à sa disposition une tripotée de fonctions, pour allouer ou désallouer des pages, savoir combien de pages sont libres, etc.

Beaucoup de programmes savent exploiter EMS ; citons :

- le tableur *Lotus 123*, pour loger une feuille de calcul gigantesque
- l'environnement *Windows*, pour stocker des portions de programme dont on n'a pas besoin pour l'instant
- *Turbo Pascal 5.5*, qui y place le texte source du programme, s'il devient trop gros

Si l'on manque d'imagination, on peut toujours utiliser EMS pour un disque virtuel (qui n'empiètera pas sur la mémoire conventionnelle, mais gardera un temps d'accès très faible, comparé à un disque dur). Je signale que ce n'est pas avec le VDISK.SYS usuel du DOS 3.3, avec option /E que ceci marchera : il faudra utiliser un gestionnaire fourni (sur disquette) avec la carte, ou encore un VDISK.SYS admettant l'option /A (comme Above?) ; ou alors, passer à DOS 4.0 ou 4.1 et utiliser RAMDRIVE.SYS (pourquoi ce changement de nom?), qui accepte /E (extended) et /A (expanded). Autres suggestions d'utilisation : un spoule (gestionnaire d'impression en différé) qui exploitera EMS comme un gigantesque tampon, ou encore SMARTDrive (voir plus bas au sujet de ce dernier).

Et la mémoire étendue, alors ?

Examinons maintenant l'architecture AT : si l'on travaille en mode protégé (ce qui est le cas sous OS/2 mais pas sous DOS), les 24 lignes d'adresse sont utilisées. On peut donc gérer *directement* jusqu'à 16 Mo de mémoire. Physiquement, la mémoire qui se trouve à partir de l'adresse physique 100000 est dite *étendue* (en anglais : *extended*). A priori, elle n'est pas utilisable quand on est en mode réel, mais il existe une astuce d'architecture qui permet d'en utiliser les 65520 premiers octets. Elle repose sur une petite différence de fonctionnement entre les 86/88 et le 286 : sur le premier, l'adresse logique FFFF:0010 correspond à l'adresse physique 00000 (il n'y a pas de report au-delà du vingtième bit, dans l'unité d'adressage des 86/88). Sur le 286, la même adresse logique correspond par contre à l'adresse physique 100000.

Les concepteurs de l'AT ont mis en place une électronique qui permet de valider ou non la ligne d'adresse A20. Si elle n'est pas validée, l'adresse physique 100000 sortant du 286 arrivera comme 000000 sur le bus, et tout se passera comme si l'on avait un 88. Si par contre on valide cette ligne, on va pouvoir émettre sur le bus les adresses physiques 100000 à 10FFEF, qui correspondront aux adresses logiques FFFF:0010 à FFFF:FFFF. Ceci, sans avoir à sortir du mode réel, bien entendu. Cette zone de mémoire sera appelée *mémoire haute* (HMA ou High Memory Area, outre-Atlantique).

Microsoft, Lotus, AST et Intel ont rédigé en 1988 une *spécification de mémoire étendue* (XMS, eXpanded Memory Specification) qui permet aux applications sous DOS d'exploiter en bonne harmonie la zone HMA, aussi bien que le reste de la mémoire étendue (dont on parlera plus loin)

ou la zone *upper memory* (située au-dessus de 640 Ko mais en dessous de 1 Mo).

Windows et DOS 4 sont maintenant livrés avec un utilitaire HIMEM.SYS (comme **HI**gh **MEM**ory) qui réalise l'interface XMS. Il semble qu'à ce jour, seul *Windows* sache exploiter la zone HMA. A noter : HIMEM n'a pas été conçu par Microsoft, comme on pourrait le penser, mais par Qurterdeck, pour son environnement *DESQview*.

Le programmeur curieux prendra conscience du fait que l'on adresse cette zone en activant la ligne A20 (jeu sur un port d'E/S), et en utilisant une adresse logique constituée du segment FFFF et d'un offset compris entre 0010 et FFFF. Le fait que l'offset ait une valeur minimale de 0010 interdit de loger dans HMA un programme DOS, puisque celui-ci a besoin d'informations logées aux offsets 0000 à 00FF (le fameux PSP, ou Préfixe de Segment Programme).

L'accès au reste de la mémoire étendue est possible indirectement, depuis le mode réel : en effet, IBM a placé dans le BIOS de l'AT deux nouvelles fonctions, dans le traitement de l'INT 15H, autrefois dévolu à la gestion du lecteur de cassettes (oui, vous avez bien lu). L'une des fonctions permet de connaître la taille de la mémoire étendue ; l'autre de copier un bloc d'un endroit à l'autre de la mémoire visible en mode protégé, donc aussi bien conventionnelle qu'étendue (ces fonctions sont utilisées par VDISK.SYS et RAMDRIVE.SYS avec option /E).

Malheureusement, le BIOS ne fournissait pas de primitives pour allouer et désallouer de la mémoire. IBM lui-même manquait de cohérence, en utilisant dans VDISK.SYS une technique d'allocation « par le bas », et dans IBMCACHE.SYS une technique d'allocation « par le haut ». Cette lacune et ces incohérences justifient la mise au point de XMS.

SMARTdrive : j'en ai entendu causer

SMARTdrive, c'est tout simplement un logiciel livré par Microsoft avec *Windows* et DOS 4, et capable d'exploiter au choix une mémoire étendue ou une mémoire EMS, pour en faire une antémémoire (ou cache-disque) susceptible d'améliorer les performances de *Windows* (ce qui n'est pas un luxe sur certaines bécanes). Voir description détaillée dans l'annexe y afférente du *Guide d'Utilisation* de *Windows* ou du *Manuel de référence* du DOS. Un simple détail : on y parle abondamment de mémoire *paginée*, il faut bien évidemment comprendre *EMS*.

Compléments de programme

Comprenons bien que ni EMS ni XMS ne permettent d'augmenter la mémoire conventionnelle contiguë. Or c'est dans celle-ci qu'il est le plus facile de travailler. En particulier, DOS ne sait allouer de mémoire (pour charger un programme, par exemple, ou à la demande de celui-ci lors de son exécution) que dans de la mémoire conventionnelle contiguë.

De petits futés ont noté que, comme la plupart des PC n'utilisaient pas en fait la zone A0000 à AFFFF, on avait là un moyen d'ajouter immédiatement 64 Ko de mémoire conventionnelle contiguë. On peut même aller encore un peu plus loin : souvent, la carte écran utilisée est une EGA ou une CGA, dont la mémoire commence en B8000, ce qui permet d'étendre à B7FFF la mémoire conventionnelle ; le gain total, cette fois, est de 96 Ko. On trouve effectivement de telles cartes sur le marché, mais leur prix est prohibitif : par exemple la Hi Card 2 est commercialisée 3900 francs H.T. pour 256 Ko ; la carte Maxit est à 295 $, ce qui doit la mettre dans la même zone, avec le dollar informatique à 13 francs...

Pour ce prix, on peut se payer une *vraie* carte EMS : par exemple, BOCARAM/AT Plus, à 2450 FHT équipée de 2 Mo ; ou encore Everex EMS AT, à 1390 FTTC nue. On en profitera pour récupérer un *shareware* baptisé EEMRAM, qui permet d'assigner une partie de l'EMS dans la zone A0000 à AFFFF (voire B7FFF si votre écran est CGA). Solution bien plus élégante, comme c'est souvent le cas des solutions logicielles.

Les cartes d'extension mémoire les plus futées sont évidemment celles pouvant fonctionner aussi bien comme extension de mémoire conventionnelle, que comme EMS ou mémoire étendue (avec possibilité de partage en deux zones, l'une exploitée en EMS, l'autre en étendue).

Vous avez dit virtuel ?

Profitons-en pour parler d'une créature étrange, *Above Disc*. Il s'agit d'un programme qui simule une mémoire EMS dans un disque dur. Surprenant au premier abord, puisque fonctionnant exactement en sens inverse d'un disque virtuel. L'idée est la suivante : lors de la conception de ce produit, la RAM était chère. Par contre, les prix des disques durs étaient en chute libre. D'où l'idée : permettre l'exploitation, par exemple sous *Lotus 123*, d'une grande quantité de mémoire, logée en fait sur le disque dur. Les performances ne sont évidemment pas celles d'une vraie carte EMS, mais le prix se réduit à celui du logiciel, si vous avez de

l'espace disponible sur votre disque. Pas bête, mais en voie d'obsolescence rapide pour cause de chute des tarifs chez NEC, TOSHIBA, OKI et autres grands producteurs de puces. Une remarque amusante : on devrait pouvoir lancer Above Disc, puis dans la foulée y installer un disque virtuel. On aboutirait ainsi à l'étrange situation où un disque *virtuel* est logé dans un disque *réel*... Signalons pour clore ce paragraphe qu'Above Disc peut aussi simuler l'EMS dans de la mémoire étendue, et qu'il possède des concurrents, TC!Power et Turbo-EMS.

La tournée des popotes

Un bref sondage dans la grande banlieue montre une grande diversité de situations. Ainsi, tel compatible AT (disons qu'il est de la marque *Craffougnat*, pour ne pas provoquer de larmes chez certains constructeurs) est livré avec 1 méga, mais il est *rigoureusement* impossible d'utiliser les 384 Ko situés au-delà de la mémoire conventionnelle contiguë. Tel autre est également livré avec 1 méga, mais *personne* n'a réussi à loger ne serait-ce qu'un malheureux VDISK dans les 384 Ko inutilisés. Dernier exemple en date : une superbe chose à base de 386 avec 2 Mo de RAM ; si l'on installe EMM386.SYS, la combinaison de touches Ctrl+Alt+Del plante le bestiau purement et simplement au lieu de le redémarrer comme à l'accoutumée.

Un bon conseil avant achat : demandez à maniper sur l'objet, pour vérifier que vous pouvez *effectivement* exploiter le supplément de RAM. N'oubliez pas que ce n'est pas encore demain que vous tournerez sous OS/2, et qu'il vaut donc mieux s'assurer que l'on peut compter sur des solutions du moment (même si elles relèvent de l'emplâtre sur une jambe de bois).

Il nous resterait à parler des apports de DOS 4, du 386, de la pagination (la *vraie*), du mode virtuel 86, des divers *DOS extenders* censés briser la fatidique barrière des 640 Ko. Ce sera pour une autre fois. Un simple détail pour finir : je garde un excellent souvenir du Sil'Z 16, ordinateur au clavier remarquable, à l'écran superbe (avec des caractères redéfinissables dans une matrice 8x14), et surtout conçu avec une mémoire contiguë de 1 Mo, sans trou (ni bosse). Avec son PC, IBM a certes fourni une base stable pour le développement de logiciels, mais a totalement *tué* les initiatives en matière d'architecture de micro-ordinateur à base de 86/88.

Le coin des glousseurs

Cognons un peu sur les incompétents pour finir : un article paru dans le numéro 5 de la revue *DEBUG* et intitulé *DOS et mémoire étendue* présente le standard EMS, et affirme dans la foulée que le BIOS de l'AT permet d'accéder à cette mémoire. Méchante confusion dans l'esprit de l'auteur, qui nous présente plus loin un *mini-gestionnaire de mémoire dynamique en RAM étendue* : accrochez les wagons, la langue de bois a toujours cours dans le canton !

Dans le genre nul et marrant, un article paru dans le numéro 30 de la *Revue de l'Utilisateur de l'IBM-PC* évoque les 384 Ko non adressés *par le dos*. Sans doute parce qu'ils sont adressables *par le ventre*. Le même article, présentant la gestion des mémoires EMS, utilise à tour de bras l'expression *mémoire étendue* : le chapeau de présentation précise même que *Expanded Memory Spécification* (avé l'assent) signifie *Spécification de Mémoire Etendue*. Une simple question : comment le traducteur appointé par la *Revue gnagnagna...* traduit-il *extended* ?

Beaucoup plus gênant, l'article *Mémoire étendue* du numéro de Mars 1990 de *Pascalissime*. D'habitude, on ne peut reprocher à cette revue qu'une orthographe consternante. L'auteur commence par affirmer que les XT sont *souvent* dotés d'une mémoire étendue, et enchaîne en signalant que seules les machines basées 286 ou 386 peuvent accéder à la mémoire étendue : il faudrait s'entendre sur la signification de XT. Ensuite, il dit ne pas disposer des spécifications du VDISK (curieusement calligraphié VDISC) : alors que le listing source en était gracieusement fourni par IBM avec les DOS 3.1 et 3.2 ; lesquels, sans être du domaine public, sont quand même des produits de référence.

Le rappel promis en tête d'article

Sur les microprocesseurs 86/88, comme sur les 286 et 386 lorsque ces derniers fonctionnent en mode *réel*, une adresse *logique* est composée d'un *segment*, sur 16 bits, et d'un *offset*, également sur 16 bits. Si le segment est XXXX et l'offset YYYY (en hexadécimal), on note XXXX:YYYY cette adresse logique. La conversion d'une adresse logique en adresse *physique* se fait en collant un zéro à droite du segment et en l'ajoutant à l'offset. Ainsi, l'adresse logique 47B3:26C2 devient 47A30+23C2 soit 49DF2.

A venir...

La suite de cet article examinera les commandes du DOS liées à la gestion de la mémoire : BUFFERS, FASTOPEN, VDISK, SMARTDRV, XMAEM et XMA2EMS, etc.

Bruno PETAZZONI