COS 2000

Compatible **O**perating **S**ystem

mailto:cos2003@free.fr

Présentation

COS2000, par définition, est système d'exploitation. Celui-ci prend la direction des opérations à partir du moment où le PC est mis sous tension (Après le BIOS). Il gère tous les périphériques rattachés au PC et offre aux programmeurs les moyens de développer des applications compatibles en fournissant des APIs (Application Programming Interface). COS2000 est basé sur un concept particulier qui est d'offrir aux programmeurs un maximum de fonctions intégrées pour faciliter le travail des programmeurs et réduire la taille des programmes.

Comment l'installer ?

Pour installer COS2000 :

Sous dos/windows 9x

- Insérez une disquette 1.44 Mo vierge ou inutile dans votre lecteur.

- Lancez le programme SETUP.COM situé dans le dossier de COS2000.
- Si celui-ci ne détecte pas d'erreur, COS2000 est installé !

Sous windows NT/Xp

- Insérez une disquette 1.44 Mo vierge ou inutile dans votre lecteur.
- Lancez le programme SETUP.EXE situé dans le dossier de COS2000.
- Si celui-ci ne détecte pas d'erreur, COS2000 est installé !

Sous Linux

- Insérez une disquette 1.44 Mo vierge ou inutile dans votre lecteur.
- Lancez le programme SETUP.SH situé dans le dossier de COS2000.
- Si celui-ci ne détecte pas d'erreur, COS2000 est installé !

Pour lancer COS2000 :

- Insérez la disquette où COS2000 est installé.
- Veillez que dans le BIOS vous puissiez démarrer à partir de A:.
- Redémarrer votre ordinateur et vous serez sur COS2000.

Mode d'emploi

L’interpréteur de commande COS est le premier logiciel qui est lancé au démarrage. A partir de celui-ci vous pouvez exécuter quelques commandes ou logiciels.

En plus des logiciels, l'interpréteur de commandes peut exécuter 6 commandes :

QUIT
Quitte l'interpréteur.

VERS
Donne la version de COS2000.

CLEAR
Efface l'écran.

REBOOT

Redémarre le PC.

CMDS
Donne la liste des commandes disponibles.

MODE [mode]
Permet de changer de mode vidéo. [mode] doit être un entier compris entre 1 et 9. les modes au delà de 4 sont des modes graphiques à texte émulé. Il est déconseillé de les utiliser car il est parfois impossible de revenir aux modes texte.

DISK
Permet de lire un support disquette 1.44 Mo au format FAT12.

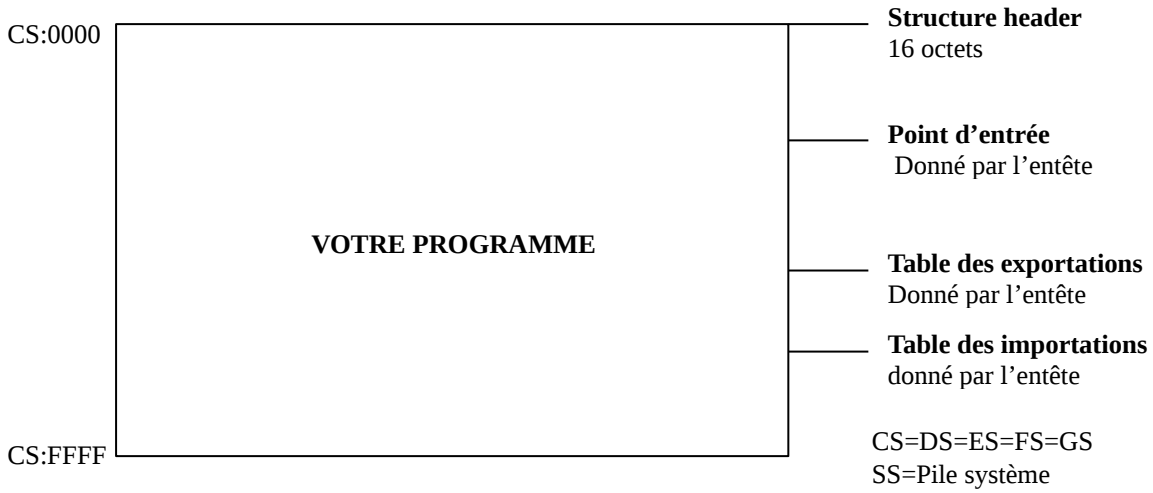
CD
Change le dossier actuel vers celui spécifié.

DIR
Permet de lister le contenu du dossier actuel.

MEM
Permet de lister le contenu du dossier actuel.

Faire un programme pour COS

Toute contribution à COS 2000 en terme de programme est la bienvenue, un répertoire « contribs » contiendra les programmes des différents contributeurs. Aucune modification a ceux-ci ne sera faire sans l'accord explicite de l'auteur. Pour une contribution écrivez moi a l'adresse mailto:cos2003@free.fr.



Quelques macros on étés rajouté dans le fichier « mem.h » pour facilité la tache du programmeur :

Mots clé	**Definition**
heading	L'entête
importing	Début de la table d'importation
endi	Fin de la table d'importation
exporting	Début de la table d'exportation
ende	Fin de la table d'exportation
declare	Déclaration d'un export
use	Déclaration d'un import
noexporting	pas de table d'import
noimporting	pas de table d'export

Pour clore le programme il suffit de faire un retour far.
Exemple avec un Hello Word.

Avec tasm

```
.model tiny,stdcall      ;model tiny (.com)
                         ;convention appel windows
.486                     ;Pour processeur 80486
locals                   ;Labels locals
jumps                    ;Optimisations des sauts
.code                    ;Segment de code
ideal                    ;Mode ideal

org 0h                     ;début du programme en 0h

start:

heading   1,0,offset beginning ; Point d'entrée en start
                               ; version 1.0

noexporting              ;pas d'exportation de fonction
importing                ;début des importations
use VIDEO.LIB,print      ;importe la fonction print
endi                     ;fin des importations


include mem.h            ;include le fichier
                         ;de gestion mémoire

beginning:
     call [cs:print],offset msg
     retf

msg db 'Hello World',0

end start
```

Avec lzasm

```
model tiny,stdcall       ;model tiny (.com)
                         ;convention appel windows
p486                     ;Pour processeur 80486
locals                   ;Labels locals
jumps                    ;Optimisations des sauts
codeseg                  ;Segment de code
options procalign:byte   ;Pas d'alignement du code

org 0h                     ;début du programme en 0h

heading   1,0,offset start ; Point d'entrée en start
                           ; version 1.0

noexporting              ;pas d'exportation de fonction
importing                ;début des importations
use VIDEO.LIB,print      ;importe la fonction print
endi                     ;fin des importations


include mem.h            ;include le fichier
                         ;de gestion mémoire

start:
     call [cs:print],offset msg
     retf

msg db 'Hello World',0
```

Comme vous pouvez le constater l'appel des APIs de Cos se réalise par le biais d'appels far de convention stdcall.

Liste des APIs

disque.sys : Gestionnaires FAT12 et Disquette
SYSTEME

readsector
writesector
verifysector
initdrive
loadfile
compressrle
decompressrle
findfirstfile
findnextfile
getfreespace
searchfile
getname
getserial
changedir
readcluster
writecluster
getdir

projfile
execfile

port.sys : Gestionnaires port parallèle

Aucune dépendance

getlptin
getlptout
getlptinout
setlptin
setlptout
setlptinout
getlpt
getfirstlpt
setemettor
setreceptor
settimeout
gettimeout
receivelpt
sendlpt
receivelptblock
sendlptblock
receivecommand
sendcommand

souris.sys : Gestionnaires souris

Aucune dépendance

cmdmouse
cmdmouse2
detectmouse
getmouse
getmousescreen
configmouse

detect.lib : Gestionnaires PCI, PNP et CPU

Aucune dépendance

enableirq
cpuinfo
setinfo
pciinfo
getpciclass
getpcisubclass
getcardinfo
pcireadbyte
pcireadword
pcireaddword
detectvmware

SYSTEM : Gestionnaires mémoire, interruptions, divers

Aucune dépendance

enableirq
disableirq
readmaskirq
readirr
readisr
seteoi

mbinit
mbfree
mbcreate
mbresident
mbnonjresident
mbloadfuncs
mbsearchfunc
mbget
mbfind
mbchown
mballoc
mbclean
mbfindsb
bioswaitkey
biosprint
biosprinth
flatmode
enablea20
disablea20

VIDEO : Gestionnaires de la carte vidéo

SYSTEME

setvideomode
getvideomode
clearscreen
setfont
loadfont
getfont
addline
showchar
setcolor
getcolor
scrolldown
getxy
setxy
savescreen
restorescreen
xchgPages
savepage
restorepage
waitretrace
waithretrace
getvgainfos
savedac
restoredac
savestate
restorestate
enablescroll
disablescroll
getchar
enablecursor
disablecursor

video.lib : Librairie d’affichage formaté

VIDEO

print
showint

showsigned
showhex
showbin
showstring
showstring0
showdate
showtime
showname
showattr
showsize
showbcd
showintr
showintl

Les possibilités de COS2000 sont aujourd'hui très limitées car il est en cours de développement.

Détail des APIs

Le format de sortie est normalisé sous la forme suivante :

NULL →	/
BYTE →	AL
WORD →	AX
DWORD→	EAX
PTR→	DX :AX
PACKET→	AL,AH

*null Readsector(word sector)

Lit le secteur spécifié et le met en ES:DI. Met le flag carry à 1 si erreur.

*Writesector

Ecrit le secteur CX avec les données pointés par DS:SI. Met le flag carry à 1 si erreur.

*Verifysector

Vérifie si le secteur CX n'est pas défectueux. Met le flag carry à 1 si erreur et flag equal à 0 si secteur défectueux.

*Initdrive

Fonction initialisant le pilote et le matériel afin d'utiliser ultérieurement les fonctions de disque.sys. Met le flag carry à 1 si erreur.

*Loadfile

Charge le fichier dont le nom est pointé par DS:SI en mémoire dans ES:DI et renvoie le nombre d'octets lu en ECX. Met le flag carry à 1 si erreur.

*Compressrle

Compresse le contenu de la mémoire pointé par DS:SI (selon une méthode RLE) et dont la taille est spécifié par CX. Le résultat sera mis en ES:DI ainsi que la nouvelle taille mémoire (octets) en BP.

*Decompressrle

Décompresse le contenu de la mémoire pointé par DS:SI (selon une méthode RLE) et dont la taille est spécifié par CX. Le résultat sera mis en ES:DI ainsi que la nouvelle taille mémoire (octets) en BP.

*Findfirstfile

Renvoie en ES:DI la première entrée du répertoire courant (format BRUT). Met le flag carry à 1 si erreur. Cette fonction prépare aussi l'usage de la fonction findnextfile.

Format d'une entrée de répertoire cf structure Entries

*Findnextfile

Renvoie en ES:DI l'entrée suivante du répertoire courant (format BRUT). Met le flag carry à 1 si erreur.

*Getfreespace

Renvoie en EDX l'espace disque libre du volume en octets. Met le flag carry à 1 si erreur.

*Searchfile

Renvois dans ES:DI l'entrée de répertoire du fichier pointé par DS:SI. Met le flag equal a 0 si pas existant. Met le flag carry à un si erreur.

*Getname

Renvois dans ES:DI le nom du support courant.

*Getserial

Renvois le numéro de série du support courant en EDX.

*Changedir

Change le répertoire courant a celui dont le nom est pointé par DS:SI. Met le flag carry à un si erreur.

*Readcluster

Lit le cluster (groupe) CX et le met en ES:DI. Met le flag carry à 1 si erreur.

*Writecluster

Ecrit le cluster (groupe) CX avec les données pointés par DS:SI. Met le flag carry à 1 si erreur.

*Getdir

Renvoie en ES:DI sous forme de chaîne a zéro terminal le nom du répertoire courant.

*Projfile

Charge le fichier dont le nom est pointé par DS:SI dans un bloc mémoire. Renvoie en ECX le nombre d'octets lus et en ES l'adresse du bloc de mémoire. Met le flag carry à 1 si erreur.

*Execfile

Exécute le fichier dont le nom est pointé par DS:SI. Met le flag carry à 1 si erreur.

Cpuinfo(word *cpu)

Renvoie la structure cpu dans l'adresse « cpu » qui contient des informations sur le processeur N°1

Setinfo(word *cpu, word *pointer)

Renvoie à l'adresse « pointer » une chaîne de caractère ASCIIZ contenant les différentes technologies supporté décrites dans la structure pointé par « cpu ».

Pciinfo(word *pciinf)

Renvoie la structure pciinf dans l'adresse « pciinf » qui contient des informations sur le bus PCI.

Getpciclass(word class) :ptr

Renvoie un pointeur vers une chaîne de caractère ASCIIZ contenant la classe spécifiée par « class ».

Getpcisubclass(word class ,word subclass) :ptr

Renvoie un pointeur vers une chaîne de caractère ASCIIZ contenant la sous-classe spécifiée par « class » et « subclass ».

Getcardinfo(word bus, word device, word function, word *pcidata)

Renvoie la structure pcidata dans l'adresse « pcidata » qui contient des informations sur périphérique situé sur le bus « bus », avec le n° composant « device » et la fonction « function ».

Pcireadbyte(word bus, word device, word function, word index) :byte

Renvoie l'octet « index » lu depuis le périphérique situé sur le bus « bus », avec le n° composant « device » et la fonction « function ».

Pcireadword(word bus, word device, word function, word index) :word

Renvoie le mot « index » lu depuis le périphérique situé sur le bus « bus », avec le n° composant « device » et la fonction « function ».

Pcireaddword(word bus, word device, word function, word index) :dword

Renvoie le double mot « index » lu depuis le périphérique situé sur le bus « bus », avec le n° composant « device » et la fonction « function ».

Detectvmware

Renvoie carry si la machine sur laquelle tourne l'OS est une machine virtuelle vmware.

Setvideomode(word mode)

Fixe le mode vidéo courant. Met le flag carry à 1 si erreur.

MODES :
0 -> 40x25x16 couleurs en texte
1 -> 80x25x16 couleurs en texte
2 -> 80x50x16 couleurs en texte
3 -> 100x50x16 couleurs en texte
4 -> 100x60x16 couleurs en texte
5 -> 320x200x256 couleurs en graphique
6 -> 320x400x256 couleurs en graphique
7 -> 320x480x256 couleurs en graphique
8 -> 360x480x256 couleurs en graphique
9 -> 400x600x256 couleurs en graphique
10 -> 640x480x16 couleurs en graphique
11 -> 800x600x16 couleurs en graphique

Les modes sont généralement utilisable avec une carte VGA 256ko, mais la plupart surexploitent le contrôleur vidéo donc ne fonctionne pas toujours. L'utilisation des fonctions caractères est disponible en mode graphique par l'usage de polices émulés mais beaucoup de bugs sont encore présent.

Getvideomode :word

Retourne le mode vidéo courant.

Setfont(word font)

Active une police parmi les 8 disponibles.

Getfont :word

Récupère le n° de la police active actuellement.

Loadfont(word *pointer, word size, word font)

Charge une police pointée par « pointer » dans la carte vidéo sous le n° de police « font ». La taille en hauteur de la police (en pixel) doit être renseigné dans « size ». Met le flag carry à 1 si erreur.

Addline

Affiche un retour a la ligne à l'écran après le curseur.

Showchar(word char, word attr)

Affiche un caractère à l'écran après le curseur dont le code ASCII est contenu dans « char » et dont l'attribut est spécifié dans « attr ». Si « attr » est égal a 0FFFFh, l'attribut déjà affiché sera utilisé. Si « char » est égal a 0FFFFh, le caractère déjà affiché sera utilisé.

Print(word *pointer,……)

Affiche la chaîne de caractère au format ASCIIZ (0 terminal) en utilisant des caractères spéciaux et des caractère de substitution.

- Caractères substitution

  Ils sont précédés de « % » et le(s) paramètre(s) de substitution doit(vent) être empilé selon l'ordre d'apparition dans la chaîne.

Caractères	Types d'affichage	Type C	Taille (par défaut)
c	caractère	unsigned char	byte***
u	nombre décimal non signé	unsigned int	dword
v	nombre décimal non signé centré a gauche	unsigned int	dword+word
w	nombre décimal non signé centré a droite	unsigned int	dword+word
i	nombre décimal signé	int	dword*
h	nombre hexadécimal	unsigned int	dword*
b	nombre binaire	unsigned int	dword*
y	nombre binaire codé decimal	unsigned int	dword*
s	chaîne type pascal	pstr	ptr**
0	chaîne type C	cstr	ptr**
z	nombre avec multiple (Mo,Ko…)	unsigned int	dword
a	attributs de fichier	short int	word
n	nom de fichier	void	ptr**
t	heure	dostime	word
d	date	dostime	word
q	date et heure	dostime	dword
f	float	float	dword
g	double	double	qword

*Ces types peuvent être suffixé d'une taille

Suffixes	Taille
B	8 bits (1 octet)
W	16 bits (2 octets)
D	32 bits (4 octets)

**Ces types peuvent être suffixé pour l'usage d'un pointeur avec offset

Suffixes	**Taille**
P	usage d'un pointeur avec offset de la forme SEG : OFFSET

***Ces types peuvent utiliser un suffixe d'usage multiple

Suffixes	**Taille**
M	répète m fois l'affichage

- Caractères spéciaux

Ils sont précédés de « \ » et les paramètres doivent être ecrit a la suite dans la chaîne.

Caractères	**Effets**	**Paramètres**
l	retour a la ligne	
g	Met le curseur en « x » « y »	x,y (décimal)
h	Met le curseur en « x »,ancien y	x (décimal)
c	Change la couleur courante a « c »	c (hexa)
m	Change le mode graphique en « m »	m (décimal)
e	Efface l'ecran	
s	Sauvegarde l'état de la carte	
r	Restore l'état de la carte	
i	Autorise le defilement	
j	Désactive le defilement	
f	Choisi la police « p »	p (décimal)

- Exemples

call [print],offset essai,large 199000,large A5985h, es,di

essai db « Un nombre décimal : %u\l Un nombre binaire sur 8 bits : %hB\l Une chaine de caractère avec pointeur : %0P\l on saute au milieu de la ligne \h40 HOP ! »,0

Showint(dword number**)**

Affiche le nombre contenu dans « number » sous forme décimale signé à l'écran après le curseur.

Showintl(word size**,dword** number**)**

Affiche le nombre contenu dans « number » (sous forme décimale) de façon centré a gauche par rapport a « size » caractères.

Showintr(word size**,dword** number**)**

Affiche le nombre contenu dans « number » (sous forme décimale) de façon centré a droite par rapport a « size » caractères.

Showsigned(word size, **dword** number**)**

Affiche le nombre contenu dans « number » et de taille « size » bits sous forme décimale signé à l'écran après le curseur.

Showhex(word size**, word** number**)**

Affiche le nombre contenu dans « number » et de taille « size » bits sous forme hexadécimal à l'écran après le curseur.

Showbin(word size**, word** number**)**

Affiche le nombre contenu dans « number » et de taille « size » bits sous forme binaire à l'écran après le curseur.

Showbcd(word size**, word** number**)**

Affiche le nombre contenu dans « number » et de taille « size » bits sous forme binaire codé décimal à l'écran après le curseur.

**Showstring(word ***pointer)

Affiche la chaîne de caractère (type fixe, pascal) pointée par « pointer » à l'écran après le curseur.

Showstring0(word ***pointer)**

Affiche la chaîne de caractère (type zéro terminal, C) pointée par « pointer » à l'écran après le curseur.

Setcolor(word color**)**

Change la couleur courante (attributs) pour les opérations textes.

Getcolor :word

Récupère la couleur courante (attributs) pour les opérations textes.

Setstyle(word style**)**

Change le style (transparent ou non) courant pour les opérations graphique.

Getstyle :word

Récupère le style (transparent ou non) courant pour les opérations graphique.

Scrolldown(word nblines**)**

Défile l'écran texte ou graphique de plusieurs caractères vers le haut.

Getxy :packet

Renvoie les coordonnées x et les coordonnées y du curseur texte.

Setxy(word x**, word** y**)**

Fixe les coordonnées x et les coordonnées y du curseur texte.

Savescreen

Sauvegarde le contenu de l'écran dans un bloc mémoire appelé /vgascreen lié a l'application appelante.

Restorescreen

Restaure le contenu de l'écran précédemment sauvegardé dans un bloc mémoire.

Xchgpage(word page1**, word** page2**)**

Echange le contenu de la page vidéo n° « page1 » avec la page vidéo n° « page2 ». Ne fonctionne qu'en mode texte.

Savepage(word page**)**

Sauvegarde le contenu de la page écran dans un bloc mémoire appelé /vgapage<n°page> lié a l'application appelante.

Restorepage(word page**)**

Restaure le contenu de la page écran précédemment sauvegardé dans un bloc mémoire.

Waitretrace

Synchronisation avec la retrace verticale.

Waithretrace
Synchronisation avec la retrace horizontale.

Getvideoinfos(word *videoinfo)
Renvoie un bloc de donnée en « pointer » contenant l'état de la carte graphique.

Cf structure videoinfo

*Loadbmppalet(word *pointer)
Charge la palette (DAC) du BMP pointée par « pointer ».

*Showbmp(word *bmp, word x, word y)
Affiche le BMP pointée par « pointer » aux coordonnées spécifiées.

*Viewbmp(word *bmp, word x, word y)
Affiche le BMP pointée par « pointer » aux coordonnées spécifiées avec la préparation de la palette.

Savedac
Sauvegarde le contenu de la palette (DAC) dans un bloc mémoire appelé /vgadac lié a l'application appelante.

Restoredac
Restaure le contenu de la palette (DAC) précédemment sauvegardé dans un bloc mémoire.

Savestate
Sauvegarde l'état complet de la carte graphique dans un bloc mémoire appelé /vga lié a l'application appelante. FONCTIONNE PEUT ETRE EN MODE GRAPHIQUE.

Restorestate
Restaure l'etat complet de la carte graphique précédemment sauvegardé dans un bloc mémoire. FONCTIONNE PEUT ETRE EN MODE GRAPHIQUE.

Enablescroll
Active le défilement automatique de l'écran lors de dépassements de la page active.

Disablescroll
Désactive le défilement automatique de l'écran lors de dépassements.

Enablecursor
Active l'affichage du curseur en mode texte.

Disablecursor
Désactive l'affichage du curseur en mode texte.

Showdate(word date)
Affiche une date à l'écran après le curseur.

Showtime(word time)
Affiche une heure à l'écran après le curseur.

Showname(word *pointer)
Affiche le nom de fichier pointé par « pointer » à l'écran après le curseur.

Showattr(word attribs**)**
Affiche des attributs fichiers l'écran après le curseur.

Showsize(dword size**)**
Affiche une taille en octets (et multiples) contenue dans « size » à l'écran après le curseur.

Getchar :byte
Renvoie le caractère situé sous le curseur.

Showpixel(word x**, word** y**, word** color**)**
Affiche un pixel a l'ecran aux coordonnées « x », « y » en utilisant pour couleur « color ».

Getpixel(word x**, word** y**)** :word
Récupère la couleur du pixel aux coordonnées « x », « y ».

Mbinit
Initialise les blocs de mémoire pour une utilisation futur des fonction MBs (inutile car le système le réalise au boot). Met le flag carry à 1 si erreur.

Mbfree(word segment**)**
Libère le bloc de mémoire ainsi que tout les sous blocs de mémoire lié (fils). Un bloc de mémoire considéré résident ou un sous bloc résident ne sera pas libéré. Met le flag carry à 1 si erreur.

Mbcreate(word *pointer, word size**) :word**
Crée un bloc de « size »caractères (octets) et de nom DS: « pointer ». Retourne le bloc de mémoire alloué et met le flag carry à 1 en cas d'erreur.

Mbresident(word segment**)**
Met le bloc en situation de bloc mémoire résident (non libérable).

Mbnonresident(word segment**)**
Met le bloc en situation de bloc mémoire non résident (libérable).

Mbget(word number**) :word**
Renvoie l'adresse du bloc mémoire situé en « number » ème position. Met le flag carry à 1 si introuvable.

Mbfind(word *pointer) :word
Renvoie le bloc de mémoire dont le nom correspond a la chaîne de caractère situé en DS: « pointer ». Met le flag carry à 1 si introuvable.

Mbchown(word segment**, word** segmentpere**)**
Change le propriétaire (père) du bloc de mémoire « segment » a celui précisé par « segmentpere ».

Mballoc(word size**) :word**
Alloue un bloc de « size » caractères (octets) pour le processus (programme) qui le demande. Retourne le bloc de mémoire alloué et met le flag carry à 1 en cas d'erreur.

Mbclean

Nettoie un peu la mémoire pour fonctionner des blocs de mémoire libre contiguë. Généralement inutile car géré par le système.

Mbfindsb(word *pointer, word segment)

Renvoie le sous bloc de mémoire dont le nom correspond a la chaîne de caractère situé en DS: « pointer » et dont le propriétaire est « segment ». Met le flag carry à 1 si introuvable.

Mbloadfuncs(word segment)

Résous les dépendances dynamiques (librairies) du bloc de mémoire.

Mbsearchfunc(word *pointer) :ptr

Recherche la fonction dont le nom est pointé et renvoie son adresse sous forme de pointeur far.

Bioswaitkey

Attend l'appuie sur une touche.

Biosprint(word *pointer)

Affiche une chaîne de caractère au format ASCIIZ en utilisant le BIOS.

Biosprinth(word number)

Affiche un nombre 32 bits non signé en utilisant le BIOS.

Enableirq(word irq)

Active l'irq spécifié.

Disableirq(word irq)

Désactive l'irq spécifié.

Seteoi(word irq)

Signale la fin du traitement de l'irq spécifié.

Readmaskirq(dword pic)

Lit le masque du contrôleur (0-1) d'interruption spécifié.

Reairr(dword pic)

Lit les interruptions en requête du contrôleur (0-1) d'interruption spécifié.

Readisr(dword pic)

Lit les interruptions en service du contrôleur (0-1) d'interruption spécifié.

Enablea20

Active la 20ème broche du bus d'adresse pour un accès supérieur a 1Mo.

Disablea20

Désactive la 20ème broche du bus d’adresse pour un accès supérieur a 1Mo.

Flatmode

Active le mode Flat (unreal) permettant d’accéder a toute la mémoire vive disponible.

*Cmdmouse

Envoie une commande AL à la souris via contrôleur clavier

* Cmdmouse2

Envoie une commande type 2 AL à la souris via contrôleur clavier

*Detectmouse

Détecte et initialise une souris de type PS/2. Met le flag carry à 1 si introuvable.

*Getmouse

Envoie en BX,CX les coordonnées virtuelles de la souris (respectivement X et Y) ainsi qu’en DL l’état des boutons.

*Getmousescreen

Envoie en BX,CX les coordonnées écran de la souris (respectivement X et Y) ainsi qu’en DL l’état des boutons.

*Configmouse

Configure la vélocité de la souris dans CL et dans AH, AL les sphères X et Y.

A suivre pour les autres ressources.... (et avec exemples !)

En cas de problèmes

Si des bugs surviennent ou si COS2000 ne veut pas s'installer, veuillez s'il vous plaît m'envoyer un E-Mail à :

mailto:cos2003@free.fr

COS2000 n'exploite pas les disques durs, il est donc improbable qu'il altère vos données !