Multiboot OS/2 booter description

OS/2 booter (bootos2.mdl) is the multiboot kernel which can be used with FreeLdr as well as GRUB. It is intended for booting OS/2 using these bootloaders. OS/2 booter loads all needed files to memory (they are specified as multiboot modules) and then installs special micro- and minifsd which read files from memory by requests of os2ldr and os2krnl. So, you can think of it as an mbi (multiboot info) micro- and minifsd which boot OS/2 from a special FS-like structure in memory. So, it is like a yet another memdisk technology for booting OS/2. (Or a technology similar to a memfiles, preloaded to memory by OS/4 kernel). The difference from a Veit Kannegieser's memdisk is that OS/2 does not boot from emulated memory drive but a boot drive letter is assigned to a real boot drive (a CD, a flash disk or alike) but not a memdrive.

Also, it can be treated as universal set of mini- and microfsd's for any FS. It is FS independent and can be used with any disk FS which has support in bootloader. The files are preloaded to memory, and then retrieved from there by os2ldr or os2krnl requests. Then a control is passed to a fully-functional IFS. So far, successfully tested IFS'es are: cdfs.ifs (but CD boot still does not work due to limitations of os2dasd.dmd and os2cdrom.dmd), fat32.ifs, ext2-os2.ifs, vfat-os2.ifs, hpfs.ifs, jfs.ifs, ramfs.ifs.

The limitation now is that this is a IFS-only booting solution, so you cannot use it for booting using in-kernel FAT filesystem. As a workaround, you can use vfat-os2.ifs filesystem (unfortunately, it is unstable, but currently, it has no stable replacement).

Syntax:

```
kernel $(DRV)\boot\loader\bootos2.mdl [options]
```

where options are the following

–debug	switches on commport debug
–port=<...>	specifies a commport base address. For example, '–port=0x3f8'
–speed=<...>	specifies a commport speed, for example, '–port=115200'

The standard comport parameters for bootos2 are: 8 bits, no parity, 1 stop bit. The default port is 0x3f8. The default port speed is 9600 baud.

--drv=<drive letter|dlat|auto>	specifies a drive letter assignment.
'–drv=auto'	is intended for booting with danidasd.dmd. When using this assignment mode, uFSD scans all available drives for supported partitions and finds the 1st available drive letter. This drive letter is assigned to the boot drive. This mode is mainly intended for booting from flash drive using danidasd.dmd. The flash drive is usually a removable drive. There is a limitation of danidasd that a removable drives are assigned a drive letters immediately after the fixed drives, and no unused drive letters allowed between fixed and removable drives.
'–drv=dlat'	is intended for booting when usind os2lvm.dmd. In this mode, a drive letter is taken from the LVM DLAT info and assigned to a boot drive. This is mainly intended for booting from fixed drives.
'–drv=u:'	means to assign an 'u:' drive letter to the boot drive
'–fs=<FS name>'	specifies a Filesystem name. FS name is a value of a FS_NAME exported symbol of an IFS. A value of FS name of miniFSD must exactly match a value of the same symbol in the IFS.

–module=<module name>	specifies a boot IFS module name. An IFS is a dynamic link library which has predefined entry points. As any DLL, IFS has its module name which length cannot exceed 8 characters.

A table of known FS and module names for different IFS'es:

FS	module	FS name
fat32.ifs	FAT32	FAT32
hpfs.ifs	HPFS	HPFS
jfs.ifs	JFS	JFS
ext2-os2.ifs	EXT2-OS2	ext2
vfat-os2.ifs	VFAT-OS2	vfat
cdfs.ifs	CDFS (CDFSD for debug)	CDFS
ntfs.ifs	NTFS	NTFS
ramfs.ifs	RAMFS	RAMFS

Note: a FS name and module name are case-sensitive!

–auto-preload	specifies to use 'auto preload' feature. So, you can specify only config.sys as a module; the booter parses config.sys in memory and tries to determine which files are needed to load and loads them automatically.
–remote-fs	means a remote fs boot. This switch is used when booting from 'remote' FS'es as opposed to 'local' ones. An example of remote FS'es is ramfs.ifs (an IFS-based ramdisk).
–ramdisk-boot	means booting from the ramdisk IFS. (A memdisk-like technology). '–ramdisk-boot' implies '–remote-fs'. Files are copied to a ramdisk from multiboot structure, then the boot process continues from a ramdisk.

From:
https://ftp.osfree.org/doku/ - **osFree wiki**

Permanent link:
https://ftp.osfree.org/doku/doku.php?id=en:docs:boot:freeldr:bootos2

Last update: **2014/05/21 21:12**