

# *pmDOS*

## Protected Mode DOS

*pmDOS has been developed for embedded applications which have outgrown the limits of DOS. It allows smooth migration to 32-bit protected mode. pmDOS offers the following significant advantages over DOS:*

- Huge address space
- No royalties
- Segment protection
- Multitasking

### Components

*pmDOS* is built from the following standard Micro Digital products, which are field-proven and used in hundreds of applications:

- *pmEasy* protected mode environment and loader.
- *unDOS* protected mode DOS emulator.
- *smxFile* DOS-compatible file manager with floppy and IDE drivers
- *pmScope*<sup>™</sup> Debugger, Monitor, and Locator
- *smx* real-time multitasking kernel for protected mode. (Optional)

### Tools

*pmDOS* works with Borland and Microsoft 32-bit tools to produce 32-bit flat mode executables that can be loaded by *pmEasy32*. For debugging, the .exe file can be put onto a floppy disk and loaded from the target's floppy drive. Or the .exe file can be copied to the target's hard disk and loaded from there. (Note: the full version of *pmEasy* can also load from DiskOnChip®, CD-ROM, LS-120, and Zip® drives.)

The *pmScope* debugger runs on the host and reads in the symbol file prepared by the locate

### Features

- 4 GB address space
- Easy upgrade from real mode & DOS
- Emulates DOS calls in protected mode
- DOS-compatible file i/o
  - Floppy disk driver
  - Hard disk driver
- DPMI services
- Multitasking
- Boot loadable
- Supports Borland C and Microsoft C
- Supports most C RTL functions
- *pmScope*<sup>™</sup> debugger included
- Full manuals for *pmEasy*, *smx*, *smxFile*, *unDOS*, and Soft-Scope
- Shipped in library form

utility. The target monitor, which is linked with *pmEasy*, establishes contact with the host and the debug session begins. *pmScope* provides good symbolic debugging for C and C++ programs.

### Operation

*pmEasy* is a real-mode .exe file which can bootload (using on-board BIOS), switch the processor to protected mode, (see note 2) then load the application .exe file. The application file is a protected mode .exe file. The *unDOS*, *smxFile*, and *smx* libraries are linked with the application code. Thus they are part of the application .exe file.

*unDOS* emulates DOS, in protected mode, without switching to real mode (as DOS extenders do — actually, they switch to virtual 86 mode, which is the same thing). *smxFile* provides DOS-compatible file i/o. Floppies or hard drives written by DOS or Win9x can be read by *smxFile*, and vice versa. *unDOS* provides the DOS API for *smxFile*. *unDOS* also emulates enough of DOS so that most standard C run-time library functions can be still used in protected mode.

*smx* adds the multitasking dimension. It is frequently the case that applications, which have outgrown the DOS size limitation have also outgrown the superloop structure and are in need of better structuring. Multitasking provides that. Typically, the existing application code, except its isr's, can be brought into *smx* as a single task. The isr's require only minimal modification and will operate essentially the same as in real mode. New functionality is implemented by adding new tasks and new isr's. Doing so reduces the risk of upsetting the old application. As time goes on, the old application can be gradually restructured to take advantage of the features that *smx* offers.

## Other Features

*pmDOS* will cohabit with DOS. It can be loaded from DOS and will exit cleanly back to DOS. Although this defeats eliminating the DOS royalty, it is helpful in some systems – e.g. post-operation analysis using DOS programs of results obtained by the application during operation. Also, having DOS present is usually helpful while debugging the application.

*pmEasy* does permit switching to real mode for executing BIOS or DOS functions, provided that hardware interrupts are disabled. This is useful during initialization. For example, some boards, such as video controllers, have custom initialization BIOS's on them. It is a big benefit

to be able to run their initialization routines rather than having to write new ones.

For more information, see the literature for the individual products included in *pmDOS*.

## Determining DOS Dependency

If you are considering moving an existing DOS application to *pmDOS*, you need to know how much DOS dependency your application has. For this, we have created DOStap™ which can be downloaded from our restricted website. DOStap loads as a TSR that runs concurrently with the DOS application. It records all DOS and BIOS calls. These can be compared to the calls listed in the *unDOS* manual (also found on our restricted website).

## Price and Deliverables

*pmDOS* is offered at a reduced package price and may be purchased with or without *smx*. This includes the products listed above, which are shipped in flat model library form and full manuals. It also includes a site development license, 6 months of support, and a royalty-free incorporation license for one developed product. Source code is provided for the Protosystem and demos.

*pmDOS* is compatible with other Micro Digital products. We offer a full range of protected mode products. These products may be added to *pmDOS*, at standard prices. Also *pmDOS* source code is available for purchase. Contact us for further information.

---

Notes: 1. DiskOnChip is a registered trademark of M-Systems Inc. Zip is a registered trademark of Iomega Corp. All other trademarks are the property of their respective owners.  
2. All x86 processors from 386 upward have a 16-bit protected mode as well as a 32-bit protected mode.