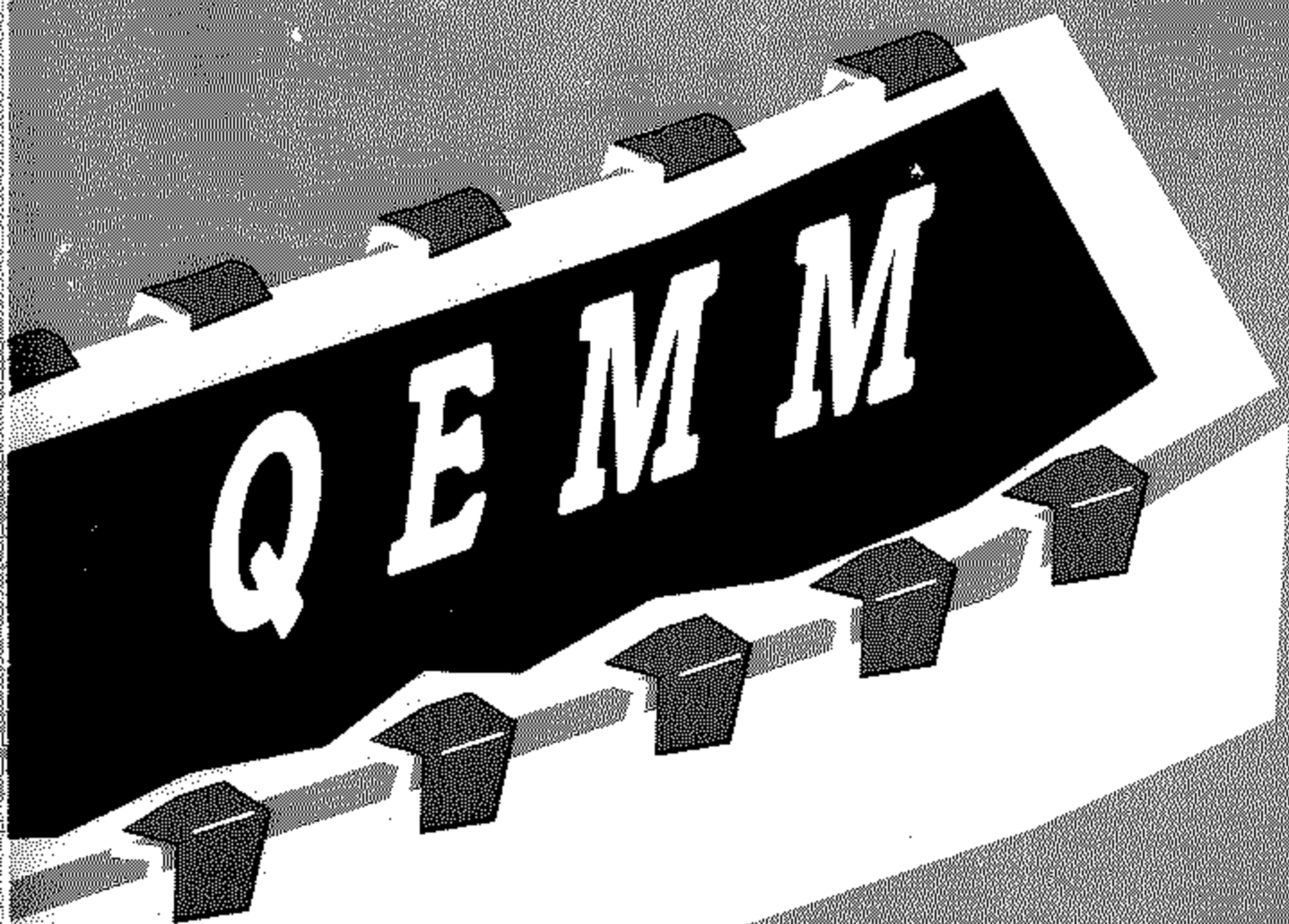


# Quarterdeck expanded memory manager

 Quarterdeck



Instant power  
for 386, 486 or  
Pentium™ PCs

## COPYRIGHT

This software is copyrighted and all rights reserved by Quarterdeck Office Systems, Inc. Portions of DOS-Up© 1992, Philip B. Gardner. DOS Extension Technology© 1991-1992, ERGO Computing, Inc. All Rights Reserved.

The distribution and sale of this software are intended for use by the original purchaser only and for use on a single machine (whether a stand-alone computer or a workstation component of a multi-terminal system). Lawful users of this software are hereby licensed only to read the software on the enclosed diskette(s) from their medium into the memory of a computer solely for the purpose of executing it. Copying, duplicating, selling, or otherwise distributing this software is a violation of the law.

This manual is copyrighted and all rights reserved. This manual may not, in whole or part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from Quarterdeck Office Systems, Inc.

©1985-1993 Quarterdeck Office Systems, Inc.  
1901 Main Street, Santa Monica, CA 90405 • (310) 392-9851

All rights reserved.  
Patent pending.

## TRADEMARKS

*QEMM*®, *Manifest*®, *DESQview*®, and *Quarterdeck*® are registered trademarks and *Quarterdeck Expanded Memory Manager*™, *DESQview 386*™, and *DESQview/XT*™ are trademarks of Quarterdeck Office Systems.

*AutoCAD*™ is a trademark of Autodesk, Inc. *Compaq*™ and *ProLinea*™ are trademarks of Compaq Computer Corporation. *MS-DOS*® is a registered trademark and *Windows*™ is a trademark of Microsoft Corporation. *PS/2*™, *Operating System/2*™ and *OS/2*™ are trademarks of International Business Machines. *Intel*™, *Pentium*™, *i386*™ and *i486*™ are trademarks of Intel Corporation. *Dell*® is a registered trademark of Dell Computer Corporation. *Norton Utilities*® is a registered trademark of Symantec Corporation. *PC-Kwik*® is a registered trademark of PC-Kwik Corporation. *Stacker*® is a registered trademark of STAC Electronics. *Novell*®, *LAN WorkPlace for DOS*® and *NetWare*® are trademarks or registered trademarks of Novell, Inc. *dBASE*® are and *QUATTRO Pro*® are registered trademarks of Borland International Inc. *DR DOS*™ is a trademark of Digital Research, Inc. *Lotus 1-2-3*® is a registered trademark of Lotus Development Corporation. *DoubleDisk*™ is a trademark of Vertisoft Systems, Inc., Mitran Software International (1989) LTD. and Merckrechten en Patenten Nederland B.V. Vertisoft is a registered trademark of Vertisoft Systems, Inc. *UNIX*® is a registered trademark of UNIX System Laboratories, Inc. *XtraDrive*™ is a trademark of Integrated Information Technology, Inc.

*SuperStor*® and *AddStor*® are registered trademarks of AddStor, Inc. *WordPerfect*® is a registered trademark of WordPerfect Corporation. *Ventura Publisher*® is a registered trademark of Xerox Corporation. *Oracle*® is a registered trademark of Oracle Corporation. *Phar Lap*® is a registered trademark of Phar Lap Software, Inc.

Other brand or product names are trademarks or registered trademarks of their respective holders.

**License**            001-37E-99875

United States:

This License is your proof of license.      Proof of  
Please treat it as valuable property.      License

## QUARTERDECK END USER LICENSE AGREEMENT

**NOTICE TO END USER: CAREFULLY READ THE FOLLOWING LEGAL AGREEMENT. USE OF THE SOFTWARE (THE "SOFTWARE") (AND FONTS, IF ANY) PROVIDED WITH THIS AGREEMENT CONSTITUTES YOUR ACCEPTANCE OF THESE TERMS. IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, PROMPTLY RETURN THE SOFTWARE AND THE ACCOMPANYING ITEMS (INCLUDING WRITTEN MATERIALS, BINDERS AND CONTAINERS) TO THE LOCATION WHERE YOU OBTAINED THEM FOR A FULL REFUND.**

- 1. License Grant.** Quarterdeck Office Systems grants to you (either as an individual or entity) a nonexclusive sublicense subject to the provisions hereof: (a) to use the SOFTWARE solely for your own internal personal or business purposes on a single computer (whether a standard computer or a workstation component of a multi-user network). You may not copy the written materials accompanying the SOFTWARE. This Agreement is effective until the year 2040.
- 2. Proprietary Rights.** You acknowledge that the SOFTWARE is proprietary to Quarterdeck and its suppliers. You agree to hold the SOFTWARE in confidence, disclosing the SOFTWARE only to authorized employees having a need to use the SOFTWARE as permitted by this Agreement and to take all reasonable precautions to prevent disclosure to other parties.
- 3. Other Copies.** You will not make or have made, or permit to be made, any copies of the SOFTWARE or portions thereof, except as necessary for its use with a single licensed computer system under the terms and conditions of this Agreement. You agree that any such copies shall contain the same proprietary notices which appear on or in the SOFTWARE.

(License agreement continued on rear inside cover.)

4. **Ownership.** Except as stated above, this Agreement does not grant you any rights to patents, copyrights, trade secrets, trade names, trademarks (whether registered or unregistered), or any other rights, franchises, or licenses in respect of the SOFTWARE. Title to and ownership of the SOFTWARE, any reproductions and any documentation shall remain with Quarterdeck and its suppliers. You will not adapt or use any trademark or trade name which is likely to be similar to or confusing with that of Quarterdeck or any of its suppliers or take any other action which impairs or reduces the trademark rights of Quarterdeck or its suppliers.

5. **Other Restrictions.** This Agreement is your proof of license to use the SOFTWARE in accordance with these terms and must be retained by you. You may not rent or lease the SOFTWARE, but you may assign your rights under this Agreement on a permanent basis to an assignee of all of your rights, title and interest to such SOFTWARE provided you transfer this Agreement, all copies of the SOFTWARE and all accompanying written materials, and such assignee agrees to be bound by all the terms and conditions of this Agreement. YOU MAY NOT ALTER, MODIFY, REVERSE ENGINEER, DECRYPT, DECOMPILE, OR DISASSEMBLE THE SOFTWARE.

6. **Limited Warranty.** Quarterdeck warrants that the SOFTWARE will perform substantially in accordance with the accompanying written materials and that the printed materials and diskettes are free from any physical defects for a period of ninety (90) days from the date of purchase. Any implied warranties on the SOFTWARE, printed materials or diskettes are limited to ninety (90) days.

7. **Customer Remedies.** Quarterdeck's entire liability and your sole and exclusive remedy shall be, at Quarterdeck's option, either to (a) correct the error, (b) help the end user work around or avoid the error or (c) authorize a refund, so long as the SOFTWARE, printed materials or diskettes are returned to Quarterdeck with a copy of your receipt. This Limited Warranty is void if failure of the SOFTWARE has resulted from accident, abuse, or misapplication. Any replacement SOFTWARE will be warranted for the remainder of the original warranty period.

8. **No Other Warranties.** QUARTERDECK DOES NOT WARRANT THAT THE QUARTERDECK SOFTWARE IS ERROR FREE. QUARTERDECK DISCLAIMS ALL OTHER WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OF THIRD PARTY RIGHTS WITH RESPECT TO THE SOFTWARE, THE ACCOMPANYING WRITTEN MATERIALS OR DISKETTES. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES OR LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY MAY LAST, OR THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU. THIS

WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM JURISDICTION TO JURISDICTION.

9. **Export.** You acknowledge that the laws and regulations of the United States restrict the export and re-export of commodities and technical data of United States origin, including the SOFTWARE. You agree that you will not export or re-export the SOFTWARE in any form without the appropriate United States and foreign government licenses. You agree that its obligations pursuant to this section shall survive and continue after any termination or expiration of rights under this Agreement.

10. **Severability.** In the event of invalidity of any provision of this Agreement, the parties agree that such invalidity shall not affect the validity of the remaining portions of this Agreement. The United Nations Convention on Contracts for the International Sale of Goods is specifically disclaimed.

11. **No Liability for Consequential Damages.** IN NO EVENT SHALL QUARTERDECK BE LIABLE TO YOU FOR ANY CONSEQUENTIAL, SPECIAL, INCIDENTAL OR INDIRECT DAMAGES OF ANY KIND ARISING OUT OF THE USE OF THE SOFTWARE, EVEN IF QUARTERDECK HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT WILL QUARTERDECK'S LIABILITY FOR ANY CLAIM, WHETHER IN CONTRACT, TORT OR ANY OTHER THEORY OF LIABILITY, EXCEED THE LICENSE FEE PAID BY YOU.

12. **Entire Agreement.** This is the entire agreement between you and Quarterdeck which supersedes any prior agreement, whether written or oral, relating to the subject matter of this Agreement.

#### U.S. GOVERNMENT RESTRICTED RIGHTS

If this product is acquired under the terms of a: **DoD contract:** Use, duplication or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of 252.227-7013. **Civilian agency contract:** Use, reproduction or disclosure is subject to 52.227-19 (a) through (d) and restrictions set forth in the accompanying end user agreement. Unpublished-rights reserved under the copyright laws of the United States. Quarterdeck Office Systems, Inc., 1901 Main Street, Santa Monica, CA 90405.

This Agreement is governed by the laws of the State of California.

Should you have any questions concerning this Agreement, or if you desire to contact Quarterdeck for any reason, please write: Quarterdeck Office Systems, Inc., 1901 Main Street, Santa Monica, CA 90405.

The QEMM program was written by Dan Spear, Larry Mayer, Steve Goodrich, Mike Wacker, Mark Indictor, Brian Breidenbach, Russell Bonakdar, Jim Susoy and Steve Sprigg.

This manual was written by Phil Glosserman and Dan Sallitt.  
The layout is by Phil Glosserman, with assistance from Yvonne Johnson and Laura Camuti.  
The layout was done using Ventura Publisher.



## Table of Contents

<b>Chapter 1: Introduction</b>	<b>5</b>	<b>Chapter 5: Stealth ROM and Stealth DoubleSpace: Maximizing High RAM</b>	<b>43</b>
What You Need	6	Enabling Stealth ROM	43
Where to Go From Here	6	Disabling Stealth ROM	44
The Basic Types of PC Memory	6	A Technical Description of Stealth ROM	44
What is High RAM?	8	Stealthing DOS 6's DoubleSpace Driver	45
QEMM's Benefits and Features	9	Enabling or Disabling Stealth DoubleSpace	45
Conventional Memory Gains	15	A Technical Description of Stealth DoubleSpace	45
About This Manual	15		
About Manifest	16		
 <b>SECTION 1: USING QEMM</b>		 <b>Chapter 6: VIDRAM: Extending Conventional Memory</b>	<b>47</b>
<b>Chapter 2: Installation and Setup</b>	<b>17</b>	How VIDRAM Works	47
If You Already Have a Memory Manager	17	Using VIDRAM	48
Installing QEMM	18	VIDRAM Parameters	50
Optimize	19	Loading VIDRAM High	51
Stealth ROM	19	Using VIDRAM with Microsoft Windows	51
Using QEMM on Battery-powered PCs	20		
Booting Without QEMM	20	 <b>SECTION 2: TECHNICAL REFERENCE</b>	
If You Experience Any Problems Running Your Programs	21	<b>Chapter 7: QEMM386.SYS Parameters</b>	<b>53</b>
If You Ever Get an Exception 13 Error Message	21	Parameters and Memory Addresses	53
How QEMM Affects Your Existing Configuration	22	Parameter Files	54
Using QEMM with DOS 5 or 6 or Microsoft Windows	24	Parameter Descriptions	54
Using QEMM with DESQview	24	Commonly Used Parameters	55
QEMM's Setup Program	25	Less Frequently Used Parameters	58
 <b>Chapter 3: The Optimize Program</b>	<b>27</b>	Using the ROM and STEALTHROM Parameters Together	79
When to Run Optimize	27	Using Parameters From Previous Versions of QEMM	81
How to Run Optimize	27	 <b>Chapter 8: The LOADHI Programs</b>	<b>83</b>
If You Encounter Problems	28	Using LOADHI	83
Optimize Options	28	The LOADHI Report	84
Forcing Stealth ROM Testing	29	Loading Device Drivers High with LOADHI.SYS	84
How Optimize Works	29	Loading TSRs High with LOADHI.COM	85
Undoing an Optimize	30	The LOADHI Parameters	86
Optimizing Embedded Batch Files	30	LOADHI Notes for MS-DOS 5 and 6	90
Excluding TSRs or Device Drivers from Optimize	31	 <b>Chapter 9: QEMM Reports, Analysis and Mode Switching</b>	<b>91</b>
Optimize Parameters	31	Changing QEMM's Mode	91
 <b>Chapter 4: DOS-UP: Loading Parts of DOS into Upper Memory</b>	<b>37</b>	QEMM.COM Reports	92
If You Have DESQview or DESQview/X	38	The Summary Report	93
Enabling or Disabling DOS-Up	38	The Type Report	94
DOS-Up Drivers	38	The Accessed Report	96
Booting Without DOS-Up	39	Resetting Memory	97
QEMM's DOS Resource Programs	39	The Analysis Report	97
BUFFERS.COM	40	The Memory Report	100
FILES.COM	40		
FCBS.COM	41		
LASTDRIV.COM	41		

<b>Chapter 10: QDPMI: QEMM's DPMI Host</b>	<b>103</b>
Software Compatibility	103
QDPMI Client Initialization	104
with Virtual Memory	105
System Interrupt Stack Configuration	105
Novell LAN WorkPlace for DOS	105
Borland C/C++ 3.1	106
QDPMI Parameters	108
The QDPMI Environment Variable	108
Using QDPMI with DESQview or	108
DESQview/X	109
Virtual Memory and Total Memory Size	109
Error Messages	109
<b>Chapter 11: EMS Utility Programs</b>	<b>113</b>
The EMS Programs	113
EMS Parameters	114
EMS2EXT.SYS	116

### SECTION 3: APPENDICES

<b>Appendix A: Troubleshooting</b>	<b>119</b>
<b>Appendix B: QEMM's Technical Bulletins</b>	<b>129</b>
Operating Systems	129
Microsoft Windows 3.0/3.1	129
QEMM's Stealth ROM Feature	129
Maximizing your DESQview and	129
DESQview/X Windows	129
Bus-Mastering Adapters	130
Disk Compression Utilities	130
Parity Errors	130
Troubleshooting QEMM	130
<b>Appendix C: QEMM/DOS 6 Memory</b>	<b>131</b>
<b>Comparisons</b>	<b>131</b>
IBM PS/2 Model 55sx:	132
DOS 6 Without QEMM	134
IBM PS/2 Model 55sx:	134
DOS 6 With QEMM	136
Compaq Prolinea: DOS 6 Without QEMM	138
Compaq Prolinea: DOS 6 With QEMM	140
Dell 450DE: DOS 6 Without QEMM	142
Dell 450DE: DOS 6 With QEMM	142
<b>Appendix D: Memory Specifications Supported</b>	<b>145</b>
<b>by QEMM</b>	<b>145</b>
<b>Index</b>	<b>147</b>



## Introduction

Thank you for purchasing Quarterdeck's Expanded Memory Manager—QEMM. QEMM works behind the scenes to provide you with as much memory as possible for running programs, thus freeing you from the details of memory management. Since 1985, Quarterdeck has been the leader in bringing memory management innovations to DOS, and for that reason, QEMM has consistently garnered top awards for memory management software in numerous computer trade publications including PC Magazine, InfoWorld, Byte and PC Computing.

**You do not have to be a computer expert or understand memory management to use QEMM. Most users will just perform a simple automated installation and immediately enjoy the benefits of using QEMM. If you are eager to start, turn to Chapter 2 now.**

**What is so important about QEMM?** Not only do today's computers have more power and memory, but PC programs are larger and make greater demands on memory than ever before. A typical PC may be simultaneously running a multitasking environment such as DESQview or Microsoft Windows, a disk compression utility such as Stacker or DoubleSpace, multiple TSRs (memory-resident programs, such as DOS 6's VSafe anti-virus program) and device drivers for a network, mouse, CD ROM, sound board, and other peripheral devices. And, different programs use different kinds of memory: conventional memory, upper memory, expanded memory, and extended memory. QEMM manages all these kinds of memory and can automatically transform your PC's memory into whatever kind of memory your program needs. QEMM is compatible with MS-DOS, IBM DOS, and DR DOS, and supports DOS, DOS Extended and Microsoft Windows applications.

QEMM uses advanced techniques to create the maximum amount of first-megabyte memory for running programs on your system. QEMM allows you to run larger applications, by giving your programs as much conventional memory as possible. QEMM even helps your applications to run faster. Plus, QEMM includes several features that can provide additional memory, over and above that provided by MS-DOS 6's memory manager.

QEMM is solid and dependable—time-proven by millions of users. Whether you are using DOS version 3, 4, 5 or 6, QEMM will provide you with superior, hands-free management.

In addition, when combined with Quarterdeck's DESQview, QEMM turns DESQview into DESQview 386—a powerful 386 multitasking control program.

PC memory management is a technical subject, but you need not understand it to use QEMM. QEMM manages your PC's memory behind

the scenes—most users will only need to perform a simple installation, then forget about memory management altogether. If you are an advanced user interested in the details of memory management, QEMM has advanced features and parameters you can use to fine-tune memory management on your PC.

If you are upgrading from an earlier version of QEMM, version 7 has several enhancements, including support for Intel's Pentium processor, the ability to load DOS in High RAM and the ability to free up more RAM by special management of MS-DOS 6's DoubleSpace device driver. Many of QEMM's improvements are "under the hood"—QEMM is faster, more efficient and can free up even more memory for your applications.

### What You Need

To use QEMM, you need the following:

- ❑ An IBM or IBM-compatible 386DX, 386SX, 386SL, i486DX, i486SX, Pentium PC or PS/2.
- ❑ 256K or more of extended memory.
- ❑ MS or IBM DOS versions 3.0 or later (or DR DOS 5.0 or later).
- ❑ A hard disk or network server.

### Where to Go From Here

The rest of this chapter gives some basic information about PC memory and describes QEMM's features. If you are eager to get started, you can skip ahead to **Chapter 2** which explains how to install QEMM. As mentioned earlier, most users will only need to perform a simple installation, then leave the details of memory management to QEMM.

**QEMM is for IBM and IBM-compatible PCs equipped with 386SX, 386DX, 386SL, i486DX, i486SX, or Pentium processors. In this guide, we will use the terms 386 and 80386 inclusively, to refer to all of these PCs.**

### The Basic Types of PC Memory

Here is a brief discussion of the different kinds of memory available on 386 PCs.

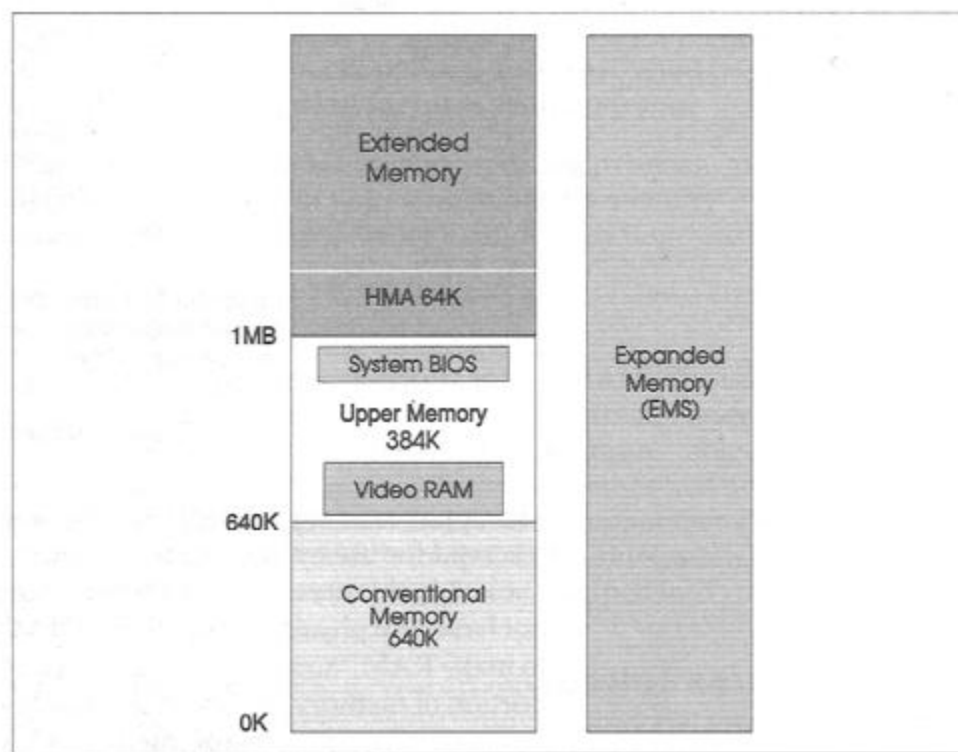
- ❑ **Conventional memory** starts at 0K and normally ends at 640K; it is where most DOS programs run. Most DOS programs run in *real mode* (the main processor's basic operating mode), which only allows access to the first megabyte of memory addresses (0-1024K). Because the addresses between 640K and 1024K are reserved for ROMs and system hardware, real mode programs must normally run in the area below 640K. Without the aid of a memory manager like QEMM, most TSRs (memory-resident programs) and device drivers also occupy conventional memory.
- ❑ **Upper memory** starts at the end of conventional memory and ends at 1024K. Normally this area is set aside for use by system ROM, video and hardware adapter cards (e.g., network adapters). On most PCs,

hardware does not use the entire upper memory area. QEMM can use the unused upper memory addresses to load device drivers and TSRs (memory-resident programs) that otherwise would occupy space in conventional memory. QEMM can also load selected parts of DOS into the upper memory area. For more information on loading items into the upper memory area, see the next section.

- ❑ **Extended memory** is the memory addressed above 1024K—it is used by programs operating in protected mode. *Protected mode* is a collective name given to several advanced operating modes of 80286 and higher processors, and is used mainly by DOS-extended programs, XMS programs, memory managers such as QEMM, and multitasking control programs such as Microsoft Windows.

A DOS-extended program is a special DOS program that runs in your PC's protected mode, but fools DOS into thinking that it is a normal real mode program. AutoCAD 12, Lotus 1-2-3 Release 3, Oracle Professional, IBM Interleaf and Microsoft Windows 3.0 and 3.1 (in standard or enhanced mode) are DOS-extended programs.

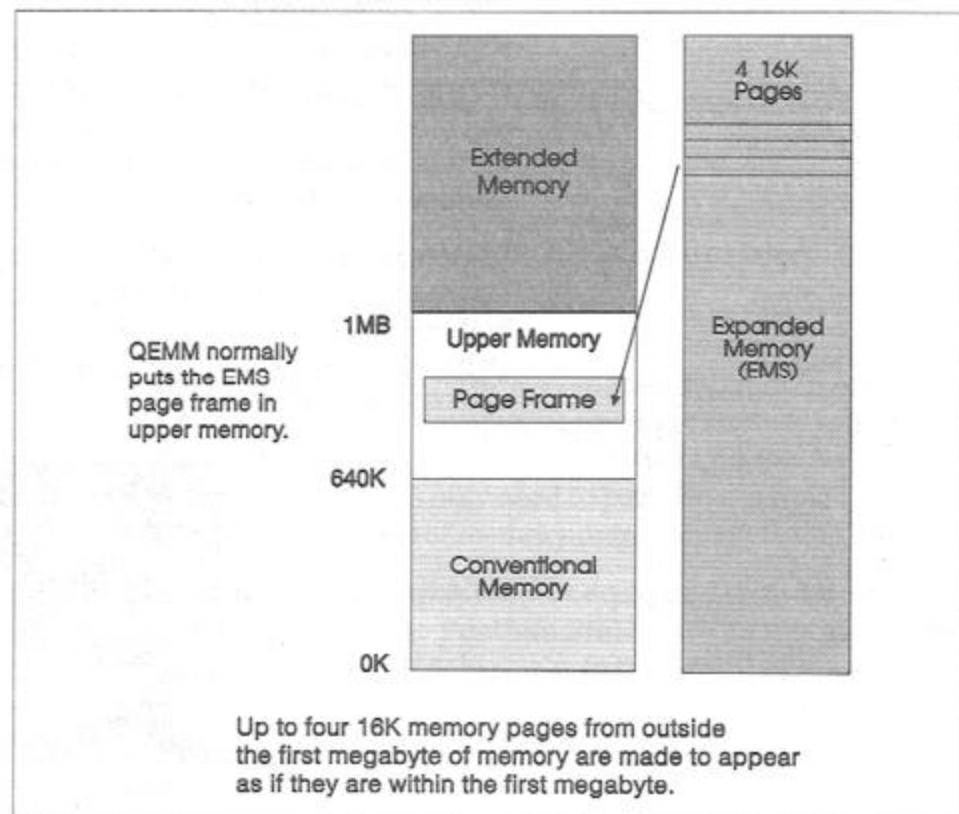
- ❑ **High memory area (HMA)** is the first 64K of extended memory. The HMA differs from the rest of extended memory because the 286 and higher processors can access it in real mode. Some programs (e.g., DOS, DESQview) can store a portion of their code in the HMA to free up conventional memory.



The basic kinds of PC memory



- Expanded memory (also called EMS) is memory outside the first megabyte of memory that an expanded memory manager such as QEMM can cause to appear within the first megabyte in 16K units called *pages*. Some programs use expanded memory to store and access data that will not fit in conventional memory. In most cases, a program that uses EMS accesses up to 64K of expanded memory at a time (in 16K units called pages) at a special area of upper memory called the *page frame*. Originally, on PC models lower than 386s, EMS was memory you added onto your PC with an EMS adapter card. QEMM uses the 386 processor's mapping capabilities to transform your PC's extended memory into expanded memory for programs that request this type of memory.



Expanded memory (EMS)

#### What is High RAM?


The upper memory area (the area between 640K and 1024K) is normally reserved for use by your system's hardware. On many PCs, there are "holes" in this area—that is, there are upper memory addresses that do not have any physical RAM chips. QEMM uses a process called mapping to make RAM "appear" in those areas. *Mapping* is the process of making a portion of memory (on 386 PCs, usually extended memory) appear at a different address. By mapping memory, QEMM can "fill out" any missing physical memory in the first megabyte of memory. *High RAM* is QEMM's name for the memory mapped into your PC's available upper memory addresses. QEMM can fill areas as small as 4K with High RAM.




## QEMM's Benefits and Features

✓ indicates a new  
feature.

+ indicates a feature  
not found in MS-DOS  
6's memory manager.

 Memory mapped into upper memory addresses that can be accessed by the XMS interface is called UMBs (Upper Memory Blocks). QEMM lets DOS 5 and 6 and other programs that need UMBs access High RAM as UMBs.

Once there is memory mapped into upper memory addresses, QEMM can use that High RAM to load TSRs, device drivers, and parts of DOS. By loading these items into upper memory instead of in conventional memory, more conventional memory is available for your other programs.

 Throughout this guide, we will use the terms High RAM and upper memory interchangeably when we speak of loading items into the area between 640K and 1024K. That is, "loading a TSR into upper memory" means the same thing as "loading a TSR into High RAM."

QEMM offers the following benefits and features:

**Benefit:** QEMM sets up your PC's memory optimally—without you having to know anything about memory.

- ✓ ☐ QEMM's Express Install automatically examines your system and determines how best to configure its memory and makes any necessary changes to your system's CONFIG.SYS and AUTOEXEC.BAT files—requiring no input from you. You can install QEMM and optimize your system in under five minutes.
- ☐ QEMM responds to your programs' requests for memory and provides them with the kind of memory they need (expanded or extended)—with no need for intervention by you.
- ☐ Manifest, Quarterdeck's system reporting program, gives you a detailed description of your system and its memory—valuable if you want over-the-phone assistance in understanding your system.

**Benefit:** QEMM gives you the maximum amount of free conventional DOS memory for running your programs.

- ☐ QEMM fills missing memory areas with RAM. The upper memory area (the area between 640K and 1MB) is normally reserved for use by your system's hardware. On many PCs, there are "holes" in this area—that is, there are upper memory addresses that do not have any physical RAM chips. QEMM finds unused upper memory addresses as small as 4K in size and fills (maps) these addresses with RAM. High RAM is QEMM's name for the memory mapped into your PC's available upper memory addresses.
- ☐ QEMM loads TSRs and device drivers into High RAM. By loading items into upper memory instead into conventional memory, QEMM gives you more free conventional memory for running DOS programs. (For example, in addition to the 64K reserved for the EMS page frame, QEMM typically finds 128K of usable upper memory on IBM PS/2s and 128K-160K on other PCs. And, on most PCs, QEMM's Stealth

✓ indicates a new feature.

+ indicates a feature not found in MS-DOS 6's memory manager.

ROM feature (see below) can find significantly more usable upper memory. When QEMM loads an item into High RAM, we say the item is "loaded high."

- + ✓ ☐ **DOS-Up can load parts of DOS into High RAM.** QEMM's DOS-Up feature can load DOS's command processor, the DOS kernel, DOS Data, and DOS resources (FILES, BUFFERS, STACKS, LASTDRIVE and FCBS) into High RAM. If you have DOS 5 or 6, this feature can typically free up 7K-70K of conventional memory, depending on how your system is configured. If you have DOS 3 or 4, this feature can free up 50 to 70K of conventional memory.
- ☐ **Optimize automatically configures upper memory.** QEMM's Optimize program examines the device drivers and TSRs (i.e., memory resident programs) that are loaded at system startup time and determines which ones can be moved from conventional to upper memory. Optimize analyzes all possible ways of loading them, then moves them to the locations that ensure the best fit. By moving these items to upper memory, Optimize frees up more conventional memory to run your programs. Optimize is the reason you do not have to be a PC expert to make the best use of your PC's memory.
- + ☐ **Squeeze finds extra space for TSRs and device drivers to initialize.** Often, TSRs and device drivers require more memory to initialize than they do once they are resident. Optimize has a feature called Squeeze that ensures that such programs get the extra memory they need to initialize. The Squeeze feature relinquishes this extra memory when the program finishes initialization. Without Squeeze, certain items might not load high because their initialization sizes are larger than any contiguous available upper memory area.
- + ☐ **QEMM's Stealth ROM feature (patent pending) uses the EMS page frame to manage your PC's ROMs, freeing up an additional 48K-115K of upper memory ROM addresses for use by TSRs and device drivers.** When Stealth ROM is enabled, QEMM can create up to 211K of High RAM on IBM PS/2s, Compaqs, and many other PC compatibles.
- + ✓ ☐ **The Stealth DoubleSpace feature uses the EMS page frame to manage MS-DOS 6's DoubleSpace device driver, freeing 40K of memory.**
- ☐ **QEMM creates and manages the High Memory Area (HMA), the first 64K of extended memory.** This enables certain programs such as DOS or DESQview to store a portion of their code and data in this area.
- ✓ ☐ **QEMM automatically handles programs that can load themselves into upper memory.** Some programs are configured to put themselves in upper memory. Optimize recognizes this, and loads them into High RAM.

✓ indicates a new feature.

+ indicates a feature not found in MS-DOS 6's memory manager.

- + □ **VIDRAM can extend conventional memory up to 96K for running DOS text-based programs.** QEMM's VIDRAM program extends conventional memory into the upper memory address space normally reserved for EGA or VGA graphics. While using VIDRAM, you cannot use EGA or VGA graphics, but you can easily turn VIDRAM on and off, before and after you run your text-based program. VIDRAM also extends conventional memory for DOS text-based programs running in Microsoft Windows or DESQview. If you have an 8514A video adapter, you can use VIDRAM to extend conventional memory for DOS text-based programs as well as 8514A graphics programs.
- ✓ □ **QEMM can create an additional 32K of High RAM from "unused" system ROM.** On certain PCs, if you are not using Stealth ROM, QEMM will find 32K of system ROM that is used only during system bootup, and include those ROM addresses for use as High RAM.
- ✓ □ **QEMM creates an additional 32K of High RAM on IBM machines.** IBM computers include an old version of the BASIC programming language on a ROM. QEMM creates the extra High RAM by mapping memory to the BASIC ROM's address space.

**Benefit: QEMM dynamically transforms your PC's memory into whatever kind of memory your programs need.**

- **QEMM automatically transforms your PC's extended memory (the memory above 1 megabyte) into expanded memory (EMS) when you run programs that use expanded memory.** QEMM is compatible with EMS 3.2, EMS 4, and XEMS and supports EMS 4 real alternate maps.
- **QEMM manages extended memory.** QEMM supports all three functions of the XMS v3 specification: HMA (High Memory Area), UMBs (Upper Memory Blocks) and EMBs (Extended Memory Blocks). QEMM provides both VCPI (Quarterdeck/Phar Lap Virtual Control Program Interface) and DPMI (DOS Protected Mode Interface) support. VCPI and DPMI are industry standards which specify how 80386 control programs and DOS extenders communicate with each other. QEMM is fully compatible with DOS-extended programs such as AutoCad 12, Lotus 1-2-3 Release 3, Oracle Professional and Interleaf. (For a technical description of the memory specifications supported by QEMM, see Appendix D.)
- **QEMM pools expanded and extended memory and allocates it to programs as needed.** You need not preallocate fixed amounts of expanded and extended memory. QEMM treats all of your PC's memory as a pool of memory—to be used as extended memory or expanded memory at the time your programs need it.
- + ✓ □ **QEMM provides DPMI (DOS Protected Mode Interface) memory for programs that utilize the DPMI 0.9 API** (e.g., Microsoft C/C++ 7.0, Borland C/C++ 3.x, Intel 386/486 C Code Builder Kit). Such programs run in the protected mode of the 386 and 486 processor, allowing them

✓ indicates a new feature.

+ indicates a feature not found in MS-DOS 6's memory manager.

to have direct access to all system memory for both their code and data. QEMM's DPMM Host also provides support for the DPMM 0.9 API under Quarterdeck's DESQview and DESQview/X multitasking environments.

**Benefit:** QEMM is Microsoft Windows-aware—it detects Windows' presence and optimizes memory for the best Windows performance.

- + ☐ QEMM can provide 8K-24K more High RAM for running DOS programs inside Windows' 386 enhanced mode. QEMM provides space in upper memory for the Windows translation buffers and other parts of Windows (8K-24K) dynamically and automatically, without having to restrict this memory from being used by other device drivers and TSRs.
- ☐ By loading TSRs and device drivers into upper memory, QEMM frees up additional memory for running DOS programs under Windows.
- + ✓ ☐ QEMM's VIDRAM program can extend the amount of conventional memory available to DOS applications running in Windows by as much as 96K.

**Benefit:** QEMM is performance-and-compatibility tuned to give you a more smoothly running system.

- + ✓ ☐ QEMM is a full 32-bit protected mode program, giving you maximum memory management performance.
- + ✓ ☐ QEMM is specially tuned for the Intel Pentium to take advantage of its features for performance and memory savings.
- + ☐ QEMM supports the Suspend/Resume feature of many laptop PCs. Many battery-powered computers have a Suspend/Resume feature that allows the system to power up in the same state it was in before you turned it off. It resumes the program you were using, as if there were no interruption. If your system supports Suspend/Resume, QEMM will try to detect and enable support of this feature.
- ☐ QEMM can map slow ROM code into faster RAM for better system performance.
- + ☐ QEMM supports Adapter Description Libraries for PS/2s. IBM PS/2s and other microchannel architecture (MCA) computers identify each peripheral hardware device with a unique adapter descriptor number. QEMM keeps a record of these descriptions in an Adapter Description Library (ADL) to resolve any memory addressing conflicts with MCA devices.
- ☐ QEMM can free up more memory on certain Compaq computers. Compaq PCs make a copy of a 16K-32K video ROM in upper memory to speed up video processing. QEMM eliminates this mechanism and



✓ indicates a new feature.

+ indicates a feature not found in MS-DOS 6's memory manager.

compensates for it by shadowing the video ROM with memory mapped over the original ROM location. This creates 16-32K more High RAM and creates a larger contiguous High RAM region. Some Compaqs have a duplicate copy of a 32K system ROM; QEMM eliminates the duplicate copy, creating an additional 32K of High RAM. Some Compaqs use 384K of the extended memory space for top memory, memory used to speed up ROMs and for Compaq utilities. QEMM performs top memory's functions and adds the 384K of memory to the memory pool so it can be used as expanded or extended memory. QEMM can also shrink down the system ROM size when a Compaq "cut table" is present.

- ✓ □ **QEMM's Optimize program automatically detects adapter ROM and RAM (including RAM mapped into memory after QEMM as with many network adapters) to prevent memory conflicts.** Adapters such as network cards, FAX cards and scanner cards generally use specific upper memory addresses. QEMM automatically detects an adapter's address space and prevents it from being used as High RAM. This prevents memory conflicts in an adapter's address space and saves you the trouble of having to tell QEMM which addresses are being used by adapters.
- ✓ □ **QEMM will automatically detect bus-mastering hard drive controllers.** QEMM will detect a bus-mastering hard drive controller and set up a special buffer to prevent memory errors that have been known to occur when using a memory manager along with these devices. QEMM is fully compatible with the VDS (Virtual DMA Services) specification for resolving compatibilities between 386 memory managers and bus-mastering devices (for information on the VDS specification, see Appendix D).
- **QEMM automatically monitors Direct Memory Access (DMA) into mapped memory to prevent destructive memory writes.**
- + □ **QEMM can use shadow memory or top memory to give you more expanded or extended memory.** QEMM can detect Chips & Technologies LEAP, NEAT, or SCAT ShadowRAM; NEC, OPTI, PEAK or TOPCAT shadow memory, or Compaq-style top memory, and add it to the memory pool, giving you up to 284K more expanded or extended memory.
- + □ **QEMM unleashes the full power of DESQview and DESQview/X.** With QEMM, DESQview and DESQview/X can run programs that write directly to the screen in the background without writing over existing windows, display each program running in a small window on the screen, and protect the system against certain program violations.

✓ indicates a new feature.

+ indicates a feature not found in MS-DOS 6's memory manager.

**Benefit: QEMM provides comprehensive state-of-the-art fine-tuning and analysis tools for power users.**

- **Manifest helps you analyze memory.** QEMM includes Manifest, Quarterdeck's comprehensive, state-of-the-art memory analysis and reporting program. Manifest can help you understand the details of your PC's memory usage.
- + □ **QEMM's LOADHI.COM and QEMM.COM programs can give you information about how High RAM is being used.**
- + □ **QEMM has many parameters that let you fine-tune memory management for your particular configuration and needs.**
- + ✓ □ **Parameter files make it easy to change QEMM configuration.** For advanced users who want to experiment with different QEMM options, parameter files can be a great time saver. Instead of editing the QEMM device driver line in CONFIG.SYS, you can specify parameters in a file. Or, you can have separate parameter files for different configurations.
- + □ **QEMM's Analysis feature can help you fine-tune memory management for your particular system and programs.** You can run your programs and QEMM will keep track of the memory they use. The Analysis feature will recommend any additional areas of upper memory that can be used as High RAM. The Analysis procedure can also tell you what areas are being accessed by your programs and therefore should not be used as High RAM. This procedure can help you easily troubleshoot any problems that might occur if a program or hardware needs access to particular upper memory addresses.
- + ✓ □ **QEMM's QSETUP program provides an easy way to turn QEMM's various features on and off, troubleshooting tips and an editor for CONFIG.SYS and AUTOEXEC.BAT.**
- + □ **Optimize processes batch files called from AUTOEXEC.BAT, loading programs specified in the batch files into High RAM when appropriate.**
- + □ **Optimize allows you to play "what if" games with the loading order of device drivers and TSRs.** This allows you to determine whether changing the loading order will give you more free memory.
- + □ **QEMM includes advanced utilities that allow you to manipulate EMS handles.** Advanced users may want to allocate EMS memory temporarily to prevent it from being taken by programs that use or allocate it indiscriminately. The EMS utilities also provide status information about EMS handles.
- + □ **QEMM lets you dynamically add DOS FILES and BUFFERS from the DOS prompt for programs that may need more of these resources.**



## Conventional Memory Gains

## About This Manual

- + ✓ □ QEMM's DPMI Host allows you to specify the amount of virtual memory to be used for programs that use DPMI services.

QEMM brings memory management to all PCs running DOS versions 3 to 6. The conventional memory gains that QEMM 7 achieves differ among DOS versions.

A new QEMM 7 feature, DOS-Up, loads the DOS kernel, DOS data, the command processor, and DOS FILES, BUFFERS, FCBS, LASTDRIVE and STACKS into upper memory. For DOS 5 and DOS 6 users, this results in a conventional memory savings of 7K for systems configured to load DOS in the HMA (high memory area) and approximately 70K when DOS is not loaded in the HMA. For DOS 3 and 4 users, DOS-Up results in a conventional memory gain of 50K-70K, depending on your configuration.

So you can better understand QEMM 7's capabilities, we have included comparisons of MS-DOS 6 running its MemMaker program and DOS 6 with QEMM 7 for three different PC brands: IBM Model 55sx, Compaq Prolinea and Dell 450DE. You will find these comparisons in **Appendix C**.

QEMM is a collection of programs that help you get the best use of your PC's memory. Each chapter of this guide covers a particular QEMM program or feature. If you are a non-technical PC user, you only need to read **Chapter 2** which tells you how to install QEMM. If you ever need detailed information about QEMM, the remaining chapters of this User's Guide are your references. At the beginning of each chapter is a brief description of what the chapter is about and why you would need to read the chapter.

If you encounter any problems using QEMM, please refer to **Appendix A** for troubleshooting information. **Appendix B** contains a list of in-depth technical "README" files that are copied to your hard disk during QEMM's installation.

❗ **IMPORTANT:** For late-breaking information that did not make it into this guide, see the READ.ME file located in the QEMM directory.

This manual is organized as follows:

**Section 1 Using QEMM** - a guide to installing and using QEMM's basic features.

- **Chapter 2** tells you how to install QEMM and use the QSETUP program to change QEMM's options.
- **Chapter 3** tells you how to use the Optimize program.
- **Chapter 4** describes DOS-Up, QEMM's feature that loads parts of DOS into upper memory.

- ❑ **Chapter 5** describes Stealth ROM, the feature that creates additional High RAM by hiding ROMs. This chapter also describes Stealth DoubleSpace, the feature that frees up additional memory by hiding DOS 6's DoubleSpace device driver.
- ❑ **Chapter 6** describes VIDRAM, the program that extends conventional memory by as much as 96K for running DOS text-based programs.

**Section 2 Technical Reference** - a guide to QEMM's technical features, mainly for advanced users who want to fine-tune memory.

- ❑ **Chapter 7** is a technical reference of the optional parameters to QEMM's device driver.
- ❑ **Chapter 8** is a technical reference for the LOADHI program that QEMM uses to load items into High RAM.
- ❑ **Chapter 9** describes QEMM.COM, a program that can change QEMM's mode, and report on memory usage.
- ❑ **Chapter 10** is a technical reference for QEMM's DPMI (DOS Protected Mode Interface) host.
- ❑ **Chapter 11** describes QEMM's advanced EMS utility programs.

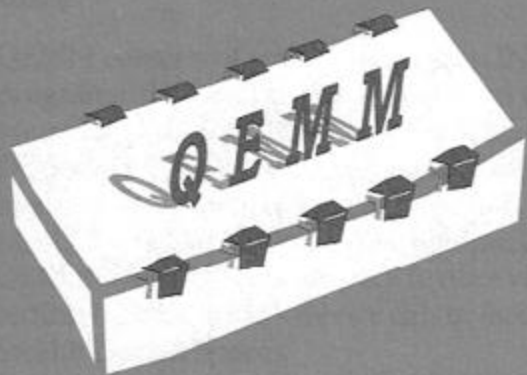
### **Section 3 Appendices**

- ❑ **Appendix A** is the troubleshooting guide.
- ❑ **Appendix B** gives a list of technical notes ("README" files) included in your QEMM directory.
- ❑ **Appendix C** compares memory management on different systems with and without QEMM.
- ❑ **Appendix D** gives a technical description of the different memory specifications supported by QEMM.

### **About Manifest**

Quarterdeck's Manifest is a standalone program that provides comprehensive memory analysis and reporting functions. Manifest is bundled with QEMM, and QEMM's installation program will automatically install Manifest in your QEMM directory. There is no separate Manifest diskette or installation procedure. For information on Manifest, see the included Manifest manual.

# Using QEMM





## Installation and Setup

### If You Already Have a Memory Manager

This chapter tells you how to install QEMM. Once you have installed QEMM, it will manage your PC's memory behind the scenes. Unless you want to explore QEMM's optional features, this is probably the only chapter you will need to read.

Installing QEMM is easy. Whether you are installing for the first time or upgrading from a previous version, you install QEMM by running the INSTALL program from your QEMM diskette. QEMM's installation takes about five minutes and requires about one megabyte of hard disk space.

QEMM consists of several programs. INSTALL will install all of QEMM's programs, then add the necessary lines to load QEMM's device drivers to your CONFIG.SYS file. One of the features that INSTALL enables is DOS-Up, QEMM's feature that loads selected parts of DOS into upper memory (see **Chapter 4** for information on DOS-Up). If you are using MS-DOS 6's DoubleSpace disk compressor, INSTALL will enable QEMM's Stealth DoubleSpace feature which frees up 40K of memory by hiding DoubleSpace's device driver (see **Chapter 5** for information on Stealth DoubleSpace).

INSTALL will offer to run Optimize—a program that examines your CONFIG.SYS and AUTOEXEC.BAT files and makes the necessary changes to load device drivers, TSRs, and selected parts of DOS into upper memory (the area between 640K and 1024K). By loading items into upper memory, Optimize frees up more conventional memory (the memory below 640K) for running programs.

Manifest is Quarterdeck's comprehensive memory reporting and analysis program. INSTALL will automatically install Manifest. Manifest analyzes your PC, and gives you detailed information on how software and hardware are using memory. In doing so, it helps you understand how you can get the best use of your PC's memory.

**If you are using the memory manager supplied with DOS version 5 or 6 or Microsoft Windows:** You may be loading the extended memory manager HIMEM.SYS and the expanded memory manager EMM386.SYS from your CONFIG.SYS file. QEMM provides all the functionality of these memory managers—and more. QEMM's installation will automatically remove DOS's memory management commands from your CONFIG.SYS and AUTOEXEC.BAT files and replace them with QEMM's commands.

**If you have a Compaq PC with more than 16 megabytes of memory:** You will need to use the HIMEM.EXE driver shipped by Compaq. Compaq's HIMEM is necessary to access memory above the 16MB limit on Compaq systems.

## Installing QEMM

**If you are using a memory manager other than DOS or QEMM:** If you are using the 386MAX, BLUEMAX, or CEMM memory manager, QEMM's installation will remove that memory manager's device driver lines from CONFIG.SYS. However, QEMM's installation will not remove any of these memory manager's load high statements from CONFIG.SYS or AUTOEXEC.BAT—you will have to do that manually (see your memory manager's manual for details on their load or load high statements). If you have any other memory manager, you should manually remove it from CONFIG.SYS. You should also remove from CONFIG.SYS or AUTOEXEC.BAT any drivers or commands that are associated with the memory manager.

After you run INSTALL you may want to scan several installation topics which appear later in this section. If you have DOS 5 or 6, Windows 3 or DESQview, you will find important recommendations about using these products with QEMM.

QEMM's installation program will ask you for your serial number, so you may want to jot it down before beginning. You will find the serial number on the QEMM diskette label. If you received an upgrade disk from Quarterdeck, you may find the serial number on a sticker attached to the invoice; we suggest you apply that sticker to the disk.

To install QEMM:

- Place the QEMM diskette into drive A.
- Type **A:INSTALL** and press **Enter** ↵



*To install from a drive other than A, substitute that drive's letter for A. If you have an LCD or gas plasma display (as on some laptops), or a monochrome monitor, we recommend that you run INSTALL in monochrome mode by typing **A:INSTALL/M**.*

You will see a screen welcoming you to the QEMM installation.

- Press **Enter** ↵ to continue.

Next you are asked to choose the type of installation: express or advanced.

- **Enter** ↵ selects an express installation. We recommend this for most users. The express installation analyzes your system and automatically installs QEMM in a directory called \QEMM using reasonable settings for your system. If you already have QEMM installed on your system, the installation program will preserve your existing configuration.
- **A** selects an advanced installation. The advanced installation gives you the opportunity to review and change the settings that the installation program recommends based on its analysis of your system. If you already have QEMM installed on your system, the installation program will offer to preserve your existing configuration.



If you indicate that you want to change your configuration, **INSTALL** will display the QEMM Setup menu (for information, see *QEMM's Setup Program*, at the end of this chapter).

- ❑ **Esc** quits the installation program without installing QEMM.

**If you have a Compaq PC with more than 16 megabytes of memory:** Select an advanced installation, and when you are asked if you want to remove HIMEM.SYS, press **Esc** to decline.

- **Press Enter ↵ to select an express installation or A to select an advanced installation. Then, follow the on-screen instructions.**

**INSTALL** copies QEMM and Manifest to your hard disk and configures QEMM.

## Optimize

Once QEMM is installed, you will see a message that the installation is complete and the installation program will offer to run **Optimize**. We strongly suggest you accept this offer. **Optimize** will set things up so TSRs, device drivers and certain parts of DOS will be loaded into High RAM to free up more conventional memory for running programs. High RAM is QEMM's name for the memory it places into available upper memory addresses (addresses between 640K and 1024K that are not being used by ROMs or adapter RAM).

To run **Optimize**, just follow the instructions on the screen. To analyze your system and make the necessary changes, **Optimize** must reboot your system one or more times; if it does not reboot when indicated, press your PC's reset button or power your PC off, then on again. For detailed information about **Optimize**, see **Chapter 3**.



*Your PC may be set up to start one of your programs (e.g., a menu program) automatically when you boot your system. If so, the program will start every time **Optimize** reboots your PC. When the program starts, just exit the program normally and **Optimize** will continue.*


## Stealth ROM

Depending on the number and size of your device drivers and TSRs, **Optimize** may not be able to fit them all into High RAM on its first pass. If this is the case, **Optimize** will test your system for compatibility with **Stealth ROM**, a QEMM feature that hides your PC's ROMs and makes their memory addresses available as High RAM (for information on **Stealth ROM**, see **Chapter 5**). If **Stealth ROM** is compatible with your system, **Optimize** will enable it and try again to load all of your TSRs and drivers into High RAM.


Once you have done the requested reboots, you will be back at the DOS prompt. Congratulations!—you have installed QEMM. You can now run your programs as usual and QEMM will handle memory management




behind the scenes. We suggest you look through the rest of this chapter for topics that pertain to your configuration.

 **IMPORTANT:** If *Optimize* does not complete successfully or if your system does not initialize properly after running *Optimize*, see **Appendix A** for Troubleshooting information.

If, for some reason, you want to load programs high manually instead of using *Optimize*, refer to **Chapter 8: The LOADHI Programs**. We do recommend that you try *Optimize* first.

 **IMPORTANT:** If you ever change your *CONFIG.SYS* or *AUTOEXEC.BAT* file, or if you add or remove hardware peripherals that have adapter cards, you should run *Optimize*. For information on running *Optimize*, see **Chapter 3**.

 **TIPS:** If you have an EGA or VGA video adapter and you routinely run large DOS text-based programs (e.g., *dBASE*), or you have an 8514A video adapter, you can make up to 96K more memory available to certain programs by using QEMM's *VIDRAM* program. See **Chapter 6** for information.

The *Manifest* program may be able to help you find out how to gain a little more High RAM. For information, see **Chapter 13: Hints in the Manifest manual**.

#### Using QEMM on Battery-powered PCs

If you have a battery-powered computer with the Suspend/Resume feature, QEMM will automatically try to support that feature. If QEMM does not recognize this capability, add the parameter **SUSPENDRESUME** to the end of the *QEMM386.SYS* line in *CONFIG.SYS*. See **Chapter 7** for additional information on the **SUSPENDRESUME** parameter.

#### Booting Without QEMM

If after rebooting, your system fails to initialize, you can recover without using a boot floppy by doing the following:

- **Reset your system.** Use the power switch if necessary.
- **Wait until you hear a beep, then press the Alt key until the boot sequence stops.** If you are using QEMM's DOS-Up feature, you will see a message asking if you want to unload it; press Esc to unload DOS-Up, then hold down Alt again.

You will see the following message:

"QEMM: Press ESC to unload QEMM or any other key to continue with QEMM."

- **Press the Esc key.**

Your system will then proceed with the boot sequence. QEMM will not be loaded—thus no programs will be loaded into High RAM, but your

**If You  
Experience Any  
Problems  
Running Your  
Programs**

**If You Ever Get  
an Exception 13  
Error Message**

system will be usable. See **Appendix A** for information on resolving the problem.

QEMM works very hard to give you as much memory as possible for running programs on your PC. On rare occasions, a program may try to access an area of memory that QEMM is using as High RAM or for expanded memory mapping. If after installing QEMM, one of your programs does not work or display properly, you can usually remedy the situation by following QEMM's Analysis procedure. For information, see the troubleshooting guide in **Appendix A**.

There is a remote possibility that at some point in the future you will see an "Exception 13" error message like the one shown below.

Exception 13: Your PC's main processor has received an invalid instruction due to an error in one of your programs, a conflict between two programs, or a conflict between a program and a hardware device.

This is not a QEMM error, but QEMM is reporting the error so you will know what happened and so you can terminate your program. Without QEMM, your system would have crashed without warning or explanation. It is likely that the system is unstable now and should be rebooted.

We suggest you write down the information below, and see the QEMM manual's troubleshooting section for information on resolving this problem. If the suggestions in the QEMM manual do not help, please consult the publisher of the program you were running when this error occurred.

Exception #13 at 2400:0103, error code: 0000  
AX=0000 BX=0000 CX=0000 DX=0000 SI=FFFF DI=0000 BP=0000  
DS=2400 ES=2400 SS=2400 SP=FFFE Flags=7246  
Instruction: A5 CC BA 80 1F E8 79 F9 A1 BE 1E B1 04 D3 E0

Press T to (T)erminate the program or R to (R)eboot. If your system does not respond, switch your machine off and on again.

**An exception 13 message**

Without QEMM, if a program causes your system to malfunction, you often have no idea why. What you do not see is that before the malfunction, your PC's main processor often puts out an emergency parcel of information about the condition that caused the malfunction. Exception 13s are typically caused by a bug in a program, or may result from a conflict between two programs or between a program and a hardware device.

Without QEMM, you would never see this information and your system would probably crash. QEMM watches for this warning from your

system and passes the information on to you. Moreover, QEMM tries to stop the illegal operation before it completely hangs your system, giving you the opportunity to terminate the program. See **Appendix A** for further information on Exception 13 error messages.

QEMM's installation makes changes to your CONFIG.SYS and AUTOEXEC.BAT files, and to any other batch file called by your AUTOEXEC.BAT file. This section is for those who want to know exactly what these changes are. If you are not interested in these technical details, feel free to skip this section.

In some of the sample lines below, you will see the parameter **/R:n** which is used to load an item into a specified High RAM region. On your system, the **n** is replaced by a number indicating the region.

QEMM's installation makes the following changes to your CONFIG.SYS file:

- ❑ If you are using QEMM's DOS-Up feature, the following line is added at the beginning of CONFIG.SYS to prepare your system for parts of DOS to be loaded into upper memory:

```
DEVICE=C:\QEMM\DOSDATA.SYS
```

Next, QEMM's installation adds the following line:

```
DEVICE=\QEMM\QEMM386.SYS R:n RAM
```

This is QEMM's device driver line; it is the line that loads QEMM whenever you boot your PC. Depending on your configuration, you may see additional parameters (e.g., ST:M or ST:F for Stealth ROM). For information on QEMM's parameters, see **Chapter 7**.

- ❑ If you are using DOS-Up, the following line which loads the various parts of DOS into upper memory, appears directly after your QEMM386.SYS line:

```
DEVICE=C:\QEMM\DOS-UP.SYS @C:\QEMM\DOS-UP.DAT
```

- ❑ The installation program also adds the following line to load QEMM's DPML driver, which supports programs that use the DOS Protected Mode Interface:

```
DEVICE=C:\QEMM\LOADHI.SYS /R:n C:\QEMM\QDPML.SYS
```

- ❑ The following syntax is added to the beginning of other device driver lines in CONFIG.SYS:

```
DEVICE=C:\QEMM\LOADHI.SYS /R:n
```

This command tells QEMM to load the device driver directly following this command into High RAM. (If Optimize has determined

that a particular driver will not fit into upper memory, this syntax will not be added to that driver's line.)

For example, if before installing QEMM, your CONFIG.SYS file contained the line `DEVICE=C:\DOS\ANSI.SYS`, Optimize would change it to read as follows:

**DEVICE=C:\QEMM\LOADHI.SYS /R:n C:\DOS\ANSI.SYS**

- If you are using DOS-Up, Optimize will add QEMM's LOADHI command to your SHELL line to load your command processor high (if you do not have a SHELL line, Optimize will add one for you). For example, if you have the following line in your CONFIG.SYS file

**SHELL=C:\DOS\COMMAND.COM C:\DOS\ /P**

Optimize will change it to read as follows:

**SHELL=C:\QEMM\LOADHI.COM /SHELL /R:x  
C:\DOS\COMMAND.COM C:\DOS\ /P**

- If you are using DOS version 5 or 6, Optimize will add the following line to your CONFIG.SYS file if it is not already there:

**DOS=HIGH**

This is a DOS command that loads part of DOS's kernel and DOS buffers into the HMA (the first 64K of extended memory).

- If you are using MS-DOS 6's DoubleSpace disk compressor, QEMM's installation will add the following line to enable the Stealth DoubleSpace feature:

**DEVICE=C:\QEMM\LOADHI.SYS C:\QEMM\ST-DBL.SYS**

- As mentioned earlier in this chapter, QEMM's installation will remove device driver lines for other memory managers from CONFIG.SYS.

Optimize makes the following changes to AUTOEXEC.BAT and to any batch file called by AUTOEXEC.BAT:

- QEMM's directory is added to your PATH statement.
- The following syntax is added to the beginning of lines that load TSRs (i.e., memory-resident programs) high:

**C:\QEMM\LOADHI /R:n**

This command tells QEMM to load the TSR directly following this command into High RAM. For example, if before installing QEMM, your AUTOEXEC.BAT file contained the line `\NET\NETX.COM`, Optimize would change it to read as follows:

**C:\QEMM\LOADHI /R:n C:\NET\NETX.COM**

### Using QEMM with DOS 5 or 6 or Microsoft Windows

- ❑ If you are using Microsoft Windows, QEMM makes a small change to Windows' SYSTEM.INI file to make Windows run optimally with QEMM.

There are a few considerations with respect to running QEMM with DOS 5 or 6 or Microsoft Windows:

- ❑ If you install Microsoft Windows *after* QEMM, you need to run a special program to make Windows compatible with QEMM. Just go to the Windows directory, then type **QWINFIX** and press **Enter** ↵.
- ❑ **Using the HMA.** QEMM creates a special area called the HMA (High Memory Area) in the first 64K of extended memory. Certain programs can use the HMA, instead of loading into conventional memory. This frees up some conventional memory so you can run larger programs. There are some restrictions on using the HMA: only one program can occupy the HMA, and there are only a few programs that can use it at all (e.g., DOS, DESQview, DESQview/X, Novell's XMSNET).

QEMM's installation normally adds the DOS command **DOS=HIGH** to your CONFIG.SYS file. This line causes DOS to load part of its code and buffers into the HMA. Depending on how many buffers you have allocated in CONFIG.SYS, this can save you up to 64K of conventional memory. Though QEMM's DOS-Up feature can load these parts of DOS into upper memory, it is generally preferable to take advantage of the HMA and leave as much upper memory free for TSRs and device drivers as possible. QEMM's DOS-Up will still load additional parts of DOS into upper memory. We normally recommend that you use **DOS=HIGH**, unless you use Quarterdeck's DESQview or DESQview/X (see below).



**If you have DR DOS version 6 or above:** Use **HIDOS=ON** instead of **DOS=HIGH**.

- ❑ **Using DOS's HIMEM.SYS.** QEMM includes all the capabilities of the HIMEM.SYS extended memory driver provided with DOS versions 5 and 6 and Microsoft Windows. As stated earlier, there is normally no need to use HIMEM.SYS if QEMM is installed. However, if you have a Compaq PC with more than 16 megabytes of memory, you should use Compaq's HIMEM.EXE driver.
- ❑ **Using DOS's or Windows' EMM386.** QEMM and EMM386 are both 80386 expanded memory managers. QEMM's installation will disable EMM386.SYS by removing it from your CONFIG.SYS file.

### Using QEMM with DESQview

When you start DESQview, it will detect the presence of QEMM and automatically activate its 80386-specific features.

To get maximum memory size for programs running inside DESQview, you will probably want to use QEMM's Stealth ROM feature. If, as part of



INSTALL, Optimize did not test your system for compatibility with Stealth ROM, you can do so at any time by rerunning Optimize as follows:

- **Type Optimize /Stealth and press Enter ↵**

If you are using DOS 5 or 6: DESQview and DESQview/X can load over 63K of their code into the HMA. If you regularly use one of these programs, we suggest you let them use the HMA, because they make the most efficient use of that area. To do that, just omit the DOS=HIGH statement from your CONFIG.SYS file, and DESQview/X will automatically use the HMA. You can then use DOS-Up to load selected parts of DOS into upper memory. To ensure that DOS-Up is enabled on your system, run QEMM's Setup program (see *QEMM's Setup Program* later in this chapter).

When you use DESQview and QEMM together, be aware that there are 386-specific commands controlled by DESQview's Change A Program and Tune Performance menus. See your DESQview manual for information on using DESQview with QEMM.

QEMM has a Setup program you can use to change QEMM's configuration easily. When you perform a non-express installation of QEMM, you are asked if you want to change QEMM's options. If you select Yes, you will automatically go to the Setup program. You can also run the Setup program from the DOS prompt any time you want.

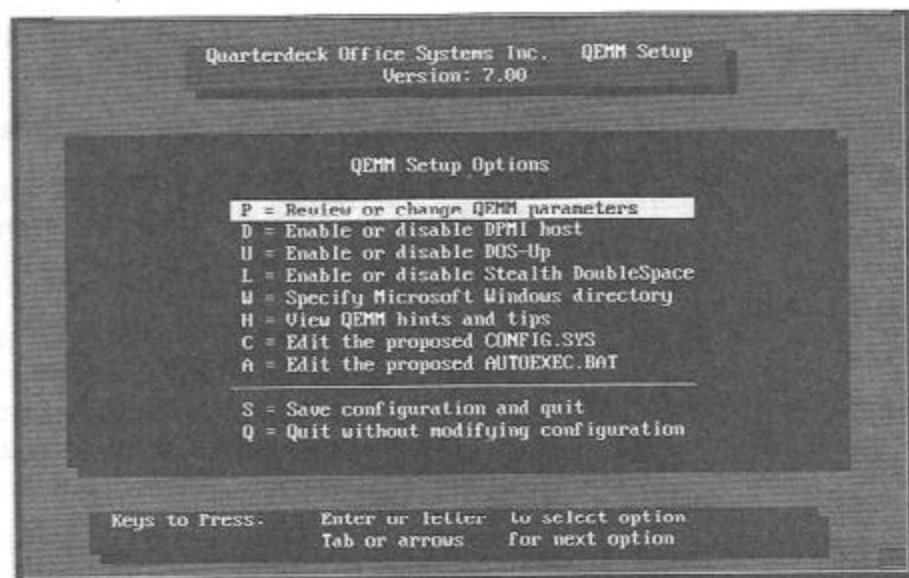
To run the Setup program:

- **Go to the QEMM directory by typing CD QEMM and pressing Enter ↵.**
  - **Type QSETUP and press Enter ↵.**
- You will see a welcome screen.
- **Press Enter ↵ to display the QSETUP menu.**

The Setup program has the following options:

- **Review or change QEMM's parameters.** This option displays another menu of QEMM features you can change. When you select an item from the menu, you will see a description of what the item does so you can review it before making the change. This menu contains more than one page; use the Page Up and Page Down keys to switch between pages. For information on QEMM's parameters, see **Chapter 7**.
- **Enable or disable DPMI host** lets you enable or disable the DOS Protected Mode Interface for programs that support DPMI. For information on DPMI, see **Chapter 10**.





### The QSETUP menu

- ❑ **Enable or disable DOS-Up** lets you turn on or off QEMM's feature that loads selected parts of DOS into upper memory. For information on DOS-Up, see **Chapter 4**.
- ❑ **Enable or disable Stealth DoubleSpace** turns on or off the feature that hides MS-DOS 6's DoubleSpace device driver, freeing up 40K of RAM. For information on Stealth DoubleSpace, see **Chapter 5**.
- ❑ **Specify Microsoft Windows directory** lets you specify the path to Microsoft Windows if you are using Windows on your system. When you give the correct path to Windows, then save your configuration, Windows' SYSTEM.INI file will be modified slightly to make Windows compatible with QEMM.
- ❑ **View QEMM hints and tips** displays troubleshooting tips and useful information about running QEMM with certain hardware and software.
- ❑ **Edit the proposed CONFIG.SYS** lets you edit your CONFIG.SYS file. When you are editing the file, you can press F1 to get online Help about the editor's features.
- ❑ **Edit the proposed AUTOEXEC.BAT** lets you edit your AUTOEXEC.BAT file. When you are editing the file, you can press F1 to get online Help about the editor's features.
- ❑ **Save configuration and Quit** saves any changes you have made and exits the Setup program.
- ❑ **Quit without modifying configuration** exits the Setup program, abandoning any changes you have made.



## The Optimize Program

### When to Run Optimize

### How to Run Optimize

Optimize is a program that determines how to load TSRs and device drivers into upper memory. When you install QEMM, the installation program offers to run Optimize for you. You should run Optimize again if you add new hardware devices or modify your AUTOEXEC.BAT or CONFIG.SYS files. This chapter is for users who need to run Optimize after changing their configuration.

Optimize determines the most efficient way to use your PC's memory. Optimize automatically sets things up so that the TSRs and device drivers you load from your AUTOEXEC.BAT and CONFIG.SYS files are loaded into High RAM (the memory that QEMM places between 640K and 1024K) instead of into conventional memory (the area between 0K and 640K). This frees up more conventional memory so you can:

- ❑ Run larger programs that would not fit in conventional memory before.
- ❑ Add TSRs you have been doing without.
- ❑ Speed up programs that were formerly storing their data on disk because there was not enough available memory to hold the data.
- ❑ Increase the memory available to applications running under multitasking control programs (e.g., DESQview, Microsoft Windows).

When you install QEMM, the installation program offers to run Optimize for you. You can also run Optimize later if necessary. You should run Optimize after:

- ❑ You add or remove a hardware component.
- ❑ You add or remove a device driver or TSR in CONFIG.SYS or AUTOEXEC.BAT.
- ❑ You change the loading order of TSRs, or change a TSR command line.
- ❑ You change the QEMM386.SYS device line in CONFIG.SYS manually or with QEMM's Setup program.

Optimize will properly handle conditional statements in AUTOEXEC.BAT.

To run Optimize:

- **Type Optimize followed by any optional parameters (explained later in this chapter) and press Enter ↵.**



*If you have an LCD, gas plasma or monochrome display, you can run Optimize in monochrome mode by typing **OPTIMIZE /M** [...other parameters...].*



*If you are sure you want Optimize's default settings, you can speed up the Optimize process by typing **OPTIMIZE /Q**.*

Follow the on-screen instructions; you can simply press the Enter ↵ key repeatedly to accept Optimize's defaults. During the optimization process your PC must be rebooted two or more times. If it does not reboot when indicated, press the reset button or power the computer off, then on again.

If all of your TSRs, device drivers and selected parts of DOS will not fit in High RAM and you are not already using QEMM's Stealth ROM feature, Optimize will offer to test your system for Stealth compatibility. Stealth ROM creates additional High RAM by mapping memory into ROM addresses (for more information on Stealth ROM see **Chapter 5**).

During the Optimize process, you will see a screen that tells you how many TSRs and device drivers were loaded high. When Optimize is completed, it displays a screen telling you how much memory you have saved. If you would like details on the items that are loaded high, type **LOADHI** and press Enter ↵ at the DOS prompt.

#### If You Encounter Problems

If Optimize does not complete successfully or if there are errors when you boot your system after running Optimize, there is a chance that Optimize loaded a program into an upper memory area used by an adapter (i.e., a network card) or another program. See Appendix A for information on running Optimize /AUTO.

#### Optimize Options

During Optimize's Setup and Analysis phases, you will see **O** for **Options**, at the bottom of the screen. If you type **O** you will see a list of further options.

If you select Options during the Setup phase, you will see a screen listing the following options:

- ❑ **Enter** causes Optimize to detect upper memory areas that should be excluded from QEMM's management. This option is a troubleshooting procedure that you can use if you ran Optimize and it did not complete successfully. This option is analogous to typing **OPTIMIZE /AUTOEXCLUDE**. For information, see the **AUTOEXCLUDE** parameter later in this chapter and the section on Optimize /AUTOEXCLUDE in **Appendix A** for details.
- ❑ **F3** tells Optimize to test your system for compatibility with QEMM's Stealth ROM feature.
- ❑ **Esc** resumes the normal Optimize process.

If you select **Options** during Optimize's Analysis phase, you can choose the following:

## Forcing Stealth ROM Testing

## How Optimize Works

- ❑ **Region Layout** shows which upper memory addresses will be used to load TSRs and device drivers.
- ❑ **Modify Data** lets you view and modify the data Optimize uses to load programs high. You can use this screen to load a program low if you know it does not work when loaded high (also see the **COMMANDFILE** parameter later in this chapter). Do not alter the other fields on this screen unless you know the possible consequences.
- ❑ **What-If** lets you see if you can gain memory by changing the loading order of the programs in **CONFIG.SYS** and **AUTOEXEC.BAT**. You cannot actually put these changes into effect at this point (and such changes are not always advisable due to dependencies of one program upon another), but you can quickly test different possibilities to see if changing the loading order of your programs in **CONFIG.SYS** or **AUTOEXEC.BAT** is worthwhile.

Optimize offers to test for Stealth ROM compatibility only if it cannot fit all the items in your **CONFIG.SYS** and **AUTOEXEC.BAT** files into High RAM. Stealth ROM can typically create 48K-115K of additional High RAM. For a discussion of how Stealth ROM works, see **Chapter 5**.

If you use **DESQview** or **DESQview/X**, you may want to enable Stealth ROM even if Optimize did not recommend Stealth ROM testing. Stealth ROM will usually give **DESQview** and **DESQview/X** users more memory for each window. If you have **DESQview** or **DESQview/X**, and you are not already using Stealth ROM, we recommend you Optimize using Stealth ROM as follows:

- Switch to the **QEMM** directory by typing **CD \QEMM** and pressing **Enter** ↵.
- Type **Optimize /STEALTH** and press **Enter** ↵.

For those who are interested, here is how Optimize works:

First, Optimize searches your **CONFIG.SYS** and **AUTOEXEC.BAT** files and any batch files called by **AUTOEXEC.BAT** for TSRs (memory resident programs) and device drivers. Optimize makes some changes to **CONFIG.SYS** and **AUTOEXEC.BAT** that cause each item to be measured in terms of its initialization size and its size when resident. Your original files are renamed to **CONFIG.QDK** and **AUTOEXEC.QDK**.

Next, Optimize determines how much memory each item needs. Some TSRs and device drivers need more memory to initialize than they do once they are resident in memory, and Optimize notes these differences. Then, Optimize determines the most efficient way to load items into High RAM by testing all possible locations (there can be thousands or millions of possibilities). The object is to leave as much free conventional memory as possible.

## Undoing an Optimize

## Optimizing Embedded Batch Files

If an item needs a larger amount of memory to initialize than is available in a contiguous chunk, Optimize may use a feature called Squeeze. Squeeze will temporarily map memory over a ROM, adapter RAM, excluded area or the EMS page frame to make a large enough contiguous area, then initialize the item in that area. The Squeeze process is safe—it will never squeeze over memory that the item uses while initializing.

If the items will not all fit into High RAM, Optimize offers to test your PC for compatibility with QEMM's Stealth ROM feature. Stealth ROM temporarily moves ROMS to make more upper memory addresses available for loading items high. If your system is Stealth ROM compatible, Optimize will use it to create more High RAM, then restart the Optimize process.

Next, Optimize modifies CONFIG.SYS, AUTOEXEC.BAT and any batch files called by AUTOEXEC.BAT to load the TSRs, device drivers and selected parts of DOS into High RAM. To load a device driver high, Optimize adds `DEVICE=C:\QEMM\LOADHI.SYS /R:n` to the beginning of the item's line. To load a TSR high, Optimize adds `C:\QEMM\LOADHI /R:n` to the beginning of the item's line. `LOADHI` is the QEMM program that actually loads items high. The `r:n` parameter (where `n` is a number) specifies which High RAM region should be used to load the item.

During the optimization process your PC must be rebooted two or more times to make the necessary changes to CONFIG.SYS and AUTOEXEC.BAT and load the various items high.

Once Optimize is finished, the items will automatically load into High RAM whenever you boot your PC.

If for any reason you want to restore your CONFIG.SYS and AUTOEXEC.BAT files to their pre-Optimized states, follow these steps:

### ■ Type **UNOPT** and press **Enter**.

CONFIG.SYS and AUTOEXEC.BAT will be restored to the state they were in before you last ran Optimize. If your AUTOEXEC.BAT file calls another batch file, that batch file will be restored to its pre-optimized state.

Optimize can detect batch files embedded in AUTOEXEC.BAT. To take advantage of this capability, be sure that any line in the AUTOEXEC.BAT that starts another batch file is preceded by the `CALL` command (supported by the COMMAND.COM shell for DOS versions 3.30 and above). You can nest batch files as many levels as you want (e.g., AUTOEXEC.BAT can call a batch file that calls a batch file, etc.).



If you have device drivers or TSRs in CONFIG.SYS or AUTOEXEC.BAT that you would like to exclude from the Optimize process (i.e., you want them to load into conventional memory), you can do so as follows:

- Create an ASCII text file called OPTIMIZE.EXC in the QEMM directory.
- List each program you want to exclude on a separate line in OPTIMIZE.EXC. When you specify a command or program in this file, use only the first part of the filename; do not specify the drive, path, extension or any parameters (e.g., for the file C:\UTILITY\THISPROG.EXE, you would specify only THISPROG).
- Save the file.
- Run Optimize.

Optimize will exclude the programs you listed from being loaded high. Also see the COMMANDFILE parameter in the next section.

Listed below are the parameters you can use with Optimize. Most of these will be of interest only to advanced users. Some parameters have an abbreviation you can use instead of the full name; abbreviations are listed in parentheses below. Options described as "internal" are for Optimize's use—there is no reason for you to specify them.

You can display the list of parameters by typing **OPTIMIZE /?** or **OPTIMIZE /HELP**.

#### **/AUTOEXCLUDE (/AUTO)**

tells Optimize to load all CONFIG.SYS and AUTOEXEC.BAT items low temporarily, then look for areas of upper memory that are accessed during the boot process. Optimize will then offer to exclude these areas from QEMM's management. The purpose of this parameter is to find upper memory areas that are accessed during the boot process so you can prevent QEMM from loading other items there. For example, a network adapter may use a particular area of upper memory, and if QEMM places High RAM there, the network will not work.

When you use **OPTIMIZE /AUTOEXCLUDE**, you will be asked to power your PC off then on. Then, Optimize will present you with a list of the accessed upper memory areas and tell you which programs use them. You can then decide whether to have Optimize exclude any of those areas from QEMM's management. We suggest you use **AUTOEXCLUDE** if you previously ran Optimize and it did not complete successfully, or if your system did not initialize properly after running Optimize. For more information on using **OPTIMIZE /AUTOEXCLUDE**, see **Appendix A**.

### **/BOOT:drive (/B:drive)**

tells Optimize where to find the CONFIG.SYS and AUTOEXEC.BAT files. The default is the root directory of your hard disk, usually the C drive. You use the BOOT parameter to tell Optimize where to look for these files (replace **d** with the drive letter) when you boot from a floppy or a drive other than C.

### **/COMMANDFILE:filename (/CMD:filename)**

specifies a file that lists commands or programs which Optimize should not process. Use this parameter when you do not want Optimize to attempt to load certain programs high (e.g., programs that you know will not run properly when loaded high or commands specific to an alternate command processor such as 4DOS or NDOS).

The file of commands and programs should be an ASCII file with one command or program per line in the file. When you specify a command or program in this file, use only the first part of the filename; do not specify the drive, path, extension or any parameters (i.e., for the file C:\UTILITY\THISPROG.EXE, specify only THISPROG). If you insert a pound sign (#) at the beginning of a line, that line will be ignored.

If you are using a command processor other than COMMAND.COM (e.g., 4DOS, NDOS) and your AUTOEXEC.BAT file contains commands specific to that command processor, Optimize may misinterpret the commands. QEMM includes a file called 4DOS.CMD which contains a list of 4DOS- and NDOS-specific commands. If you are using 4DOS or NDOS, we recommend adding /CMD=C:\QEMM\4DOS.CMD on the Optimize command line—this parameter tells Optimize to ignore the commands listed in this file.

There is an alternative method for excluding commands or programs from being processed by Optimize. Just create an ASCII file called OPTIMIZE.EXC in the QEMM directory, and in it, include the commands and programs you want Optimize to ignore. Use the same format as described above. The advantage of using OPTIMIZE.EXC is that you do not have to use the /CMD parameter; whenever you run Optimize, it will check this file. If you have 4DOS or NDOS, you may want to copy 4DOS.CMD to OPTIMIZE.EXC so the 4DOS or NDOS commands will be automatically ignored whenever you run Optimize. If you use the OPTIMIZE.EXC file and use the CMD parameter to specify another file, the programs in both files will be ignored by Optimize.

Placing a program in a command file will force a program to load low. So, if a line already has the LOADHI command, the command file will force Optimize to strip it off.

**/DIRECTORY (/DIR)**

is a command used internally by Optimize.

**/EMM:filename**

specifies the name of the memory manager you are using in case Optimize does not recognize the presence of the memory manager's driver. You should use this parameter if you have for some reason renamed the QEMM386.SYS driver.

**/FILE (/F)**

is a command used internally by Optimize.

**/FINISHED (/DONE)**

is a command used internally by Optimize.

**/HELP**

tells Optimize to list a brief description of each of its parameters.

**/LARGE (/LA)**

tells Optimize to display its screens using 43- or 50-line mode if your video adapter supports this feature.

**/LOADHIONLY (/L)**

tells Optimize to analyze or change only those lines that already contain a LOADHI command.

**/LOADLOW (/LOW)**

causes Optimize to remove all LOADHI statements and /REGION (/R) parameters. /LOADLOW forces all programs to load into conventional memory.

**/MONO (/M)**

causes Optimize to run in monochrome mode. This parameter is useful for running Optimize on systems with an LCD, monochrome or gas plasma display.

**/NOSQF (/NF)**

tells Optimize you do not want LOADHI to Squeeze using the page frame during program initialization.

### **/NOSQT (/NT)**

tells Optimize that you do not want LOADHI to Squeeze using temporary mapped memory during program initialization.

### **/NOSTEALTH (/NOST)**

tells Optimize not to test your PC for compatibility with Stealth ROM. Without this parameter, Optimize will ask if you want to test for Stealth ROM compatibility if all items will not fit into upper memory. When used with the /QUICK parameter, the /NOSTEALTH parameter speeds up optimization by eliminating the prompt for Stealth ROM testing.

### **/NOSYNC (/N)**

tells Optimize your display does not require synchronization. You should use this parameter if you have a CGA display that does not require synchronization (i.e., it does not have "snow effects").

### **/PATH (/P)**

tells Optimize to add the QEMM directory to the PATH environment variable in your AUTOEXEC.BAT file. This is helpful when the LOADHI lines in your AUTOEXEC.BAT file become too long (i.e., more than 128 characters). This can occur when Optimize adds the LOADHI command to a long line in AUTOEXEC.BAT.

### **/QUICK (/Q)**

tells Optimize to use the default options.

### **/RESPONSE (/RF)**

tells Optimize to prepare a "response file" for subsequent use by the LOADHI programs. /RESPONSE may be followed by a colon and a filename. The default filename is LOADHI.RF. This response file is essentially a database defining your system's use of High RAM. It will contain the detailed specifications for each device driver and TSR to be loaded into High RAM.

This option will be valuable if your PC is connected to a network and your AUTOEXEC.BAT file calls public batch files located on the network. Such public batch files are commonly called by multiple network users. LOADHI information generally must not be specified inside the public batch file, as each PC will likely require a different set of LOADHI parameters. By giving Optimize the /RESPONSE parameter, network PC users can create separate, private response files that contain the LOADHI information specific to each PC.

You may simply wish to use the /RESPONSE parameter to remove the clutter of details from CONFIG.SYS and AUTOEXEC.BAT, especially if the LOADHI command lines become too long to be processed correctly.

#### **/SEGMENT (/SEG)**

is a command used internally by Optimize.

#### **/STEALTH (/ST)**

causes Optimize to test your system for compatibility with Stealth ROM. Optimize will first try the Stealth ROM mapping method and, if it is appropriate for your system, Optimize will implement that method. If the mapping method is not right for your system, Optimize will try the frame method.

#### **/STPASSDONE**

is a command used internally by Optimize.

#### **/?**

tells Optimize to list its command line parameters.





## DOS-UP: Loading Parts of DOS into Upper Memory

DOS-Up is a QEMM feature that frees up conventional memory by loading selected parts of DOS into upper memory. QEMM's installation program enables DOS-Up on your system by default. Read this chapter if you want technical details on DOS-Up or you need to enable or disable it.

QEMM's DOS-Up feature loads selected parts of DOS into High RAM. Without DOS-Up, these parts of DOS would be loaded in conventional memory (if you have DOS 5 or 6, some but not all of these parts of DOS can be loaded into the HMA). Depending on how your system is configured, DOS-Up can free up 7K to 70K of conventional memory, leaving more room to run your programs.

DOS-Up is normally installed and enabled when you install QEMM. This chapter describes DOS-Up and QEMM's DOS Resource programs which provide an alternative method for loading parts of DOS high. We also tell you how to enable DOS-Up if you did not do so at installation time.

DOS-Up can load the following components of DOS into High RAM:

- ❑ The DOS kernel. This is the "core" of DOS's code and is contained in the system file, MSDOS.SYS (IBMDOS.COM if you have IBM DOS).
- ❑ DOS data (contained in MSDOS.SYS or IBMDOS.COM)
- ❑ DOS FILES, BUFFERS, STACKS, FCBS and LASTDRIVE. The sizes of these resources are normally specified in CONFIG.SYS, and have default values, if not.
- ❑ The DOS command processor, COMMAND.COM.

The amount of conventional memory freed up by DOS-Up depends on your system configuration and what version of DOS you are using. For example, each DOS BUFFER takes up approximately half a K of memory. If you have a BUFFERS=30 statement in your CONFIG.SYS, DOS-Up can save 15K of conventional memory by loading the BUFFERS into High RAM.

DOS versions 5 and 6 can load portions of DOS into the HMA (the first 64K of extended memory). To enable this feature, you need the line DOS=HIGH in your CONFIG.SYS file (QEMM's installation may have already put it there for you). Even if you use this feature of DOS, we still recommend using DOS-Up—it will load additional portions of DOS into High RAM. The HMA can only be used by a single program, and as of this writing, there are very few programs besides DOS, DESQview and DESQview/X that will use that area.



*For those who are interested, DOS-Up can load the following parts of DOS into High RAM in addition to what DOS=HIGH loads into the HMA: 5K of DOS data, 2.5K more of the command processor, FILES, STACKS, FCBS and LASTDRIVE.*

## If You Have DESQview or DESQview/X

## Enabling or Disabling DOS-Up

## DOS-Up Drivers

If you are using DOS version 3 or 4, DOS-Up will be a great benefit because these versions of DOS load all parts of DOS into conventional memory. DOS-Up can free up to 70K of conventional memory on DOS 3 and 4 systems.

In general, DESQview and DESQview/X make more efficient use of the HMA than DOS—they are designed to load over 63K of code into this 64K area. That is why we recommend letting DESQview or DESQview/X use that area. DESQview or DESQview/X will automatically use the HMA if no other program is already using it. To have DESQview or DESQview/X use the HMA, just omit the DOS=HIGH statement from CONFIG.SYS. Then to load selected parts of DOS high, enable DOS-Up (see below) and run Optimize.

To enable or disable DOS-Up:

- Start QEMM's Setup program by typing **QSETUP** and pressing **Enter**.
- Press **Enter** to go to the Setup menu.
- Select **Enable or disable DOS-Up**, then select **Yes** to enable it, or **No** to disable it.
- Run Optimize (see Chapter 3).

DOS-Up consists of two device drivers called DOSDATA.SYS and DOS-UP.SYS which are loaded in your CONFIG.SYS file. For DOS-Up to be effective, both drivers must be loaded.

The DOSDATA.SYS device driver prepares your system for the DOS Data area to be loaded into High RAM. When DOS-Up is enabled, the line

**DEVICE=C:\QEMM\DOSDATA.SYS**

should appear as the first device driver line in your CONFIG.SYS file.

The DOS-UP.SYS device driver is responsible for loading DOS data, the DOS kernel, BUFFERS, FILES, FCBS, STACKS, and LASTDRIVE high. QEMM's Optimize program will determine where to load these items in High RAM. The line

**DEVICE=C:\QEMM\DOS-UP.SYS @C:\QEMM\DOS-UP.DAT**

should appear immediately after the QEMM386.SYS device driver line in your CONFIG.SYS file. The file DOS-UP.DAT contains instructions that tell DOS-Up where in High RAM to load the various parts of DOS. You should not modify that file.

DOS-Up makes one additional change to your CONFIG.SYS file. To load the command processor high, DOS-Up adds the **LOADHI /SHELL**

command at the beginning of your command processor line. If you do not have a command processor line, DOS-Up will add one. For example, if your original command processor line was:

```
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /P
```

DOS-Up will modify it to read:


```
DEVICE=C:\QEMM\LOADHI.COM /SHELL /r:x  
C:\DOS\COMMAND.COM C:\DOS\ /P
```

where x is the number of the High RAM region into which COMMAND.COM is loaded. (The entire command above will appear on a single line in CONFIG.SYS.)

If you ever want to boot with out DOS-Up, follow these steps:

- **Reset your system.** Use the power switch if necessary.
- **Wait until you hear a beep, then press the Alt key until the boot sequence stops.**
- **You will see a message asking if you want to unload DOS-Up. Press Esc to unload DOS-Up**

Earlier versions of QEMM did not feature DOS-Up, but did include programs that could load certain DOS resources high. To be backward-compatible with earlier versions of QEMM, we have included these DOS resource programs with QEMM version 7.

 *DOS-Up provides a better and more comprehensive method for loading parts of DOS high, so we recommend using DOS-Up whenever possible. If you are using DOS-Up, you will not need to use the programs described in the remainder of this chapter.*

QEMM includes four programs you can use to load additional FILES (file handles), BUFFERS, FCBS (file control blocks), and logical disk drives on the fly, without modifying CONFIG.SYS or rebooting your PC. QEMM's programs to augment these resources are BUFFERS.COM, FILES.COM, FCBS.COM and LASTDRIV.COM.

These programs all work the same way. If you run the program without parameters, you get a report on the resource's current allocation. Each program can take a numeric parameter to increase the resource. You can also use them along with the LOADHI command to load the resource high, thus freeing up more conventional memory to run DOS programs. The additional resources will stay in effect until you power off your PC or reboot.

## BUFFERS.COM

The programs use the same parameter syntax. A number, or a number preceded by an equals sign (**nn** or **=nn**), specifies the total number of resources needed. A number preceded by a plus sign (**+nn**), adds that many more resources. All the programs add resources; you cannot use them to reduce the number of resources allocated. You run these programs from the DOS prompt.

The DOS **BUFFERS** resource can improve disk I/O response times. Most users have a **BUFFERS** statement in their **CONFIG.SYS** file (e.g., **BUFFERS=30**), but if there is none, DOS allocates a certain number of buffers (usually 15) by default. Each buffer normally uses 528 bytes of memory, so 30 **BUFFERS** take up about 15K of conventional memory.

By adding **BUFFERS**, you can improve some programs' response times, but additional **BUFFERS** also reduce the memory available to programs. Not all programs benefit from additional **BUFFERS**. By having **BUFFERS** in High RAM you can have the benefits of buffering without giving up conventional memory.

To display the number of buffers now allocated:

- Type **BUFFERS** and press **Enter** ↵.

A message will display telling you how many **BUFFERS** are allocated.

To increase the number of **BUFFERS** so that a total of **nn** **BUFFERS** are allocated:

- Type **BUFFERS=nn** and press **Enter** ↵. Substitute the number of **BUFFERS** you want for **nn**.

To load an additional **nn** **BUFFERS** into High RAM:

- Type **LOADHI BUFFERS +nn** and press **Enter** ↵. Substitute the number of additional **BUFFERS** you want for **nn**.

## FILES.COM

DOS's **FILES** (file handles) resource sets the number of files that can be open at any one time and requires about 53 bytes for each file handle. You may have a **FILES=** statement in your **CONFIG.SYS** file specifying a certain number of file handles (DOS 6's default is 8).

To see how many files are currently allocated:

- Type **FILES** and press **Enter** ↵.

To increase the number of **FILES** so that a total of **nn** **FILES** are allocated:

- Type **FILES=nn** and press **Enter** ↵. Substitute the number of **FILES** you want for **nn**.

To load an additional **nn** **FILES** into High RAM:

- Type **LOADHI FILES +nn** and press **Enter** ↵. Substitute the number of additional FILES you want for nn.

DOS uses the FCBS resource to keep track of File Control Blocks. Some older programs use FCBS instead of file handles to manage open files.

In DOS versions 5 and 6, the FCBS statement includes a number that tells DOS to allocate memory for that many FCBS.

In DOS versions 3 and 4, the FCBS statement has two numbers: the first tells DOS to allocate memory for that many FCBS; the second specifies how many FCBS to protect when DOS needs to close an open FCB. Each FCBS resource uses 53 bytes.

To see how many FCBS are now allocated:

- Type **FCBS** and press **Enter** ↵.

To increase the number of FCBS so that a total of n FCBS are allocated with x additional FCBS protected:

- Type **FCBS=n,x** and press **Enter** ↵. Substitute the number of FCBS you want for n and the number of protected FCBS for x. If you are using DOS 5 or 6, omit the second number (x).

To load an additional n FCBS with x protected FCBS into High RAM:

- Type **LOADHI FCBS +n,x** and press **Enter** ↵. Substitute the number of additional FCBS you want for n and the number of protected FCBS for x. If you are using DOS 5 or 6, omit the second number (x).



*FCBS use contiguous memory. When you add more FCBS, a new block of memory for all FCBS must be allocated in High RAM and the memory used for the original FCBS is not recovered.*

DOS uses the LASTDRIVE resource to support both physical and logical disk drives. Logical drives are useful if you use the DOS SUBST program or certain networks. DOS maintains a table of drives and each table entry requires about 100 bytes.

To see how many drives are now allocated:

- Type **LASTDRIV** and press **Enter** ↵.

If you want to allocate additional logical drives for networks or the SUBST command, we suggest changing the LASTDRIVE= statement in CONFIG.SYS to specify only your actual disk volumes (this conserves memory). If your hard disk is partitioned, specify the last partition (e.g., LASTDRIVE=D). Then, load logical drives into High RAM by adding a statement to your AUTOEXEC.BAT file. For example, to add three logical drives, add the following statement to AUTOEXEC.BAT:



### C:\QEMM\LOADHI C:\QEMM\LASTDRIV +3

You can add additional logical drives using a plus sign and a number, as above, or you can set it using a drive letter (e.g., C:\QEMM\LOADHI C:\QEMM\LASTDRIV=G). If you are using a number, you must include the plus sign.



*The DOS drive table uses contiguous memory. When you add more drives, a new block of memory for the entire table must be allocated in High RAM and the conventional memory used for the original drive table (100 bytes per drive) is not recovered.*




## Stealth ROM and Stealth DoubleSpace: Maximizing High RAM

### Enabling Stealth ROM

Stealth ROM is an exclusive QEMM feature that can typically create an additional 48K to 115K of High RAM on almost any PC. Stealth ROM hides your PC's ROMs and makes their memory addresses available for High RAM or expanded memory mapping. Stealth DoubleSpace hides MS-DOS 6's DoubleSpace device driver, freeing up 40K of memory. Depending on your configuration, QEMM's installation or Optimize may have enabled these features on your system. Read this chapter if you want to know how Stealth ROM or Stealth DoubleSpace works, or you want to enable either of these features.

Depending on your configuration and the installation options you chose, Stealth ROM may have been enabled on your system when you installed QEMM. When you run QEMM's Optimize program, Optimize will try to load your TSRs, device drivers and selected parts of DOS into High RAM. If all of them will not fit, Optimize will test your system for compatibility with Stealth ROM and will determine which Stealth ROM method is best for your system.

 *Very few systems are incompatible with Stealth ROM—Optimize will not enable Stealth ROM if it determines that your system is not Stealth-compatible.*

If you have programs (e.g., DESQview, DESQview/X) that can take advantage of additional High RAM, you may want to enable Stealth ROM even if Optimize did not recommend Stealth ROM testing—this will give DESQview and DESQview/X users more memory for each window.

If you do not know if you are using Stealth ROM, you can easily find out.

- **At the DOS prompt, type QEMM and press Enter ↵.**

A report summarizing QEMM's status will display. If you see the line **Stealth Type = M** or **Stealth Type = F**, Stealth ROM is already enabled on your system.

If you are not using Stealth ROM, you can enable it as follows:

- **Be sure you are at the DOS prompt, outside of any multitasking program such as DESQview, DESQview/X, or Microsoft Windows. Type OPTIMIZE /STEALTH and press Enter ↵, then follow the on-screen instructions.**

Optimize will first try the Stealth ROM mapping method (also called ST:M) and, if it is appropriate for your system, Optimize will ask enable that method. If the mapping method is not right for your system, Optimize will try the frame method (also called ST:F).

## Disabling Stealth ROM

## A Technical Description of Stealth ROM

If you are using Stealth ROM and would like to disable it:

- **Edit your CONFIG.SYS file and delete the ST:F or ST:M parameter from the QEMM386.SYS device driver line.**

If you have added any Stealth ROM related parameters to the QEMM386.SYS line, be sure to remove them along with the ST:M or ST:F parameter.

- **Run Optimize (see Chapter 3).**

Stealth ROM creates additional mappable areas at the addresses used by your PC's ROMs. By default, QEMM will turn these areas into High RAM that can be used to load TSRs, device drivers and selected parts of DOS. Stealth ROM monitors the interrupts pointing into those ROMs, and when those interrupts occur, maps the appropriate ROM into the page frame and passes the interrupts to the ROM's location in the page frame. In general, the ROMs targeted are your system (BIOS) ROM, video ROM and disk ROM (certain other ROMs may be "Stealthed" as well).

With the ROMs "out of the way," the amount of usable upper memory is greatly increased. Depending on the location of the ROMs, High RAM regions can become quite large and able to accommodate more or larger device drivers and TSRs.

Because of differences in the way that hardware and software deal with ROMs, Stealth ROM can be implemented using one of two methods:

- **ST:M** (the mapping method) maps system, video, and disk ROMs and any other "Stealthable" ROMs out of the first megabyte of memory. When the system needs the ROM, QEMM maps the appropriate ROM code into the EMS page frame. The ROM code then has a valid real mode address at which it can execute, and it does so normally. When the ROM routine completes, QEMM remaps the ROM out of the first megabyte.
- **ST:F** (the frame method) leaves the system, video, and disk ROMs in place. QEMM places the EMS page frame so that it lies on top of a ROM's address space. When the ROM at the location of the page frame is needed, QEMM saves the current contents of the page frame and restores the ROM to its original location. The ROM code then executes normally. When the ROM routine is finished, QEMM restores the previous contents of the page frame.

The mapping method can typically provide 83K-115K of extra High RAM, and the frame method, which is compatible with more systems, typically provides 48K-64K of extra High RAM.

Stealth ROM is enabled by adding the parameter ST:M or ST:F to the QEMM386.SYS device driver line in CONFIG.SYS. This is normally done

## Stealthing DOS 6's DoubleSpace Driver

## Enabling or Disabling Stealth DoubleSpace

## A Technical Description of Stealth DoubleSpace

for you by the Optimize program. For more information on Stealth ROM, see the STEALTHROM parameter in **Chapter 7**.

MS-DOS 6 includes a program called DoubleSpace that you can use to compress your hard and floppy disks so they will hold more data. DoubleSpace uses a 43K device driver. Without QEMM, DOS 6 will load that driver into conventional or upper memory. If you load the driver into conventional memory, you have 43K less room for running programs; if you load it into upper memory, you have 43K less space for loading TSRs and device drivers high.

QEMM's has a feature called Stealth DoubleSpace that moves the DoubleSpace driver out of conventional or upper memory and maps it into the EMS page frame whenever it is needed. The Stealth DoubleSpace feature saves you 40K of memory.

When you install QEMM, the installation program will detect if you are using DoubleSpace and will offer to enable the Stealth DoubleSpace feature. You may want to enable Stealth DoubleSpace if you used DoubleSpace to compress your hard drive after installing QEMM. To enable or disable Stealth DoubleSpace:

- Start QEMM's Setup program by typing **QSETUP** and pressing **Enter** ↵.
- Press **Enter** ↵ to go to the Setup menu.
- Select **Enable or disable Stealth DoubleSpace**, and when you are asked if you want to enable Stealth DoubleSpace, select **Yes** to enable it, or **No** to disable it..
- Run Optimize (see Chapter 3).



*You do not have to have Stealth ROM itself enabled to use Stealth DoubleSpace.*

DBLSPACE.BIN is the part of DOS that provides access to your compressed drive. If you are using DoubleSpace, DOS loads DBLSPACE.BIN before processing the commands in CONFIG.SYS. When you run DoubleSpace's setup, it adds a device command for DBLSPACE.SYS to your CONFIG.SYS file. DBLSPACE.SYS is actually a device driver that moves DBLSPACE.BIN into conventional memory.

QEMM's Stealth ROM DoubleSpace feature moves the DBLSPACE driver outside of the first megabyte of memory, freeing up the memory DBLSPACE originally occupied. QEMM hooks all DBLSPACE's entry points and maps DBLSPACE into the page frame when any program uses one of the DBLSPACE driver's entry points.

To load Stealth DoubleSpace, the DBLSPACE.SYS line in CONFIG.SYS should be replaced with the following:

**DEVICE=C:\QEMM\ST-DBL.SYS**

Stealth DoubleSpace leaves a 3K stub in conventional memory. When you run Optimize after enabling Stealth DoubleSpace, Optimize will prepend the above line with the necessary LOADHI statement to load the 3K stub high.





## VIDRAM: Extending Conventional Memory

### How VIDRAM Works

QEMM's VIDRAM program can extend conventional memory by as much as 96K for running DOS text-based programs. VIDRAM even extends conventional memory for DOS programs running in Microsoft Windows. You may want to read this chapter if you routinely run large DOS-text based programs (e.g., dBASE IV) that could benefit from the additional memory.

To use VIDRAM, your system must have an EGA or VGA video adapter or an adapter with EGA or VGA capability (this includes VGA-compatible 8514A video adapters). Your PC must have 640K of conventional memory and the programs you run using this additional memory must not use EGA or VGA graphics.

VIDRAM has one other requirement: For VIDRAM to work, no hardware device (e.g., a hard disk controller) can utilize the area just below 640K. QEMM will automatically move XBDAs (Extended BIOS Data Areas) out of this area.

VIDRAM is a standalone TSR. It does not require expanded memory, extended memory or a memory manager. If your system has the necessary video RAM, VIDRAM will work on 8088, 8086, 80286 and 80386, i486 and Pentium PCs.

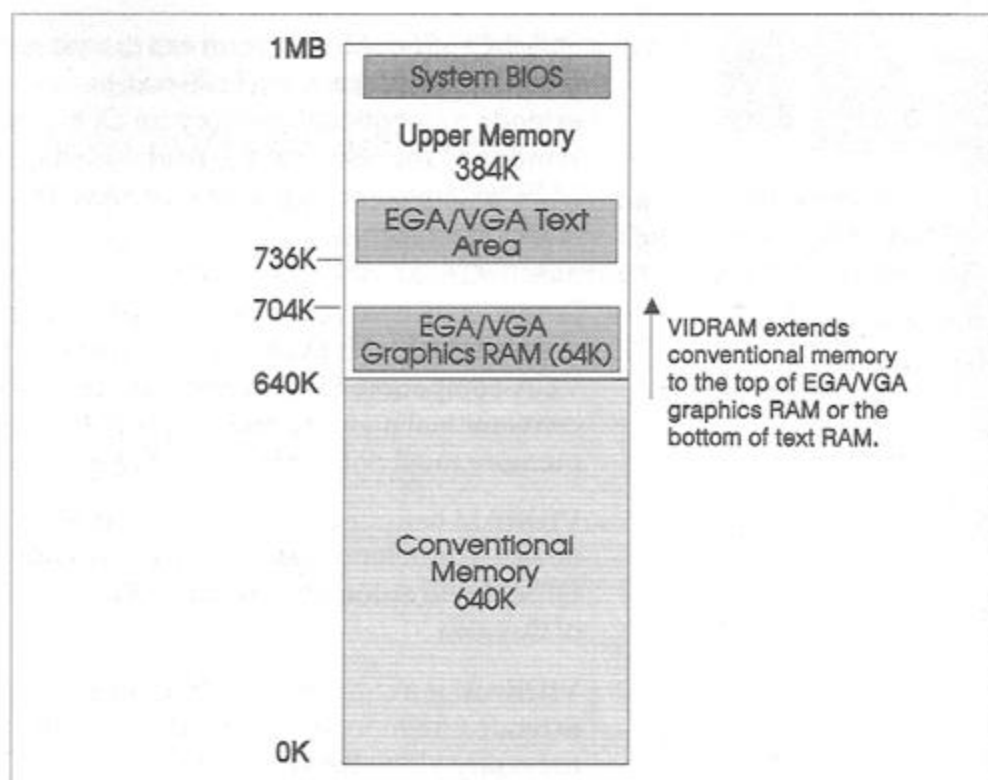


*If your system has a Hercules, monochrome or CGA video adapter (and it does not have an additional EGA or VGA adapter), you will not need VIDRAM. QEMM automatically extends conventional memory to 704K on Hercules and monochrome systems and to 736K on CGA systems.*

If your PC has an EGA or VGA video adapter, the 64K memory area just above conventional memory (640K-704K or A000-AFFF hex) is reserved for use by graphics modes. When you run text-based programs, that area is unused, so VIDRAM can appropriate it to extend the contiguous conventional memory for running programs (see the illustration on the following page).

There may be an additional 32K of unused address space directly above the video graphics area that VIDRAM can also appropriate. As long as the memory addresses are contiguous with the end of conventional memory and they are in the first megabyte, DOS can use them to run real mode programs.

**If your PC has a VGA-compatible 8514A adapter:** You can use VIDRAM to extend conventional memory for 8514A graphics programs as well as DOS text-based programs. Graphics programs configured specifically for the 8514A card do not use the EGA/VGA graphics area, so you will still be able to use these programs when VIDRAM is on.



Memory areas used by VIDRAM

VIDRAM actually does two things:

- ❑ It intercepts video requests and refuses all requests that would make use of the EGA/VGA graphics area (i.e., graphics operations).
- ❑ It maps EMS memory (obtained via a private interface to QEMM) to the EGA/VGA area, unless you specify otherwise. If you do not want to use EMS memory, you can tell VIDRAM to use your video card's memory, which is generally slower than EMS memory.

The amount of additional DOS memory VIDRAM will make available depends on several factors. The maximum is 96K with a color monitor and an EGA or VGA adapter. If you have a monochrome monitor or are using dual monitors you can still gain 64K memory for your programs.


## Using VIDRAM

It is important to understand that you cannot run EGA or VGA graphics operations while VIDRAM is in use. If you routinely use both large text-based programs and graphics programs, you can turn the VIDRAM feature on when you need it for a text program and off before you run a graphics program. If you are using an 8514A adapter, you can still use 8514 graphics programs while VIDRAM is enabled.

To turn VIDRAM on:

- Type **VIDRAM ON** and press **Enter** .J.

This command will extend conventional memory into the EGA/VGA graphics area for a total of 704K conventional memory.

 **WARNING:** *If VIDRAM is on and you start an EGA or VGA graphics program, you will see a VIDRAM warning, telling you that graphics video modes are disabled. At that point, you should press any key other than Esc to abort the graphics program and return to the DOS prompt. Then, turn VIDRAM off and you can start your graphics program again.*

To turn VIDRAM off:

- Type **VIDRAM OFF** and press **Enter** ↵.

**DESQview users:** If you want to extend conventional memory for DESQview DOS windows, you must do two things: First add the VIDRAMEMS parameter to the QEMM386.SYS device driver line in your CONFIG.SYS file, then run Optimize (see **Chapter 7** for information on the VIDRAMEMS parameter and **Chapter 3** for information on Optimize). Then use **VIDRAM ON EMS** instead of **VIDRAM ON**. This will ensure that the additional memory is available to all DOS windows in DESQview. You cannot turn VIDRAM on and off inside DESQview.

If you have a color system and you want to extend conventional memory an additional 32K for a total of 736K, you will have to prevent QEMM from using the 32K directly above the EGA/VGA graphics area as High RAM. Keep in mind that there will be 32K less space to load TSRs or device drivers high, so you will need to decide whether having the extra 32K for DOS text programs outweighs the loss of High RAM. If you decide to extend conventional memory by 32K, add the VIDRAMEGA parameter to the QEMM386.SYS device driver line in your CONFIG.SYS file, reboot your PC, then type **VIDRAM ON** (or **VIDRAM ON EMS** if you are using DESQview). See **Chapter 7** for information on the VIDRAMEGA parameter.

If you seldom use graphics programs, you may want to put VIDRAM in your AUTOEXEC.BAT file so it will load whenever you boot your PC. To do this, add the line **VIDRAM ON** to your AUTOEXEC.BAT file.

If you ever want to know whether VIDRAM is on or off:

- Type **VIDRAM** and press **Enter** ↵.

You will see a message telling you whether VIDRAM is resident and enabled. If it is enabled, it will report the type of memory in use and the new ending address of conventional memory.

## VIDRAM Parameters

VIDRAM has several parameters. These parameters will be useful if:

- ❑ You need to run graphics programs occasionally.
- ❑ You have a second video adapter and monitor.
- ❑ If you are using DESQview.

VIDRAM's optional parameters are listed below. Some parameters have an abbreviation you can use instead of the full name; abbreviations are listed in parentheses.

### RESIDENT (RES)

loads the resident portion of VIDRAM into memory, but does not enable any VIDRAM functions. This parameter is provided for those who need to load VIDRAM high. If you want to use this parameter, see the next section, *Using VIDRAM with LOADHI*.

### OFF (OF)

turns VIDRAM off, restoring video memory to the reserved video area and enabling graphics capability.

### ON

turns VIDRAM on to extend DOS memory and inhibit graphics operations. VIDRAM will remain on until you turn it off. You can modify the ON parameter with three additional options: OVERRIDE, EGA and EMS.

- ❑ **VIDRAM ON OVERRIDE (OV)** forces VIDRAM on in situations where it would normally refuse to appropriate video memory. These situations occur when you have two monitors and two video adapters in your system, or when all or part of the video RAM is already being managed by a resident memory manager. If either of these situations applies to you, we suggest you try VIDRAM ON OVERRIDE to see if it helps. OVERRIDE's utility in these situations depends on what adapters are involved or what memory management features you are using.
- ❑ **VIDRAM ON EGA** tells VIDRAM to use the memory supplied by the video adapter instead of EMS memory. You can use this option when you want to preserve every bit of EMS memory for other uses. Video adapter memory is generally slower than EMS memory. This is the default behavior when there is not enough EMS memory available or QEMM is not present. If you use VIDRAM ON EGA and you want to extend conventional memory an extra 32K, you can add the VIDRAMEGA parameter on the QEMM386.SYS device driver line of your CONFIG.SYS file (see Chapter 7 for information). If you add that parameter, you will forfeit 32K of High RAM. If you add the parameter, be sure to run Optimize (see Chapter 3).

- ❑ **VIDRAM ON EMS** tells VIDRAM to use EMS memory (provided by the standard EMS interface) to extend DOS. This option is provided for DESQview users. It ensures that the extra memory will be available to all windows in DESQview. If you use VIDRAM ON EMS, you must have the VIDRAMEMS parameter on the QEMM386.SYS device driver line of your CONFIG.SYS file (see **Chapter 7** for information). If you add that parameter, be sure to run Optimize (see **Chapter 3**).

### NOCGA

This parameter is rarely used. It prevents all graphics functions (CGA through VGA) from using the video graphics areas. This option makes VIDRAM resident but does not extend DOS memory. You might use this parameter when some of your video RAM area has been mapped with EMS memory by something other than VIDRAM (e.g., QEMM386.SYS's INCLUDE=A000-AFFF parameter) and you want to protect it from being accessed by graphics programs.

### NOEGA

This parameter is rarely used. It prevents EGA and VGA graphics requests from being honored. This option makes VIDRAM resident but does not extend DOS memory. You might use this parameter when some of your video RAM area has been mapped with EMS memory by something other than VIDRAM (e.g., QEMM386.SYS's INCLUDE=A000-AFFF parameter) and you want to protect it from being accessed by graphics programs.

If you are using VIDRAM, you can save a small amount of memory by loading the resident portion of VIDRAM high. Once you load VIDRAM high, you will need to type a second VIDRAM command to extend conventional memory. To load VIDRAM's resident portion high and extend conventional memory into the VGA graphics area:

- Type **LOADHI VIDRAM RESIDENT** and press **Enter** ↵
- Type **VIDRAM ON** and press **Enter** ↵.

This allows the code that intercepts graphics requests to be in High RAM, while the memory management portion is in memory only briefly to extend or return memory. The resident portion of VIDRAM is quite small.

You can use VIDRAM with Microsoft Windows 3.0 and 3.1 386 enhanced mode to extend conventional memory for running DOS text-based applications under Windows.



**IMPORTANT:** If VIDRAM is on before you start Windows, turn it off by typing VIDRAM OFF.



To use VIDRAM to extend conventional memory for a DOS text-based program in Windows:

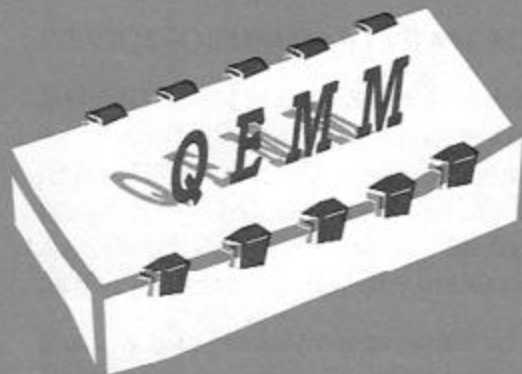
- **Start Windows.** Be sure you are running in 386 enhanced mode. You can use `WIN /3` to ensure that you start in that mode.
- **Double-click on the MS-DOS Prompt icon** (usually in the Main group).
- **When you see the DOS prompt, type `VIDRAM ON` and press Enter ↵.**
- **If you want to run the DOS application in a window, press `Alt+Enter` ↵.**
- **Type the command to start the program.**
- **When you close the window, the extra memory that VIDRAM provides will be automatically relinquished.**

If you want to use VIDRAM every time you run a certain DOS text program under Windows, you could create a batch file containing two lines: `VIDRAM ON` and the command to start the program. Then you could use Windows PIF editor to put the batch file's name in the PIF's Program Filename field (see your Windows manual for information).



*If you want to extend conventional memory an additional 32K (for a total of 736K) for DOS text programs running in Windows, add the parameter `VIDRAMEGA` to the `QEMM386.SYS` device driver line in your `CONFIG.SYS` file (see **Chapter 7**). Keep in mind that there will be 32K less space to load TSRs or device drivers high, so you will need to decide whether having the extra 32K for DOS text programs outweighs the loss of High RAM.*

# Technical Reference






## QEMM386.SYS Parameters


### Parameters and Memory Addresses

This chapter is the technical reference guide for the parameters that can be used with the QEMM386.SYS device driver in your CONFIG.SYS file. Since QEMM automatically configures itself for your system, you should only need to use the parameters in this chapter if you want to fine-tune your memory configuration, or you are experiencing problems.

To use a QEMM command line parameter, place the parameter name on the same line as **DEVICE=QEMM386.SYS** in your CONFIG.SYS file. For example, the QEMM command line with the RAM, ROM=C000, and STEALTHROM:M parameters set looks like:

**DEVICE=QEMM386.SYS RAM ROM=C000 STEALTHROM:M**

 **IMPORTANT:** Do not put spaces within a parameter. In the example above, make sure that you do not have a space before or after the = sign in ROM=C000. Also, any parameters you use must appear on the same line as **DEVICE=QEMM386.SYS**. Optionally, you can put the parameters you want to use in a file and reference the file on the QEMM386.SYS line (see the section *Parameter Files*, later in this chapter).

 If you have upgraded from an earlier version of QEMM, be aware that many of the parameter names have changed in QEMM version 7. You can still use the older parameter names if you like. At the end of this chapter is a list of the parameter names that have changed, cross referenced with their former names.

All memory addresses in this chapter that are used as arguments to QEMM parameters are specified as hexadecimal memory addresses. Even if you do not understand hexadecimal addresses, you should be able to use these parameters; you can generally get the appropriate values from either the documentation for your hardware or software, or from the QEMM program (e.g., Manifest, Optimize or QEMM.COM) that suggests the parameter.

QEMM's parameters also accept decimal numbers followed by the letter K, to denote a number of kilobytes, or the number M, to denote a number of megabytes. So the parameter **EXCLUDE=B000-B7FF**, where B000 and B7FF are hexadecimal segment addresses, can also be specified as **EXCLUDE=704K-736K**. You can mix different kinds of numbers in the same parameter (e.g., **ROM=896K-1M**).

Furthermore, you can specify an address range either with a hyphen (-), which indicates that the number following the range's starting point is its end point, or with a colon (:), which indicates that the number following the range's starting point is its length. For example, the parameter **RAM=C000-C0FF** can also be specified as **RAM=C000:4K**, or as **RAM=768K:4K**.

## Parameter Files

If you do not want to place some or all of the parameters you desire on the QEMM386.SYS line in your CONFIG.SYS file, you can tell QEMM to read parameters from a text file. You can specify a full pathname to the text file, or you can specify only its name, and place it in the directory that is current when QEMM loads (usually the root directory of your boot drive). To use a parameter file, place a @ character on the QEMM line, immediately followed by the filename. For example, the following line tells QEMM to read parameters from a file called SWITCHES.CMD in the root directory:

```
DEVICE=C:\QEMM\QEMM386.SYS RAM @SWITCHES.CMD
```

Note that some parameters can be specified directly on the QEMM line and others in the SWITCHES.CMD file. The parameter file can contain a separate parameter on each line, or you can place all the parameters on the same line with a space between each parameter. A semicolon in a parameter file means that everything to the right of the semicolon on that line is a comment and will not be processed by QEMM. So SWITCHES.CMD might look something like this:

```
ROM ;Run ROMs fast
DB=10 ;Remove when VDS driver arrives
ST:M ;Stealth
```

You can specify parameter files from within parameter files, down to five levels. A parameter file could look like this:

```
RAM
@EXCLUDES.CMD
ST:M
ROM
```

In this case, a separate file in the root directory called EXCLUDES.CMD would contain additional parameters for QEMM to process.

## Parameter Descriptions

Many of the parameters have an abbreviation you can use instead of the full name; abbreviations are listed in parentheses below.

Several parameters take a Y or N argument (for "yes" or "no"). Even though these parameters can take either Y or N as an argument, we often list only one of these arguments (e.g., SORT:Y or USEXMS:N) because the other is QEMM's default condition at all times. If the parameter's default varies depending on other conditions (e.g., whether or not Stealth ROM is enabled), then we describe all the parameter's arguments. If you specify neither a Y nor an N to such a parameter, QEMM will act as if you specified a Y.

Often a parameter has a certain effect when used along with another parameter, so we have added comments to describe the interactions and relations between parameters. QEMM processes its parameters

## Commonly Used Parameters

sequentially—if two mutually exclusive parameters are placed on the QEMM386.SYS line in the CONFIG.SYS file, the one that comes later overrides the earlier one.

The most commonly used parameters are INCLUDE, RAM, ROM, STEALTHROM and EXCLUDE. The first four of these parameters activate QEMM features, and the last one is the most useful troubleshooting parameter. These five parameters are described below in alphabetical order.

### **EXCLUDE=xxxx-yyyy (X)**

tells QEMM to make the region **xxxx-yyyy** unavailable for High RAM, expanded memory mapping or ROM mapping. EXCLUDE is the most important troubleshooting parameter, and is usually used to stop QEMM from placing High RAM in a section of upper memory that a program or a piece of hardware needs to access. For example, if you have a network adapter that puts 16K of RAM at address CC00, and QEMM is not detecting the network adapter, then you should use the parameter EXCLUDE=CC00-CFFF. You can use the Analysis procedure (see *page 97*) to find regions that should be excluded.

You may specify multiple ranges by using EXCLUDE several times. If you exclude any part of a 4K region aligned on a 4K boundary, QEMM will exclude that entire 4K region.

By default, QEMM will try to reserve 64K of contiguous upper memory addresses that are not used by ROMs, video RAM or adapter RAM, and designate this area as the EMS page frame. When you use the EXCLUDE parameter, you reduce the number of possible locations for the page frame. If QEMM cannot find 64K of free contiguous addresses in upper memory, it places the page frame in conventional memory, diminishing by 64K the amount of memory available to run programs.

### **INCLUDE=xxxx-yyyy (I=xxxx-yyyy)**

tells QEMM to use the address range **xxxx-yyyy** for High RAM or for expanded memory mapping. If you include an address range, QEMM will use it regardless of whether it is used by ROMs, video RAM, or adapter RAM. We advise you to use the INCLUDE parameter only when recommended by the Analysis procedure (see *page 97*).

Used properly, the INCLUDE parameter may increase the amount of available High RAM, particularly on systems that are not using the Stealth ROM feature. You can specify multiple regions by using INCLUDE several times. INCLUDE operates only on 4K regions aligned on 4K boundaries; QEMM will not include any part of a 4K region aligned on a 4K boundary unless the entire region is included.



### **RAM or RAM=xxxx-yyy**

specifies that QEMM should create High RAM—Quarterdeck's term for RAM that QEMM makes appear in upper memory (the area between 640K and 1024K). You need High RAM if you want to load TSRs, device drivers or portions of DOS high, to free up more conventional memory for running DOS programs. By default, the QEMM installation adds the RAM parameter to the QEMM386.SYS line.

If you specify the RAM parameter without an address range, QEMM sets aside room for an EMS page frame (unless you are using the or FRAME=NONE parameters) and places High RAM at all other upper memory addresses at which QEMM has not detected ROM, video RAM and adapter RAM. You should not specify an address range unless you know exactly what regions of upper memory are safe to use.

High RAM can be used by QEMM's LOADHI programs, by DESQview and DESQview/X, and by programs that allocate UMBs (Upper Memory Blocks) through the XMS specification. When the RAM parameter is specified, you cannot turn QEMM off.

### **ROM, ROM=xxxx or ROM=xxxx-yyy**

tells QEMM to speed up the operation of ROMs by copying them into faster RAM and making the RAM appear in upper memory in place of the ROMs. This process is called *ROM shadowing*. If your system does not shadow ROMs in hardware, the ROM parameter may speed up some system operations. It particularly affects programs that write to the screen using BIOS or DOS video calls (like DOS's COMMAND.COM). If your system shadows ROMs in hardware and you have this feature enabled in your system setup, you normally would not need the ROM parameter.

If you specify ROM without an address, QEMM copies all ROMs into RAM. If you specify only the starting address of a ROM, QEMM will determine the ROM's size, and shadow that entire ROM and no other. For example, ROM=C000 will make QEMM shadow a video ROM that begins at C000 and will leave all other ROMs unaffected. You can specify both the beginning and end of an address range if you want QEMM to shadow only part of a ROM, but it is usually easier to use the EXCLUDE parameter to tell QEMM what areas should not be affected by the ROM parameter. For instance, if you want to shadow all ROMs, but you know that shadowing the F400-F4FF region causes your floppy drives to malfunction, you can use the parameter ROM EXCLUDE=F400-F4FF.

When the ROM parameter is in use, you cannot turn QEMM off. The smallest region that the ROM parameter can affect is a 4K region aligned on a 4K boundary, and specifying any part of such a region causes the whole 4K region to be shadowed. However, when a ROM has been

relocated with the STEALTHROM parameter, the smallest region that the ROM parameter can affect is a 16K region aligned on a 16K boundary. In this case, the 16K region will not be shadowed unless a portion of each of its 4K regions has been specified. For example, if the C000 ROM has been relocated by STEALTHROM, ROM=C000-C2FF will have no effect, but ROM=C000-C37F will cause the entire C000-C3FF region to be shadowed.

If you use the EXCLUDE parameter to prevent a ROM region from being shadowed by the ROM parameter, then any EXCLUDE will normally prevent the entire 4K ROM region (aligned on a 4K boundary) in which it falls from being shadowed. However, if that ROM is being relocated with the STEALTHROM parameter, then any EXCLUDE, however small, will prevent the entire 16K ROM region (aligned on a 16K boundary) in which it falls from being shadowed. For more information on using Stealth ROM with the ROM parameter, see *Using the ROM and STEALTHROM Parameters Together* on page 79.

### **STEALTHROM:M or F (ST:M or F)**

enables the Stealth ROM feature—the QEMM technology that relocates ROMs from upper memory so that their addresses can be used for High RAM or for expanded memory mapping. When a ROM service is requested, QEMM will temporarily place the ROM in the EMS page frame to handle the request. Stealth ROM is an important feature—with it, QEMM can typically create an additional 48K-115K of High RAM for loading TSRs, device drivers, and portions of DOS.

When you specify the STEALTHROM:x parameter, replace the letter x with the letter that designates the Stealth ROM method that you want to use. **M** specifies the **mapping method**, which relocates as many ROMs as possible out of upper memory. **F** specifies the **frame method**, which relocates only the ROMs in the area where the EMS page frame lies. The mapping method usually makes more upper memory available, but the frame method may work in some cases when the mapping method does not. On a typical system, the mapping method frees an additional 83K-115K, and the frame method frees 48K-64K.

Depending on how your system is configured, QEMM's Optimize process may test your system for Stealth ROM compatibility and specify a Stealth ROM parameter automatically. If you want to test for Stealth ROM compatibility, run Optimize with the /STEALTH parameter (see Chapter 3).

Using the EXCLUDE parameter in a ROM region does not prevent QEMM from relocating the ROM when the STEALTHROM parameter is in use. However, it does prevent QEMM from placing High RAM or allowing memory to be mapped in that region. When the EXCLUDE parameter is used on an area from which ROMs have been moved, programs are able to access the ROM at its original location, even though

## Less Frequently Used Parameters

all ROM BIOS calls are still redirected away from this region by the Stealth ROM process. This is sometimes the most efficient way to preserve the benefits of Stealth ROM when you are running a program that insists on accessing a ROM at a fixed address. For instance, the parameters `STEALTHROM:M EXCLUDE=FC00-FCFF` allow you to use the address space of all Stealthed ROMs with the exception of the 4K region at FC00-FCFF, in the event that you have a program that jumps directly to that location to access a ROM routine.

For information on using Stealth ROM with the ROM parameter, see *Using the ROM and STEALTHROM Parameters Together* on page 79. See also the `EXCLUDESTEALTH` and `EXCLUDESTEALTHINT` parameters for discussions of how to modify the Stealth ROM feature. You cannot use Stealth ROM if you have specified the `EMS:N` or the `FRAME=NONE` parameter.

The previous section of this chapter described QEMM's most commonly used parameters. This section describes the rest of QEMM's parameters. QEMM parameters that you may use on occasion are:

<code>COMPAQFEATURES:N</code>	<code>DISKBUF</code>
<code>DISKBUFFRAME</code>	<code>DMA</code>
<code>EMBMEM</code>	<code>EMS:N</code>
<code>EXCLUDESTEALTH</code>	<code>EXCLUDESTEALTHINT</code>
<code>FILL:N</code>	<code>FORCEEMS:Y</code>
<code>FORCESTEALTHCOPY</code>	<code>FRAME</code>
<code>LOCKDMA</code>	<code>MAPREBOOT:N</code>
<code>ROMHOLES:N</code>	<code>SHADOWRAM:NONE</code>
<code>SORT</code>	<code>SUSPENDRESUME</code>
<code>TOKENRING:N</code>	<code>TOPMEMORY:N</code>
<code>UNMAPFREEPAGES:N</code>	<code>VCPISHARE</code>
<code>VDS:N</code>	<code>VIDEORAM:N</code>
<code>VIDRAMEGA</code>	<code>VXDDIRT</code>
<code>XBDA:N</code>	<code>TXMS:N</code>

All other parameters listed in this section are infrequently used.

The QEMM386.SYS command line parameters are described below in alphabetical order.

**?**

displays a list of the QEMM386.SYS command line parameters and their abbreviations. QEMM will not load if you specify this parameter.

### **ADAPTERRAM=xxxx-yyyy (ARAM)**

tells QEMM to make the region xxxx-yyyy unavailable for High RAM, expanded memory mapping or ROM mapping. ADAPTERRAM is

exactly like the EXCLUDE parameter, except that the QEMM.COM Type display and the Manifest QEMM Type display will identify this region as "Adapter RAM" instead of "Excluded." Like the EXCLUDE parameter, it is needed only if QEMM does not identify an adapter's RAM.

#### **ADAPTERROM=xxxx-yyyy (AROM)**

tells QEMM to make the region xxxx-yyyy unavailable for High RAM, expanded memory mapping or ROM mapping. ADAPTERROM is exactly like the EXCLUDE parameter, except that the QEMM.COM Type display and the Manifest QEMM Type display will identify this region as "Adapter ROM" instead of "Excluded." Like the EXCLUDE parameter, it is needed only if QEMM does not identify an adapter's ROM.

#### **AUTO (AU) or ON or OFF (OF)**

determines when QEMM puts the processor into virtual-8086 mode, which is required for most of QEMM's functions. These parameters are generally used only for troubleshooting purposes.

AUTO means that QEMM will put the system into virtual-8086 mode only when necessary to provide a feature, like expanded memory. ON means that QEMM will always run real-mode programs in virtual-8086 mode. OFF means that QEMM will never use virtual-8086 mode; QEMM will then provide almost no features, though it will still take extended memory and make it unavailable to other programs. (You can use the EXTMEM or MEMORY parameters to limit QEMM's memory allocation even when QEMM is off.)

Various other parameters (e.g., RAM, ROM, STEALTHROM) force QEMM on and cause AUTO and OFF to be ignored. If no other parameter forces QEMM on, QEMM's state defaults to AUTO, but can be changed from the DOS prompt with the QEMM.COM program (see Chapter 9).

#### **COMPAQ386S:Y (C386S:Y)**

tells QEMM to make the slight adjustments necessary to implement its Compaq features on the Compaq 386s. This parameter may help minimize "101 ROM error" messages that sometimes prevent warm boots on Compaq 386s systems. If you have run version 6.02 or later of Compaq's Setup program on your 386s system, you do not need the COMPAQ386S:Y parameter. You should use this parameter only with Compaq 386 systems.

#### **COMPAQEGAROM:Y or N (CER:Y or N)**

tells QEMM whether to implement the Compaq EGA ROM feature, which relocates Compaq's video ROM to increase the amount of available upper memory. Many Compaq systems keep two copies of video ROM code in upper memory: a slow ROM at C000 and a faster



RAM copy of the ROM at E000. If QEMM detects this configuration on a Compaq, it makes the system use the C000 ROM and makes the E000 area available for High RAM or expanded memory mapping. In compensation, QEMM also copies the C000 ROM into faster RAM and places that RAM at C000 to avoid slowing the system.

COMPAQEGAROM:N stops QEMM from implementing the Compaq EGA ROM feature, at a cost of as much as 32K of upper memory addresses on Compaq systems. The COMPAQFEATURES:N parameter disables this feature and two other Compaq features, and is a useful troubleshooting parameter, especially on Compaq systems. QEMM cannot be turned off if the Compaq EGA ROM feature is implemented.

#### **COMPAQFEATURES:Y or N (CF:Y or N)**

enables or disables all three of QEMM's Compaq features: Compaq EGA ROM, Compaq Half ROM, and Compaq ROM Memory. By default, QEMM enables all appropriate Compaq features on Compaq systems. You can enable/disable the three features separately with the COMPAQEGAROM, COMPAQHALFROM, and COMPAQROMMEMORY parameters. You can use COMPAQFEATURES:N for convenience, when troubleshooting.

#### **COMPAQHALFROM:Y or N (CHR:Y or N)**

tells QEMM whether to implement the Compaq Half ROM feature, which reclaims half of the Compaq system ROM's address space. Some Compaq system ROMs consist of a single 32K code area at address F000 that is duplicated at address F800. If QEMM detects this configuration on a Compaq, it makes the system use the F800-FFFF half of the System ROM, and makes F000-F7FF available for High RAM or expanded memory mapping.

COMPAQHALFROM:N stops QEMM from implementing the Compaq Half ROM feature, at a cost of 32K of upper memory addresses on some Compaq systems. The COMPAQFEATURES:N parameter disables this feature and two other Compaq features, and is a useful troubleshooting parameter, especially on Compaq systems.

#### **COMPAQROMMEMORY:Y or N (CRM:Y or N)**

tells QEMM whether to implement the Compaq ROM Memory feature, which appropriates 128K of reserved memory that some Compaq systems use for speeding up ROMs. If QEMM detects this memory, it places it into QEMM's memory pool, thus making it available through any interface that QEMM supports (e.g., EMS, XMS, VCPI, or DPMI). In compensation, QEMM also copies the Compaq ROMs into faster RAM and places that RAM at the ROMs' addresses to avoid slowing the system.



COMPAQROMMEMORY:N stops QEMM from implementing the Compaq ROM Memory feature. The COMPAQFEATURES:N parameter disables this feature and two other Compaq features, and is a useful troubleshooting parameter, especially on Compaq systems. QEMM cannot be turned off if the Compaq ROM Memory feature is implemented.

#### **DISKBUF=nn (DB=nn)**

tells QEMM to allocate an additional nnK of conventional memory as a disk buffer, and to send through this buffer all BIOS disk reads and writes into and out of areas where QEMM has relocated memory. The purpose of this buffer is to prevent memory addressing errors on systems that have a bus-mastering hard disk controller. The most common symptom of a conflict between QEMM and a bus-mastering hard disk controller is a system failure during the final phase of the Optimize process, or when loading any program into High RAM.

QEMM automatically creates a 2K disk buffer whenever it detects an addressing problem with a bus-mastering hard disk controller. A larger disk buffer uses more conventional memory but may improve disk performance; you can use the DISKBUF parameter to change the size of the disk buffer that QEMM creates by default (2K and 10K are commonly used sizes), or to force QEMM to create a disk buffer.

You do not need the DISKBUF parameter if your bus-mastering hard disk uses the Virtual DMA Services (VDS) specification to resolve conflicts with 386 memory managers. DISKBUF will not resolve conflicts between QEMM and bus-mastering devices that are not hard disks (e.g., network adapters); such devices must support the VDS specification to work properly with 386 memory managers.

#### **DISKBUFFRAME=nn (DBF=nn)**

tells QEMM to allocate an additional nnK of conventional memory as a disk buffer, and to send through this buffer all BIOS disk accesses that transfer information into or out of the EMS page frame. QEMM should prompt you with an error message if you are using a program that requires DISKBUFFRAME. This parameter is usually needed if Stealth ROM is enabled and an EMS-using program accesses the page frame during BIOS-level disk interrupt calls. Disk caches that use EMS (e.g., PC-CACHE, Norton 5.0 N-CACHE, and the Compaq cache) and disk compression utilities (e.g., Stacker) are the programs most likely to need the DISKBUFFRAME parameter.

For performance reasons, you may prefer to reconfigure these programs to use XMS memory instead of EMS memory if possible, thereby avoiding the need for the DISKBUFFRAME parameter. A larger disk buffer uses more conventional memory but may improve disk

performance; 2K and 10K are commonly used values. DISKBUFFRAME is a weaker version of DISKBUF and is not needed when you are using DISKBUF.

#### **DMA=nnn (DM=nnn)**

specifies the size in K of QEMM's DMA (Direct Memory Access) buffer. The DMA buffer should be large enough to contain the largest DMA transfer that any hardware device in the system will make into or out of memory.

The default values of **nnn** are 64 on PS/2 and XT-based systems, and 12 on other systems; the maximum useful value is 128. If you need a larger DMA buffer, QEMM should prompt you with an error message that tells you what size buffer to specify with the DMA parameter. The DMA buffer comes out of QEMM's memory pool and does not affect the size of conventional memory.

#### **DOS4:Y (D4:Y)**

alters the way that QEMM assigns EMS physical page numbers to physical page addresses. The **DOS4:Y** parameter attempts to minimize the ill effects of DOS 4's nonstandard use of expanded memory, but it does not make DOS 4's methods safe. If you are using PC- or MS-DOS 4.00 or PC-DOS 4.01, we recommend that you not use the **/X** switch to **BUFFERS**, **VDISK**, or **FASTOPEN**.

#### **EMBMEM=nnnnn (EMB=nnnnn)**

sets the maximum amount of extended memory that programs can allocate as EMBs (Expanded Memory Blocks) through the Extended Memory Specification (XMS). **nnnnn** is the number of K to be available as EMBs.

By default, QEMM does not limit the amount of XMS memory that can be allocated. (The one exception is that QEMM detects when Microsoft Windows 3.0 starts up in standard mode and limits available XMS memory to 12 megabytes while it is running.) The **EMBMEM** parameter can be useful when any XMS-using program threatens to allocate so much memory that other programs will be unable to get memory through any specification (EMS, XMS, VCPI, or DPMI). In particular, **EMBMEM** is helpful when trying to run DOS programs that use EMS, VCPI, or DPMI memory inside of Microsoft Windows 3.0 standard mode.

#### **EMS:N**

tells QEMM not to provide expanded memory services, even for multitasking environments. **EMS:N** is sometimes useful on XT machines that have been converted to 80386 systems, where you may want to load a different expanded memory manager to manage an EMS board and still

use QEMM's other features. Otherwise, you should only use the EMS:N parameter for troubleshooting. If you want to sacrifice QEMM's expanded memory services to gain more High RAM, use the FRAME=NONE parameter instead of EMS:N.

#### **EXCLUDE=xxxx-yyy (X=xxxx-yyy)**

tells QEMM to make the region xxxx-yyy unavailable for High RAM, expanded memory mapping or ROM mapping. For a full description, see *page 55*.

#### **EXCLUDESTEALTH=xxxx (XST=xxxx)**

tells QEMM not to use the Stealth ROM feature to relocate the ROM that begins at address xxxx. You should only provide the starting address of the ROM, not the entire range.

This parameter and the EXCLUDE=xxxx-yyy parameter are frequently used to troubleshoot problems with Stealth ROM. The EXCLUDESTEALTH parameter prevents an entire ROM from being relocated, whereas the EXCLUDE parameter can be narrowed to cover only a part of a ROM. If you find that you can use either parameter to solve your problem, you will probably be able to get more High RAM by using the EXCLUDE parameter and following the Analysis procedure (see *page 97*) to narrow the exclude as much as possible.

If you want to use the EXCLUDESTEALTH parameter on more than one ROM region, you must use a separate EXCLUDESTEALTH parameter for each ROM. On IBM PS/2 systems, the video ROM at E000 and the system ROM at F000 are considered separate ROMs, and should be specified separately with the EXCLUDESTEALTH parameter, even though the two regions are contiguous. This parameter, can in some cases, cause a previously stable system to malfunction, especially with the Stealth frame method.

#### **EXCLUDESTEALTHINT=xx (XSTI=xx)**

tells QEMM not to intercept interrupt xx (where xx is a hexadecimal number between 0 and 7F) as part of the Stealth ROM process. Normally QEMM must intercept all interrupts that are handled by ROMs that QEMM has relocated with the Stealth ROM feature.

The EXCLUDESTEALTHINT parameter is occasionally useful when troubleshooting problems with Stealth ROM, but it is sometimes difficult to determine which interrupt to choose without technical insight. Furthermore, you must usually use the EXCLUDE=xxxx-yyy parameter along with the EXCLUDESTEALTHINT=xx parameter, where xxxx-yyy is the range of ROM addresses that contain the interrupt handler for interrupt xx. See the file STEALTH.TEC in the \QEMM\TECHNOTE directory for information on how and when to use this parameter.

**EXTMEM=nnnnn (EXT=nnnnn)**

specifies the amount of extended memory (in K) that should not be under QEMM's control. This parameter should only be used to provide memory for old utilities (for example, VDISK) that allocate extended memory without using the XMS, VCPI, or DPBI specifications. We recommend that, if possible, you get a new XMS-using version of your utility instead of using the EXTMEM parameter. Memory reserved by the EXTMEM parameter will be unavailable to programs that use EMS, XMS, VCPI, or DPBI.

QEMM will not load if the EXTMEM parameter takes away so much memory from QEMM that it cannot find enough memory for its own overhead. QEMM will also refuse to load if you have a system with more than 16 megabytes of memory and EXTMEM takes away so much of the first 16 megabytes of memory that QEMM cannot use any of it. (Unmanaged memory is always left behind at the bottom of the expanded memory region.) The highest number you can safely use depends on your configuration and the amount of memory on your system, but numbers over 15000 are likely to cause a failure no matter how much memory you have.

You can use either the EXTMEM or MEMORY parameter to limit the memory under QEMM's control. EXTMEM=nnnnn specifies that nnnnnK should not be controlled by QEMM. MEMORY=nnnnn specifies that QEMM should control no more than nnnnnK, which much be used for its own overhead as well as its memory pool. Too large an EXTMEM number or too small a MEMORY number will cause QEMM not to load because of memory shortage.

**FASTINT10:N (F10:N)**

tells QEMM not to replace certain BIOS video functions with its own routines which speed up video access. The purpose of FASTINT10:N is to eliminate possible compatibility problems with video adapters when Stealth ROM is in effect. This parameter has an effect only when used in conjunction with QEMM's Stealth ROM feature.

**FILL:N**

tells QEMM not to increase the size of conventional memory on systems that have unused memory addresses immediately above the top of conventional memory. By default, QEMM will fill conventional memory out to 640K on systems with less than 640K of conventional memory, and will extend conventional memory to 704K on monochrome and Hercules systems and to 736K on CGA systems. If QEMM has increased the size of conventional memory on your system, you will need the FILL:N parameter to run Microsoft Windows 3.0 or 3.1 in 386 enhanced mode, or to turn QEMM off when troubleshooting.



**FORCEEMS:Y (FEMS:Y)**

tells QEMM to allow programs to detect an EMS manager even when the EMS page frame is smaller than 64K due to the FRAMELENGTH=x or FRAME=NONE parameter.

Because many programs that use expanded memory assume that the page frame is 64K in size, QEMM usually makes the standard EMS detection tests fail when you specify a number smaller than 4 to the FRAMELENGTH parameter or use the FRAME=NONE parameter. FORCEEMS:Y makes an EMS manager detectable with a small page frame or with no page frame, for the benefit of expanded memory programs that may not use all 64K of the page frame. Some VCPI programs that will not run without a page frame sometimes work if you use FORCEEMS:Y. However, some programs that use expanded memory will fail if you use FORCEEMS:Y with a small page frame or no page frame.

**FORCESTEALTHCOPY:Y (FSTC:Y)**

forces QEMM to copy certain tables from the video ROM, disk ROM, and system ROM into its own data areas when the STEALTHROM parameter is relocating these ROMs. These tables contain video and disk parameters, and must be accessible at all times, unlike the rest of the ROM regions, which must be accessible only at particular times.

Normally, QEMM copies these tables to a new location in High RAM whenever Stealth ROM relocates the ROMs, unless the tables are accessible at their original locations because the EXCLUDE parameter covers those locations. FORCESTEALTHCOPY:Y makes QEMM copy these tables to a new location even when the tables lie in regions that you have excluded. This parameter can be useful to make QEMM's Analysis procedure give meaningful results when Stealth ROM is enabled.

**FRAME=xxxx or FRAME=NONE (FR=xxxx or FR=NONE)**

tells QEMM where to put the EMS page frame, a 64K memory region reserved for expanded memory mapping. xxxx is the four-digit hexadecimal segment address of the beginning of the page frame, which must be on a 16K boundary (i.e., the last three digits of the address should always be 000, 400, 800, or C00.)

QEMM automatically chooses an appropriate page frame location, so normally you would not need to use the FRAME parameter. If QEMM places the page frame in an area used by another program or hardware device, it is better to use the EXCLUDE parameter to prevent access to the disputed region than it is to use the FRAME parameter to move the page frame. You may want to use the FRAME parameter to change the location of the page frame if doing so will consolidate two or more High RAM areas into one larger area. Check the appropriate Manifest screens or the



QEMM.COM report display for information on your memory configuration. Be careful to avoid conflicts with memory that QEMM does not relocate, like areas that QEMM designates as Adapter RAM. Do not place the page frame at F000, even when Stealth ROM is enabled—some ROM code at the end of the F000 region must be in place at all times even with Stealth ROM in effect. QEMM will not prevent you from creating memory conflicts with a poorly chosen page frame location.

The `FRAME=xxxx` parameter is most often useful when you are troubleshooting the `STEALTHROM` parameter. Some ROMs work with the Stealth ROM feature only when you place the page frame exactly at the beginning address of the ROM. For instance, if your VGA ROM is at addresses C000-C7FF, and you experience video problems when you use the Stealth ROM feature to relocate this ROM, you can try the `FRAME=C000` parameter (assuming that the page frame at C000 does not overlay other regions filled with adapter RAM or un-Stealthed ROMs).

When the Stealth ROM mapping method is enabled, QEMM automatically excludes the C000-C0FF region (for compatibility with Super VGA video adapters) unless the page frame is at C000. This means that placing the page frame at C000 (the default location when the Stealth ROM mapping method is enabled) gives you 4K more of upper memory to use for High RAM.

The `FRAME=NONE` parameter tells QEMM not to create an EMS page frame. If you use this parameter, most programs will not be able to use expanded memory. The notable exceptions are DESQview and DESQview/X, which can still multitask in expanded memory even with the `FRAME=NONE` parameter. In addition, you will not be able to use the `STEALTHROM` parameter if you use `FRAME=NONE`. However, if you have no programs other than DESQview and DESQview/X that use expanded memory, and if you cannot make full use of the Stealth ROM feature, you may be able to create more High RAM by using `FRAME=NONE` to free 64K of upper memory addresses.

#### **FRAMEBUF:Y or N (FB:Y or N)**

enables or disables QEMM's feature of breaking up disk reads into the page frame and disk writes from the page frame (when these reads and writes are done using file handles as opposed to FCBs), so that the reads and writes are diverted into DOS's buffers.

By default, this feature is enabled when Stealth ROM is used, and disabled otherwise. `FRAMEBUF:Y` causes this feature to be enabled at all times; `FRAMEBUF:N` causes this feature to be disabled at all times. If you are not using Stealth ROM and you experience problems when running EMS-using TSRs or drivers at the same time as EMS-using applications, `FRAMEBUF:Y` may help.

**FRAMELENGTH=x (FL=X)**

tells QEMM to create an EMS page frame containing x 16K pages. x defaults to 4, which means that QEMM creates a four-page page frame covering 64K of memory addresses. Nearly all programs that use expanded memory expect a 64K page frame, so the FRAMELENGTH parameter is rarely needed. If you need to increase the amount of High RAM on your system, and if none of your expanded memory programs use all four pages of the page frame, you can use the FRAMELENGTH parameter with a number smaller than four. However, many expanded memory programs will not work properly with a small page frame, even if they do not use four pages for EMS mapping.

You must specify the FORCEEMS:Y parameter along with the FRAMELENGTH parameter if you want an expanded memory program to use a page frame smaller than four pages.

If you specify a number greater than four to the FRAMELENGTH=x parameter, QEMM creates a page frame larger than 64K. QEMM will change the numbering of EMS physical page addresses so that the pages in the large page frame have contiguous physical page numbers. Almost no programs benefit from a page frame larger than four pages, and a large page frame decreases the area available for High RAM, so you should use this option only if you have a specific purpose. If QEMM cannot fit the large page frame into an available region in upper memory, it will put the page frame in conventional memory, decreasing the memory available to run real-mode programs.

FRAMELENGTH=0 is equivalent to the FRAME=NONE parameter—it tells QEMM not to create a page frame. See the FRAME parameter for more information.

**GETSIZE (GS)**

is an internal parameter set by Optimize.

**HANDLES=nnn (HA=nnn)**

tells QEMM how many EMS handles to provide. Each program that uses expanded memory needs at least one handle; a single program can require multiple handles if it allocates expanded memory a bit at a time. nnn must be in the range 16 to 255; QEMM's default is 64 EMS handles, which should be adequate for most purposes.

If you think you need more handles, consult Manifest's Expanded Overview screen, which gives the total number of EMS handles and the number currently available. Each handle uses only 32 bytes of QEMM's memory pool.

## HELP

displays a list of the QEMM386.SYS parameters and a short description of what each does. QEMM will not load if you specify this parameter.

## HMA:N

tells QEMM to report the XMS High Memory Area (HMA) as already allocated, thus preventing programs from using it. This parameter is used only for troubleshooting purposes.

The XMS:N parameter also prevents XMS-using programs from accessing the HMA or any other XMS services, but will not prevent DESQview or DESQview /X from allocating it through a private interface to QEMM; the HMA:N parameter stops all programs, including DESQview and DESQview /X, from using the HMA, but allows programs to use other XMS services to allocate extended memory and upper memory.

## HMAMIN=nn

tells QEMM not to let any program use the XMS High Memory Area (HMA) unless it requests at least nnK of the HMA's memory. nn must be in the range 1-63. Because only one program can allocate the HMA through QEMM, this parameter might be useful if you have more than one program asking for the HMA and you want to be sure that the most memory-efficient program gets access.

## IBMBASIC:Y (IB:Y)

tells QEMM not to use the BASIC area of the IBM system ROM for High RAM or expanded memory mapping. A large area of the system ROM on IBM systems (F600-FDFF, a 32K region) contains BASIC code that is used only by some older BASIC programs. By default, QEMM makes this area available for High RAM or expanded memory mapping; use IBMBASIC:Y if you have an old BASIC program that uses the IBM BASIC area of the ROM. Newer BASIC programs like DOS 5 and 6's DOS Edit or DOS QBasic do not need the IBMBASIC:Y parameter.

## INCLUDE=xxxx-yyyy (I=xxxx-yyyy)

tells QEMM to use the address range xxxx-yyyy for High RAM or for expanded memory mapping. For a full description, see *page 55*.

## INCLUDE386=xxxx-yyyy (I386=xxxx-yyyy)

is, for QEMM's purposes, identical to the ;INCLUDE parameter. It is meant to be used in files (such as the MCA.ADL file) that contain parameters for QEMM's use but that may also be read by other Quarterdeck products like QEMM-50/60. Use it in these files instead of

INCLUDE when an inclusion is only appropriate on an 80386 or higher processor.

### **IOTRAP=nn**

changes the way that QEMM monitors I/O (input/output) ports. The default value of **nn** is 1, and the only other legitimate value is 64, which tells QEMM to assume that the same 1K of I/O ports is duplicated 64 times in the 64K I/O port address space. IOTRAP=64 is rarely useful, but you can try it whenever a program has difficulty accessing a hardware device when QEMM is loaded.

### **LABEL (LB)**

is an internal parameter set by Optimize.

### **LOCKDMA:Y (LD:Y)**

tells QEMM not to unlock interrupts at certain times during the setup of DMA (Direct Memory Access) transfers between hardware devices and memory. By default, QEMM enables interrupts as much as possible during DMA transfers for performance reasons. Certain networks, like 10NET, will fail when QEMM is loaded unless you use the LOCKDMA:Y parameter.

### **MAPREBOOT:N (MR:N)**

tells QEMM not to replace 4K of the System ROM with RAM. By default, QEMM shadows a particular 4K piece of the system ROM (that is, QEMM copies the ROM into RAM and makes the RAM appear in upper memory at the ROM's former location) and changes part of the ROM code so that QEMM can better detect warm reboots, which may fail if QEMM does not intervene in the reboot process. This 4K of system ROM is speeded up by being shadowed, and there is a small chance that some ROM code may fail if it executes at a faster speed.

QEMM can often detect warm reboots even without shadowing 4K of the ROM, so you can try the MAPREBOOT:N parameter if QEMM is causing problems with your floppy drives or other devices controlled by the system ROM. This parameter returns 4K to QEMM's memory pool. When the Stealth ROM feature is enabled, QEMM does not shadow 4K of the ROM to detect reboots, and the MAPREBOOT:N parameter has no effect. If, however, the FF00-FFFF area is excluded, then QEMM shadows this 4K of the ROM even when Stealth ROM is enabled, and the MAPREBOOT:N parameter will have an effect.

### **MAPS=nnn (MA=nnn)**

tells QEMM how many EMS alternate maps to provide. **nnn** must be in the range 0 to 255; 32 alternate maps is the default and is adequate for

almost all circumstances. Alternate maps are used by multitasking environments to switch EMS contexts quickly. DESQview and DESQview/X need alternate maps to multitask high-speed communications programs effectively, and to control the video output of programs that write directly to the video hardware.

QEMM dynamically allocates and deallocates memory out of its memory pool to satisfy requests for alternate maps; each map requested uses 4K of QEMM's memory pool. This means that there is no need to set MAPS to a lower number to save memory.

### **MEMORY=nnnnn (ME=nnnnn or MEM=nnnnn)**

specifies the amount of extended memory that should be under QEMM's control. The amount of memory specified by the MEMORY parameter will be used for QEMM's own overhead and for its memory pool; the remainder of the memory on the system will not be managed by QEMM. By default, QEMM manages all extended memory on the system. This parameter should only be used to provide memory for old utilities (for example, VDISK) that allocate extended memory without using the XMS, VCPI, or DPMI specifications. We recommend that, if possible, you get a new XMS-using version of your utility instead of using the MEMORY parameter. Memory excluded from QEMM's management by the MEMORY parameter will be unavailable to programs that use EMS, XMS, VCPI, or DPMI.

QEMM will not load if the MEMORY parameter restricts QEMM so severely that it cannot find enough memory for its own overhead. QEMM will also refuse to load if you have a system with more than 16 megabytes of memory and MEMORY restricts QEMM so that it cannot use any of the first 16 megabytes. (Unmanaged memory is always left behind at the bottom of the expanded memory region.) The lowest number you can safely use depends on your configuration, but leaving over 15000 of unmanaged memory is likely to cause a failure. See the EXTMEM parameter.

### **OFF (OF)**

See AUTO ON OFF.

### **OLDDV:Y (ODV:Y)**

makes changes to QEMM so that it is compatible with DESQview versions 1.3 and 2.00. You do not need this parameter if you use DESQview 2.01 or later.

### **ON**

See AUTO ON OFF.



## **PAUSE**

tells QEMM to pause for input when it loads, regardless of whether it detects an error. You can then hit the Esc key to unload QEMM or any other key to continue the loading process. The PAUSE parameter gives you an easy way to prevent QEMM from loading when you are trying out new parameters that may make the system unstable.

## **PAUSEONERROR:N (PE:N)**

tells QEMM not to pause to give an error message. QEMM normally displays an error message and pauses for input when you specify an incorrect parameter, when it detects a microchannel adapter not listed in the MCA.ADL file, and in other circumstances. When you use this parameter, QEMM will display error messages but will not pause. You can use PAUSEONERROR:N if QEMM is giving you an error message about a harmless condition that you prefer not to address (e.g., an "Unknown MCA Adapter ID" error for a microchannel adapter that you are certain uses no addresses in upper memory).

## **RAM or RAM=xxxx-yyyy**

specifies that QEMM should create High RAM in upper memory. For a full description, see *page 55*.

## **REGION:n (R:n)**

tells QEMM in which region of High RAM to place pieces of its own code, as well as 12-13K of video and disk parameter tables that it relocates when the Stealth ROM feature is enabled. The Optimize program will place the REGION parameter on the QEMM386.SYS device driver line when it is appropriate; you should normally not add this parameter manually.

## **ROM, ROM=xxxx or ROM=xxxx-yyyy**

tells QEMM to speed up the operation of ROMs by copying them into faster RAM and making the RAM appear in upper memory in place of the ROMs. For a full description, see *page 56*.

## **ROMHOLES:N (RH:N)**

tells QEMM not to detect unused areas of ROM and make their memory addresses available for High RAM or expanded memory mapping. By default, QEMM examines the system ROM for areas that are either empty or used only during the system's power-on self test, and marks these areas as available for other uses. This feature is implemented only when QEMM does not relocate the entire system ROM with the Stealth ROM feature.

If QEMM mistakenly reclaims a ROM area that is in use, you may experience problems with your floppy drives or with other devices or functions controlled by ROMs. Both the ROMHOLES:N and the EXCLUDE=xxxx-yyyy parameter prevent QEMM finding "ROM holes," and the address range of the EXCLUDE parameter can be adjusted to leave you some of the benefits of QEMM's reclamation of ROM addresses; therefore, if the ROMHOLES:N parameter fixes a problem, you should probably use QEMM's Analysis procedure (see The Analysis Report on *page 97*) to find appropriate EXCLUDE parameters to use instead of ROMHOLES:N.

### **SHADOWRAM:LEAP, NEAT, NEC, OPTI, PEAK, SCAT, TOPCAT, 386, or NONE (SH)**

forces QEMM to reclaim various types of "shadow memory" for its memory pool, or tells QEMM not to reclaim it. Shadow memory is 384K of RAM that can be made to appear in the upper memory area on some systems. It serves the same function as QEMM's ROM parameter: systems with shadow memory can copy ROMs and other memory in the upper memory area into this 384K of RAM to speed up system activities.

By default, QEMM automatically detects various kinds of shadow memory and reclaims any unused portions for its memory pool. In the unlikely event that QEMM fails to detect a type of shadow memory that it knows how to manage, you can use the SHADOWRAM parameter to force QEMM to recognize the shadow memory. LEAP, NEAT, PEAK, SCAT, and 386 refer to different varieties of Chips & Technologies ShadowRAM; NEC refers to a variant of ShadowRAM that is found on some NEC systems; and OPTI and TOPCAT refer to the shadow memory of the OPTI and VLSI TOPCAT chip sets, respectively.

The SHADOWRAM:NONE parameter disables this QEMM feature, and is a common troubleshooting option. You should try this parameter if your system fails at the point when QEMM loads, or if your system reboots when QEMM loads.

You must sometimes enable shadow memory in your system setup for QEMM to detect it. If your system lets you reconfigure shadow memory as extended memory, you should do so and let QEMM shadow your ROMs with its ROM parameter. This typically increases QEMM's memory pool by 96K (192K if you do not use the ROM parameter) on EGA/VGA systems.

### **SORT:Y**

tells QEMM to test the speed of all system memory and to use the fastest memory first. This parameter can sometimes speed up systems that contain memory chips of different speeds—especially systems with slower motherboard memory and faster memory on an add-in card.

However, Microsoft Windows 3.0 and 3.1 will not run in 386 enhanced mode if the SORT:Y parameter makes QEMM use faster extended memory in place of conventional memory. You can use Manifest's Expanded Timings screen to see if different parts of your memory run at different speeds.

#### **STEALTHROM:x (ST:x)**

enables the Stealth ROM feature—the QEMM technology that relocates ROMs from upper memory so that their addresses can be used for High RAM or for expanded memory mapping. For a full description, see *page 57*.

#### **SUSPENDRESUME (SUS) or SUSPENDRESUME:nn (SUS:nn)**

forces QEMM to enable its support for Suspend/Resume, a feature built into many laptop systems that allows you to operate the laptop on low power when it is not in use and to restore the system to its previous state when you return to it.

QEMM detects Suspend/Resume automatically on some systems and will enable Suspend/Resume support on them without the **SUSPENDRESUME** parameter. If your laptop supports the Suspend/Resume feature and that feature does not work properly after installing QEMM, the **SUSPENDRESUME** parameter will make QEMM search for the hardware interrupt that the feature uses, and enable its Suspend/Resume support if it finds the interrupt.

If QEMM cannot find the Suspend/Resume hardware interrupt, you can use the **SUSPENDRESUME:nn** parameter, where **nn** is the hexadecimal number of the appropriate hardware interrupt. The documentation for your laptop may tell you which interrupt the Suspend/Resume feature uses; if it does not, 2, D, 72, 73, and 77 are the most likely possibilities.

#### **TASKS=nn (TA=nn)**

tells QEMM how many interrupts it can keep track of at one time when it is in protected mode. **nn** must be in the range 2 to 40; the default is 16. Though you will probably not need to use the **TASKS=nn** parameter, you can try increasing the value of **nn** if your system halts suddenly in the middle of high-speed communications or other interrupt-intensive activity.

#### **TOKENRING:N (TR:N)**

tells QEMM not to detect the presence of a Token-Ring adapter. By default, QEMM will use special methods to ensure that it does not place High RAM or allow expanded memory mapping in upper memory areas occupied by the Token-Ring card's adapter RAM and ROM. You should try the **TOKENRING:N** parameter if you have another adapter that does

not work or cannot be detected when QEMM is loaded. If this parameter solves your problem, and if you have a Token-Ring adapter as well, you can simply use the EXCLUDE=xxxx-yyyy parameter for any Token-Ring memory addresses that QEMM has not already detected.

#### **TOPMEMORY:N (TM:N)**

tells QEMM not to try to reclaim "top memory," memory located just below the 16-megabyte point on some systems and reserved for speeding up ROMs and for other system functions. By default, QEMM automatically detects unused portions of top memory and places them into QEMM's memory pool, thus making it available through any interface that QEMM supports. TOPMEMORY:N disables this QEMM feature, and is a common troubleshooting option. You should try this parameter if your system fails at the point when QEMM loads, or if your system reboots when QEMM loads.

#### **TRAP8042:Y (T8:Y)**

tells QEMM to monitor output to the 8042 keyboard controller. The keyboard controller contains a bit that enables A20, a hardware address line that determines whether programs can access extended memory. You can use the TRAP8042:Y parameter if you believe that a program that is accessing extended memory is interfering with QEMM's memory management. You may also need TRAP8042:Y to warm reboot successfully on Intel Inboard-AT systems.

Forcing QEMM to monitor the controller port adds overhead to the processing of keystrokes. If you are using a resident program or device driver that reads keystrokes incorrectly from the keyboard controller port, TRAP8042:Y can sometimes cause the shift states on your keyboard to become reversed, the arrow keys on your cursor keypad to produce numbers instead of cursor movement, and other problems with the keyboard or with the mouse on a PS/2-style mouse port.

#### **UNMAPFREEPAGES:Y or N (UFP:Y or N)**

tells QEMM whether to remove pages of expanded memory from the EMS page frame when an EMS-using program has freed the expanded memory handle associated with these pages.

UNMAPFREEPAGES:Y is the default when Stealth ROM is enabled—this makes it possible to use Stealth ROM along with certain Super VGA video cards without excluding parts of the video ROM area. However, UNMAPFREEPAGES:Y is incompatible with certain EMS-using applications, including some versions of Glyphix, VP Planner, and 1DIR Plus. If you have a problem running an EMS-using program when Stealth ROM is enabled, you should try UNMAPFREEPAGES:N, though this parameter may force you to exclude parts of the C000-C7FF area.

UNMAPFREEPAGES:N is QEMM's default when Stealth ROM is not in effect, and there is usually no need to try UNMAPFREEPAGES:Y when Stealth ROM is not active.

### **UNUSUALEXT:Y (UX:Y)**

tells QEMM to change its method of determining the amount of extended memory on the system. You will probably not need this parameter, but you can try it if your system hangs as soon as QEMM loads.

UNUSUALEXT:Y may cause QEMM.COM's Memory report and Manifest's QEMM Memory screen to have inaccurate extended memory reports.

### **USERAM=xxxx-yyyy (UR=xxxx-yyyy)**

tells QEMM that the system has placed RAM in the address range xxxx-yyyy that QEMM should reclaim and place in its memory pool. If your system is using a type of shadow memory that QEMM cannot detect automatically, you can specify the USERAM parameter for any regions of upper memory that do not contain ROMs, video RAM, or adapter RAM. QEMM will add the shadow memory in these unused regions to its memory pool, and automatically make these regions available for High RAM or expanded memory mapping.

You can also specify the USERAM parameter to tell QEMM about RAM at extended memory addresses that QEMM cannot detect automatically. To do this, it is easier to use the alternative methods of specifying an address range that are discussed in the section *Parameters and Memory Addresses* earlier in this chapter. For instance, USERAM=16M:384K tells QEMM that your system has 384K of available RAM starting at the 16 megabyte point.

### **USEXMS:N**

tells QEMM not to allocate memory from a previously loaded XMS manager, if one exists, and to obtain extended memory only through the BIOS interface. Normally, QEMM will allocate all memory away from a previously loaded XMS manager (e.g., DOS's HIMEM.SYS). USEXMS:N also prevents QEMM's default practice of letting a previously loaded XMS manager do all manipulations of the A20 address line, which determines whether programs can access extended memory. This parameter is rarely used.

### **VDS:N**

disables QEMM's support of the Virtual DMA Services (VDS) specification, which resolves incompatibilities between 386 memory managers and bus-mastering hardware devices. This parameter should be used only for troubleshooting purposes.



### **VIDEOFILL:N (VF:N)**

tells QEMM not to increase the size of conventional memory on systems that do not use the video areas just above 640K. By default, QEMM will extend conventional memory to 704K on monochrome and Hercules systems and to 736K on CGA systems, unless some hardware or software on your system has placed its code or data just below 640K and QEMM cannot move that code or data. If QEMM has increased the size of conventional memory on your system, you will need the VIDEOFILL:N or the FILL:N parameter to run Microsoft Windows 3.0 or 3.1 in 386 enhanced mode, or to turn QEMM off when troubleshooting. The VIDEOFILL:N parameter is a weaker version of the FILL:N parameter, which prevents filling conventional memory into areas both above and below 640K.

### **VIDEORAM:N (VR:N)**

tells QEMM not to create high RAM in upper memory areas reserved for video RAM (640K-768K). By default, QEMM will create High RAM in all available areas of upper memory when the RAM parameter is in effect, including any unused areas between 640K and 768K.

You can use either the VIDEORAM:N parameter or the VIDRAMEGA parameter if you plan to use the VIDRAM program's VIDRAM ON or VIDRAM ON EGA option to extend conventional memory on a color EGA or VGA system, and you want conventional memory to be extended to 736K instead of 704K. (The larger amount of conventional memory comes at the expense of 32K of High RAM at B000-B7FF.) Of the two parameters, VIDRAMEGA may be the safer option if you have programs that make extensive use of expanded memory mapping; VIDEORAM:N, on the other hand, has the advantage of leaving the B000-B7FF area available for expanded memory mapping by DESQview or DESQview /X.

If you want to stop QEMM from putting High RAM in the video RAM areas for troubleshooting purposes, you should normally use the EXCLUDE=A000-BFFF parameter instead of the VIDEORAM:N parameter. See **Chapter 6** for information on the VIDRAM Program.

### **VIDRAMEGA (VREGA)**

tells QEMM to make certain video regions unavailable for High RAM or expanded memory mapping. It is equivalent to the parameter EXCLUDE=A000-B7FF.

The VIDRAMEGA parameter is useful when you are using QEMM's VIDRAM program. If you use VIDRAM ON or VIDRAM ON EGA, the VIDRAMEGA parameter will insure that VIDRAM increases the size of conventional memory by 96K instead of 64K. Also, VIDRAMEGA is useful if you are using VIDRAM to increase the memory available to a DOS window inside Microsoft Windows' 386 enhanced mode, and you

want the Windows DOS window to increase in size by 96K instead of 64K. In both cases, VIDRAMEGA will decrease by 32K the amount of upper memory available for High RAM and expanded memory mapping on most EGA/VGA systems. See **Chapter 6** for information on VIDRAM.

### **VIDRAMEMS (VREMS)**

tells QEMM to make the video regions just above 640K available for expanded memory mapping, but not to fill them with High RAM or to extend conventional memory into them.

If you are using the VIDRAM program with the VIDRAM ON EMS option, you will need the VIDRAMEMS parameter. It will decrease by 32K the amount of upper memory available for High RAM on most EGA/VGA systems. This parameter may cause problems with EMS-using programs that make extensive use of expanded memory mapping. Do not use VIDRAMEMS with versions of DESQview prior to Version 2.26. See **Chapter 6** for information on VIDRAM.

### **VIRTUALHDIRQ:N (VHI:N)**

tells QEMM not to disable the “advanced disk features” of some disk caches when Stealth ROM is enabled. Sophisticated disk caches often take advantage of a BIOS feature that lets them give processor time to applications during the waiting time while the disk controller is writing a sector to disk. Unless the disk cache takes the proper precautions, these advanced disk features are incompatible with QEMM’s Stealth ROM feature. Therefore, QEMM disables advanced disk features by default whenever Stealth ROM causes QEMM to intercept the hard disk’s software interrupt, INT 13. VIRTUALHDIRQ:N tells QEMM to permit advanced disk features even when Stealth ROM causes QEMM to intercept INT 13.

When Stealth ROM is not enabled or is not causing INT 13 to be intercepted, the VIRTUALHDIRQ parameter has no effect.

Disk caches that have advanced disk features that are compatible with Stealth ROM will communicate with QEMM to enable these features; it is therefore unwise to use the VIRTUALHDIRQ:N parameter when Stealth ROM is enabled and you are loading a disk cache.

### **VCPISHARE:Y (VS:Y)**

tells QEMM to make updates to VCPI protected-mode programs’ first page table, which contains the VCPI program’s map of the first four megabytes of memory. This parameter may help some VCPI programs run properly, especially when the VCPI program uses EMS memory or when it is being swapped out of memory by DESQview. However, VCPISHARE:Y can also cause some VCPI programs to fail.

**VXDDIR=directory\_name**

tells QEMM the location of its .VXD files, which it uses to provide various features when Microsoft Windows 3.0 and 3.1 is running in 386 enhanced mode. You need this parameter if you use a diskless workstation on a network, or if your .VXD files are not in the directory from which QEMM is loaded.

**WATCHDOG=0,1, or 2 (WD=0, 1 or 2)**

tells QEMM the type of hardware, if any, that is available on your system for QEMM to use to implement Level 1 of DESQview and DESQview/X's Protection Level feature. Protection Level 1 notifies the user when a program has locked interrupts for too long—a common cause of system failures. Under normal circumstances, QEMM supports this DESQview and DESQview/X feature only on IBM PS/2 systems and on Compaq systems, each of which contains watchdog timer hardware that QEMM automatically detects.

You can use the WATCHDOG=1 or WATCHDOG=2 parameter if your system contains watchdog timer hardware that QEMM knows how to use but does not detect. WATCHDOG=1 tells QEMM to look for a PS/2-style watchdog timer, and WATCHDOG=2, a Compaq-style or an EISA-style watchdog timer. You should note in particular that QEMM does not automatically detect the watchdog hardware on EISA systems, which need WATCHDOG=2 to make Protection Level 1 work at all in DESQview and DESQview/X. WATCHDOG=0 is the default on systems that are not PS/2s or Compaqs, and disables Protection Level 1 in DESQview and DESQview/X. If you experience problems when using Protection Level 1 in DESQview or DESQview/X, you can try the WATCHDOG=0 parameter.

**WINDOWS3:N (W3:N)**

disables QEMM's special support for Microsoft Windows 3.0 in standard and 386 enhanced modes and Microsoft Windows 3.1 in 386 enhanced mode. When you specify the WINDOWS3:N parameter, you can run Microsoft Windows 3.0 only in real mode, and Microsoft Windows 3.1 only in standard mode. This parameter changes slightly the information that QEMM returns to some EMS calls, and therefore has a small chance of preventing problems with programs that use expanded memory.

**WINSHRINKUMBS:N (WSU:N)**

tells QEMM not to shrink the amount of available High RAM when Microsoft Windows 3.0 or 3.1 starts up in 386 enhanced mode. By default, QEMM gets rid of most unused areas of High RAM when it sees Windows start up in this mode, because freeing these areas of upper memory usually increases the size of DOS windows by 8-24K.

**Using the ROM  
and  
STEALTHROM  
Parameters  
Together**

**WINSHRINKUMBS:N** makes it possible for programs running inside Microsoft Windows' 386 enhanced mode to use available High RAM, but also will frequently decrease the size of DOS windows when Windows is running in 386 enhanced mode.

**XBDA:N**

tells QEMM not to move the Extended BIOS Data Area (XBDA) away from the top of conventional memory. By default, QEMM detects the XBDA on systems that have it (including IBM PS/2s and nearly all systems with PS/2-style mouse ports) and moves it to a lower place in memory so that it does not prevent video filling on monochrome, Hercules and CGA systems or the use of VIDRAM on EGA/VGA systems, and does not decrease the size of DESQview or DESQview/X windows by 16K. XBDA:N is a common troubleshooting option, and is needed whenever a ROM or an application assumes that the XBDA is located at the top of conventional memory.

**XMS:N**

tells QEMM not to provide support for the Extended Memory Specification (XMS), which many programs use to access extended memory and upper memory. You should normally only use this parameter for troubleshooting. The XMS:N parameter will not prevent DESQview and DESQview/X from using the High Memory Area (HMA); use the HMA:N parameter for that purpose.

When you specify the **STEALTHROM:x** parameter to QEMM, ROM code is managed by QEMM in such a way that upper memory ROM areas are then free for High RAM or expanded memory mapping.

QEMM processes its parameters sequentially—if two mutually exclusive parameters appear on the QEMM386.SYS line in the CONFIG.SYS file, the one that comes later overrides the earlier one. This is primarily important when two QEMM parameters make claims on the same address range.

One situation in which the ordering of parameters is important is when the ROM parameter and the **STEALTHROM:x** parameter are both specified. There is no conflict or overlap between the function of these two parameters—the presence of the ROM parameter simply means that the **STEALTHROM:x** parameter will be relocating fast ROMs instead of slow ones. However, when you specify another parameter that affects a ROM address range, whether that parameter affects the mapping of ROM into faster RAM depends on if it precedes or follows the **STEALTHROM:x** parameter. Here is an example of such a parameter set:

**ROM EXCLUDE=FC00-FFFF STEALTHROM:M**

Here, the ROM parameter tells QEMM to map ROM areas into faster RAM. The EXCLUDE=FC00-FFFF parameter prevents RAM from being mapped into the FC00-FFFF area, so the ROMs in that area will not be mapped into RAM. (The ROM code that controls the floppy drive may be in the FC00-FFFF range, and in some cases, floppy drive access is impaired if the ROM code that controls the floppy drive controller is sped up.)

The following parameter set will have a different effect:

#### **ROM STEALTHROM:M EXCLUDE=FC00-FFFF**

In this case, the EXCLUDE=FC00-FFFF parameter no longer affects the mapping of ROM into faster RAM; the sequencing of the parameters ensures that the ROM has been relocated by the STEALTHROM:M parameter before the exclusion comes into play. The EXCLUDE=FC00-FFFF parameter still tells QEMM not to make that range available for High RAM or EMS mapping. The ROM code at FC00-FFFF can still be accessed at its old address. By placing the EXCLUDE after the STEALTHROM:M parameter, you can speed up the entire system ROM and still prevent other access to the excluded area.

In addition to EXCLUDE, there are other parameters that can affect the ROM parameter's range, and will therefore behave differently on different sides of the STEALTHROM:x parameter. They are:

```
INCLUDE=xxxx-yyyy  
INCLUDE386=xxxx-yyyy  
ADAPTERROM=xxxx-yyyy  
ADAPTERRAM=xxxx-yyyy  
RAM=xxxx-yyyy
```

When you specify the RAM parameter without any address range, it does not require the same care as RAM=xxxx-yyyy when you are ordering your parameters. RAM without any addresses simply means "Put High RAM everywhere that it will not conflict with anything else," and will not change the extent of the ROM parameter's influence.

If you specify an address range with the ROM parameter (e.g., ROM=C000-C7FF), you must place the ROM parameter before the STEALTHROM:x parameter; otherwise the ROM parameter will be ignored.



Using  
Parameters  
From Previous  
Versions of  
QEMM

If you have upgraded from an earlier version of QEMM, be aware that many of the old parameters have new names. You can still use the older parameter names if you like. This section provides you with a list of the old parameter names cross referenced with the parameters' new names. Some parameters have abbreviations which are listed in parentheses.

Old Name	New Name
COMPAQ386S (C386S)	COMPAQ386S:Y (C386S)
COMPAQEGAROM (CER)	COMPAQEGAROM:Y (CER)
COMPAQHALLFROM (CHR)	COMPAQHALLFROM:Y (CHR)
COMPAQROMMEMORY CRM	COMPAQROMMEMORY:Y (CRM)
DONTUSEXMS (DUX)	USEXMS:N
DOS4 (D4)	DOS4:Y (D4)
FORCEEMS (FEMS)	FORCEEMS:Y (FEMS)
FORCESTEALTHCOPY (FSTC)	FORCESTEALTHCOPY:Y (FSTC)
IGNOREA20 (IA)	TRAP8042:Y (T8) (the default is reversed from earlier versions)
LOCKDMA (LD)	LOCKDMA:Y (LD)
NOCOMPAQFEATURES (NCF)	COMPAQFEATURES:N (CF)
NOEMS	EMS:N
NOFILL (NO)	FILL:N
NOHMA	HMA:N
NOPAUSEONERROR (NOPE)	PAUSEONERROR:Y (PE)
NOROM (NR)	MAPREBOOT:N (MR)
NOROMHOLES (NRH)	ROMHOLES:N (RH)
NOSHADOWRAM (NOSH)	SHADOWRAM:NONE (SH)
NOTOKENRING (NTR)	TOKENRING:N (TR)
NOTOPMEMORY (NT)	TOPMEMORY:N (TM)
NOVDS	VDS:N
NOVIDEOFILL (NV)	VIDEOFILL:N (VF)
NOVIDEORAM (NVR)	VIDEORAM:N (VR)
NOWINDOWS3 (NW3)	WINDOWS3:N (W3)

NOXBDA (NX)	XBDA:N
NOXMS	XMS:N
OLDDV (ODV)	OLDDV:Y (ODV)
UNUSUALEX (UX)	UNUSUALEX:Y (UX)



## The LOADHI Programs

This chapter is a technical reference for the QEMM's LOADHI programs which load device drivers and TSRs into High RAM. QEMM's Optimize program automatically adds LOADHI commands where necessary to cause these items to be loaded high. Read this chapter if you want to find out what items are loaded high or learn how to use the LOADHI commands.

QEMM's LOADHI.SYS and LOADHI.COM programs load TSRs and device drivers into available regions of High RAM. The Optimize program adds the necessary LOADHI statements to your CONFIG.SYS and AUTOEXEC.BAT files to load these items high. The purpose of loading TSRs and device drivers high is to free up more conventional memory for running DOS programs. If LOADHI cannot find enough High RAM to load an item high, it will use conventional memory instead.

To load items high, your PC must have High RAM. The installation automatically creates High RAM by adding the RAM parameter to the QEMM386.SYS line of your CONFIG.SYS file. If you do not have High RAM, you can run Optimize to create it and to add LOADHI statements to your CONFIG.SYS and AUTOEXEC.BAT files (see Chapter 3 for information on Optimize).

This chapter explains when and how you can use the LOADHI programs. There are several optional parameters that determine LOADHI's effectiveness in freeing up conventional memory. Freeing up this memory may enable you to:

- ☐ Run programs that would not fit in memory before.
- ☐ Add TSRs you have been doing without.
- ☐ Speed up memory-starved programs that no longer need to go to disk for their data.
- ☐ Increase the memory available to applications running in multitasking programs such as Microsoft Windows, DESQview, and DESQview/X.

If LOADHI is unable to load a particular item high, the item will load into conventional memory. Thus, your device drivers and TSRs will be available even if they are not loaded high.

### Using LOADHI

Both LOADHI.SYS and LOADHI.COM load programs into High RAM. You use LOADHI.SYS to load device drivers with an appropriate **DEVICE=** statement in your CONFIG.SYS file. You use LOADHI.COM to load programs from the DOS prompt, or from within your AUTOEXEC.BAT or other batch file. Both LOADHI programs support the same set of parameters which you can use to modify the way LOADHI normally allocates and uses High RAM. These parameters are described later in this chapter.

## The LOADHI Report

The LOADHI report tells you which High RAM addresses are in use, the amount of High RAM used and what resources are using it. This report is your starting point if and when you need to use LOADHI's optional parameters.

To display the LOADHI report:

- At the DOS prompt, type **LOADHI** and press Enter ↵.

C:\>loadhi

Region	Area	Size	Status
1	B001 - B07A	1.8K	Used (QDPMI)
1	B07B - B0AB	0.7K	Used (SETVER)
1	B0AC - B0B7	0.1K	Used (DU)
1	B0B8 - B5D7	20K	Used (LSL)
1	B5D8 - B7FE	0.5K	Used (DU)
2	D000 - D38B	14K	Used (QEMM386)
2	D38C - D391	0.1K	Used (DU)
2	D392 - DA9A	28K	Used (SMARTDRV)
2	DA9B - DAA4	0.1K	Used (DU)
2	DAAS - DED0	16K	Used (MOUSE)
2	DED1 - DEDC	0.1K	Used (DU)
2	DEDD - DFDE	4K	Used (NE2000)
2	DFDF - E3C0	15K	Used (IPXODI)
2	E3C1 - E8EB	20K	Used (TCP/IP)
2	E8EC - F364	41K	Used (NETX)
2	F365 - FFAS	49K	Used (DU)

The LOADHI report

High RAM areas may be scattered throughout the upper memory area due to the presence of ROM and adapter RAM. Each contiguous area of memory converted into High RAM is called a *region* and QEMM gives each region a number (listed in the left-most column of the report). These regions may vary in size.

To load items high, LOADHI allocates blocks of memory from these regions. Each allocated block displays in the list (the Area column) for the region, and the region's available memory is reduced accordingly. Any unused areas will be marked as Available.

If you want to see the number of regions, their address ranges, and sizes, display the LOADHI report before loading anything high. The sum of the regions' sizes gives you the maximum amount of space available to load items high.

You use LOADHI.SYS in your CONFIG.SYS file to load device drivers into High RAM at system startup time. LOADHI.SYS can load a device driver file that contains any number of device driver definitions.

If you have not already run QEMM's Optimize program, we suggest you do so. Optimize adds the appropriate LOADHI statements to your CONFIG.SYS file to load your device drivers high. This section is for

## Loading Device Drivers High with LOADHI.SYS

those who want to manually add or edit LOADHI statements in CONFIG.SYS.

Any statement in your CONFIG.SYS file that begins with the keyword **DEVICE=** (or **DEVICEHIGH=**) tells DOS to load the device driver following that keyword. Some common devices are memory managers, add-on peripheral device drivers (e.g., disk drive, mouse), and extensions to existing devices (e.g., DOS's ANSI.SYS driver).

A few device drivers are location-sensitive and will not work properly when loaded high and some other device drivers may require that you experiment with LOADHI parameters (you should not load QEMM386.SYS, DOSDATA.SYS or DOS-UP.SYS high). But generally, you should have no trouble loading device drivers high. To load a device driver high:

- Add **DEVICE=C:\QEMM\LOADHI.SYS**, followed by the device driver's full pathname and any parameter it takes, to your CONFIG.SYS file.

All LOADHI statements must appear after the QEMM386.SYS line. Be sure to include any parameters you normally use with your device driver.

The ANSI.SYS device driver is included with DOS, so we will use it to demonstrate how to load a device high. Normally, you load ANSI.SYS with the line **DEVICE=C:\DOS\ANSI.SYS**. To load it high, you would use

```
DEVICE=C:\QEMM\LOADHI.SYS C:\DOS\ANSI.SYS
```



*If you need control over the placement of device drivers or if you have problems loading a device driver high, see **The LOADHI Parameters** later in this chapter.*

#### Loading TSRs High with LOADHI.COM

You can use LOADHI.COM to load TSRs into High RAM. You can load TSRs from your AUTOEXEC.BAT file, another batch file or from the DOS prompt.

If you have not already run QEMM's Optimize program, we suggest you do so. Optimize adds the appropriate LOADHI statements to any TSR lines in your AUTOEXEC.BAT file to load those TSRs high. This section is for those who want to manually add or edit LOADHI statements in AUTOEXEC.BAT or load a TSR high by typing a command at the DOS prompt.

To load a TSR high:

- Type **C:\QEMM\LOADHI** followed by the command you normally use to load the TSR. You could add this statement to AUTOEXEC.BAT or type it at the DOS prompt.



## The LOADHI Parameters

To demonstrate how to load a TSR high, we will use an imaginary TSR program called DOITALL. To run DOITALL, you normally use the command `C:\UTILS\DOITALL /R`. This statement could be in your AUTOEXEC.BAT file, in some other batch file, or you could simply type it at the DOS prompt.

To load DOITALL into High RAM, change the above statement to `C:\QEMM\LOADHI C:\UTILS\DOITALL /R`



*If you need control over the placement of TSRs or if you have problems loading a TSR high, see **The LOADHI Parameters** later in this chapter.*

LOADHI has several optional parameters that may help if you have trouble loading one or more of your programs into High RAM.

The LOADHI command line has the following syntax:

**LOADHI [loadhi-opts] target-program [target-program-opts]**

The brackets in the statement indicate that the item specified is optional. The target program represents the device driver or TSR you are loading.

Note the following:

- ❑ All LOADHI options begin with a forward slash (/).
- ❑ Following the slash is the option name, or its abbreviation.
- ❑ If the option can take a value, then the next character must be a colon (:) followed by the value.

For example, `/LARGEST[:n]` (`/L`) means you can use any of the following:

`/LARGEST`  
`/L`  
`/LARGEST:2`  
`/L:2`

The LOADHI parameters are described below. You can combine options if necessary. Some parameters have an abbreviation you can use instead of the full name; abbreviations are listed in parentheses below. If a parameter has an optional value, it is enclosed in square brackets ([ ]).

### **/BESTFIT (/B)**

tells LOADHI to use the smallest block of memory in which the program fits. Using BESTFIT tends to leave larger regions for loading larger items. See also **SIZE** and **GETSIZE**.

### **/HAPPIEST (/H)**

tells LOADHI to use the smallest block of memory in which the program will fit, provided that it does not terminate with an error. Both device

drivers and TSRs return after initializing, and the LOADHI programs can determine if the load was successful. If it was not, then LOADHI will try again with a larger area until the program exits successfully. If necessary the program will be loaded in conventional memory.

**/REGION:n (/R:n)**

tells LOADHI to load the program into region number **n**. If you want to specify a region for a program that is capable of loading itself high (e.g., Hyperdisk, Super PC-Kwik, certain Norton Utilities programs) use LOADHI /LO /R:n.

**/LARGEST[:n] (/L[:n])**

tells LOADHI to load the program into the largest available block or a particular block out of several large blocks available. The number indicated by **n** indicates which block to use. For instance, the option /L:2 specifies the second largest available block.

**/SMALLEST[:n] (/S[:n])**

tells LOADHI to load the program into the smallest available block or one particular block out of several small blocks. The number indicated by **n** indicates which of these to use. For instance, the option /S:2 specifies the second smallest block.

**/EXCLUDEREGION:n (/XR[:n])**

tells LOADHI to not use region number **n** to load the target program.

**/EXCLUDELARGEST[:n] (/XL[:n])**

tells LOADHI to not use the largest block (or the **n**th largest block) to load the target program.

**/EXCLUDESMALLEST[:n] (/XS[:n])**

tells LOADHI to not use the smallest block (or the **n**th smallest block) to load the target program.

**/GETSIZE[:file] (/GS[:file])**

tells LOADHI to report the amount of memory a program requires. With this option LOADHI loads the program you specify and reports three memory usage values. The first value tells you how many bytes the program required to load and initialize. The second value tells you how many bytes the program uses once resident. The third value is the amount of upper memory used once the program is resident. LOADHI also tells you if the program can use the squeeze feature (see the SQUEEZEF and SQUEEZET parameters below). You use the GETSIZE

option to help you custom fit your device drivers and TSRs into High RAM.

The optional variable, **file**, specifies a filename. When a filename is present, LOADHI writes the program name, the three size values and other information into this file so that you can examine it later. If the filename you specify already exists, LOADHI appends the information to this file. By systematically using the GETSIZE option with a filename you can compile the memory requirements of all the programs you wish to relocate to High RAM. This is the procedure that the Optimize program uses.

#### **/SHELL**

tells LOADHI to load a command processor into upper memory. QEMM's DOS-Up feature adds the LOADHI command with this parameter to the SHELL line in CONFIG.SYS. The /SHELL parameter only works with LOADHI.COM.

#### **LABEL:nn (/LB:nn)**

is an internally used parameter.

#### **/LINKTOP (/LINK)**

connects free memory in High RAM to the DOS memory chain. Programs which make heavy demands on DOS memory may benefit from the additional pool of memory. Run LOADHI /LINK before loading the application program which can use this extra memory. When using LINK, it must be the only parameter on the LOADHI command line—you cannot also specify a program to be loaded.

#### **/UNLINKTOP (/UNLINK)**

disconnects free memory in High RAM from the DOS memory chain. When using UNLINK, it must be the only parameter on the LOADHI command line. See the /LINK parameter above.

#### **/SIZE:nnnn[K]**

tells LOADHI to allocate from a block that will best fit the size **nnnn**, where **nnnn** is a number expressed in bytes (e.g., 4096) or in kilobytes (e.g., 4K). The number you supply may come from the report issued by the GETSIZE option, or a number that you have determined by other means. In either case this number should represent the amount of memory the program needs to successfully initialize.

#### **/NOLO (/NL)**

tells LOADHI to prevent the program from being loaded in conventional memory if it does not fit in a High RAM region. This allows you to specify

device drivers or TSRs that you would like to use, but only if they will fit in High RAM.

#### **/LO**

tells LOADHI to unconditionally use conventional memory instead of High RAM. You can use this parameter to temporarily change LOADHI statements in CONFIG.SYS or AUTOEXEC.BAT files without removing them completely. See the **REGION** parameter.

#### **/QUIET (/Q)**

suppresses the display of non-fatal error messages when LOADHI loads a program.

#### **/RESIDENTSIZE=nnnn[K] (/RES=nnnn[K])**

is a parameter used internally by the Optimize program. It tells LOADHI the resident size of the program being loaded into High RAM. The size nnnn is expressed in bytes (e.g., 4096) or kilobytes (e.g., 4K).

/RESIDENTSIZE is used in conjunction with the /SQF or /SQT parameter; it allows LOADHI to verify that there is enough memory left in the chosen region to safely load the program.

#### **@[:file]**

tells LOADHI to look into a file to find the parameters for the program being relocated. @ may be followed by a colon and a filename (or pathname, if necessary). If you do not specify a file, LOADHI.COM will look for the environment variable LOADHIDATA (automatically created by Optimize when it is run with the /RESPONSE parameter), to determine the name and location of the file. The default filename is \QEMM\LOADHI.RF.

#### **/SQUEEZEF (/SQF)**

is an internally used parameter generated by the Optimize program. It instructs LOADHI to make temporary use of the page frame to give a device or TSR program sufficient room to initialize. The target program must not require the presence of this additional memory after it becomes resident. See also the /SQT parameter below. QEMM's Stealth ROM feature uses the page frame, so if Stealth ROM is enabled, LOADHI's SQUEEZEF parameter will be ignored.

#### **/SQUEEZET (/SQT)**

is an internally used parameter generated by the Optimize program. It instructs LOADHI to make temporary use of an area of upper memory for the purpose of giving a device or TSR program sufficient room to initialize. The temporary memory is not High RAM, but is memory that

Optimize determines can be safely used during the program's initialization. The target program must not require the presence of this additional memory after it becomes resident. See also the */SQF* parameter above.

#### **/TERMINATERESIDENT (/TSR)**

tells LOADHI to terminate as a TSR, leaving a small stub of code (about 100 bytes) resident. This option is only useful if you are using LOADHI in combination with DOS's INSTALL command in CONFIG.SYS. In this case, the effect of this option is simply to suppress an error message issued by DOS indicating a failure to load a TSR when in fact LOADHI successfully relocated the program.

#### **/HELP**

displays a list of the LOADHI parameters and a brief description of each one.

#### **/?**

displays a list of the LOADHI parameters.

Optimize will replace DOS DEVICEHIGH and LOADHIGH statements with the equivalent LOADHI commands in your system configuration files. However, if you have not run Optimize, and you have programs loaded in High RAM using the DEVICEHIGH or LOADHIGH commands, these programs will be included in the reports produced by LOADHI.COM and Manifest just as if they had been placed there by either of the LOADHI programs.





## QEMM Reports, Analysis and Mode Switching

This chapter describes QEMM.COM, a program that reports on how QEMM is managing your PC's memory. You can also use QEMM.COM to change QEMM's mode (e.g., ON or OFF), and to help you diagnose and correct memory conflicts. Read this chapter if you are interested in finding out about QEMM's memory management on your PC, or if you want to diagnose a possible memory conflict or change QEMM's state.

QEMM includes a program called QEMM.COM which you can use to:

- ❑ Change QEMM's mode (state) to ON, OFF or AUTO.
- ❑ Report status information about QEMM and information about the first megabyte of memory. Some of the information displayed by QEMM.COM is the same as that displayed in Quarterdeck Manifest's QEMM category.
- ❑ Report on what memory your programs have accessed and recommend additional upper memory addresses that you can include for use as High RAM. You can also use QEMM.COM to help resolve memory conflicts that may occur when a program or hardware device needs access to a specific upper memory area that is being used as High RAM. In this case, QEMM.COM will recommend what addresses to exclude from being used as High RAM.

QEMM.COM commands consist of **QEMM** followed by a parameter. Some parameters have an abbreviation you can use instead of the name. The abbreviation is shown in parentheses following the parameter name.

To get an on-screen list of QEMM.COM's parameters:

- Type **QEMM ?** and press **Enter** ↵ to see a list of the names and abbreviations of all QEMM.COM parameters.
- or Type **QEMM HELP** and press **Enter** ↵ to see a list of the parameters with a short description of what they do.

There may be times when you want to change QEMM's current mode. QEMM has the following modes:

- ❑ **AUTO (AU)** - QEMM will turn ON when a program requests expanded memory or any service that puts QEMM into virtual-8086 mode.
- ❑ **ON** - Expanded memory is available and the processor is in virtual-8086 mode. The mode is forced ON if High RAM is created, ROMs are mapped into RAM, conventional or video memory is filled, conventional memory is sorted, expanded memory is in use or Stealth ROM is enabled. During QEMM's installation, the RAM parameter is added to the QEMM386.SYS line of your CONFIG.SYS file; that parameter forces the mode to ON.

### Changing QEMM's Mode

## QEMM.COM Reports

- ❑ **OFF** - Expanded memory is unavailable and the processor is in real mode.

To change the mode, type QEMM followed by the mode. For example to change the mode to ON:

- Type **QEMM ON** and press **Enter** ↵.
- You can similarly change the mode to **OFF** or **AUTO**, provided there are no conditions that have forced the mode ON.

QEMM.COM has five reports which give you information you can use to optimize your memory configuration. To see a report, you type QEMM followed by the report name. These five reports are:

- ❑ **Summary (SUM)** - This report tells you QEMM's current mode, the amount of expanded memory currently available, the page frame address and which Stealth ROM method (if any) is enabled.
- ❑ **Type (T)** - This report tells you how QEMM is managing the first megabyte of memory.
- ❑ **Accessed (ACC)** - This report indicates the areas of memory that have and have not been accessed from the time QEMM started running (or you did a QEMM RESET) until the time of the display.
- ❑ **Analysis (AN)** - This report cross-references the information in the Accessed and Type reports. Based upon what memory has been specified and how QEMM is configured, Analysis suggests command line parameters you can add to the QEMM386.SYS device driver line in CONFIG.SYS to use upper memory more efficiently. You should use these suggestions only after following the Analysis procedure explained later in this chapter.
- ❑ **Memory (MEM)** - This report gives you an accounting of the memory in your PC (conventional, High RAM, extended, and expanded) both before and after QEMM386.SYS has configured this memory.

The Type, Analysis, and Accessed reports have an optional MAP format which gives you a block diagram of memory. To use the MAP format just include the MAP parameter on the command line. For example, to display the Type report as a map, you would type **QEMM TYPE MAP**.



*If you type QEMM without any parameters, you will get the Summary report followed by the Type report.*

## The Summary Report

The Summary report lists basic status information about QEMM. To see a Summary report:

- Type **QEMM SUMMARY** and press **Enter** ↵.

The Summary report displays.

```
C:\>qemm summary
```

```
Current Mode           = ON
Expanded Memory Available = 4640K
Page Frame Address     = C000H
Stealth Type           = M
Stealth ROMs           = F000: 64K
                       C000: 32K
```

Expanded memory is being used.

### The Summary report

- ❑ **Current Mode** is QEMM's mode: ON, OFF, or AUTO.
- ❑ **Expanded Memory Available** is how much EMS memory is available.
- ❑ **Page Frame Address** gives the address of the page frame. This entry indicates "none" if there is no page frame.
- ❑ **Stealth Type** indicates the Stealth ROM method in use, if any. M is the mapping method and F is the frame method.
- ❑ **Stealth ROMs** indicates the starting address and size of any Stealthed ROMs.

## The Type Report

The Type report offers two views of the first megabyte of memory as managed by QEMM. The default view is a list—showing memory areas, their size and use. The second view is a map, showing a block diagram of the first megabyte. To see the Type report:

- Type **QEMM TYPE** or **QEMM TYPE MAP** and press **Enter** ↵.

C:\>qemm type

Area	Size	Status
0000 - 0FFF	64K	Excluded
1000 - 9FFF	576K	Mappable
A000 - AFFF	64K	Video
B000 - B7FF	32K	Excluded
B800 - BFFF	32K	Video
C000 - CFFF	64K	Page Frame
D000 - FFFF	192K	High RAM

The Type report

Key terms used in the Type reports are:

- ❑ **Mappable (+)**: Upper memory addresses that can be filled with memory using EMS 4.0 and EEMS function calls. Mappable areas must be 16K and aligned on 16K boundaries. QEMM386.SYS's RAM parameter automatically converts mappable areas to High RAM.
- ❑ **Rammable (\*)**: Memory areas that QEMM can fill with High RAM but are too small to be accessed by EMS function calls (i.e., less than 16K in size). QEMM can convert rammable areas to High RAM.
- ❑ **Page Frame (F)**: A 64K mappable area that EMS programs use for expanded memory access.
- ❑ **High RAM (H)**: Areas between 640K and 1024K that QEMM has filled with RAM. QEMM will convert Mappable or Rammable areas to High RAM if the RAM parameter is present on the QEMM386.SYS line in CONFIG.SYS.
- ❑ **Mapped ROM (M)**: ROM that has been mapped to RAM by QEMM386.SYS's ROM parameter. Mapping ROMs lets ROM code run in faster RAM. When Stealth ROM is not enabled, QEMM automatically maps a 4K piece of system ROM somewhere between F000 and FFFF to detect warm reboots.
- ❑ **Excluded (X)**: Memory addresses explicitly made unavailable for High RAM, expanded memory mapping, or ROM mapping by QEMM386.SYS's EXCLUDE parameter. If QEMM does not automatically detect an area specifically needed by an adapter or a

program, you must explicitly exclude the area. On 386-based PCs only, QEMM automatically excludes the addresses from 0000 to 0FFF.

- ❑ **Video (V):** Addresses reserved for video display memory.
- ❑ **Adapter RAM (A):** Addresses that have RAM placed into them by adapter cards (e.g., a network adapter).
- ❑ **ROM (R):** ROM areas not remapped by QEMM386.SYS's ROM parameter.
- ❑ **Split ROM (/):** Addresses that have ROM which occupies only a part of the 4K area. These areas cannot be Stealthed or remapped by QEMM386.SYS's ROM parameter.

In the map, each 16-character row represents 64K of memory. Each character represents 4K of memory, and has a special meaning (described above). The left column shows addresses (0n00 through Fn00). To find a particular 4K area, replace the letter *n* with the hex number at the top of the table.

C:\>qemm type map					
	n=0123	4567	89AB	CDEF	
0n00	XXXX	XXXX	XXXX	XXXX	
1n00	++++	++++	++++	++++	
2n00	++++	++++	++++	++++	
3n00	++++	++++	++++	++++	
4n00	++++	++++	++++	++++	
5n00	++++	++++	++++	++++	
6n00	++++	++++	++++	++++	
7n00	++++	++++	++++	++++	
8n00	++++	++++	++++	++++	
9n00	++++	++++	++++	++++	
An00	UUUU	UUUU	UUUU	UUUU	
Bn00	XXXX	XXXX	UUUU	UUUU	
Cn00	FFFF	FFFF	FFFF	FFFF	
Dn00	HHHH	HHHH	HHHH	HHHH	
En00	HHHH	HHHH	HHHH	HHHH	
Fn00	HHHH	HHHH	HHHH	HHHH	

M = Mappable  
 R = Remappable  
 F = Page Frame  
 T = High RAM  
 J = Mapped ROM  
 X = Excluded  
 U = Video  
 A = Adapter RAM  
 R = ROM  
 / = Split ROM

The Type report (map)



## The Accessed Report

The Accessed report offers two views (list and map) of the first megabyte of memory, indicating the 4K regions of memory that have been accessed since QEMM started or since you typed QEMM RESET (see below). You can use this report to see how much memory a program uses and whether it accesses upper memory. This report is valid only if QEMM's mode is ON. To see the Accessed report:

- Type **QEMM ACCESSED** or **QEMM ACCESSED MAP** and press Enter ↵.

C:\>qemm accessed

Area	Size	Status
0000 - 20FF	164K	Accessed
2000 - 2CFF	16K	Unaccessed
2D00 - 2DFF	4K	Accessed
2E00 - 95FF	416K	Unaccessed
9600 - B7FF	136K	Accessed
B800 - BBFF	16K	Unaccessed
BC00 - BFFF	16K	Accessed
C000 - CFFF	64K	Unaccessed
D000 - FFFF	192K	Accessed

The Accessed report

- ❑ **Unaccessed** means an area has not been read or written by a program. Unaccessed areas above 640K can potentially be filled with High RAM.
- ❑ **Accessed** areas have been accessed by a program. In the Accessed report's map format, **Accessed** means the area has been read by a program and **Written** means a program has written to the area.

C:\>qemm accessed map

	n=0123	4567	89AB	CDEF
0n00	UUUU	UUUU	UUUU	UUUU
1n00	UUUU	UUUU	UUUU	UUUU
2n00	UUUU	UUUU	UUUU	UUUU
3n00	UUUU	UUUU	UUUU	UUUU
4n00	UUUU	UUUU	UUUU	UUUU
5n00	UUUU	UUUU	UUUU	UUUU
6n00	UUUU	UUUU	UUUU	UUUU
7n00	UUUU	UUUU	UUUU	UUUU
8n00	UUUU	UUUU	UUUU	UUUU
9n00	UUUU	UUUU	UUUU	UUUU
An00	UUUU	UUUU	UUUU	UUUU
Bn00	UUUU	UUUU	UUUU	UUUU
Cn00	UUUU	UUUU	UUUU	UUUU
Dn00	UUUU	UUUU	UUUU	UUUU
En00	UUUU	UUUU	UUUU	UUUU
Fn00	UUUU	UUUU	UUUU	UUUU

U = Unaccessed  
A = Accessed  
W = Written

The Accessed report (map)

## Resetting Memory

## The Analysis Report



*The Accessed report may be inaccurate for mappable areas. When QEMM maps EMS memory into a mappable area, it clears the area's state, making it appear to be unaccessed. Active use of the page frame for mapping will also clear the "accessed indicators."*

You can reset the state of memory to Unaccessed by typing **QEMM RESET**. You may want to reset memory before running a program to help determine which memory areas the program accesses.

As you run programs, QEMM keeps a record of the memory they access. The Analysis report evaluates which memory is available for High RAM or expanded memory mapping and tells you if there are particular areas you should exclude from QEMM's management (e.g., adapter RAM or ROM that was not detected by Optimize). The Analysis report will also tell you if there are additional areas you can use as High RAM.

The Analysis report is a useful troubleshooting device if you have a particular program that does not run or display properly you install QEMM. The Analysis report can tell you if the program in question is attempting to access an upper memory area that QEMM can fill with High RAM. If this turns out to be the case, you can exclude that area from QEMM's management. You must follow the procedure below for the Analysis report to be accurate.

1. The first step is to determine if QEMM's Stealth ROM feature is enabled and, if so, which Stealth ROM method, F or M is in use. At the DOS prompt, type **QEMM** and press Enter ↵.

A report summarizing QEMM's status will display. If you see the line **Stealth Type = M** or **Stealth Type = F**, Stealth ROM is enabled; remember which letter is listed, M or F.

2. Use a text editor to edit your **CONFIG.SYS** file and type **REM** followed by a space at the beginning of the line that starts as follows:

**DEVICE=C:\QEMM\QEMM386.SYS** (REM causes the line to be ignored).

The line should look something like this:

**REM DEVICE=C:\QEMM\QEMM386.SYS** (parameters not shown)

3. Add a new line directly below the line you just edited. What the line says depends on whether Stealth ROM is enabled on your system.

**If you are using Stealth ROM:** Add the following line, substituting the appropriate Stealth ROM letter, M or F for the x in ST:x.

**DEVICE=C:\QEMM\QEMM386.SYS ON MAPS=0 ST:x FSTC:Y**

**If you are not using Stealth ROM:** Add the following line:

`DEVICE=C:\QEMM\QEMM386.SYS ON MAPS=0`

4. Save your CONFIG.SYS file and reboot your PC.
5. What you do next depends on what kind of analysis you want to do.

**Partial Analysis:** If you are troubleshooting a program that does not display or run properly, or a hardware operation that does not work properly (e.g., a floppy drive), you will want to do a partial analysis, which simply entails using the program or device that has been giving you trouble since you installed QEMM.

To do a partial analysis, just run the programs that have not been working and access their basic features. Or, if you are having trouble with a particular hardware operation, perform that operation. When you are done, skip to step 6.

**Comprehensive Analysis:** If you want to fully analyze your system and programs to see if you can gain additional High RAM, you will want to do a *comprehensive analysis*. This kind of analysis is somewhat time-consuming—you will need to run all your programs and access all hardware devices.

To do a comprehensive analysis, run your programs and use their basic functions (be sure to access any graphics-based programs). If you use DESQview, run its programs and features, then quit DESQview. Do not use any hardware or memory analysis utilities other than QEMM. Access all your PC's hardware and peripherals. If you have two monitors, display information on both of them. Access all disk drives. Format a diskette. Print a document. If you have a network adapter card or terminal emulation adapter, access the network or terminal emulator. When you are done, continue with step 6.



*Be aware that the new QEMM line you added for the Analysis procedure did not include the RAM parameter. That means you no longer have High RAM, so device drivers and TSRs will load low and you will not have as much conventional memory to run programs. We will add the RAM parameter back later.*

6. Display the Analysis report by typing `QEMM ANALYSIS` or `QEMM ANALYSIS MAP` and pressing Enter ↵. If you are unfamiliar with hexadecimal addressing, we suggest using `QEMM ANALYSIS` without MAP—the report will be easier to interpret.

The default view is in list format which explicitly lists addresses you may instruct QEMM386.SYS to include;I or exclude.

C:\>qemm analysis

Area	Size	Status
0000 - AFFF	704K	OK
B000 - B7FF	32K	Exclude
B800 - C7FF	64K	OK
C800 - F7FF	192K	Exclude
F800 - FFFF	32K	OK

C:\>

The Analysis report (list)

This report uses three terms:

**OK (O):** Memory areas QEMM has properly identified.

**Exclude (X):** Areas of memory that have been accessed, but that QEMM expected would remain unaccessed. You should use the QEMM386.SYS's EXCLUDE parameter to prevent QEMM from using these areas.

**Include (I):** Upper memory areas that are reserved by QEMM but have not been used by a program. You may use QEMM386.SYS's INCLUDE parameter to allow QEMM to use these areas.

C:\>qemm analysis

n=0123 4567 89AB CDEF	
0n00 0000 0000 0000 0000	
1n00 0000 0000 0000 0000	
2n00 0000 0000 0000 0000	
3n00 0000 0000 0000 0000	
4n00 0000 0000 0000 0000	
5n00 0000 0000 0000 0000	
6n00 0000 0000 0000 0000	
7n00 0000 0000 0000 0000	
8n00 0000 0000 0000 0000	
9n00 0000 0000 0000 0000	
An00 0000 0000 0000 0000	
Bn00 XXXX XXXX 0000 0000	
Cn00 0000 0000 XXXX XXXX	
Dn00 XXXX XXXX XXXX XXXX	
En00 XXXX XXXX XXXX XXXX	
Fn00 XXXX XXXX 0000 0000	

0 = OK  
X = Exclude  
I = Include

The Analysis report (map)

- Examine the Analysis report and jot down the suggestions. Edit CONFIG.SYS and delete the line you added earlier. It looks like one of the following:

DEVICE=C:\QEMM\QEMM386.SYS ON MAPS=0 ST:x FSTC:Y

or DEVICE=C:\QEMM\QEMM386.SYS ON MAPS=0

## The Memory Report

8. Delete the word REM you added at the beginning of the original QEMM386.SYS line. Then, use the EXCLUDE or INCLUDE parameters to exclude or include any memory addresses specified in the Analysis report. You can abbreviate EXCLUDE as X, and INCLUDE;I as I. For example, if the Analysis report recommended excluding B000-B7FF, you would add EXCLUDE=B000-B7FF to the end of the QEMM386.SYS line. For information on the INCLUDE and EXCLUDE parameters, see Chapter 7.
9. Save the CONFIG.SYS file, reboot your computer and rerun Optimize (see Chapter 3 for information on Optimize).

You can use the Memory report to see how QEMM has configured your PC's memory. The Memory report gives information on conventional memory, High RAM, extended and expanded memory, before and after QEMM has configured memory. For some PCs there will be another category: top memory (on Compaq PCs and Compaq compatibles) or Shadow RAM (on PCs using the Chips & Technologies chip set or other shadow memory supported by QEMM). To see a Memory report:

- Type QEMM MEMORY and press Enter ↵.

```
C:\>qemm memory
```

	Initial	Unavailable to QEMM	Converted by QEMM	Leaving
Conventional:	640K	- 0K	- 0K	= 640K
Extended:	16640K	- 0K	-16640K	= 0K
Expanded:	0K	- 0K	+16288K	=16288K
High RAM:	0K	- 0K	+ 192K	= 192K
<b>TOTAL:</b>	<b>17280K</b>	<b>- 0K</b>	<b>- 160K</b>	<b>=17120K</b>

160K QEMM Overhead			
Code & Data:	119K	Maps:	0K
Tasks:	18K	Mapped ROM:	0K
DMA Buffer:	12K	Unassigned:	11K
4.1K Conventional Memory Overhead			

```
C:\>
```

The Memory report

- The **Initial** column summarizes your PC's memory before QEMM loads. The **Expanded** and **High RAM** categories always start at 0K. The column's total should equal the amount of your PC's memory.
- The **Unavailable to QEMM** column shows any memory QEMM cannot manage. Device drivers that use extended memory and load before QEMM386.SYS make that memory unavailable to QEMM. Memory excluded from QEMM's management by QEMM386.SYS's EXTMEM or MEMORY parameter will also be listed as unavailable.



PCs with shadow memory make some of that memory unavailable to QEMM.

- The **Converted by QEMM** column shows QEMM converting extended memory (and shadow memory or Top memory, if any) into expanded memory and High RAM. QEMM can fill in missing conventional memory, so you may find it creates more conventional memory. This column's total shows how much memory QEMM keeps for its own use and for mapping ROMs.
- The **Leaving** column shows how much memory is left once QEMM initializes. **Conventional** shows the available contiguous memory for DOS programs. **Extended** (usually 0K) shows how much extended memory is left for programs that access extended memory through the obsolete BIOS interface (e.g., VDISK). Programs that can access memory through EMS, XMS, VCPI or DPMI do not use this interface. **Expanded** shows how much memory is available to programs which support EMS, XMS, VCPI, or DPMI. **High RAM** is the amount of upper memory available for loading TSRs and device drivers.
- The lower part of the Memory report details how QEMM itself uses memory. At the bottom of the report is a number showing how little conventional memory QEMM uses.
- The **Unassigned** category of this report represents memory that QEMM controls but which has not been given any function. After QEMM has set up its data, High RAM and other resources, leftover memory is made into expanded memory. Since expanded memory is allocated in 16K units, any memory smaller than 16K, or fragmented memory, is left unassigned. You can alter some of the QEMM386.SYS parameters (e.g., RAM, ROM) to increase or decrease the amount of unassigned memory, either to use it for these new purposes, or to increase it until it becomes more expanded memory.

**If your PC has more memory than the report shows:** Be sure your PC's CMOS is set up properly, and that there are no defective or loose memory chips. Also see the USERAM parameter in Chapter 7.

**Why QEMM may not be able to use all your shadow memory:** QEMM cannot use shadow memory which overlaps ROM or adapter or video RAM.

**Let QEMM allocate extended memory:** We suggest you load device drivers that use extended memory after QEMM386.SYS. Most extended memory programs can access memory through the QEMM's XMS, VCPI or DPMI support. If your extended memory program does not use XMS, VCPI or DPMI (e.g., VDISK), use QEMM386.SYS's EXTMEM parameter (described in Chapter 7) and load the program after QEMM386.SYS.



## QDPMI: QEMM's DPMI Host

This chapter is a technical reference for Quarterdeck's DPMI (DOS Protected Mode Interface). Most users will not need to know anything about QDPMI. This chapter is provided for advanced users who would like technical information about QDPMI.

The Quarterdeck DPMI Host (QDPMI) is a full implementation of Version 0.9 of the DOS Protected Mode Interface (DPMI) specification, including MS-DOS extensions and virtual memory. DPMI is an emerging industry standard for allowing DOS applications to access the protected mode features of 286 and higher processors. The DPMI specification was developed by a consortium of computer hardware and software companies, including Quarterdeck, Borland, Ergo, IBM, Intel, Intelligent Graphics, Locus, Lotus Development, Microsoft, Phar Lap, Phoenix Technologies and Rational Systems.

QDPMI is automatically installed on your system when you install QEMM.

Protected mode multitasking environments, memory managers, or operating systems that implement DPMI functions are called DPMI hosts. Protected mode applications that request DPMI functions (directly or indirectly) are called DPMI clients.

DOS applications using DPMI (DPMI clients) can run in a variety of operating environments, including DOS, DR DOS, Microsoft Windows, OS/2, 386-based UNIX, DESQview 386 and DESQview/X, provided a DPMI host is present.

The Quarterdeck DPMI Host (QDPMI) provides mode-switching services and extended memory management to DPMI clients. It runs at a more privileged level than its clients and uses the hardware to enforce a supervisor/user protection model. This allows QDPMI to support virtualization of memory, maintaining full control over client programs' address spaces.

Quarterdeck's DESQview 386 and DESQview/X can run multiple QDPMI clients simultaneously, each optionally having its own virtual memory.

The major DOS applications that require DPMI support are programmer's tools, including Microsoft's C/C++ Development System for Windows (version 7), Borland's C/C++ (version 3) and Intel's Code Builder Kit (version 1.1). QDPMI is compatible with all of these programs.

Microsoft Windows version 3.1 includes its own DPMI host. When QDPMI detects the startup of Microsoft Windows 386 enhanced mode, it allows Windows to provide the DPMI services for programs running in its environment.

QDPMI is automatically installed when you install QEMM. QEMM must be present for QDPMI's services to be made available to DPMI clients.

By default, QEMM's installation sets up a virtual memory swapfile on your hard disk for times when there is not enough physical memory available for DPMI clients. The swapfile starts out as zero-length and grows as needed up to the default maximum size of one megabyte. You can change the maximum swapfile size or specify that there should be no swapfile. To modify the swapfile option:

- At the DOS prompt, type **QSETUP**.
- Press **Enter** at the Setup program's opening screen to go to the Setup menu.
- At the Setup menu, select **Change DPMI options**.
- Follow the on-screen instructions.

QSETUP makes the necessary changes to your CONFIG.SYS file.

If you change your configuration, we suggest you run QEMM's Optimize program to ensure the best use of your system's memory (for information on Optimize, see Chapter 3).

To see that DPMI services are enabled:

- At the DOS prompt, type **QDPMI**.

When you start an application that requires DPMI services, QDPMI will attempt to create the virtual memory swapfile whose name and maximum size are specified as parameters to the QDPMI device driver line in CONFIG.SYS (see *QDPMI Parameters* later in this chapter), or optionally in the QDPMI environment variable. If the swapfile name is not specified in either of these locations, a swapfile with the base name QDPMI.SWP is opened or created in the directory from which QDPMI.SYS was loaded (normally, \QEMM).

If the swapfile already exists, it will be truncated to zero length, and increased as needed by QDPMI.

If you specify the KILLSWAP option (see *QDPMI Parameters*, later in this chapter), QDPMI will delete the swapfile when the DPMI client terminates.

In DESQview or DESQview/X, each task or DESQview window can have its own QDPMI options in force. For information, see *Using QDPMI with DESQview or DESQview/X* later in this chapter.

Each simultaneously executing DPMI client utilizing virtual memory has its own virtual memory swapfile. The name of the swapfile provided by the QDPMI device driver's SWAPFILE parameter in CONFIG.SYS (or by

### System Interrupt Stack Configuration

### Novell LAN WorkPlace for DOS

### Borland C/C++ 3.1

the QDPMI environment variable) is therefore modified by QDPMI before it is opened or created, by appending a digit to the end of its base name. For example: QDPMI.SWP would become QDPMI0.SWP, and QDPMISWP.SWP would become QDPMISW0.SWP. The actual digit will vary depending on whether DESQview is active or not, and how many other DPMI clients are active in other DOS partitions.

If QDPMI cannot find or create its swapfile, it will issue a message and disable the virtual memory feature for the current client's invocation. Causes for this may be:

- ❑ Insufficient disk space.
- ❑ Improperly specified swapfile name (SWAPFILE parameter to QDPMISYS or environment variable).
- ❑ Non-existent directory specified.

For some applications, it is important that a line exist in CONFIG.SYS setting up system interrupt stack configuration. The suggested values are:

**stacks = 9,256**

This means that DOS should allocate nine system stacks, each 256 bytes in size. The default for this setting is stacks = 9,128 which may not be sufficient for some programs.

QDPMI supports 16- and 32-bit DPMI clients. When a 32-bit client is executing, it expects all 32 bits of the processor's 32-bit registers to be preserved across interrupts. Some device drivers and TSRs utilize the processor's 32-bit registers, but do not preserve the high-order 16 bits, causing 32-bit DPMI clients to behave erratically. A notable example is Novell's LAN WorkPlace for DOS versions 4.0 and 4.01 (TCPIP.EXE). You can prevent the erratic behavior by running the supplied TSR LWPFIX.COM (in the QEMM directory) after loading the LAN WorkPlace TCPIP driver. Novell has fixed this problem in versions 4.10 and later.

Borland's compiler can allocate memory through both EMS and DPMI APIs. QDPMI cannot limit EMS allocation by its clients, so it is possible for QDPMI's memory resources to be depleted without QDPMI's knowledge. To prevent this occurrence, use Borland's compile-time switch -QE=nnnn or -QE-. The former tells Borland's compiler not to exceed nnnn K of EMS memory for use as a swap buffer. The latter tells the compiler to use NO EMS for swapping purposes when compiling. The recommended setting is -QE-, as all necessary swapping will be performed by QDPMI.

## QDPMI Parameters

This section is the reference guide for optional parameters used with both QDPMI.SYS and QDPMI.COM. This chapter also explains the way to use the optional QDPMI environment variable to control QDPMI's behavior.

Both QDPMI.SYS and QDPMI.COM respond to the same set of command line parameters. Command line parameters may be used on the QDPMI device driver line in CONFIG.SYS to initially set QDPMI's options. QDPMI.COM may also be used to modify any or all QDPMI's options from the DOS command line at a later time.

To use a QDPMI.SYS command line parameter, you type the parameter name on the same line as `DEVICE=C:\QEMM\QDPMI.SYS` in your CONFIG.SYS file. To use a QDPMI.COM command line parameter you type QDPMI followed by the parameter name at the DOS prompt. You may use an abbreviation instead of the parameter name. The abbreviation for each QDPMI command line parameter is shown below in parentheses after each parameter name.

### ON

turns DPMI 0.9 services on. This is a default setting.

### OFF

turns DPMI services off.

### MAXMEM nnnnn

is the maximum number of kilobytes of physical memory that will be reserved by QDPMI. This can be any number up to the amount of physical memory available on your system, and will be allocated from VCPI and conventional memory resources, in that order. The memory used by QDPMI includes all memory for use by its code, data, and page tables, as well as memory requested for use by the DPMI client application. If MAXMEM is greater than the available physical memory, MAXMEM will be adjusted downward to be within those limits. Similarly, if you are using DESQview or DESQview/X and MAXMEM is greater than the limits DESQview places on extended memory in a window, MAXMEM will be adjusted downward to be within those limits.

The default value is the total amount of physical memory installed in your system. If omitted or explicitly set to zero, MAXMEM is set to the largest value supported on your system. This will either be the size of all of your physical memory, or that which is available to DESQview's or DESQview/X's current partition, whichever is smaller.

### MINMEM nnnnn

is the amount of physical memory in kilobytes that will be reserved by QDPMI immediately upon DPMI client initialization for use by the



QDPMI kernel itself. QDPMI will initialize only if at least this amount of memory is available for its use.

MINMEM's default value depends on the amount of physical memory installed in your computer, and whether you are using QDPMI's Virtual Memory features. Any attempt to set MINMEM to a value lower than QDPMI's calculated minimum will result in QDPMI ignoring the request, and using its own calculated MINMEM value.

MAXMEM must be greater than or equal to MINMEM, regardless of whether it was user specified or internally calculated. If this is not the case, QDPMI will exit with a message indicating that insufficient memory is available for initialization.

The difference between MAXMEM and MINMEM is the amount of memory that will be managed dynamically by QDPMI, and made available to other applications (including other DPMI clients) when it is free.

#### **MAXLOW nnnnn (LOW nnnnn)**

is the maximum number of kilobytes of conventional memory allowed by QDPMI for use as its memory resources. Setting it to any non-zero value will allow QDPMI to allocate conventional memory for use in its memory pool. QDPMI will use DOS conventional memory only if expanded and extended memory resources are insufficient to satisfy the MINMEM request. The default value is 0. Conventional memory is only used if MINMEM cannot be satisfied from other resources, and only if this value is not zero.

#### **VMON (VM)**

turns QDPMI virtual memory on. This is the default setting.

#### **VMOFF (NOVM)**

turns QDPMI virtual memory off.

#### **KILLSWAP (KLSW)**

causes the QDPMI host to delete its virtual memory swapfile upon the termination of the DPMI client. This is the default setting.

#### **KEEPSWAP (KPSW)**

prevents the QDPMI host from deleting its virtual memory swapfile upon termination of the DPMI client. This means that the same swapfile will be preserved on disk for future DPMI clients. This may take more disk space than KILLSWAP, but will save time.

**SWAPFILE name (SWAP name)**

is the base name of the virtual memory swapfile that QDPMI will use.

**SWAPSIZE nnnnn (SWSZ nnnnn)**

is the virtual memory swapfile's size in kilobytes. The size of the virtual memory swapfile that you specify may be any value up to 64 megabytes, and defaults to 1MB.

**EXTCHKON (XCHK)**

causes the QDPMI host to step out of the way for DOS extenders provided by Phar Lap, Rational Systems, and Ergo; these DOS extenders prefer to manage DPMI in their own way. This is the default setting.

**EXTCHOFF (NOXCHK)**

causes the QDPMI host not to step out of the way for DOS extenders provided by Phar Lap, Rational Systems, and Ergo. These DOS extenders may not functional properly if QDPMI is providing DPMI services.

**OVLDIR=path**

specifies where QDPMI will look for its QDPMIVM.OVL file during DPMI client initialization. This defaults to the same directory from which QDPMI.SYS was loaded at system initialization. Under some network implementations such as diskless workstations, this initial path may cease to exist after the system is initialized. Use this switch tell QDPMI where to look for QDPMIVM.OVL at DPMI client initialization in such cases.

You can optionally use an environment variable to determine several aspects of QDPMI's configuration. Its syntax is as follows:

```
SET QDPMI=[MAXLOW nnnnn] [MINMEM nnnnn]
          [MAXMEM nnnnn] [SWAPFILE name] [SWAPSIZE nnnnn]
```

All of the options set in the environment variable are normally set by the parameters to the QDPMI device driver line (see above). If you choose to use the environment variable, its settings will override the corresponding settings in the QDPMI device driver line. You cannot use parameter abbreviations with the QDPMI environment variable.

If you are using DESQview or DESQview /X, you may want to have a different set of DPMI options in force for different windows. There are two ways to do this:

- ❑ Set a QDPMI environment variable for each window you want to affect, using the desired options (see the previous section). You can do

The QDPMI  
Environment  
Variable

Using QDPMI  
with DESQview  
or DESQview/X

## Virtual Memory and Total Memory Size

## Error Messages

this by typing the SET QDPMI= command at a window's DOS prompt or in a script.

- ❑ Run the QDPMI.COM program with the desired parameters in each window you want to affect. You can do this by typing the QDPMI command at a window's DOS prompt or in a script. The only parameters that are limited in scope to a particular window are ON, OFF, VMON, VMOFF, KILLSWAP, KEEPSWAP, EXTCHKON, and EXTCHOFF. If you use any of the other QDPMI parameters, they will affect all windows. (For information on the parameters, see the section *QDPMI Parameters*, earlier in this chapter).

The sum of the value of MAXMEM and the maximum size of the virtual swapfile is the amount of virtual memory (in kilobytes) available to QDPMI client applications when the virtual memory option is enabled.

QDPMI may display error messages when trying to load itself or when trying to initialize a DPMI client. Error messages may also be generated by the QDPMI kernel or the DOS extender.

- ❑ **QDPMI: Client already active — Cannot modify parameters.**

A DPMI client has spawned COMMAND.COM, and is still active in the current DOS partition. It is inappropriate to change QDPMI parameters while a QDPMI client is currently running. This condition can sometimes be caused by the abnormal termination of a DPMI client.

- ❑ **QDPMI: Already supported — Not reloaded.**

A DPMI Host other than QDPMI has been detected. Quarterdeck's QDPMI is not loaded.

- ❑ **QDPMI: Version incompatibility error — Re-install or use same version.**

This message only appears if the device driver QDPMI.SYS and the utility program QDPMI.COM are not the same version. This is an error in installation or updating. You must reinstall QDPMI.

- ❑ **QDPMI: Cannot locate QEMM386.SYS Memory Manager — Not loaded.**

QDPMI Host will only function in conjunction with QEMM.

- ❑ **QDPMI: Error accessing XDI — Not loaded.**

QDPMI has encountered an error initializing the DESQview XDI interface for external devices and cannot load.

- ❑ **QDPMI: Too Many DPMI Processes — Not loaded.**

QDPMI can support up to fifteen concurrent processes under DESQview and DESQview/X. If a DPMI client is started in the 16th or higher window, the DPMI client will fail to initialize, and QDPMI.COM will display this message if run in this window.

- ❑ **QDPMI: Overlay File Path/Name is too long.**

The QDPMIVM.OVL pathname is too long. Use a path less removed from the root.

- ❑ **QDPMI: Quarterdeck QDPMI Host is not resident.**

QDPMI.COM issues this message if run without having previously loaded QDPMI.SYS in CONFIG.SYS.

QDPMI client initialization error messages are as follows:

- ❑ **QDPMI: Open Error.**

QDPMI could not open its overlay file QDPMIVM.OVL. Be sure that it is either in the same directory as QDPMI.SYS, or in the directory specified by the OVLDIR parameter on the QDPMI command line.

- ❑ **QDPMI: Processor Error.**

QDPMI must have an 80386 or higher processor in order to run.

- ❑ **QDPMI: Read Error.**

QDPMI cannot read its overlay file QDPMIVM.OVL.

- ❑ **QDPMI: Seek Error.**

QDPMI cannot seek in its overlay file QDPMIVM.OVL.

- ❑ **QDPMI: Close Error.**

QDPMI can not close its overlay file QDPMIVM.OVL.

- ❑ **QDPMI: Bad .OVL Error.**

QDPMIVM.OVL file is corrupted.

- ❑ **QDPMI: Exec Error.**

QDPMI could not transfer control to its overlay.

- ❑ **QDPMI: Init Error.**

QDPMIVM.OVL could not complete its initialization.

- ❑ **QDPMI: Mode Error.**

The QDPMI client is attempting to re-initialize the QDPMI Host in a mode (16/32 bit) other than the one in which it was originally

initialized. This can occur if an attempt is made to start a DPMI client of another "bitness" than the currently running DPMI client.

❑ **QDPMI: Task Error.**

The maximum of fifteen simultaneous DPMI clients are already running in other DESQview or DESQview /X windows.

❑ **QDPMI: Version Error.**

There is a discrepancy between the QDPMI version of the device driver (QDPMI.SYS), and the kernel (QDPMIVM.OVL).





## EMS Utility Programs

### The EMS Programs

This chapter describes three advanced EMS utility programs that can give you more control over EMS memory. This chapter is provided for programmers and advanced users who want to control EMS memory allocation.

QEMM includes three advanced utility programs: EMS.COM and EMS.SYS, which give you behind-the-scenes access to QEMM's management of EMS memory; and EMS2EXT, which lets you allocate extended memory on the fly through the pre-XMS INT 15 interface. You may never need these programs, but they can prove useful in cases where you need more control of memory allocation than your programs give you.

The EMS.COM and EMS.SYS programs provide several informative and powerful functions to help you make the best use of your EMS memory in cases in which you have special or unusual requirements. Although anyone may benefit from seeing the EMS status report and the details of expanded memory allocation, other uses of EMS which will be described in these sections are for technically sophisticated users.

Most of the functions of EMS.SYS and EMS.COM involve the manipulation of expanded memory handles. An *EMS handle* is the information that the expanded memory manager uses to identify a block of memory that it allocates. A handle is represented by a number and may optionally have a name.

An expanded memory handle is the token of interaction between an EMS-using program and an expanded memory manager. EMS.SYS and EMS.COM give you command-line control of some of the EMS functions that are usually available only at the programming level. Since these EMS utilities are capable of granting you access to handles which may belong to other programs, you should exercise caution when using these utilities.

With the EMS programs, you can allocate and name a block of memory with the CREATE option, and optionally specify that this block of memory consist of the fastest or slowest memory on your system. You can use the FREE option to free the expanded memory associated with a handle. You can read data from a file into expanded memory or write the data from expanded memory to a file with the LOAD or SAVE options. You can rename an EMS handle and change the amount of memory associated with it.

The most common reason for using the EMS programs is to prevent a specific application from using all of the memory in your system. By issuing an EMS CREATE command before running an application, you effectively "hide" the specified amount of memory from that application. Many programs (e.g., Microsoft Windows, AutoCAD, Quattro, Lotus 1-2-3 version 3) allocate a great deal of available memory to themselves at

startup time—sometimes as much as you have on your system. By creating an EMS handle in the following fashion:

**EMS CREATE handle\_name 2048K**

you reserve 2 megabytes of memory, identified by the name `HANDLE_NAME`, that other programs will see as already assigned, and therefore will not touch. Once your program has started, you could go to the DOS prompt and issue the command:

**EMS FREE handle\_name**

to release the 2 megabytes of memory, which would leave 2 megabytes available after your application is running. Because QEMM gives out both expanded and extended memory from the same memory pool, you can use this method to withhold memory from programs that allocate their memory through EMS, XMS, VCPI, or DPML. This method is particularly useful for preventing Microsoft Windows 3.1 standard mode from allocating all memory, so that you can run programs that get their memory through EMS, VCPI, or DPML inside Windows.

If parts of the expanded memory in your system run at different speeds, you can use EMS to allocate memory of one speed before you load a device driver or TSR so that it can only use the faster or slower memory that remains; then you can free the memory for use by your other applications. Manifest can show you if your memory runs at different speeds.

If you are a programmer using expanded memory, you can use the `LOAD` and `SAVE` functions when you need to save and restore the contents of expanded memory during development and debugging.

You use `EMS.SYS` in the `CONFIG.SYS` file to manipulate expanded memory during the system boot sequence. You use `EMS.COM` in the `AUTOEXEC.BAT` file or directly from the DOS prompt, as needed.

To get a summary report of your expanded memory:

- **At the DOS prompt, type `EMS` and press `Enter`.**

EMS will report the total amount of expanded memory, the amount currently available and the address of the page frame.

Both `EMS.SYS` and `EMS.COM` respond to the same parameters. The parameters are described below. Some parameters have an abbreviation you can use instead of the full name; abbreviations are listed in parentheses below. Brackets ([ ]) in a statement indicate that the enclosed item is optional.

## EMS Parameters

## **DIR**

displays a breakdown of the current expanded memory allocated. For each allocated handle, DIR gives the number of expanded memory pages associated with it, the number of kilobytes of memory those pages represent, and the name assigned to that handle, if any.

## **CREATE name nnnnnn[K] (CR)**

allocates expanded memory. CREATE requires two arguments: a name for the block of memory you are allocating and the amount of memory. The name may be one to eight characters long. The name need not be enclosed in quotation marks unless it contains blanks. The amount of memory you are allocating may be expressed in EMS pages (16K per page) or in kilobytes. If you specify the number of kilobytes, the memory manager will round the number up if necessary to a multiple of 16. To specify kilobytes, you must use the letter K immediately following the amount of memory. If you do not follow the number with a K, the number will represent EMS pages. Follow the EMS CREATE command with EMS DIR command to confirm the allocation and to determine the handle number assigned to the name.

## **CREATEFAST name nnnnnn[K] (CFAST) and CREATESLOW name nnnnnn[K] (CSLOW)**

are alternate forms of the CREATE option that instruct the memory manager to allocate the memory from either faster or slower memory. Use Manifest's Expanded Memory Timings to determine if any speed difference does in fact exist.

## **FREE name or number**

frees memory and deallocates a handle. FREE requires that you specify a handle to deallocate, either by its name or number. Beware of doing this to someone else's handle.

## **RENAME name or number new\_name (REN)**

lets you assign a new name to a handle. The first parameter to RENAME is the original handle. You may refer to this handle by its number or its name. The second argument is the new handle name. RENAME can be useful to name an unnamed handle to help you keep track of it.

## **RESIZE name or number nnnnnn[K] (RES)**

lets you increase or decrease the amount of memory assigned to a handle. Its two arguments are the same as those of CREATE.

**SAVE name or number filename**

allows you to save the contents of the expanded memory pages associated with an EMS handle to a file. This option requires that you specify the handle name (or number) and the filename.

**LOAD name or number filename**

allows you to restore the contents of expanded memory pages that have been stored in a file. This option requires that you specify the handle name (or number) and the name of the file containing the data you want to restore. The number of pages required will be automatically allocated based on the file's size.

**HELP**

displays help on the EMS programs and their options.

**?**

lists the EMS programs' parameters.

**EMS2EXT.SYS**

EMS2EXT.SYS converts expanded memory to extended memory, for programs that rely upon the old INT 15 method of accessing extended memory. This method is no longer widely used, and has been replaced by XMS (the Extended Memory Specification). Older versions of DOS shipped with utilities which relied upon the old INT 15 interface, most notably VDISK.SYS. These drivers have since been replaced by programs that use XMS instead, and as a result EMS2EXT is rarely useful.

EMS2EXT is not needed for programs that access memory through XMS, VCPI, or DPMI. It is intended only to provide on-the-fly control over extended memory allocated through the older INT 15 interface. Programs which support XMS, VCPI or DPMI can allocate and deallocate memory directly from QEMM's memory pool and have no need for EMS2EXT.

Even if you have an old extended memory utility, you cannot use EMS2EXT if your program expects to access extended memory directly at physical addresses above 1024K. Quarterdeck's QEXT.SYS driver, supplied with DESQview, cannot use memory supplied by EMS2EXT. Likewise, Microsoft's HIMEM.SYS cannot use memory supplied by EMS2EXT.

If you do have an old extended memory program that uses the INT 15 interface, EMS2EXT lets you allocate memory for that program out of QEMM's memory pool. The advantage of allocating this memory with EMS2EXT instead of with QEMM parameters is that the memory allocation can later be increased or decreased with the EMS.COM program without rebooting your system.

EMS2EXT is a device driver and therefore needs to be loaded with a **DEVICE=** statement in your CONFIG.SYS file. The statement to load EMS2EXT should look like this:

**DEVICE=C:\QEMM\EMS2EXT.SYS MEMORY=nnn speed**

The **nnn** parameter in **MEMORY=nnn** is the number of kilobytes of expanded memory to allocate initially (e.g., **MEMORY=512**). EMS2EXT will allocate an EMS handle named EMS2EXT for a block of memory **nnnK** in size. You can also load EMS2EXT without specifying any **MEMORY** parameter. EMS2EXT will be resident, but it will not allocate any memory. It will, however, reserve for itself a handle with the name EMS2EXT.

The optional **SPEED** parameter tells EMS2EXT to allocate faster or slower memory if there are different speeds of memory on your system. You may specify **FAST**, **SLOW** or no **SPEED** option at all.

You can, as needed, grow, or shrink the amount of extended memory for the EMS2EXT handle using EMS.COM. You can use this capability to give a program INT 15 extended memory only while it is running. For instance, if you loaded EMS2EXT with no **MEMORY** parameter, you could make a batch file which included the line:

**EMS RESIZE EMS2EXT 128K**

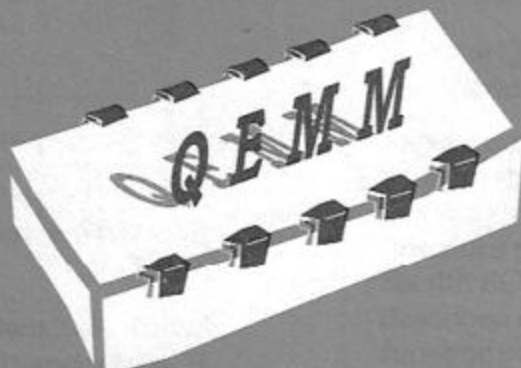
before running an application that needs 128K of extended memory through the old INT 15 interface. When the program terminates, another EMS statement could free the memory:

**EMS RESIZE EMS2EXT 0**

The memory is then returned to QEMM's memory pool for the use of other programs.



# Appendices



**A**

## Troubleshooting

This section contains tips on handling some common problems users may experience after installing QEMM. If your question is not answered in this section, you will find a list of "README" files that contain further troubleshooting tips in Appendix B.

Some of the troubleshooting instructions will ask you to edit your CONFIG.SYS file. If you edit that file be sure to use a text editor (e.g., EDIT supplied with DOS 5 and 6). If you use a word processor, be sure it is in ASCII mode (see the word processor's manual for details). Alternatively, you can use the QSETUP program provided with QEMM. Just type QSETUP and press Enter ↵, press Enter ↵ again to go to the QSETUP menu, then select **Edit the proposed CONFIG.SYS**.

### Your PC does not function properly after installing QEMM.

1. Follow the steps below to boot your PC without QEMM.
  - **Reset your system.** Use the power switch if necessary.
  - **Wait until you hear a beep, then hold down the Alt key until the boot sequence stops.** If you are using QEMM's DOS-Up feature, you will see a message asking if you want to unload it; press Esc to unload DOS-Up, then hold down Alt again. You will see the following message: "QEMM: Press ESC to unload QEMM or any other key to continue with QEMM." If your system does not beep on bootup, hold down Alt for three or four seconds after you reset the system.
  - **Press the Esc key.**

Your system will then proceed with the boot sequence. QEMM will not be loaded and no programs will be loaded into High RAM. If your problem still occurs, it is not related to QEMM386.SYS; try removing other QEMM programs or investigating other software and hardware. If the problem no longer occurs, you can now modify the QEMM386.SYS driver line in your CONFIG.SYS file (as explained below) to investigate the problem.

2. You may have a Stealth-related problem. Edit your CONFIG.SYS file and see if one of the Stealth ROM parameters (ST:F or ST:M)

is on the QEMM386.SYS line; if there is no such parameter, skip to step 3 now.

Make a note of which parameter (ST:F or ST:M) is there, then delete that parameter. Save CONFIG.SYS and reboot. If your PC functions properly, the problem is related to Stealth ROM; run the Analysis procedure described on *page 97*. If your PC does not function properly, continue with step 3, rebooting without QEMM (see step 1 above) if you cannot edit CONFIG.SYS otherwise.

3. Edit CONFIG.SYS, and if there are any INCLUDE (I) or ROM parameters on the QEMM386.SYS line, remove them and reboot. If your system functions properly now, you may want to try to narrow down which parameter or region caused the problem by replacing the parameters one by one.
4. There may be a conflict between QEMM and one of your PC's devices (e.g., a network adapter) or features. If you have the RAM parameter on the QEMM386.SYS line in CONFIG.SYS, remove the RAM parameter, save CONFIG.SYS then reboot. If you do not have the RAM parameter, skip to step 5.

If your PC now functions properly, a device may have been using some area between 640K and 1024K in such a way that QEMM could not detect it. You should exclude that area from QEMM's management. To do that, run the Analysis procedure and add the necessary EXCLUDE parameters to the QEMM386.SYS line (see *page 97*). After following the recommendations of the

Analysis report and running Optimize, see if the problem is corrected.

5. If you still have not determined the conflict, then you should create a "pure environment" to isolate the source of the trouble. To create a pure environment, you need to temporarily remove all lines that are not absolutely necessary from your AUTOEXEC.BAT and CONFIG.SYS files.
  - a. First, make backup copies of these files (just in case you need them later) by copying them to CONFIG.XXX and AUTOEXEC.XXX.
  - b. Next, edit CONFIG.SYS and AUTOEXEC.BAT with a text editor (e.g., QSETUP's editor, DOS EDIT or EDLIN) and "comment out" all lines except those that are absolutely necessary. To comment out a line, add the word REM at the beginning of the line (REM causes the line to be ignored when the file is processed). When you edit CONFIG.SYS, do *not* comment out the QEMM386.SYS device driver line, the FILES line, or any drivers that are absolutely necessary (e.g., a hard disk manager). Also, you need not comment out the PROMPT line in AUTOEXEC.BAT. If you are troubleshooting problems with a specific device, do not comment out the lines related to that device (e.g., if your network is not functioning properly, do not comment out the network drivers or any lines relating to the network). Once you have edited each file, save it.
  - c. Reboot your PC and test to see if the problem still occurs. If it does, you may have a hardware problem or your software may be incompatible with your hardware. You may need to have your hardware checked out. If you think the hardware is OK, you may want to contact the software publisher to see if they are aware of a solution. If you do not suspect a hardware or software problem, skip to step 6.

If the problem does not occur, one or more of the lines you commented out may have been causing the problem. Examine CONFIG.SYS and AUTOEXEC.BAT files to see if you can determine which commented line may be at fault.

- d. Remove the word REM a line at a time, reboot and test to see if the problem still occurs. When you are able to recreate the problem, you have isolated the offending line. You have a few alternatives here: 1) contact the publisher of the driver or TSR and see if they know of a solution, 2) remove the driver or TSR from your CONFIG.SYS or AUTOEXEC.BAT file, or 3) contact Quarterdeck Technical Support for further assistance. If you are able to fix the problem and made the necessary changes to your CONFIG.SYS file and AUTOEXEC.BAT files, run Optimize (see **Chapter 3**).

If you have not solved the problem, continue with step 6.

6. If you are NOT using the RAM parameter (or you were using it and removing it did not help) and the computer will still not boot properly, there is some other conflict. Try adding the following group of parameters (listed here in their abbreviated form) to the QEMM386.SYS device driver line in your CONFIG.SYS file: DB=2, CF:N, FILL:N, MR:N, RH:N, SH:NONE, TR:N, TM:M, XBDA:N. For information on these parameters, see **Chapter 7**.

If adding the above parameters fixes the problem, remove them one at a time and reboot until the problem returns. At that point you have found at least one parameter that helps fix the problem, so add it back. Repeat this process for the remaining parameters. You may want to read about these parameters to see what they do and to find out the ramifications of using them (see **Chapter 7**).

7. If you still have not solved the problem, try reinstalling QEMM. If that does not help, contact Quarterdeck Technical Support.

Optimize does not complete successfully or your system does not initialize properly after running Optimize.

QEMM may be using certain upper memory addresses needed by your system or by a program. In this case, you can rerun Optimize and have it attempt to uncover RAM areas that should not be used.

- Switch to the QEMM directory by typing **CD \QEMM** and pressing **Enter** ↵.
- Type **OPTIMIZE /AUTO** and press **Enter** ↵.

This will begin the Optimize / AUTOEXCLUDE detection. You will see a message asking you to power off your PC, then turn it back on. Do not do a warm reboot or use your PC's Reset button.

- Power off your PC, wait five seconds, then turn it back on.

Optimize may reboot your system again. When your PC finishes booting, you will see Optimize's Exclusion Processing screen, listing in hexadecimal notation the upper memory addresses Optimize thinks should be excluded from use as High RAM.

If there are several addresses listed, you may not need to exclude them all. If you feel comfortable doing so, we suggest you view the exclusions and try to determine which ones are necessary. If you want to accept Optimize's recommendations, just press **Enter** ↵ to continue. Otherwise, to view the exclusions:

- Press **F1**.

You will see a list of programs and drivers. Some of them are followed by addresses—these are the upper memory addresses that the program or driver accesses. If there are multiple address ranges, you may not need to exclude them all. We suggest you first try excluding only those that are related to programs or drivers associated with an adapter card (e.g., a network

card). You can always run **OPTIMIZE /AUTO** later if you need to exclude more. To remove an item from the exclusion list:

- Use the arrow keys to highlight the **Y** in the item's first column and press the **spacebar** to change it to **N**.

If there is a down arrow at the bottom of the list, there are more items below. You can press **Page Down** to see more items.

- When you are finished editing the exclusion list, press **Esc**.

You are back at the Exclusion Processing screen. To continue optimizing:

- Press **Enter** ↵ and follow the on-screen instructions.

Optimize will add **EXCLUDE (X)** parameters to the **QEMM386.SYS** device line in your **CONFIG.SYS**. For information on the **EXCLUDE** parameter, see **Chapter 7**.

If there are still problems after running **OPTIMIZE /AUTO**, you may want to run it again and exclude any addresses you may have removed from the exclusion list. If the Optimize procedure still does not complete, perform the steps in the first troubleshooting tip in this section.

When trying to load a TSR or device driver high, you see the message, "Not enough room to load high."

Try rerunning Optimize (see **Chapter 3**).

After optimizing, a TSR or device driver does not load high.

Many TSRs and device drivers require more memory to initialize than they do once they are resident. For example, a 10K mouse driver may require 22K to initialize. So, if the largest single High RAM region were 20K, QEMM may not be able to load the driver high. We suggest you find out how much memory the TSR or device driver that did not load high needs to initialize. Run Optimize and use the "Modify Data" option



(available when you see the "Analysis Complete" screen). The number of K that your program needs to initialize is listed next to your program's name in the Initial Size column.

Once you find out how much memory is needed, escape out of the "Modify Data" screen and use the "Region Layout" option to see the sizes of your system's available High RAM areas. What you do next depends on whether any of the High RAM areas are large enough to load your TSR or device driver.

**If you have a High RAM area large enough to load the item high:** You may be able to create enough room to load the item high by changing the load order of device drivers or TSRs. Run Optimize and use the "What if" option; try to load the bigger items first (see Chapter 3).

**If you do not have a High RAM area large enough to load the item high:** You may have an adapter that uses upper memory addresses and the adapter RAM or ROM is splitting up a large upper memory region. Use QEMM.COM's Type Map report to find out (see Chapter 9). If this is the case, you may be able to move the adapter's RAM or ROM to another upper memory area, thereby making enough contiguous space to load the TSR or device driver high. See the adapter's manual for information on using an alternative address for the adapter's RAM or ROM. Also, if QEMM's Stealth ROM feature is not enabled, try enabling it to get more High RAM (see Chapter 5).

One of your TSRs or device drivers does not work well when loaded high, but Optimize still tries to load it high when you re-optimize your system.

You can create a file called OPTIMIZE.EXC in your QEMM directory that will prevent some of your programs from being loaded high by Optimize. See Chapter 3's section *Excluding TSRs and Device Drivers from Optimize* and the discussion of Optimize's /COMMANDFILE parameter for details on how to create the OPTIMIZE.EXC file.

### Solving a memory conflict with the Analysis procedure.

An Analysis procedure determines what upper memory addresses are accessed by programs and hardware devices. It will recommend addresses that can or should not be used as High RAM or for EMS mapping. You may want to run an Analysis if:

- ❑ After installing QEMM, a program functions or displays improperly or crashes the system.
- ❑ A hardware device (e.g., floppy disk, CD ROM drive) no longer works properly.
- ❑ You want to see if it is possible to create more High RAM to load device drivers or TSRs that QEMM was unable to load high.

1. The first step is to determine if QEMM's Stealth ROM feature is enabled and, if so, which Stealth ROM method, F or M is in use. At the DOS prompt, type QEMM and press Enter ↵.

A report summarizing QEMM's status will display. If you see the line Stealth ROM Type = M or Stealth ROM Type = F, Stealth ROM is enabled; remember which letter is listed, M or F.

2. Use a text editor to edit your CONFIG.SYS file and type REM followed by a space at the beginning of the line that starts as follows:

**DEVICE=C:\QEMM\QEMM386.SYS**  
(REM causes the line to be ignored).

The line should look something like this:

**REM DEVICE=C:\QEMM\QEMM386.SYS**

3. Add a new line directly below the line you just edited. What the line says depends on whether you have Stealth ROM enabled.

**If you are using Stealth ROM:** Add the following line, substituting the appropriate Stealth letter, M or F for the x in ST:x.

**DEVICE=C:\QEMM\QEMM386.SYS ON  
MAPS=0 ST:x FSTC:Y**

Be sure to type the above text on a single line.



**If you are not using Stealth ROM:** Add the following on a single line:

```
DEVICE=C:\QEMM\QEMM386.SYS ON  
MAPS=0
```

4. Save your CONFIG.SYS file and reboot your PC.
5. What you do next depends on what kind of analysis you want to do.


**Partial Analysis:** If you are troubleshooting a program that does not display or run properly, or a hardware operation that does not work properly (e.g., a floppy drive), you will want to do a partial analysis, which simply entails using the program or device that is giving you trouble since you installed QEMM.

To do a partial analysis, just run the programs that have not been working and access their basic features. Or, if you are having trouble with a particular hardware operation, perform that operation. Things should work properly now; if there are still problems, restore the original QEMM386.SYS line in CONFIG.SYS and try some of the procedures explained on page 119. When you are done, skip to step 6.

**Comprehensive Analysis:** If you want to fully analyze your system and programs to see if you can gain additional High RAM, you will want to do a comprehensive analysis. This kind of analysis is somewhat time-consuming—you will need to run all your programs and access all hardware devices.

To do a comprehensive analysis, run your programs and use their basic functions (be sure to access any graphics-based programs). If you use DESQview, run its programs and features, then quit. Do not use any hardware or memory analysis utilities other than QEMM. Access all your PC's hardware and peripherals. If you have two monitors, display information on both of them. Access all disk drives. Format a diskette. Print a document. If you have a

network adapter card or terminal emulation adapter, access the network or terminal emulator. When you are done, continue with step 6.

 *Be aware that the new QEMM line you added for the Analysis procedure did not include the RAM parameter. That means you no longer have High RAM, so device drivers and TSRs will load low and you will not have as much conventional memory to run programs. We will add the RAM parameter back later.*

6. Display the Analysis report by typing **QEMM ANALYSIS** or **QEMM ANALYSIS MAP**. If you are unfamiliar with hexadecimal addressing, we suggest using **QEMM ANALYSIS** without MAP—the report will be easier to interpret.

The default view is in list format which explicitly lists addresses you may instruct QEMM386.SYS to include or exclude.

This report uses three terms:

**OK (O):** Memory areas QEMM has properly identified.

**Exclude (X):** Areas of memory that have been accessed, but that QEMM expected would remain unaccessed. You should use the QEMM386.SYS's EXCLUDE parameter to prevent QEMM from using these areas.

**Include (I):** Upper memory areas that are reserved by QEMM but have not been used by a program. You may use QEMM386.SYS's INCLUDE parameter to allow QEMM to use these areas.

7. Examine the Analysis report and jot down the suggestions. Edit CONFIG.SYS and delete the line you added earlier. It looks like one of the following:

```
DEVICE=C:\QEMM\QEMM386.SYS ON  
MAPS=0 ST:x FSTC:Y
```

```
or DEVICE=C:\QEMM\QEMM386.SYS ON  
MAPS=0
```

8. Delete the word REM you added at the beginning of the original QEMM386.SYS line. Then, use the EXCLUDE or INCLUDE parameters to exclude or include any memory addresses specified in the Analysis report. You can abbreviate EXCLUDE as X, and INCLUDE as I. For example, if the Analysis report recommended excluding B000-B7FF, you would add EXCLUDE=B000-B7FF to the end of the QEMM386.SYS line. For information on the INCLUDE and EXCLUDE parameters, see Chapter 7.
9. Save the CONFIG.SYS file, reboot your computer, and rerun Optimize (see Chapter 3 for information on Optimize).

#### If you see an Exception 13 message...

An Exception 13 message is an error message from your PC's main processor, giving information about an invalid instruction. The invalid instruction may have come from a bug in a program, or may result from a conflict between two programs or between a program and a hardware device.

Exception 13 is not a QEMM error, but rather an error from your PC's main processor that QEMM passes on to you as a courtesy, so you will know what happened and can take steps to deal with it. Without QEMM, your system may have crashed or hung without warning.

A program that uses protected mode but does not support VCPI (Virtual Control Program Interface) or DPMI (DOS Protected Mode Interface) cannot run under QEMM. We suggest you contact the developer to see whether they provide support for these specifications.

Here is a tip for solving the problem that caused the Exception 13 error: Look at the Exception 13 message and write down the address directly following the words "Exception 13" (e.g., C801:012F). This is the memory address of the code being executed when the exception occurred. (The other numbers are the contents of the CPU's registers at the time, and fifteen bytes of the code at the exception location.)

- ❑ If the address you wrote down starts with a letter from A to F, the address is in upper memory. If so, run Manifest and look at the First Meg Programs display to see if any program's Memory Area contains the first four digits of the address you wrote down. If so, try loading the program into conventional memory instead. To do that, delete the LOADHI syntax from the line in CONFIG.SYS or AUTOEXEC.BAT that loads the program (see Chapter 8 for information on LOADHI).
- ❑ If the address you wrote down begins with a number or if you have not solved the problem, follow the troubleshooting procedure on page 119 under the heading **Your PC does not function properly after installing QEMM.**

Also see the text file EX13FLOW.TEC in the \QEMM\TECHNOTE directory.

An Exception 13 inside of DESQview can often be fixed by using the Change a Program feature to allocate more memory to the program. If you are using DESQview/X, you can use DVP Manager to change the Maximum Program Memory Size. If that does not solve the problem, setting Protection Level to 3 will provide more information about what is causing the exception. Once you solve the problem, return Protection Level to 0.

**The network does not load or work properly after you power on your system.**

Your network adapter may be using an upper memory area that QEMM has filled with High RAM. If so, you will need to exclude the network adapter's upper memory addresses on the QEMM386.SYS line of CONFIG.SYS. There are two ways to find out what addresses to exclude:

- ❑ The network adapter's memory address range may display when the network driver initializes. Also, you can look in the network adapter's documentation to see if the addresses are listed or if there is a command

you can type to display the addresses. Once you find out the adapter's upper memory addresses, edit CONFIG.SYS and use the EXCLUDE parameter to exclude those addresses on the QEMM386.SYS line (see the EXCLUDE parameter in Chapter 7).

- If you cannot find the network adapter's memory addresses as explained above, use the method described in step 4 on page 119.

Another possibility is that you have a bus-mastering network card (see the next troubleshooting problem).

A less likely possibility is that your network requires the QEMM parameter LOCKDMA:Y to prevent occasional crashes. See Chapter 7 for more information on LOCKDMA:Y.

**Your system locks up or gives you an "exception" error message during one of Optimize's reboots or when a device is loading high.**

This problem may be related to a bus-mastering device (e.g., certain SCSI drive controllers or network adapters). Bus-mastering devices handle their own DMA (direct memory access) functions, bypassing the DMA controller on the PC's motherboard. These devices may not recognize QEMM's memory mapping.

The best solution is to contact the manufacturer of the device, who may have a VDS (Virtual DMA Services) device driver. VDS is an industry-wide specification supported by IBM, Microsoft, Quarterdeck, and many other hardware and software suppliers. A VDS driver requests memory addresses from the VDS manager (QEMM). If the VDS driver is for a hard disk controller, you can add the DISKBUF=1 parameter to the QEMM386.SYS line in the CONFIG.SYS to try to load the VDS driver high. Alternatively, you can load the VDS driver before QEMM.

If you cannot obtain a VDS device driver for a hard drive controller, try adding the parameter DISKBUF=2 to the QEMM386.SYS line in CONFIG.SYS. This parameter creates a buffer in unmapped memory for the disk drive to use

when necessary. You can use a higher number to get better disk performance, but it will cost you conventional memory.

**Your system reboots spontaneously when QEMM loads.**

This problem is sometimes solved with the SHADOWRAM:NONE or the TOPMEMORY:N parameters. See Chapter 7 for more information on these parameters. If neither of these parameters solve your problem, try the general troubleshooting procedure detailed above under the heading "Your PC does not function properly after installing QEMM."

**You have a problem with the Stealth ROM feature that goes away when you do not use Stealth ROM.**

The text file STEALTH.TEC in the \QEMM\TECHNOTE directory outlines the steps to take in troubleshooting a problem with Stealth ROM.

**Microsoft Windows does not run properly with QEMM installed.**

The text file WINFLOW.TEC in the \QEMM\TECHNOTE directory gives a procedure for troubleshooting some common problems with Microsoft Windows and QEMM.

**Your DOS program running in Microsoft Windows standard mode cannot find any EMS, VCPI, or DPMI memory.**

In standard mode, Microsoft Windows takes all the memory normally controlled by QEMM for its own use. Non-Windows applications started in Microsoft Windows, which might otherwise use some of QEMM's memory using EMS or VCPI, will not find any, since Windows has taken all memory. If you want to run such programs under Windows, you can limit the memory Microsoft Windows 3.0 standard mode uses by using the EMBMEM parameter (see Chapter 7).

Windows 3.1 standard mode allocates memory through more than one interface, and so the EMBMEM parameter does not limit its memory



allocation effectively. However, you can use the EMS.COM program that comes with QEMM to reserve memory before starting Windows in standard mode, and then free the memory from within a Windows DOS window. See **Chapter 11** for an example of how to use the EMS CREATE and EMS FREE commands for this purpose.

Your disk compression utility (e.g., **Stacker**, **SuperStor**) is preventing **Optimize** from completing properly.

See **Appendix B** for a list of technotes on the most popular disk compression utilities. The technotes are located in the \QEMM\TECHNOTE directory.

A graphics program has a corrupted or fuzzy display.

There is probably High RAM in an area that the graphics program needs to access for some of its display functions.

1. You may be able to correct the problem by excluding certain addresses from being used as High RAM. To exclude addresses, edit your CONFIG.SYS file and add the appropriate EXCLUDE (abbreviated "X") parameter to the QEMM386.SYS device driver line. See the next paragraph for suggestions on what addresses to exclude. Once you have added a parameter, save your CONFIG.SYS file, re-run the **Optimize** program (see **Chapter 3**), then try running your program.

If your program is running in a Super VGA or 1024x768 mode, try the parameter EXCLUDE=B000-B7FF. If you are running in standard VGA or any other mode, try EXCLUDE=FE00-FFFF or EXCLUDE=C000-C7FF. If these exclusions do not help, continue with step 2.

2. This could be a Stealth-related problem. Edit your CONFIG.SYS file and see if one of the Stealth ROM parameters (ST:F or ST:M) is on the QEMM386.SYS line. If neither of the ST parameters are there, skip to step 3.

If you see the Stealth parameter, note whether it is ST:F or ST:M, then remove it. Reboot your PC and try your program again. If everything works, continue with the rest of this step; otherwise, skip to step 3.

Run the **Analysis** procedure (see **page 97**) and when you get to step 2, use DEVICE=QEMM386.SYS ON MAPS=0 ST:x FORCESTEALTHCOPY (substitute the appropriate letter, F or M, for x in ST:x). When running the **Analysis**, be sure to access the programs whose displays are corrupted. Exclude the addresses recommended by the **Analysis**. The FORCESTEALTHCOPY parameter is only for the **Analysis** procedure; do not add it back to your original QEMM386.SYS line.

3. If you are not using Stealth ROM, run the **Analysis** procedure (see **page 97**) and be sure to access the programs whose displays are corrupted. The **Analysis** report should give you information on what addresses need to be excluded. Edit the QEMM386.SYS line in CONFIG.SYS and exclude those areas.

You cannot use 1024x768 graphics modes after installing QEMM.

Some video cards use the upper memory area B000-B7FF (the monochrome text area) to display 1024x768 graphics, and QEMM usually makes this area available for High RAM or expanded memory mapping. If you are having problems running any graphics program in 1024x768 resolution, try adding the EXCLUDE=B000-B7FF parameter to the QEMM386.SYS device driver line in CONFIG.SYS.

After installing QEMM, your floppy drives do not work properly.

If you are using the ROM parameter on the QEMM386.SYS device driver line in CONFIG.SYS, try removing it. Some ROM code that manages floppy disks is sensitive to the speed at which it runs. If removing the ROM parameter helps the problem, you should be able to restore the ROM parameter if you use the

EXCLUDE parameter to prevent QEMM from shadowing the part of the ROM that controls the floppy drives. See **Chapter 7** for more information on the ROM parameter.

If the problem persists, try removing any INCLUDE parameters that you have on the QEMM386.SYS line in CONFIG.SYS. You may be including a 4K region that is used when the floppy drives are accessed. Other parameters that may prevent floppy problems in some circumstances are the SHADOWRAM:NONE parameter and the MAPREBOOT:N parameter. See **Chapter 7** for more information on these parameters.

**You have less conventional memory after installing QEMM than you did before.**

Type QEMM and press Enter ↵ to see a Summary report. If you see the message **QEMM is not resident**, something has prevented QEMM from loading. Perhaps there is another memory manager in the CONFIG.SYS file, or perhaps there is an error in the QEMM386.SYS device driver line in CONFIG.SYS.

If the Summary report tells you QEMM is on, check the page frame address (also in the Summary report). By default, QEMM puts the page frame in upper memory (above 640K) to preserve as much free conventional memory as possible.

If the information displayed indicates Page Frame Address = 9000H or some other address that starts with a number from 0 to 9, QEMM has not found a contiguous 64K upper memory area to locate the page frame and is instead reserving 64K of conventional memory for programs that use expanded memory and require a page frame.

There are several possible ways to stop the page frame from using conventional memory. The most desirable is to use QEMM's Stealth ROM parameter, which creates more usable memory above 64K. Run OPTIMIZE /STEALTH to optimize and enable Stealth ROM.

Another solution is to consult your hardware manuals and reconfigure your system so that your adapters are at locations that do not prevent QEMM from putting the page frame in upper memory.

Finally, the last resort solution is to eliminate the page frame altogether by adding the parameter FRAME=NONE to your QEMM386.SYS device line. This saves the memory, but leaves no page frame for programs that use EMS, so these programs will not be able to use EMS.

**The Suspend/Resume feature on your laptop or notebook computer does not work when QEMM is loaded.**

See the information on the SUSPENDRESUME parameter in **Chapter 7**.

**Your Intel Inboard-AT system does not warm boot properly with QEMM loaded.**

Use the QEMM TRAP8042:Y parameter. See **Chapter 7** for more information on this parameter.

**High-speed communications fail when QEMM is active.**

If high-speed communications stop suddenly, try QEMM's TASKS=xx parameter with xx set to a number higher than QEMM's default of 16. See **Chapter 7** for a discussion of the TASKS parameter.

**QEMM posts an "Unknown MCA Adapter ID" message when loading on your microchannel system.**

QEMM's MCA.ADL file contains no information about one of the adapters on your system. If your adapter uses no RAM or ROM in upper memory, or if QEMM can detect this RAM or ROM without the help of the MCA.ADL file, then you will experience no problems because of the missing MCA.ADL entry. In this case, you can tell QEMM386.SYS not to pause for the error message by using its PAUSEONERROR:N parameter, and simply ignore the message.



If you are experiencing problems in addition to the error message, contact Quarterdeck Technical Support to have information on your adapter added to your MCA.ADL file.

#### Finding additional troubleshooting information.

We have provided additional troubleshooting information in several "README" files. See Appendix B for a list of these files and what information they contain.

#### Contacting Quarterdeck Technical Support

If you need further assistance, you can contact Quarterdeck Technical Support. See the Passport booklet included with QEMM for information on technical support.

If you contact us by mail, fax or on one of our BBS systems, please include the following information:

- ☐ The version number and serial number of QEMM.
- ☐ If you are contacting us by mail or fax include a printout from Quarterdeck's Manifest. Press F2 to print, and select **All Manifest** from the **What to Print** portion of Manifest's print menu. If you have other important hardware in the system, or if Manifest's list is incomplete, please include any additional information you think may help us diagnose your problem.
- ☐ If you cannot run Manifest, print out your CONFIG.SYS and AUTOEXEC.BAT files, and write down what hardware (include the make and model) and software (include the version) you are using.
- ☐ Give a precise description of the problem that is occurring, and the exact text of any error messages. Describe in detail the results of your troubleshooting efforts.
- ☐ Please tell us how to respond to you via mail, fax or one of the other methods we support. See your Quarterdeck Passport Support

booklet for information on contacting Quarterdeck Technical Support.

If you are contacting us by telephone:

- ☐ Be at your computer.
- ☐ Please gather the information listed above.
- ☐ When you contact our technical support representative, you need only give your customer ID or product serial number and a brief description of your hardware, software and the problem you are encountering. If the support technician requires additional information, he or she will ask for specific details.

**B**

## **QEMM's Technical Bulletins**

### **Operating Systems**

### **Microsoft Windows 3.0/3.1**

### **QEMM's Stealth ROM Feature**

### **Maximizing your DESQview and DESQview/X Windows**

### **Bus-Mastering Adapters**

QEMM includes several technical bulletins that provide detailed information on a variety of subjects. Some of these "technotes" contain troubleshooting information, others contain information on fine-tuning your system and others contain technical background information on QEMM features and how they relate to other products (e.g., DOS 6). The technotes are ASCII files located in the \QEMM\TECHNOTE directory on your hard disk. You can read them by loading them into a text editor or word processor (if you use a word processor, be sure to load the file as an ASCII file). Below is a summary of the topics discussed in the technotes and the names of the files.

QEMM is fully compatible with DOS versions 3.x and higher and DR DOS versions 5.0 and higher. For information on optimally configuring your system with your operating system, see the appropriate technote: MSDOS6.TEC, DOS5.TEC, DRDOS6.TEC.

QEMM is fully compatible with Microsoft Windows 3.0 and 3.1. For information specific to version 3.1, refer to technical bulletin WIN31.TEC.

If you experience any problems running Windows or a Windows application, we suggest you follow the troubleshooting suggestions in WINFLOW.TEC. In most cases, this guide will help you pinpoint the problem and resolve it.

**Chapter 5** of this manual describes QEMM's Stealth ROM feature. For a more detailed explanation of Stealth ROM, see STLTECH.TEC.

If you experience a problem while running a program or accessing a hardware device and you believe the problem may be related to Stealth ROM, see STEALTH.TEC for troubleshooting suggestions.

If during bootup you see the message, "QEMM386: Disabling Stealth because QEMM could not locate the ROM handler for INT XX," see XSTI.TEC for an explanation and solution.

Each time your favorite software package is upgraded, new features are added by the developers. Often, you pay for those new features, however, in additional memory overhead. The increasing memory requirements for today's software packages is a major concern for users of DESQview and DESQview/X. To help you get the most out of your PC's memory and run large programs from within DESQview and DESQview/X, we have written two technical bulletins. WINSIZE.TEC provides tips on increasing the size of DESQview windows, and MAXWINDO.TEC does the same for DESQview/X.

Certain hard disk controllers and other adapters use a technique called bus mastering to speed up access times. While this technique does speed

Disk  
Compression  
Utilities

up your hard drive, it can cause conflicts with 386 memory managers (such as QEMM), particularly when the memory manager attempts to load device drivers or memory resident programs into upper memory. Fortunately, there are solutions to the problems created by bus-mastering hardware. For a detailed explanation of bus-mastering and what can be done to resolve the problems associated with its use, see BUS-MAST.TEC.

If you are using SuperStor version 2.04 or SuperStor Pro, refer to technical bulletin SSTOR.TEC for information on using QEMM's Optimize program with SuperStor.

If you have Stacker 2.01 or later, you will want to read STACKER3.TEC for tips on using your program with QEMM. If you have version 2.0, you should contact STAC Electronics for an upgrade.

If you are using XtraDrive or DoubleDisk, you should read XTRADRV.TEC or DBLDISK.TEC for information on using these programs with Quarterdeck products.

Parity Errors

Usually, a parity error indicates a hardware problem of some kind. PARITY.TEC explains why these errors occur and tells you how to determine if you have defective memory chips or other hardware.

Troubleshooting  
QEMM

If you should experience problems after installing QEMM and you cannot find the solution in **Appendix A** of this guide, see TROUBLE.TEC. This technote provides basic troubleshooting tips and may point you toward other useful technotes.

Other problem solving technotes are:

QEMMFLOW.TEC - A thorough, advanced QEMM troubleshooting guide.

EXCLUDE.TEC - Finding specific areas of memory that need exclusions.

EXCEPT13.TEC - Detailed explanation of Exception 13 errors.

EX13FLOW.TEC - Troubleshooting the cause of an Exception 13 error.

**C**

## **QEMM/DOS 6 Memory Comparisons**

At Quarterdeck, we use Manifest to compare different PCs and memory configurations.

To assist you in understanding what you can expect from QEMM, we have included Manifest printouts comparing MS-DOS 6's MemMaker to QEMM running on three different PC brands. The information in this appendix is information that Manifest reports on its First Meg Overview and First Meg Programs screens.

The PCs that we chose for this comparison were:

- IBM Model 55
- Compaq Prolinea
- Dell 450DE

Each system was configured with MS-DOS 6 and three DOS 6 utilities (SMARTDrive, VSafe and DoubleSpace), a Microsoft mouse, Novell Netware and Microsoft Windows 3.1. Buffers were set to 15 and files set to 20. At the top of each comparison is a summary of the CONFIG.SYS and AUTOEXEC.BAT files as reported by Manifest's System/CONFIG and System/AUTOEXEC screens.

We ran Microsoft's DOS 6's MemMaker in Custom Mode—instructing MemMaker to be aggressive in using the first megabyte of memory. For MemMaker, we indicated that an EMS page frame should not be used in order to give MemMaker 64K more upper memory.

We then ran QEMM 7 with its Stealth ROM, Stealth DoubleSpace and DOS-Up features enabled. The information that follows includes Manifest reports for this configuration. We also ran QEMM 7 with no page frame—which disabled Stealth ROM and Stealth DoubleSpace—to see how well QEMM does at managing the first megabyte of memory without Stealth. The amounts of conventional memory available from this configuration are shown as footnotes.

IBM PS/2  
Model 55sx:  
DOS 6 Without  
QEMM

PC Brand/Model: IBM Model 55sx  
Operating System: MS-DOS 6  
Memory Manager: MemMaker (Custom; no page frame)  
DOS 6 Utilities: DoubleSpace, SMARTDrive, VSafe  
Extended Memory Manager: HIMEM.SYS  
Expanded Memory Manager: EMM386.SYS  
Network Driver: Novell IPXODI and NETX  
Mouse Driver: Microsoft Mouse

#### Manifest System/CONFIG

DEVICE=C:\DOS\HIMEM.SYS  
DEVICE=C:\DOS\EMM386.EXE NOEMS HIGHSCAN I=B000-B7FF x=d000-d3ff  
X=D000-D2FF ... WIN=B500-B7FF WIN=B200-B4FF  
BUFFERS=15,0  
FILES=20255D  
DOS=UMB  
LASTDRIVE=E  
FCBS=4,0RDOS=HIGH  
DEVICEHIGH /L:2,44400=C:\DOS\DBLSPACE.SYS /MOVE

#### Manifest System/AUTOEXEC

LH /L:0 C:\DOS\SMARTDRV.EXE  
C:\DOS\VSAFE  
C:\DOS\MOUSE  
LH /L:4,22496 C:\DOS\LSL  
LH /L:3,32256 C:\DOS\smc8000  
LH /L:3,30576 C:\DOS\IPXODI  
C:\DOS\NETX

#### Manifest First Meg/Overview

Memory Area	Size	Description
0000 - 003F	1K	Interrupt Area
0040 - 004F	0.3K	BIOS Data Area
0050 - 006F	0.5K	System Data
0070 - 04AB	16K	DOS
04AC - 21CA	116K	Program Area
21CB - 9FFE	504K	[Available]
9FFF - 9FFF	0K	High RAM
====Conventional memory ends at 640K====		
A000 - AFFF	64K	VGA Graphics
B000 - B139	4.9K	Unused
B13A - B1FE	3.1K	High RAM
B1FF - B7FF	24K	Unused
B800 - BFFF	32K	VGA Text
C000 - C001	0K	Unused
C002 - CFFE	63K	High RAM
CFFF - D3FF	16K	Unused
D400 - D57F	6K	ROM
D580 - D600	2K	Unused
D601 - DFFE	39K	High RAM
DFFF - DFFF	0K	Unused
E000 - F600	88K	System ROM
F601 - FDFF	31K	High RAM
FE00 - FFFF	8K	System ROM
HMA	64K	First 64K Extended



**Manifest First Meg/Programs**

Memory Area	Size	Description
04AC - 0550	2.6K	COMMAND
0551 - 0555	0.1K	[Available]
0556 - 0566	0.3K	COMMAND Environment
0567 - 056C	0.1K	VSAFE Environment
056D - 0C15	26K	SMARTDRV
0C16 - 11A6	22K	VSAFE
11A7 - 11AC	0.1K	[Available]
11AD - 15E5	16K	MOUSE
15E6 - 21C5	47K	NETX
21C6 - 21CA	0.1K	COMMAND Data
21CB - 9FFE	504K	[Available]
====Conventional memory ends at 640K====		
B13A - B1FD	3.1K	[Available]
C002 - CAD5	43K	DBLSPACE
CAD6 - CFFD	20K	[Available]
D601 - D606	0.1K	[Available]
D607 - D83B	8.8K	SMC8000
D83C - DC36	15K	IPXODI
DC37 - DFFD	15K	[Available]
F601 - F606	0.1K	[Available]
F607 - FB75	21K	LSL
FB76 - FDFF	10K	[Available]

---

**IBM PS/2  
Model 55sx:  
DOS 6 With  
QEMM**

PC Brand/Model: IBM Model 55sx  
Operating System: MS-DOS 6  
Memory Manager: QEMM 7  
DOS 6 Utilities: DoubleSpace, SMARTDrive, VSafe  
Extended Memory Manager: QEMM386.SYS  
Expanded Memory Manager: QEMM386.SYS with Stealth  
Network Driver: Novell IPXODI and NETX  
Mouse Driver: Microsoft Mouse

---

**Manifest System/CONFIG**

```
DEVICE=C:\QEMM\DOSDATA.SYS
DEVICE=C:\QEMM\QEMM386.SYS R:2 x=d000-d3ff i=d500 ST:M RAM
DEVICE=C:\QEMM\DOS-UP.SYS @C:\QEMM\DOS-UP.DAT
DEVICE=C:\QEMM\LOADHLSYS /R:1 C:\QEMM\ST-DBL.SYS
DEVICE=C:\QEMM\LOADHLSYS /SHELL /R:2 C:\COMMAND.COM /P
DOS=HIGH
BUFFERS=15
FILES=20
```

---

**Manifest System/AUTOEXEC**

```
set comspec=c:\command.com
path c:\util;c:\qemm;c:\dos
C:\QEMM\LOADHI /R:2 C:\DOS\SMARTDRV.EXE
C:\QEMM\LOADHI /R:2 C:\DOS\VSAFE
C:\QEMM\LOADHI /R:2 C:\DOS\MOUSE
C:\QEMM\LOADHI /R:1 C:\DOS\LSL
C:\QEMM\LOADHI /R:2 C:\DOS\smc8000
C:\QEMM\LOADHI /R:2 C:\DOS\IPXODI
C:\QEMM\LOADHI /R:2 C:\DOS\NETX
```

---

**Manifest First Meg/Overview**

Memory Area	Size	Description
0000 - 003F	1K	Interrupt Area
0040 - 004F	0.3K	BIOS Data Area
0050 - 006F	0.5K	System Data
0070 - 023B	7.2K	DOS
023C - 0274	0.9K	Program Area
0275 - 9FFF	630K	[Available]
====Conventional memory ends at 640K====		
A000 - AFFF	64K	VGA Graphics
B000 - B7FF	32K	High RAM
B800 - BFFF	32K	VGA Text
C000 - CFFF	64K	Page Frame
D000 - D3FF	16K	Unused
D400 - FFA5	174K	High RAM
FFA6 - FFFF	1.4K	System ROM
HMA	64K	First 64K Extended

---

**Manifest First Meg/Programs**

Memory Area	Size	Description
023C - 024C	0.3K	COMMAND
024D - 0251	0.1K	COMMAND Data
0252 - 0263	0.3K	[Available]
0264 - 0274	0.3K	COMMAND Environment
0275 - 9FFF	630K	[Available]
====Conventional memory ends at 640K====		
B001 - B0AC	2.7K	ST-DBL
B0AD - B0E6	0.9K	FILES
B0E7 - B0F8	0.3K	FCBS
B0F9 - B11A	0.5K	WKBUFFER
B11B - B138	0.5K	LASTDRIV
B139 - B1AE	1.8K	STACKS
B1AF - B1B9	0.2K	INSTALL
B1BA - B1C0	0.1K	[Available]
B1C1 - B72F	21K	LSL
B730 - B7FE	3.2K	[Available]
D400 - D743	13K	QEMM386
D744 - D751	0.2K	DOSDATA
D752 - D88F	5K	DOSDATA
D890 - D934	2.6K	COMMAND
D935 - D93B	0.1K	VSAFE Environment
D93C - DFE4	26K	SMARTDRV
DFE5 - E186	6.5K	VSAFE
E187 - E18D	0.1K	[Available]
E18E - E5C6	16K	MOUSE
E5C7 - E7FB	8.8K	SMC8000
E7FC - EBF6	15K	IPXODI
EBF7 - F7D6	47K	NETX
F7D7 - FFA5	31K	[Available]

\* QEMM 7 with no page frame and no Stealth ROM:

Available conventional memory: 532K; available upper memory: 53K

Compaq  
Prolinea:  
DOS 6 Without  
QEMM

PC Brand/Model: Compaq Prolinea  
Operating System: MS-DOS 6  
Memory Manager: MemMaker (Custom; no page frame)  
DOS 6 Utilities: DoubleSpace, SMARTDrive, VSafe  
Extended Memory Manager: HIMEM.SYS  
Expanded Memory Manager: EMM386.SYS  
Network Driver: Novell IPXODI and NETX  
Mouse Driver: Microsoft Mouse

Manifest System/CONFIG

DEVICE=C:\DOS\HIMEM.SYS  
DEVICE=C:\DOS\EMM386.EXE NOEMS HIGHSCAN I=B000-B7FF X=C700-C7FF  
WIN=B500-B7 ...FF WIN=B200-B4FF  
BUFFERS=15,0  
FILES=20  
DOS=UMB  
LASTDRIVE=H  
FCBS=4,0  
DOS=HIGH  
DEVICEHIGH /L:2,44400 =C:\DOS\DBLSPACE.SYS /MOVE

Manifest System/AUTOEXEC

LH /L:0 C:\DOS\SMARTDRV.EXE  
LH /L:2,63088 C:\DOS\VSAFE  
C:\DOS\MOUSE  
LH /L:2,22496 C:\DOS\LSL  
LH /L:2,20304 C:\DOS\NE1000  
C:\DOS\IPXODI  
LH /L:2,69376 C:\DOS\NETX

Manifest First Meg/Overview

Memory Area	Size	Description
0000 - 003F	1K	Interrupt Area
0040 - 004F	0.3K	BIOS Data Area
0050 - 006F	0.5K	System Data
0070 - 04C7	17K	DOS
04C8 - 145D	62K	Program Area
145E - 9FFE	558K	[Available]
9FFF - 9FFF	0K	High RAM
====Conventional memory ends at 640K====		
A000 - AFFF	64K	VGA Graphics
B000 - B139	4.9K	Unused
B13A - B1FE	3.1K	High RAM
B1FF - B7FF	24K	Unused
B800 - BFFF	32K	VGA Text
C000 - C5FF	24K	ROM
C600 - C67F	2K	Unused
C680 - C6FF	2K	ROM
C700 - C801	4K	Unused
C802 - EFFF	159K	High RAM
F000 - FFFF	64K	System ROM
HMA	64K	First 64K Extended

**Manifest First Meg/Programs**

Memory Area	Size	Description
04C8 - 056C	2.6K	COMMAND
056D - 0571	0.1K	[Available]
0572 - 0582	0.3K	COMMAND Environment
0583 - 0588	0.1K	[Available]
0589 - 0C31	26K	SMARTDRV
0C32 - 105D	16K	MOUSE
105E - 1458	15K	IPXODI
1459 - 145D	0.1K	COMMAND Data
145E - 9FFE	558K	[Available]
====Conventional memory ends at 640K====		
B13A - B1FD	3.1K	[Available]
C802 - D2D5	43K	DBLSPACE
D2D6 - D2DB	0.1K	VSAFE Environment
D2DC - D86C	22K	VSAFE
D86D - D872	0.1K	[Available]
D873 - DDE1	21K	LSL
DDE2 - DF02	4.5K	NE1000
DF03 - EAE2	47K	NETX
EAE3 - EFFF	20K	[Available]

---



**Compaq  
Prolinea: DOS 6  
With QEMM**

PC Brand/Model: Compaq Prolinea  
Operating System: MS-DOS 6  
Memory Manager: QEMM 7  
DOS 6 Utilities: DoubleSpace, SMARTDrive, VSafe  
Extended Memory Manager: QEMM386.SYS  
Expanded Memory Manager: QEMM386.SYS with Stealth  
Network Driver: Novell IPXODI and NETX  
Mouse Driver: Microsoft Mouse

---

**Manifest System/CONFIG**

```
DEVICE=C:\QEMM\DOSDATA.SYS
DEVICE=C:\QEMM\QEMM386.SYS R:1 DBF=2 I=C000-C7FF ST:M RAM
DEVICE=C:\QEMM\DOS-UP.SYS @C:\QEMM\DOS-UP.DAT
DEVICE=C:\QEMM\LOADHI.SYS /R:1 C:\QEMM\ST-DBL.SYS
DEVICE=C:\QEMM\LOADHI.COM /SHELL /R:2 C:\COMMAND.COM /P
DOS=HIGH
BUFFERS=1
FILES=20
```

---

**Manifest System/AUTOEXEC**

```
C:\QEMM\BUFFERS=
C:\QEMM\LOADHI /R:2 C:\DOS\SMARTDRV.EXE
C:\QEMM\LOADHI /R:2 C:\DOS\VSAFE
C:\QEMM\LOADHI /R:2 C:\DOS\MOUSE
C:\QEMM\LOADHI /R:2 C:\DOS\LSL
C:\QEMM\LOADHI /R:2 C:\DOS\NE1000
C:\QEMM\LOADHI /R:2 C:\DOS\IPXODI
C:\QEMM\LOADHI /R:2 C:\DOS\NETX
```

---

**Manifest First Meg/Overview**

Memory Area	Size	Description
0000 - 003F	1K	Interrupt Area
0040 - 004F	0.3K	BIOS Data Area
0050 - 0131	3.5K	System Data
0132 - 02E6	6.8K	DOS
02E7 - 02FC	0.3K	Program Area
02FD - 9FFF	628K	[Available]
====Conventional memory ends at 640K====		
A000 - AFFF	64K	VGA Graphics
B000 - B4CA	19K	High RAM
B4CB - B4CB	0K	Unused
B4CC - B7FF	12K	High RAM
B800 - BFFF	32K	VGA Text
C000 - CFFF	64K	Page Frame
D000 - FFA5	190K	High RAM
FFA6 - FFFF	1.4K	System ROM
HMA	64K	First 64K Extended

---

**Manifest First Meg/Programs**

Memory Area	Size	Description
02E7 - 02F7	0.3K	COMMAND
02F8 - 02FC	0.1K	COMMAND Data
02FD - 9FFF	628K	[Available]
====Conventional memory ends at 640K=====		
B000 - B37E	13K	QEMM386
B37F - B38C	0.2K	DOSDATA
B38D - B4CA	5K	(B38E)
B4CC - B577	2.7K	STM-DBL
B578 - B5B1	0.9K	FILES
B5B2 - B5C3	0.3K	FCBS
B5C4 - B5E5	0.5K	WKBUFFER
B5E6 - B613	0.7K	LASTDRIV
B614 - B689	1.8K	STACKS
B68A - B694	0.2K	INSTALL
B695 - B6A5	0.3K	COMMAND Environment
B6A6 - B7FE	5.4K	[Available]
D000 - D0A4	2.6K	COMMAND
D0A5 - D0AA	0.1K	VSAFE Environment
D0AB - D753	26K	SMARTDRV
D754 - D8F5	6.5K	VSAFE
D8F6 - D8FB	0.1K	[Available]
D8FC - DD27	16K	MOUSE
DD28 - E296	21K	LSL
E297 - E3B7	4.5K	NE1000
E3B8 - E7B2	15K	IPXODI
E7B3 - F392	47K	NETX
F393 - FFA5	48K	[Available]

\* QEMM 7 with no page frame and no Stealth ROM:

Available conventional memory: 576K; available upper memory: 27K

**Dell 450DE:  
DOS 6 Without  
QEMM**

PC Brand/Model: Dell 450DE  
Operating System: MS-DOS 6  
Memory Manager: MemMaker (Custom; no page frame)  
DOS 6 Utilities: DoubleSpace, SMARTDrive, VSafe  
Extended Memory Manager: HIMEM.SYS  
Expanded Memory Manager: EMM386.SYS  
Network Driver: Novell IPXODI and NETX  
Mouse Driver: Microsoft Mouse

**Manifest System/CONFIG**

DEVICE=C:\DOS\HIMEM.SYS  
DEVICE=C:\DOS\EMM386.EXE NOEMS I=B000-B7FF WIN=B500-B7FF WIN=B200-B4FF  
BUFFERS=15,0  
FILES=20  
DOS=UMB  
LASTDRIVE=E  
FCBS=4,0  
DOS=HIGH  
DEVICEHIGH /L:2,44304 =C:\DOS\DBLSPACE.SYS /MOVE

**Manifest System/AUTOEXEC**

LH /L:0 C:\DOS\SMARTDRV.EXE  
LH /L:2,63088 C:\DOS\VSAFE  
C:\DOS\MOUSE  
LH /L:2,22496 C:\DOS\LSL  
LH /L:2,20304 C:\DOS\NE1000  
C:\DOS\IPXODI  
LH /L:2,69376 C:\DOS\NETX

**Manifest First Meg/Overview**

Memory Area	Size	Description
0000 - 003F	1K	Interrupt Area
0040 - 004F	0.3K	BIOS Data Area
0050 - 006F	0.5K	System Data
0070 - 04A7	16K	DOS
04A8 - 149D	63K	Program Area
149E - 9FFE	557K	[Available]
9FFF - 9FFF	0K	High RAM
-----Conventional memory ends at 640K-----		
A000 - AFFF	64K	VGA Graphics
B000 - B139	4.9K	Unused
B13A - B1FE	3.1K	High RAM
B1FF - B7FF	24K	Unused
B800 - BFFF	32K	VGA Text
C000 - C7FF	32K	Video ROM
C800 - C801	0K	Unused
C802 - EFFF	159K	High RAM
F000 - FFFF	64K	System ROM
HMA	64K	First 64K Extended

# **Manifest First Meg/Programs**

Memory Area	Size	Description
04A8 - 054C	2.6K	COMMAND
054D - 0551	0.1K	[Available]
0552 - 0562	0.3K	COMMAND Environment
0563 - 0568	0.1K	[Available]
0569 - 0C71	28K	SMARTDRV
0C72 - 109D	16K	MOUSE
109E - 1498	15K	IPXODI
1499 - 149D	0.1K	COMMAND Data
149E - 9FFE	557K	[Available]
-----Conventional memory ends at 640K-----		
B13A - B1FD	3.1K	[Available]
C802 - D2CF	43K	DBLSPACE
D2D0 - D2D5	0.1K	VSAFE Environment
D2D6 - D866	22K	VSAFE
D867 - D86C	0.1K	[Available]
D86D - DDD8	21K	LSL
DDDC - DEFC	4.5K	NE1000
DEFD - EADC	47K	NETX
EADD - EFFF	20K	[Available]

**Dell 450DE:  
DOS 6 With  
QEMM**

PC Brand/Model: Dell 450DE  
Operating System: MS-DOS 6  
Memory Manager: QEMM7  
DOS 6 Utilities: DoubleSpace, SMARTDrive, VSafe  
Extended Memory Manager: QEMM386.SYS  
Expanded Memory Manager: QEMM386.SYS with Stealth  
Network Driver: Novell IPXODI and NETX  
Mouse Driver: Microsoft Mouse

**Manifest System/CONFIG**

DEVICE=C:\QEMM\DOSDATA.SYS  
DEVICE=C:\QEMM\QEMM386.SYS R:1 dbf=2 I=B000-B7FF RAM ST:M  
DEVICE=C:\QEMM\DOS-UP.SYS @C:\QEMM\DOS-UP.DAT  
DEVICE=C:\QEMM\LOADHI.COM /SHELL /R:2 C:\COMMAND.COM /P  
DOS=HIGH  
BUFFERS=1  
FILES=20  
DEVICE=C:\QEMM\LOADHLSYS /R:1 C:\QEMM\ST-DBL.SYS

**Manifest System/AUTOEXEC**

C:\QEMM\LOADHI /R:2 C:\QEMM\BUFFERS=14  
C:\QEMM\LOADHI /R:2 C:\DOS\SMARTDRV.EXE  
C:\QEMM\LOADHI /R:2 C:\DOS\VSAFE  
C:\QEMM\LOADHI /R:2 C:\DOS\MOUSE  
C:\QEMM\LOADHI /R:2 C:\DOS\LSL  
C:\QEMM\LOADHI /R:2 C:\DOS\NE1000  
C:\QEMM\LOADHI /R:2 C:\DOS\IPXODI  
C:\QEMM\LOADHI /R:2 C:\DOS\NETX

**Manifest First Meg / Overview**

Memory Area	Size	Description
0000 - 003F	1K	Interrupt Area
0040 - 004F	0.3K	BIOS Data Area
0050 - 0125	3.3K	System Data
0126 - 02DA	6.8K	DOS
02DB - 02F0	0.3K	Program Area
02F1 - 9FFF	628K	[Available]
====Conventional memory ends at 640K====		
A000 - AFFF	64K	VGA Graphics
B000 - B48E	18K	High RAM
B48F - B48F	0K	Unused
B490 - B7FF	13K	High RAM
B800 - BFFF	32K	VGA Text
C000 - CFFF	64K	Page Frame
D000 - FFA5	190K	High RAM
FFA6 - FFFF	1.4K	System ROM
HMA	64K	First 64K Extended



**Manifest First Meg/Programs**

Memory Area	Size	Description
02DB - 02EB	0.3K	COMMAND
02EC - 02F0	0.1K	COMMAND Data
02F1 - 9FFF	628K	[Available]
====Conventional memory ends at 640K====		
B000 - B342	13K	QEMM386
B343 - B350	0.2K	DOSDATA
B351 - B48E	5K	(B352)
B490 - B53B	2.7K	ST-DBL
B53C - B575	0.9K	FILES
B576 - B587	0.3K	FCBS
B588 - B5A9	0.5K	WKBUFFER
B5AA - B5C7	0.5K	LASTDRIV
B5C8 - B63D	1.8K	STACKS
B63E - B648	0.2K	INSTALL
B649 - B659	0.3K	COMMAND Environment
B65A - B7FE	6.6K	[Available]
D000 - D0A4	2.6K	COMMAND
D0A5 - D0AA	0.1K	VSAFE Environment
D0AB - D284	7.4K	BUFFERS
D285 - D98D	28K	SMARTDRV
D98E - DB2F	6.5K	VSAFE
DB30 - DB35	0.1K	[Available]
DB36 - DF61	16K	MOUSE
DF62 - E4D0	21K	LSL
E4D1 - E5F1	4.5K	NE1000
E5F2 - E9EC	15K	IPXODI
E9ED - F5CC	47K	NETX
F5CD - FFA5	39K	[Available]

\* QEMM 7 with no page frame and no Stealth ROM:

Available conventional memory: 580K; available upper memory: 21K



## Memory Specifications Supported by QEMM

QEMM supports a variety of industry-standard specifications that are designed to remove incompatibilities between programs, or between a program and hardware. Below is a brief description of each of these specifications.

- **EMS (Expanded Memory Specification)** lays down the guidelines for the use of expanded memory. Expanded memory is memory outside of the first megabyte of memory addresses that can be made to appear in the first megabyte. Its purpose is to give programs access to more memory than exists in the one megabyte address space to which real-mode programs are limited. Most programs use EMS memory to store large amounts of data; multitasking environments like DESQview and DESQview/X use it to run multiple large programs at the same time. The Expanded Memory Specification describes programming calls that programs can make and that an expanded memory manager (EMM) like QEMM must service. For copies of the EMS documentation, call (800) 548-4725 or write Intel Literature JP26, 3065 Bowers Ave., P.O. Box 58065, Santa Clara, CA 95051-8065.
- **XMS (Extended Memory Specification)** governs access to three different areas of memory: extended memory, which is memory addressed above the one-megabyte boundary and accessible only in protected mode; upper memory, which is the "reserved area" between 640K and one megabyte; and the High Memory Area, the first 64K of extended memory, and the only region of extended memory that can be accessed in real mode.

These three areas correspond to the three kinds of memory that XMS programs obtain from an XMS manager: Extended Memory Blocks (EMBs), Upper Memory Blocks (UMB), and the High Memory Area (HMA). EMBs are used by real-mode programs to store large amounts of data, or by protected-mode programs for both code and data. UMBs are used by real-mode programs to place relatively small amounts of code and data above 640K, thereby reducing their footprint below 640K. The HMA, which can only be used by specially written programs, is generally allocated by operating systems or environments to reduce their footprint below 640K. It is common for one XMS manager to provide EMBs and the HMA and another to provide UMBs; QEMM provides all three XMS services. For copies of the XMS documentation, call (800) 227-4679 or write Microsoft Corporation, One Microsoft Way, Redmond, WA 98052.

- **VCPI (Virtual Control Program Interface)** makes it possible for a program to go into protected mode when a 386 memory manager like QEMM has already taken control of the system. Without such a specification, programs would cause system errors when they tried to go into protected mode with a 386 memory manager active. The protected mode application, known as the VCPI client, makes programming calls to QEMM, the VCPI host, to allocate memory and

to switch the processor's mode. A VCPI client takes control of the system from the VCPI host when it is operating in protected mode, and hands control back to the VCPI host when it switches out of protected mode. For copies of the VCPI documentation, call (617) 661-1510 or write Phar Lap Software, Inc., 60 Aberdeen Ave., Cambridge, MA 02138.

- **DPMI (DOS Protected Mode Interface)** is, like VCPI, a specification to allow the coexistence of protected mode programs and 386 memory managers. Unlike VCPI, DPMI dictates that the DPMI host be in control of the system at all times. DPMI thus gives more power to the host, whereas VCPI gives more power to the client. QEMM's support for DPMI is in the QDPMI program, which is loaded automatically when you install QEMM. For copies of the DPMI documentation, call (800) 548-4725 or write Intel Literature JP26, 3065 Bowers Ave., P.O. Box 58065, Santa Clara, CA 95051-8065.
- **VDS (Virtual DMA Services)** makes it possible for bus-mastering devices to coexist with 386 memory managers like QEMM. Bus-mastering devices (like some hard disk controllers and network adapters) do not use the system's DMA (Direct Memory Access) hardware, which provides a way to transfer data back and forth from devices to memory while the main processor is busy with other work. Instead, the bus-mastering device uses its own hardware to perform this same function. By bypassing the well-documented DMA interface, the bus-mastering device circumvents QEMM's memory mapping, and is likely to send data to the wrong locations in memory. VDS provides a programming interface that lets devices ask the VDS host (QEMM) for the correct address in memory to which to transfer data. VDS can also be used to avoid incompatibilities between QEMM and devices that use the conventional DMA hardware in unusual and unsupported ways. For copies of the VDS documentation, call (800) 227-4679 or write Microsoft Corporation, One Microsoft Way, Redmond, WA 98052.

## Index

### Numbers and symbols

1024x768 graphics 126  
10NET 69  
1DIR Plus 74  
386 ShadowRAM 72  
386MAX.SYS 18  
43-line display 33  
486 computers 6  
4DOS 32  
4DOS.CMD file 32  
50-line display 33  
8086 computers  
    See XT computers  
8088 computers  
    See XT computers  
8514a display 11, 20, 47  
?  
    EMS programs parameter 116  
    LOADHI parameter 90  
    Optimize parameter 31, 35  
    QEMM.COM parameter 91  
    QEMM386.SYS parameter 58

### A

A20 address line 74-75  
Accessed  
    QEMM.COM report 92, 96  
Accessed memory 91-92, 96-97, 99, 123  
Adapter RAM 13, 66, 84, 95, 122, 127  
Adapter ROM 13, 122, 127  
ADAPTERRAM  
    QEMM386.SYS parameter 58, 80  
ADAPTERROM  
    QEMM386.SYS parameter 59, 80  
Advanced disk features 77  
Advanced installation 18  
Alternate maps  
    See EMS, alternate maps  
Analysis 14, 21, 55, 63, 65, 72, 119, 122, 126  
    comprehensive 98, 123  
    partial 98, 123  
    QEMM.COM report 92, 97  
ANSI.SYS 85  
ARAM  
    QEMM386.SYS parameter 58, 80

### AROM

QEMM386.SYS parameter 59, 80

### AT/386 ShadowRAM

See 386 ShadowRAM

### AU

QEMM.COM parameter 91  
QEMM386.SYS parameter 59

### AUTO

Optimize parameter 28, 31  
QEMM.COM parameter 91  
QEMM386.SYS parameter 59

### AUTOEXCLUDE

Optimize parameter 28, 31, 121

### AUTOEXEC.BAT 35, 128

changed by INSTALL 22-23  
changed by Optimize 30  
embedded batch files 30  
long lines 34-35  
Optimize finding 32  
restoring pre-Optimize copy 30

### B

### B

LOADHI parameter 86  
Optimize parameter 32

### BASIC 11, 68

### BASIC programs 68

### Batch file conditional statements 27

### Batch files 14, 22, 30

on network drives 34

### Battery-powered computers 12, 20

See also Laptop computers

### BESTFIT

LOADHI parameter 86

### BLUEMAX.SYS 18

### BOOT

Optimize parameter 32

### Borland C/C++ 103, 105

### BUFFERS 14, 37-40, 62, 66

### BUFFERS.COM 14, 39-40

### BUS-MAST.TEC

See Technotes, BUS-MAST.TEC

### Bus-mastering devices 13, 61, 75, 125, 129, 146

network adapters 61, 125

### C

### C386S

QEMM386.SYS parameter 59

### CALL (DOS batch language) 14, 30

CEMM.EXE 18

CER  
     QEMM386.SYS parameter 59

CF  
     QEMM386.SYS parameter 60, 120

CFAST  
     EMS programs parameter 115

CGA display 34, 47, 51, 64, 76, 79

Chips & Technologies 13, 72, 100

CHR  
     QEMM386.SYS parameter 60

CMD  
     Optimize parameter 32

CMOS 101

Code Builder Kit 103

Command processors 32, 37, 39, 88

COMMAND.COM 30, 37, 39, 109

COMMANDFILE  
     Optimize parameter 32, 122

Communications programs 70, 73, 127

Compaq 386s computer 59

Compaq computers 12, 17, 19, 24, 59-60, 78  
     Setup program 59  
     CEMM memory manager 18  
     cut table 13  
     disk cache 61  
     half ROM 13, 60  
     HIMEM.EXE 17, 24  
     Prolinea 15, 131, 136, 138

COMPAQ386S  
     QEMM386.SYS parameter 59

COMPAQEGAROM  
     QEMM386.SYS parameter 59-60

COMPAQFEATURES  
     QEMM386.SYS parameter 60-61, 120

COMPAQHALFROM  
     QEMM386.SYS parameter 60

COMPAQROMMEMORY  
     QEMM386.SYS parameter 60

Conditional statements in batch files 27

CONFIG.SYS 35, 53-54, 128  
     changed by INSTALL 22  
     changed by Optimize 30  
     how to edit 119  
     long lines 35  
     Optimize finding 32  
     restoring pre-Optimize copy 30

Conventional memory 6, 33, 47  
     decreasing the size of 55, 61, 67, 98, 101, 123  
     definition 6  
     increasing the size of 9, 11, 15, 19, 24, 27, 37-40, 43, 45-52, 56, 64, 76-77, 79, 83, 91, 127  
     and QDPMI 107  
     QEMM vs. DOS 6 131

CR  
     EMS programs parameter 115

CREATE  
     EMS programs parameter 115, 126

CREATEFAST  
     EMS programs parameter 115

CREATESLOW  
     EMS programs parameter 115

CKM  
     QEMM386.SYS parameter 60

CSLOW  
     EMS programs parameter 115

D

D4  
     QEMM386.SYS parameter 62

DB  
     QEMM386.SYS parameter 61-62, 120, 125

DBF  
     QEMM386.SYS parameter 61

DBLDISK.TEC  
     See Technotes, DBLDISK.TEC

DBLSPACE.BIN 45

DBLSPACE.SYS 45-46

Dell computers  
     450DE 15, 131, 140, 142

DESQview 7, 10-13, 24-25, 27, 29, 37, 43, 56, 66, 68, 70, 76-77, 79, 83, 98, 109, 116, 123-124, 145  
     DESQview 386 5, 103  
     and DOS=HIGH 38  
     and the HMA 25  
     increasing window size 129  
     protection level 78  
     and QDPMI 103-104, 106, 108, 110-111  
     and VIDRAM 49-51

DESQview/X 12-13, 24, 29, 37, 43, 56, 66, 68, 70, 76, 79, 83, 103, 124, 145  
     and DOS=HIGH 38  
     and QDPMI 103-104, 106, 108, 110-111  
     and the HMA 25  
     increasing window size 129  
     protection level 78

Device drivers 84-85, 101



DEVICEHIGH (DOS command) 90  
 DIR  
     EMS programs parameter 115  
     Optimize parameter 33  
 Direct Memory Access  
     See DMA 16  
 DIRECTORY  
     Optimize parameter 33  
 Disk caches 61, 77  
     advanced disk features 77  
 Disk compression programs 45, 61, 130  
 Disk controllers 13, 47, 129, 146  
 Disk parameter tables 65, 71  
 DISKBUF  
     QEMM386.SYS parameter 61-62, 120, 125  
 DISKBUFFRAME  
     QEMM386.SYS parameter 61  
 Diskless workstation 78  
 DM  
     QEMM386.SYS parameter 62  
 DMA 13, 62, 69, 125, 146  
     QEMM386.SYS parameter 62  
 DONE  
     Optimize parameter 33  
 DOS 7, 10, 37-42, 45, 103, 105, 129  
     loading DOS high 37  
 DOS version 3 10, 15, 38, 41  
 DOS version 4 10, 15, 38, 41, 62  
     /X parameter 62  
 DOS version 5 9-10, 15, 17, 23-24, 37, 41, 90  
 DOS version 6 9-10, 15, 17, 23-24, 26, 37, 41, 43,  
     45, 90  
     memory gains vs. QEMM 131  
 DOS BUFFERS  
     See BUFFERS  
 DOS COMMAND.COM  
     See COMMAND.COM  
 DOS data 37-38  
 DOS devices 84  
     See also Device drivers  
 DOS Edit  
     See EDIT.COM  
 DOS extenders 7, 108  
     definition 7  
 DOS FASTOPEN  
     See FASTOPEN.EXE  
 DOS FILES  
     See FILES  
 DOS HIMEM  
     See HIMEM.SYS

DOS memory chain 88  
 DOS Protected Mode Interface  
     See DPMI  
 DOS Qbasic  
     See QBASIC.EXE  
 DOS Resource programs 37, 39  
     See also BUFFERS.COM, FILES.COM, FCBS.COM,  
     LASTDRIV.COM  
 DOS STACKS  
     See STACKS  
 DOS SUBST  
     See SUBST.EXE  
 DOS VDISK  
     See VDISK.SYS  
 DOS-Up 10, 15, 17, 20, 22-26, 37-39, 88  
     DOS-UP.SYS 22, 38  
     DOSDATA.SYS 22, 38  
     enabling/disabling 38  
 DOS-UP.DAT file 22  
 DOS-UP.SYS 22, 38  
 DOS4  
     QEMM386.SYS parameter 62  
 DOS5.TEC  
     See Technotes, DOS5.TEC  
 DOS=HIGH 23-25, 37-38  
 DOSDATA.SYS 22, 38  
 DoubleDisk 130  
 DoubleSpace 6, 10, 17, 23, 26, 43, 45-46  
 DPMI 11, 15, 22, 25, 62, 64, 70, 101, 103-111, 114,  
     116, 124-125, 146  
     16- and 32-bit support 105, 111  
     clients 103, 105, 110  
     enabling 104  
     hosts 103  
 DR DOS 24, 103, 129  
     HIDOS=ON 24  
 DRDOS6.TEC  
     See Technotes, DRDOS6.TEC  
 Dual monitors 50  
  
 E  
 EDIT.COM 68  
 Editing configuration files 26  
 EEMS 11, 94  
 EGA  
     VIDRAM parameter 50, 76  
 EGA display 11, 20, 47-49, 51, 76-77, 79  
 EISA computers 78  
 EMB 11, 62, 145  
     QEMM386.SYS parameter 62, 125

## EMBMEM

QEMM386.SYS parameter 62, 125

## EMM

Optimize parameter 33

EMM386.SYS 17, 24

EMS 4, 11, 48, 50-51, 62, 64-65, 67, 70, 77-78, 91-94,  
97, 101, 105, 113, 125, 127, 145

alternate maps 11, 69

definition 4

disk access and the page frame 66

EMS-using programs 61, 66, 74, 76-78

handle names 113

handles 14, 113, 117

page frame 8-10, 30, 33, 44-45, 55-57, 61, 65-67, 74,  
89, 92-94, 97, 114, 127

page ordering 62, 67

pages 8, 115

QEMM386.SYS parameter 56, 58, 62

version 3.2 11

version 4 11

VIDRAM parameter 49, 51, 77

EMS programs 14, 113

EMS.COM 113-114, 116, 126

EMS.SYS 113-114

summary report 114

EMS programs parameter

? 116

CFAST 115

CR 115

CREATE 115, 126

CREATEFAST 115

CREATESLOW 115

CSLOW 115

DIR 115

FREE 115, 126

HELP 116

LOAD 116

REN 115

RENAME 115

RES 115

RESIZE 115

SAVE 116

EMS2EXT 113, 116

EMS2EXT parameter

MEMORY 117

SPEED 117

Enhanced Expanded Memory Specification

See EEMS

Ergo 108

## EX13FLOW.TEC

See Technotes, EX13FLOW.TEC

## EXCEPT13.TEC

See Technotes, EXCEPT13.TEC

Exception 13 error message 21, 124-125, 130

## EXCLUDE

QEMM386.SYS parameter 55-57, 59, 63, 65, 72, 74,  
76, 80, 94, 99-100, 119, 121, 123-127, 130

## EXCLUDE.TEC

See Technotes, EXCLUDE.TEC

## EXCLUDELARGEST

LOADHI parameter 87

## EXCLUDEREGION

LOADHI parameter 87

## EXCLUDESMALLEST

LOADHI parameter 87

## EXCLUDESTEALTH

QEMM386.SYS parameter 58, 63

## EXCLUDESTEALTHINT

QEMM386.SYS parameter 58, 63

Excluding programs from Optimize 31-32, 122

Expanded memory

See EMS

Expanded Memory Specification

See EMS

Express installation 9, 18

## EXT

QEMM386.SYS parameter 59, 64, 100

## EXTCHKON

QDPMI parameter 108

## EXTCHOFF

QDPMI parameter 108

Extended BIOS Data Area

See XBDA

Extended memory 7, 64, 70, 72-75, 79, 100-101,  
103, 145

See also XMS

definition 7

INT 15 interface 101, 113, 116

Extended Memory Block

See EMB

Extended Memory Specification

See XMS

## EXTMEM

QEMM386.SYS parameter 59, 64, 101

F

F

Optimize parameter 33

F10

QEMM386.SYS parameter 64

FASTINT10

QEMM386.SYS parameter 64

FASTOPEN.EXE 62

FCBS 37-39, 41

FCBS.COM 39, 41

FEMS

QEMM386.SYS parameter 65, 67

50-line display 33

FILE

Optimize parameter 33

File Control Blocks

See FCBS

FILES 14, 37-40, 120

FILES.COM 14, 39-40

FILL

QEMM386.SYS parameter 64, 76, 120

FINISHED

Optimize parameter 33

FL

QEMM386.SYS parameter 65, 67

Floppy drives 56, 69, 72, 80, 126

FORCEEMS

QEMM386.SYS parameter 65, 67

FORCESTEALTHCOPY

QEMM386.SYS parameter 65, 98, 122, 126

43-line display 33

486 computers 6

FR

QEMM386.SYS parameter 56, 58, 63, 65, 67, 127

FRAME

QEMM386.SYS parameter 56, 58, 63, 65, 67, 127

FRAMELENGTH

QEMM386.SYS parameter 65, 67

FREE

EMS programs parameter 115, 126

FSTC

QEMM386.SYS parameter 65, 97, 122, 126

G

Gas plasma display 18, 27

GETSIZE

LOADHI parameter 87-88

QEMM386.SYS parameter 67

Glyphix 74

Graphics programs 48-51, 126

GS

LOADHI parameter 87-88

QEMM386.SYS parameter 67

H

H

LOADHI parameter 86

HA

QEMM386.SYS parameter 67

HANDLES

QEMM386.SYS parameter 67

HAPPIEST

LOADHI parameter 86

Hard disk controllers

See Disk controllers

Hard disk managers 120

HELP

EMS programs parameter 116

LOADHI parameter 90

Optimize parameter 31, 33

QEMM.COM parameter 91

QEMM386.SYS parameter 68

Hercules display 47, 64, 76, 79

Hexadecimal numbering 53

HIDOS=ON 24

High Memory Area

See HMA

High RAM 8-9, 19, 27-28, 37, 42-43, 49-52, 55, 57, 61

63, 65-67, 71, 76-78, 80, 83-84, 88, 91, 94, 96, 98,

See also RAM, QEMM386.SYS parameter

122-123

definition 8

region 30, 84, 87

and UMBs 9

HIMEM.EXE 17, 24

HIMEM.SYS 17, 24, 75, 116

HMA 7, 10-11, 15, 23-25, 37-38, 68, 79, 145

definition 7

QEMM386.SYS parameter 68, 79

HMAMIN

QEMM386.SYS parameter 68

Hyperdisk 87

I

I

QEMM386.SYS parameter 51, 55, 68, 80, 99-100,

119, 127

I/O port trapping 69

I386  
     QEMM386.SYS parameter 68, 80  
 i486 computers 6  
 IB  
     QEMM386.SYS parameter 68  
 IBM BASIC area 11, 68  
 IBM computers 11, 68  
     See also PS/2 computers  
     Model 55sx 15, 131-132, 134  
 IBM DOS  
     See DOS  
 IBMBASIC  
     QEMM386.SYS parameter 68  
 IBMDOS.COM 37  
 Inboard-AT computers 74, 127  
 INCLUDE  
     QEMM386.SYS parameter 51, 55, 68, 80, 99-100,  
     119, 123-124, 127  
 INCLUDE386  
     QEMM386.SYS parameter 68, 80  
 Increasing conventional memory  
     See Conventional memory, increasing the size of  
 INSTALL  
     advanced 18  
     express 9, 18  
     on gas plasma display 18  
     on LCD display 18  
     M parameter 18  
     on monochrome display 18  
     QEMM program 17-18, 27, 37, 43, 45, 56  
 INSTALL (DOS command) 90  
 Installing QEMM 17  
 Intel Code Builder Kit 103  
 Intel Inboard-AT computers  
     See Inboard-AT computers  
 Interrupts 44, 63, 73, 77, 129  
     locking 69, 78  
 IOTRAP  
     QEMM386.SYS parameter 69  
  
 K  
 KEEPSWAP  
     QDPMI parameter 107  
 8042 keyboard controller 74  
 Keyboard problems 74  
 KILLSWAP  
     QDPMI parameter 104, 107  
 KLSW  
     QDPMI parameter 104, 107

KPSW  
     QDPMI parameter 107  
  
 L  
  
 L  
     LOADHI parameter 87  
     Optimize parameter 33  
 LA  
     Optimize parameter 33  
 LABEL  
     LOADHI parameter 88  
     QEMM386.SYS parameter 69  
 Laptop computers 12, 18, 20, 73, 127  
 LARGE  
     Optimize parameter 33  
 LARGEST  
     LOADHI parameter 87  
 LASTDRIV.COM 39, 41  
 LASTDRIVE 37-39, 41  
 LB  
     LOADHI parameter 88  
     QEMM386.SYS parameter 69  
 LCD display 18, 27  
 LD  
     QEMM386.SYS parameter 69, 125  
 LEAP 72  
 LH (DOS command) 90  
 LINK  
     LOADHI parameter 88  
 LINKTOP  
     LOADHI parameter 88  
 LO  
     LOADHI parameter 89  
 LOAD  
     EMS programs parameter 116  
 LOADHI parameter  
     ? 90  
     B 86  
     BESTFIT 86  
     EXCLUDELARGEST 87  
     EXCLUDEREGION 87  
     EXCLUDESMALLEST 87  
     GETSIZE 87-88  
     GS 87-88  
     H 86  
     HAPPIEST 86  
     HELP 90  
     L 87  
     LABEL 88  
     LARGEST 87

## LOADHI parameter (continued)

LB 88  
LINK 88  
LINKTOP 88  
LO 89  
NOLO 88  
NL 88  
Q 89  
QUIET 89  
R 22, 30, 87  
REGION 22, 30, 87  
RES 89  
RESIDENTSIZE 89  
S 87  
SHELL 23, 38, 88  
SIZE 88  
SMALLEST 87  
SQF 89  
SQT 89  
SQUEEZEF 89  
SQUEEZET 89  
TERMINATERESIDENT 90  
TSR 90  
UNLINK 88  
UNLINKTOP 88  
XL 87  
XR 87  
XS 87  
LOADHI programs 32-35, 56, 83, 124  
LOADHI.COM 23, 30, 34, 39, 41, 85, 89  
LOADHISYS 22, 30, 46, 84-85  
loading programs in conventional memory 83, 88-89  
parameter files 89  
LOADHI report 14, 28, 84, 90  
LOADHI.COM  
See LOADHI programs, LOADHI.COM  
LOADHIRF file 34, 89  
LOADHISYS  
See LOADHI programs, LOADHISYS  
LOADHIDATA environment variable 89  
LOADHIGH (DOS command) 90  
LOADHIONLY  
Optimize parameter 33  
LOADLOW  
Optimize parameter 33  
LOCKDMA  
QEMM386.SYS parameter 69, 125

## LOW

Optimize parameter 33  
QDPMI parameter 107  
LWPFIX.COM 105

## M

### M

INSTALL parameter 18  
Optimize parameter 27, 33  
MA  
QEMM386.SYS parameter 69, 98, 122, 126  
Manifest 9, 14, 16-17, 20, 53, 59, 65, 67, 73, 75, 90-91, 114-115, 124, 128, 131  
MAP  
QEMM.COM parameter 92, 94, 96, 98, 122-123  
Mappable memory addresses 94, 97  
Mapping 8  
MAPREBOOT  
QEMM386.SYS parameter 69, 120, 127  
MAPS  
QEMM386.SYS parameter 69, 98, 122, 126  
MAXLOW  
QDPMI parameter 107-108  
MAXMEM  
QDPMI parameter 106-109  
MAXWINDO.TEC  
See Technotes, MAXWINDO.TEC  
MCA.ADL file 12, 68, 71, 127  
ME  
QEMM386.SYS parameter 59, 64, 70, 101  
MemMaker 15  
Memory  
See also Conventional memory, EMS, Extended memory, High RAM  
EMS2EXT parameter 117  
memory pool 11  
more than 16 megabytes 17, 19, 24  
QEMM.COM report 92, 100  
QEMM386.SYS parameter 59, 64, 70, 100  
required by a program 87  
speed of 50, 72, 91, 114-115, 117  
Menu programs 19  
Microchannel computers 12, 127  
See also PS/2 computers  
Microsoft C/C++ Development System for Windows 103



Microsoft Windows 7, 11-12, 17, 24, 26-27, 43, 47,  
51, 103, 125, 129  
386 enhanced mode 12, 51, 64, 73, 76, 78  
standard mode 62, 114, 125  
SYSTEM.INI 24, 26  
version 3.0 62, 78, 125  
version 3.1 103, 114, 125, 129  
and VIDRAM 12

MINMEM  
QDPMI parameter 106-108

Mode (QEMM) 91

Modify Data  
Optimize option 29

MONO  
Optimize parameter 27, 33

Monochrome display 18, 27, 47-48, 64, 76, 79

Mouse problems 74

MR  
QEMM386.SYS parameter 69, 120, 127

MS-DOS  
See DOS

MSDOS.SYS 37

MSDOS6.TEC  
See Technotes, MSDOS6.TEC

Multitasking 5, 13, 62, 66, 70, 145

N

N  
Optimize parameter 34

NDOS 32

NEAT 72

NEC computers 13, 72

Network adapters 55, 61, 124, 146  
Token-Ring 73

Networks 69, 124  
diskless workstation 78  
Novell LAN WorkPlace for DOS 105

NF  
Optimize parameter 33

NL  
LOADHI parameter 88

NOCGA  
VIDRAM parameter 51

NOEGA  
VIDRAM parameter 51

NOLO  
LOADHI parameter 88

Norton N-CACHE 61

Norton Utilities 87

NOSQF  
Optimize parameter 33

NOSQT  
Optimize parameter 34

NOST  
Optimize parameter 34

NOSTEALTH  
Optimize parameter 34

NOSYNC  
Optimize parameter 34

Notebook computers  
See Laptop computers

Novell  
LAN WorkPlace for DOS 105  
XMSNET.EXE 24

Novell LAN WorkPlace for DOS 105

NOVM  
QDPMI parameter 107

NOXCHK  
QDPMI parameter 108

NT  
Optimize parameter 34

O

ODV  
QEMM386.SYS parameter 70

OF  
QEMM386.SYS parameter 59  
VIDRAM parameter 50

OFF  
QDPMI parameter 106  
QEMM386.SYS parameter 59  
VIDRAM parameter 49-51

OLDDV  
QEMM386.SYS parameter 70

ON  
QDPMI parameter 106  
QEMM.COM parameter 91  
QEMM386.SYS parameter 59, 98, 122, 126  
VIDRAM parameter 48-52, 76-77

OPTI chip set 13, 72

Optimize 10, 13, 17, 19, 23, 27-29, 38, 43, 45-46,  
49-51, 53, 57, 61, 67, 69, 71, 83-85, 89-90, 100, 104,  
121, 124-126  
and batch files 30  
excluding programs from 31-32, 122  
Modify Data option 29, 121  
option 28  
Region Layout option 29, 122  
response file 34

## Optimize (continued)

Squeeze feature 10, 30, 33-34, 89

Stealth ROM testing 19, 25, 28-30, 34-35

undoing 30

What-If option 14, 29, 122

when to run 20, 27

Optimize parameter 31

? 31, 35

AUTO 28, 31

AUTOEXCLUDE 28, 31, 121

B 32

BOOT 32

CMD 32

COMMANDFILE 32, 122

DIR 33

DIRECTORY 33

DONE 33

EMM 33

F 33

FILE 33

FINISHED 33

HELP 31, 33

L 33

LA 33

LARGE 33

LOADHIONLY 33

LOADLOW 33

LOW 33

M 27, 33

MONO 27, 33

N 34

NF 33

NOSQF 33

NOSQT 34

NOST 34

NOSTEALTH 34

NOSYNC 34

NT 34

P 34

PATH 34

Q 28, 34

QUICK 28, 34

RESPONSE 34, 89

RF 34, 89

SEG 35

SEGMENT 35

ST 25, 29, 35

STEALTH 25, 29, 35, 43, 127

STPASSDONE 35

OPTIMIZE.EXC file 31-32, 122

OS/2 103

## OV

VIDRAM parameter 50

## OVERRIDE

VIDRAM parameter 50

## OVLDIR

QDPMI parameter 108, 110

## P

## P

Optimize parameter 34

Page frame

See EMS, page frame

Parameter files 14, 54

Parity errors 130

PARITY.TEC

See Technotes, PARITY.TEC

PATH

Optimize parameter 34

PATH environment variable 23, 34

PAUSE

QEMM386.SYS parameter 71

PAUSEONERROR

QEMM386.SYS parameter 71, 127

PC-CACHE 61

PE

QEMM386.SYS parameter 71, 127

PEAK 72

Pentium 6, 12

Phar Lap 108

Protected mode 7, 103

definition 7

PS/2 computers 9, 12, 62-63, 74, 78-79

See also IBM computers

See also Microchannel computers

Model 55sx 15, 131-132, 134

## Q

## Q

LOADHI parameter 89

Optimize parameter 28, 34

QBASIC.EXE 68

QDPMI 12, 15, 22, 25, 103-111, 146

and 80386 processors 110

environment variable 104-105, 108

error messages 105, 109

QDPMI.COM 106, 109-110

QDPMI.SYS 22, 104-106, 108-111

and QEMM386.SYS 109

QDPMI parameter 106, 109

- EXTCHKON 108
- EXTCHOFF 108
- KEEPSWAP 107
- KILLSWAP 104, 107
- KLSW 104, 107
- KPSW 107
- LOW 107
- MAXLOW 107-108
- MAXMEM 106-109
- MINMEM 106-108
- NOVM 107
- NOXCHK 108
- OFF 106
- ON 106
- OVLDIR 108, 110
- SWAP 105, 108
- SWAPFILE 105, 108
- SWAPSIZE 108
- SWSZ 108
- VM 107
- VMOFF 107
- VMON 107
- XCHK 108

QDPMI.COM

See QDPMI, QDPMI.COM

QDPMI.SWP file 104-105

See also Virtual memory, swapfile

QDPMI.SYS

See QDPMI, QDPMI.SYS

QDPMI.VM.OVL file 108, 110-111

QEMM

- feature list 9
- installing 17
- memory gains vs. DOS 6 131
- requirements 6
- unloading 20

QEMM.COM 53, 59, 91-101

QEMM.COM parameter

- ? 91
- AU 91
- AUTO 91
- HELP 91
- MAP 92, 94, 96, 98, 122-123
- ON 91
- RESET 92, 96-97

QEMM.COM reports 14, 66, 92

- Accessed 92, 96
- Analysis 92, 97-100, 123-124
- Memory 75, 92, 100
- Summary 92-93, 127
- Type 59, 92, 94, 122

QEMM386.SYS 22

- how to unload 119
- ordering of parameters 54, 79
- parameter files 14, 54

QEMM386.SYS parameter 25

- ? 58
- ADAPTERRAM 58, 80
- ADAPTERROM 59, 80
- ARAM 58, 80
- AROM 59, 80
- AU 59
- AUTO 59
- C386S 59
- CER 59
- CF 60, 120
- CHR 60
- COMPAQ386S 59
- COMPAQECAROM 59-60
- COMPAQFEATURES 60-61, 120
- COMPAQHAFROM 60
- COMPAQROMMEMORY 60
- CRM 60
- D4 62
- DB 61-62, 120, 125
- DBF 61
- DISKBUF 61-62, 120, 125
- DISKBUFFRAME 61
- DM 62
- DMA 62
- DOS4 62
- EMB 62, 125
- EMBMEM 62, 125
- EMS 56, 58, 62
- EXCLUDE 55-57, 59, 63, 65, 72, 74, 76, 80, 94, 99-100, 119, 121, 123-127, 130
- EXCLUDESTEALTHI 58, 63
- EXCLUDESTEALTHINT 58, 63
- EXT 59, 64, 101
- EXTMEM 59, 64, 101
- F10 64
- FASTINT10 64
- FB 66
- FEMS 65, 67
- FILL 64, 76, 120

# QEMM386.SYS parameter (continued)

FL 65, 67  
 FORCEEMS 65, 67  
 FORCESTEALTHCOPY 65, 98, 122, 126  
 FR 56, 58, 63, 65, 67, 127  
 FRAME 56, 58, 63, 65, 67, 127  
 FRAMEBUF 66  
 FRAMELENGTH 65, 67  
 FSTC 65, 98, 122, 126  
 GETSIZE 67  
 GS 67  
 HA 67  
 HANDLES 67  
 HELP 68  
 HMA 68, 79  
 HMAMIN 68  
 I 51, 55, 68, 80, 99-100, 119, 127  
 I386 68, 80  
 IB 68  
 IBMASIC 68  
 INCLUDE 51, 55, 68, 80, 99-100, 119, 123-124, 127  
 INCLUDE386 68, 80  
 IOTRAP 69  
 LABEL 69  
 LB 69  
 LD 69, 125  
 LOCKDMA 69, 125  
 MA 69, 98, 122, 126  
 MAPREBOOT 69, 120, 127  
 MAPS 69, 98, 122, 126  
 ME 59, 64, 70, 101  
 MEMORY 59, 64, 70, 101  
 MR 69, 120, 127  
 ODV 70  
 OF 59  
 OFF 59  
 OLDDV 70  
 ON 59, 98, 122, 126  
 PAUSE 71  
 PAUSEONERROR 71, 127  
 PE 71, 127  
 R 71  
 RAM 22, 56, 71, 80, 83, 91, 94, 98, 119, 123  
 REGION 71  
 RH 71, 120  
 ROM 56, 58, 71-72, 79, 95, 119, 126  
 ROMHOLES 71, 120  
 SH 72, 120, 125, 127  
 SHADOWRAM 72, 120, 125, 127  
 SORT 72  
 ST 44, 57, 65-66, 73, 79, 98, 122, 126

# QEMM386.SYS parameter (continued)

STEALTHROM 44, 57, 65-66, 73, 79, 98, 122, 126  
 SUS 20, 73, 127  
 SUSPENDRESUME 20, 73, 127  
 T8 74, 127  
 TA 73, 127  
 TASKS 73, 127  
 TM 74, 120, 125  
 TOKENRING 73, 120  
 TOPMEMORY 74, 120, 125  
 TR 73, 120  
 TRAP8042 74, 127  
 UFP 74  
 UNMAPFREEPAGES 74  
 UNUSUALEX 75  
 UR 75  
 USERAM 75  
 USEXMS 75  
 UX 75  
 VCPISHARE 77  
 VDS 75  
 VF 76  
 VHI 77  
 VIDEOFILL 76  
 VIDEORAM 76  
 VIDRAMEGA 49-50, 52, 76  
 VIDRAMEMS 49, 51, 77  
 VIRTUALHDIRO 77  
 VR 76  
 VREGA 49-50, 52, 76  
 VREMS 49, 51, 77  
 VS 77  
 VXDDIR 78  
 W3 78  
 WATCHDOG 78  
 WD 78  
 WINDOWS3 78  
 WINSHRINKUMBS 78  
 WSU 78  
 X 55-57, 59, 63, 65, 72, 74, 76, 80, 99-100, 119, 121,  
 125-127, 130  
 XBDA 79, 120  
 XMS 68, 79  
 XST 58, 63  
 XSTI 58, 63  
 QEMMFLOW.TEC  
 See Technotes, QEMMFLOW.TEC  
 QEXT.SYS 116  
 QSETUP 14, 19, 25, 27, 45, 104, 119  
 editing configuration files 26  
 Quarterdeck Technical Support 128

## QUICK

Optimize parameter 28, 34

## Quick BASIC

See QBASIC.EXE

## QUIET

LOADHI parameter 89

## QWINFIX 24

## R

### R

LOADHI parameter 22, 30, 87

## RAM

See also High RAM

QEMM386.SYS parameter 22, 56, 71, 80, 83, 91,  
94, 98, 119, 123

Ramnable memory addresses 94

Rational Systems 108

READ.ME file 15

Real mode 6-7

## Reboots

warm 19, 28, 30, 69, 74, 94, 125

when QEMM loads 72, 74

## REGION

LOADHI parameter 22, 30, 87

QEMM386.SYS parameter 71

## Region Layout

Optimize option 29

Region of High RAM 30, 84, 87

## REN

EMS programs parameter 115

## RENAME

EMS programs parameter 115

## RES

EMS programs parameter 115

LOADHI parameter 89

VIDRAM parameter 50-51

## RESET

QEMM.COM parameter 92, 96-97

## RESIDENT

VIDRAM parameter 50-51

## RESIDENTSIZE

LOADHI parameter 89

## RESIZE

EMS programs parameter 115

## RESPONSE

Optimize parameter 34, 89

Response file 34

## RF

Optimize parameter 34, 89

## RH

QEMM386.SYS parameter 71, 120

## ROM

See also ROM shadowing

QEMM386.SYS parameter 56, 58, 71-72, 79,  
95, 119, 126

searching for unused areas 11

ROM shadowing 12-13, 56, 60, 69, 72, 74, 79, 91,  
94, 127

See also ROM, QEMM386.SYS parameter

## ROMHOLES

QEMM386.SYS parameter 71, 120

## S

### S

LOADHI parameter 87

## SAVE

EMS programs parameter 116

## SCAT 72

## SCSI hard disk controllers

See Bus-mastering devices

## SEG

Optimize parameter 35

## SEGMENT

Optimize parameter 35

Self-high-loading programs 10

## SH

QEMM386.SYS parameter 72, 120, 125, 127

Shadow memory 13, 72, 75, 101

See also Shadow memory

See also ShadowRAM

ShadowRAM 13, 72, 100

QEMM386.SYS parameter 72, 120, 125, 127

## SHELL

LOADHI parameter 23, 38, 88

SHELL (DOS command) 23

## SIZE

LOADHI parameter 88

## SMALLEST

LOADHI parameter 87

## SORT

QEMM386.SYS parameter 72

## SPEED

EMS2EXT parameter 117

Split ROM 95

## SQF

LOADHI parameter 89

## SQT

LOADHI parameter 89

Squeeze feature 10, 30, 33-34, 89



SQUEEZEF  
     LOADHI parameter 89  
 SQUEEZET  
     LOADHI parameter 89  
 ST  
     Optimize parameter 25, 29, 35  
     QEMM386.SYS parameter 44, 57, 65-66, 73, 79, 98, 122, 126  
 ST-DBL.SYS 23  
 Stacker 61, 130  
 STACKER3.TEC  
     See Technotes, STACKER3.TEC  
 STACKS 37-38, 105  
 STEALTH  
     Optimize parameter 25, 29, 35, 43, 127  
 Stealth DoubleSpace 10, 17, 23, 26, 43, 45-46  
     enabling/disabling 45  
     ST-DBL.SYS 23  
 Stealth ROM 9-10, 19, 24, 28-30, 34-35, 43-45, 55, 57, 61, 63-66, 69, 71, 73-74, 77, 89, 91-94, 97-98, 119, 122-123, 125-127, 129  
     See also STEALTHROM, QEMM386.SYS parameter  
         disabling 44  
         enabling 43  
         frame method 43-44, 57  
         mapping method 43-44, 57  
 Stealth ROM testing 19, 25, 28-30, 34-35  
 STEALTH.TEC  
     See Technotes, STEALTH.TEC 129  
 STEALTHROM  
     See also Stealth ROM  
     QEMM386.SYS parameter 44, 57, 65-66, 73, 79, 98, 122, 126  
 STLTECH.TEC  
     See Technotes, STLTECH.TEC  
 STOR.TEC  
     See Technotes, SSTOR.TEC  
 STPASSDONE  
     Optimize parameter 35  
 SUBST.EXE 41  
 Summary  
     QEMM.COM report 92-93  
 Super PC-Kwik 87  
 Super VGA graphics 126  
 SuperStor 130  
 SUS  
     QEMM386.SYS parameter 20, 73, 127  
 Suspend/Resume 12, 20, 73, 127  
 SUSPENDRESUME  
     QEMM386.SYS parameter 20, 73, 127

SWAP  
     QDPMI parameter 105, 108  
 SWAPFILE  
     QDPMI parameter 105, 108  
 Swapfile  
     See Virtual memory, swapfile  
 SWAPSIZE  
     QDPMI parameter 108  
 SWSZ  
     QDPMI parameter 108  
 SYSTEM.INI 24, 26  
  
 T  
 T8  
     QEMM386.SYS parameter 74, 127  
 TA  
     QEMM386.SYS parameter 73, 127  
 TASKS  
     QEMM386.SYS parameter 73, 127  
 TCPIP.EXE 105  
 Technical Support 128  
 Technotes 126, 129  
     BUS-MAST.TEC 130  
     DBLDISK.TEC 130  
     DOS5.TEC 129  
     DRDOS6.TEC 129  
     EX13FLOW.TEC 130  
     EXCEPT13.TEC 130  
     EXCLUDE.TEC 130  
     MAXWINDO.TEC 129  
     MSDOS6.TEC 129  
     PARITY.TEC 130  
     QEMMFLOW.TEC 130  
     SSTOR.TEC 130  
     STACKER3.TEC 130  
     STEALTH.TEC 63, 125, 129  
     STLTECH.TEC 129  
     TROUBLE.TEC 130  
     WIN31.TEC 129  
     WINFLOW.TEC 125, 129  
     WINSIZE.TEC 129  
     XSTL.TEC 129  
     XTRADRV.TEC 130  
 TERMINATERESIDENT  
     LOADHI parameter 90  
 Text programs 20, 47, 51  
 TM  
     QEMM386.SYS parameter 74, 120, 125

Token-Ring network adapter 73

TOKENRING

QEMM386.SYS parameter 73, 120

Top memory 13, 74, 100

TOPCAT 13, 72

TOPMEMORY

QEMM386.SYS parameter 74, 120, 125

TR

QEMM386.SYS parameter 73, 120

TRAP8042

QEMM386.SYS parameter 74, 127

TROUBLE.TEC

See Technotes, TROUBLE.TEC

TSR

LOADHI parameter 90

TSRs 23, 85

Type

QEMM.COM report 92, 94

## U

UFP

QEMM386.SYS parameter 74

UMB 9, 11, 56, 145

definition 9

UMB-using programs 10

UMB-using programs 10

Unaccessed memory 91, 99, 123

Unassigned memory 101

UNIX 103

UNLINK

LOADHI parameter 88

UNLINKTOP

LOADHI parameter 88

Unloading QEMM 20

UNMAPFREEPAGES

QEMM386.SYS parameter 74

UNOPT.BAT 30

UNUSUAEXT

QEMM386.SYS parameter 75

Upper memory 9, 145

See also High RAM

definition 6

Upper Memory Block

See UMB

UR

QEMM386.SYS parameter 75

USERAM

QEMM386.SYS parameter 75

USEXMS

QEMM386.SYS parameter 75

## UX

QEMM386.SYS parameter 75

## V

VCPI 11, 62, 64, 70, 77, 101, 106, 114, 116, 124-125, 145-146

VCPISHARE

QEMM386.SYS parameter 77

VDISK.SYS 62, 64, 70, 101, 116

VDS 13, 61, 75, 125, 146

QEMM386.SYS parameter 75

VF

QEMM386.SYS parameter 76

VGA display 11, 20, 47-49, 51, 76-77, 79

VHI

QEMM386.SYS parameter 77

Video

problems 66, 126

speed of video operations 56, 60, 64

Video adapters 47-52, 64, 66, 74

Video memory 48, 50

Video memory area 11, 47-52, 76-77, 95

Video parameter tables 65, 71

VIDEOFILL

QEMM386.SYS parameter 76

VIDEORAM

QEMM386.SYS parameter 76

VIDRAM 11, 20, 47-52, 76-77, 79

loading VIDRAM high 50-51

and Microsoft Windows 12

VIDRAM parameter 50

EGA 50, 76

EMS 49, 51, 77

NOCGA 51

NOEGA 51

OF 50

OFF 49-51

ON 48-52, 76-77

OV 50

OVERRIDE 50

RES 50-51

RESIDENT 50-51

VIDRAM report 49

VIDRAMEGA

QEMM386.SYS parameter 49-50, 52, 76

VIDRAMEMS

QEMM386.SYS parameter 49, 51, 77

Virtual Control Program Interface

See VCPI

Virtual DMA Services  
     See VDS  
 Virtual memory 15, 103-104, 107, 109  
     swapfile 104-105, 107-109  
 Virtual-8086 mode 59, 91  
 VIRTUALHDIRQ  
     QEMM386.SYS parameter 77  
 VLSI chip set 13, 72  
 VM  
     QDPMI parameter 107  
 VMOFF  
     QDPMI parameter 107  
 VMON  
     QDPMI parameter 107  
 VP Planner 74  
 VR  
     QEMM386.SYS parameter 76  
 VREGA  
     QEMM386.SYS parameter 49-50, 52, 76  
 VREMS  
     QEMM386.SYS parameter 49, 51, 77  
 VS  
     QEMM386.SYS parameter 77  
  
 W  
 W3  
     QEMM386.SYS parameter 78  
 Warm boots  
     See Reboots, warm  
 WATCHDOG  
     QEMM386.SYS parameter 78  
 Watchdog timer hardware 78  
 WD  
     QEMM386.SYS parameter 78  
 What-If  
     Optimize option 29  
 WIN31.TEC  
     See Technotes, WIN31.TEC  
 WINDOWS3  
     QEMM386.SYS parameter 78  
 WINFLOW.TEC  
     See Technotes, WINFLOW.TEC  
 WINSHRINKUMBS  
     QEMM386.SYS parameter 78  
 WINSIZE.TEC  
     See Technotes, WINSIZE.TEC  
 WSU  
     QEMM386.SYS parameter 78

X  
 X  
     QEMM386.SYS parameter 55-57, 59, 63, 65, 72,  
     74, 76, 80, 99-100, 119, 121, 125-127, 130  
 XBDA 47, 79  
     QEMM386.SYS parameter 79, 120  
 XCHK  
     QDPMI parameter 108  
 XL  
     LOADHI parameter 87  
 XMS 7, 9, 11, 56, 61-62, 64, 70, 75, 79, 101, 113-114,  
     116, 145  
     See also EMB, Extended memory, HMA, UMB  
     QEMM386.SYS parameter 68, 79  
     XMS-using programs 62  
 XMSNET.EXE 24  
 XR  
     LOADHI parameter 87  
 XS  
     LOADHI parameter 87  
 XST  
     QEMM386.SYS parameter 58, 63  
 XSTI  
     QEMM386.SYS parameter 58, 63  
 XSTI.TEC  
     See Technotes, XSTI.TEC  
 XT computers 62  
 XtraDrive 130  
 XTRADRV.TEC  
     See Technotes, XTRADRV.TEC