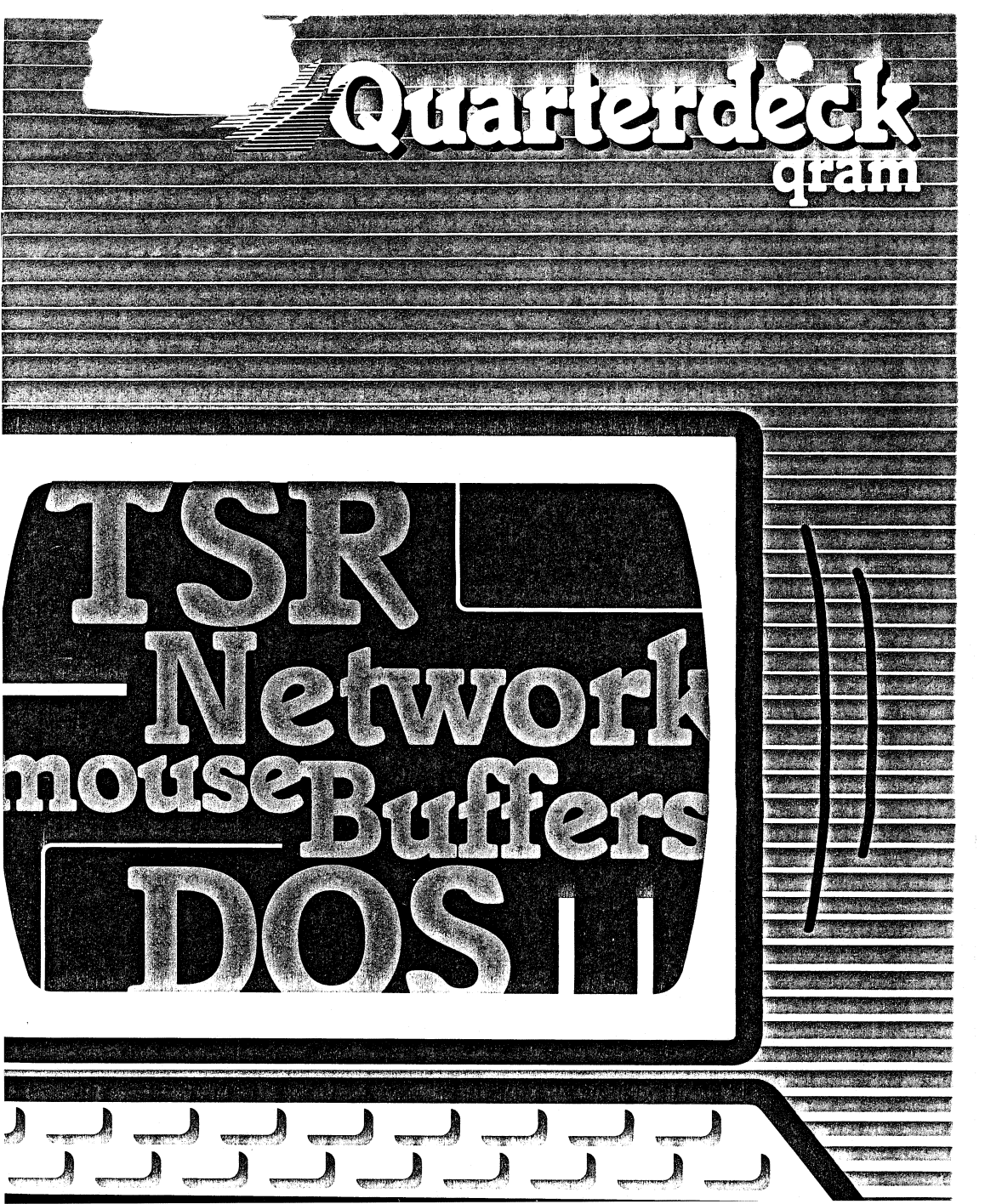

Quarterdeck
qram
TSR
Network
mouse
Buffers
DOS

)PYRIGHT

ADEMARKS

EDITS

graphy by Ventura Publisher Professional. Cover and
Page designed by The Design Works

LICENSE AGREEMENT

Carefully read the following terms and conditions. **Use of this product constitutes your acceptance of these terms and conditions and your agreement to abide by them.**

You, the original purchaser, are granted a nonexclusive personal license to use this Software under the terms stated n this Agreement. You may not assign or transfer the Software or this license to any other person without the express written consent of Quarterdeck Office Systems. Any attempt to sublicense, assign, or transfer any of the rights, duties, or obligations herunder is void. You may not copy, modify, alter, electronically transfer, or lease the Software or this manual. The license is effective until terminated. You may terminate it at any time by destroying the Software. It will also terminate if you fail to comply with any term or condition of this Agreement. You agree upon such termination to destroy the Software.

LIMITED 90-DAY WARRANTY
Covering the Physical Disks and Printed Materials

What is Covered?

Any physical defect in the physical software diskette(s) or in the printed materials.

For How Long?

Ninety (90) days from the date of purchase or from the date of your receipt of the package.

What We Will Do

Replace the defective diskette or printed materials.

What You Must Do

To replace defective diskettes or printed materials during the 90-day warranty period, you must send, postage paid, to:

QUARTERDECK OFFICE SYSTEMS
150 Pico Boulevard, Santa Monica, CA 90405
ATTN: Customer Support

the following information:

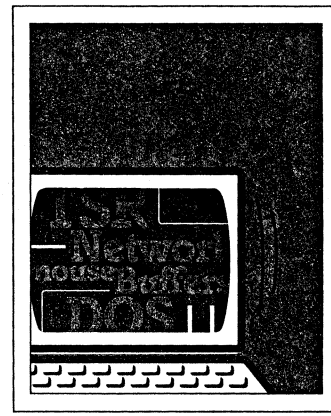
- ❑ Your name and address, including ZIP code.
- ❑ A dated proof of purchase—such as a copy of your receipt.
- ❑ Your Quarterdeck Product Registration Card—or, if your registration card is already on file with us, the serial number of your product.

Then:

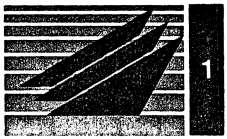
- ❑ If you're replacing a defective diskette, you must also send us the diskette with an explanation of the problem.

Continued on Inside Back Cover

Quarterdeck
qram



ontents



Introduction

QRAM is a memory enhancement tool for 8088, 8086 and 80286 PC's. QRAM enhances the utilization of your PC's memory by managing your PC's high memory, extended memory, and EGA/VGA video memory.

High memory (the memory addresses between 640K and 1024K) has been traditionally reserved by IBM for use by your system hardware. As DOS programs have become larger, and as you have need to run several programs at once, high memory has increased in importance. The reason is that there are often more memory addresses reserved for system hardware than are actually being used. So there are precious available memory addresses, usable by DOS, waiting to be used!

If you have expanded memory hardware (compatible with EMS 4 or EEMS expanded memory specifications), QRAM uses its mapping capabilities to fill unused memory addresses in your PC's high memory with expanded memory. This enables QRAM's LOADHI program to load TSR's, device drivers (such as networks), and DOS resources in this memory. If your PC has Chips & Technologies shadow RAM, QRAM detects this memory and makes it usable in the same ways it uses expanded memory.

QRAM is also an extended memory manager, compatible with the Microsoft XMS extended memory specification. The amount of extra memory QRAM can give you depends on your PC and any hardware add-ons you have.

If you need memory more than you need EGA or VGA graphics, QRAM can give you an additional 96K of memory that can be used by your DOS programs. It does that by making the memory reserved for your EGA or VGA adapter available to your DOS programs. The caveat is that while you are using this memory, you can't be doing graphics. But, QRAM makes it easy for you to use or not use this memory.

QRAM Features

Below is a summary of QRAM's features. QRAM:

- Is an extended memory manager, compatible with XMS. Together with QEXT.SYS, QRAM supports all three forms of the specification—high memory area (HMA), upper memory blocks (UMB), and extended memory blocks (EMB).
- Is a high memory manager when you have expanded memory or shadow RAM. Together with the LOADHI programs, QRAM enables you to run device drivers, TSR's and DOS resources (such as FILES, BUFFERS, etc.) specified in your PC's CONFIG.SYS and AUTOEXEC.BAT files, in high memory.
- Provides extended memory support (EMS2EXT and QEXT programs).
- Enables a user, willing to forego the enhanced graphics capabilities permitted by EGA and VGA adapter cards, to increase the memory used by DOS by 96K—thus making as much as 736K of memory, instead of the standard 640K, available to DOS. See QRAM's VIDRAM program.

Getting the Best Use of Memory

QRAM.SYS is a very powerful program. Because of this power, QRAM is very technical in nature. Knowing this, we've tried to make it as easy as possible for non-technical users to install and optimally configure their memory.

As a result, most users need only to know the command, INSTALL. QRAM's installation program looks at your PC and, if you have EMS 4 or EEMS expanded memory, sets whatever command line parameters it needs to map unused memory addresses between 640K and 1024K with expanded memory. Your high memory is now ready to be used to load TSR's, drivers, and DOS resources in high memory. So that you don't have to be a PC guru to load programs high, QRAM's Optimize program can automatically load TSRs, drivers, and DOS resources optimally into high memory for you.

If you are interested in seeing a map of the first megabyte of your PC's memory, we recommend that you refer to Quarterdeck's Manifest memory reporting program. Manifest's First Meg screens show you what programs are loaded in your PC's memory from 0K to 640K as well as the programs loaded in your PC's high memory. QRAM calls the areas of high memory in which programs are loaded high RAM.

About This Manual

QRAM is really a set of programs that help you get the best utilization of your memory. If you are a non-technical PC user, there is no need to read any further than Chapter 2, Installation. The remaining chapters of the manual are intended as references and should be used if you have a problem installing QRAM or if you wish to fine-tune QRAM or need the features supplied by the other programs. The QRAM manual is organized as follows:

- Chapter 2, Installation describes how to install QRAM and use the QRAM Optimize program.
- Chapter 3, The QRAM Programs lists the QRAM command line parameters used to set up your PC's memory.
- Chapter 4, The LOADHI Programs describes how to load TSR's and device drivers in high memory.
- Chapter 5, DOS Resources Programs describes the programs used to load DOS BUFFERS, FILES, FCBS, and LASTDRIVE in high memory.
- Chapter 6, The VIDRAM Program describes how to use the EGA/VGA video memory to run programs.
- Chapter 7, The QEXT program describes Quarterdeck's Extended Memory Manager providing an enhanced XMS support.
- Chapter 8, Other QRAM Programs describes EMS2EXT, which lets you specify how much memory to set as expanded and extended memory, and EMS, which lets a technical user manipulate EMS handles.

Installation

You install QRAM by running the QRAM installation program, INSTALL.

If you have a new PC, be sure to complete the standard installation and setup procedures of your PC before installing QRAM. If you have expanded memory you should check that the expanded memory manager is loaded in your CONFIG.SYS. An expanded memory manager characteristically has the letters EMM in its name.

QRAM is a device driver, which means that it must be loaded as part of the boot sequence of your PC. INSTALL automatically creates the CONFIG.SYS statement for QRAM and places it just after the statement that loads your EMM driver.

INSTALL also installs Manifest, Quarterdeck's comprehensive memory reporting and analysis program, if Manifest is bundled together with QRAM. Manifest describes your PC, specifically the software and hardware that use your memory. In doing so, it gives you an understanding of how best to use your PC's memory.

Installing QRAM

To install QRAM:

- **Place the QRAM diskette in drive A.**
- **Type** A:INSTALL **and press** ↵.

NOTE: If you have an LCD or Gas Plasma display, as on some laptops, we recommend that you run INSTALL in monochrome mode:

- **Type** A:INSTALL /m **and press** ↵.

Follow the instructions on the screen. INSTALL copies the QRAM files to your hard disk and modifies your CONFIG.SYS file.

That's all there is to installing QRAM. Since QRAM is a device driver, you do have to reboot your system for its settings to take effect. Most likely, your memory has been configured by INSTALL in the following ways:

- If you have EMS 4.0 or EEMS expanded memory, expanded memory is mapped into available memory addresses between 640K and 1024K and able to be used by QRAM's LOADHI programs.
- If you have Chips & Technology's shadow RAM, any available addresses between 640K and 1024K may be usable by the QRAM and LOADHI programs, without needing expanded memory mapped there.

To see whether you can use any of the available memory addresses between 640K and 1024K for LOADHI, refer to the LOADHI report, Chapter 4, page 14.

Also, if you are unfamiliar with memory, please refer to Quarterdeck's Manifest program. Manifest explains memory (first meg, expanded, extended). It describes the various expanded memory specifications. It also

gives you information about your DOS configuration and the hardware in your PC. Manifest also gives you a map of your first one megabyte of memory.

The last screen displayed by the QRAM installation program is a screen that tells you to reboot your PC to load QRAM and, if your high memory can be used by LOADHI, recommends that you then run the Optimize program.

If you aren't using large DOS programs and hence aren't suffering from RAM cram, you won't need to run Optimize. Also if you are a sophisticated user, you may wish to choose what is loaded high. In this case, please refer to Chapter 4, The LOADHI Programs, and Chapter 5, DOS Resource Programs. But first, you should give OPTIMIZE a chance. It can do an excellent job for you.

Optimize looks at your CONFIG.SYS and AUTOEXEC.BAT files to determine what can be loaded in high RAM using QRAM's LOADHI, and DOS resource programs. It then changes your CONFIG.SYS and AUTOEXEC.BAT files accordingly. By loading your TSR's, device drivers, and DOS resources in high memory, you have more memory below 640K available for programs.

After rebooting, if your PC has EEMS, EMS 4 expanded memory or shadow RAM, one or more address ranges between 640K and 1024K should now be available. Note that in addition, you can make up to 96K more memory available to your DOS program—if you have an EGA or VGA adapter and you don't need to use their graphic capabilities. See Chapter 6, the VIDRAM program.

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

- **Reset your system again. Use the power switch if necessary.**
- **Wait until you hear a beep, then hold down the Alt key.**

  QRAM pauses with the message, "QRAM: Press any key to continue, Esc to abort."

- **Press the Esc key.**

Your system then proceeds with the boot sequence. QRAM is not loaded and thus, no programs are loaded into high RAM. Your system is, however, usable. Refer to Chapter 3, The QRAM.SYS Program, for parameter changes needed on your system. See also Appendix A, Troubleshooting, for additional guidelines.

Optimizing Your Memory

The Optimize program scans your CONFIG.SYS and AUTOEXEC.BAT files to find device drivers, TSRs, and DOS resources that can be put in the high memory area. From its analysis, Optimize makes the changes to these files to make use of this memory for you. It also makes sure you have a memory manager installed that can turn the high memory area into high RAM.

For many, the use of Optimize during installation is all that is necessary to make immediate use of the high memory area. To run Optimize:

- Change to the subdirectory that you used to install QRAM.
- **Type** Optimize **and press** ↵.

If you choose not to run Optimize now, you can run it at any time in the future. In fact, you should run Optimize whenever you add device drivers, TSRs, change your DOS resources memory or change your high memory configuration in any way.

Optimize changes CONFIG.SYS and AUTOEXEC.BAT. Your original files are first copied to CONFIG.QDK and AUTOEXEC.QDK. Optimize also re-boots your system twice before it is done. When it completes, your system should have programs and data in high RAM. And, your application programs should have more memory available to them.

NOTE: If your AUTOEXEC.BAT file runs a program that is not a TSR (that is, it does not exit immediately on its own), then you will have to quit that program manually to allow Optimize to continue.

Optimize does not attempt to read or analyze any batch files it finds within your AUTOEXEC.BAT. Any TSRs loaded from within an embedded batch file will not be loaded into high RAM. If at all possible, consider putting the contents of these batch files directly into AUTOEXEC.BAT. Then re-run Optimize to gain even more memory.

Optimize signals its successful completing with the following message:

**** The OPTIMIZE process is complete ****

If you do not see this message, then your original CONFIG.SYS and AUTOEXEC.BAT files have been restored. This does not mean, however, that optimize did not identify some memory enhancements. The likely reason, is that there is a batch file embedded in your AUTOEXEC.BAT which prevents any further processing of the AUTOEXEC.BAT file. You should find this statement and remove it (or follow the suggestion in the paragraph above) and then rerun optimize.

In special situations, some of you will want to customize your system further. The reference sections in this manual are for that purpose. In particular, see the Chapter 4 The LOADHI Programs, and Chapter 5 The DOS Resource Programs. In any case, we recommend that you run Optimize now. The changes you will see in your start-up files will help you use these programs.

Running Optimize After Installation

To run Optimize at any time in the future, change to the subdirectory that you used to install QRAM and

■ **Type** Optimize **and press** ↵.

Optimize changes CONFIG.SYS and AUTOEXEC.BAT. Your original files are first copied to CONFIG.QDK and AUTOEXEC.QDK. Optimize also re-boots your system twice before it is done. When it completes, your system should have programs and data in high RAM. And, your application programs should have more memory available to them.

Note: Optimize is only appropriate if you have EEMS or EMS 4 expanded memory or Chips & Technologies Shadow RAM.

Providing XMS Support

The QRAM.SYS program provides access for Upper Memory Blocks (UMB) of XMS (Microsoft Extended Memory Specification). In order to provide support for the XMS features of Extended Memory Blocks (EMB) and the High Memory Area (HMA) you will also need to use QEXT.SYS. This program is provided with QRAM. You will want this extra support if you have a program which uses the HMA, such as DESQview or Microsoft Windows. Normally you will place a line with:

```
DEVICE=QEXT.SYS
```

in your CONFIG.SYS file.

If you currently have a line which loads HIMEM.SYS, you will want to remove that line, since the combination of QRAM and QEXT now provide all of the necessary services, possibly with lower memory overhead.

You will want to place the DEVICE=QEXT.SYS line AFTER the DEVICE=QRAM.SYS line, since the QEXT program can be loaded into high memory. You will need to have the first 64K of extended memory available for QEXT. This may require that you change the switch settings on your expanded memory card to provide some extended memory at 1024K. The first 64K of extended memory will be used for the HMA. If you have other programs which use extended memory, the DEVICE=QEXT.SYS line must be the very first statement that will request extended memory.

The QRAM.SYS Program

This chapter is the reference guide for the command line parameters used with QRAM.SYS. As installed, the statement in your CONFIG.SYS file, should look like this:

```
DEVICE=QRAM.SYS
```

This default specification should make as much of the high memory area available as is possible. You should not have to use the commands in this chapter unless you want to fine-tune your memory configuration or you are experiencing problems.

The function of QRAM is to create high RAM. To do this it relies upon the presence of either expanded memory hardware, shadow RAM, or both.

In creating high RAM from expanded memory, it must use an expanded memory manager (EMM). It is the EMM which identifies areas of memory which can be mapped. QRAM, with the support of the memory manager, allocates and maps these usable areas in high memory as high RAM. The LOADHI and DOS resource programs can then be used to make use of these memory areas.

Shadow RAM is memory already present in the high memory area. Many 80286 and 80386 machines are delivered with one megabyte of memory. On some, the extra 384K (over and above the 640K conventional memory) is present as extended memory. On PCs with the Chips & Technologies 386 chip set or NEAT chip set this 384K of memory is present in the high memory area and is called shadow RAM. QRAM can create high RAM from unused address ranges of shadow RAM.

It is extremely important to realize that QRAM can only add value to your existing hardware and software memory capabilities. It cannot perform miracles. It can and does build upon what you have. In looking over the QRAM parameters, remember this partnership. First of all, consider whether a change you are thinking of making to the QRAM command line, might be more appropriately provided by the expanded memory manager. And secondly, remember that the function of a QRAM parameter is only meaningful if the underlying hardware and the memory manager support the feature or control the memory affected.

QRAM.SYS Parameters

To use a QRAM.SYS command line parameter you type the parameter name on the same line as DEVICE=QRAM.SYS in your CONFIG.SYS file. As an example CONFIG.SYS entry, a QRAM.SYS command line with the FRAMELENGTH=0 parameter set, looks like:

```
DEVICE=QRAM.SYS FRAMELENGTH=0
```

Several QRAM.SYS parameters require or optionally accept an address or address range. The INCLUDE parameter, for instance might be written:

```
INCLUDE=xxxx-yyyy
```

Notice that there are no spaces before or after the equals sign or before or after the dash separating the address range. In this example, and throughout this section, **xxxx** and **yyyy** are hexadecimal numbers which specify an address (**xxxx** alone) or address range (**xxxx-yyyy**). When you specify an address or range in a QRAM parameter, you should take care to use only those addresses in high memory which have been mapped (or identified as mappable). Your address specifications must also be specified in multiples of 16K, which is the page size for expanded memory and shadow RAM.

You may specify more than one parameter on the command line. With some parameters you also may use an abbreviation instead of the full parameter name. The following paragraphs describe each parameter. The abbreviation for a parameter, if it has one, is shown in parentheses after the full parameter name.

- **RAM or RAM=xxxx-yyyy** specifies that QRAM should "fill in" areas of memory above 640K and below 1024K which your expanded memory manager has identified as usable (mappable). Specifying QRAM with RAM by itself is equivalent to the default (without any parameters—as shown above). That is, all mappable areas will become high RAM. Use the RAM parameter with an address range when you do ***not*** want all of the high memory to be made into high RAM.

- **INCLUDE=xxxx-yyyy, (I)**, specifies a mappable area of memory QRAM should use.

- **EXCLUDE or EXCLUDE=xxxx-yyyy, (X),** specifies an area of memory in the first megabyte which should ***not*** be under the control of QRAM.SYS. EXCLUDE without a range will not fill any areas of high memory. Otherwise, you only need to specify an excluded region if QRAM.SYS fills an area it should not. For example, if you have a network adapter that uses 16K of memory at address CC00, then you could use EXCLUDE=CC00-CFFF if you find that QRAM has mapped that area. However, this is the type of problem where it would be better to use your expanded memory manager's "exclude" capability if it has one.

  NOTE: You may specify multiple ranges to INCLUDE or EXCLUDE by using the parameters several times on the QRAM.SYS command line. Also, parameters are processed sequentially, so it is possible to EXCLUDE an area and then INCLUDE a part of it later on the command line.

- **FORCEEMS, (FEMS),** instructs QRAM to allow EMS memory requests to be honored. Use this parameter only if you have used the FRAMELENGTH parameter (see below) with a value less than four. This will allow programs limited access to expanded memory even without a full page frame. Be aware that some programs which use expanded memory may not work with a partial or missing page frame if you use this option.

- **FRAMELENGTH=x, (FL),** instructs QRAM to assume a page frame containing **x** pages, where **x** is a number from 0 to 4. Setting **x** to zero is equivalent to not having a page frame at all. Setting **x** to four is equivalent to the EMS standard. Normally a page frame is established by the expanded memory manager and consists of four 16K pages. With this parameter, QRAM allows you to free up one or more of these pages for use as high RAM. Use this option with FORCEEMS if you have programs which can make use of expanded memory in the absence of a page frame, or with a partial page frame. See FORCEEMS above.

- **NOFILL, (NO),** specifies that QRAM.SYS should not fill conventional memory below 640K. This option is only meaningful if your machine has less than 640K of conventional memory. If it does, QRAM normally fills in this missing memory, up to 640K from expanded memory. NOFILL specifies not to do this. NOFILL also prevents video filling (see NOVIDEOFILL).

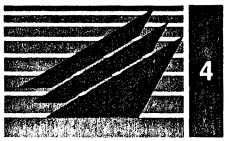
- **NOVIDEOFILL, (NV),** specifies that the memory area from A000 to B7FF is ***not*** to be filled with memory. This option is only meaningful on PCs with a monochrome or a CGA adapter. On these systems QRAM will increase DOS' memory from 640K to 704K (mono) or 736K (CGA). Occassionally, extending conventional memory in this way causes problems. This parameter disables this feature.

- **NOXBDA, (NX),** specifies that QRAM should not move the extended BIOS data area. On machines that have one, QRAM.SYS automatically moves the eXtended BIOS Data Area (XBDA), which is normally at 639K, into another address space. Doing so usually enables other memory management features, such as extending the memory available to DOS. It is possible, however, for a program to expect that the XBDA is at 639K. If you believe your system has an XBDA, and are encountering difficulty with a program, the NX parameter may remedy the difficulty.

- **NOSHADOWRAM, (NOSH),** instructs QRAM to not use Shadow RAM. QRAM automatically detects shadow RAM and uses it to create high RAM. Use this parameter if QRAM is failing and you have shadow RAM which may be the source of the problem.

- **NOPAUSEONERROR, (NOPE),** doesn't pause on an error. QRAM usually displays a "Press any key to continue, ESC to abort" message after parsing command line instructions if there are parameter errors. This instructs QRAM.SYS to not display the message. When an instruction causes QRAM.SYS to generate the above error, and when the error generated has been shown to not cause any disruption of the system, use the NOPE instruction

- **PAUSE** tells QRAM.SYS to pause the display when outputting messages. The PAUSE command permits you to hit the "Esc" key to stop QRAM.SYS from installing. This is useful when trying out new parameters, so you can abort the load if there was an error. Note that if there is an error on the QRAM.SYS command line, QRAM will automatically pause to show you the source of the error.
- **HELP** lists each QRAM.SYS parameter with a one line description.
- **?** lists each QRAM.SYS parameter and its abbreviation if it has one.



The LOADHI Programs

The two LOADHI programs, (LOADHI.COM and LOADHI.SYS) enable you to load TSRs, drivers, and DOS resources into available regions of high RAM. Your high memory is transformed into high RAM by QRAM.SYS by default, if you have EMS 4 or EEMS expanded memory or shadow RAM. If you have run the Optimize program, the LOADHI programs may have already been used to put programs and data in high RAM for you.

This chapter explains how and when the LOADHI programs may be used, and describes the command line switches available. These switches help you determine and modify the effectiveness of LOADHI 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've been doing without,
- speed up memory-starved programs that no longer need to go to disk for their data,
- add DOS resources to improve system response time and program effectiveness, and
- increase the memory available to applications running in DESQview.

The LOADHI programs are very effective with or without customization by you. However, it is important to note that if LOADHI is unable to make use of high RAM for a particular program or a DOS resource, it will use conventional memory instead. Thus, your programs and DOS resources are available for use whether they were relocated or not.

In order to use these two programs effectively, you need high RAM. In this chapter, it is assumed that QRAM has mapped memory into available memory addresses. If this is not the case, you may wish to install and setup QRAM as described in Chapter 2, Installation.

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 either from COMMAND.COM'S command line or from within your AUTOEXEC.BAT file. Both LOADHI programs support the same set of command line options (switches) which let you alter the way LOADHI normally allocates and uses high RAM. These switches are described more fully on page 18, LOADHI Options.

In addition, LOADHI.COM can be executed from the command line at any time to give you a listing of your current usage of high RAM. The following section describes this listing. It defines key terminology used in this chapter. And it is also the starting point for you, if and when you need to employ LOADHI's command line options.

The LOADHI Report

You display the LOADHI report by running LOADHI from the command line without specifying any command line option or program file to load:

Type LOADHI **and press** ↵.

The LOADHI report, shown below, describes what high RAM is in use and still available, the amount of high RAM used, and the names of the programs or DOS resources using the memory. It also shows you the memory address range in use by each block within a region.

As described in Chapter 2, QRAM locates and maps high memory areas into high RAM. These memory areas may be scattered throughout the high memory area due to the presence of BIOS and system ROM and RAM memory already in use by peripheral adapters. Each contiguous area of memory converted into high RAM is referred to as a ***region***. Each region is given a number. Memory is allocated from a region in ***blocks***. These regions may vary in size. Each time you load something into high RAM, LOADHI allocates a block of memory from one of these regions. That block shows up in the list for that region and the region's available size and memory area are reduced accordingly.

However often you call upon LOADHI to populate high RAM, there will never be more or less regions available (barring a fundamental change in your hardware, such as adding or removing a peripheral adapter card). The number of blocks listed for a given region will, however, increase as LOADHI uses this memory.

If you display the LOADHI report before you have loaded TSR's or drivers in high RAM, you can see the number of regions found and mapped, the high memory address ranges, and the size of each region. Each region is marked as Available in the status column. This report serves as an excellent baseline for planning the best use of high RAM. The sum total of the regions' sizes gives you the upper limit to the conventional memory that can be saved by using LOADHI.

```
C>loadhi

Region      Area        Size     Status
  1     B000 - B006     0.1K   Used (IPX)
  1     B007 - B00C     2.0K   Used (FILES)
  1     B00D - B4AF      15K   Used (IPX)
  1     B4B0 - B7FE      13K   Available
  2     C400 - C533     4.8K   Used (CACHE)
  2     C534 - C53A     0.1K   Used (MOUSE)
  2     C53B - C66A     4.8K   Used (BUFFERS)
  2     C66B - C8D6     9.7K   Used (MOUSE)
  2     C8D7 - D226      37K   Used (NET3)
  2     D227 - DFFF      55K   Available

C>_
```

LOADHI.SYS

You use the LOADHI.SYS program when you want to load a device driver into high RAM. Device drivers are essentially extensions to DOS which DOS will load on system startup as it processes your CONFIG.SYS file.

Any statement in your CONFIG.SYS file which begins with the keyword DEVICE= instructs DOS to load the device driver which is specified following that keyword. Some common devices are memory managers, add-on peripheral device drivers, such as a disk drive or a mouse, and extensions to existing devices, such as the ANSI.SYS driver supplied with DOS that some programs use to address the video display.

With the exception of QRAM.SYS, which LOADHI relies upon, any device driver that you currently use or anticipate using in the future is a candidate to consider loading into high RAM. Occasionally you may find a device driver that is sensitive to its location in memory and which will not work properly, if at all, when relocated. Other device drivers may require that you experiment with the customization capabilities of LOADHI. However, for the most part you will encounter little difficulty in using LOADHI.SYS with your device drivers.

Since ANSI.SYS is available on every DOS system, we use it in the example below to show how you take advantage of LOADHI.SYS In this example, the sample CONFIG.SYS file has a statement which loads QRAM.SYS, as well as the following statement loading ANSI.SYS:

DEVICE=C:\DOS\ANSI.SYS

This statement instructs DOS to load the file ANSI.SYS which can be found in the DOS subdirectory of disk drive C.

To load ANSI.SYS in high memory, you must instruct DOS to load the file LOADHI.SYS instead of ANSI.SYS. LOADHI will take care of loading the ANSI.SYS file. You do this by modifying the CONFIG.SYS statement so that it reads:

DEVICE=C:\QRAM\LOADHI.SYS C:\DOS\ANSI.SYS

NOTE: Editing an existing DEVICE= statement simply displaces the device driver specification on the statement line. If the device driver you want to relocate takes command line switches of its own (the example ANSI.SYS has none), and you make use of any of these options, they should be retained in the edited statement also.

This easy modification to your CONFIG.SYS file represents the simplest way to take advantage of LOADHI's features. For many systems, this will be adequate. When you need more control over the placement of device drivers or encounter difficulty in relocating a device driver, you should see the LOADHI switches section of this chapter.

LOADHI.COM

You use LOADHI.COM to load TSR (Terminate and Stay Resident) programs in high memory or to relocate DOS resources (such as disk buffers and file handles) to high memory. The TSRs you customarily use are probably found in your AUTOEXEC.BAT file. DOS resources are allocated, if at all, from your CONFIG.SYS file. Other TSRs which you use only occassionally may be loaded directly from the DOS prompt (or by way of a batch file).

This section first explains how to use LOADHI.COM to load your TSRs into high RAM. It then briefly describes how you may use LOADHI.COM in conjunction with QRAM's programs (BUFFERS.COM, FILES.COM, etc.) that place DOS resources in high RAM. Since these programs also perform other functions, they are more fully described in Chapter 5, The DOS Resource Programs.

In describing how to use LOADHI.COM, we use, as an example TSR, a program called DOITALL. This imaginary program resides in a disk directory called UTILS and has several command line options which modify its behavior. Our example shows two of these options: the /m option to force the program to display in monochrome and the /r option which instructs it to stay resident (it has a stand-alone mode) and available to be "popped up" on demand. To run DOITALL, the following statement needs to be executed:

C:\UTILS\DOITALL /m /r

This statement could be present in your AUTOEXEC.BAT file, in some other batch file you execute as needed, or a statement you enter from the DOS prompt whenever you want the program in memory. To run DOITALL out of high RAM, the above statement needs to be changed so that it executes LOADHI first:

C:\QRAM\LOADHI C:\UTILS\DOITALL /m /r

Notice that the statement that executed the program is now a parameter to the program LOADHI. All that we have done is insert the drive and directory path for LOADHI and made LOADHI the program to be loaded. LOADHI in turn loads and executes DOITALL in high RAM. DOITALL still sees its parameters when it initializes. In many cases this simple change to your system is sufficient to free up a significant amount of memory and still enjoy the benefits of your TSR programs.

There are also several programs included with QRAM which are meant to be used in conjunction with LOADHI.COM. These programs are BUFFERS.COM, FILES.COM, FCBS.COM and LASTDRIVE.COM. As their names suggest, they allow you to allocate and use the associated DOS resources out of high RAM. If your CONFIG.SYS file is currently allocating any of these resources, the use of LOADHI and these utilities frees up much of the conventional memory now devoted to them.

Each of these programs allocate memory before they terminate. If they are in high RAM, the memory they allocate comes from high RAM. Although these programs are fully described in the Chapter 5, The DOS Resource Programs, a

typical example of the use of each one with LOADHI should give you an idea about their use. These examples do omit the details about the disk and sub-directory location of the programs.

Example 1-Allocate buffers in high memory:

```
Before LOADHI.COM:
    CONFIG.SYS file:      BUFFERS=20
    AUTOEXEC.BAT file :
After LOADHI.COM:
    CONFIG.SYS file:      BUFFERS=1
    AUTOEXEC.BAT file:    LOADHI BUFFERS=20
```

Example 2-Add 10 file handles:

```
Before LOADHI.COM:
    CONFIG.SYS file:      FILES=20
    AUTOEXEC.BAT file:
After LOADHI.COM:
    CONFIG.SYS file:      FILES=10
    AUTOEXEC.BAT file:    LOADHI FILES +10
```

Example 3-Load FCBS in high memory:

```
   Before LOADHI.COM:
CONFIG.SYS file: FCBS=8,2
    AUTOEXEC.BAT file:
After LOADHI.COM:
    CONFIG.SYS file:      FCBS=1,0
    AUTOEXEC.BAT file:    LOADHI FCBS=8,2
```

Example 4-Load last drive in high memory:

```
Before  LOADHI.COM:
    CONFIG.SYS file:      LASTDRIVE=G
    AUTOEXEC.BAT file:
After LOADHI.COM:
    CONFIG.SYS file:      LASTDRIVE=D
    AUTOEXEC.BAT file:    LOADHI LASTDRIVE=G
```

LOADHI Options

There are several command line options for LOADHI which can help you make better use of your high RAM if you find that the simple, straightforward approach is inadequate.

The command line options may help if you have trouble loading one or more of your programs into high RAM. Some command line options are to be used if you have more candidates for high RAM than seem to fit. Another command line option lets you specify a particular region to use or exclude. There is an option that will allow you to find out how much memory a program really needs, when that isn't clear either from the program documentation, or from a guess-timate based on its file size.

The LOADHI command line has the following format:

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 other program you are loading. The last item shows the placement of options for the target program if there are any.

Now that we know where LOADHI's options belong on the command line, we also need to know what an option looks like.

- A LOADHI option always begins with a 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.

One LOADHI option is LARGEST, which optionally can take a value. You can indicate this option in any of the following ways:

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

In all, there are thirteen options you may specify on a LOADHI command line, each of which are described below. Combinations of two or more options may be specified to give you greater control over the use of high RAM. Each paragraph begins with the full option name followed by its abbreviation. If a value is called for that is optional, it is enclosed in square brackets ([ ]). The options are:

- **/BESTFIT** or **/B** instructs LOADHI to use the smallest block of memory in which the program fits. Using BESTFIT tends to reserve larger regions for your larger programs. See also SIZE and GETSIZE.
- **/HAPPIEST** or **/H** instructs LOADHI to use the smallest block of memory in which the program will fit, provided that it does not terminate with an

error. 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** or **/R:n** instructs LOADHI to load the program into the region numbered "n".

- **/LARGEST[:n]** or **/L[:n]** instructs LOADHI to load the program into the largest block or one particular block out of several large blocks available. The number indicated by n indicates which of these to use. For instance the option /L:2 specifies the second largest block.

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

- **/EXCLUDEREGION:n** or **/XR:n** instructs LOADHI to not use region number n to load the target program.

- **/EXCLUDELARGEST[:n]** or **/XL[:n]** instructs LOADHI to not use the largest (or the nth largest block) to load the target program.

- **/EXCLUDESMALLEST[:n]** or **/XS[:n]** instructs LOADHI to not use the smallest (or the nth smallest) block to load the target program.

- **/GETSIZE[:f]** or **/GS[:f]** allows you to determine precisely the amount of memory a program requires. With this option LOADHI loads the program you specify and reports two memory usage values. The first value tells you how much memory (in bytes) the program required to load and initialize. The second value tells you the amount of memory the program has permanently retained for its own use. You use this option to help you custom fit your device drivers and TSRs into high RAM.

  The optional value *f* should be a file name. When a file name is present, LOADHI writes the program name and the two size values into this file so that you may examine it later. If the file name you specify already exists, LOADHI appends the information to this file. By systematically using the GETSIZE option with a file name 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.

  The GETSIZE option always causes LOADHI to load programs into conventional memory, not high RAM. Thus you will use it only as a preliminary step in customizing your system.

  NOTE to DESQview users: LOADHI /GETSIZE can be used to help you determine how much memory a program requires to run, so you can properly set an appropriate memory size. Simply run the program you are interested in in a maximum memory window with LOADHI /GS. Do

what you would normally do in the program. When you exit it, LOADHI displays the amount of memory that the program used. That number (increased slightly) can be used for the memory configuration for that program.

- **/SIZE:nnnn** or **/SIZE:nnnnK** instructs LOADHI to allocate from a block that will best fit the value **nnnn**. This number can be expressed in bytes (i.e 4096) or in kilobytes (i.e. 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 must represent the amount of memory the program needs to successfully initialize.

- **/NOLO** or **/NL** instructs LOADHI to not load the specified program at all if it will not fit in a high RAM region. This allows you to specify additional drivers, TSRs or DOS resources, that you would like to use, but only if they will be placed in high RAM.

- **/LO** instructs LOADHI to unconditionally use conventional memory instead of high RAM. You can use this option to temporarily change LOADHI statements in CONFIG.SYS or AUTOEXEC.BAT files without removing them completely.

- **/TERMINATERESIDENT** or **/TSR** instructs 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 4's INSTALL command in CONFIG.SYS. In this case, the effect of this option is simply to suppress an inaccurate error message issued by DOS indicating a failure to load a TSR when in fact LOADHI successfully relocated the program.

Finally, as a quick reminder, LOADHI will also respond to the following options on its command line:

- **/NOPAUSEONERROR** or **NOPE** instructs LOADHI not to pause on error.

- **/PAUSE** instructs LOADHI to pause while parsing commands.

- **/HELP** displays the LOADHI help screen.

- **/?** lists all LOADHI command switches.

The DOS Resource Programs

The DOS Resource Programs let you manage the data structures used by DOS. You may use these programs by themselves, either to determine the current memory used or to add resources out of conventional memory. Their use with the LOADHI programs will benefit you the most.

DOS always allocates memory for each resource. The amount allocated varies according to the DOS version. Earlier versions of DOS do not have support for some of the resources. Users increase the memory for a resource by including a statement in their CONFIG.SYS file. DOS reads and processes this file during its boot sequence.

This chapter discusses each of the DOS resource allocation programs. Each DOS resource is described briefly. Changes or additions to your CONFIG.SYS and AUTOEXEC.BAT files are given where needed. This information should also help those of you who are taking advantage of increasing these DOS resources for the first time.

Using the DOS Resource Programs

The programs described in this chapter are all used in much the same way. Running the program without any parameters gives you a report on the current memory allocation for the resource. All programs accept a numeric parameter which increases the memory available to DOS for that resource. And each allocates this additional memory from high RAM when you use LOADHI.

To use high RAM effectively, you change CONFIG.SYS to reduce the memory allocation. You then change AUTOEXEC.BAT to increase the resource.

The parameters these programs use share a common syntax. A number, or a number preceded by an equals sign (*nn* or *=nn*), gives the total number of data structures needed. A number preceded by a plus sign (*+nn*), adds that many more structures. All the programs add memory; you cannot use them to reduce the memory allocated to a resource.

BUFFERS.COM

The DOS buffers resource can improve disk I/O response times. DOS always has some number of disk sector buffers allocated. The default number varies, according to the version of DOS, from 2 to 15 buffers. You can add more buffers with a statement in your CONFIG.SYS file. For example, the statement, BUFFERS=30, allocates space for a total of thirty sector buffers. Each sector buffer requires 528 bytes of memory.

NOTE: Don't use BUFFERS.COM in DOS 4.0 systems. The structure of the disk buffers is different.

Without the services of BUFFERS with LOADHI, buffers are allocated out of conventional memory. Adding buffers can improve program response time but reduces the memory available to your applications. Also, one application may benefit from sector buffering while another is unaffected. Much depends upon how they access the files they use. Having your disk buffers allocated out of high RAM can, therefore, keep the benefits of disk buffering and avoid the associated penalty.

You can display a report of the number of buffers now allocated:

■ **Type** BUFFERS **and press** ↵.

The program responds by reporting " BUFFERS=30 now".

To add 30 buffers more in High RAM:

■ **Type** LOADHI BUFFERS +30 **and press** ↵.

The program responds by reporting:

BUFFERS=30 before
30 buffers added
BUFFERS=60 now

If you have a BUFFERS= statement in your CONFIG.SYS file, we recommend loading your buffers in high memory. Let's assume for our example that you have previously determined that you require 40 disk buffers, so CONFIG.SYS file now contains the statement, BUFFERS=40. To load these in high memory:

■ **First change the BUFFERS statement in your CONFIG.SYS to** BUFFERS=1.

This specifies a small number of buffers to override any default DOS allocation. If you simply remove this statement from CONFIG.SYS you may end up with as many as fifteen disk buffers allocated out of conventional memory.

■ **Next change your AUTOEXEC.BAT file so the first statement is** LOADHI BUFFERS=40 (or LOADHI BUFFERS +39).

NOTE: Make it the first statement so other AUTOEXEC.BAT statements can take advantage of disk buffering.

FILES.COM

DOS uses the files resource to keep track of disk files while they are open. The files resource is a data structure requiring about 53 bytes for each open file. This data structure is closely related to a DOS file handle, so that term is used here.

The default number of file handles DOS allocates is small and seldom enough. Many of the larger applications require more. A program such as DESQview, which allows multiple programs to be running concurrently, requires at least 20. Depending on the active applications, DESQview may require even more. It is quite likely that you already have a FILES= statement in your CONFIG.SYS file. FILES=20 is a common setting.

You can display a report of the number of files now allocated:

■ **Type** FILES **and press** ↵.

The program responds by reporting "FILES=20 now".

To add 20 files in high memory:

■ **Type** LOADHI FILES +20 **and press** ↵.

The program responds by reporting:

FILES=20 before
20 files added
FILES=40 now

Whether you do or do not currently have a FILES= statement in your CONFIG.SYS file, we recommend loading files in high memory. Let's assume for our example that you have previously determined that you require 40 files, so your CONFIG.SYS file now contains the statement, FILES=40. To load FILES in high memory:

■ **Change the FILES statement in your CONFIG.SYS file to** FILES=10.

This statement allocates 10 file handles. This number puts only a small burden on the use of conventional memory, while avoiding problems with certain applications.

■ **Add the statement,** LOADHI FILES=40 **in your AUTOEXEC.BAT file.**

FCBS.COM

DOS uses the FCB resource to keep track of File Control Blocks (FCBs). Versions of DOS prior to DOS 2.0 used FCBs exclusively to open and manage open disk files. Since DOS 2.0, the preferred method of file access involves the use of file handles. Because many programs written before DOS 2.0 are still in use, the use of FCBs is still supported.

Skip this section if you are not using SHARE and do not have an FCBS statement in your CONFIG.SYS file. Also, this resource is not available in DOS version 2 systems.

The FCB management by DOS involves two numbers. The first number instructs DOS to allocate memory for that many FCBs. The second number indicates how many of those FCBs should be protected when DOS needs to close an open FCB. Each FCB resource allocated requires about 53 bytes.

NOTE: The FCB resource uses contiguous memory. When you add FCBs, a new block of memory for the entire table must be allocated. The memory used by the original table is not recovered.

You can display a report of the number of files now allocated:

■ **Type** FCBS **and press** ↵.

The program responds by reporting "FCBS=4,0 now".

To add 4 more FCBs, with 2 additional FCBs protected.

■ **Type** LOADHI FCBS +4,2 **and press** ↵.

The program responds by reporting:

FCBS=4,0 before
4 FCBs added
2 protected FCBs added
FCBS=8,2 now

If you currently have a FCBS= statement in your CONFIG.SYS file, we recommend loading FCBS in high memory. Let's assume for our example that you have previously determined that you require 8,2 FCBS, so your CONFIG.SYS file now contains the statement, FCBS=8,2. To load FCBS in high memory:

- **First change the** FCBS= **statement in your CONFIG.SYS to** FCBS=1,0.
- **Add the statement,** LOADHI FCBS 8,2, **to your AUTOEXEC.BAT file.**

LASTDRIV.COM

DOS uses the last drive resource to support both physical and logical disk drives. Logical disk drives are useful if you use the DOS SUBST program. Each drive entry in DOS' drive table requires about 80 bytes.

You can add drive table support using the numerical parameters you're seen throughout this section or you can set it using a drive letter.

NOTE: The DOS drive table uses contiguous memory. When you add more drives, a new block of memory for the entire table must be allocated. The memory used for the original drive table is not recovered.

You can display a report of the number of files now allocated:

- **Type** LASTDRIV **and press** ↵.

The program responds by reporting:

LASTDRIVE=E now

To load a new drive table in high RAM:

- **Type** LOADHI LASTDRIV G **and press** ↵.

The program responds by reporting:

LASTDRIVE=E before
2 drives added
LASTDRIVE=G now

We recommend that you change your LASTDRIVE= statement in CONFIG.SYS so only your actual disk volumes are specified (For example, LASTDRIVE=D). If your hard disk is partitioned, the drive letter should correspond to the last partition. This minimizes the "lost" memory. Then load logical drives in high memory. For example to add three logical drives:

- **Add the statement,** LOADHI LASTDRIV+3, **to your AUTOEXEC.BAT file.**

The VIDRAM Program

The VIDRAM.COM program extends the conventional memory managed by DOS by as much as 96K if your system has an EGA or VGA video adapter card. Your PC must also already have a full 640K of conventional memory and the program or programs you intend to execute with this additional memory must not use graphics while VIDRAM is enabled. In other words, VIDRAM can be of use to you if you make use of large text based programs, such as dBASE IV, that will be more responsive in a 736K machine.

VIDRAM is a standalone TSR. It does not need any expanded memory, extended memory or a memory manager (other than itself and DOS). If your system has the necessary video RAM, VIDRAM will work equally well on 8088, 8086, 80286 and 80386 machines. If you do have additional memory and a memory manager that can make this memory available to you, VIDRAM may not be as useful to you.

VIDRAM steals the video memory from the adapter RAM that is used for graphic pages. This memory is available on EGA and VGA adapters, (not on CGA or monochrome adapters) and lies just beyond the 640K address boundary. VIDRAM makes this memory available to DOS. DOS in turn will make the memory available to every program it loads. VIDRAM also intercepts video requests and refuses all requests that would make use of this memory, i.e. graphics operations.

If you routinely use both large text based programs and graphics programs, you can turn the VIDRAM feature off and on as needed after you exit one program and before you execute the other. (NOTE: VIDRAM cannot be turned on and off inside DESQview).

Using VIDRAM

If you seldom use graphics programs you can make the best use of VIDRAM if you simply load it into memory and turn it **ON** every time you power up your machine. To do this you need to create or modify your AUTOEXEC.BAT file, adding the statement:

VIDRAM ON

This brings VIDRAM into memory and makes the additional 96K of memory available to all of the applications you may run throughout the day.

The **ON** parameter on the command line is an option. VIDRAM has several options which it will recognize, including the options which all the programs in the QRAM package support. In this section, when an option is first mentioned, the alternate abbreviated form of the option immediately follows it and is enclosed in parentheses.

You will need to understand what these options do if:

- you want to know the status of VIDRAM
- you need to run graphics programs occassionally

- you have a second video adapter and monitor
- you also have expanded or extended memory and a memory manager that also is managing the video RAM memory area

To determine the current status of VIDRAM, you need to enter just the VIDRAM command (no option) from the DOS prompt:

■ **TYPE** VIDRAM **and press** ↵.

The VIDRAM program reports whether or not VIDRAM is resident and enabled. For example, when VIDRAM is not resident, it reports:

```
Status:     NOT Resident
```

If VIDRAM is resident and turned on, then the following is displayed:

```
Status:     Resident
Graphics:   Disabled
Conventional memory ends at:  736K
```

This example shows you that DOS will see the additional 96K of memory (to 736K) and reminds you that graphic capabilities are not available.

To turn the VIDRAM function **OFF (OF)**, and return the 96K of memory for use by the graphics adapter, you would enter the following command from the DOS prompt:

■ **Type** VIDRAM OFF **and press** ↵.

The VIDRAM program will respond with a display which shows that graphics are now enabled and the memory extension has been removed.

There are two parameters you can use that inhibit graphics functions but which do not extend your memory. You use these when you have a memory manager that already is including this memory and you want to prevent graphics programs from using the memory.

■ **Type** VIDRAM NOEGA **and press** ↵.

to prevent EGA requests from being honored.

■ **Type** VIDRAM NOCGA **and press** ↵.

to prevent ***any*** graphics function (CGA through VGA).

The final VIDRAM specific option is used to force the VIDRAM function ON in situations where VIDRAM would normally refuse to appropriate the video RAM 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.

This option is the **OVERRIDE** (**OV** or **OR**) option. It may not work well for you in either of these situations, for it depends upon what adapters may be involved or what memory management features you may already be using. You will simply have to determine for yourself VIDRAM's utility in this situation.

To use the **OVERRIDE** option, you would enter the following command from the DOS prompt:

■ **Type** VIDRAM ON OVERRIDE **and press** ↵.

The VIDRAM program will respond with a display that shows that memory has been extended and that graphics are now disabled.

When you need to use the override option because of a second monitor VIDRAM will not make the entire 96K extent of video RAM available. Take, for example, a system with both a VGA and a monochrome video adapter present. Leaving enough video RAM to support both monitors in their text modes, VIDRAM is still able to extend DOS' conventional memory by 64K.

Finally, as a quick reminder, VIDRAM will also respond to the following options on its command line:

- **NOPAUSEONERROR (NOPE)** tells VIDRAM to not pause on error.
- **PAUSE** tells VIDRAM to pause while parsing commands.
- **HELP** displays the help screen for VIDRAM.
- **?** lists all commands.

Using VIDRAM with LOADHI

You can use the LOADHI program (Chapter 4) with VIDRAM to make the resident portion of VIDRAM remain in high RAM. VIDRAM resident in high RAM cannot, however, extend the memory. To extend the memory, you must run VIDRAM a second time. For example:

■ **TYPE** LOADHI VIDRAM ON **and press** ↵.

■ **TYPE** VIDRAM ON **and press** ↵.

This allows the code that intercepts graphics requests to be in high RAM, while the memory management portion is in memory only briefly while it extends or returns memory. The resident portion of VIDRAM is quite small. You will want to use this feature only if your 'RAM CRAM' problem is extreme.

NOTES:

The QEXT.SYS Program

This chapter describes the QEXT.SYS program supplied with QRAM. The QEXT program provides extended memory management services. Previous versions of QEXT have been in use for several years with Quarterdeck's DESQview program to increase available conventional memory by running a large part of DESQview in extended memory. This feature is now covered by part of the proposed Microsoft Extended Memory Specification (XMS). The current version of QEXT supports XMS and is consequently of use in a wider context.

The program HIMEM.SYS is Microsoft's implementation of XMS. If you have programs that can use XMS features or call for the use of HIMEM, you can use QEXT to supply these features. DESQview users should, of course, use QEXT.

QEXT Features

As an XMS driver, QEXT supports the services defined for two of the memory areas covered by that specification. XMS refers to these areas as the high memory area (HMA), the upper memory blocks area (UMB) and the extended memory blocks area (EMB).

- HMA is a 64K block of memory starting at the one megabyte boundary (1024K), the beginning of extended memory in 80286 and 80386 systems. Its primary and most beneficial use is in enabling program code, nominally restricted to the conventional memory area, to be loaded and executed in HMA. Programs that make use of this feature can free up a large amount of memory to other programs or for increased data capacity.

- UMB refers to the memory areas above the 640K boundary and below the one megabyte boundary which could be used for programs or data if memory were supplied by shadow RAM, expanded memory or extended memory. The QRAM program supplies this memory function. Like HMA, UMB memory can effectively increase available conventional memory.

- EMB refers to the remaining extended memory available to an XMS driver for allocation as extended memory. Extended memory is often used for ram disk storage and disk caching by utility programs.

  NOTE: The terms defined by XMS are in some ways unfortunate. In particular, HMA although called the "high memory area" does not carry the same meaning as the term high memory used elsewhere throughout this manual. The high memory areas that QRAM converts to high RAM corresponds to the UMB of the XMS definition. You need to be aware of these terms and their meaning when you have programs which make use of an XMS driver and which may have configuration parameters that you have to change or specify. This chapter will make use of these terms.

QEXT Parameters

This section is the reference guide for the command line parameters used with QEXT.SYS. The default extended memory support provided by QEXT

should be correct for most users. If you are having difficulty with QEXT or have special needs, one or more of these QEXT options may help.

To use a QEXT.SYS command line parameter you type the parameter name on the same line as DEVICE=QEXT.SYS in your CONFIG.SYS file. You may use an abbreviation instead of the full parameter name, if the parameter has an abbreviated form. The abbreviated form will be shown in parentheses following each parameter name in the paragraphs describing each option. The QEXT.SYS command line with two parameters, might look like:

```
DEVICE=QEXT.SYS MEMORY=512 HANDLES=40
```

IMPORTANT: Separate each parameter by at least one space. If the parameter specifies a value, do not put spaces anywhere within the parameter specification. Also, all of the parameters ***must be*** typed on the same line as DEVICE=QEXT.SYS.

- **BLOCKSIZE=xxxxx** specifies the block size, in bytes, of memory transfers to and from extended memory blocks. Legal values for xxxxx range from 2 to 65535. EMB transfers go through the BIOS extended memory support routines. These routines lock out interrupts during their operation. This interrupt lockout may be unacceptable for large memory transfers done with one request. QEXT will break up large memory transfer requests according to the block size you specify.

- **EXTMEM=xxxxx, (EXT)**, specifies the amount of extended memory which should not be used by QEXT.SYS. **xxxxx** must be in the range 1 to 15360 representing an allocation of 1K to 15360K. This parameter is useful when using RAM disks or disk cache utilities which use extended memory, since you can ask QEXT to "leave alone" the memory needed. Without EXTMEM parameter, QEXT uses all the extended memory it can find.

  The EXTMEM and MEMORY parameters have opposite meanings. The EXTMEM means "leave at least this much memory". MEMORY means "use up to this much memory".

- **HANDLES=xxx**, (HA), specifies the number of EMB handles available. **xxx** must be in the range 16 to 255. Each application which uses EMBs requires at least one handle. Some applications require extra handles. The default is 32 which should be adequate for most purposes. Larger values increase the amount of extended memory used by QEXT. Each HANDLE requires 20 bytes of memory.

- **HMAMIN=xx** specifies the minimum amount of memory, in kilobytes, that a program which uses the XMS High Memory Area (HMA) can request. Legal values of **xx** range from 0 to 63. Since only one program can use the HMA while several programs might want to use it, this parameter only allows a program which uses at least **xx**K of memory in

the HMA. The parameter is not necessary if you have only one program which uses the HMA, such as DESQview.

- **MEMORY=xxxxx, (ME or MEM),** specifies the amount of extended memory that QEXT makes available as extended memory blocks. **xxxxx** should be in the range 128 to 32128, representing an allocation of 128K to 32128K. If the MEMORY parameter is omitted, QEXT takes all of the available extended memory.

- **NOXMS** instructs QEXT ***not*** to be an extended memory manager. Normally QEXT provides extended memory specification (XMS) services. With this parameter, QEXT still supplies the DESQview interface.

- **UNUSUAL8042, (U8),** specifies that the computer has a "non-standard" 8042 keyboard controller. If running DESQview (or some other program) disables the keyboard then the UNUSUAL8042 switch should be specified.

- **VDISKVERSION=x.xx, (VD)** tells QEXT the VDISK.SYS version number you are using. Most often, the VDISK you use will be the one that came with your DOS and will have the same version number. Use this QEXT parameter to tell QEXT your actual VDISK version number, when they are different. VDISK identifies its version number when it starts up.

- **NOPAUSEONERROR, (NOPE),** doesn't pause on an error. QEXT usually displays a "Press any key to continue, ESC to abort" message after parsing command line instructions if there are parameter errors. This instructs QEXT.SYS to not display the message. When an instruction causes QEXT.SYS to generate the above error, and when the error generated has been shown to not cause any disruption of the system, use the NOPE instruction

- **PAUSE** tells QEXT.SYS to pause the display when it starts up. The PAUSE command permits you to hit the "Esc" key to stop QEXT.SYS from installing. This is useful when trying out new parameters, so you can abort the load if there was an error. Note that if there is an error on the QEXT.SYS command line, QEXT will automatically pause to show you the source of the error.

- **HELP** displays all QEXT.SYS parameters and a one line description of what each does.

- **?** displays a list of all QEXT.SYS command line parameters and where appropriate the abbreviation for the parameter.

Notes:

Other QRAM Programs

This chapter describes two other QRAM programs, EMS2EXT and EMS, which give additional control over expanded memory. These two programs can be used separately or they can be used together to give you some dynamic control over your extended memory allocation.

EMS2EXT.SYS

If your system has EMS 4.0 expanded memory, but not extended memory, you may want to use the EMS2EXT.SYS device driver supplied with QRAM. EMS2EXT can supply expanded memory as extended memory to programs that access extended memory in a certain prescribed manner. Two such programs are IBM's DisplayWrite and the VDISK.SYS RAM disk utility supplied with DOS. Many disk cache utilities often use extended memory in this way also.

These and other programs can be used with EMS2EXT.SYS because they do not require that extended memory (memory above 1024K) be physically present, nor do they need to directly address this memory. They make demands upon this memory in a logical as opposed to a physical sense. And they access the contents of this memory through requests of a memory manager, which makes this memory available to the program. Programs which make use of extended memory in this way provide a performance boost over programs that rely on disk storage alone, since the in-memory transfer is much faster.

Programs that expect extended memory to be physically present cannot make use of EMS2EXT. Quarterdeck's QEXT.SYS driver, or other XMS drivers, cannot use memory supplied by EMS2EXT.

Because this "extended" memory capability enhances a program's performance, you should have little difficulty in determining which of your applications support this memory. They will surely mention it. And, if your complement of expanded memory lends itself to this use, you will want to make this memory available to these programs.

If you decide to use EMS2EXT you may also want to consider the relative speed of the various types of memory you have in your system. It is possible for there to be a significant speed differential between these types of memory. EMS2EXT is capable of determining this difference and can allocate either slow memory or fast memory for use as extended memory.

EMS2EXT is a device driver and therefore needs to be loaded when your system boots up. This requires a DEVICE= statement in your CONFIG.SYS file. When you have determined the amount of memory to ask EMS2EXT to manage as extended memory, you need to modify CONFIG.SYS to load

EMS2EXT. The statement to load EMS2EXT must come after the one that loads your expanded memory manager and looks like:

DEVICE=C:\QRAM\EMS2EXT.SYS MEMORY=nnn speed

Several components of this statement need to be explained.

- The **nnn** parameter in **MEMORY=nnn** is the number of kilobytes of expanded memory to allocate, i.e. **MEMORY=512**. This is optional if you plan to use EMS and EMS2EXT together. See the Section Using EMS with EMS2EXT.
- The **speed** parameter is the optional specification for using faster or slower memory for allocation. Leave this part of the line blank or use either **FAST** or **SLOW**.

The EMS Programs

The EMS programs provide a number of informative and powerful functions to help you make the best use of your EMS 4.0 expanded memory where you might have special or unusual requirements. Although anyone might benefit from the EMS summary status report and others by seeing the detail of expanded memory allocation, other uses of EMS which will be described in these sections are for the more technically minded.

Most of the functions of the two EMS programs, EMS.SYS and EMS.COM, involve the manipulation of expanded memory handles. An EMS handle is a reference to a block of memory (zero or more EMS pages) that the expanded memory manager issues when it allocates memory. A handle is represented by a number and it may be given a name.

An expanded memory handle, its name, and the pages of memory associated with the handle are the fundamental tokens of interaction between an application program and an expanded memory manager. The two EMS programs give you some of the same control functions available to programs. The principal need for caution when using the EMS programs arises because they give you access to handles which may belong to other programs and EMS will not prevent you from inadvertent mischief making.

With the EMS programs you can allocate and name a block of memory with the **CREATE** option, and specify that the memory be fast or slow memory. You can free expanded memory pages allocated to a handle with the **FREE** option. You can read data from a file into allocated expanded memory or write the data from allocated expanded memory to a file with the **LOAD** or **SAVE** options. An EMS handle can be renamed, and the number of memory pages can be changed.

Along with the report features of EMS, you can experiment with the functions using EMS as a command line interface to the expanded memory manager.

If parts of the expanded memory in your system run at different speeds, you can use EMS to pre-allocate memory of one speed before you load a device

driver or TSR so that it can only use the memory that remains at the other speed; and you can then free the memory for use by your other applications. Manifest can show you if your memory runs at different rates.

Developers using expanded memory may make use of the **LOAD** and **SAVE** function to help them when they need to work with the same context repeatedly during development and debugging.

All of these uses of EMS require an extensive knowledge of the various aspects of your system, from the details involved in the DOS boot sequence, to uses and capabilites of expanded and extended memory as well as an understanding of the applications you are attempting to control and optimize.

Both EMS.SYS and EMS.COM respond to the same command line options. You use EMS.SYS from within the CONFIG.SYS file to manipulate expanded memory during the system boot sequence. You use EMS.COM from within the AUTOEXEC.BAT file at the end of the boot sequence or directly from DOS prompt as needed.

Described below are the command line options of EMS and the parameters required by each option. Some of the options have an abbreviated form. Where this is the case, the alternate form is shown immediately following the option when it is first introduced, and is enclosed in parentheses.

To get a summary report of your expanded memory, you issue the EMS command from the command line without any option specified. EMS outputs the following information:

- the total amount of expanded memory, and
- the amount currently available, and
- the address of the Page Frame.

The **DIR** option displays, in tabular form, a breakdown of the current expanded memory allocated. For each allocated handle, the number of expanded memory pages is given, the number of kilobytes of memory those pages represent, and the name assigned to that handle, if any.

The **CREATE (CR)** option is used to allocate pages of expanded memory. It takes two parameters. **CREATE** requires that you provide a name for the memory you are allocating and that you specify 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 determine the number of pages. Follow the **EMS CREATE** command with **EMS DIR** command to confirm the allocation and to determine the handle number assigned to the name.

The **CREATEFAST (CFAST)** and **CREATESLOW (CSLOW)** options are forms of the **CREATE** option that also instruct the memory manager to allocate the memory from either faster or slower memory. Use Manifest's Memory Timings to determine if this form of **CREATE** is important to you.

The **FREE** option frees memory and deallocates a handle. **FREE** requires that you specify the handle to deallocate—by its name or its number.

The **RENAME (REN)** option lets you assign a new name to a handle. The first argument 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.

The **RESIZE (RES)** option lets you increase or decrease the number of pages assigned to a handle. Its two arguments are the same as those of **CREATE.**

The **SAVE** option allows you to save the contents of the expanded memory pages associated with an EMS handle to a file. This option requires that you also specify the handle name (or number) and the file name.

The **LOAD** option allows you to restore the contents of expanded memory pages that have been stored in a file. This option requires that you also specify the handle name (or number) and the file name which contains the data you want to restore. The number of pages required will be automatically allocated based on the file's size.

The EMS programs also respond to the options common to all the programs supplied with your QRAM programs:

- **NOPAUSEONERROR (NOPE)** instructs EMS not to pause on error.
- **PAUSE** instructs the program to pause while parsing commands.
- **HELP** displays EMS help text.
- **?** lists all EMS command parameters.

Using EMS with EMS2EXT

You can load EMS2EXT as described in the EMS2EXT section of this chapter, but without specifying any memory parameter at all. EMS2EXT will be resident, but it will not allocate any memory. It will, however, recognize as its memory a handle with the name "EMS2EXT". You can then, as needed, create, grow, or shrink the amount of extended memory for this handle using EMS.COM.

This capability should only be used in special situations when you know how and when the program will use extended memory. For instance, a program could be given extended memory only while it is running. You could make a batch file which ran **EMS CREATE "EMS2EXT" 128K** before running the application. When the program terminates, another EMS statement, **EMS RESIZE "EMS2EXT" 0**, could release the memory.

Appendix A Troubleshooting and Special Conditions

Using LOADHI without QRAM.SYS

It is possible to use LOADHI if you have Quarterdeck's QEMM-386 or QEMM-50/60 and the appropriate memory. This may even be more efficient on an 80286 based PS/2 Model 50 or 60, or an 80386, 80386SX or i486 with QEMM's RAM parameter. The RAM parameter of these programs allows memory areas as small as 4K in size to be managed as opposed to 16K at a time with QRAM.SYS. The LOADHI programs require that you either have QEMM.SYS or QRAM.SYS.

QRAM.SYS reports "no EMS 4 or EEMS driver is present"

The QRAM installation or OPTIMIZE program attempts to locate the expanded memory manager in your CONFIG.SYS file by finding a "DEVICE=" line with a driver name having "EMM" in it. Most expanded memory managers have "EMM" in their name. If yours does not, then the DEVICE=QRAM.SYS line may not have been placed AFTER the expanded memory manager. Simply use your favorite editor to move the DEVICE=QRAM.SYS line after the expanded memory manager and reboot your computer.

Another possibility is that you have not installed your expanded memory manager, or you do not have expanded memory (or shadow RAM). The QRAM.SYS program REQUIRES expanded memory or shadow RAM before it can work.

Make sure your expanded memory manager is installed. If you don't have expanded memory, see if the VIDRAM program features can be used with your system to obtain some extra memory.

QRAM.SYS reports "Nothing Useful to do"

In order for QRAM.SYS to provide some benefit, there needs to be MORE than 64K of expanded memory available between 640K and 1024K. The first 64K is used for the "Page Frame", which is the area most programs use as a "port hole" into the expanded memory pool. Some expanded memory boards, and motherboards with expanded memory hardware, provide an EMS 4 driver even though the expanded memory hardware cannot provide more than 64K of expanded memory at a time. These boards can give software programs the ability to access expanded memory using the more advanced features of EMS 4, but the hardware cannot provide the extra support for more mappable pages. Programs which use expanded memory, require all 4 of the available pages that make up the page frame. This kind of setup can be seen by checking the Manifest Expanded Pages display (without QRAM.SYS loaded). If all you see is four "F" characters and no other "+" characters between A000 and FFFF then there are no "extra" mappable pages for QRAM.SYS to use. The AboveBoard Plus offers an option chip which upgrades the board from four mappable pages to more than four pages. Contact your board maker to see if this may be an option for you.

Another possibility is that the expanded memory manager you are using will only allow contiguous memory areas to be mapped. If the "Page Frame" is placed into the highest 4 mappable pages, then there won't be any other mappable pages between 640K and 1024K available. This is usually easily fixed by specifying the "Page Frame" to start as low as possible (try C000, C400, or C800 first). Then the first four pages will be the "Page Frame" and the rest of the pages will be available for QRAM.SYS to use. Early memory manager versions for the AboveBoard Plus were like this.

If you are NOT going to be using expanded memory in your programs, or you would just like to see what QRAM can do, then you can use the FRAMELENGTH=0 parameter to QRAM.SYS. This will allow the 64K Page Frame to be used as an area of High RAM. Other programs will no longer be able to detect expanded memory, but the LOADHI programs will now work. While it seems counterproductive to have several megabytes of memory yet only use 64K of it, you may have an overwhelming need to get about 64K of resident programs out from below 640K. Note that, if the board allows mapping BELOW 640K, then a program such as DESQview will NOT be able to use the expanded memory pages below 640K when FL=0 is in effect, unless FORCEEMS is also specified. Therefore, if you want DESQview to multitask with expanded memory below 640K and have QRAM make the page frame into high RAM, both FL=0 and FORCEEMS must be used.

PC does not function after booting with QRAM.SYS

This kind of problem may happen when QRAM and some other feature or device of your computer are unable to cooperate. You may need a Boot Floppy to proceed, so be sure to have one ready (see Appendix B). This item covers problems when just having DEVICE=QRAM.SYS in the CONFIG.SYS file causes the

computer to "lock up". Another item covers problems with LOADHI, although the information here may prove to be useful as well.

The first thing to do is to try and figure out what the conflict is. Remove the DEVICE=QRAM.SYS line and reboot. If the computer now functions correctly, then some area between 640K and 1024K is being used by a device but your expanded memory manager does not "see" it as in use. It will tell QRAM the pages are mappable and QRAM will map memory there when it should not. Using the Manifest First Meg Overview display, check to see if any areas which are "Mappable" are currently in use by another device. The most common devices are video or network adapters. Usually your expanded memory manager can "see" these cards, but if not, then you should use the "exclude" parameter of your expanded memory manager (if it has one) to keep the area from being "Mappable". If your expanded memory manager does not support exclusion, you can use the EXCLUDE parameter on the QRAM.SYS line to force these areas not to be used by QRAM.SYS. Often a network adapter has some RAM between 640K and 1024K, and if so, then that area should be excluded.

If you cannot decide on the areas to EXCLUDE, perhaps QRAM is "filling" the memory area between 640K and 736K (where video cards may have their memory). You can add the NOFILL parameter to the QRAM.SYS line in the CONFIG.SYS. If NOFILL allows the computer to work, then QRAM.SYS is confused about your video adapter. Check the documentation for your video adapter and use EXCLUDE to avoid having QRAM.SYS put memory into the conflicting video areas.

If you still haven't determined the conflict, then you should follow the instructions in Appendix C, to create a "pure environment" in order to isolate the exact problem which is causing the trouble.

PC does not function when using LOADHI

If the computer "locks up" after loading programs in high RAM (using the LOADHI programs or Quarterdeck's XDV.COM or DV.COM), then there is probably some memory conflict between 640K and 1024K (see discussion above), but only when the conflicting memory is used. It's not always easy to tell which program using LOADHI is causing the problem, but a patient, systematic approach (using LOADHI with one program at a time) is the fastest and most reliable method.

While the memory conflict is the most common problem when loading items into High RAM, there are programs which are not capable of being loaded high. These programs may require addresses below 640K to work for example. As long as there is enough room to load the program (you can use the /GETSIZE option of LOADHI to be sure) most programs can be loaded into high memory. Watch the program's initialization display to see if it loads successfully.

If the problem is with XDV.COM or DV.COM (which is just a copy of XDV.COM) when running DESQview, then you can use the "/L" (no quotes) option of XDV.COM to determine the areas of High RAM which XDV is using to load the DV.EXE file into. You can then use the "/X=xxxx-yyyy" option to XDV.COM to specifically exclude areas which may be a problem. If you find an area, exclude it from use by your memory manager if you can. If your memory manager doesn't have this feature, use QRAM's EXCLUDE parameter. The /L and /X options to XDV.COM can be used at the same time, and the /X option may be used more than once if needed.

Network drivers "lock up" or act unpredictably

If you have this problem, then it is probably a memory conflict with memory between 640K and 1024K. Token Ring, ARCnet, Proteon, Ethernet, and some others may not be "seen" by your expanded memory manager at boot time, so QRAM.SYS may have placed High RAM into the memory area used by the network card. Use the methods described previously to isolate and exclude the conflicting area. You may even find the memory address listed when the network driver initializes, and most network cards list the addresses they use in their documentation.

Graphics programs which have corrupted displays

The most likely cause of a graphics program having its display appear corrupted or "fuzzy" is that some High RAM has been placed into an area that the graphics adapter was not using until a particular program started using it. Remove QRAM.SYS and check the Manifest System Adapters and the Manifest First Meg Overview display to help identify the area of conflict. You should then explicitly EXCLUDE the video area being used (usually somewhere in the A000 to CFFF area, but probably not all of it).

Attempting to LOADHI a program gives the message "Not enough room to load high" even though there appears to be enough High RAM

When using LOADHI, a program will need its "normal" amount of memory in High RAM in order to get started. Many TSRs and device drivers take much more room to initially load into memory and then reduce their memory size to only the amount needed to stay resident. The /GETSIZE (/GS) parameter can be used with LOADHI to determine if this is happening. Using the OPTIMIZE program will automatically keep track of the amount of "run time" memory as well as its "resident" size for all programs in the CONFIG.SYS and AUTOEXEC.BAT files, and will try to find a way to place as many programs in High RAM as possible.

Not very many areas between 640K and 1024K are "Mappable" so there is not much room to use LOADHI

QRAM.SYS is completely dependent on your expanded memory manager in order to be able to provide High RAM. If your expanded memory manager does not allow many pages between 640K and 1024K, then QRAM.SYS will be limited in providing large areas of usable High RAM. On 80286 computers, it is common for the E000-EFFF area to have a "ROM socket". If this is the case, then your expanded memory card will NOT be able to put any expanded memory in the E000 area. Some 80286 computers DO allow this area to be used, but some expanded memory managers cannot place memory there. Some experimentation may result in finding if the E000 area is usable, but it is not very common. On 8088 and 8086 computers, there is usually a "Disk ROM" at C800. This will fragment the free memory area between 640K and 1024K into pieces. If your expanded memory manager cannot manage multiple discontiguous areas between 640K and 1024K, then you will only be able to convert the largest contiguous area into High RAM in addition to the Page Frame. This is true for 8088, 8086, and 80286 computers. You may want to contact your expanded memory board maker to see if they have produced an update which allows discontiguous areas to be managed.

Some TSRs or drivers seem to run slower when loaded into High RAM

Since QRAM is completely dependent on the memory from the expanded memory board, it is possible the memory supplied in the 640K to 1024K area is slower than other memory in your system. This may be due to the expanded memory board using 8-bit memory accesses above 640K. The Manifest First Meg Timings may show you this slowdown effect. If the memory is indeed accessed more slowly above 640K, then the program running in the 640K to 1024K area will run slower when loaded into this memory.

Using QRAM with Microsoft Windows

QRAM.SYS will work with Microsoft Windows/286 2.xx and Windows/386 2.xx.

When used with Windows/286, your conventional memory area should be larger due to using LOADHI to load items between 640K and 1024K. However Windows also tries to load some of itself into this area as well. If you find that you still have more than 16K of free memory available (use Manifest First Meg Programs or LOADHI without any parameters), then you may want to EXCLUDE an area so that Windows can use the memory.

When used with Windows/386, the above conditions apply. In addition, you will be using WIN86.COM to run Windows. You will not be able to run non-Windows applications, but you should have more memory available to your Windows Programs. You may want to use the HIMEM.SYS or QEXT.SYS driver as well.

If you are running several Windows programs, it may be to your advantage to run them inside DESQview. When run this way, each Windows program gets its own memory area to use and does not have to share the memory with other Windows programs. In addition "off-the-shelf" (non-Windows) programs can be run as well. If you run Windows inside DESQview, then you won't need HIMEM.SYS. Use QEXT.SYS instead.

Using LOADHI FILES with Microsoft Windows (strange beeping when starting Windows)

Microsoft Windows 2.xx seems to not like having the DOS resource FILES loaded into High RAM. If there are fewer than 10 FILES resources below 640K, then Windows will usually act very strangely when started, and will never actually start working. You must have at least FILES=10 in your CONFIG.SYS file, and if you run other programs in DESQview before using Windows, you may need more before Windows will start. If the problem persists, try adding 5 more FILES at a time in CONFIG.SYS until it goes away. Once you've determined the minimum needed for Windows, additional FILES can added using LOADHI.

aximum memory ize programs in DESQview

hen using DESQview, many people would like to have largest possible program area in which to run their grams. Indeed, DESQview allows you to "re-use" the mory area below 640K many times, so having a large a is very important. You should be sure that the rameters (using Change a Program) for "Memory Size K)" is quite large (400K is a good number) and "Maxi- m Program Memory Size (in K)" is even larger (800K ll work). This makes sure that the window will be AT AST 400K in size, and then DESQview will allocate re up to the Maximum specified. (The 800K is far re than is even possible, so you're sure to get all of rest without having to figure it out.) In order to gain maximum size of memory, you should be using the V.COM program (or have copied XDV.COM to COM, since DOS will always load a ".COM" file ore loading a ".EXE" file).

e XDV.COM program puts some of DV.EXE into High M. You can find out how much by adding "/L" (no otes) to the XDV command line to get a display of the areas, their size, and how much of DV.EXE is placed re. This points out an interesting dilemma: since ADHI puts items into High RAM, and XDV puts ces of DV.EXE there too, you may actually end up ving some of DV.EXE in Conventional memory be- se LOADHI has used up so much of it. You can check ur free memory before running DESQview (with KDSK or Manifest) and then check the "Total Avail- e Conventional Memory" using DESQview's Memory tus command. You may want to evaluate what TSRs u are running before DESQview, perhaps they would better run inside DESQview, and this can free up mory.

aking sure that you are using memory in the st efficient way possible

e of QEMM's and QRAM's greatest features is that u can control the memory you have better than you r could before. Many people want to make sure that available memory between 0K and 640K is as large as ssible. Using High RAM and the LOADHI programs go a long way to make sure that the conventional mory area is as big as possible. The OPTIMIZE pro- m can assist in maximizing conventional memory by alyzing the possible methods of loading items into gh RAM and placing them into the best possible loca- n. Using Quarterdeck's Manifest program, you can ob- ve how much memory is still being used below 640K looking at the First Meg Programs display. The DOS Overview display will show you if more of DOS is being left below 640K that could be loaded high. A word of caution: you can spend a lot of time trying to get an extra few bytes of memory and not really gain much. Try to determine what the memory savings will be before attempting to spend a lot of time rearranging things. Loading TSRs and device drivers into High RAM is usually the quickest way to gain useful amounts of memory. On DOS 2.xx and 3.xx systems, loading BUFFERS into High RAM is easy and provides a lot of memory savings.

You may find that a particular adapter which is using a lot of memory addresses above 640K can either be removed or moved to a different location to get larger contiguous areas of High RAM. Having many small areas of High RAM is not as useful as a few large contiguous areas.

If you are not using ANY programs which use graphics, or if some large programs you use don't need graphics, you can use the VIDRAM program to gain a significant amount of memory from your EGA or VGA adapter.

Making a Boot Disk

You should make sure that you always have a boot floppy available for your system. There are things that can go wrong which may make it impossible to boot from your hard disk. Many of these are simple problems which can easily be fixed, but only if you can start up the computer.

Even if you don't consider yourself a power user, you can make sure that you can boot from a floppy. With a boot floppy, you may be able to restore the computer yourself. You may know exactly what went wrong and how to fix it.

For more difficult problems, you may be able to get help from a service department or technical support by telephone. If you can describe the symptom and can access your hard disk, someone may be able to walk you through the steps necessary to restore operations. This can save you hours or days of unnecessary downtime.

If you have the original diskettes that came with your system, you can use the system disk as a boot floppy. A better idea would be to make copies of them, and keep the originals safe. With these you have a complete operating system, but you will not have any custom features you may have installed yourself.

If you don't have your original system disks, you can make a boot floppy. You will need to use the DOS format command with the system option. The statement:

FORMAT A: /S

will format and transfer the operating system programs to a diskette in your A drive. This would be a minimum boot floppy. You should also copy other DOS programs to this floppy. If you can, copy all the DOS programs. If you don't have room, copy at least EDLIN.COM and DEBUG.COM. You can use EDLIN to edit files. You may be guided in the use of DEBUG by others. Other programs of more general use are MODE.COM, CHKDSK.COM, and FORMAT.COM. Include these also.

You can also copy your CONFIG.SYS and AUTOEXEC.BAT files to this floppy. However, they will probably not work well without some additional effort. This effort involves editing the files, copying other program files to the floppy, or both. Since these files represent custom features, the changes you might need to make cannot be covered here.

Finally, assuming you have a boot floppy now, you should try it out. Make sure you can boot from a floppy disk. See what features and capabilities you have under these circumstances. Observe what features you will have to do without if necessary.

eating a Pure vironment

One of the main reasons for QEMM's and QRAM's existence is to help manage your memory—so that you can use all of the TSRs, device drivers, and add-on hardware you want without using so much conventional memory that your applications programs can no longer start up. However, with this power there also may be danger. That danger is that, now that your programs can fit into memory together, there is some conflict between their functions which cause one or both of them to not work. This incompatibility might, in extreme cases, cause your system not to function at all.

This class of problems, (the incompatibility between two or more elements of your system), is not new. But now that you have the ability to use so many pieces of software at once, it may be more likely to occur. For this reason, you should know how to diagnose a problem should one happen. Your ability to do this could save you hours of time on long-distance phone calls trying to get technical support from one vendor after another.

To diagnose a problem when you have many pieces of software loaded, you must determine whether one of your programs simply doesn't work at all, or whether there is some incompatibility between two or more of the programs you are using. The technique we suggest for isolating what is going wrong is to create a "pure" environment for your program. A pure environment is one in which any non-essential drivers or resident programs have been removed from your CONFIG.SYS and AUTOEXEC.BAT files. In difficult cases, where alternate hardware is available, creating a pure environment can even be extended to removing non-essential hardware devices or replacing non-standard ones with devices that are more common and known to be sound.

Here is a step-by-step procedure for trouble-shooting and creating a pure environment:

- ❑ Determine whether the problem might be something you have recently changed. If the problem began when you installed something new into your system, or changed the configuration of one of the elements of your system, then uninstall the new element or restore the old configuration for it, and see if the problem goes away. If it does, you have already isolated one part of the problem and you may be able to determine what to do about it.

- ❑ Make a bootable floppy disk. If you normally boot from your hard disk, make sure before you make any changes that you have the means to boot from a floppy disk, and that you'll be able to access and edit CONFIG.SYS or AUTOEXEC.BAT on your hard disk, having booted from a floppy. If you are not totally familiar with the operation of the items in your CONFIG.SYS, you could find that removing an item may make your system unable to boot. A bootable floppy enables you to get started again. Make a dry run to be sure you can boot from the floppy. Appendix B provides more insight into making a boot floppy.

- Make copies of your current CONFIG.SYS and AUTOEXEC.BAT files. Use file names other than CONFIG.BAK and AUTOEXEC.BAK, as these tend to get overwritten by text editors. These copies will give you a reference to what you are about to change and provide a way to get the system back to the state it was in before you started experimenting.

- Remove statements from the CONFIG.SYS and AUTOEXEC.BAT files by putting the word "REM" (without the quotes) and a space on the lines in each file which load the resident programs or drivers. This way you still remember the whole statement, but it won't be used. If your DOS version is prior to 4.0, the use of REM in the CONFIG.SYS file will cause an error message to be displayed on boot which says "Unrecognized command in CONFIG.SYS", but this error message can be ignored. DOS will bypass the loading of the driver, which is what you want at this point.

- Avoid removing statements from either CONFIG.SYS or AUTOEXEC.BAT that are absolutely needed for your system's basic functionality. If Hard disk drivers are needed to boot your system, they should generally not be removed, while RAM disk drivers or disk caches probably should be removed. It is not necessary to remove PATH= or PROMPT= statements from your AUTOEXEC.BAT file as these usually don't cause problems and removing them will simply make life difficult for you. You should probably be more aggressive about removing other statements in your AUTOEXEC.BAT with "REM". Do not assume that any utility is trouble free just because you have been using it for a long time. Remove it now—you can add it back and ***exonerate*** it later.

- Now that the environment is "pure", test to see if a problem still occurs. If it does, then either your hardware is faulty, or the program you are testing is fundamentally incompatible with your hardware. You may need to have your hardware checked out. If it seems OK, contact the maker of the hardware and software to see if they are aware of a solution. If the problem does not occur, then there must be something in the CONFIG.SYS or AUTOEXEC.BAT which is creating the incompatibility.

- Add back the statements to either CONFIG.SYS or AUTOEXEC.BAT which pertain to the specific area you feel could be the problem by taking the "REM" off of ONE statement at a time. For instance, if you are analyzing or trouble-shooting in the memory area, you would want to enable your memory manager at this point.

- Retest with this new addition. If the driver you thought might be the problem proves not to be, reactivate other resident programs or drivers ONE AT A TIME, starting with the ones you feel are most essential to the proper operation of your computer and test again after rebooting with each new addition. Be patient and systematic. Once the specific

item which causes the problem is isolated, you can perhaps go on to resolve the problem fully by doing the following:

Remove the item permanently—in some cases this might be a little used utility, which was added at some point in the past and forgotten. If you are not using it and it is causing a conflict, perhaps the simple solution is to not load it at all. Since all resident software and drivers require some memory in which to operate, it makes sense to avoid loading them if they are not being used.

Check the documentation for the utility or driver—it may point out areas of conflict which did not previously exist in your system, but do now. It may specify command line parameter switches which may have not been needed previously, but are now. For example, if the problem seems to be a conflict between a memory manager, and the use of a network or a special video mode on your graphics card, the problem could be that the memory manager is mapping memory in the area between 640K and 1024K that is also used by the network or video card. By determining where the memory manager is mapping high memory and referring to the documentation for the network or video card to see what high memory addresses they use, you can add the correct parameters to tell the memory manager to exclude the mapping of the areas of conflict, and that could be enough to resolve the problem.

Try loading the drivers in a different order—in some cases, the order in which resident programs or drivers are loaded can be important.

Contact the developer of the resident program or driver. If you are encountering a problem, in many cases you will not be alone. A problem or conflict that may seem quite difficult to you may be commonplace to the developers of the program and they will be able to assist you.

Contact Quarterdeck Technical Support if the problem seems to revolve around one of Quarterdeck's products and you are unable to resolve the problem using the foregoing technique. You should not feel that these efforts have been in vain. Be sure to document the findings of your investigation while they are fresh in your mind. The information you have gained will be of much use in helping our personnel to further analyze and hopefully, resolve your problem.

Contacting Quarterdeck Technical Support

If you are contacting us by mail, fax or via information systems, please include in your correspondence:

- ❑ The version number and serial number of the software product or products you are inquiring about.
- ❑ A printout from Quarterdeck's Manifest, if you have it. If you like, you can select "All Manifest" from the "What to Print" part of Manifest's print menu, and send the entire printout to us. If you are sending by FAX or

want to save paper or connect time by printing out only specific sections, probably only the following Manifest screens are necessary for our technical support: System Overview, CONFIG, AUTOEXEC, and Adapters; First Meg Overview and Programs; DOS Overview and Drivers. If you think that there is a problem with your expanded memory, include the Expanded Overview, Pages, and Handles screens. If you think there is a problem with your QRAM installation, include all of the QRAM screens. If you think there is a problem with your DESQview installation, include both of the DESQview screens. If you feel you have other important hardware in the system, or if Manifest's list is in some way incorrect or incomplete, please include information on this as well.

- ❑ If the problem is that you can't run Manifest at all, then you cannot, of course, send us a print out from Manifest. Instead, print out your CONFIG.SYS and AUTOEXEC.BAT files, and write down what hardware (including make and model) and software (including version) you are using. The "Problem Report Card" included with your Quarterdeck product will serve as a template for the information we consider essential.

- ❑ A detailed description of the problem you are encountering. Please describe EXACTLY what happens, and the exact text of any error messages that might be encountered. Describe in detail the results of your troubleshooting efforts using the method described to create a pure environment as outlined at the beginning of this Appendix, especially if the problem does not occur in the "pure environment" but does after using particular drivers.

- ❑ Naturally, give us information on 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 the various avenues for contacting us.

If you are contacting us by telephone:

- ❑ Please prepare yourself by following the trouble-shooting procedure and collecting the same information you would prepare if you were mailing your request for support.

- ❑ When you contact our technical support representative, you need only give your serial number and a brief description of your hardware, software and the problem you are encountering. If the support technician requires additional information, you will be asked for specific details. As indicated earlier in the this section, there are some problems which become very familiar to technical support and in some cases, only a brief description of the system and the problem is all that is needed.

RAM and panded mory Boards

Below you will find a portion of the Manifest Expanded Pages display for 4 different expanded memory boards. For each display the expanded memory board was configured in its best possible way. The same computer was used each time. The purpose of this appendix is to explain why each of the boards is capable of using QRAM, but some features will be different depending on the board. Compare the Manifest Expanded Pages display for your memory board with the four boards below and pick the one which most closely matches, then read its associated description. Note that the "+" signs in the area below A000 are "backfilled" expanded memory pages. This backfilling is very useful for multitasking, but is not essential for QRAM to function.

Board #1	Board #2	Board #3	Board #4
Expanded/Pages	Expanded/Pages	Expanded/Pages	Expanded/Pages
n=048C	n=048C	n=048C	n=048C
0n00 UUUU	0n00 UUUU	0n00 UUUU	0n00 UUUU
1n00 UUUU	1n00 UUUU	1n00 UUUU	1n00 ++++
2n00 UUUU	2n00 UUUU	2n00 UUUU	2n00 ++++
3n00 UUUU	3n00 UUUU	3n00 UUUU	3n00 ++++
4n00 UUUU	4n00 ++++	4n00 ++++	4n00 ++++
5n00 UUUU	5n00 ++++	5n00 ++++	5n00 ++++
6n00 UUUU	6n00 ++++	6n00 ++++	6n00 ++++
7n00 UUUU	7n00 ++++	7n00 ++++	7n00 ++++
8n00 UUUU	8n00 ++++	8n00 ++++	8n00 ++++
9n00 UUUU	9n00 ++++	9n00 ++++	9n00 ++++
An00 UUUU	An00 UUUU	An00 UUUU	An00 UUUU
Bn00 UUUU	Bn00 UUUU	Bn00 ++UU	Bn00 ++UU
Cn00 UUUU	Cn00 UUFF	Cn00 UUFF	Cn00 UU++
Dn00 FFFF	Dn00 FF++	Dn00 FF++	Dn00 ++++
En00 UUUU	En00 UUUU	En00 UUUU	En00 FFFF
Fn00 UUUU	Fn00 UUUU	Fn00 UUUU	Fn00 UUUU

Board #1

This board has only a "Page Frame" (indicated by the "FFFF") in the memory area above 640K (A000). This is typical of a memory board designed to work with EMS 3.2 yet has an EMS 4.0 driver. The Expanded Memory chapter of the Manifest manual calls this board "EMS 3.2 Hardware (Only 64K mapping)". The lack of extra "mappable" areas above 640K (there are no areas above 640K with "+") limits QRAM's effectiveness. You will be able to use FRAMELENGTH=0 to see how LOADHI can work, but then no other programs will be able to access expanded memory. If you do NOT use the FL=0 parameter, then QRAM will report "Nothing useful to do" since there aren't any extra mappable pages to turn into High RAM. You may still use the VIDRAM program to increase the DOS memory size. Some boards in this category may have an upgrade to EMS 4 hardware available. You should check this if you want to take advantage of other features.

Board #2

This board has two extra pages (32K) of mappable memory above 640K (at D800 and DC00). Although no Monochrome adapter was installed in the PC, the area from B000-B7FF is not used by this memory board. The two available areas will be converted to High RAM by QRAM and then the LOADHI programs can use it. If the E000-EFFF area did not have anything in it, then the board could fill in that area too. Most 80286 computers do not actually have anything in the E000 area, but there may be a ROM "socket" using the area which keeps the expanded memory from being placed there. The 32K High RAM area will be filled as best as it can by the OPTIMIZE program.

Board #3

This board has four extra pages (64K) of mappable memory above 640K (at B000, B400, D800, and DC00). The two pages at B000 and B400 are in the area normally used for a Monochrome adapter, but no adapter was installed, so the memory manager allows the area to be used. All four of the pages will be converted to High RAM. Note that, although there is 64K of High RAM, a program larger than about 30K CANNOT be loaded high since the 64K of memory is not in one piece. Once again (see the description for Board #2), the E000-EFFF area is not being used since most 80286 PCs cannot allow a memory board to occupy the addresses where a ROM socket also exists.

Board #4

This is actually not a board at all. This is the display after placing an All Computers CHARGECARD with ALLEMM4 or an 80386 or 80386SX conversion into the computer and using QEMM-386 as the expanded memory manager. There are many more pages now available, and the E000-EFFF area may now be used (the Page Frame has been placed there). Using a CHARGECARD or 80386 allows previously unmappable areas to be used since they can "map" pages before the computer's memory cards or motherboard have a chance to see the memory values used by the computer. Also note that the "backfilling" is much lower and yet there was no need to "disable" memory below 640K to achieve this. All of the areas above 640K will be converted to High RAM using either QRAM or QEMM-386's RAM parameter. (On a PS/2, the memory at E000 and above is not available.)

D

E

F

G

H

I

K

L

P

Q

X

'otes:

- If you're replacing printed materials, you need only send a photocopy of the bad page (or pages) with the error clearly annotated. If the entire manual is defective, send us an annotated photocopy that shows the major defect.

Before sending in a defective diskette or requesting replacement of printed materials, we encourage you to call Customer Support. We may be able to solve the problem for you over the phone or expedite the replacement. Customer Support is available Monday through Friday at:
(213) 392-9701, 9:00am through 5:00pm (Pacific Time)

However, please note that:

- You assume all risk of loss or damage to any returned materials while in transit.

- We reserve the right to revise the printed materials for this package and to make changes from time to time in the content thereof without any obligation to notify you (or any other person) of such revision or changes.

Covering the Software

What is Covered?
Significant errors in the software that make it unusable.

For How Long?
Ninety (90) days from the date of purchase or form the date of your receipt of the package.

What We Will Do
Make the effort (that we believe suitable to the nature and extent of the problem) to correct the error or to help you work around or avoid the error—or, at our opinion, authorize a refund of your license fee.

What You Must Do
To correct a suspected software error during the 90-day warranty period, you must:

- Complete the Quarterdeck Problem Report Card.

- Notify Customer Support by calling us at the number listed above.

Our Customer Support staff will attempt to correct your problem in accordance with our standard support policy and this Limited Warranty. Completion of the problem report prior to calling Customer Support will expedite handling of your problem. In the event a refund is authorized, Customer Support will provide you with the necessary information.

Alternately, you may mail in the Quarterdeck Problem Report card to the address given above.

OTHER LIMITATIONS

What We Will Not Do
This warranty does not cover material that has been lost, stolen, or damaged by accident, misuse, or unauthorized modifications.

No Other Warranty
We make no other warranty, express or implied, to you or any other person or entity. Specifically, we make no warranty that the software is fit for a particular purpose. Any implied warranty is merchantability is limited to the 90-day duration of this limited warranty and is otherwise expressly and specifically disclaimed.

No Liability for Consequential Damages

We will not be liable for special, incidental, consequential, indirect, or other similar damages, even if we or our agent have been advised of the possiblity of such damages.

This means that we are not responsible or liable for damages or costs incurred as a result of loss of time, loss of data, loss of profits or revenue, or loss of use of the software. In addition, we are not responsible or liable for damages or costs incurred in connection with obtaining subsitute software, claims by others, inconvenience, or similar costs.

Any Liability Not to Exceed Purchase Price

In no event will our liability for any damages to your or any other person ever exceed the price paid for the license to use the software, regardless of the form of the claim.

No Unauthorized Person Can Change the Terms of This Warranty
Additional statements by agents, employees, distributors, and dealers of Quarterdeck Office System—such as dealer advertising or presentations—do not constitute warranties by Quarterdeck Office Systems, do not bind Quarterdeck Office Systems, and should not be relied upon.

State Law Rights
This warranty gives you specific legal rights. You may have other rights which vary from state to state. Some states do not allow the exclusion of incidental or consequential damages, or the limitation on how long an implied warranty lasts. So, some of the above may not apply to you. This agreement will be governed by the laws of the State of California.

US Government Restricted Rights
The software and documentation are provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of The Rights in Technical Data and Computer Software clause at 52.227-7013. Contractor/manufacturer is Quarterdeck Office Systems, 150 Pico Blvd, Santa Monica, CA 90405.

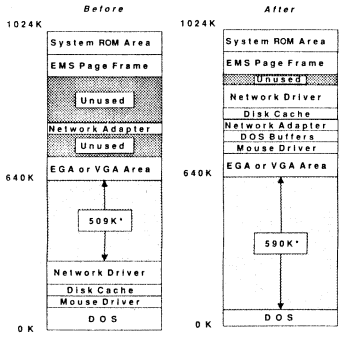
QRAM™

Memory is like gold—it's precious. Especially when you use a mouse, a network, and several terminate and stay resident (TSR) programs—all loaded before your favorite DOS program. QRAM helps ease RAM cram. For QRAM, a set of memory management utilities, tries its best, depending upon your hardware, to find unused, hidden memory that can be used by your programs.

QRAM is both a high memory (640-1024K) manager and an extended memory manager (compatible with the XMS extended memory specification). Its purpose is to give you ways to load drivers, such as networks, TSRs, DOS resources, and even parts of programs in memory addresses above 640K—leaving more memory below 640K for your DOS program. QRAM's ability to do this does depend upon your PC memory and processor hardware.

On PCs with only 640K, QRAM can help in just one way, and only sometimes. If you have an EGA or VGA adapter, and memory is more important than graphics, QRAM can give your DOS program 96K more memory. The caveat is that you can't run a graphics program while this memory is in use.

On PCs with EMS 4 or EEMS expanded memory hardware, or Chips & Technologies shadow RAM, QRAM lets you load and run programs in the memory area between 640-1024K (high memory). Depending upon your hardware, available memory in that area ranges from 30K to 130K.



* Example memory values only

FEATURES

- ❑ **XMS compatible extended memory management.**
- ❑ **High memory management.** For PCs with EMS 4/EEMS memory or shadow RAM, QRAM enables you to run device drivers and TSR's or load DOS resources in high memory (640-1024K).
- ❑ **DOS conventional memory extension.** Extends conventional memory by 96K (from 640K to 736K) for PCs with EGA/VGA graphic adapters, when graphics capabilities are not needed.
- ❑ **Low overhead.** Requires only 1.5K conventional memory.

SYSTEM REQUIREMENTS

8088, 8086, 80286 PCs. Use of high memory is only available when PC has EMS 4 or EEMS expanded memory or Chips & Technologies shadow RAM.

Operating System: PC DOS 2.0-4.x; MS DOS 2.0-4.x

Quarterdeck Office Systems
150 Pico Boulevard Santa Monica, CA 90405
(213) 392-9701
FAX: (213) 399-3802