DECEMBER 1993

NOVELL® RESEARCH

Workstation Memory Management: Using QEMM386 7.01, 386 To The MAX 7.0, and MS-DOS 6

Edward A. Liebing
Principal Technical Writer
Systems Research Department

This AppNote is the fourth in an ongoing serious on using various memory managers to maximize the amount of memory available for running applications. This installment looks at the use of Quarterdeck's QEMM386 7.01, Qualitas' 386 To The Max 7.0, and MemMaker in MS-DOS 6.0. It gives results from a test suite of various workstation configurations: basic NETX or VLM with ODI, TCP/IP or NetWare/IP connections, and a multimedia setup with a CD-ROM drive and sound board installed.

Previous AppNotes in This Series

Managing Memory in a DOS Workstation: Using Novell DOS 7 (Oct 93) Managing Memory in a DOS Workstation: Using MS-DOS 5.0 and Windows 3.1 (Oct 92)
Managing Memory in a DOS Workstation: Part 1 (Aug 92)

Disclaimer

Contents

Acknowledgements

Thanks to the following vendors for providing products and assistance for this AppNote:

Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
800-426-9400
206-882-8080

Qualitas
7101 Wisconsin Avenue, Suite 1386
Bethesda, MD 20814
800-676-0386
301-907-6700

Quarterdeck Office Systems
1901 Main St.
Santa Monica, CA 90405
800-354-3222
310-392-9851

Trademarks

Novell, the N design, and NetWare are registered trademarks of Novell, Inc. Internetwork Packet Exchange, NetWare Loadable Module, NLM, Virtual Loadable Module, and VLM are trademarks of Novell, Inc. IBM and OS/2 are registered trademarks of International

Introduction

We began the workstation management series with an Appnote on how memory management works for DOS workstations. This introduction explains the types of memory available to DOS: conventional memory, upper memory area, high memory area, expanded memory, and extended memory. (See Managing Memory in a DOS Workstation: Part 1 in the August 1992 NetWare Application Notes.)

The second AppNote in this series discusses the use of the memory manager options that come with MS-DOS 5.0 and Windows 3.1. In particular, it covers EMM386.EXE and HIMEM.SYS and shows how they can be used to get more conventional memory available to DOS applications. (See Managing Memory in a DOS Workstation: Using MS-DOS 5.0 and Windows 3.1 in the October 1992 NetWare Application Notes.)

The third AppNote covers the suite of memory managers integrated into Novell DOS 7: EMM386.EXE, EMMXMA.SYS, HIMEM.SYS, and DPMS.EXE. These, along with other DOS Protected Mode Interface capabilities, allow Novell DOS 7 to relocate DOS, TSRs, and other device drivers almost completely outside of conventional memory. (See Managing Memory in a DOS Workstation: Using Novell DOS 7 in the October 1993 NetWare Application Notes.)

This AppNote looks at two of the memory managers that can be purchased separately from a DOS or Windows package. We have chosen Quarterdeck's QEMM 386 7.01 and Qualitas' 386 To The MAX 7.0 because of their similar approaches to memory management. We'll also cover the MemMaker program included with MS-DOS 6.x, which you can run to set up a workstation's memory.

About the Memory Managers

The memory managers we have chosen to cover in this AppNote basically use the same memory areas for extending conventional memory: conventional memory, upper memory area (UMBs), high memory area (HMA), expanded memory, and extended memory. The methods they use for scanning the configuration and implementing memory management are similar. However, their approaches on what to do with the available memory, as well as their final results, can be different. This is especially true with QEMM 386 and 386MAX, as both take different approaches with their 7.x releases.

MS-DOS 6.0 and 6.2, and PC-DOS 6.1, come with a MemMaker utility that scans the workstation configuration and implements memory management without you having to figure it out yourself. Since MemMaker is included in the DOS package, it provides a good baseline to help you evaluate whether or not you need to purchase a separate memory manager.

This AppNote takes a look at each of these programs and highlights some of their main features. After discussing the programs, we'll introduce the configuration matrix we used to test these memory management programs and show how much conventional memory they leave you when applied to different network configurations.

QEMM 386 v7.01

One of the best memory managers that you can purchase separately is the Quarterdeck Expanded Memory Manager, or QEMM 386. The 7.01 upgrade to version 7 offers a number of additional features that earlier versions didn't. These include:

- An express installation process that can have you up and running in a short period of time
- The ability to load parts of DOS into high RAM (UMBs)
- Support for the Pentium processor

- Support for DPMI 0.9
- A new VIDRAM program for DOS applications running in Windows
- A feature that looks for adapter ROM and RAM and uses ROM locations after boot-up
- The ability to detect bus-mastering hard drive controllers
- Special manipulation of MS-DOS's DoubleSpace disk compression program
- A QSETUP program to troubleshoot CONFIG.SYS and AUTOEXEC.BAT file options

The OPTIMIZE Program

As with most memory management packages, QEMM 386 comes with a program designed to arrive at an optimal memory configuration. In QEMM 386, this program is called OPTIMIZE. OPTIMIZE follows a pattern similar to what the other memory managers do.

1. It first assesses the workstation's hardware and software configuration, then analyzes all the TSRs and drivers you are loading in your CONFIG.SYS and AUTOEXEC.BAT files. (The original files are backed up as *.QDK files, so that if anything goes wrong, QEMM can restore them.)
2. OPTIMIZE then runs the programs found in the CONFIG.SYS and AUTOEXEC.BAT files to see how large they are when they initially load as well as after they are loaded. OPTIMIZE tests them in memory to find a best fit.

   For those programs that are larger upon loading than they are when they run, OPTIMIZE uses a Squeeze feature to map over memory until the program loads. If you choose the Stealth ROM feature, OPTIMIZE tests for computer compatibility as a means to temporarily move ROM in order to get more UMBs to work with.
3. OPTIMIZE next restarts the computer to see if everything can load into its respective place in RAM and UMBs. You will see lines similar to the following attached to the TSRs it seeks to relocate out of conventional memory into specific upper memory regions:

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

   DEVICE=C:\QEMM\LOADHI.SYS /R:n
   ```
4. If everything works out, OPTIMIZE saves the configuration and reboots one more time, leaving you with its final configuration.

QEMM's Programs

QEMM's Express option (OPTIMIZE /Q) will give you the best settings for general use. But there are numerous other parameters and programs you can use to focus on its many options. Some of these programs are listed below.

Optimize /STEALTH. This option initiates the Stealth ROM feature, which first tests your hardware to ensure it is compatible with the Stealth ROM mapping method. If it is, QEMM will implement Stealth ROM into the procedure and you end up with an ST:M or ST:F parameter as part of the QEMM386.SYS statement.

Briefly, ST:M uses a mapping technique to move system, video, and disk ROM out of the first megabyte of memory. ROM calls are then handled through an EMS page frame. ST:F uses a page frame technique that places an EMS page frame on top of the ROM's address space. ROM calls are then executed as they normally do, but the EMS page frame is swapped in the ROM's address space when ROM calls are not being made.

DOS-Up. DOS-Up is a program that loads parts of DOS into UMBs above and beyond what gets loaded using the DOS=HIGH command. Specifically, DOS-Up loads 5KB of DOS data, 2.5KB more of the command processor, as well as FILES, STACKS, FCBS and LASTDRIVE statements that you designate the CONFIG.SYS file.

QSETUP. Suppose you have ALREADY installed QEMM, but now you have added something such as MS-DOS's DoubleSpace disk compression program. You can use QSETUP to add and change QEMM's parameters, enable or disable DPMI capabilities, enable or disable DOS-Up, and enable or disable Stealth DoubleSpace. You can also specify the Windows directory, look at QEMM hints and tips, and edit your CONFIG.SYS and AUTOEXEC.BAT files.

VIDRAM. This program takes the first 64KB of UMB area (A000-AFFF) that is unused by EGA and VGA monitors and makes that available to Text-based programs. VIDRAM can add another 32KB as well through the monochrome area (B000-BFFF). However, if you do this you can't run the monitors in EGA or VGA graphics mode. This means you can't run any programs that use graphics modes or has graphics capabilities. (You'll need to read the documents on when you can use this feature.)

QEMM Reports. By using QEMM.COM, you can see how QEMM is managing your computer's memory. These include a number of reports that you can see by typing such commands as:

```
QEMM SUMMARY <Enter>
        QEMM ACCESSED <Enter>
```

Example Configuration Files for TCP/IP and NWIP Connection

The following example illustrates the power of QEMM. In this example, I am loading TCP/IP and NetWare/IP instead of IPXODI. The difference between running NWIP and running just IPXODI (without normally running TCP/IP) is about 33-43KB overall; the difference between running NWIP and IPXODI while normally running TCP/IP is about 3KB overall. (NWIP is about 19KB when loaded and IPXODI is about 16KB when loaded--the difference of 3KB.)

To run NetWare/IP, you also need to run TCPIP.EXE, which is about 23KB loaded. Also, the size of LSL.COM becomes larger when you add the Link Support heading for TCP/IP. When you run the defaults, link support for TCP/IP will increase the loaded size of the LSL.COM program from 5KB to 22KB. If you add more memory for greater performance, the file can grow much larger.

In this example, I was able to load TCP/IP and NWIP and still have 594KB of conventional memory for running applications. Before I ran the Optimize program, I made a few additions through the QSETUP program. First, I selected P from the main menu in order to Review or change QEMM parameters. Inside that option, I left the Remove or set address of page frame option to Auto and I left the Fill upper memory with RAM option to Yes. I then answered Yes to the Copy ROMs to RAM option and set the "Stealth system and video ROMs" option to Mapping. (I used this setting because the initial installation process showed that I could use the ST:M setting.) Then I chose A to Accept these settings.

Back at the QEMM Setup Options window, I chose U to Enable or disable DOS-Up, to which I answered Yes. Back at the QEMM Setup Options window, I chose L, Enable or disable Stealth DoubleSpace and answered Yes to that option. I then chose S for Save configuration and exit. Because of these changes, QEMM invited me to run Optimize again.

Below are the example files and a brief explanation of what QEMM does to the CONFIG.SYS and AUTOEXEC.BAT files. I included the NET.CFG file because it also needs some explanation.

CONFIG.SYS

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

```
DEVICE=C:\QEMM\QEMM386.SYS RAM ROM ST:M ARAM=CB80-CBFF R:2
DEVICE=C:\QEMM\DOS-UP.SYS @C:\QEMM\DOS-UP.DAT
BUFFERS=30,0
FILES=65
LASTDRIVE=Z
FCBS=4,0
DEVICE=C:\QEMM\LOADHI.SYS /R:1 C:\DOS\SETVER.EXE
SHELL=C:\QEMM\LOADHI.COM /R:3 C:\DOS\COMMAND.COM C:\DOS\ /E:600 /p BREAK=ON
STACKS=9,128
DEVICE=C:\QEMM\LOADHI.SYS /R:1 C:\QEMM\ST-DBL.SYS
DOS=HIGH
```

In this example, DOSDATA.SYS helps prepare the system for DOS-Up and for loading DOS into upper memory and other places. On the QEMM386.SYS line, QEMM creates high RAM in upper memory (RAM) and copies ROM into RAM (ROM). The ST:M statement enables the Stealth ROM feature, excludes CB80-CBFF through the ARAM parameter, and places this information into memory region 2 (R:2). Next, DOS-Up is loaded to move some portions of DOS out of conventional memory.

After this, QEMM uses LOADHI to move different programs into different regions of upper memory, including SETVER, the Command processor, and ST-DBL.SYS (which takes the place of MS-DOS 6.x's DoubleSpace disk compression program). If you start having problems with ST-DBL.SYS, you can disable it through QSETUP and simply use DBLSPACE.SYS again.

AUTOEXEC.BAT

```
C:\QEMM\LOADHI /R:3 C:\DOS\SMARTDRV C 1024
PATH C:\DOS;C:\WIN31;C:\UTIL;C:\NET\BIN;C:\
SET TEMP=C:\WIN31\TEMP
PROMPT $P$G
SET WPC=/NT-1/U-EAL/PS-Y:\APPS\WP51\SETUP
C:\DOS\MOUSE
C:\QEMM\LOADHI /R:1 C:\NWCLIENT\LSL
C:\QEMM\LOADHI /R:2 C:\NWCLIENT\NE2000
C:\QEMM\LOADHI /R:3 C:\NET\BIN\TCPIP
CD\NWIP
C:\QEMM\LOADHI /R:3 NWIP
CD\
C:\QEMM\LOADHI /R:3 /LO C:\NWCLIENT\VLM /PS=SEIZURE
SET NAME=ELIEBING
BREAK ON
F:
LOGIN SEIZURE/ELIEBING
```

Here QEMM simply loads the various drivers into different memory regions. Once LSL, NE2000, and TCPIP are loaded, the AUTOEXEC.BAT loads NWIP.EXE, which is installed in the C:\NWIP directory.

Next, the DOS Requester is loaded and has a preferred server connection to server SEIZURE (/ps=seizure). This is followed by the LOGIN to the server..

NET.CFG

```
        Link Support
                Max Stacks 8
                Buffers 8 1500
                MemPool 4096

Link Driver NE2000
                INT 3
                PORT 300
```

```
                    MEM D0000
                    FRAME Ethernet_802.3
                    FRAME Ethernet_II
                    protocol ipx 0 Ethernet_802.3
                    protocol tcpip 8137 Ethernet_II

Protocol TCPIP
                    PATH SCRIPT      C:\NET\SCRIPT
                    PATH PROFILE      C:\NET\PROFILE
                    PATH LWP_CFG              C:\NET\HSTACC
                    PATH TCP_CFG              C:\NET\TCP
                    ip_router                137.65.96.254
                    ip_netmask           255.255.255.0
                    ip_address           137.65.96.10
                    TCP_SOCKETS 8
                    UDP_SOCKETS 8
                    RAW_SOCKETS 1
                    NB_COMMANDS 0
                    NB_ADAPTER  0

NWIP
                    NWIP_DOMAIN_NAME  NWIP.SED.PROVO.NOVELL.COM
                    NSQ_BROADCAST                   ON

SHOW DOTS ON
          NETWARE DOS REQUESTER
                    VLM STACK SWITCH = ON
            PREFERRED SERVER = SEIZURE
                    CHECKSUM = 0
                    LARGE INTERNET PACKETS = ON
                    CONNECTIONS = 16
                    FIRST NETWORK DRIVE = F
                    MESSAGE LEVEL = 3
                    SIGNATURE LEVEL = 0
```

The Link Support heading is automatically created for you if you run the installation process for LAN Workplace for DOS (LWP) or NetWare/IP (NWIP). Under this heading are three subheadings that must be indented: Max Stacks, Buffers, and MemPool. For TCP/IP to work, you need to have at least the minimum settings (Max Stacks 6, Buffers 4 600, and MemPool 4096). The default settings made by LWP and NWIP are reflected in the example above.

The Protocol TCPIP and NWIP headings are necessary to add the parameters for using TCP/IP and NetWare/IP. Neither is needed if you are running just IPX.

Note: The NET.CFG headings and settings needed for TCP/IP and NetWare/IP are discussed in greater detail under "The Link Support Heading" later in this AppNote. A much more detailed explanation of TCP/IP and NetWare/IP settings will be covered in a future AppNote.

Things Learned About QEMM During the Tests

QEMM gave some significant conventional memory numbers. It's a good idea to take time to run QSETUP before running the Optimize program. If you don't have DOS=HIGH in the CONFIG.SYS, QSETUP will add it if you enable DOS-Up in QSETUP. There are also some nice hints and tips about running QEMM found in QSETUP.

About the only problem I did encounter was an inordinate amount of C: drive errors when I was running C:\QEMM\ST-DBL.SYS instead of using DBLSPACE.SYS that comes with DOS 6.0. These problems went away when I went back to running DBLSPACE.SYS. I'm still not sure whether the problem stems from a bad disk sector, DoubleSpace, ST-DBL.SYS, or a combination of all three.

386 To The MAX v7.0

Another excellent memory management product that can be purchased is Qualitas' 386 To The MAX version 7.0. In their 7.0 releases, 386MAX takes a different slant from QEMM by focusing on the Windows platform as well as DOS sessions running under Windows. The upgrade to version 7 offers a number of additional features for DOS and Windows that its earlier versions didn't. These include:

- DOSMAX for Windows (for larger DOS sessions in Windows)
- Qualitas PIF Editor
- MultiConfig Maximize for MS-DOS 6.0 users
- ExtraDOS for moving more DOS out of conventional memory
- Qualitas Memory Tester
- Support for DPMI 0.9 and 1.0
- Support for Pentium processor
- Stack Overflow protection added

The Maximize Program

The Maximize program follows a similar pattern to QEMM's Optimize program.

1. In the first phase of installing Maximize, it examines how much memory is available and makes backup copies of the CONFIG.SYS and AUTOEXEC.BAT files. It next analyzes the TSRs you are loading in your CONFIG.SYS and AUTOEXEC.BAT files.
2. Maximize then runs the programs found in the CONFIG.SYS and AUTOEXEC.BAT files with its loader program called 386LOAD, which is used to see how large programs are when they are loading and running. Maximize reboots and tests each program in memory to find its size through the GETSIZE utility. Maximize then works through a number of configurations to see where the programs will work best in memory.
3. Maximize next restarts the computer to see if everything can load into its new designations in RAM and UMBs. You will see lines such as the following attached to the programs in the CONFIG.SYS and AUTOEXEC.BAT:

   ```
   C:\386MAX\LOAD.SYS size=nnnn prog=C:\program_name
   ```

   ```
   C:\386MAX\386LOAD size=nnnn prgreg=n prog=C:\program_name
   ```
4. If everything works out, Maximize saves the configuration leaves you at the DOS prompt.

386MAX's New Programs

386MAX version 7 comes with a number of new programs that needs to be mentioned. (However, there are a number of others that are not covered here.)

DOSMAX for Windows. DOSMAX is a Windows session memory manager that allows a DOS session to access up to 736KB or memory if the program is run in text mode. This program won't run with applications that jump to graphics mode while running.

Qualitas PIF Editor. The Qualitas PIF Editor allows for easier editing of PIF files. It also allows you to turn on and off DOSMAX from within the editor.

MultiConfig Maximize. MS-DOS 6.x comes with the ability to have multiple configurations within the CONFIG.SYS file, thus allowing you to choose which configuration you wish to boot up. MultiConfig Maximize fixes the problems that MS-DOS 6.0 had with its own memory tools.

ExtraDOS. This program moves a number of DOS parameters that you set in the CONFIG.SYS into UMBs. Since these are setable parameters in the CONFIG.SYS, any changes you make to them may necessitate running Maximize again to better accommodate the changes.

Example Configuration Files for a Multimedia Setup

This example shows 386MAX loading the DoubleSpace disk compression program, CD-ROM drivers, and SoundBlaster board drivers for multimedia presentations running in Windows. Maximize was able to get 557KB of conventional memory for applications after loading the above scenario.

Below are the example files along with explanations of some of the parameters you will see.

CONFIG.SYS

```
        BUFFERS=30,0
        FILES=65
        LASTDRIVE=Z
        FCBS=4,0
        DEVICE=C:\386MAX\386MAX.SYS PRO=C:\386MAX\386MAX.PRO
        DEVICE=C:\386MAX\386LOAD.SYS SIZE=11504 PRGREG=2 PROG=C:\DOS\SETVER.EXE
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /E:600 /P
        BREAK=ON
        STACKS=0,0
        DEVICE=C:\386MAX\386LOAD.SYS SIZE=44352 FLEXFRAME PROG=C:\DOS\DBLSPACE.SYS
/MOVE DOS=HIGH
        DEVICE=C:\386MAX\386LOAD.SYS SIZE=22656 PRGREG=2 PROG=C:\DEV\MDSCD_FD.SYS
/D:MSCD000 /N:1 DEVICE=C:\STW\SNDBK12.SYS 1 5 220 2
        REM MAXIMIZE: EXTRADOS MUST COME AT THE END OF CONFIG.SYS
        DEVICE=C:\386MAX\EXTRADOS.MAX PRO=C:\386MAX\EXTRADOS.PRO
        INSTALL=C:\386MAX\EXTRADOS.MAX
```

The 386MAX installation program adds a configuration profile of your workstation's CONFIG.SYS and AUTOEXEC.BAT files through the 386MAX.PRO file. The 386LOAD.SYS and 386LOAD.EXE programs are the ones that load the programs into UMBs.

The Size parameter specifies how much memory the program needs to load into memory; the PRGREG parameter places the programs into a memory area; the FLEXFRAME parameter uses the EMS frame area to load a particular program, which allots 64KB for loading purposes. This is quite useful for programs that look for 64KB or look for the program's size of contiguous upper memory in order to load high.

This particular example is loading DoubleSpace and the CD-ROM drivers into UMBs, while leaving the SoundBlaster's system file in conventional memory. The example then loads ExtraDOS to move as much of DOS out of conventional memory as possible. However, you may need to add the DOS=HIGH line into the CONFIG.SYS, as Maximize does not automatically add this. All the comments you see in this example were placed in the file by Maximize.

AUTOEXEC.BAT

```
        C:\DOS\SMARTDRV C 1024
        PATH C:\DOS;C:\WIN31;C:\UTIL;C:\;C:\NU
        SET NU=C:\NU
        IMAGE C
        SET TEMP=C:\WIN31\TEMP
        PROMPT $P$G
        SET WPC=/NT-1/U-EAL/PS-Y:\APPS\WP51\SETUP
        CALL C.BAT
        SET BLASTER=A220 I5 D1 H5 P330 T6
        SET SOUND=C:\SB16
        C:\SB16\SB16SET /M:220 /VOC:220 /CD:220 /MIDI:220 /LINE:220 /TREBLE:0
C:\SB16\SBCONFIG.EXE /S
        C:\386MAX\386LOAD SIZE=106080 PRGREG=2 PROG=C:\BIN\MSCDEX.EXE /E /D:MSCD000
/M:20 C:\386MAX\386LOAD SIZE=56976 PRGREG=2 PROG=C:\DOS\MOUSE
        C:\386MAX\386LOAD SIZE=22736 PRGREG=2 ENVREG=3 FLEXFRAME
PROG=C:\NWCLIENT\LSL C:\NWCLIENT\NE2000
        C:\NWCLIENT\IPXODI
        C:\NWCLIENT\VLM
        F:
        LOGIN GEORGIE/ED
```

Maximize was selective on what it put into UMBs. But when it was finished, there were only a few kilobytes of UMB space left which would then take some hand tweaking to use up. But the whole idea behind Maximize is to let 386MAX's memory management capabilities do the tweaking for you.

The results I got were from running the Express option in Maximize. Running the Full option and playing with different load orders will often yield different results. If you're not getting the numbers you think you should be, try a different order and rerun Maximize.

NET.CFG

```
        Link Driver NE2000
                INT 3
                PORT 300
                MEM D0000
                FRAME Ethernet_802.3

SHOW DOTS ON
        NETWARE DOS REQUESTER
                VLM STACK SWITCH = ON
          PREFERRED SERVER = GEORGIE
                CHECKSUM = 0
                LARGE INTERNET PACKETS = ON
                CONNECTIONS = 16
                FIRST NETWORK DRIVE = F
                MESSAGE LEVEL = 3
                SIGNATURE LEVEL = 0
```

If you're running only IPX and no other protocol, you can remove the Link Support heading and its parameters from the NET.CFG. IPX does not use any of the parameters found under this heading. Normally the Link Support layer is added in the NET.CFG when you install LAN WorkPlace for DOS and other such programs. This particular NET.CFG doesn't have any Link Support settings and leaves the loaded size of LSL.COM at 5KB. Adding TCP/IP or NetWare/IP default settings will increase the LSL file up to 22KB. Because of the size increase, this can dramatically affect how the memory management programs place this and other files in memory.

Things Learned About Maximize During the Tests

Using 386MAX was a bit trickier than the other memory managers and I experienced a number of problems running the program until I took off all past installations and remnants. After that, the program worked very well.

Another initial problem is the amount of memory that 386MAX needs to simply install. If you have too little memory because of everything you are loading, you won't be able to run the Install program. I had to bring the workstation up under another memory manager and then run Install.

386MAX keeps a profile of your present settings after you run Maximize. If you change your settings, 386MAX will tell you that your present configuration was different and that you should rerun Maximize. If the changes were too drastic, you may need to run Install with the /N and /R options in order for Maximize to work. The Install program then runs a memory manager stripper that lets you begin again with a fresh start.

While in normal use you won't normally face such drastic changes in your CONFIG.SYS and AUTOEXEC.BAT files, the procedure was much more time consuming in comparison to the other two programs.

Be sure to add the DOS=HIGH line in your CONFIG.SYS, because Maximize doesn't do this for you.

DOS 6.x and MemMaker

MS-DOS 6.0, PC-DOS 6.1, and MS-DOS 6.2 all come with newer versions of HIMEM.SYS and EMM386.EXE memory managers than previous versions of DOS. The 6.x version of DOS also comes with a memory management program called MemMaker. I include it here as a baseline for helping you decide when you do need to purchase a memory management program external to DOS. (Also note that the other memory management programs offer more than just memory management.)

MemMaker's Process

When you first type MemMaker, you will be asked to choose between express setup and a custom setup. Because I chose the express setups on the other memory managers, I did the same here. When you choose the Express setup, you are asked if you need EMS page frame capabilities. For our tests, I chose both options in order to relate the differences.

From here, MemMaker goes through a procedure similar to the other programs. MemMaker reboots the computer and runs the drivers and TSRs found in the CONFIG.SYS and AUTOEXEC .BAT files. MemMaker uses its SIZER program that looks to see how the programs are loading into memory and where to place them for future bootups. As items are loaded into memory, MemMaker then assesses where everything should fit.

MemMaker next restarts the computer to see if everything can load into their respective places in RAM and UMBs. MemMaker adds DEVICEHIGH= to additions in the CONFIG.SYS and attaches LH /L:n,nnnnnn to the TSRs it seeks to relocate out of conventional memory. MemMaker asks if everything worked, and then tells you what memory you have gained (in bytes, not in kilobytes). Pressing <Enter> saves the changes to the CONFIG.SYS and AUTOEXEC.BAT. If there are no gains or if you have less memory than when you began, you can press <Escape> to undo the changes.

MemMaker does not put the DOS=HIGH command into your CONFIG.SYS, so you will need to do this if you wish DOS to move into HMA. This action does give you at least 50KB more of conventional memory.

Example Configuration Files for Basic Setup

The network setup for this example is running IPXODI with the DOS Requester, one of the more basic installations that was tested. However, the example is also loading DoubleSpace disk compression program as well as SmartDrive for Windows caching. The end result was 599KB of conventional memory (without running expanded memory) for applications to run in. This example does not show a NET.CFG file as it used the same one that is in the Maximize example.

CONFIG.SYS

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS HIGHSCAN
BUFFERS=30,0
FILES=65
DOS=UMB
lastdrive=Z
FCBS=4,0
DEVICEHIGH /L:2,12048 =C:\DOS\SETVER.EXE
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /E:600 /p
BREAK=ON
STACKS=9,128
DEVICEHIGH /L:2,44304 =C:\DOS\DBLSPACE.SYS /MOVE
DOS=HIGH
```

In this example, MemMaker loads MS-DOS's two memory managers, HIMEM.SYS and EMM386.EXE. The EMM386 example shows that only an extended memory manger is being loaded (NOEMS) and that EMM386 is scanning all of upper memory (HIGHSCAN) for places to put programs.

You also need the DOS=UMB and DOS=HIGH lines in the CONFIG.SYS to (1) make the UMB area available to EMM386, and (2) to place DOS into HMA. MemMaker will add the DOS=UMB line as it adds the memory managers, but it does not automatically add the DOS=HIGH line. You'll need to do this yourself.

EMM386 uses DEVICEHIGH and LOADHIGH (LH) to place programs into UMBs. The /L:n,nnnnnn parameter places the programs into a memory area (/L:2). The set of numbers after the comma specifies how much memory the program needs to load into memory, such as 44304 for DoubleSpace.

AUTOEXEC.BAT

```
LH /L:0;2,45488 /S C:\DOS\SMARTDRV C 1024
PATH C:\DOS;C:\WIN31;C:\UTIL;C:\
SET TEMP=C:\WIN31\TEMP
PROMPT $P$G
SET WPC=/NT-1/U-EAL/PS-Y:\APPS\WP51\SETUP
LH /L:2,56928 C:\DOS\MOUSE
LH /L:2,9296 C:\NWCLIENT\LSL
LH /L:2,21696 C:\NWCLIENT\NE2000
LH /L:2,30576 C:\NWCLIENT\IPXODI
LH /L:0;2,54816 /S C:\NWCLIENT\VLM /PS=SRD
F:
LOGIN SRD/ELIEBING
```

In the AUTOEXEC.BAT file, everything was able to fit into UMBs to leave 599KB of conventional memory. Because the DOS Requester can be broken into three loading pieces (VLM.EXE, transient swap block, and global block), you see MemMaker responding to this capability by giving two memory areas. However, by taking out the LH command from VLM.EXE, I was able to get 8KB more of conventional memory.

This brings up an interesting point when using memory managers that perform memory placement for you. Once you get the hang of how these work, you can often do better without the memory management program. I have often taken off the /L: designations in MS-DOS 6.0 or added LOADHIGH and DEVICEHIGH statements to programs that MemMaker didn't and gotten more conventional memory than I did when running MemMaker. Depending on your time (and the number of workstation you have to work on), you may want to experiment a bit with the load order and LH and DEVICEHIGH statements.

Things Learned About MemMaker During the Tests

MemMaker follows lines very similar to QEMM and 386MAX, but I have often found that you could do as well manipulating the load order on your own than by using MemMaker. Still, MemMaker is a good free alternative to taking all that time manipulating memory management, and it comes with DOS.

Be sure to add DOS=HIGH in your CONFIG.SYS, because MemMaker doesn't do this for you.

Don't be satisfied with the amount of memory you initially receive when you run MemMaker. Sometimes just changing a few parameters or the load order can yield significant results. If MemMaker can deliver the kind of memory you need to run your applications, you won't need to buy anything else. For large businesses with hundreds or thousands of workstations, this can be an attractive advantage.

The Test Setup

The testing configuration matrix used for this AppNote is based on a DOS workstation running Windows on a NetWare network. It is not intended to compare the memory managers used, but rather to show you how much memory you can expect if you load certain workstation and NetWare files with and without memory mangers.

Note: In order to log in under NetWare 4.x, you need about 450KB of free conventional memory. This is because LOGIN.EXE loads Directory Services, internationaliz-ation tables, security authentication and other files. In all configurations used in this test, you wouldn't be able to log in to NetWare 4.x using the DOS Requester without running a memory manager.

All devices are loaded into conventional memory. The computer I used for these memory tests is a Compaq DeskPro 50M with 16MB of memory and 200MB hard disk drive, and has a NE2000 network interface board on Ethernet topology.

The Compaq is running MS-DOS 6.0 (MS-DOS 6.2 was in beta at the time and was not tested) with Windows 3.1 running locally, and is using MS-DOS 6.0's disk compression technology. The workstation is also running DOS's SmartDrive for better Windows performance. The Compaq has a SuperVGA monitor along with multimedia elements--an attached CD-ROM and SoundBlaster Pro card. The CD-ROM part of the system is only used in some of the later tests.

To provide a memory baseline, Examples 1 and 2 do not use any memory managers.

Baseline Example 1: NETX Shell

Let's first look at a basic setup running the NetWare shell (NETX). Example 1 contains the following parameters in the CONFIG.SYS and AUTOEXEC.BAT files:

CONFIG.SYS

```
    BUFFERS=30,0
    FILES=65
    LASTDRIVE=E
    FCBS=4,0
    DEVICE=C:\DOS\SETVER.EXE
    SHELL=C:\DOS\COMMAND.COM C:\DOS\ /E:600 /P
    BREAK=ON
    STACKS=9,128
    DEVICE=C:\DOS\DBLSPACE.SYS /MOVE
```

AUTOEXEC.BAT

```
    C:\DOS\SMARTDRV C 1024
    PATH C:\DOS;C:\WIN31;C:\UTIL;C:\
    SET TEMP=C:\WIN31\TEMP
    PROMPT $P$G
    C:\DOS\MOUSE
    C:\NETX\LSL
```

```
        C:\NETX\NE2000
        C:\NETX\IPXODI
        C:\NETX\NETX
        F:
        LOGIN SRD/ELIEBING
```

In Example 1, the LASTDRIVE statement is set to drive letter E, which lets the NetWare shell handle all the network drive mappings. Because I am running MS-DOS 6.0, I needed to run SETVER.EXE to incorporate application and NetWare utility compatibility. The machine is also using MS-DOS's DoubleSpace disk compression technology (the DBLSPACE.SYS statement). If you are not running a disk compression technology or don't need to run SETVER, you will see different numbers.

The AUTOEXEC.BAT file contains a SmartDrive statement that sets up the caching program for Windows. The C parameter turns off the write-deferred process to the local disk and the 1024 sets a 1MB memory limit for caching purposes. The PATH statement sets up the local environment for accessing DOS, Windows, and user preference utilities. (If your environment accesses these items from the network, don't place them in the local PATH statement--the login script should do this for you.)

The AUTOEXEC.BAT next loads the mouse driver for programs other than Windows. Windows loads a mouse driver of its own for applications and utilities that are written to run under Windows. However, you'll need to load the mouse driver for the DOS applications and utilities that you are running in a DOS session within Windows. Hence, the example is loading the mouse driver before entering Windows so it can be globally accessed throughout all DOS sessions in Windows.

The NetWare ODI drivers (LSL, NE2000, and IPXODI) are loaded next as well as the NetWare shell (NETX.COM). The user then logs into a NetWare server--in this instance, server SRD.

Explanation of NET.CFG Settings

The NET.CFG file I used for both examples is actually larger than it needs to be for half the tests, but it is designed to cover the entire testing suite.

Link Support Heading. In order for TCP/IP and NetWare/IP to work, you need to have a Link Support heading with at least the minimum settings of Max Stacks 6, Buffers 4 600, and MemPool 4096). This heading is automatically created for you if you run the installation process for LAN Workplace for DOS or NetWare/IP.

NET.CFG

```
        LINK SUPPORT
                MAX STACKS 8
                BUFFERS 8 1500
                MEMPOOL 4096

PROTOCOL TCPIP
                PATH SCRIPT      C:\NET\SCRIPT
                PATH PROFILE      C:\NET\PROFILE
                PATH LWP_CFG              C:\NET\HSTACC
                PATH TCP_CFG              C:\NET\TCP
                IP_ROUTER            137.65.96.254
                IP_NETMASK       255.255.255.0
                IP_ADDRESS       137.65.96.10
                TCP_SOCKETS 8
                UDP_SOCKETS 8
                RAW_SOCKETS 1
                NB_COMMANDS 0
                NB_ADAPTER  0
```

```
NWIP
                    NWIP_DOMAIN_NAME  NWIP.SED.PROVO.NOVELL.COM
                    NSQ_BROADCAST     ON

LINK DRIVER NE2000
                    INT 3
                    PORT 300
                    MEM D0000
                    FRAME ETHERNET_802.3
                    FRAME ETHERNET_II
                    PROTOCOL IPX 0 ETHERNET_802.3
                    PROTOCOL TCPIP 8137 ETHERNET_II

NETWARE DOS REQUESTER
                    CHECKSUM = 0
                    CONNECTIONS = 16
                    MESSAGE LEVEL = 3
                    SIGNATURE LEVEL = 0
                    FIRST NETWORK DRIVE = F
              PREFERRED SERVER = SEIZURE
             SHOW DOTS = ON
             FILE HANDLES = 65
```

The Link Support heading has three subheadings that must be indented: Max Stacks, Buffers, and MemPool.

- Max Stacks defaults to 4, but you really need 6 as a minimum. You can set it higher (this example shows 8).

- Buffers designates the number of buffers and the size of buffers used to receive packets from the network. In this example, we have 8 buffers set to a size of 1500 bytes for Ethernet. For Token-Ring, you would set this to 4202 or less, depending on the actual frame size (4212 maximum).

While you can set the buffer settings differently (to something like 4 and 600), smaller numbers can greatly affect performance and network traffic (and trigger a number of warnings as it loads). Smaller Buffer numbers mean a smaller LSL in memory. The example above gives you a 22KB LSL file, but if you used the minimum settings (Max Stacks 6, Buffers 4 600, and MemPool 4096), LSL would drop to 12KB and you will take a performance hit.

- MemPool is a required parameter for TCP but not for IPX. The bare minimum is 4096, but you can get better performance if you increase this number. Some people have found good performance at 5760, while others have needed 8 to 16KB to handle TCP/IP and/or other protocols on the same board.

Again, all these settings affect the overall load size of LSL, which in turn affects how NetWare loads using memory managers. So if you change these numbers in the NET.CFG file, you may need to rerun the memory manager. If you are running straight IPX, you don't need a Link Support heading. IPX doesn't use it and your loaded LSL program will be about 5KB without it.

Protocol TCPIP Heading. Protocol TCPIP heading contains some pointer-type information, such as Path Script, Path Profile, Path LWP_CFG, and Path TCP_CFG. These are used to tell the NET.CFG where to find the TCP/IP configuration files. NetWare stores the configuration files in this way because DOS doesn't follow the UNIX convention for storing this information. In earlier versions, you will see the directory path C:\XLN instead of C:\NET.

The IP_ROUTER, IP_NETMASK, and IP_ADDRESS entries need to be present somewhere. If they're not present here, the workstation can use the BOOTP option to get this information from a BOOTP server. The

BOOTP server then sends this information back to the workstation (based on the workstation's hardware address).

The TCP_SOCKETS entry shows the number of literal sockets which can be defined for TCP connection-oriented communications. The default is 8, but if you need more Telnet and FTP sessions that need a multiplexor port, you can increase the number of sockets to accommodate.

The UDP_SOCKETS entry is used less by LAN WorkPlace software, but it is used by NetWare/IP. However, NetWare/IP only uses 1, so 8 is a very healthy number. This entry defines connectionless protocol points of entry and destination. RAW_SOCKETS defaults to 1 (leave it at that) and is used for pinging host connections to ensure they are alive. If you are not performing some NetBIOS encapsulation, you won't need the NB_COMMANDS and NB_ADAPTER commands.

NWIP Heading. This is a new section for NetWare/IP. It has only two parameters. The NWIP_DOMAIN_NAME entry includes the name of the domain (spelled-out with dots), which looks like a virtual DNS domain. The domain name is used instead of name servers for NWIP DSS server lookups. Leave the NSQ_BROADCAST parameter set to ON.

Link Driver Heading. The Int and Port (and MEM and DMA) settings under the Link Driver heading must match the actual physical settings on the network interface card. In this case, the network adapter is set to the manufacturer defaults. Note that there are both Ethernet 802.3 and Ethernet_II frame type designations. This is necessary because the test environment is running both 3.1x servers (which default to 802.3) and 4.x servers (which default to support both 802.3 and 802.2 frame types) and I wanted ensure the workstation could log in to both.

ETHERNET_II is the traditional IP frame type. You don't have to use it, but if you are talking to any UNIX machine rather than just standard IPXODI machines, you'll want to have it. Ethernet_II is the standard frame for the UNIX world. Everybody takes it as a given, and that's why you see it in the test NET.CFG instead of a newer frame type.

With the frame types defined, we then set up two protocol statements: IPX 0 for 802.3 and TCPIP 8137 for Ethernet_II. Ethernet_II points it back to the frame type and 8137 is the Ether type for TCP/IP. It is a fixed value, so don't change it; otherwise nothing out there will recognize it.

NetWare DOS Requester Heading. This heading applies to both the DOS Requester and the NetWare shell, only because of the entries listed under it. While not all the entries need to be indented under this heading, doing so makes it easier to read and administer. The Checksum through Signature Level entries apply to the DOS Requester exclusively, while the rest of the entries apply to both the DOS Requester and the NetWare shell. (See Using the DOS Requester with NetWare 4.0 in the April 1993 NetWare Application Notes for more information.)

File Handles=65 Entry. For tests running NETX.COM, the NET.CFG file will need to contain a FILE HANDLES=nn statement for open files used by applications on network drives. This is especially important in the Windows environment. However, some applications, such as WordPerfect 5.1, also use DOS files locally (you need at least FILES=35 in your CONFIG.SYS).

If you set the FILE HANDLES statement in the NET.CFG to match the FILES statement in CONFIG.SYS, you should be fine. (Sixty-five is a good number that can cure many ills.) If you are running the DOS Requester only, you can remove this parameter. The DOS Requester uses DOS's FILES= entry from CONFIG.SYS. Since this is a mixed shell/requester test, I left it in and the DOS Requester ignores it.

Amount of Conventional Memory Left

Figure 1 shows the amount of conventional memory that is available for applications using the configuration in Example 1. This is the display you see when you type MEM /C <Enter> at the network prompt.

Figure 1: Display of memory used and available for baseline example 1 (without a memory manager).

```
Modules using memory below 1 MB:

Name         Total      =  Conventional  +  Upper Memory
        --------  ----------------  ----------------  ----------------
        MSDOS       69629   (68K)      69629   (68K)         0    (0K)
        SETVER        832    (1K)        832    (1K)         0    (0K)
        DBLSPACE    44992   (44K)      44992   (44K)         0    (0K)
        COMMAND      5344    (5K)       5344    (5K)         0    (0K)
        MOUSE       17088   (17K)      17088   (17K)         0    (0K)
        LSL         22448   (22K)      22448   (22K)         0    (0K)
        NE2000       5264    (5K)       5264    (5K)         0    (0K)
        IPXODI      16320   (16K)      16320   (16K)         0    (0K)
        NETX        43744   (43K)      43744   (43K)         0    (0K)
        Free       428464  (418K)     428464  (418K)         0    (0K)

Memory Summary:

Type of Memory        Total      =      Used       +      Free
        ----------------  -----------------  -----------------  ---------------
        Conventional       654336   (639K)     225872   (221K)     428464 (418K)
        Upper                   0     (0K)          0     (0K)          0 (0K)
        Adapter RAM/ROM         0     (0K)          0     (0K)          0 (0K)
        Extended (XMS)    15597568 (15232K)  15597568 (15232K)          0 (0K)
        ----------------  -----------------  -----------------  ---------------
        Total memory      16251904 (15871K)  15823440 (15453K)     428464 (418K)

Total under 1 MB   654336   (639K)     225872   (221K)     428464 (418K)

Largest executable program size         428000   (418K)
        Largest free upper memory block            0     (0K)
```

Baseline Example 2: VLMs

The second baseline follows the same setup, only it runs the DOS Requester (VLM.EXE) instead of the NetWare shell. Example 2 uses the following CONFIG.SYS and AUTOEXEC.BAT files:

CONFIG.SYS

```
        BUFFERS=30,0
        FILES=65
        LASTDRIVE=Z
        FCBS=4,0
        DEVICE=C:\DOS\SETVER.EXE
        SHELL=C:\DOS\COMMAND.COM C:\DOS\ /E:600 /P
        BREAK=ON
        STACKS=9,128
        DEVICE=C:\DOS\DBLSPACE.SYS /MOVE
```

AUTOEXEC.BAT

```
        C:\DOS\SMARTDRV C 1024
        PATH C:\DOS;C:\WIN31;C:\UTIL;C:\
        SET TEMP=C:\WIN31\TEMP
        PROMPT $P$G
        C:\DOS\MOUSE
        C:\NWCLIENT\LSL
        C:\NWCLIENT\NE2000
        C:\NWCLIENT\IPXODI
        C:\NWCLIENT\VLM
        F:
        LOGIN GEORGIE/ED
```

In this second CONFIG.SYS example, the only change is in the LASTDRIVE=Z statement. Since the DOS Requester uses DOS's network tables, setting LASTDRIVE=Z ensures you will have search drive mappings. Since the DOS Requester uses DOS's file handles instead of its own, the FILES=65 statement reflects the additional open files necessary for multiple applications open in Windows as well as network drive mappings. (You may want this number to be higher.)

The DOS Requester also uses DOS's environment settings for creating its environment, such as when you type MAP <Enter> at the network prompt. If the /E:600 setting is too low, the workstation won't be able to add drive mappings (you can set this as high as 4096, but between 800 and 1024 seems to be plenty).

The only change in the AUTOEXEC.BAT file is loading VLM.EXE instead of the NetWare shell. All other changes are handled in the NET.CFG file shown for Example 1.

Amount of Conventional Memory Left

When I typed MEM /C <Enter> at the network prompt, Example 2 displayed the numbers shown in Figure 2.

Figure 2: Display of memory used and available for baseline example 2 (without a memory manager).

```
Modules using memory below 1 MB:

Name       Total      =   Conventional   +   Upper Memory
        --------  ----------------  ----------------  ----------------
        MSDOS      71469   (70K)     71469   (70K)        0    (0K)
        SETVER       832    (1K)       832    (1K)        0    (0K)
        DBLSPACE   44992   (44K)     44992   (44K)        0    (0K)
        COMMAND     5344    (5K)      5344    (5K)        0    (0K)
        MOUSE      17088   (17K)     17088   (17K)        0    (0K)
        LSL        22448   (22K)     22448   (22K)        0    (0K)
        NE2000      5264    (5K)      5264    (5K)        0    (0K)
        IPXODI     16320   (16K)     16320   (16K)        0    (0K)
        VLM        90656   (89K)     90656   (89K)        0    (0K)
        Free      379712  (371K)    379712  (371K)        0    (0K)

Memory Summary:

Type of Memory       Total      =      Used      +      Free
        ----------------  -----------------  -----------------  ---------------
        Conventional       654336   (639K)     274624   (268K)    379712 (371K)
        Upper                   0     (0K)          0     (0K)         0   (0K)
        Adapter RAM/ROM         0     (0K)          0     (0K)         0   (0K)
        Extended (XMS)   15597568 (15232K)  15597568 (15232K)         0   (0K)
        ----------------  -----------------  -----------------  ---------------
        Total memory     16251904 (15871K)  15872192 (15500K)    379712 (371K)

Total under 1 MB   654336   (639K)    274624   (268K)    379712 (371K)

Largest executable program size        379248   (370K)
        Largest free upper memory block           0     (0K)
```

This particular setup can apply to either Bindery Services or Directory Services, depending on how much memory the workstation ends up with. This particular configuration left me with only 370KB of conventional memory, which isn't enough to log in to NetWare 4.x. You need at least 450KB free for that.

Memory Test Configurations

For my test suite, I modified the base CONFIG.SYS and AUTOEXEC.BAT files in order to load each the following configuration scenarios and their respective files:

- ODI and NETX; ODI and VLM
- ODI, NETX, and TCP/IP; ODI, VLM, and TCP/IP
- ODI, NETX, TCP/IP and NetWare/IP (without using IPXODI); ODI, VLM, TCP/IP and NetWare/IP (without using IPXODI)
- ODI, NETX, Toshiba CD-ROM and SoundBlaster Pro; ODI, VLM, Toshiba CD-ROM and SoundBlaster Pro

For each of these tests I was able to use the same CONFIG.SYS file as shown in Examples 1 and 2, except for the addition of the CD-ROM for the last two tests. With CD-ROM technology added, the CONFIG.SYS file looked like this:

CONFIG.SYS (with CD-ROM)

```
BUFFERS=30,0
FILES=65
DOS=UMB
LASTDRIVE=Z
FCBS=4,0
DEVICE=C:\DOS\SETVER.EXE
SHELL=C:\DOS\COMMAND.COM C:\DOS\ /E:600 /p
STACKS=9,128
DOS=HIGH
DEVICE=C:\DOS\DBLSPACE.SYS /MOVE
DEVICE=\DEV\MDSCD_FD.SYS /D:MSCD000 /N:1
DEVICE=C:\STW\SNDBK12.SYS 1 5 220 2
```

The AUTOEXEC.BAT files using the DOS Requester are as follows:

AUTOEXEC.BAT (for VLM and TCP/IP)

```
C:\DOS\SMARTDRV C 1024
PATH C:\DOS;C:\WIN31;C:\UTIL;C:\NET\BIN;C:\
SET TEMP=C:\WIN31\TEMP
PROMPT $P$G
C:\DOS\MOUSE
C:\NWCLIENT\LSL
C:\NWCLIENT\NE2000
C:\NWCLIENT\IPXODI
C:\NET\BIN\TCPIP
C:\NWCLIENT\VLM
F:
LOGIN SRD/ELIEBING
```

AUTOEXEC.BAT (for VLM, TCP/IP, and NWIP)

```
C:\DOS\SMARTDRV C 1024
PATH C:\DOS;C:\WIN31;C:\UTIL;C:\NET\BIN;C:\
SET TEMP=C:\WIN31\TEMP
PROMPT $P$G
C:\DOS\MOUSE
C:\NWCLIENT\LSL
```

```
        C:\NWCLIENT\NE2000
        C:\NET\BIN\TCPIP
        CD\NWIP
        NWIP
        CD\
        C:\NWCLIENT\VLM /PS=SEIZURE
        SET NAME=ELIEBING
        BREAK ON
        F:
        LOGIN SEIZURE/ELIEBING
```

AUTOEXEC.BAT (for VLM and CD-ROM)

```
        C:\DOS\SMARTDRV C 1024
        PATH C:\DOS;C:\WIN31;C:\UTIL;C:\
        SET TEMP=C:\WIN31\TEMP
        PROMPT $P$G
        SET BLASTER=A220 I5 D1 H5 P330 T6
        SET SOUND=C:\SB16
        C:\SB16\SB16SET /M:220 /VOC:220 /CD:220 /MIDI:220 /LINE:220 /TREBLE:0
C:\SB16\SBCONFIG.EXE /S
        C:\BIN\MSCDEX.EXE /E /D:MSCD000 /M:20
        C:\DOS\MOUSE
        C:\NWCLIENT\LSL
        C:\NWCLIENT\NE2000
        C:\NWCLIENT\IPXODI
        C:\NWCLIENT\VLM
        F:
        LOGIN SRD/ELIEBING
```

For your reference, here is a list of file sizes before and after loading for the TSRs and drivers mentioned in this AppNote:

File	Size Before Loading	Size After Loading
CONFIG.SYS		
SETVER.EXE	12,015	864
DBLSPACE.SYS	???	44,992
MDSCD_FD.SYS	22,647	15,920
AUTOEXEC.BAT		
SMARTDRV.EXE	45,145	27,504
MOUSE.COM	56,425	17,088
LSL.COM	8,780	22,448*
NE2000.COM	21,172	5,264
IPXODI.COM	30,051	16,320
VLM.EXE	35,471	49,712
TCPIP.EXE	40,120	24,160
NWIP.EXE	60,663	19,???
MSCDEX.EXE	25,431	27,952

*5,312 without TCP/IP link support

These numbers may not be reflective of the changes you can see as programs load, but they give you an idea of how files change when they are loaded. Also, changes to their configuration parameters will affect their overall size as well.

Memory Test Results

Figure 3 summarizes the results of testing each memory manager using each of the above scenarios.

Figure 3: Memory management test results.

Configuration	No Memory Mgrs.	QEMM386 7.01	386MAX 7.0	MS-DOS 6 (with EMS)	MS-DOS 6 (no EMS)
IPXODI, NETX	436KB	612KB	618KB	547KB	599KB
IPXODI, VLM	385KB	611KB	608KB	523KB	593KB
IPXODI, TCP/IP, NETX	396KB	629KB	578KB	511KB	582KB
IPXODI, TCP/IP, VLM	335KB	603KB	543KB	502KB	577KB
TCP/IP, NWIP, NETX	393KB	599KB	576KB	510KB	541KB
TCP/IP, NWIP, VLM	332KB	594KB	543KB	499KB	574KB
IPXODI, NETX, CD-ROM	326KB	617KB	546KB	493KB	519KB
IPXODI, VLM, CD-ROM	266KB	581KB	557KB	448KB	502KB

All numbers are subject to any changing of the conditions. Again, this test is not conclusive, but rather gives an idea of what kind of memory you can expect from the different memory managers. The results that you see very much dependent on the order in which you load programs, their overall size, and any number of other variables.

Summary

In today's workstation environments, running basic workstation elements for Windows and NetWare without a memory manager can leave you in memory troubles. This AppNote was not meant as an endorsement for any memory managers on the market, but rather it was written to give you an idea of what kind of memory you can expect from applying the different memory managers to NetWare configuration scenerios. The other criteria was that the memory manager needed to do the work so you didn't have to physically go in and perform the memory tweeking yourself.

All numbers presented here are subject to change by simply adding or subtracting parameters on the examples used. You may not need to load a number of the drivers mentioned in the article, such as local disk compression. But because of how memory hungry DOS and network basics are becoming, this AppNote gives you an idea of the need for memory managers in today's DOS environment.

With memory management discussed, future AppNotes will turn to other DOS workstation performance issues. Since users perceive NetWare through their workstation environment, we will look at how to get the best overall performance out of DOS and Windows workstations and NetWare applications, including software cache programs and hardware issues.