

# **INVISIBLE RAM 386 Shadow RAM Manager And Expanded Memory Manager For 386 And 486 Based PCs**

**Version 2.00**

© Copyright 1990 Invisible Software, Inc.

Web:

[www.invisiblesoft.com](http://www.invisiblesoft.com)

Email:

[invisoft@invisiblesoft.com](mailto:invisoft@invisiblesoft.com)



# License Agreement And Limited Warranty

The programs in this package are copyrighted. By using them, you indicate acceptance of the following terms and conditions. If you do not agree with these terms and conditions, return the programs to the place of purchase; and your money will be refunded.

**1. License.** You may use the programs on a single machine, and you may make a reasonable number of copies of the programs for backup purposes.

**2. Restrictions.** You may not use the programs on more than one machine at the same time. You may not distribute copies of the programs or documentation to others, or place them into any electronic bulletin board or information retrieval system. You may not modify or reverse engineer the programs.

**3. Limited Warranty on Diskettes.** If a diskette in this package is defective, Invisible Software will replace it at no charge if the defective diskette is returned to Invisible Software, shipping prepaid, within 90 days of the date of purchase.

**4. Limited Warranty on Software.** Invisible Software warrants that the programs will operate substantially as described in the documentation. If you report a significant software defect in writing within 90 days of the date of purchase, and Invisible Software is not able to correct the defect, then you may return the programs and documentation to the place of purchase; and your money will be refunded.

**5. Disclaimer.** IN NO EVENT WILL INVISIBLE SOFTWARE BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAMS EVEN IF INVISIBLE SOFTWARE OR AN AUTHORIZED REPRESENTATIVE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY. SOME STATES DO NOT ALLOW THE LIMITATION OR EXCLUSION OF LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU. This agreement will be governed by the laws of the State of California.

## Trademarks

Invisible RAM 386, Invisible RAM, Invisible EMS, Invisible Network, Invisible Ethernet, NET/30, and NET/30-EMS are trademarks of Invisible Software, Inc.

NEAT and AT/386 are trademarks of Chips and Technologies.

IBM, IBM PC, PC/XT, PC AT, PS/2, and Micro Channel are trademarks of International Business Machines Corporation.

Desqview is a trademark of Quarterdeck Office Systems.

Lotus 1-2-3 is a trademark of Lotus Development Corporation.

Ventura Publisher is a trademark of Xerox.

Microsoft Windows is a trademark of Microsoft Corporation.

Periscope is a trademark of The Periscope Company.

# Contents

About Invisible RAM 386 . . . . .	vii
About Invisible Software . . . . .	viii

---

## **1 What Is Invisible RAM 386? . . . . . 1-1**

What Invisible RAM 386 Does For You . . . . .	1-2
Hardware Requirements . . . . .	1-2
Getting Help . . . . .	1-3

---

## **2 Quick Software Installation . . . . . 2-1**

Software Modules . . . . .	2-1
The 80386 Control Program . . . . .	2-2
The Expanded Memory Manager . . . . .	2-2
Software Installation . . . . .	2-3
Configuration 0 — Expanded Memory Only . . . . .	2-4
Configuration 1 — Expanded Memory And Shadow RAM . . . . .	2-5
Configuration 2 — Shadow RAM Only . . . . .	2-5
Starting Invisible RAM 386 . . . . .	2-6
Bypassing Invisible RAM 386 . . . . .	2-6
Displaying DOS Memory Information . . . . .	2-7

---

## **3 Loading Memory- Resident Programs And Device Drivers Into Shadow RAM . . . . . 3-1**

Loading Memory-Resident Programs . . . . .	3-2
Loading Device Drivers . . . . .	3-4
Displaying Shadow RAM Usage . . . . .	3-5

---

## **4 Using VGA And EGA Graphics . . . . . 4-1**

Enabling VGA Or EGA Graphics . . . . .	4-2
Disabling VGA Or EGA Graphics . . . . .	4-2
Controlling The Default VGA Or EGA Graphics State . . . . .	4-3

---

## **5 Advanced Parameters For CP386.SYS . . . . . 5-1**

Software Installation . . . . .	5-2
/E — Extended Memory Allocation . . . . .	5-5
/H — High Memory Area XMS Interface . . . . .	5-6
/R — EMS Alternate Register Sets . . . . .	5-7
/S, /Z — Interrupt Stack Frame Allocation . . . . .	5-8
/O to /7 — DMA Data Buffer Allocation . . . . .	5-9

---

## **6 Advanced Parameters For C386EMM.SYS . . . . . 6-1**

Software Installation . . . . .	6-2
Page Frames . . . . .	6-6
/Z — System Configuration . . . . .	6-7
/I, /R, /S, /X — Memory Allocation Control . . . . .	6-7
The /X Parameter — Exclude Page Frames . . . . .	6-8
The /I Parameter — Create Paged EMS Memory . . . . .	6-9
The /S Parameter — Create Shadow RAM . . . . .	6-10
The /R parameter — Create High RAM . . . . .	6-11
/A, /K — Standard EMS Area Control . . . . .	6-12
The /A parameter — Standard EMS Area Start Address . . . . .	6-12
The /K Parameter — Standard EMS Area Size . . . . .	6-12
/F, /L, /V — DOS Memory Control . . . . .	6-13
The /F Parameter — Frontfill Size . . . . .	6-13
The /L Parameter — Force Driver Into Low Memory . . . . .	6-14
The /V Parameter — VGA/EGA Graphics Enable State . . . . .	6-14
/C, /H, /P — Internal Expanded Memory Parameters . . . . .	6-15
The /P Parameter — Create Non-Standard Page Frames . . . . .	6-15
The /H Parameter — Number of EMS Handles . . . . .	6-16

The /C Parameter — Number of EMS Contexts . . . . .	6-16
/T — Hardware Test . . . . .	6-17
The /T Parameter — Memory Test Level . . . . .	6-17

---

**A   Advanced Memory Configuration Information . . . . .   A-1**

Conventional Memory . . . . .	A-2
Extended Memory . . . . .	A-2
Expanded Memory . . . . .	A-3
Shadow RAM And High RAM . . . . .	A-5

---

**B   Displaying The Memory Map . . . . .   B-1**

Displaying Memory Information . . . . .	B-2
/D — DOS Memory Information . . . . .	B-3
/X — Expanded Memory Information . . . . .	B-4
/H — Expanded Memory Handles . . . . .	B-5
/E — Extended Memory Size . . . . .	B-5
/C — Chips And Technologies Chipset Information . . . . .	B-6
/N — NET/30-EMS Network Software Information . . . . .	B-9
/S — Shadow RAM Usage Information . . . . .	B-10
/M — Memory Map . . . . .	B-11

---

**C   Using LSHADOW With DOS 4.0 . . . . .   C-1**

---

**D   Error Messages For CP386.SYS . . . . .   D-1**

CP386 Initialization Error Messages . . . . .	D-1
CP386 Run-Time Error Messages . . . . .	D-4

---

<b>E</b>	<b>Error Messages For C386EMM.SYS . . . . .</b>	<b>E-1</b>
----------	---	------------

---

<b>F</b>	<b>Invisible RAM 386 Command Summary . . . . .</b>	<b>F-1</b>
----------	--	------------



# About Invisible RAM 386

---

Invisible RAM 386 is a software package that unlocks the extended memory that is present in many 80386, 80386SX, and 80486 based PCs.

Invisible RAM 386 increases the size of DOS memory from the normal 640K up to as much as 736K. Invisible RAM 386 also lets you load memory-resident programs and device drivers into shadow RAM, so that they do not use any DOS memory. As much as 224K of shadow RAM can be made available for memory-resident programs and device drivers.

With Invisible RAM 386, you can tap the power of shadow RAM to greatly increase the amount of RAM available for DOS applications.

In addition, Invisible RAM 386 is an expanded memory manager. Invisible RAM 386 uses the microprocessor's built-in memory management facilities to provide expanded memory compatible with EMS version 4.0. You can use the expanded memory to run EMS applications such as Lotus 1-2-3, Desqview, and Ventura Publisher. As much as 14 megabytes of expanded memory can be made available.

# About Invisible Software

---

Invisible Software, Inc., manufactures local area network hardware and software for PCs. With our local area network products, you can connect two or more PCs together to exchange data, share disks and printers, and run multiuser software.

We have two different lines of network boards: the Invisible Network, and the Invisible Ethernet. Boards are available for both the standard IBM PC/XT/AT/386 computers, and the IBM Micro Channel. All boards come complete with all required hardware and software; no extra purchases are necessary.

Invisible RAM technology was originally developed as part of our NET/30-EMS network operating system. NET/30-EMS solves the number one problem facing network users today: memory shortage. NET/30-EMS is the only network operating system which can load into expanded memory or shadow RAM. It can also increase the size of DOS memory, so that there is actually more free memory available after loading the network than before — as much as 630K free on a non-dedicated file server.

In response to many requests, we are making Invisible RAM technology available as a separate software package. Now all users of 386 and 486 based PCs can unleash the extra RAM hidden in their machines.

NET/30-EMS, complete with support for both expanded memory and shadow RAM, is included with all our network boards. If you have two or more PCs that you want to connect, contact Invisible Software for the latest information about our network products.

# 1 What Is Invisible RAM 386?

Invisible RAM 386 is a shadow RAM manager. It is a software package that provides additional memory to DOS applications. It also lets you load memory-resident programs outside of DOS memory.

In addition, Invisible RAM 386 is an expanded memory manager. It lets you run application programs that are designed to take advantage of expanded memory, without purchasing an expanded memory board.

Invisible RAM 386 works by turning on the memory-management functions that are built in to the 80386 and 80486 microprocessors. With these functions, your computer's unused extended memory can be converted into shadow RAM and expanded memory, thereby increasing the amount of memory available to DOS.

This chapter covers the following topics:

- What Invisible RAM 386 does for you
- Hardware requirements
- Getting help

# What Invisible RAM 386 Does For You

---

The Invisible RAM 386 software does three things for you:

- Invisible RAM 386 increases the size of DOS memory from the normal 640K up to as much as 736K.
- Invisible RAM 386 lets you load memory-resident programs and device drivers outside of DOS memory, so that they do not use up memory that you need for your application programs. As much as 224K of memory can be made available to memory-resident programs and device drivers outside of DOS memory.
- Invisible RAM 386 acts as an expanded memory manager, compatible with the expanded memory specification (EMS) version 4.0. You can make up to 14 megabytes of expanded memory available to applications such as Lotus 1-2-3, Desqview, and Ventura Publisher.

The exact amount of additional memory you get varies depending on what hardware you have installed in your computer. Invisible RAM 386 includes programs that let you determine what hardware you have installed, and configure the system to match your exact needs.

## Hardware Requirements

---

You do not need to purchase extra hardware to use Invisible RAM. All you need is a computer based on the 80386, 80386SX, or 80486 microprocessor. The computer must have at least 2 megabytes of memory installed.

# Getting Help

---

All Invisible RAM 386 programs have built-in help information. To get help, simply type the name of the program followed by a question mark. For example, to get help on the **SHADOW** command, type

**SHADOW ?**



# 2 Quick Software Installation

This chapter describes the basic procedures for installing the Invisible RAM 386 software. It is intended to get you up and running as quickly as possible. Later chapters describe the advanced features of the Invisible RAM 386 software.

## Software Modules

---

The Invisible RAM 386 software consist of two separate modules: an 80386 control program, and an expanded memory manager. You need to install both modules.

The following paragraphs describe the functions of each module.

## The 80386 Control Program

---

The 80386 control program is called **CP386.SYS**, and it is supplied on the Invisible RAM 386 diskette.

The function of the 80386 control program is to activate the memory management features that are built in to the 80386, 80386SX, and 80486 microprocessors. Once the memory management features are activated, they can be used by the expanded memory manager.

## The Expanded Memory Manager

---

The expanded memory manager is called **C386EMM.SYS**, and it is supplied on the Invisible RAM 386 diskette.

The expanded memory manager has three basic functions:

- Provide access to expanded memory (up to 14 megabytes) for specially-written application programs, such as Lotus 1-2-3, Desqview, Ventura Publisher, and NET/30.
- Create shadow RAM that can be used to hold ordinary memory-resident programs (TSRs) and device drivers, thereby freeing up more memory for DOS.
- Increase the size of DOS memory beyond the normal 640K limit, to as much as 736K on a color system or 704K on a monochrome system.

In addition, the expanded memory manager offers many optional advanced functions, which are described in Chapter 6.



# Software Installation

---

First, copy files CP386.SYS and C386EMM.SYS from the Invisible RAM 386 diskette to the root directory on the disk that you use to boot your system. Then install it as shown.

To install Invisible RAM 386, place the following lines into the CONFIG.SYS file, in the order shown:

```
STACKS=0,0  
DEVICE=CP386.SYS [/E=size]  
DEVICE=C386EMM.SYS [/Z=config]
```

where the parameters have the following meaning:

<b>/E=<i>size</i></b>	Extended memory size. The value of <i>size</i> is the number of kilobytes of memory that you want to reserve for extended memory applications, such as VDISK. The square brackets indicate that /E is optional; if omitted, it defaults to 0.
<b>/Z=<i>config</i></b>	System configuration. The only allowed values for <i>config</i> are 0, 1, and 2. The meaning of this parameter is described below. The square brackets indicate that /Z is optional; if omitted, it defaults to /Z=0.

**Note** — If you are using DOS version 3.1 or earlier, omit the **STACKS** command.

**IMPORTANT** — If you use any type of disk cache software (such as IBMCACHE or PC-KWIK), you must install the disk cache *after* installing Invisible RAM 386.

**Note** — If you want to use an extended memory application program, such as VDISK, you should install the extended memory application

*after* installing Invisible RAM 386. For example, suppose you want to use VDISK to create a RAMDISK with 512K capacity. You would install VDISK *after* installing Invisible RAM 386, as follows:

```
STACKS=0,0  
DEVICE=CP386.SYS /E=512  
DEVICE=C386EMM.SYS  
DEVICE=VDISK.SYS 512 /E:8
```

In the second line, the parameter **/E=512** reserves 512K of extended memory for use by extended memory applications. In the last line, the parameter **512** tells VDISK to create a RAMDISK with a capacity of 512K, while the parameter **/E:8** tells VDISK to use extended memory. (For information about VDISK, refer to the DOS command reference manual.)

As indicated above, the **/Z** parameter selects among three different system configurations. The following sections describe the three configurations.

**Note** — There are additional parameters that select other advanced functions, and allow you to customize the configuration to your exact requirements. The additional parameters are described in Chapters 5 and 6.

## **Configuration 0 — Expanded Memory Only**

---

In configuration 0, Invisible RAM 386 provides expanded memory, but no shadow RAM. You can run programs designed for expanded memory (like Lotus and Desqview), but you can't load memory-resident programs and device drivers outside DOS memory.

This configuration is the “standard” expanded memory configuration provided by most expanded memory managers.

Configuration 0 is selected with the parameter **/Z=0**, as shown below. If you omit **/Z**, it defaults to configuration 0.

```
STACKS=0,0  
DEVICE=CP386.SYS  
DEVICE=C386EMM.SYS /Z=0
```

## Configuration 1 — Expanded Memory And Shadow RAM

---

In configuration 1, Invisible RAM 386 provides both expanded memory and shadow RAM. You can run programs designed for expanded memory (like Lotus and Desqview), and you can also load memory-resident programs and device drivers outside DOS memory.

Configuration 1 is selected with the parameter **/Z=1**, as shown below.

```
STACKS=0,0  
DEVICE=CP386.SYS  
DEVICE=C386EMM.SYS /Z=1
```

## Configuration 2 — Shadow RAM Only

---

In configuration 2, Invisible RAM 386 provides shadow RAM, but no expanded memory. You can load memory-resident programs and device drivers outside DOS memory, but you can't run application programs that require expanded memory.

Configuration 2 provides an additional 64K of shadow RAM as compared to configuration 1.

Configuration 2 is selected with the parameter **/Z=2**, as shown below.

```
STACKS=0,0  
DEVICE=CP386.SYS  
DEVICE=C386EMM.SYS /Z=2
```

# Starting Invisible RAM 386

---

Invisible RAM 386 starts automatically when you boot the system. To start it now, either turn on the computer, or press **Ctrl-Alt-Del** if the computer is already on.

When Invisible RAM 386 starts, it performs a test of all the expanded memory and shadow RAM. At the end of the test, your computer continues to boot up as usual.

On many systems, Invisible RAM 386 automatically increases the size of DOS memory to 704K (for a monochrome system) or 736K (for a color system). This process is called “frontfill”. For additional information about frontfill, see Chapter 6 and Appendix A.

**NOTE** — If you have an EGA or VGA video system, you need to run a program called **VGAOFF** in order to increase DOS memory size above 640K. Refer to Chapter 4 for details.

**NOTE** — Invisible RAM 386 reserves approximately 256K of extended memory for its own use. If you include the **/E** parameter, the amount of extended memory specified by **/E** is also reserved. All remaining extended memory is converted into expanded memory and shadow RAM. The memory total displayed by the memory test is the amount of available expanded memory.

## Bypassing Invisible RAM 386

---

You can prevent Invisible RAM 386 from installing itself by pressing the **Ctrl-Alt-Shift** keys. You must press **Ctrl-Alt-Shift** before the Invisible RAM 386 message appears on the screen, and you must hold the keys down until both software modules tell you they are not

installing.

This feature is useful if you accidentally give incorrect parameters in the **DEVICE=CP386.SYS** or **DEVICE=C386EMM.SYS** statement. Sometimes, incorrect parameters can prevent your system from booting. By pressing **Ctrl-Alt-Shift**, you should be able to boot the system. (You can always boot the system successfully by turning off the computer, placing your original DOS diskette in drive A:, and then turning on the computer.)

## Displaying DOS Memory Information

---

You can use the program called **SHADOW.EXE** to display information about DOS memory. This shows you how much additional memory you obtained as a result of the frontfill feature of Invisible RAM 386.

To display DOS memory information, type

```
SHADOW /D
```

The program produces a display similar to the following:

```
DOS version number:  3.30.  
Total DOS memory:   736K.  
Free DOS memory:    638K.
```

The first line tells you which version of DOS you are using.

The second line shows the total amount of memory available for DOS.

The third line shows how much of the DOS memory is currently

available for use.

**NOTE** — If you have an EGA or VGA video system, you need to run a program called **VGAOFF** in order to increase DOS memory size above 640K. Refer to Chapter 4 for details.

# **3 Loading Memory-Resident Programs And Device Drivers Into Shadow RAM**

Many computer users like to use memory-resident programs (also called terminate-stay-resident programs or TSRs). The memory-resident programs include pop-up notepads, calculators, calendars, telephone dialers, spelling checkers, and keyboard enhancers, as well as network software.

A problem that many users encounter is that the memory-resident programs may use a lot of memory, especially if several are installed. After loading all your memory-resident programs, you may not have enough DOS memory left to run your application software.

Invisible RAM 386 lets you load memory-resident programs into shadow RAM, so that they do not use any DOS memory. Depending on your system configuration, you may have as much as 224K of shadow RAM available for loading memory-resident programs.

This chapter covers:

- Loading memory-resident programs
- Loading device drivers
- Displaying shadow RAM usage

## Loading Memory-Resident Programs

---

Invisible RAM 386 includes a program called **LSHADOW.COM** which is used to load memory-resident programs into shadow RAM.

To load a program into shadow RAM, type

**LSHADOW** [/A] *filename* [*parameters*]

The parameters have the following meanings:

**/A**                      An optional parameter that alters the behavior of **LSHADOW**, and is required for some programs. Most programs do not require **/A**.

***filename***              The name of the memory-resident program. You must specify the complete file name of the program, including the .COM or .EXE extension. If the program file is not in the current directory, you should include the complete drive and path specification.



## *parameters*

Optional command-line parameters that you want to specify for the memory-resident program. The meaning of these parameters varies depending on the program. Type the same parameters that you would use if you were loading the program from the DOS command line.

**Note** — **LSHADOW** does not use the DOS PATH to find program files. If the memory-resident program is not in the current directory, you must include a path specification with the filename.

**Note** — In order to use **LSHADOW**, you must tell the expanded memory manager (**C386EMM.SYS**) to create some shadow RAM, using either the **/Z** or **/S** parameter.

**Example 1** — You have a memory-resident program called **SK.COM**. To load the program into shadow RAM, type

```
LSHADOW SK.COM
```

**Example 2** — You have memory-resident network software called **SERVER.EXE**, which is installed on your hard disk in directory **C:\NET30**. To load the program into shadow RAM, type

```
LSHADOW C:\NET30\SERVER.EXE
```

**Example 3** — You want to install the DOS print spooler program, **PRINT.COM**, which is located in directory **C:\DOS**. You want to include parameters to tell the print spooler to use printer **LPT2** and a print buffer size of 1000 bytes. Type

```
LSHADOW C:\DOS\PRINT.COM /D:LPT2 /B:1000
```

In this example, **/D:LPT2** and **/B:1000** are parameters that are passed to the **PRINT.COM** program. Refer to the DOS Reference Manual for detailed information on the **PRINT.COM** program.

**Example 4** — For this example, suppose that the memory-resident program **UTILITY.COM** does not load successfully without the **/A**

parameter. To load it with the **/A** parameter, type

```
LSHADOW /A UTILITY.COM
```

## Loading Device Drivers

---

Invisible RAM 386 includes a program called **LSHADOW.SYS** which is used to load device drivers into shadow RAM.

To load a device driver into shadow RAM, place the following line into the CONFIG.SYS file:

```
DEVICE=LSHADOW.SYS [/A] filename [parameters]
```

The parameters have the following meanings:

<b>/A</b>	An optional parameter that alters the behavior of <b>LSHADOW.SYS</b> , and is required for some device drivers. Most device drivers do not require <b>/A</b> .
<b><i>filename</i></b>	The name of the device driver. You must specify the complete file name of the device driver, including the .SYS extension. If the file is not in the root directory of your boot disk, you should include a drive and path.
<b><i>parameters</i></b>	Optional command-line parameters that you want to specify for the device driver. The meaning of these parameters varies depending on the device driver. Use the same parameters that you would use if you were loading the device driver into DOS memory.

**Note** — You must load **CP386.SYS** and **C386EMM.SYS** before using **LSHADOW.SYS**.

**Note** — In order to use **LSHADOW.SYS**, you must tell **C386EMM.SYS** to create some shadow RAM, using either the **/Z** or **/S** parameter.

**Note** — **LSHADOW.SYS** uses approximately 120 bytes of DOS memory. If the device driver you are loading is smaller than 120 bytes, you are better off to simply load it into DOS memory.

**Example 1** — To load the device driver **ANSI.SYS** into shadow RAM, use the statement

```
DEVICE=LSHADOW.SYS ANSI.SYS
```

**Example 2** — You have a mouse driver called **MOUSE.SYS** which is on your hard disk in directory C:\MOUSE. To load it in shadow RAM, use the statement

```
DEVICE=LSHADOW.SYS C:\MOUSE\MOUSE.SYS
```

**Example 3** — You want to install the IBM DOS RAMDISK program, **VDISK.SYS**, which is located in directory C:\DOS. **VDISK** requires the **/A** parameter. Use the statement shown below. In this example, 128 is a parameter passed to **VDISK**.

```
DEVICE=LSHADOW.SYS /A C:\DOS\VDISK.SYS 128
```

## Displaying Shadow RAM Usage

---

You can use the **SHADOW** program to display the usage of shadow RAM.

To display the usage of shadow RAM, type

**SHADOW /S**

**SHADOW** displays information similar to the following:

```
Shadow RAM Manager: Installed.  
Total shadow RAM: 192K.  
Free shadow RAM: 84K.  
Largest loadable program size: 68K
```

The first line indicates that **C386EMM.SYS** is configured to support shadow RAM.

The second line displays the total amount of shadow RAM that was created by **C386EMM.SYS**. This figure includes all shadow RAM available for use by **LSHADOW**, plus all RAM used to frontfill DOS memory.

The third line displays the amount of shadow RAM that has not yet been used, and is available for loading additional memory-resident programs.

The fourth line indicates the largest single program that can be loaded into the remaining free shadow RAM.

**Note** — The **SHADOW** program can display a great amount of additional information about the memory in your computer. Refer to Appendix B for details.

# 4 Using VGA And EGA Graphics

Invisible RAM 386 increases the size of DOS memory beyond 640K. This presents a problem if you have a VGA or EGA video system, because the high-resolution graphics modes use the same memory that is required to increase DOS memory.

Invisible RAM 386 deals with this problem by allowing you to disable the high-resolution graphics modes of the VGA or EGA video system. When you disable high-resolution graphics, DOS memory is increased to 736K, but you can only use the text mode and the low-resolution graphics modes (also called the CGA graphics modes).

When you enable high-resolution graphics, DOS memory is limited to a maximum of 640K.

This chapter describes:

- Enabling VGA or EGA high-resolution graphics
- Disabling VGA or EGA high-resolution graphics
- Controlling the default VGA or EGA graphics state

## Enabling VGA Or EGA Graphics

---

To enable the high-resolution graphics modes of a VGA or EGA video system, type

**VGAON**

After running **VGAON**, you can use the high-resolution graphics modes. **VGAON** reduces the size of DOS memory down to 640K.

## Disabling VGA Or EGA Graphics

---

To disable the high-resolution graphics modes of a VGA or EGA video system, type

**VGAOFF**

After running **VGAOFF**, you cannot use the high-resolution graphics modes. **VGAOFF** increases the size of DOS memory as much as possible, usually to 736K.

# Controlling The Default VGA Or EGA Graphics State

---

Normally, the VGA or EGA high-resolution graphics are enabled when **C386EMM.SYS** is first loaded. In order to disable high-resolution graphics and increase DOS memory size above 640K, you must run the **VGAOFF** program.

If you wish, you may instruct **C386EMM.SYS** to disable the high-resolution graphics as soon as it loads. Then, DOS memory size is immediately increased above 640K. You can run the **VGAON** program to enable the high-resolution graphics whenever you want to.

To disable the VGA or EGA high-resolution graphics at the time **C386EMM.SYS** is loaded, include the **/V=0** parameter, as shown:

```
STACKS=0,0  
DEVICE=CP386.SYS  
DEVICE=C386EMM.SYS /V=0
```





# 5 Advanced Parameters For CP386.SYS

This chapter describes the the advanced features supported by the 80386 control program, **CP386.SYS**. The advanced features include:

- Allocating extended memory
- Installing the XMS interface for the high memory area
- Allocating EMS register sets
- Allocating stack space for interrupt processing
- Allocating buffers for DMA

These parameter are fairly technical in nature. Except for allocating extended memory, you will probably not have to change these parameters from their default values.

You will probably find that the advanced parameters for **C386EMM** (which are described in the next chapter) are much more useful than the advanced parameters for **CP386**. Therefore, on a first reading of this manual, you may want to skip this chapter.

# Software Installation

---

The following shows all the parameters available when installing the 80386 control program.

To install the 80386 control program, place the following lines into the CONFIG.SYS file, in the order shown:

```
STACKS=0,0  
DEVICE=CP386.SYS [parameters]
```

(Exception — If you are using DOS version 3.1 or earlier, omit the STACKS line.)

The optional *parameters* may include the following:

<b>/E=</b> <i>size</i>	Extended memory allocation. With this parameter, you can reserve part of your computer's extended memory for use by extended memory programs such as VDISK.SYS. <i>size</i> is a decimal number that gives the desired amount of extended memory, in kilobytes. The remaining extended memory not reserved by <b>/E</b> is used for expanded (EMS) memory, shadow RAM, and the <b>CP386</b> program code. If <b>/E</b> is omitted, it defaults to 0 if <b>/H</b> is not included, or 64 if <b>/H</b> is included.
------------------------	---

**/H [=hmamin]**

High memory area. If you include **/H**, then programs can access the high memory area (i.e., the first 64K of extended memory) using the Microsoft XMS interface. This is equivalent to Microsoft's HIMEM.SYS device driver. **hmamin** is the minimum size program, in kilobytes, that is allowed to use the high memory area; its value can range from 0 to 63 (this is equivalent to the **/HMAXIN=** parameter of HIMEM.SYS). The square brackets indicate that **hmamin** is optional; if omitted, it defaults to 0. If the **/H** parameter is completely omitted, then **CP386** does not supply an XMS interface.

**/R=regsets**

Number of expanded memory (EMS) alternate register sets. **regsets** is a decimal number giving the desired number of alternate register sets. These can be used by multitasking systems (such as Desqview) to speed up multitasking. Each register set uses 128 bytes of extended memory. The minimum allowed value of **regsets** is 0, and the maximum is 255. If **/R** is omitted, the default is 7.

**/S=stacks**

Number of stack frames used for interrupt processing. **stacks** is a decimal number giving the desired number of stack frames. This is equivalent to the first number in the DOS STACKS command. Each stack frame uses 512 bytes of extended memory, plus the number of bytes of DOS memory specified in the **/Z** parameter. The minimum allowed value of **stacks** is 8, and the maximum is 64. If **/S** is omitted, the default is 16.

***/Z=stacksize***

Size of stack frames used for interrupt processing. ***stacksize*** is a decimal number giving the desired size of the interrupt stack frames, in bytes. This is equivalent to the second number in the DOS STACKS command. ***/Z*** determines the amount of DOS memory used by each interrupt stack frame; the number of stack frames is determined by the ***/S*** parameter. The minimum allowed value of ***stacksize*** is 32, and the maximum is 512. If ***/Z*** is omitted, the default is 128.

***/0=DMAsize***

The size of the data buffer for DMA channel 0, in kilobytes.

***/1=DMAsize***

The size of the data buffer for DMA channel 1, in kilobytes.

***/2=DMAsize***

The size of the data buffer for DMA channel 2, in kilobytes.

***/3=DMAsize***

The size of the data buffer for DMA channel 3, in kilobytes.

***/4=DMAsize***

The size of the data buffer for DMA channel 4, in kilobytes.

***/5=DMAsize***

The size of the data buffer for DMA channel 5, in kilobytes.

***/6=DMAsize***

The size of the data buffer for DMA channel 6, in kilobytes.

***/7=DMAsize***

The size of the data buffer for DMA channel 7, in kilobytes.

The following sections give additional details and examples of the use of the parameters.

## /E — Extended Memory Allocation

---

The **/E** parameter lets you reserve part of your extended memory for use by extended memory applications, such as the VDISK RAMDISK program. The remainder of your extended memory is converted into expanded memory and shadow RAM.

The extended memory reserved by **/E** begins at segment address 10000, immediately above the first megabyte of memory. For example, suppose you have 3 megabytes of extended memory, occupying segments 10000 through 3FFFF. If you specify **/E=1024**, then segments 10000 through 1FFFF are reserved for extended memory applications, while segments 20000 through 3FFFF are converted into expanded memory and shadow RAM.

**Example** — To reserve 512K of extended memory, use

```
STACKS=0,0
DEVICE=CP386.SYS /E=512
DEVICE=C386EMM.SYS
```

In this example, you could use VDISK to make a 512K RAMDISK as follows:

```
STACKS=0,0
DEVICE=CP386.SYS /E=512
DEVICE=C386EMM.SYS
DEVICE=VDISK.SYS 512 128 256 /E:8
```

On the VDISK command line, **512** is the size of the RAMDISK (in kilobytes), **128** is the sector size (in bytes), **256** is the number of directory entries, and **/E:8** tells VDISK to use extended memory.

**Technical note** — The reserved extended memory works with programs that access extended memory through interrupt INT 15H functions 87H and 88H, or through DMA. It does not work with programs that actually switch the processor into protected mode.

## /H — High Memory Area XMS Interface

---

The **/H** parameter allows programs to access the high memory area using the Microsoft XMS interface.

The *high memory area* is the first 64K of extended memory. This area of extended memory is special, because DOS programs can access it directly. Certain DOS applications, such as Microsoft Windows, are designed to make use of the high memory area.

Microsoft has developed an interface specification, called *XMS*, that allows programs to use the high memory area. If you include the **/H** parameter, **CP386** installs an XMS interface; this allows programs such as Microsoft Windows to utilize the high memory area. If you omit the **/H** parameter, the XMS interface is not installed.

**Example** — To install the XMS interface, use

```
STACKS=0,0  
DEVICE=CP386.SYS /H  
DEVICE=C386EMM.SYS
```

Only one program at a time can use the high memory area. For this reason, the **/H** parameter can include a number which indicates the minimum size, in kilobytes, that a program must have in order to use the high memory area. This allows you to exclude small programs from the high memory area, thereby ensuring that big programs can obtain access to the high memory area. This can optimize your use of memory.

**Example** — Suppose that you want to limit the use of the high memory area to programs that use at least 48K out of the 64K available. You would use

```
STACKS=0,0  
DEVICE=CP386.SYS /H=48  
DEVICE=C386EMM.SYS
```

**Note** — Microsoft distributes a device driver called **HIMEM.SYS** that provides access to the high memory area. You cannot use **HIMEM.SYS** with **CP386**. The **/H** parameter is a *replacement* for the **HIMEM.SYS**

device driver. The number included with the **/H** parameter is equivalent to the **/HMAMIN=** parameter supported by HIMEM.SYS.

**Note** — When you include the **/H** parameter, **CP386** automatically allocates 64K of extended memory. It is not necessary to use **/E** to allocate extended memory for use as the high memory area.

## **/R — EMS Alternate Register Sets**

---

The **/R** parameter selects the number of alternate register sets available to expanded memory applications.

Expanded memory works by mapping each 16K page frame into any 16K page of expanded memory (refer to Appendix A for details). A *register set* is simply a complete set of mappings for all the page frames in the system. That is, a register set contains information that specifies which expanded memory page is mapped into each page frame.

There is always at least one register set. If there is more than one register set, then it is possible to quickly switch *all* the page mappings simply by switching from one register set to another. This is very useful in multitasking systems such as Desqview, because it allows each task to define its own expanded memory page mappings in a separate register set. Then you can switch quickly from one task to another simply by switching register sets.

The **/R** parameter specifies the number of register sets to allocate *in addition* to the standard register set.

**Example** — To allocate 32 alternate register sets (for a total of 33 register sets), use

```
STACKS=0,0  
DEVICE=CP386.SYS /R=32  
DEVICE=C386EMM.SYS
```

## /S, /Z — Interrupt Stack Frame Allocation

---

DOS versions 3.2 and later include a **STACKS** command that can be placed in the **CONFIG.SYS** file. The **STACKS** command allocates stack frames that are used to process hardware interrupts. Whenever a hardware interrupt occurs, DOS automatically switches from the application program's stack to one of the interrupt stack frames. The reason for this is that many application programs do not provide enough stack space to handle hardware interrupts.

The DOS **STACKS** command includes two parameters that let you specify the number of interrupt stack frames, and the size (in bytes) of each stack frame.

**CP386.SYS** provides its own interrupt stack frames. When using Invisible RAM 386, you use the stack frames provided by **CP386.SYS**, instead of the stack frames provided by DOS. To do this, you specify the command **STACKS=0,0** in order to tell DOS not to allocate any interrupt stack frames.

The **/S** and **/Z** parameters replace the two parameters in the DOS **STACKS** command. **/S** specifies the number of interrupt stack frames to allocate. **/Z** specifies the size in bytes of each interrupt stack frame.

**Example** — To allocate 32 stack frames, each 256 bytes in size, use

```
STACKS=0,0
DEVICE=CP386.SYS /S=32 /Z=256
DEVICE=C386EMM.SYS
```

**Note** — DOS versions 3.1 and earlier do not have a **STACKS** command, and do not allocate any interrupt stack frames. When using DOS version 3.1 or earlier, you would not place a **STACKS=0,0** command in the **CONFIG.SYS** file. **CP386.SYS** always allocates its own interrupt stack frames, regardless of which version of DOS you use.



## /0 to /7 — DMA Data Buffer Allocation

---

DMA (*Direct Memory Access*) is a method for transferring data between the computer's memory and a peripheral device such as a disk drive, tape drive, or network.

The computer has several built-in *DMA channels* which are used to perform the data transfer. The number and capabilities of the DMA channels vary depending on the type of computer you have:

- On computers with the Industry Standard Architecture (ISA), the standard PC AT design, there are seven DMA channels. Channels 0 to 3 perform 8-bit data transfers, and are able to transfer up to 64K bytes in a single operation. Channels 5 to 7 perform 16-bit data transfers, and are able to transfer up to 128K bytes in a single operation. There is no channel 4.
- On computers with the Micro Channel Architecture (MCA), there are eight DMA channels, numbered 0 to 7. All channels can perform either 8-bit or 16-bit data transfers. When performing 8-bit transfers, they are able to transfer up to 64K bytes in a single operation. When performing 16-bit transfers, they are able to transfer up to 128K bytes in a single operation.

With **CP386** installed, the DMA channels cannot transfer data directly between an application's memory and a peripheral device. Instead, **CP386** uses a separate *data buffer*:

- When transferring data from an application's memory to a peripheral device, **CP386** first copies the data from the application's memory to the data buffer, and then uses DMA to transfer the data from the buffer to the peripheral device.
- When transferring data from a peripheral device to an application's memory, **CP386** first uses DMA to transfer the data from the peripheral device to the data buffer, and then copies the data from the buffer to the application's memory.

There is a separate data buffer for each DMA channel. The buffers are located in extended memory, so they do not take up any DOS memory. Each buffer must be big enough to hold the largest DMA transfer that is ever performed on the corresponding DMA channel. For example, the buffer for DMA channel 2 must be at least as large as the largest DMA transfer that is ever performed on DMA channel 2.

With the `/0` to `/7` parameters, you can specify the size of each DMA channel's data buffer. `/0` specifies the buffer size for DMA channel 0, `/1` specifies the buffer size for DMA channel 1, and so on. You can specify a buffer size ranging from 0K to 128K bytes.

The minimum, maximum, and default buffer sizes are determined as follows:

- For all channels, the minimum buffer size is 0K, with the following two exceptions: (1) For channel 2, the minimum buffer size is 9K. (2) If the hard disk uses DMA, the minimum buffer size for the hard disk's DMA channel is 64K.
- For channels 0 to 3 on ISA computers, the maximum buffer size is 64K.
- For channel 4 on ISA computers, the maximum buffer size is 0K.
- For channels 5 to 7 on ISA computers, and for channels 0 to 7 on MCA computers, the maximum buffer size is 128K.
- For all channels, the default buffer size is 12K, with the following two exceptions: (1) On ISA computers, the default buffer size for channel 4 is 0K. (2) If the hard disk uses DMA, the default buffer size for the hard disk's DMA channel is 64K.

With the `/0` to `/7` parameters, if you specify a buffer size less than the permitted minimum, **CP386** automatically uses the minimum. Similarly, if you specify a buffer size larger than the permitted maximum, **CP386** automatically uses the maximum.

**Example** — You have a tape drive that uses DMA channel 3. The tape drive transfers data 48K bytes at a time. To specify a buffer size of 48K bytes for DMA channel 3, use

```
STACKS=0,0  
DEVICE=CP386.SYS /3=48  
DEVICE=C386EMM.SYS
```

**Example** — You can free up memory by specifying a buffer size of 0K for unused DMA channels. This frees up 12K bytes per channel, which becomes available for use as expanded memory. For example, if you have no peripheral devices that use DMA channels 0, 5, 6, or 7, you can use

```
STACKS=0,0  
DEVICE=CP386.SYS /0=0 /5=0 /6=0 /7=0  
DEVICE=C386EMM.SYS
```



# 6 Advanced Parameters For C386EMM.SYS

This chapter describes the the advanced features supported by the expanded memory manager, **C386EMM.SYS**. The advanced features include:

- Controlling allocation of expanded memory
- Specifying internal expanded memory manager parameters
- Controlling frontfill
- Controlling the memory test
- Creating high RAM
- Creating shadow RAM

To understand this chapter, you need a basic understanding of IBM PC memory. Refer to Appendix A for general information about IBM PC memory. Refer to Appendix B for instructions on how to display a complete memory map of your system.

# Software Installation

---

The following shows all the parameters available when installing the expanded memory manager.

To install the expanded memory manager, place the following line into the CONFIG.SYS file:

**DEVICE=C386EMM.SYS** [*parameters*]

where the optional *parameters* may include the following:

<b>/A=addr</b>	The starting address of the standard EMS area. <i>addr</i> is a segment address, in hexadecimal.
<b>/C=contexts</b>	The number of EMS context save areas. This controls a resource that is used by EMS applications. <i>contexts</i> is a decimal number. The minimum allowed value is 32, and the maximum allowed value is 255. If <b>/C</b> is omitted, it defaults to the value of <b>/H</b> .
<b>/F=size</b>	Frontfill size. <i>size</i> is a decimal number giving the desired size of DOS memory, in kilobytes. A value of 0 can be specified to request that the size of DOS memory not be changed. If <b>/F</b> is omitted, the default is to make DOS memory as large as possible. The <b>/F</b> parameter cannot be used to reduce the size of DOS memory below its original value.
<b>/H=handles</b>	Number of EMS handles. This controls a resource that is used by EMS applications. <i>handles</i> is a decimal number. The minimum allowed value is 32, and the maximum is 255. If <b>/H</b> is omitted, the default is 64.

**/I=iii [-jjj]**

Include memory area. *iii* and *jjj* are segment addresses, in hexadecimal. All 16K page frames that overlap the specified address range are used as paged EMS memory. The square brackets indicate that *jjj* is optional; if *jjj* is omitted, the single 16K page frame containing address *iii* is used as paged EMS memory. The **/I** parameter may be repeated as many times as desired. **I** is not valid if you specify **/Z=2**.

**/K=frames**

Number of 16K page frames in the standard EMS area. *frames* is a decimal number. The minimum allowed value is 2, and the maximum is 12. If **/K** is omitted, it defaults to 4. **Note** — Most EMS applications won't work if **/K** has a value other than 4.

**/L**

Force driver into low memory. If you create any shadow RAM (with the **/S** or **/Z** parameter), the expanded memory manager automatically copies itself into the shadow RAM, in order to free up DOS memory. If you include **/L**, the expanded memory manager does not copy itself into shadow RAM. (If you don't create any shadow RAM, the **/L** parameter is ignored.)

**/P=page**

Create non-standard physical page. *page* is a decimal number; the only allowed values are 254 and 255. This parameter is only useful for IBM DOS 4.0 (*not* MS-DOS 4.0), if you want to place DOS BUFFERS, FASTOPEN, or VDISK into expanded memory. The **/P** parameter may be repeated.

**/R=iii [-jjj]**

High RAM memory area. *iii* and *jjj* are segment addresses, in hexadecimal. All 16K page frames that overlap the specified address range are used as non-paged high RAM. The square brackets indicate that *jjj* is optional; if *jjj* is omitted, the single 16K page frame containing address *iii* is used as non-paged high RAM. The **/R** parameter may be repeated as many times as desired. **Note** — high RAM can only be used by specially designed applications.

**/S=iii [-jjj]**

Shadow RAM memory area. *iii* and *jjj* are segment addresses, in hexadecimal. All 16K page frames that overlap the specified address range are used as non-paged shadow RAM. The square brackets indicate that *jjj* is optional; if *jjj* is omitted, the single 16K page frame containing address *iii* is used as non-paged shadow RAM. The **/S** parameter may be repeated as many times as desired. **Note** — Shadow RAM can be used to hold DOS device drivers and TSRs, using the programs **LSHADOW.COM** and **LSHADOW.SYS**.

**/T=level**

Memory test level. The only allowed values of *level* are 0, 1, and 2. **/T=0** forces the program to always use the short memory test. **/T=2** forces the program to always use the long memory test. **/T=1** makes the program use the long test following a cold boot (power-on), and the short test following a warm boot (Ctrl-Alt-Del). If **/T** is omitted, the default is 1.



***/N=state***

VGA or EGA high-resolution graphics enable state. The only allowed values of ***state*** are 0 and 1. ***/V=0*** disables high-resolution graphics, allowing DOS memory to increase above 640K. ***/V=1*** enables high-resolution graphics, limiting DOS memory to a maximum of 640K. In either case, the **VGAON** and **VGAOFF** programs can be used to enable and disable high-resolution graphics as needed. If ***/V*** is omitted, it defaults to 1. If you don't have a VGA or EGA video system, ***/V*** is ignored.

***/X=iiii [-jjjj]***

Exclude memory area. ***iiii*** and ***jjjj*** are segment addresses, in hexadecimal. All 16K page frames that overlap the specified address range are not used. The square brackets indicate that ***jjjj*** is optional; if ***jjjj*** is omitted, the single 16K page frame containing address ***iiii*** is not used. The ***/X*** parameter may be repeated as many times as desired.

***/Z=config***

System configuration. The only allowed values of ***config*** are 0, 1, and 2. ***/Z=0*** selects expanded memory only, with no shadow RAM. ***/Z=1*** selects expanded memory and shadow RAM. ***/Z=2*** selects shadow RAM only, with no expanded memory. If ***/Z*** is omitted, it defaults to ***/Z=0***. The ***/Z*** parameter can be overridden or modified by the ***/I***, ***/R***, ***/S***, and ***/X*** parameters.

The following sections give additional details and examples of the use of the parameters.

# Page Frames

---

To use the various parameters, you need to understand something about page frames and the types of memory that Invisible RAM 386 can create.

The IBM PC can address a total of 1024K of memory. Invisible RAM 386 divides this 1024K into 64 *page frames*, each 16K in size. The first page frame occupies segments 0000-03FF, the second page frame occupies segments 0400-07FF, and so on; the last (sixty-fourth) page frame occupies segments FC00-FFFF.

Within each page frame, Invisible RAM 386 can create one of three types of memory:

- ***Paged EMS memory.*** A page frame set up as paged EMS memory can access any 16K of expanded memory. In effect, the page frame acts as a “window” into the expanded memory. The window can be moved under software control. Page frames set up as paged EMS memory can be used by any application that is designed to use expanded memory.
- ***Shadow RAM.*** A page frame set up as shadow RAM contains a fixed 16K block of memory. Page frames set up as shadow RAM can be used to load DOS device drivers and TSRs (terminate-stay-resident programs) outside of DOS memory. The programs **LSHADOW.COM** and **LSHADOW.SYS** provide access to shadow RAM.
- ***High RAM.*** A page frame set up as high RAM contains a fixed 16K block of memory. High RAM is very similar to shadow RAM, except that high RAM cannot be accessed with the **LSHADOW** programs. High RAM can only be used by specially-designed application programs.

Additional information about segment addressing and IBM PC memory can be found in Appendix A.

## /Z — System Configuration

---

The **/Z** parameter selects one of three basic configurations. In effect, **/Z** establishes the default configuration for Invisible RAM 386. The other parameters act to modify the default configuration selected by **/Z**.

**/Z=0** selects an expanded memory only configuration, with no shadow RAM. The action of **/Z=0** depends on the type of video board you have:

- If you do not have EGA or VGA video, **/Z=0** locates all unused page frames and sets them up as paged EMS memory.
- If you have EGA or VGA video, **/Z=0** locates all unused page frames except segments A000-AFFF, and sets them up as paged EMS memory. Segments A000-AFFF are set up as shadow RAM. This permits DOS memory frontfill, while still allowing the use of high-resolution graphics.

**/Z=1** selects a configuration that supports both expanded memory and shadow RAM. Invisible RAM 386 locates all unused page frames. Four of the page frames are set up as paged EMS memory, and the rest are set up as shadow RAM.

**/Z=2** selects a shadow RAM only configuration, with no expanded memory support. Invisible RAM 386 locates all unused page frames, and sets them up as shadow RAM.

## /I, /R, /S, /X — Memory Allocation Control

---

The **/I**, **/R**, **/S**, and **/X** parameters let you change the default behavior selected by the **/Z** parameter. With these parameters, you can specify the type of memory to be created in each page frame. Each parameter includes a range of addresses that selects one or more page frames.

**NOTE** — You can display a complete memory map of your system by giving the command

**SHADOW /M**

This memory map can help you figure out how to customize your memory allocation. Refer to Appendix B for a full description of the **SHADOW** program.

## **The /X Parameter — Exclude Page Frames**

---

The **/X** parameter lets you exclude one or more page frames. This means Invisible RAM 386 does not use the specified page frames.

Normally, Invisible RAM 386 automatically excludes all page frames that are already used by other equipment in the computer. However, Invisible RAM 386 may not be able to detect all other equipment in the computer. With **/X**, you can tell Invisible RAM 386 not to use page frames that are required for other equipment.

**Example** — If you have EGA or VGA graphics, Invisible RAM 386 by default uses memory locations A000-AFFF as shadow RAM. This allows you to disable and enable high-resolution graphics with the **VGAOFF** and **VGAON** programs. If you never disable high-resolution graphics, you can free up an additional 64K of expanded memory by telling Invisible RAM 386 not to use memory locations A000-AFFF. To do this, use **/X** as shown:

**DEVICE=C386EMM.SYS /X=A000-AFFF**

**Example** — The IBM VGA graphics adapter uses memory locations CA00-CA7F. (Non-IBM graphics adapters do *not* use these locations. Also, the VGA that is built in to IBM PS/2 computers does *not* use these locations.) Invisible RAM 386 is not able to detect this memory usage, and so it won't automatically exclude these locations. So, if you have an IBM VGA graphics adapter, you must use the **/X** parameter as follows:

**DEVICE=C386EMM.SYS /X=CA00-CA7F**

In this example, the **/X** parameter actually excludes the entire page frame C800-CBFF. So the above command is actually equivalent to any of the following commands:

```
DEVICE=C386EMM.SYS /X=C800-CBFF
```

```
DEVICE=C386EMM.SYS /X=CA00
```

```
DEVICE=C386EMM.SYS /X=C800
```

**Example** — Many debugger boards (for example, Periscope) use memory locations that cannot be detected by Invisible RAM 386. If the debugger board uses memory locations D000-D7FF, you would include the **/X** parameter as follows:

```
DEVICE=C386EMM.SYS /X=D000-D7FF
```

**Example** — Some network boards use memory locations that cannot be detected by Invisible RAM 386. For instance, the IBM Token Ring adapter uses 16K of memory at address D800. If you had both the IBM Token Ring adapter and the IBM VGA graphics adapter, you would use two **/X** parameters, as shown:

```
DEVICE=C386EMM.SYS /X=D800-DBFF /X=CA00-CA7F
```

## The **/I** Parameter — Create Paged EMS Memory

---

The **/I** parameter lets you set up one or more page frames as paged EMS memory. This means applications designed to use expanded memory are able to use the page frames.

By default, Invisible RAM 386 sets up all unused page frames as paged EMS memory or shadow RAM. So the two main uses for **/I** are:

- To create paged EMS memory in a page frame where Invisible RAM 386 can't tell that the page frame is available.
- To create paged EMS memory in a page frame that the **/Z** parameter would otherwise set up as shadow RAM.

**Example** — With monochrome video, segments B000-B7FF are used

as video RAM. However, with some monochrome video boards, segments B400-B7FF can be converted into paged EMS memory. Specify

```
DEVICE=C386EMM.SYS /I=B400-B7FF
```

**Example** — With color video, segments B800-BFFF are used as video RAM. However, with some color video boards, segments BC00-BFFF can be converted into paged EMS memory. Specify

```
DEVICE=C386EMM.SYS /I=BC00-BFFF
```

**Example** — EGA and VGA graphics adapters use segments A000-AFFF for high-resolution graphics. By default, Invisible RAM 386 sets up these segments as shadow RAM, so that you can use high-resolution graphics. If you want to use segments A000-AFFF as paged EMS memory, you can use the **/I** parameter as shown below. Note that using A000-AFFF as paged EMS memory permanently disables the high-resolution graphics; you can't use **VGAON**. (Refer to Chapter 4 for additional information on using EGA or VGA graphics.)

```
DEVICE=C386EMM.SYS /I=A000-AFFF
```

## The /S Parameter — Create Shadow RAM

---

The **/S** parameter lets you set up one or more page frames as shadow RAM. The shadow RAM can be used to hold memory-resident programs and device drivers, thus freeing up DOS memory. The **LSHADOW** programs allow access to the shadow RAM; refer to Chapter 3 for details.

In most cases, you can allocate required shadow RAM with the **/Z=1** or **/Z=2** parameters. So the two main uses for **/S** are:

- To create shadow RAM in a page frame where Invisible RAM 386 can't tell that the page frame is available.
- To create shadow RAM in a page frame that the **/Z** parameter would otherwise set up as paged EMS memory.

**Example** — With some monochrome video boards, you can get an

extra 16K bytes of shadow RAM by turning on shadow RAM in the region from B400 to B7FF. Specify

**DEVICE=C386EMM.SYS /S=B400-B7FF**

**Example** — With some color video boards, you can get an extra 16K bytes of shadow RAM by turning on shadow RAM in the region from BC00 to BFFF. Specify

**DEVICE=C386EMM.SYS /S=BC00-BFFF**

**Example** — Suppose you specify **/Z=0** (or omit **/Z**), so that Invisible RAM 386 creates paged EMS memory in all unused page frames. If you would like to have 32K of shadow RAM at address C800, you would specify

**DEVICE=C386EMM.SYS /S=C800-CFFF**

**NOTE** — The **LSHADOW** programs can use shadow RAM much more efficiently if the shadow RAM is organized in one large, contiguous block, rather than a number of smaller blocks. When allocating shadow RAM, try to allocate large blocks rather than small blocks.

## **The /R parameter — Create High RAM**

---

The **/R** parameter lets you set up one or more page frames as high RAM. High RAM cannot be used by expanded memory applications, and it cannot be accessed with the **LSHADOW** programs. High RAM can only be used by specially-designed programs.

By default, Invisible RAM 386 never creates high RAM. So, if you want to have high RAM, you must create it explicitly with the **/R** parameter.

**Example** — Suppose you want 32K of high RAM at address C800. You would specify

**DEVICE=C386EMM.SYS /R=C800-CFFF**

## /A, /K — Standard EMS Area Control

---

There are two types of paged EMS page frames: *standard* and *enhanced*. The standard EMS page frames always form a contiguous block of memory, called the *standard EMS area*.

By default, there are four standard EMS page frames, arranged in a contiguous 64K block. Invisible RAM 386 automatically selects an appropriate starting address. All other EMS page frames automatically become enhanced EMS page frames.

Most expanded memory applications use only the standard EMS page frames. Very few require any enhanced EMS page frames.

With the /A and /K parameters, you can customize the location and size of the standard EMS area.

### The /A parameter — Standard EMS Area Start Address

---

The /A parameter specifies the starting address of the standard EMS area.

**Example** — If you would like to place the 64K standard EMS area in segments E000-EFFF, use

```
DEVICE=C386EMM.SYS /A=E000
```

**NOTE** — The /A parameter does not create paged EMS memory. It only specifies which paged EMS memory should be used as standard EMS page frames. If you need to create paged EMS memory, use the /I parameter in addition to /A. (However, if you use /Z=1, the /A parameter specifies which page frames to use as paged EMS memory; remaining page frames are used as shadow RAM.)

### The /K Parameter — Standard EMS Area Size

---

The /K parameter specifies the size of the standard EMS area. Nor-



mally, the standard EMS area consists of 4 page frames (64K). However, with **/K** you can adjust the size from a minimum of 2 page frames to a maximum of 12 page frames. Note that there must be a contiguous block of paged EMS memory large enough to hold the specified standard EMS area.

**Example** — If you would like to make the standard EMS area 2 page frames (32K) in size, use

```
DEVICE=C386EMM.SYS /K=2
```

**NOTE** — Most expanded memory applications won't work if the value of **/K** is different from 4.

## **/F, /L, /V — DOS Memory Control**

---

By default, Invisible RAM 386 automatically makes DOS memory as large as possible. The **/F**, **/L**, and **/V** parameters let you alter this behavior, if you so desire.

### **The /F Parameter — Frontfill Size**

---

The **/F** parameter lets you control the total size of DOS memory. In most cases, Invisible RAM 386 automatically increases the size of DOS memory to 704K on a monochrome system, or 736K on a color system. This process is called *frontfill*. With **/F**, you can control the amount of frontfill.

**Example** — If you want the total DOS memory size to be 672K, use

```
DEVICE=C386EMM.SYS /F=672
```

**Example** — If you don't want Invisible RAM 386 to change the size

of DOS memory, specify a value of 0 for **/F**, as shown:

**DEVICE=C386EMM.SYS /F=0**

**NOTE** — **/F** cannot be used to reduce the size of DOS memory below its initial value.

**NOTE** — Page frames not used for frontfill become available for use as paged EMS memory, shadow RAM, or high RAM.

**NOTE** — If you have VGA or EGA video, you can still specify a value for **/F** larger than 640. When **C386EMM.SYS** first loads, DOS memory size is limited to a maximum of 640K (unless you specify **/V=0**). However, **C386EMM.SYS** remembers the value you specified for **/F**, and DOS memory size is increased to the specified value when you run **VGAOFF**.

## **The /L Parameter — Force Driver Into Low Memory**

---

When there is shadow RAM in the system not used for frontfill, **C386EMM.SYS** automatically copies itself into the shadow RAM. This frees up approximately 9K of additional DOS memory.

If you specify **/L**, then **C386EMM.SYS** does not copy itself into shadow RAM; it remains in low DOS memory. This frees up about 9K of additional shadow RAM.

**Example** — To keep **C386EMM.SYS** in low DOS memory, use

**DEVICE=C386EMM.SYS /L**

## **The /V Parameter — VGA/EGA Graphics Enable State**

---

The **/V** parameter lets you specify whether high-resolution VGA or EGA graphics should be enabled or disabled. Disabling high-resolution graphics allows DOS memory size to be increased above 640K.

**Example** — To disable VGA or EGA high-resolution graphics, specify a value of 0 as shown:

**DEVICE=C386EMM.SYS /V=0**

**Example** — To enable VGA or EGA high-resolution graphics, specify a value of 1 as shown:

**DEVICE=C386EMM.SYS /V=1**

**NOTE** — Regardless of the value you specify for **/V**, you can use **VGAON** and **VGAOFF** to enable and disable high-resolution graphics as needed.

**NOTE** — **/V** is ignored if you don't have a VGA or EGA video system.

## **/C, /H, /P — Internal Expanded Memory Parameters**

---

The **/C**, **/H**, and **/P** parameters adjust certain internal parameters of the Expanded Memory Manager. Generally, they are needed only for special applications.

### **The /P Parameter — Create Non-Standard Page Frames**

---

The **/P** parameter lets you create enhanced EMS page frames with non-standard page frame numbers. This is used only with IBM DOS 4.0 (*not* with MS-DOS 4.0), to use the expanded memory features of DOS.

**Example** — If you want to use VDISK or FASTOPEN in expanded memory, you must create page number 254, as shown:

**DEVICE=C386EMM.SYS /P=254**

**Example** — If you want to use BUFFERS in expanded memory, you

must create page number 255, as shown:

```
DEVICE=C386EMM.SYS /P=255
```

**NOTE** — **/P** does not create paged EMS memory. It merely specifies that one of the enhanced EMS page frames should be reserved for DOS (note that **/P** can't use a standard EMS page frame). If necessary, use **/I** to create paged EMS memory.

## The /H Parameter — Number of EMS Handles

---

The **/H** parameter controls the number of EMS handles. *Handles* are used by expanded memory application programs to request memory from the Expanded Memory Manager. Generally, you need one or two handles for each expanded memory application that is running concurrently in the computer.

The default value of **/H** should be adequate for almost any application. However, if an application reports that there are no expanded memory handles available, increase the value of **/H**.

**Example** — If you want to have 150 EMS handles, use

```
DEVICE=C386EMM.SYS /H=150
```

## The /C Parameter — Number of EMS Contexts

---

The **/C** parameter controls the number of EMS contexts. *Contexts* are used by some (but not all) expanded memory application programs.

The default value of **/C** should be adequate for almost any application. However, if an application reports that there are no expanded memory contexts available, increase the value of **/C**.

**Example** — If you want to have 220 EMS contexts, use

```
DEVICE=C386EMM.SYS /C=220
```

## /T — Hardware Test

---

The **/T** parameter controls the testing of the expanded memory.

### The /T Parameter — Memory Test Level

---

When the Expanded Memory Manager is loaded, it automatically performs a test of the expanded memory. Two tests are provided: a short test and a long test. Since the long test can take an annoyingly long time, the **/T** parameter is provided to let you select the memory test you want.

**Example** — If you always want to use the short test, specify a value of 0 as shown:

```
DEVICE=C386EMM.SYS /T=0
```

**Example** — If you always want to use the long test, specify a value of 2 as shown:

```
DEVICE=C386EMM.SYS /T=2
```

**Example** — If you specify a value of 1, the Expanded Memory Manager performs the long test after a cold boot (power-on), and the short test after a warm boot (Ctrl-Alt-Del). This is the default.

```
DEVICE=C386EMM.SYS /T=1
```

**Note** — We have found that with some versions of BIOS, **/T=1** always results in the long test. If you find this happens on your computer, simply use **/T=0** to get the short test.



# A Advanced Memory Configuration Information

In this chapter we describe technical information about memory in the PC. The topics covered are:

- Conventional memory
- Extended memory
- Expanded memory
- Shadow RAM and high RAM

# Conventional Memory

---

The IBM PC can address 1024K bytes of memory. This 1024K is divided into *segments*, each containing 16 bytes. Thus, there are 65,536 different segments. It is conventional to number the segments using a *hexadecimal* numbering system (a *base-16* number system with digits that range from 0 to 9 and A to F). Thus, segments are numbered from 0000 to FFFF.

Segments 0000-9FFF are reserved for DOS. This is the *standard DOS 640K area*.

Segments B000-BFFF are reserved for the video system. Segments A000-AFFF are used by the high-resolution graphics modes of VGA and EGA video systems.

The highest-numbered segments (FFFF down) are used for the *system BIOS*, a ROM which is physically located on the motherboard. The size of the system BIOS can range from a little as 8K to as much as 128K. Most AT-type computers have either a 64K BIOS which occupies segments F000-FFFF, or a 32K BIOS which occupies segments F800-FFFF. Most XT-type computers have an 8K BIOS which occupies segments FE00-FFFF.

Segments from C000 to the bottom of the BIOS are available for use by plug-in adapter boards. Many adapters locate ROM in this area.

## Extended Memory

---

The 80286, 80386, and 80486 microprocessors can address more than one megabyte of memory. Memory beyond one megabyte is called *extended memory*.



Like conventional memory, extended memory is numbered in segments. On the 80286, there can be up to 15 megabytes of extended memory, occupying segments 10000 to FFFFF. On the 80386 and 80486, there can be up to 4095 megabytes of extended memory, occupying segments 10000 to FFFFFFFF.

Unfortunately, extended memory is not directly accessible to DOS. This is because the microprocessor must be switched into a special mode of operation, called *protected mode*, in order to access the extended memory. DOS cannot operate in protected mode, and so DOS cannot use extended memory.

With Invisible RAM 386, your computer's extended memory is converted into memory that DOS programs can use. Some of the extended memory is used to increase the size of DOS memory (*frontfill*), and create shadow RAM that can be used by memory-resident programs and device drivers. The rest of the extended memory is converted into *expanded memory*, which can be used by many application programs.

## Expanded Memory

---

*Expanded memory* is a method for adding as much as 32 megabytes of memory, while remaining within the 1024K addressing limit of the PC. Expanded memory is also called *EMS memory* (for Expanded Memory Specification) or *LIM memory* (for Lotus-Intel-Microsoft, the companies that originally defined expanded memory).

The expanded memory is divided into *pages*, each 16K bytes in size. Since expanded memory can be as large as 32 megabytes, there can be as many as 2048 pages.

The PC's 1024K address space is divided into 64 *page frames*, each 16K bytes in size. Each page frame acts as a "window" into the

expanded memory. The memory hardware allows any page frame to map into any page. Software can change the mapping at any time, so that every page frame can view any part of the expanded memory. As a result, a handful of 16K byte page frames can provide access to many megabytes of memory.

Not all of the 64 page frames can be used to access the expanded memory. For example, some of the page frames are required for the system BIOS, plug-in adapter boards, and the video system. A page frame may be used to access expanded memory only if it does not conflict with any equipment installed in the computer. Typically, page frames available for expanded memory are located in the area from segment C000 to the start of the BIOS, and, if the video system permits, in the area A000-AFFF.

Application programs do not exercise direct control over the expanded memory page frames. Instead, they make requests to a program called the *expanded memory manager*. The expanded memory manager is responsible for manipulating the expanded memory hardware.

Every expanded memory system provides at least four 16K page frames, arranged in a contiguous 64K block of memory. This is called the *standard EMS* area. For example, the standard EMS area may consist of four 16K page frames located in segments D000-D3FF, D400-D7FF, D800-DBFF, and DC00-DFFF; notice that the page frames form a contiguous 64K block from D000 to DFFF.

Invisible RAM 386 provides additional page frames beyond the four standard page frames. The additional page frames are called the *enhanced EMS* area. They may not be contiguous. Invisible RAM 386 creates a page frame wherever there is a 16K block of memory that is not being used for some other purpose. For example, there may be enhanced EMS windows in segments A000-A3FF, A400-A7FF, A800-ABFF, AC00-AFFF, C800-CBFF, and CC00-CFFF; notice that they are not all contiguous.

Invisible RAM 386 also creates enhanced EMS page frames within the DOS 640K area. This allows certain programs, such as Desqview, to provide multitasking by changing the mapping of conventional DOS memory.

# Shadow RAM And High RAM

---

We have explained how a page frame acts as a window into expanded memory. By calling the expanded memory manager, software can use a page frame to access any 16K byte page of expanded memory.

With Invisible RAM 386, it is possible to make some page frames access a *fixed* 16K page of memory. Such page frames are called *shadow RAM* or *high RAM*. A page frame used for shadow RAM or high RAM cannot access any 16K page of expanded memory; it only accesses one, fixed, unchangeable 16K page of memory.

From a hardware standpoint, shadow RAM is exactly the same as high RAM. The difference between shadow RAM and high RAM lies in the way they are utilized by software.

Shadow RAM is used by the **LSHADOW** programs supplied with Invisible RAM 386 (see Chapter 3 for details). **LSHADOW.COM** lets you load TSR's (terminate-stay-resident programs) into the shadow RAM. **LSHADOW.SYS** lets you load DOS device drivers into the shadow RAM. In either case, more DOS memory is freed up for application programs. Notice that you don't need specially-designed TSR's or device drivers in order to use shadow RAM; **LSHADOW** works with ordinary TSR's and device drivers.

High RAM is used only by specially-designed programs. It cannot be used by **LSHADOW**. If you don't have any programs that are designed to work with high RAM, then there is no use in creating any high RAM.



# B Displaying The Memory Map

We have already described how to use the **SHADOW.EXE** program to display the current shadow RAM usage. We now describe the advanced capabilities of **SHADOW.EXE** to display extensive information about the memory in your computer, including a complete memory map.

The information that **SHADOW** can display includes:

- DOS memory information
- Expanded memory information
- Table of expanded memory handles
- Extended memory size
- Chips and Technologies chipset information
- NET/30-EMS network software information
- Shadow RAM usage
- Memory map

# Displaying Memory Information

---

The following shows all the parameters available when using **SHADOW.EXE** to display information about the memory in your computer.

To display information about the memory in your computer, type

**SHADOW** [*options*]

where the *options* determine which information is to be displayed. The *options* may include one or more of the following:

- |           |   |
|-----------|---|
| <b>/A</b> | All. Display all of the following information.  |
| <b>/C</b> | Chips and Technologies chipset. Display information about special memory-management chips manufactured by Chips and Technologies. (These chips are built in to many AT-type computers.) |
| <b>/D</b> | DOS. Display information about DOS memory.  |
| <b>/E</b> | Extended. Display the size of extended memory.  |
| <b>/H</b> | Handles. Display a table of all expanded memory handles.  |
| <b>/M</b> | Memory map. Display a complete memory map of the system.  |
| <b>/N</b> | NET/30-EMS network software. Display information about the NET/30-EMS network operating system that is manufactured by Invisible Software.  |

<code>/S</code>	Shadow RAM. Display the usage of shadow RAM.
<code>/X</code>	Expanded. Display information about expanded memory.

The following sections describe each type of memory information in detail.

## `/D` — DOS Memory Information

---

If you include the `/D` or `/A` parameter, **SHADOW** produces a display similar to the following:

```
DOS version number:  3.30.  
Total DOS memory:   736K.  
Free DOS memory:    638K.
```

The first line indicates which version of DOS you are using.

The second line indicates the total amount of memory available for DOS.

The third line indicates the amount of DOS memory that is available for application programs.

## /X — Expanded Memory Information

---

You can display information about expanded memory by including the **/X** or **/A** parameter.

If there is no expanded memory manager software installed, **SHADOW** displays the following:

```
Expanded Memory Manager:  Not installed.
```

If there is an expanded memory manager installed, **SHADOW** produces a display similar to the following:

```
Expanded Memory Manager:  Installed.  
Version number:  4.0.  
Total expanded memory:  2048K.  
Free expanded memory:  1824K.  
EEMS (enhanced EMS) functions:  Available.
```

The first line indicates that the expanded memory manager is installed.

The second line gives the version number of the expanded memory manager software.

The third line gives the total amount of expanded memory in the system.

The fourth line gives the amount of expanded memory that is available for use by application programs.

The fifth line tells whether the expanded memory manager software supports the *enhanced EMS* (EEMS) functions. These extra functions are not part of the official EMS specification, but are used by some application programs.



## /H — Expanded Memory Handles

---

When an application uses expanded memory, it obtains a *handle* from the expanded memory manager. The handle lets the expanded memory manager keep track of which portion of the expanded memory belongs to each application program.

If you include the **/H** or **/A** parameter, **SHADOW** displays a table of all the expanded memory handles currently in use, in a form similar to the following:

EMS handles, names, and sizes:

0	SYSTEM	512K.
1		1088K.

Each line in the table shows three items of information:

- The handle number. This can range from 0 to 255. With Invisible RAM 386, handle number 0 is created by the expanded memory manager itself; other handles are created by application programs.
- The handle name, if any. An application program can optionally assign a name to a handle. With Invisible RAM 386, handle number 0 is always named SYSTEM.
- The amount of expanded memory allocated to the handle, in kilobytes.

## /E — Extended Memory Size

---

If you include the **/E** or **/A** parameter, **SHADOW** displays the amount

of extended memory in the system, in a form similar to the following:

Extended memory size: 1024K.

If **CP386** is installed, the size displayed is the amount of extended memory you reserved with the **/E** parameter, not the total extended memory in the computer.

**NOTE** — On an XT-type computer, the size displayed is always zero.

## /C — Chips And Technologies Chipset Information

---

Chips and Technologies is a company that makes special chips which are used in many PCs. These special chips provide advanced memory management features. For this reason, **SHADOW** displays information about these special chips, if you specify the **/C** or **/A** parameter.

Chips and Technologies makes two different special chips that provide advanced memory management features:

- **The NEAT chipset.** This chipset is used with the 80286 or 80386SX microprocessors. NEAT has the ability to provide RAM in the area between 640K and 1024K, in the area normally used for BIOS ROMs and the video system. With NEAT, you can speed up the execution of BIOS code by copying the BIOS ROMs into RAM. NEAT also has a built-in expanded memory controller.
- **The AT/386 chipset.** This chipset is used with the 80386 microprocessor. Like the NEAT chipset, it can provide RAM in the area between 640K and 1024K. With AT/386, you can speed up the execution of BIOS code by copying the BIOS ROMs into RAM. Unlike the NEAT chipset, AT/386 does not have a built-in ex-

panded memory controller.

**NOTE** — Invisible Software manufactures a software package called Invisible RAM that allows you to take advantage of the built-in features of the Chips and Technologies chipsets. However, Invisible RAM 386 provides all the features of Invisible RAM on any 386 or 486 based PC, regardless of whether or not it has a Chips and Technologies chipset.

If there is no Chips and Technologies chipset in your computer, **SHADOW** displays the following:

Chips and Technologies chipset: Not present.

If the NEAT chipset is present in your computer, **SHADOW** produces a display similar to the following:

Chips and Technologies chipset: NEAT version B.  
Total RAM on motherboard: 1024K.  
640K to 1M memory relocation: Disabled.  
Shadow RAM: Available.  
Expanded memory controller: Enabled.  
EMS I/O port address: 02E8.  
EMS memory base address: E000.

The first line indicates that NEAT is installed, and tells you which version of the chipset you have.

The second line gives the total amount of RAM installed on the motherboard. It can range from 512K to 8192K. This number does not include RAM installed on memory expansion boards.

If there is exactly 1024K RAM on the motherboard, the first 640K is used for DOS. NEAT offers two options for the use of the remaining 384K. If *memory relocation* is disabled, the remaining 384K is mapped into segments A000-FFFF, where it can be used to hold copies of BIOS ROMs. If *memory relocation* is enabled, the remaining 384K is mapped into segments 10000-15FFF, where it can be used as extended or expanded memory. (If there is more than 1024K RAM on the motherboard, then memory relocation must be disabled.) The third line tells if memory relocation is enabled or disabled.

The fourth line indicates if RAM is available to hold copies of BIOS ROMs, and for use as shadow RAM. In order for RAM to be available, there must be at least 1024K RAM on the motherboard, and memory relocation must be disabled. **NOTE** — With Invisible RAM 386 installed, any shadow RAM you create (using the `/S` or `/Z` parameters with **C386EMM.SYS**) is created using extended memory, not the NEAT chipset RAM. In order to use the NEAT chipset RAM as shadow RAM, you need to obtain the Invisible RAM software package.

The fifth line indicates if the NEAT chipset's built-in expanded memory controller is enabled or disabled. When using Invisible RAM 386, you would normally disable NEAT's built-in expanded memory controller.

The sixth and seventh lines appear only if the expanded memory controller is enabled. They indicate the address of the I/O port used by the controller, as well as the starting address of the 64K byte standard EMS area.

**NOTE** — To change the configuration of the NEAT chipset, you need to use your computer's set-up program.

If the AT/386 chipset is present in your computer, **SHADOW** produces a display similar to the following:

```
Chips and Technologies chipset:  AT/386 version B.  
Total RAM on motherboard:  1024K.  
Shadow RAM:  Available.
```

The first line indicates that AT/386 is installed, and tells you which version of the chipset you have.

The second line gives the total amount of RAM installed on the motherboard. It can range from 1024K to 16384K. This number does not include RAM installed on memory expansion boards.

The third line indicates if RAM is available to hold copies of BIOS ROMs, and for use as shadow RAM. With AT/386, this RAM is always available. **NOTE** — With Invisible RAM 386 installed, any shadow RAM you create (using the `/S` or `/Z` parameters with

**C386SEMM.SYS**) is created using extended memory, not the AT/386 chipset RAM. In order to use the AT/386 chipset RAM as shadow RAM, you need to obtain the Invisible RAM software package.

**NOTE** — To change the configuration of the AT/386 chipset, you need to use your computer's set-up program.

## **/N — NET/30-EMS Network Software Information**

---

NET/30-EMS is a network software package manufactured by Invisible Software. It allows PC's to share data when they are interconnected with Invisible Software's local area network boards. One of the unique features of NET/30-EMS is its ability to use expanded memory.

You can display information about NET/30-EMS by including the **/N** or **/A** parameter. If NET/30-EMS is not installed, **SHADOW** displays the following:

NET/30-EMS network software: Not installed.

If NET/30-EMS is installed, **SHADOW** produces a display similar to the following:

NET/30-EMS network software: Installed.  
NETBIOS 32K data segment address: C800.

Refer to the NET/30 instruction manual for an explanation of this message.

## /S — Shadow RAM Usage Information

---

If you include the /S or /A parameter, **SHADOW** displays information about shadow RAM.

If you have not created shadow RAM (by specifying appropriate parameters when starting **C386EMM.SYS**), the **SHADOW** program displays the following:

Shadow RAM Manager: Not installed.

If you have created shadow RAM, **SHADOW** displays information similar to the following:

Shadow RAM Manager: Installed.  
Total shadow RAM: 192K.  
Free shadow RAM: 128K.  
Largest loadable program size: 84K.

The first line indicates that the shadow RAM manager is installed. This means you can use the **LSHADOW** programs to load TSR's (terminate-stay-resident programs) and device drivers into shadow RAM.

The second line indicates the total amount of shadow RAM that is available for loading memory-resident programs. This figure includes not only shadow RAM, but also all RAM that is used to increase (*frontfill*) the size of DOS memory; thus, it indicates the total increase in the amount of RAM available under DOS.

The third line indicates the amount of unused shadow RAM available for loading memory-resident programs.

The fourth line indicates the size of the largest contiguous block of shadow RAM. This is the size of the largest single program you can load into shadow RAM, since a program must always be loaded into a single contiguous block of shadow RAM. **Note** — Most TSRs use more memory when first loaded than they do after they become resident. To load a TSR into shadow RAM, the “largest loadable program size” must be big enough for all the memory required when the program first loads, as well as after it becomes resident.

# /M — Memory Map

---

If you include the **/M** or **/A** parameter, **SHADOW** displays a map of all the memory in the computer system. It is similar to the following:

0000-9FFF	DOS	RAM
A000-AFFF	DOS	SHADOW RAM
B000-B7FF	DOS	ENHANCED-EMS RAM
B800-BFFF	VIDEO	RAM
C000-C5FF	ROM	
C600-C7FF		
C800-DFFF	SHADOW	RAM
E000-EFFF	STANDARD-EMS	RAM
F000-FFFF	BIOS	

Each line shows a range of addresses, followed by a description of what memory is located there. The following words may appear in the description:

**DOS** — The memory is usable by DOS applications.

**RAM** — The address range contains read/write memory.

**ROM** — The address range contains a BIOS ROM for a plug-in adapter board.

**BIOS** — The address range contains the motherboard's built-in ROM BIOS.

**VIDEO** — The address range is reserved for use by the video system.

**STANDARD-EMS** — The address range contains page frames used for expanded memory. These page frames make up the standard EMS area that is used expanded memory (EMS) applications.

**ENHANCED-EMS** — The address range contains page frames used for expanded memory. These page frames are not part of the standard EMS area. Only a few applications use this type of page frame.

**SHADOW** — The address range contains shadow RAM supplied by Invisible RAM 386. This memory can be used to hold memory resident programs and device drivers, using the **LSHADOW** programs.

**HIGH** — The address range contains high RAM supplied by Invisible RAM 386. This memory can only be used by specially designed application programs.

**CHIPSET** — The address range contains memory supplied by the Chips and Technologies chipset. This memory can be used to hold copies of BIOS ROMs, thereby speeding up execution of BIOS code.

**WRITE-PROTECTED** — The address range contains memory supplied by the Chips and Technologies chipset which has been write-protected, so that it cannot be modified. This allows the chipset memory to act as if it was a ROM (ROMs cannot be modified)

Where appropriate, more than one word can appear in the description. Here are explanations for the more commonly encountered combinations:

**DOS RAM** — Read/write memory available to DOS applications. The word DOS should never appear without RAM.

**DOS SHADOW RAM** — Shadow RAM that has been used to increase (frontfill) DOS memory.

**DOS HIGH RAM** — High RAM that has been used to increase (frontfill) DOS memory.

**DOS ENHANCED-EMS RAM** — Expanded memory that is located within the DOS memory space.

**VIDEO RAM** — Read/write memory used by the video system. If VIDEO appears without RAM, it means that the address range is reserved for the video system, but there is currently no memory in the range; this occurs for address range A000-AFFF with an EGA or VGA video card operating in text mode.

**SHADOW VIDEO** — An address range that can contain shadow RAM, but the shadow RAM was temporarily disabled with the **VGAON** program.

**HIGH VIDEO** — An address range that can contain high RAM, but the high RAM was temporarily disabled with the **VGAON** program.



**STANDARD-EMS RAM** — Indicates a portion of the standard EMS area that currently has memory enabled. If STANDARD-EMS appears without RAM, it means the address range is part of the standard EMS area, but there is no memory currently mapped there.

**ENHANCED-EMS RAM** — Indicates an enhanced EMS page frame that currently has memory enabled. If ENHANCED-EMS appears without RAM, it means the address range is mappable by the expanded memory manager, but there is no memory currently mapped there.

**SHADOW RAM** — Read/write memory supplied by Invisible RAM 386 that is useable as shadow RAM. The memory can be used by the **LSHADOW** programs for loading TSR's and device drivers, or for frontfill.

**HIGH RAM** — Read/write memory supplied by Invisible RAM 386 that is useable as high RAM. The memory can be used by specially designed programs, or for frontfill.

**CHIPSET RAM** — Read/write memory supplied by the Chips and Technologies chipset.

**ROM CHIPSET** — Memory supplied by the Chips and Technologies chipset, which contains a copy of a BIOS ROM for a plug-in adapter board.

**ROM CHIPSET WRITE-PROTECTED** — Memory supplied by the Chips and Technologies chipset, which contains a copy of a BIOS ROM for a plug-in adapter board, and is write-protected so it cannot be modified.

**BIOS CHIPSET WRITE-PROTECTED** — Memory supplied by the Chips and Technologies chipset, which contains a copy of the motherboard's built-in ROM BIOS, and is write-protected so it cannot be modified.

**NOTE** — A blank line indicates an address range that has no memory of any type.



# C Using LSHADOW With DOS 4.0

With DOS 4.0, you can install some memory-resident programs from within the CONFIG.SYS file. (Not all programs can be installed this way.) For example, to install the memory-resident program called **SHARE.EXE**, you would place the following line into the CONFIG.SYS file:

```
INSTALL=SHARE.EXE
```

When installing programs this way, you can use **LSHADOW** to load them into shadow RAM. For example, you could install **SHARE.EXE** into shadow RAM as follows:

```
INSTALL=LSHADOW.COM SHARE.EXE
```

**Note** — When using **LSHADOW** with the DOS 4.0 **INSTALL** command, you may get an error message saying “Error in CONFIG.SYS line XX”. You may ignore this error message; the program is correctly installed.



# D Error Messages For CP386.SYS

This chapter describes the error messages that can be generated by the Invisible RAM 386 control program, **CP386.SYS**. There are two types of error messages:

- Initialization error messages
- Run-time error messages

## CP386 Initialization Error Messages

---

Initialization error messages occur when **CP386** is first loaded. They indicate a condition that makes it impossible to install **CP386**.

## Invalid parameters.

**Explanation:** One or more parameters on the **CP386.SYS** command line are invalid, or given in incorrect form, or have unacceptable values.

**Recommended action:** Edit the CONFIG.SYS file and correct the erroneous parameters. If you don't know which parameters are erroneous, remove all parameters from the **CP386.SYS** command line, and then add them back one-by-one.

## This program requires an 80386 or 80486.

**Explanation:** Your computer does not have an 80386, 80386SX, or 80486 microprocessor.

**Recommended action:** Use a computer with an 80386, 80386SX, or 80486 microprocessor.

## Insufficient extended memory.

**Explanation:** There is not enough extended memory to install **CP386**.

**Recommended action:** (1) Reduce the amount of extended memory reserved with the **/E** parameter. (2) Reduce the size of the DMA data buffers. (3) Add more extended memory to your computer.

**Ctrl-Alt-Shift detected. At user request, program will not install.**

**Explanation:** You pressed Ctrl-Alt-Shift. **CP386.SYS** does not load if these keys are pressed.

**Recommended action:** None.

**The 386/486 Control Program is already installed.**

**Explanation:** You tried to install **CP386** when **CP386** or some other 80386 control program is already installed.

**Recommended action:** (1) Do not try to install **CP386** more than once.  
(2) Remove any other 80386 control program you have installed in your computer.

**Cannot determine DMA controller type (standard PC or Micro Channel).**

**Explanation:** When **CP386** is installed, it needs to determine whether your computer has the Industry Standard Architecture (ISA) or the Micro Channel Architecture (MCA). This error message indicates that **CP386** was not able to make the required determination.

**Recommended action:** Write down the make and model of computer that you have, and contact Invisible Software technical support.

## Fixed disk read error - cannot determine fixed disk DMA channel.

**Explanation:** When **CP386** is installed, it needs to determine which DMA channel is used by your hard disk, if any. This error message indicates that **CP386** was not able to make the determination because a disk read error occurred.

**Recommended action:** Correct the problem with your hard disk. Make sure that the hard disk is correctly formatted, and that the disk type is correctly specified in your computer's set-up program.

## Cannot determine fixed disk DMA channel.

**Explanation:** When **CP386** is installed, it needs to determine which DMA channel is used by your hard disk, if any. This error message indicates that **CP386** was not able to make the required determination.

**Recommended action:** Write down the make and model of your computer and hard disk, and contact Invisible Software technical support.

## CP386 Run-Time Error Messages

---

Run-time error messages occur after **CP386** is installed. They indicate a condition that makes it impossible for **CP386** to continue executing.



When a run-time error occurs, **CP386** halts execution of the system and displays the following information:

- *An error code.* This numerical code identifies the exact cause of the error.
- *An error message.* This message describes the general nature of the error.
- *A diagnostic memory dump.* This is a hexadecimal display of the contents of certain memory locations internal to **CP386**. In some cases, Invisible Software personnel can use the diagnostic memory dump to help determine the source of an error.

After a run-time error, it is not possible to continue executing your application program. You must re-boot the system, either by pressing **Ctrl-Alt-Del** or by turning off the power.

## **CP386 fatal system failure: Error code 13-1**

### **General protection error (exception 0D hex)**

**Explanation:** A general protection exception occurred while executing the **CP386** kernel.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 13-2

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing protected-mode code, and the **CP386** kernel cannot read the limit of the offending code segment.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 13-3

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing protected-mode code, and the **CP386** kernel cannot read the access rights of the offending code segment.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 13-4

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing protected-mode code, and the **CP386** kernel cannot read the offending instruction because the offending code segment is marked execute-only.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 13-5

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred because an attempt was made to execute past the end of the code segment. This can occur in both real-mode and protected-mode code.

**Recommended action:** There is probably a bug in your DOS application program, or in a DOS memory-resident program (TSR) or device driver. Try to determine which program is causing the error.

## CP386 fatal system failure: Error code 13-6

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred because an attempt was made to execute a privileged 80386 instruction. This can occur in both real-mode and protected-mode code.

**Recommended action:** Most likely, a DOS application program tried to switch the processor into protected mode, or tried to use special features of the 80386. See if the application can be configured not to do this.

## CP386 fatal system failure: Error code 13-7

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred because an attempt was made to read a non-existent 80386 control register. This can occur in both real-mode and protected-mode code.

**Recommended action:** There is probably a bug in your DOS application program. Try to determine which program is causing the error.

## CP386 fatal system failure: Error code 13-8

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred because an attempt was made to read a non-existent 80386 debug register. This can occur in both real-mode and protected-mode code.

**Recommended action:** There is probably a bug in your DOS application program. Try to determine which program is causing the error.

## CP386 fatal system failure: Error code 13-9

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred because an attempt was made to write a non-existent 80386 debug register. This can occur in both real-mode and protected-mode code.

**Recommended action:** There is probably a bug in your DOS application program. Try to determine which program is causing the error.

## CP386 fatal system failure: Error code 13-10

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred because an attempt was made to execute a privileged 80386 instruction. This can occur in both real-mode and protected-mode code.

**Recommended action:** Most likely, a DOS application program tried to switch the processor into protected mode, or tried to use special features of the 80386. See if the application can be configured not to do this.

## CP386 fatal system failure: Error code 13-11

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing real-mode code, because an attempt was made to read or write past the end of the data segment (i.e., to an offset greater than 0FFFFH).

**Recommended action:** There is probably a bug in your DOS application program. Try to determine which program is causing the error.

## CP386 fatal system failure: Error code 13-12

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing protected-mode code, and the **CP386** kernel cannot read the access rights of the offending data segment.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 13-13

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing protected-mode code, and the offending data segment has an invalid type code in its segment descriptor.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 13-14

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing protected-mode code, because an attempt was made to write into a read-only data segment.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 13-15

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing protected-mode code, and the **CP386** kernel cannot read the limit of the offending data segment.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.



## CP386 fatal system failure: Error code 13-16

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing protected-mode code, because an attempt was made to read or write outside the data segment limit.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 13-17

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing protected-mode code, because an attempt was made to write into a code segment.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 13-18

### General protection error (exception 0D hex)

**Explanation:** A general protection exception occurred while executing protected-mode code, because an attempt was made to read from an execute-only code segment.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 32-1

### DMA channel *x* requires a buffer of at least *yy*K

**Explanation:** An attempt was made to perform a DMA operation too large to fit in the DMA data buffer. In the error message, *x* indicates the offending DMA channel number, and *yy* gives the buffer size, in kilobytes, that would be required for the DMA operation to proceed.

**Recommended action:** Use the appropriate parameter on the **CP386** command line to increase the buffer size for the offending DMA channel. For example, if the error message is “DMA channel 3 requires a buffer of at least 48K”, you would specify the parameter **/3=48** on the **CP386** command line.

## CP386 fatal system failure: Error code 33-1

### Unexpected operating system service call (interrupt 5B hex)

**Explanation:** The **CP386** kernel attempted to call itself recursively.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 34-1

### Exhausted resources for handling hardware interrupts

**Explanation:** There were no kernel interrupt stack frames available to handle a hardware interrupt.

**Recommended action:** Increase the value of the **/S** parameter on the **CP386** command line.

## CP386 fatal system failure: Error code 34-2

### Exhausted resources for handling hardware interrupts

**Explanation:** There were no real-mode interrupt stack frames available to handle a hardware interrupt.

**Recommended action:** Increase the value of the /S parameter on the CP386 command line.

## CP386 fatal system failure: Error code 34-3

### Exhausted resources for handling hardware interrupts

**Explanation:** An error occurred while attempting to free a real-mode interrupt stack frame at the conclusion of a hardware interrupt handler.

**Recommended action:** There is probably a bug in a DOS device driver, memory-resident program (TSR), or application program. Try to determine which program is causing the error.

## CP386 fatal system failure: Error code 35-1

### Unable to process software interrupt *xx* hex

**Explanation:** The **CP386** kernel attempted to execute an invalid software interrupt (INT) instruction. In the error message, *xx* indicates which software interrupt was attempted.

**Recommended action:** There is probably a bug in **CP386**. Write down the diagnostic memory dump and contact Invisible Software technical support.

## CP386 fatal system failure: Error code 35-2

### Unable to process software interrupt *xx* hex

**Explanation:** Real-mode code attempted to execute a software interrupt (INT) instruction, but there was insufficient stack space to process the instruction. In the error message, *xx* indicates which software interrupt was attempted.

**Recommended action:** There is probably a bug in a DOS device driver, memory-resident program (TSR), or application program. Try to determine which program is causing the error.



# E Error Messages For C386EMM.SYS

This chapter describes the error messages that can be generated by the Invisible RAM 386 expanded memory manager **C386EMM.SYS**.

**Invisible RAM 386/486 Control Program is not installed.**

**Explanation:** The Invisible RAM 80386 control program, **CP386**, is not installed.

**Recommended action:** Install **CP386.SYS**.

**Memory hardware is not present or incorrectly configured.**

**Expanded memory hardware malfunction.**

**Explanation:** Either of these messages can indicate that the memory in your computer is not functioning correctly.

**Recommended action:** (1) Check the switches and jumpers on your computer's motherboard, and any memory expansion boards, to be sure they are set correctly. (2) Make sure that you ran your computer's set-up program correctly. (3) Make sure that the memory chips in your computer are fast enough. If necessary, replace the memory chips with faster chips.

**Unable to locate required standard window for expanded memory.**

**Explanation:** The expanded memory manager cannot locate a 64K-byte block of available memory to use for the standard EMS area.

**Recommended action:** (1) Try to change the addresses used by ROM and RAM in your computer to free up a contiguous 64K-byte block. You can use **SHADOW /M** to help determine what ROM and RAM is present in your computer. (2) Use the **/I** parameter to include additional page frames as paged EMS memory. (3) If you specified the **/A** parameter, make sure there is 64K of memory available at the specified address. (4) If you specified the **/P** parameter, remove it. (5) In certain special applications, you can use the **/K** parameter to reduce the size of the standard EMS area.



## No RAM available for use as expanded memory.

**Explanation:** All the RAM in the computer was used up for backfill, frontfill, shadow RAM, and high RAM.

**Recommended action:** (1) Reduce the amount of RAM used for backfill, frontfill, shadow RAM, and high RAM. (2) Use the **/Z=2** configuration. (3) Reduce the amount of extended memory reserved by the **/E** parameter on the **CP386.SYS** command line.

## Invalid parameters.

**Explanation:** One or more parameters on the **C386EMM.SYS** command line are invalid, or given in incorrect form, or have unacceptable values.

**Recommended action:** Edit the **CONFIG.SYS** file and correct the erroneous parameters. If you don't know which parameters are erroneous, remove all parameters from the **C386EMM.SYS** command line, and then add them back one-by-one.

## Conflicting parameters.

**Explanation:** Two or more of the parameters on the C386EMM.SYS command line are in conflict with each other. For example, the same page frame may be specified in both /X and /I parameters.

**Recommended action:** Edit the CONFIG.SYS file and correct the erroneous parameters. If you don't know which parameters are erroneous, remove all parameters from the **C386EMM.SYS** command line, and then add them back one-by-one.

**Ctrl-Alt-Shift detected. At user request, program will not install.**

**Explanation:** You pressed Ctrl-Alt-Shift. **C386EMM.SYS** does not load if these keys are pressed.

**Recommended action:** None.

**Include, Shadow, or High RAM addresses conflict with installed hardware.**

**Explanation:** The page frames you specified with **/I**, **/S**, or **/R** are in conflict with equipment installed in your computer.

**Recommended action:** (1) Change the **/I**, **/S**, or **/R** parameters. (2) Change the addresses used by RAM or ROM in your computer.

**Include addresses are too low in memory.**

**Explanation:** The page frame address specified in the **/I** parameter is too small.

**Recommended action:** Change the **/I** parameter.

**Cannot create High RAM inside DOS memory.**

**Explanation:** The page frame address specified in the **/R** parameter is too small.

**Recommended action:** Change the **/R** parameter.

**Cannot create Shadow RAM inside DOS memory.**

**Explanation:** The page frame address specified in the **/S** parameter is too small.

**Recommended action:** Change the **/S** parameter.

**Cannot locate standard EMS area inside DOS memory.**

**Explanation:** The standard EMS address specified in the **/A** parameter is too small.

**Recommended action:** (1) Change the **/A** parameter. (2) Delete the **/A** parameter and let **C386EMM.SYS** select the standard EMS address.

**Memory hardware cannot create Shadow or High RAM at the specified addresses.**

**Explanation:** The page frames specified in the **/S** or **/R** parameter cannot be used for shadow or high RAM.

**Recommended action:** Change the **/S** or **/R** parameter.

## No Shadow RAM available.

**Explanation:** You specified the **/Z=2** configuration, but **C386EMM.SYS** is unable to create any shadow RAM because (1) all available RAM was used up for backfill and frontfill, or (2) there are no page frames available.

**Recommended action:** (1) Reduce the amount of RAM used for backfill, frontfill, and the high memory area. (2) Change the addresses used by RAM or ROM in your computer. (3) Reduce the amount of extended memory reserved by the **/E** parameter on the **CP386.SYS** command line.

## Unexpected internal driver error.

**Explanation:** The **C386EMM.SYS** program has a bug.

**Recommended action:** Write down the circumstances under which this error occurred and contact Invisible Software technical support.

Parameters /A, /C, /E, /H, /I, /K, and /P are not valid with /Z=2.

**Explanation:** When you select the /Z=2 configuration, you cannot use the /A, /C, /H, /I, /K, or /P parameters.

**Recommended action:** Edit the CONFIG.SYS file and correct the parameters.

Memory hardware cannot include mappable memory at the specified addresses.

**Explanation:** The page frames specified in the /I parameter cannot be used for paged EMS memory.

**Recommended action:** Change the /I parameter.

Requested frontfill size is out of range.

**Explanation:** The value specified with the /F parameter is either too large or too small.

**Recommended action:** Change the /F parameter.

**Memory hardware does not provide page frames for requested frontfill.**

**Explanation:** The amount of frontfill requested by the **/F** parameter cannot be provided because one or more of the required page frames is not available.

**Recommended action:** Change the **/F** parameter.

**Insufficient memory to perform frontfill.**

**Explanation:** The requested frontfill cannot be provided because all the RAM in the computer was used up for backfill, shadow RAM, and high RAM.

**Recommended action:** (1) Add or change the **/F** parameter to reduce the requested frontfill. (2) Reduce the amount of RAM used for backfill, shadow RAM, and high RAM. (3) Reduce the amount of extended memory reserved with the **/E** parameter on the **CP386.SYS** command line.

**No page frames available for requested non-standard pages.**

**Explanation:** The expanded memory manager cannot find any page frames to use for the non-standard pages requested with the **/P** parameter.

**Recommended action:** (1) Try to change the addresses used by ROM and RAM in your computer to free up more page frames. You can use **SHADOW /M** to help determine what ROM and RAM is present in your computer. (2) Use the **/I** parameter to include additional page frames as paged EMS memory. (3) In certain special applications, you can use the **/K** parameter to reduce the size of the standard EMS area.

**Testing page frames: XXXX**  
**Memory test failed.**

**Explanation:** The expanded memory manager was unable to read and write memory in the page frame indicated by **XXXX**.

**Recommended action:** (1) Check the switches and jumpers on your computer's motherboard, and any memory expansion boards, to be sure they are set correctly. (2) Make sure that you ran your computer's set-up program correctly. (3) Check to see that the memory chips in your computer are fast enough. If necessary, replace the memory chips with faster chips.



**Testing memory: XXXX**  
**Memory test failed.**

**Explanation:** An error was detected in testing the expanded memory supplied by Invisible RAM 386. **XXXX** indicates how much memory was successfully tested before the error occurred.

**Recommended action:** (1) Check the switches and jumpers on your computer's motherboard, and any memory expansion boards, to be sure they are set correctly. (2) Make sure that you ran your computer's set-up program correctly. (3) Check to see that the memory chips in your computer are fast enough. If necessary, replace the memory chips with faster chips.

**Memory size set incorrectly.**

**Explanation:** The computer's memory is not functioning correctly.

**Recommended action:** (1) Check the switches and jumpers on your computer's motherboard, and any memory expansion boards, to be sure they are set correctly. (2) Make sure that you ran your computer's set-up program correctly. (3) Check to see that the memory chips in your computer are fast enough. If necessary, replace the memory chips with faster chips.

**EGA/VGA high resolution graphics have been disabled.  
Use VGAON to enable high resolution graphics.**

**Explanation:** Invisible RAM 386 is using the page frames required for high-resolution graphics, thereby disabling high-resolution graphics. This can be caused by (1) specifying **/V=0** to allow **C386EMM.SYS** to frontfill into the high resolution graphics memory (segments A000-AFFF), or (2) using the **/I** parameter to create paged EMS memory within memory segments A000-AFFF.

**Recommended action:** (1) Run the **VGAON** program before running an application that uses high-resolution graphics. (2) Configure your applications to use low-resolution graphics (also called *CGA graphics*) or text mode. (3) Use the **/X=A000-AFFF** parameter to prevent **C386EMM.SYS** from using the area of memory required for EGA or VGA high-resolution graphics. (4) Remove the **/V=0** parameter if you prefer for high-resolution graphics to be enabled when **C386EMM.SYS** first loads. (5) Remove the **/I** parameter that creates paged EMS memory in segments A000-AFFF, if you want to be able to use high-resolution graphics (note that using **/I=A000-AFFF** permanently disables high-resolution graphics; you can't use **VGAON** if there is paged EMS memory in segments A000-AFFF).

EGA/VGA high resolution graphics are enabled.

Use **VGAOFF** to disable high resolution graphics and increase DOS memory size.

**Explanation:** DOS memory size is limited to 640K, because high-resolution graphics uses the same page frames required to increase DOS memory size above 640K. However, you can increase DOS memory size by using the **VGAOFF** program to disable high-resolution graphics.

**Recommended action:** (1) Run the **VGAOFF** program to disable high-resolution graphics and increase the size of DOS memory. (2) Use the **/V=0** parameter if you prefer to have high-resolution graphics disabled at the time **C386EMM.SYS** first loads.



# F Invisible RAM 386 Command Summary

Install the 80386 control program:

**DEVICE=CP386.SYS** [*parameters*]

<b>/E=</b> <i>size</i>	Extended memory size.
<b>/H</b> [= <i>hmamin</i> ]	High memory area (XMS) support.
<b>/R=</b> <i>regsets</i>	Number of EMS alternate register sets.
<b>/S=</b> <i>stacks</i>	Number of interrupt stack frames.
<b>/Z=</b> <i>stacksize</i>	Size of interrupt stack frames.
<b>/0=</b> <i>DMAsize</i>	Size of buffer for DMA channel 0.
<b>/1=</b> <i>DMAsize</i>	Size of buffer for DMA channel 1.
<b>/2=</b> <i>DMAsize</i>	Size of buffer for DMA channel 2.
<b>/3=</b> <i>DMAsize</i>	Size of buffer for DMA channel 3.
<b>/4=</b> <i>DMAsize</i>	Size of buffer for DMA channel 4.
<b>/5=</b> <i>DMAsize</i>	Size of buffer for DMA channel 5.
<b>/6=</b> <i>DMAsize</i>	Size of buffer for DMA channel 6.
<b>/7=</b> <i>DMAsize</i>	Size of buffer for DMA channel 7.

Install the expanded memory manager:

<b>DEVICE=C386EMM.SYS</b>	<b>[parameters]</b>
<b>/A=addr</b>	Address of standard EMS area.
<b>/C=contexts</b>	Number of EMS context save areas.
<b>/F=size</b>	DOS memory frontfill size.
<b>/H=handles</b>	Number of EMS handles.
<b>/I=iiii [-jjjj]</b>	Create paged EMS memory.
<b>/K=frames</b>	Size of standard EMS area.
<b>/L</b>	Force driver into low memory.
<b>/P=page</b>	Create non-standard physical page.
<b>/R=iii [-jjjj]</b>	Create high RAM.
<b>/S=iii [-jjjj]</b>	Create shadow RAM.
<b>/T=level</b>	Memory test level.
<b>/V=state</b>	VGA or EGA high-resolution graphics enable state
<b>/X=iii [-jjjj]</b>	Exclude page frames.
<b>/Z=config</b>	System configuration.

Load memory-resident program into shadow RAM:

**LSHADOW [/A] filename [parameters]**

Load device driver into shadow RAM:

**DEVICE=LSHADOW.SYS [/A] filename [parameters]**

Display memory map information:

<b>SHADOW</b>	<b>[options]</b>
<b>/A</b>	All information
<b>/C</b>	Chips and Technologies chipset information
<b>/D</b>	DOS memory usage
<b>/E</b>	Extended memory information
<b>/H</b>	Table of expanded memory handles
<b>/M</b>	Memory map
<b>/N</b>	Network information for NET/30
<b>/S</b>	Shadow RAM usage
<b>/X</b>	Expanded memory information

Enable high-resolution VGA or EGA graphics:

**VGAON**

Disable high-resolution VGA or EGA graphics:

**VGAOFF**

Get help:

**LSHADOW ?**

**SHADOW ?**

**VGAON ?**

**VGAOFF ?**







# **INVISIBLE RAM 386**

## **Shadow RAM Manager**

### **And Expanded Memory Manager**

#### **For 386 And 486 Based PCs**

Invisible RAM 386 is a software package that unlocks the extended memory that is present in many 80386, 80386SX, and 80486 based PCs.

Invisible RAM 386 increases the size of DOS memory from the normal 640K up to as much as 736K. Invisible RAM 386 also lets you load memory-resident programs and device drivers into shadow RAM, so that they do not use any DOS memory. As much as 224K of shadow RAM can be made available for memory-resident programs and device drivers.

With Invisible RAM 386, you can tap the power of shadow RAM to greatly increase the amount of RAM available for DOS applications.

In addition, Invisible RAM 386 is an expanded memory manager. Invisible RAM 386 uses the microprocessor's built-in memory management facilities to provide expanded memory compatible with EMS version 4.0. You can use the expanded memory to run EMS applications such as Lotus 1-2-3, Desqview, and Ventura Publisher. As much as 14 megabytes of expanded memory can be made available.

## **Hardware Requirements**

80386, 80386SX, or 80486 based personal computer, equipped with at least 2 megabytes of RAM.