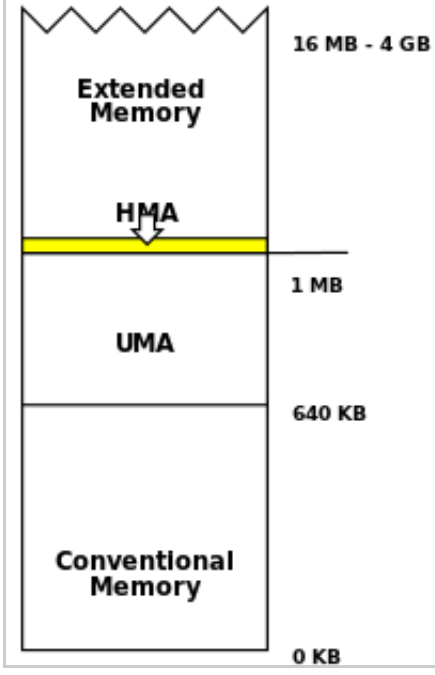
Upper memory area

From Wikipedia, the free encyclopedia

In DOS memory management, the **upper memory area** (**UMA**) refers to memory between the addresses of 640 KB and 1024 KB (0xA0000–0xFFFFF) in an IBM PC or compatible. IBM reserved the uppermost 384 KB of the 8088 CPU's 1024 KB address space for ROM, RAM on peripherals, and memory-mapped input/output. For example, the monochrome video memory area runs from 704 to 736 KB (0xB0000–B7FFF).

However, even with video RAM, the ROM BIOS and I/O ports for expansion cards, much of this 384 KB of address space was unused. As the 640 KB memory restriction became ever more of an obstacle, techniques were found to fill the empty areas with RAM. These areas were referred to as **upper memory blocks** (**UMBs**).



The upper memory area is located between 640 KB and 1024 KB.

Contents

Usage

The next stage in the evolution of DOS was for the operating system to use upper memory blocks (UMBs) and the high memory area (HMA). This occurred with the release of DR DOS 5.0 in 1990. DR DOS' built-in memory manager, EMM386.EXE, could perform most of the basic functionality of QEMM and comparable programs.

The advantage of DR DOS 5.0 over the combination of an older DOS plus QEMM was that the DR DOS kernel itself and almost all of its data structures could be loaded into high memory. This left virtually *all* the base memory free, allowing configurations with up to 620 KB out of 640 KB free.

Configuration was not automatic - free UMBs had to be identified by hand, manually included in the line that loaded EMM386 from CONFIG.SYS, and then drivers and so on had to be manually loaded into UMBs from CONFIG.SYS and AUTOEXEC.BAT. This configuration was not a trivial process. As it was largely automated by the installation program of QEMM, this program survived on the market; indeed, it worked well with DR DOS' own HMA and UMB support and went on to be one of the best-

selling utilities for the PC.

This functionality was copied by Microsoft with the release of MS-DOS 5.0 in June 1991. Eventually, even more DOS data structures were moved out of conventional memory, allowing up to 631 KB out of 640 KB to be left free. Starting from version 6.0 of MS-DOS, Microsoft even included a program called MEMMAKER which was used to automatically optimize conventional memory by moving TSR programs to the upper memory.

For a period in the early 1990s, manual optimisation of the DOS memory map became a highly prized skill, allowing for the largest applications to run on even the most complex PC configurations. The technique was to first create as many UMBs as possible, including remapping allocated but unused blocks of memory, such as the monochrome display area on colour machines. Then, DOS' many subcomponents had to be loaded into these UMBs in the correct sequence to use the blocks of memory as efficiently as possible. Some TSR programs required additional memory while loading, which was freed up again once loading was complete. Fortunately, there were few dependencies amongst these modules, so it was possible to load them in almost any sequence. Exceptions were that to successfully cache CD-ROMs, most disk caches had to be loaded after any CD-ROM drivers, and that the modules of most network stacks had to be loaded in a certain sequence, essentially working progressively up through the layers of the OSI model.

A basic yet effective method used to optimize conventional memory was to load HIMEM.SYS as a device, thereafter loading EMM386.EXE as a device with the "RAM AUTO" option which allows access into the UMA by loading device drivers as devicehigh. This method effectively loads the fundamental memory managers into conventional memory, and thereafter everything else into the UMA. Conventional memory glutton programs such as MSCDEX could also be loaded into the UMA in a similar fashion, hence freeing up a large amount of conventional memory.

Windows

The increasing popularity of Windows 3.0 made the necessity of the upper memory area less relevant, as Windows applications were not directly affected by DOS' base memory limits, but DOS programs running under Windows (with Windows itself acting as a multitasking manager) were still thus constrained. With the release of Windows 95, it became less relevant still, as this version of Windows provides much of the functionality of the DOS device drivers to DOS applications running under Windows, such as CD, network and sound support; the memory map of Win95 DOS boxes was automatically optimised. However, not all DOS programs could execute in this environment. Specifically, programs that tried to directly switch from real mode to protected mode would not work as this was not allowed in the virtual 8086 mode it was running in. This point is now being addressed by x86 virtualization technologies such as Intel VT-x (Vanderpool) and AMD-V (Pacifica). Also, programs that tried making the switch using the Virtual Control Program Interface (VCPI) API (which was introduced to allow DOS programs that needed protected mode to enter it from the virtual 8086 mode set up by a memory manager, as described above) didn't work in Windows 95. Only the DOS Protected Mode Interface (DPMI) API for switching to protected mode was supported.

Implementation

Virtual x86 Mode

Upper memory blocks can be created by mapping extended memory into the upper memory area when running in virtual x86 mode. This is similar to how expanded memory can be emulated using extended memory so this method of providing upper memory blocks is usually provided by the expanded memory manager (for example EMM386). Ironically the application programming interface for managing the upper memory blocks is specified in the *eXtended Memory Specification*.

Shadow RAM

On many systems including modern ones it is possible to use memory reserved for shadowing expansion card ROM as upper memory. Many chipsets reserve up to 384 KB RAM for this purpose and since this RAM is generally unused it may be used as real mode upper memory with a custom device driver, such as UMBPCI (http://www.uwe-sieber.de/umbpci_e.html).

IBM XT

On IBM XT computers, it was possible to add more memory to the motherboard and use a custom address decoder PROM to make it appear in the upper memory area [1] (http://www.textfiles.com/computers/pc869kb.txt). As with the 386-based upper memory described above, the extra RAM could be used to load TSR files, or as a RAM disk.

The AllCard, an add-on memory management unit for XT-class computers, allowed normal memory to be mapped into the 0xA0000-EFFFF address range, giving up to 952 KB for DOS programs. Programs such as Lotus 1-2-3, which accessed video memory directly, needed to be patched to handle this memory layout. Therefore, the 640 KB barrier was removed at the cost of hardware compatibility. This usage of the upper memory area is different from using upper memory blocks, which was used to free conventional memory by moving device drivers and TSRs into the upper 384 KB of the 1 MB address space, but left the amount of addressable memory (640 KB) intact.

See also

- DOS memory management
- Conventional memory
- Extended memory (XMS)
- Expanded memory (EMS)
- High Memory Area (HMA)
- Global EMM Import Specification (GEMMIS)
- LOADHIGH