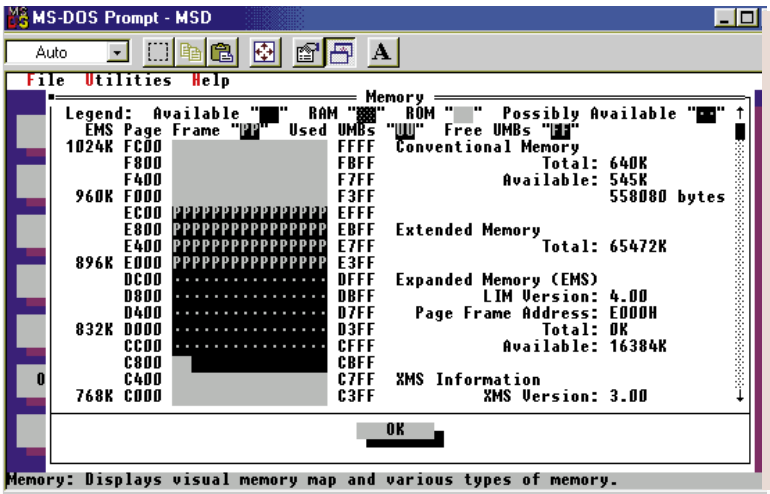
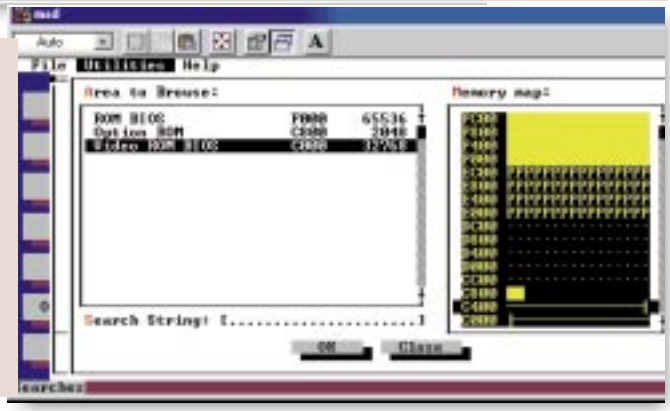
RAM nirvana

Roger Gann shows you how to optimise memory.

Optimising your memory? Why bother?Trying to squeeze every last drop of usable memory might not be high on the list of priorities for a Windows 98 user running a Pentium III but it's still pretty important to all Windows 3.1x users running on 486s or Pentiums. In particular, freeing up as much conventional memory as possible is pretty important to the smooth running of Windows 3.1x and for some applications. If you run out of memory below 640Kb, it does not matter how much extended memory you have got, those applications will not load and run.



◀FIG 1 IF YOU WANT TO GO ABOUT MANUALLY LOADING PROGRAMS INTO UPPER MEMORY, THE MSD UTILITY CAN DISPLAY A MEMORY MAP OF THE 384KB UPPER MEMORY, LISTING USED AND FREE BLOCKS OF MEMORY



▶FIG 2 MSD ALSO HAS A USEFUL MEMORY BROWSER UTILITY WHICH GRAPHICALLY DISPLAYS WHICH AREAS OF UPPER MEMORY ARE OCCUPIED BY WHICH ROM

Gazing back through the mists of time, the problems with memory boil down to that age-old problem of scarce resources caused by a lack of foresight. In 1981, the original IBM PC shipped with 64Kb, and later 128Kb of RAM. Sounds pathetic, but when you put it in the context of other computers with 1Kb or 16Kb, 64Kb was a lot of RAM. It could, of course, seem much more than this. The Intel 8088 which powered the IBM PC could address a massive 1,024Kb of RAM, so you can forgive the PC's designers for being optimistic.

But this optimism proved to be short lived when Lotus 1-2-3 arrived in 1983 and needed an astronomical 256Kb of RAM or more if it could get it — sounds frighteningly like the impact of Windows 9x on memory requirements.

OK, so users could add more memory but they soon came across an inherent limitation in the basic PC's architecture; the 640Kb ceiling. As 1-2-3 became a runaway success and users concocted ever-larger spreadsheets, memory shortage became an acute problem.

If you want to maximise your free memory, it pays to be modern

The 640Kb barrier

While the 8088 can handle one full megabyte of RAM, in practice only about two-thirds of this is available for use (640Kb). This is because IBM erred on the side of caution and allocated a large chunk of the top of that memory address space (384Kb) for system ROMs: things like the system BIOS, the display BIOS and the hard disk BIOS.

IBM over-anticipated ROM usage of this space, and on a typical modern PC only half (192Kb or less) is actually used.

This is largely forgivable, considering the amount of ROM-based software about at the time. The problem is that, as IBM specified the address of the first ROM at 640Kb, the 192Kb of available memory space above this is wasted and cannot be 'seen' by MS-DOS. These chunks of inaccessible memory are called Upper Memory Blocks. With hindsight, if IBM had been a little more daring we could all now be moaning about the 832Kb DOS barrier instead of the one at 640Kb.

So, with an 8088 processor, once your PC had 640Kb fitted, that was it as far as memory expansion was concerned. And any memory fitted above this was more or less wasted.

A way around the barrier was devised by Lotus, Intel and Microsoft, called Expanded or LIM memory which worked by swapping 16Kb (later 64Kb) chunks of memory in and out of a memory window in the 384Kb of memory reserved for ROMs [Figs 1&2]. If the swapping was done fast enough it looked as though you had access to oodles of RAM. The problem was that it couldn't be done fast enough and so expanded memory was slow — if you scrolled to the end of a big spreadsheet that lived in EMS, then things could get turgid.

The 640Kb barrier, as it was later known, became even more of a problem as users wanted to use things like caches and network drivers. Even today, the main culprit is games software: modern DOS games expect to be able to access more than 600Kb of conventional memory and they won't load if it's not available. Curiously, early DOS games were unable to access extended memory, that is memory above 1,024Kb, but preferred to use kludgy old EMS instead.

```
C:\>mem /c/p

Modules using memory below 1 MB:

  Name           Total              Conventional          Upper Memory
--------    ----------------    ----------------    ----------------
MSDOS        29,376   (29K)      29,376   (29K)          0     (0K)
HIMEM         1,120    (1K)       1,120    (1K)          0     (0K)
EMM386        9,856   (10K)       9,856   (10K)          0     (0K)
DISPLAY       8,304    (8K)       8,304    (8K)          0     (0K)
DBLBUFF       2,976    (3K)       2,976    (3K)          0     (0K)
IFSHLP        2,864    (3K)       2,864    (3K)          0     (0K)
WIN           3,728    (4K)       3,728    (4K)          0     (0K)
vmm32        16,832   (16K)      16,832   (16K)          0     (0K)
KEYB          6,944    (7K)       6,944    (7K)          0     (0K)
COMMAND      10,352   (10K)      10,352   (10K)          0     (0K)
DOSKEY        4,688    (5K)       4,688    (5K)          0     (0K)
Free        558,096  (545K)     558,096  (545K)          0     (0K)

Memory Summary:

Type of Memory      Total       Used        Free
--------------  ----------  ----------  ----------
Conventional       655,360      97,264     558,096
Upper                    0           0           0
Reserved                 0           0           0
Extended (XMS)  67,043,328           ?  132,804,608
--------------  ----------  ----------  ----------
Total memory    67,698,688           ?  133,362,704

Total under 1 MB   655,360      97,264     558,096

Total Expanded (EMS)            67,108,864   (64M)
Free Expanded (EMS)             16,777,216   (16M)
Largest executable program size    558,080   (545K)
Largest free upper memory block          0   (0K)
MS-DOS is resident in the high memory area.
```

◄FIG 3 THE MEM UTILITY IS ANOTHER USEFUL TOOL FOR TRACKING DOWN WHAT DRIVERS AND TSRS HAVE LOADED WHERE IN THE MEMORY. HERE, NOTHING AT ALL HAS BEEN LOADED IN UPPER MEMORY

What's possible?

With a modern PC it's possible to access almost all 'lost' memory above 640Kb, to load device drivers and other memory-resident programs there, and thus reduce the hit on conventional memory. It's also possible to load parts of DOS in the 64Kb High Memory Area, (which sits just above the 1,024Kb mark), freeing up even more conventional memory. Making use of all these tweaks, it's possible to reduce the DOS conventional memory footprint to a tiny 13Kb giving you as much as 627Kb of free conventional memory, which should be enough to satisfy the needs of the most awkward game.

The more modern your DOS, the easier it is to manage memory

How do you achieve this state of RAM nirvana? If you want to maximise your free memory, it pays to be modern. The more modern your DOS, the easier it is to manage memory. Prior to MS-DOS 5, little attention was devoted to this subject. Indeed, apart from that spawn of the devil, CHKDSK, there was no way of checking your memory usage.

That all changed with the release of MS-DOS 5 which included the very useful MEM program [Fig 3] which told you exactly what you had in RAM and how much was free. It was an essential tool for optimising memory.

As well, Microsoft shipped an improved EMM386.EXE with MS-DOS 5, an Expanded Memory Manager, which also supported the use of Upper Memory Blocks; the chunks of memory space that lay between the various ROMs in Upper Memory. Even better, it was now possible to exclude device drivers and TSRs from conventional memory and force them to load into Upper Memory.

While it was now possible to maximise your free memory with MS-DOS 5, it wasn't a particularly easy job to do manually and so MS-DOS 6.0 shipped with MemMaker, a utility which automatically carried out all the memory management donkey work . It is not perfect — it is beaten by the optimisers which accompany Quarterdeck's QEMM, the original and still the best DOS memory management software — but even so, it is still pretty good. Additionally, the MS-DOS 6.2 version of EMM386.EXE was improved, which permitted access to a wider range of UMBs than its predecessor.

So, the conclusion is clear: if you've got a low memory problem, one of the first things to do is upgrade your DOS. It doesn't have to be MS-DOS 6.21, either. Caldera or IBM PC DOS 7 is, if anything, even better when it comes to the vexed subject of memory management.

Processor dependence

The same is also true of your PC. The more modern it is, the more free conventional memory you'll be able to squeeze from it. It all depends on what sort of processor it's got:

- **8088** — if you have an XT, that is a PC powered by an Intel 8086 processor or similar (such as the dear old Amstrad PC1640), then your memory management options are severely limited and you are confined to being just plain miserly with RAM, unless you can track down an EMS memory card like the old AST RAMpage.
- **80286** — Things get a little better if you have a 286-powered PC because then you have Extended memory. You'll be able to use the High Memory Area but you won't be able to access Upper Memory, although other memory managers such as QEMM can, in certain circumstances, give you this feature.
- **386SX, 80386 or better** — those with 386SXes or better have their RAM cake and eat it, too. This processor class has the best memory management yet and so provides the greatest flexibility when it comes to optimising memory. Armed with a 386 or better, you'll be able to achieve RAM nirvana.

In next month's column, I'll look at the various changes you can make to your system to maximise free memory. I will show you how to audit your startup files and trim memory-hungry settings.

PCW CONTACTS

Roger Gann welcomes your comments about the 16-bit column. He can be contacted via the PCW editorial office (address, p10) or email 16bit@pcw.co.uk