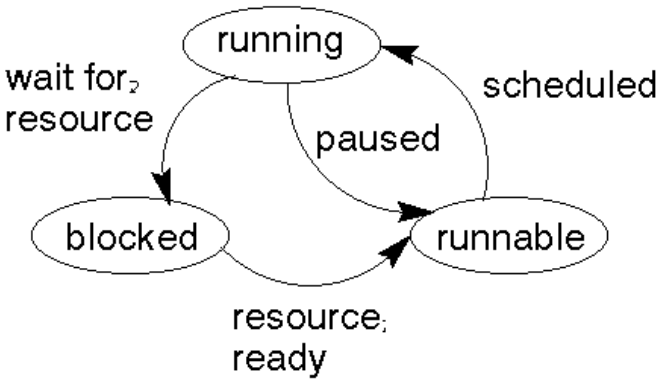
Spencer Oxe

Windows 95

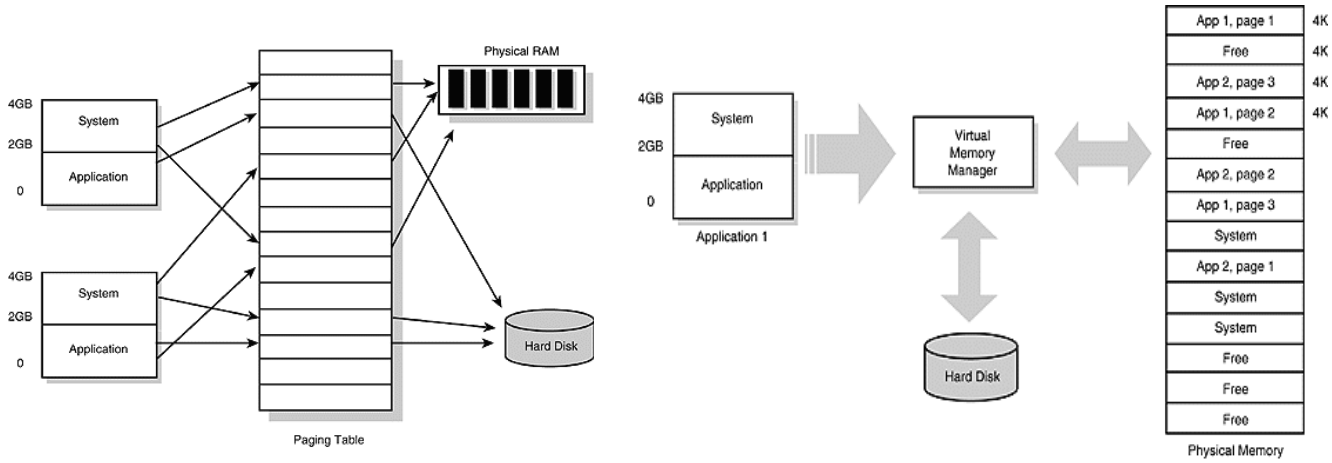
“The Most Important Operating System of All Time” -Forbes

Windows 95 was the first commercially available operating system that was aimed at regular people. In addition, it was powerful enough to support things like modems and CD drives. In a sense it was the next building block for the modern operating system.

Within the Windows 95 framework, processes are assigned a priority number between 0-31. However, processes more typically share one of four priority classes. 24 for real time, 13 for high priority, 9 or 7 for normal priority, and 4 for idle priority. Normal priority has two values because 9 is for processes within the user's focus, and 7 is for other processes. Threads will inherit the priority of their parent process, or can be set to a difference of up to 2. The real time priority is not referencing the OS, because Windows 95 is not a real time operating system. The priority of 24 for real time actually outranks the internal processes of the OS. Meaning overuse of priority above 24 could cause data loss and other problems. Currently used applications will have a better response because of a priority of 9. The idle priority actually means low priority, it is best for processes that can go un-updated for several seconds. Threads can have 4 states in the Windows 95 system. These are active, ready, suspended, and blocked. Active is the currently executing thread. Ready are threads waiting to be run. Suspended is for threads that have called sleep() or suspended themselves. It is also for threads that have finished running. Blocked is for threads that need a resource that isn't currently available, such as a locked section of code. The quantum or time slice for Windows 95 is much shorter than that of UNIX. The former uses only a 20ms slice compared to the latters 100ms. This was done because Windows 95 expected more threads to be of the same process. So the switching between them had less overhead. This allows Windows 95 to have multithreading capabilities.



The Windows 95 scheduler was a single-level half-preemptive scheduler with a feedback mechanism that could briefly alter the priority of a thread through dynamic priority boosting. This is when a thread will get a boost to its priority (such as 4 to 7) because an event such as an I/O occurred to the thread. This can happen to a thread in the ready queue or in the blocked queue which is then moved to the ready queue. In this sense, an I/O bound thread will typically have more priority than a CPU bound thread. Dynamic priority boosting is Windows 95's way of preventing starvation of threads. When ties in priority occur in the ready queue, the scheduler selects the least recently executed thread in another way to prevent starvation. After a few cycles, the boosted priority of the thread degrades back to what it was previously. The scheduler is called half-preemptive because it was preemptive for 32 bit processes but cooperative for 16 bit processes. This is most likely due to 16 bit processes being memory constrained and for compatibility. Due to its combination of priority and preemption, there is the possibility that the scheduler could fall into priority inversion. However, the scheduler has a way to combat this problem. The thread holding access to a resource that is blocked by the high priority process will get a temporary priority

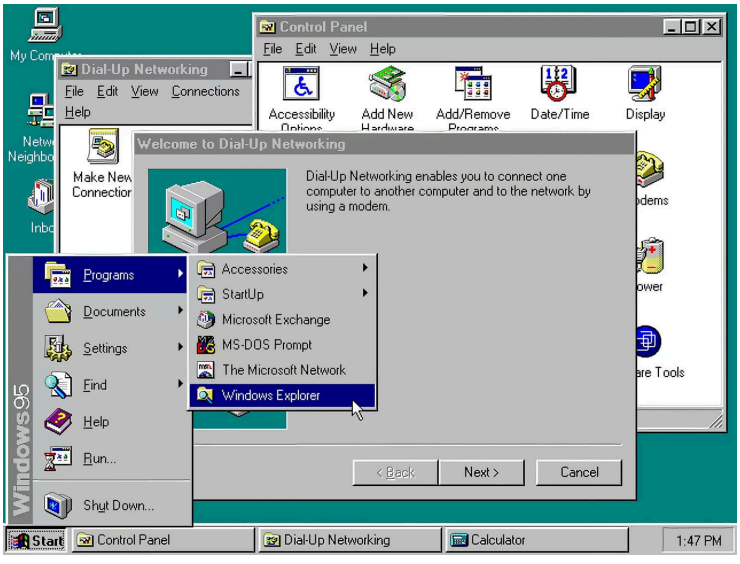


boost. This will ensure the resource will become unblocked and the high priority process gets to run. Pros of this scheduler are the defenses against starvation and priority inversion, and guaranteed responsiveness to threads. Cons of this are that high priority tasks can still starve out low priority ones in a

situation where they don't get a dynamic priority boost. In addition, all preemptive schedulers have the problem of wasting more time switching tasks.

Windows 95 implemented virtual memory using a virtual memory manager. It split the memory into 4k blocks. The memory is managed in a demand paging style, using a paging table for each process to load memory between RAM and disk. The OS utilizes a 32 bit memory space which has a maximum addressable size of four gigabytes. The OS actually reserves 2 GB of the 4 total for kernel mode to prevent overhead from TLB flushes. Windows 95 presumably uses demand paging because it was intended to make use of multitasking multiple processes by keeping more stuff in memory. The page replacement policy used is least recently used. In addition, it also prefers to keep pages in physical memory if they have been modified. Unmodified pages in memory are preferably swapped out. The protection of memory was there on the surface, but programmers could exploit it. Supposedly, it was possible to enter kernel mode from user mode by modifying the CR0 register. Instead of raising a General Protection Fault, it allowed it. In addition, CLI and STI instructions to change interrupts were available in user mode.

Interestingly, Windows 95's architecture is implemented through the use of virtual machines. Using virtual machines, the OS can trick applications into thinking they are running with full access to the computer's resources. There are two types of VM the OS uses, system VM and MS-DOS VM. The system VM is the only one on the OS. It contains the system's components, interface, GDI, kernel, 16 bit and 32 bit applications. The MS-DOS VM is run for every MS-DOS application that is run on the system. This is because MS-DOS applications require sole ownership of the system and multiple cannot be run on the same system. By using VM for each application, the OS can run multiple at once because they each think they have the whole system. The reason Windows 95 needed to use VM to support MS-DOS was because it was used as the bootloader for the system. In addition, MS-DOS supported legacy DOS drivers and some 16 bit applications. Since this operating system was not far removed from MS-DOS, Microsoft most likely kept the MS-DOS bootloader and capabilities in Windows 95 so they didn't have to build a new system from the ground up. In addition, the ability to have more compatibility with MS-DOS systems.



Windows 95 handled I/O asynchronously. This means the OS permitted other processing to continue

while the I/O process finished. For example, the OS could ask for a file copy and then sleep that process until the full operation is complete, allowing other processes to make use of the CPU. Subsequently, due to this and some problems on low end machines, continually wiggling the mouse causes processes like installs to run much quicker. Basically, the OS would get too many messages about completed I/O but not immediately service them. But this did wake the application for user input. Meaning it will handle user input. All in all, wiggling the mouse (user input) will cause the application to process I/O quicker and do the install faster.

Overall, Windows 95 had its pros and its cons. At the time of its release it revolutionized the computer industry with its new framework of operating system. This system had backwards compatibility with its predecessor MS-DOS. In addition, it introduced multitasking and was one of the first OS to make use of the MMU with virtual memory. Despite its flaws, Windows 95 was able to pave the way for the modern brand of operating systems, and it should be heralded as one of the greats.

Bibliography

Boyle, Steve, Mike Forster, Maggie Hamill, and Nancy O'brien. “Windows 95/98 CS 450 Operating Systems Section 002 Fall 2002,” 2002.

Hall, Brian. “Windows95 Scheduling Policies.” beej.us, 1995. https://beej.us/guide/win95sch.html.

Hickey, Matt. “Windows 95 Was the Most Important Operating System of All Time.” *Forbes*, August 24, 2015. https://www.forbes.com/sites/matthickey/2015/08/24/windows-95-was-the-most-important-operating-system-of-all-time/?sh=c532900eb126.

Perforce. *SourcePro*. *Help.perforce.com*, n.d. https://help.perforce.com/sourcepro/current/HTML/index.html#page/SourcePro_Core/threadspl-Win32.38.08.html.

Simpson, Dessa. “Why Did Moving the Mouse Cursor Cause Windows 95 to Run More Quickly?” *Retrocomputing Stack Exchange* (blog), July 19, 2019. https://retrocomputing.stackexchange.com/questions/11533/why-did-moving-the-mouse-cursor-cause-windows-95-to-run-more-quickly#:~:text=Windows%2095%20applications%20often%20use.

Tanenbaum, A. (2001).*Modern Operating Systems*, 2nd edn, Prentice Hall.

Unklejoe. “Wait, There Was No Memory Protection in Windows 98? Did It Implement Virtual Mem... | Hacker News.” *News.ycombinator.com* (blog), December 27, 2016. https://news.ycombinator.com/item?id=13263976.

www.fandecheng.com. “Windows 95 Multi-Tasking.” Accessed May 4, 2024. http://www.fandecheng.com/personal/interests/ewindows/advanced_windows/win95_multitask.htm.

www.informit.com. “Windows 95 Memory Management | 0672311836 | InformIT,” December 11, 2001. https://www.informit.com/articles/article.aspx?p=131307&seqNum=3.