

IT TAKES VIRTUALIZATION TO MAKE AN AGILE INFRASTRUCTURE

Part IV of a Series of AMD White Papers on 64-bit Computing

Just sixty years ago, scientists powered up the ENIAC¹, generally regarded as the first electronic computer. ENIAC's clock ran at five kilohertz, and the system's 17,468 vacuum tubes consumed over 150 kilowatts of power.² ENIAC's software environment was primitive by today's standards. Programs consisted of sequences of register settings, entered via dials on control panels, and small modifications to the internal circuits, implemented like the connections in operator-assisted telephone switchboards.

The industry has come a long way in sixty years. First, the transistor and, later, the integrated circuit enabled the creation of inexpensive microprocessors containing hundreds of millions of transistors, running at multi-gigahertz frequencies, and consuming less than 100 watts. Advances in software technology enabled the productive deployment of these powerful systems. Technological evolution both drives, and is driven by, ever-increasing levels of abstraction in hardware and software architectures. High-level programming languages like Fortran, COBOL, BASIC, C, and Java

allowed programmers to implement software algorithms in a manner divorced from underlying machine architectures. Operating systems provided abstractions that freed programs from the complex and varied details needed to manage memory and I/O devices. Contemporary application software, swaddled within layers of middleware and dynamic linked libraries, must work overtime to determine the physical characteristics of the hardware on which it runs.

Although application packages and middleware have become blissfully unaware of the vagaries of specific hardware implementations, the operating systems that provide this isolation must themselves be totally cognizant of the hardware on which they reside. Details like MAC and IP addresses, SAN LUN assignments, physical memory configurations, processor counts, and system serial numbers become enmeshed within the OS state at the time of system installation. This stateful information locks the OS to the specific hardware on which it was installed and complicates hardware fault recovery, system upgrades, and application consolidation.



¹ The name ENIAC was an acronym for "Electronic Numerical Integrator and Computer."

² Transistor technology had yet to be invented at the time of ENIAC's design. Since power cycling shortened the life of the vacuum tubes, operators soon decided to leave the system powered-up, even when it had nothing to do.

Of course, the same hardware and software architects who created the abstract environments in which today's software and middleware packages reside were not about to let a few annoying details like stateful information stand in their way. They realized that if they could abstract the hardware as seen by the operating system, then they could finesse software's view of the physical configuration on which it was installed. They called their approach "virtualization," and it turns out to be harder than you might imagine. Operating system software likes to think it owns the hardware on which it runs, and does not like to be fooled. It is harder still, given that the x86 architecture came into existence long before notions of virtualization entered the industry, and does not lend itself easily to such notions. In the remainder of this document, we will survey the kinds of problems virtualization can address and how it addresses them. We will review the features AMD added to Next-Generation AMD Opteron™ processors and AMD Athlon™ 64 processors to make AMD Virtualization™ (AMD-V™) an industry leader. We will look at how AMD's ecosystem partners are adapting their software to take advantage of these new virtualization extensions. Finally, we'll delve into the future of virtual technology, and assess the impact it's likely to have on the industry in years to come.

Who Needs Virtual Technology?

Practically everyone who uses or supports computer systems stands to benefit from the emergence of advanced virtual technology that enhances the agility and efficiency of server and client systems alike. Until now, the benefits from virtualization came at some cost in terms of application performance, but the latest AMD Athlon 64 processor and AMD Opteron processor, as well as AMD Turion™ 64 mobile technology, all include AMD-V enhanced hardware that helps reduce the software overhead needed to support virtual machine environments. Suppliers like Microsoft®, Virtual Iron, VMware, and XenSource are rushing to

incorporate support for AMD-V in upcoming software releases. Why all the fuss and effort around virtualization?

The x86-based server market grew exponentially over the past decade, driven largely by a philosophy of "one application, one server." This approach filled datacenters with rack after rack of over-provisioned systems, most operating at less than 15 percent of capacity, but consuming power and generating heat on a 24x7 basis. Even with these low utilization rates, IT managers often need to dedicate three separate systems to each application: one to run the application, one to back up the first system in the event of a hardware failure, and one to serve as a development platform for ongoing development and problem analysis. These systems operate under a variety of current and legacy OS environments, including Windows® NT, Windows Server 2000, Windows Server 2003, Solaris™, UNIX®, and Linux®. IT managers would love to consolidate these disparate workloads onto a smaller number of hardware systems, but are understandably wary of the potential software problems that can arise when they make several independent applications share a single instance of an operating system. Virtualization provides a mechanism to consolidate these applications, along with their existing OS, middleware, and communications environments, on a larger shared system. Each workload continues to see a virtual environment that corresponds exactly to the physical environment of its earlier dedicated system, eliminating the need to change OS or communications parameters. These virtual machines stand ready to respond to external requests, but consume almost no machine resources or power in the absence of such requests — a far cry from the real resources consumed in real, or non-virtual, deployments.

In addition to consolidating existing workloads, virtualization also facilitates the introduction of new



applications in the datacenter. Once IT gets the go-ahead on a new project, development can begin immediately on brand new virtual machines added to an existing physical server. Virtualization essentially allows the enterprise to base its hardware acquisition plans on aggregate demand, rather than the vagaries of any given program. Although not strictly a portion of virtual technology per se, most virtual software environments include utilities that facilitate operational tasks such as the provisioning of new virtual servers, the allocation (and reallocation) of virtual resources, and the assignment of virtual machines (VMs) to physical systems. These utilities simplify the scheduling of planned hardware outages as well as the recovery from unplanned outages. To accommodate planned outages, IT management merely relocates the VMs running on a particular hardware configuration to an alternate system, allowing the original hardware to be taken off-line for service. For unplanned outages, the system operator simply reinitiates the relevant VMs on an operational system until the failed hardware configuration can be restored to service.

Virtualization is changing the way software developers work. Developers must often adapt their code for operation in a wide variety of operating system environments, and then test those codes in the relevant environments. To accomplish this, they would often dedicate specific developer machines to different versions of Windows, UNIX, and Linux. When pursuing software anomalies, they would find the machine with the OS environment on which the bug had been observed, and attempt to come up with a correction. Of course, if the particular system with the required environment had not been used for a while, there was no assurance it would be in good working order when needed. Virtualization lets development organizations maintain a library of virtual machines corresponding to all the specific hardware and software environments on which their software runs. Then, if and when they need



to pursue a software anomaly, they merely load the proper virtual machine, and they're ready to pursue the problematic code.

Virtualization greatly eases the task of application migration to new versions of operating systems for both client and server applications. IT can install multiple VMs, each running different versions of the OS, and migrate specific applications to the "new and improved" OS at a pace convenient to IT and the end-user, rather than on the all-or-nothing basis that has characterized software transitions in the past.

Virtualization may even alter the way organizations deploy desktop technology. The recently formed Virtual Desktop Infrastructure Alliance allows IT administrators to create and manage "desktop virtual machines" on servers within datacenters. End-users can access these desktop environments at any time and from any place, using thin client devices (or thin client access utilities on more fully configured systems). Even old, underpowered systems can use the Remote Desktop Protocol (RDP) to access more powerful virtual PC desktops arrayed with up-to-date software versions. This approach to client

deployment can lower support expenses, as well as hardware acquisition costs, since the virtual PCs reside on centralized servers in a managed IT environment, eliminating the need to visit the client's actual desktop system for most maintenance activities.

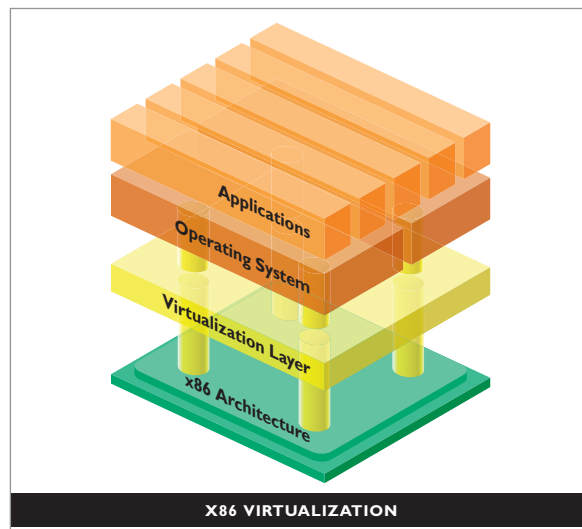
Last, but far from least, virtualization will play an ever increasing role in creating more secure and robust client environments. For example, on one client system, IT might install two virtual machines, one that handles sensitive corporate data, and a second for less secure end-user tasks. It could block the first VM from downloading unauthorized applications, screen savers, and other security threats, while allowing the second VM access to less secure downloads and material.

The Technology of Virtualization

Anyone who has ever mastered a magic trick like sawing a woman in half or pulling a rabbit out of a hat knows that creating an illusion requires planning and deft execution. Convincing an operating system like Windows or Linux that it has exclusive control of a computer system that, in fact, it shares with other operating systems also requires the same degree of planning and execution. Architects have pursued three different strategies in their pursuit of virtual technology. All require the presence of *hypervisor software* that allocates basic machine resources including CPU time and memory. All consider the OS software running on virtual machines to be *guest operating systems*. They differ with regard to the techniques they use to convince the guest OS that it is in charge of the system.

- Software-implemented virtualization techniques cause the hypervisor to trap the machine operations the OS uses to read or modify the system's status or to perform input/output operations. Once trapped, the hypervisor emulates these operations in software and returns status codes consistent with what the real hardware would deliver. The

good news is that since this approach operates invisibly from the perspective of the guest OS, it requires no changes to the guest OS or the applications running under the guest. Off-the-shelf versions of software developed long before virtualization ever came to the x86 world, like MS-DOS, Windows 3.1, or Windows NT 3.5 can be installed as guest operating systems. The bad news is that all that instruction trapping and emulation can reduce overall system performance significantly in I/O-intensive environments. EMC's VMware, the best known and most popular virtual environment for x86 processors, has used variations on this software approach in all the products it has delivered to this point. Microsoft's Virtual PC and Virtual Server packages use a similar approach.



- Partial-virtualization (sometimes known as para-virtualization) eliminates much of the trapping and emulation overhead of the software approach by requiring that the guest OS cooperates in creating the virtual illusion—that it essentially agrees to be fooled by the hypervisor. Para-virtualization requires the use of specially modified guest operating systems that understand the game is



rigged and know how to play along to maintain the virtual illusion. This approach precludes the ability to run off-the-shelf and legacy operating software in para-virtual environments. Xen, the open-source community's approach to virtualization, uses para-virtualization as the basis for its technology.

- Hardware-assisted virtualization relies on hardware extensions to x86 system architecture to eliminate much of the hypervisor overhead associated with trapping and emulating I/O operations and status instructions executed within a guest OS. The latest processors from both AMD and Intel include hardware virtualization assists known as AMD-V™ and Intel VT. Key hypervisor suppliers (Microsoft, VMware, and XenSource) support these features in their software. CPU extensions solve only part of the virtualization problem, since they require the hypervisor to do lots of work to finesse input-output operations, adding overhead to each I/O call. A complete solution requires the virtual mapping of I/O devices, which in turn requires changes to the chipsets and I/O bridges that link system processors to I/O buses like PCI Express. AMD and Intel have each issued specifications for chipset extensions consistent with their system architectures, and hardware incorporating these extensions will likely emerge in 2007.

The good news is that once these processor and chipset extensions, and the software that supports them, reach the market, end-users will have access to advanced virtualization technology that rings in an era of enhanced agility in IT operations with little incremental software overhead. There is no bad news in this regard.

The Virtual Ecosystem

The delivery of all important industry-standard technologies, including virtualization, relies on a

network of technology providers. Software developers like VMware and Connectix (now part of Microsoft) offered the first packages that enabled x86 virtual environments. Over the years, VMware has augmented its basic hypervisor product with a suite of utilities that allows end-users to create, provision, and manage an array of virtual machines within a datacenter. XenSource and Virtual Iron have emerged with Xen-based virtualization schemes principally targeted at the Linux market. Some have assumed that the introduction of hardware virtualization assists in processors will narrow or eliminate the need for virtualization software, but this is far from the actual case. All virtual environments require a hypervisor of some form to allocate real machine resources to virtual machines; processor and chipset extensions assist the hypervisor, but cannot replace it. So, as enhanced hardware shrinks the overhead associated with virtualization, end-user adoption of the technology will increase and create new opportunities for these ISVs.

AMD has extended its architectures to improve virtualization performance, but there remains much room for subsequent innovation in this area. Unlike instruction set enhancements (like AMD's pioneering 64-bit extensions to the 32-bit x86 architecture), AMD-V operates behind the scenes, and directly impacts only a few areas of operating system and hypervisor code. This allows CPU designers to innovate with regard to virtualization support, both to enhance functionality or improve performance. End users should not assume that competitor's virtualization extensions do the same things, and offer similar levels of performance. The devil is in the details, a few of which are outlined in the feature box. Even if system purchasers do not "lift the hood and check out the engine," they should ascertain virtual (as opposed to real) machine performance as part of their acquisition checklist.

Chipset suppliers like ATI, Nvidia, and VIA plan to incorporate I/O virtualization capabilities into their I/O bridges. AMD's I/O virtualization spec differs from Intel's, which becomes just one more way the chipsets designed for AMD's revolutionary Direct Connect Architecture must differ from those designed to tie into Intel's traditional front-side bus approach.

Suppliers of system management software like CA's Unicenter and HP's System Insight Manager have extended their packages to manage virtual as well as physical resources. The shift to virtual environments will enable IT managers to consolidate workloads and allocate system resources with a far greater degree of granularity than they have at present. This in turn will create the need for software to meter, provision, and allocate system resources in a more autonomic manner.

AMD's Real Roadmap for Virtual Technology

Just in case you haven't been paying attention up to now, the goal is to make sure you understand that AMD believes virtualization will play a big part in the future of client and server computing, and that AMD is backing up its words with a solid roadmap of virtualization solutions.

AMD added hardware virtualization capabilities (along with DDR2 support and a few other enhancements) to the Next-Generation AMD Opteron™ processor, AMD Athlon™ 64 processor, and AMD Athlon 64 X2 Dual-Core processor, as well as AMD Turion™ 64 X2 dual-core mobile technology. Of course, AMD is not planning on resting on its laurels. It is working with chipset partners to incorporate the features outlined in the IOMMU (I/O Memory Management Unit) spec it issued early in 2006. And, it's working with ecosystem partners to help make sure they will be ready to support these new features on a timely basis.

AMD is not stopping there. Its quad-core processor architecture plans include features that extend the capabilities of AMD-V™ and improve its performance. AMD plans to demonstrate once again how its Direct Connect Architecture enables AMD processor-based systems to outperform those based on legacy front-side bus based approaches. You won't have to wait too long to try these processors yourself—AMD plans to start shipping quad-core processors in 2007. Because of AMD's commitment to platform stability, its OEM system partners will be able to drop these quad-core processors into the DDR2-based platforms they started shipping in 2006.

The Future Virtually at Your Fingers

Virtual technology is the latest in a long line of technical advancements that have increased the level of system abstraction and enabled IT users to harness ever-increasing levels of computer performance. Virtualization essentially decouples users and applications from the specific hardware characteristics of the systems they use to perform computational tasks. This change will usher in an entirely new wave of hardware and software innovation in years to come. Virtualization will simplify system upgrades (and in some cases may eliminate the need for such upgrades), by capturing the state of a virtual machine, and transporting that state in its entirety from the old to new host system. Virtualization will enable a generation of more energy-efficient computing. Processor, memory, and storage resources that today must be delivered in fixed amounts determined by real hardware system configurations will be delivered with finer granularity via dynamically tuned virtual machines in the future. The combination of virtual technology and powerful AMD technology-based processors allows users to deploy computing resources in more agile, efficient, and cost-effective ways. AMD is proud to participate in this process and assist its customers in extracting ever-increasing value from their IT investments.



MEMORY: THE FIRST VIRTUAL RESOURCE

Long before computer scientists came up with the notion of virtualizing an entire system, architects had already invented techniques to virtualize memory management¹. Virtual memory technology lets a system with a limited amount of physical memory look much larger to application software. To create this illusion, the OS stores the full memory image of the application and its data on the system's hard drive, and transfers required pieces of this image into the system's DRAM memory as the program executes. The system you are using to read this document on-line (assuming you didn't get a printed version) undoubtedly uses some form of virtual memory management.

To translate the virtual addresses seen by each application into physical DRAM memory addresses, the system relies on a map (known as a page table) that contains vectors linking chunks of virtual memory to real memory. Modern x86 processors include hardware features known as translation look-aside buffers (TLBs) that cache the translation vectors for recently accessed chunks of memory, thus speeding up the process. TLBs play a role in virtually all memory references, so the manner in which they perform their translations can play a big role in determining overall system performance.

Architects soon learned that TLB design can seriously impact multitasking system operation. Most tasks in such systems have unique page tables. This forces the operating system to reset (or, more colorfully, "flush") the TLB each time it switches from one task to another. Then, as the new task executes, its page table entries fill up the TLB, at least until the next task switch. This constant flushing and reloading can really eat into performance, especially if each task runs for only a few milliseconds before the next switch.

To mitigate the impact of task switching, architects added a "task ID field" to each TLB entry. This allows the system to retain the mapping information in the TLB when switching tasks, since it only uses the entries for the task actually executing at any point, which in turn eliminates the need for performance-inhibiting TLB flushes. At least until virtualization entered the scene. Since the guest OS running on a virtual machine is unaware of other guests, it can only assign unique task IDs within its own environment. Thus, multiple VMs can have tasks with the same ID, confusing the TLB and making a real mess. There's a simple solution to this problem — the hypervisor merely flushes the TLB every time it switches from one VM to another. This forces the tasks executing in the next VM to reload the TLB with their own page table entries. Unfortunately, this approach seriously impacts virtual system performance, giving architects everywhere that *déjà vu* feeling all over again.

AMD's architects had a better idea. They merely added a new, VM-specific tag called an "address space identifier" (ASID) to the TLBs in their new AMD-V™ enhanced processors. Each VM has a unique ASID value, known only to the hypervisor and the TLB hardware. The ASID is invisible to the guest OS, thus eliminating the need to modify the guest, preserving the virtual illusion and avoiding any performance degradation. Sounds simple, doesn't it?

¹ The Atlas computer at the University of Manchester was the first system to incorporate virtual memory technology and became operational in 1962.

