

About hardware virtualization features and real-time hypervisor software

by Paul Fischer, TenAsys

The result of combining hardware virtualization features with real-time hypervisor software is a single embedded computer system platform that can serve the real-time and reliability requirements of tomorrow's applications.

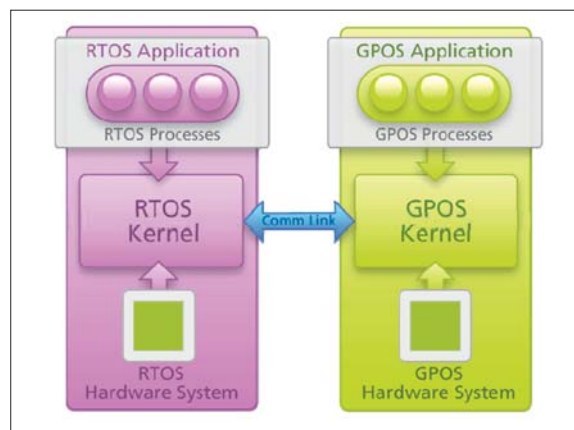


Figure 1. Deploying an RTOS and GPOS on a single hardware platform

■ We are now in a period when real-time operating system development is proceeding hand-in-hand with the evolution of processor silicon. The most interesting thing to happen to processor silicon over the past year is the advent of multi-core CPU chips with hardware support for virtualization of operating systems. These processor developments are enabling a new generation of real-time systems that combine fast responsiveness with reliability and lower system cost. Using system software enabled by the new processor features, real-time operating environments can now work cooperatively with a wider range of application environments than has been possible before.

Virtualization support in silicon is being promoted by CPU vendors as a technique for enabling more-efficient servers to be built for commercial applications, but it has the potential of benefiting embedded RTOS applications just as much or more. The new processors hardware support for virtualization enables more efficient sharing of a single hardware platform between multiple operating systems. In the case of embedded applications, this allows for more secure and reliable partitioning of I/O resources between real-time and general-purpose operating system tasks. The new processors simplify the development of high-performance multi-OS systems by performing in hardware many of the functions that tradition-

al virtual machine managers (VMMs) have performed in software.

An example of a processor family that offers hardware virtualization support, the new Intel Core microarchitecture processors include a collection of instructions referred to by Intel as VT-x. The VT-x instructions are designed to support the hosting of multiple virtual machines on a single hardware platform, where each virtual machine can host its own private copy of a complete protected-mode operating system. The VT-x instructions, along with a special virtual-machine control structure (VMCS), enhance the ease with which a VMM can monitor and control access to hidden elements of the Intel Architecture processor. For example, they improve the ability to control access to certain system level instructions and to monitor the state of page tables.

Virtual machine managers, sometimes called hypervisors, have existed for many years. They allow multiple general-purpose operating systems (GPOSs) to share the resources of a single hardware platform. These existing solutions usually coordinate multiple copies of a single GPOS (e.g., different revision levels or different configurations of the same OS) on an IT server to simplify maintenance and eliminate interference between incompatible server applications that must run on a single hardware plat-

form. These existing VMM implementations will eventually take advantage of the new processor virtualization support to improve their performance and feature sets.

There is a different class of system that can also benefit from the VMM operating approach, it includes machine control applications that require real-time determinism and also need the services of a general-purpose operating system in a single platform solution. The VMMs used on servers don't offer specific support for real-time systems, so new VMM software is being developed for embedded systems.

Traditional VMM software is designed for IT server applications, where the virtualization of and sharing of all the hardware resources on a single platform are the primary goal. In the case of real-time applications, virtualization of all the platform I/O for use by multiple operating systems is not necessarily a desirable goal. It impacts the performance and determinism of real-time operations and makes it impossible to guarantee with certainty the state of those I/O devices that are to be used exclusively by the real-time control software. Without a VMM approach designed for real-time systems, embedded system designers will often employ multiple hardware platforms, dedicating separate system hardware to a GPOS (e.g., for the implementation of human-machine interfaces) and

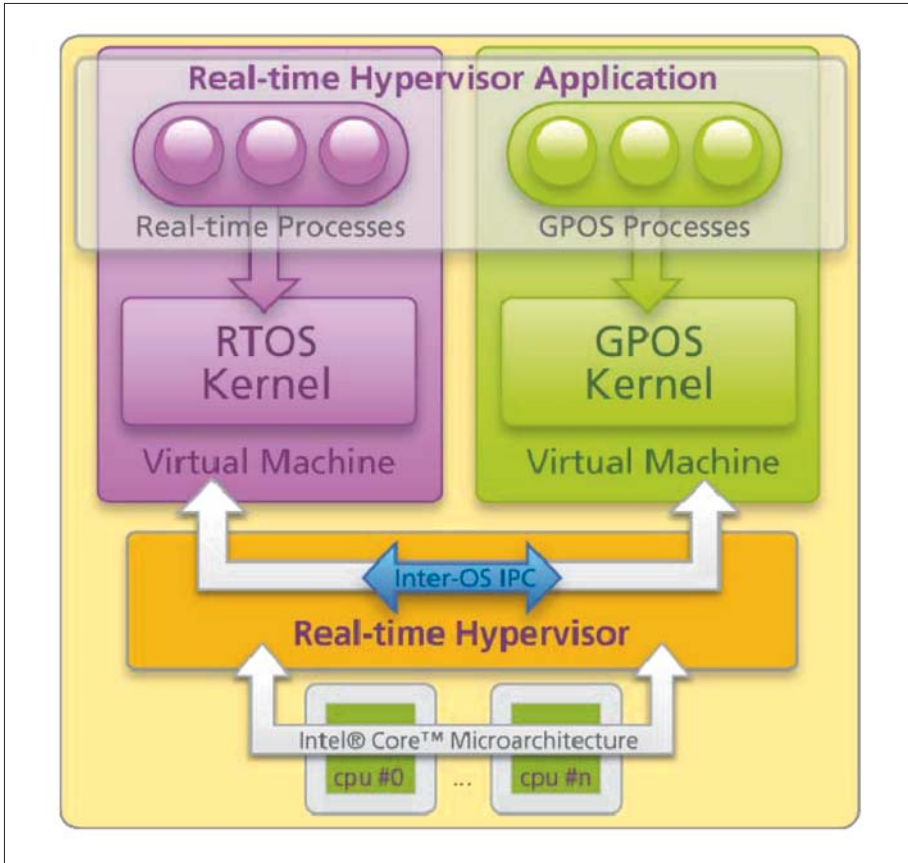


Figure 2. Multiple cores can substantially improve the performance of multi-OS solutions.

time hypervisor it is also possible to restart, or reboot, the Windows environment without disturbing the operation of the RTOS environment, something that cannot be achieved with existing approaches to real-time Windows.

Does the OS need to change in order to take advantage of hardware support for parallel processing? Not when a real-time hypervisor is used. A real-time hypervisor leverages the virtualization facilities of the silicon, to partition the system's I/O between real-time and general-purpose processing. With the real-time hypervisor installed, heterogeneous "guest" operating systems can be hosted with little or no modification. An example of software that implements the above functionality is TenAsys' new real-time hypervisor, code-named "Hearth". The Hearth software runs on Intel Core microarchitecture processors and dedicates OS environments to run on separate cores, with fast inter-OS communication. Hearth enables hard real-time performance with worst-case interrupt latencies of less than three to five microseconds. Using the Hearth real-time hypervisor, control peripherals can be managed by the RTOS, and enterprise and GUI peripherals can be managed by a GPOS such as Linux or Windows. ■

an RTOS (for tasks that require reliability and deterministic responsiveness). By taking advantage of the virtualization support being provided by CPU vendors, it is possible to combine real-time and general-purpose processing on a single system, by using a real-time VMM designed for machine control applications. Among the embedded system applications that can benefit from a real-time VMM approach are those with legacy RTOS code that needs to be upgraded to incorporate GPOS applications without making significant changes to the real-time software, especially where concerns of reliability and certification of the real-time code are involved (for example real-time medical, military, and aerospace applications). The benefits of deploying an RTOS and GPOS on a single hardware platform include lower system cost and shorter product development time, plus higher system performance through faster and simpler communications between the two operating environments.

Most of Intel's Core microarchitecture processors available today, or those that will be available in the future, contain multiple processor cores to enhance system performance while controlling system costs (multiple processor cores can share system resources such as memories and power supply components). Running a VMM on a dual-core CPU offers an additional degree of freedom, the ability to provide

multiple operating systems with simultaneous access to CPU cycles. Further, a VMM that is real-time aware can ensure that real-time tasks run on one CPU core under control of the RTOS, while general purpose processing is restricted to the remaining core(s). With dedicated CPU hardware, the RTOS has access to 100% of the bandwidth of that CPU core for real-time processing, minimizing interrupt latency.

Multiple cores can substantially improve the performance of multi-OS real-time solutions. For example, TenAsys' INtime real-time for Windows product already takes advantage of dual-core CPUs to improve interrupt latency performance by a factor of ten when compared to an identical configuration running on a single-core processor. INtime's support for real-time task processing shares the hardware platform with Windows but isolates the two environments in order to allow the INtime RTOS to run unimpeded by Windows.

Implementing a real-time Windows system with a VT-x-aware VMM, called a real-time hypervisor, further improves task isolation and real-time performance of an embedded system. The real-time hypervisor can completely isolate I/O and interrupts meant for the real-time system from the Windows environment, ensuring that no Windows application or driver will ever interfere with real-time resources. With a real-