I_2O IRTOS

David Wilner
Wind River Systems
Chief Technical Officer

IxWorks: Delivering the Promise of I_2O

A wise economist once observed that human wants are insatiable. Nowhere is this phenomenon more visible than in the computer market. Users clamor for--and get--ever-faster and more powerful central processors for their systems. With these faster CPUs--and the subsequent push into the world of graphics, multimedia and the Internet that they have engendered--it hasn't taken long for another bottleneck to become apparent. No longer does the user lament: "It doesn't calculate my spreadsheets fast enough." Now the complaint is: "It doesn't load Web pages fast enough."

Evidence of this exploding demand for I/O bandwidth is everywhere. For years, 10 megabits per second for Ethernet transmission was entirely acceptable; now 100 megabits is not uncommon, and users look to FDDI and ATM to provide several hundred megabits or gigabits per second. Not long ago, a 100 megabyte disk drive was impressive; now two gigabytes is virtually a prerequisite for the serious developer. It all adds up to tremendous--and growing--I/O requirements.

The right tool for the task

Faster CPUs cannot solve this problem unaided, because what makes a good central processor is not necessarily what makes a good I/O processor. In fact, as CPUs get faster through the use of sophisticated techniques like pipelining and parallel processing, they may actually get worse at coping with the constant interrupts generated by I/O processing. The answer: a highly flexible, powerful I/O processing system to offload these functions and allow the CPU to proceed as efficiently as possible.

The applications of such an advanced I/O processing system include, at the outset, primarily servers--Internet servers, video servers, network servers--centralized systems managing a lot of data, often in symmetric multiprocessing configurations. But the same I/O performance improvements could also apply to workstations, such as those used for EDA and CASE applications.

This next-generation I/O system requires a software framework into which many drivers from multiple vendors can fit, and some arbiter among them so they can share processing resources. In addition, it must be able to manage the increasing complexity and rapid evolution of the software going into I/O subsystems, including such things as network protocols and RAID software. (The I/O processing system can even bypass the CPU altogether for peripheral-to-peripheral communications: for example, loading multimedia files off the disk directly to the network, without causing any interruption to the main processor.)

To address all these requirements, this ideal I/O system must incorporate a powerful, general-purpose real-time operating system. This operating system would have clearly defined, published application program interfaces (APIs), so network operating system (NOS) vendors like Microsoft and Novell could simply create generic drivers for each type of peripheral device that a user might attach to a system. From the perspective of the peripherals vendors, these APIs would also greatly reduce the burden of writing of device drivers; rather than having to create separate versions for each NOS, they need only write their drivers to be API-compliant.

A new model of partnership

When Intel set out to turn this ideal into the reality now known as I_2O, they started a strategic relationship with Wind River Systems. David Wilner, Wind River's co-founder and chief technical officer, was involved in the I_2O working committee from its inception, and wrote the software specification (IRTOS) for the I_2O core API. Wilner also served as head of the engineering team that created IxWorks(tm). IxWorks is a customized version of Wind River's VxWorks®, the most popular operating system for the i960***** processor family. It is fully compliant with the I_2O IRTOS specification.

To facilitate the adoption of I_2O, a license for IxWorks comes with--and is included in the price of--each i960RP processor sold. For the peripherals developer, this can eliminate the time-consuming evaluation and negotiation process involved in the purchase of real-time operating systems and development tools. In addition, since IxWorks can only be used on the i960RP, and i960RP users automatically have the IxWorks license, Wind River can make useful information such as IxWorks updates freely available to developers on the World Wide Web.

To complete the design, developers then need only acquire Wind River's powerful Tornado(tm) for I_2O integrated development environment to complete their projects. Intel is including a 30-day evaluation copy of Tornado for I_2O with each i960RP evaluation board, with additional support available from Wind River's field application engineers and customer support staff.

Developers can thus become familiar with the Wind River tools immediately, which will shorten the learning curve and greatly accelerate product development. This single standards-compliant turnkey package--complete with API, operating system, guaranteed hardware integration, compilers, debuggers, shell, incremental loader for iterative testing, and access to state-of-the-art development tools--should help to bring the first I_2O -compatible peripherals to market in record time.

IxWorks

IxWorks provides for all the elements of the I_2O standard: an event-driven driver framework, host message protocols, and executive modules for configuration and control. It greatly simplifies the writing of basic device drivers, provides NOS-to-driver independence, and provides a prioritized multi-threaded framework that allows I/O software from multiple vendors to coexist safely. It is fully scalable across all I_2O configurations, from dedicated on-card i960RP processors to open "on-motherboard" implementations to complex distributed I/O systems servicing multiple CPUs.

In addition, IxWorks contains a number of features that make advanced I/O subsystems possible. It supports peer-to-peer communications, enabling two IxWorks-based systems to talk to one another without intervention, for greater offloading of the CPU. It also manages hierarchical driver modules and intermediate service routines (ISMs)--those software tasks like network management and RAID control algorithms that are increasingly becoming the responsibility of the I/O processing system, despite being a step or two removed from the actual hardware functions.

As with VxWorks, the core of IxWorks is the wind(tm) kernel. This microkernel design allows the real-time operating system to minimize system overhead and respond quickly to external events. The wind kernel supports a full range of real-time features including fast multitasking with 256 priority levels, microsecond interrupt handling, and both preemptive and round-robin scheduling. It also provides efficient intertask communication mechanisms and three types of semaphores for controlling critical system resources -- binary, counting, and mutual exclusion with priority inheritance.

In creating IxWorks, Wind River engineers made a number of enhancements and customizations for the demanding I/O processor (IOP) environment. The event queues are implemented inside

the wind kernel for better performance. By taking advantage of the i960RP hardware's message passing capabilities, IxWorks both eliminates the inefficiencies of spin-locking and provides high performance in the symmetric multiprocessing (SMP) configurations that characterize many servers. High-utilization "fast-paths" are optimized to ensure that critical operations are executed as rapidly as possible, and key elements of the API are written in assembly language for the fastest possible processing speed. In addition, IxWorks utilizes a robust object-oriented structure in which all objects are owned by other objects. If an object is destroyed, so are all its related objects--a technique that enables drivers to be loaded and unloaded "on the fly" without leaving vestiges of themselves behind.

Because DMA is so critical to efficient I/O, the developers of IxWorks came up with a highly sophisticated DMA model. Typically, the complexity and potential coordination problems of DMA have given rise to inefficient implementations, whereby a single driver will "attach itself" to a single DMA unit and keep it even though the unit sits idle for most of the time. Worse, other drivers are prevented from carrying out DMA operations because they are not attached to a particular DMA unit. Responding to this problem, Wind River developed an elegant queued DMA structure. In this scheme, any driver can request a DMA operation, all the requests are queued, and each is processed in turn as soon as an appropriate DMA unit becomes available. Notification of the DMA completion is sent back asynchronously through the event queue mechanism. In addition to being far more efficient, this model also makes DMA far easier to use. IxWorks also shields the driver writer from detailed configuration issues, such as the number of interrupts hooked up in a particular peripheral or the number of available DMA channels. The API is designed to hide all such specifics; in effect, IxWorks acts as the BIOS for the i960RP.

Wind River plans to offer additional modules to the base IxWorks, leveraging the company's years of experience in embedded development to offer a variety of components to I_2O developers.

How the API works

Rather than building custom drivers for each network operating system (NOS) they support, developers using I_2O need only write their hardware drivers to comply with the API. Wind River's design goal was to create a powerful driver framework that would make writing basic drivers as simple as possible--essentially a "fill in the blanks" approach. The driver writer merely supplies a group of functions that correspond to all the possible events that could occur to that driver. For example, if you as a developer want to write a disk driver, the API specifies the handful of routines you have to supply: one to respond "read" messages, one for "write" messages, and so on.

All the events that come into the system--the interrupts, the timers, the requests from the host, replies from other drivers--are placed into the event queue for the driver. The system then sets priorities and automatically dispatches calls to the appropriate driver function in response.

Tornado for I_2O

Users of IxWorks will benefit greatly from their access to Tornado for I_2O, the industry's richest and most advanced environment for driver development. As its name suggests, Tornado for I_2O is a version of Wind River's market-leading Tornado development environment customized specifically for I_2O peripheral development.

Tornado for I_2O provides a complete, visual and intuitive environment that is easily extended and customized. Its rich collection of integrated host-resident tools allows developers of intelligent peripherals to:

* Build I_2O drivers

* Test and debug I_2O drivers

* Package and deliver I_2O drivers and products

* Monitor and optimize I_2O system performance

The Tornado for I_2O tools focus on the developer's need for interactive response, interpretive interfaces and incremental development. Because they are fully integrated with and aware of IxWorks, they give the developer a feeling of being close to the target system--thereby reducing driver development time and resource consumption.

Build tools. Tornado for I_2O includes the i960RP C compiler, as well as a collection of supporting tools that provide a complete development tool chain: compiler, assembler, linker and binary utilities. Tornado for I_2O also provides an I_2O module builder, which creates I_2O -loadable modules.

Test and debug tools. Tornado for I_2O test and debug tools include the dynamic loader, the CrossWind(tm) debugger, the WindSh(tm) interactive shell, and a powerful system browser.

The dynamic loader allows the driver writer to perform interactive loading, testing, and replacement of the individual object modules that comprise a driver. It offers high-speed download over the I_2O host-target connection (PCI bus, serial line, or I2C interface).

The remote source-level debugger, CrossWind, is an extended version of the popular GNU Source-Level Debugger (GDB) with a powerful native Windows GUI. Using it, the developer can debug I_2O drivers by setting breakpoints on desired I_2O components--for example, at particular lines of code, or when any event occurs for a device, or when specific events occur for a device.

CrossWind also lets the developer view I_2O data structures such as formatted I_2O messages, IxWorks object descriptors, and driver-private data structures. A variety of windows display source code, registers, locals, stack frame, memory and so on. Application code can be viewed as high-level C source, as assembly-level code, or in a mixed mode that shows both the high-level source code and the corresponding assembly code. CrossWind's comprehensive Tcl scripting interface allows the I_2O developer to create sophisticated debugger macros or extensions for specific requirements.

WindSh is a command shell that provides interactive access to all IxWorks facilities. It can interpret and execute almost all C-language expressions. The shell can be used to control and monitor I_2O drivers; format, send and receive driver messages; examine hardware registers; and run automated I_2O test suites. The shell also provides essential debugging capabilities, including breakpoints, single stepping, stack checking, and disassembly. This interactive environment with incremental linking allows for a highly productive "fix and go" development cycle, where the driver can be built and tested piece by piece.

The system browser is a graphical companion to WindSh. The browser provides visibility into memory allocation and all I_2O objects including drivers, devices, timers, interrupts, DMA queues, events, event queues, semaphores, message queues, and threads. The information on these objects is updated either automatically or on demand. The browser makes it simple to monitor the state of the IxWorks system, including memory pools, system buses and configuration tables. Objects are displayed in an intuitive manner, often with collapsible "treeviews" to hide or show information. Much of the browser is written in Tcl, allowing considerable customization.

Packaging and delivery tools.

Tornado for I_2O provides a variety of tools for packaging and delivering I_2O drivers and products. These include configuration tools for tuning of memory usage, utilities to create I_2O -compatible driver distribution disks, and tools to create ROM-based I_2O drivers for dedicated IOPs on adapter cards.

Monitoring and optimizing tools. The optional WindView system visualizer is a powerful tool for tuning driver and system performance. With one-microsecond time-stamped resolution, it enables the developer to see and measure exact sequences of I_2O events and messages, driver execution and communication times and durations, interrupts, timers, and DMA activity. WindView monitors all these functions on the i960RP target without stopping I_2O activity.

Service and support

Although the IxWorks license will be sold through Intel, I_2O developers worldwide can contact Wind River directly for Tornado for I_2O and a wealth of related Wind River services, ranging from telephone support and training classes to consulting and driver development.

* Trademark of Intel Corporation