New intelligent multithread processors benefit from integrated and innovative tools for code debug, programming, and test

By Bob Burrill

With the increasing need to move processing outside the confines of a single box to an interconnected world of devices, today’s single-thread processor architecture is showing its performance limitations. At every point in the new networked infrastructure lie devices in need of new services, improved bandwidth, deeper content inspection, complete data security, and accelerated applications.

This new packet-oriented environment is characterized by high memory access latencies, which are not effectively managed by conventional processor architectures. In this case, single-thread processing will severely impact processor performance and work-load efficiency when memory accesses aren’t immediately serviced and packets back up in the queue, sending conventional processors into a stall and wasting valuable processing cycles.

The new wave of intelligent, multithread processors are system-on-chip processors that combine the functions of several ASICs, network processing units, and coprocessors to reduce hardware and software costs and complexity. The multicore, multithread processing capability of these processors enables throughput many times faster than traditional single-thread processors.

The need for a new generation of processors

The fundamental difference between core speeds and memory or I/O latencies means that today’s processor architectures cannot deliver meaningful performance for connected computing. These processors are usually waiting for data. The need for data security significantly contributes to these delays (stalls) as well, as the processor must not only move the data, but also run security application code at the same time. These stalls can occur as much as 90 percent of the time that the processor is active.

Until now, processor designs attempted to overcome these barriers by using deeper pipelines, superscalar (multi-issue) operations, and larger caches. However, those efforts are yielding diminishing returns because the data processing in network applications occurs at the macro or packet level, not at the instruction level.

Next-generation processors, such as the Raza Microelectronics (RMI) XLR processor family, overcome these issues by integrating a set of enhanced MIPS64-based cores into a single processor chip. Each core includes four hardware threads, integrating up to 32
threads or virtual CPUs in eight identical processor cores. Unlike single-threaded, multi-issue designs, this architecture is well suited to today's increasingly network-oriented computing environment and can significantly reduce stalls. Using up to 32 virtual processors enables deep inspection of multiple packets at once, fast decisions on routing within the chip, and quick passing of the processed packets on their way, in their proper order.

Applications of the processors include integrated security (firewall, VPN, anti-virus, intrusion detection and prevention), Web services, virtualized storage, load balancing, server offload, and intelligent routing and switching, in which content awareness and application intelligence are becoming more important to increasing enterprise productivity.

There are a variety of ways to use the threads of the new processor to increase execution efficiency and accommodate the specific needs of networking-centric applications. In this environment, when one hardware thread is waiting on a memory request, the next thread in the core can immediately begin processing additional data, mitigating latencies, and dramatically improving overall throughput. The threads also can be used in multiple combinations and to replace typical discrete coprocessor functions. And in more compute-intensive applications, a single core can be programmed to dedicate all processing resources and available clock cycles to a single, focused task.

A single-core, single-thread approach runs all tasks in a serial fashion. New processing cannot begin until the previous process is complete. This results in significant memory latency delay accumulation as shown in Figure 1, courtesy of RMI.

The multithread approach takes advantage of, and more fully exploits, the packet-level parallelism commonly found in today's converged computing and networking applications. This workload efficiency improvement is shown in Figure 1. Note in this four-way threaded example, a packet is operated on every fourth cycle. In the clock cycle immediately following that used for packet 1 (yellow), thread 2 can begin operating on packet 2 (blue). Likewise, thread 3 (green) can immediately follow thread 2, and thread 4 (red) can immediately follow thread 3. The process continues among the four separate threads and corresponding packets. The benefit becomes most apparent in the event of a cache miss, during which time the processor can apply useful work to other independent packets while other threads wait for memory. The memory latencies are effectively mitigated and the workload efficiency is highly optimized.

**Development tools to support multicore and multithread features**

Multiple cores and multiple threads in the chip, and combining multiple functions into a single chip design, make for a very powerful and cost-effective processor. However, the chips also bring a whole new level of complexity to product design, debugging, programming, and test.

To overcome that complexity, implementing the new technology into designers' products, a key element of introducing new chips, must be made easier. Code debugging, programming, and test must be considered early in the cycle of new chip design. This part
of chip development is especially important with the proliferation in the number of available chipsets. Chipmakers need to have tools ready and available to support development when people jump on the bandwagon of a new chipset and production volumes explode.

### Debugging multicore, multithread processors

The design engineer faces a new challenge when implementing multicore, multithread chips—to separate and track all of the simultaneous processing occurring in the processor. The JTAG emulator software provided by Corelis for the XLR gives engineers source-level debugging for multiple processors that partitions the individual processor hardware and software resources into a multiwindowed environment. Each thread appears as a separate (virtual) CPU. Engineers can view and modify registers and memory, set software breakpoints, and step through code for every thread and every core independent of the other threads and cores. The XLR processor has an eight-bank Level-2 cache that can process up to eight simultaneous requests from any of the cores and provide a separate hardware instruction and hardware data breakpoint for each of the threads.

Seeing each thread separately, as depicted in Figure 2, provides a view into the multiple ways these processors operate on each packet. For example, threads can request security operations using a new high-speed communications ring architecture within the chip. The engineer can monitor this dedicated security thread at the same time as other processing functions to verify that information will remain secure and identify any packets out of sequence.

The Corelis emulator paired with the XLR processor offers an extensive macro and scripting capability, and can interpret command files written in a structured C-like language, providing an excellent tool for board bring-up, driver and firmware development, and software application debugging of these new processors.

The benefits of quicker development cycles, faster time to market, and flexibility in reacting to the market in terms of new features or technology trends, is obvious. By partitioning the code so that each task is treated as a separate virtual CPU, design engineers can isolate and redesign the code for only the virtual CPU that will handle that new feature or process. In viewing a separate CPU, the engineer gains a significant performance improvement in debugging source code over the traditional method of using thread-dependent breakpoints and kernel-awareness modules.

### Continuing tool development

JTAG emulator functions can be expanded into boundary-scan testing and programming by using the same JTAG controller that the emulator is using (saving on hardware) and adding additional software for boundary-scan testing and in-system programming such as the Corelis ScanExpress family of boundary-scan test tools.

Tracking the sometimes simultaneous transactions on the unique communication rings within each XLR chip poses another challenge. These are extremely high-speed (processor core clock frequency) communication rings providing pointers to the packet data as it is sent between processors, I/O, and so on. A boundary-scan chain cannot track these transactions at the 1.5 GHz processor clock speed. Solving this problem involves implementing an internal trace in the silicon and accessing it with the JTAG interface, which has seen similar successful implementations for chips developed by several CPU vendors.

### Summary

Multicore, multithread intelligent networking processors can provide a key to enable the next generation of information infrastructure devices. The development environment around these chips has moved up to a new level of complexity, but the reasons for using the new architecture are so compelling that multithread processors are sure to grow quickly in application. Fortunately, a new set of debugging, programming, and testing tools closely integrated with the architecture are evolving right alongside these chips. The combination is enabling the flexibility to feed a fast-changing market and shorter development cycles for product designers and manufacturers.

**Bob Burrill** is CodeRunner product manager at Corelis, Inc., with more than 30 years of experience in the hardware, software, and system design of embedded systems. Five years ago, he moved into the development of software development tools after having used a variety of them for many years. Bob has a BSEE from the University of Washington and an MSE from Arizona State University.

For more information, contact Bob at:

Corelis, Inc.
12607 Hidden Creek Way
Cerritos, CA 90703-2146
Tel: 562-926-6727 ext. 115
E-mail: bob.burrill@corelis.com
Website: www.corelis.com

MIPS64 is a trademark of MIPS Technologies, Inc., in the United States and other countries. Other company product and service names may be the trademarks or service marks of others.