

# Low Overhead Interrupt Handling with SMT Threads

Greg Gibeling                      Andrew Schultz

gdgib@berkeley.edu, alschult@cs.berkeley.edu  
*University of California, Berkeley*

## 1 Annotated Bibliography

[1] describes and evaluates the performance impact of tighter integration of the network interface with the CPU core. The paper provides justification for putting the NI on the core, and also evaluates several placement options and potential optimizations such as header splitting.

[2] introduces the M5 simulator as a suitable simulation environment for network oriented simulation. M5 provides a full-system simulation of multiple nodes connected by an arbitrary network topology. Each node can be simulated with varying degrees of detail, all the way down to a detailed, out-of-order core, event driven memory system, and realistic I/O system.

[3] discusses and introduces modifications to the Click modular router which allow Click to run on multiple processors. To evaluate Click with a new interrupt architecture requires the SMP-enabled version of Click.

[4] introduces the Click modular router. Click is a software based router that is composed of flexible software modules. Click is designed to run on commodity hardware and in commodity operating systems. Click provides a motivation for our interrupt handling work as Click typifies the overhead problems with traditional OS interrupts. Click removes the interrupt handling completely, opting instead for a polling approach.

[5] essentially a position paper on why the network interface should be more tightly coupled to the processor and become more of a first class member of the architecture.

[6] discusses and evaluates several code transformation methods which improve cache performance of SMT threads. While not directly related to our proposal, this provides some insight into the cache issues that come up when using SMT threads which is important for our cache pinning idea.

[7] this paper is most closely related to what we are proposing. The paper provides an architecture which uses SMT threads to allow a NIC to schedule its handling code directly. The approach is slightly different from ours, and they don't look into the idea of cache pinning, but it is a good starting point to work from.

[8] describes tools and methodologies for profiling I/O interrupts. The paper provides a good source of information on the general properties, characteristics, and methodology for measuring the performance of

I/O interrupts.

[9] describes the EMP network interface which implements true zero-copy networking. All of the network processing is done on the NIC which allows it to completely bypass the OS code.

[10] introduces the idea of an “application data unit” as a more desirable unit of transfer for a networked application. The basic idea is that the application has a better sense of the proper unit of data, and forcing applications to accept/reject packets sized by the underlying network protocol is inefficient. Additionally, U-Net provides the idea of zero kernel copies by advocating a smarter NIC that can talk to the MMU and copy data directly into user buffers.

[11] provides an extension to U-Net which allows the NIC to transfer data directly into any part of the user address space. By coupling the “smarter NIC” with the MMU, it is possible to fix some of the shortcomings of the previous U-Net paper.

[12] discusses using SMT threads to reduce context switching overhead for handling exceptions. The paper is primarily concerned with synchronous interrupts (more specifically, the paper addresses TLB misses). They provide the basic justification for using SMT threads to reduce the context switch overhead in interrupt handling.

## References

- [1] Nathan L. Binkert, Ronald G. Dreslinski, Erik G. Hallnor, Lisa R. Hsu, Steven E. Raasch, Andrew L. Schultz, and Steven K. Reinhardt. The performance potential of an integrated network interface. In *Advanced Networking and Communications Hardware Workshop*, June 2004.
- [2] Nathan L. Binkert, Erik G. Hallnor, and Steven K. Reinhardt. Network-oriented full-system simulation using M5. February 2003.
- [3] Benjie Chen and Robert Morris. Flexible control of parallelism in a multiprocessor pc router. In Yoonho Park, editor, *USENIX Annual Technical Conference, General Track*, pages 333–346. USENIX, 2001.
- [4] Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. The click modular router. *ACM Trans. Comput. Syst.*, 18(3):263–297, 2000.
- [5] Shubhendu S. Mukherjee and Mark D. Hill. Making network interfaces less peripheral. *Computer*, 31(10):70–76, 1998.
- [6] Dimitrios S. Nikolopoulos. Code and data transformations for improving shared cache performance on smt processors. In Alexander V. Veidenbaum, Kazuki Joe, Hideharu Amano, and Hideo Aiso, editors, *ISHPC*, volume 2858 of *Lecture Notes in Computer Science*, pages 54–69. Springer, 2003.
- [7] Mike Parker, Al Davis, and Wilson Hsieh. Integrating user-level networks with smt.
- [8] Lambert Schaelicke, Al Davis, and Sally A. McKee. Profiling i/o interrupts in modern architectures. In *MASCOTS '00: Proceedings of the 8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, page 115, Washington, DC, USA, 2000. IEEE Computer Society.
- [9] Piyush Shivam, Pete Wyckoff, and Dhabaleswar K. Panda. Emp: zero-copy os-bypass nic-driven gigabit ethernet message passing. In *SC*, page 57, 2001.

- [10] Thorsten von Eicken, Anindya Basu, Vineet Buch, and Werner Vogels. U-net: A user-level network interface for parallel and distributed computing. In *SOSP*, pages 40–53, 1995.
- [11] Matt Welsh, Anindya Basu, and Thorsten von Eicken. Incorporating memory management into user-level network interfaces. Technical report, Ithaca, NY, USA, 1997.
- [12] Craig B. Zilles, Joel S. Emer, and Gurindar S. Sohi. The use of multithreading for exception handling. In *MICRO*, pages 219–229, 1999.