Care and Feeding of High-Performance Processors with Reconfigurable Memory Systems

Frederic T. Chong, Mark Oskin, Timothy Sherwood, and Justin Hensley

Department of Computer Science
University of California at Davis

August 1997

Microprocessor performance continues to follow phenomenal growth curves which drive the computing industry. Unfortunately, memory systems are falling behind when "feeding" data to these processors. Processor-centric optimizations to bridging this *processor-memory gap* [WM95] include prefetching, speculation, out-of-order execution, and multithreading. Unfortunately, many of these approaches can lead to memory-bandwidth problems [BGK96]. We propose a model of computation which *partitions* applications between processor and an *intelligent* memory system.

Specifically, our goal is to keep processors running at peak speeds by off-loading data-manipulation to reconfigurable logic strategically placed in the memory system. For example, a sparse-matrix multiply can be "filtered" by reconfigurable logic such that a processor receives a dense stream of data to multiply and accumulate.

Reconfigurable logic, such as Field-Programmable Gate Arrays (FPGAs), can be re-programmed at run-time in 100s of milliseconds to speed up a specific application [Kat94]. FPGAs perform well for fine-grained data manipulation and specialized arithmetic. Conventional microprocessors, on the other hand, are considerably more efficient for general arithmetic, especially floating point. The key is to partition our computations to exploit the strengths of both technologies while minimizing the data flowing between them.

Our research explores this partitioning on a range of uniprocessor and multiprocessor applications, including: a computationally-intensive, "super-optimizing" compiler, filters for image and video editing, internet routing for a quality-of-service protocol, internet intrusion detection, cache coherency for software distributed shared memory, and protein sequence matching.

Our preliminary results indicate that our compiler application would run 130 times faster on a 166 MHz Ultrasparc v9 workstation with a reconfigurable memory system than on a equivalent workstation with a conventional memory system. The reconfigurable logic required in each memory chip would cover approximately 10% of a 256-Mbit DRAM.

Many issues, however, remain to be resolved. For example, with a reconfiguration time of 100s of milliseconds, how will context-switching be handled in the reconfigurable memory system? Application-specific circuits need to be quickly disabled if an interrupt handler expects the conventional memory system. Additionally, the operating system may need to manage several application-specific circuits, much as hardware queues are managed in network interfaces. Our goal is to explore such issues and demonstrate the viability of reconfigurable memory in real systems.

References

[BGK96] D. Burger, J. Goodman, and A. Kagi. Quantifying memory bandwidth limitations in future microprocessors. In *International Symposium on Computer Architecture*, Philadelphia, Pennsylvania, May 1996. ACM.

[Kat94] R Katz. *Contemporary Logic Design*. Benjamin/Cummings Publishing Company, Redwood City, California, 1994.

[WM95] W. Wulf and S. McKee. Hitting the memory wall: Implications of the obvious. *Computer Architecture News*, 23(1), March 1995.

Fred Chong is supported by the Altera University Program, a UC Davis Junior Faculty Research Fellowship, and a UC Davis Junior Faculty Research Grant. Tim Sherwood and Justin Hensley are supported by an NSF REU grant awarded to Matt Farrens.

For more information, contact **chong@cs.ucdavis.edu** or visit the Reconfigurable Architectures at Davis (RAD) homepage at **http://arch.cs.ucdavis.edu/RAD**.