Processor-In-Memory System Simulator

Andrew Huang

Artificial Intelligence Laboratory
Massachusetts Institue Of Technology
Cambridge, Massachusetts 02139

http://www.ai.mit.edu

The Problem: The extent of previous work on processor in memory systems using merged DRAM-logic process technology [1, 2, 3, 4] indicates that the technology has great promise. Most of the results to date, however, have been architectural studies with simulation results or proposed architectures. Relatively few designs make it to final silicon due to the prohibitively high Non-Recurring Engineering (NRE) costs of this new process technology. Of the designs that do make it to silicon, many have found that a merged DRAM-logic process is fickle: subtle noise or leakage problems can lead to a non-working DRAM array. This difficulty in realizing PIM architectures in silicon has left the research community with a lack of real platforms for compiler and application development, and thus there is much uncertainty about our ability to efficiently utilize these typically fine-grained parallel processor systems.

Motivation: Advances in FPGA technology have helped close the gap between RTL- and HDL- based simulation and final silicon performance levels. The density of FPGAs has also become sufficient to implement a reasonably-sized microarchitecture on a single device. However, FPGAs alone are not sufficient to simulate a PIM system despite the availability of small banks of high-bandwidth memory macros integrated into the FPGA arrays. These banks of memory are simply too small and sparsely populated to do the job.

Fortunately, FPGA vendors such as Xilinx and Altera have also been aggressively developing the configurable pad rings of their devices as well. Modern FPGAs can drive signals at rates exceeding 622 MBits/s per pin, with well over a thousand I/Os available per FPGA. This, combined with the explosion of high-bandwidth SSRAM standards, provides an opportunity to build an adequate but low-cost PIM system simulator.

Previous Work: FPGA-based simulation systems that contain some mix of SSRAM and host connectivity are ubiquitous today. Annapolis Microsystems, Aptix, Virtual Computer Corporation, and Microtech, to name a few, all provide viable, high performance simulation solutions. However, none of them provide memory systems with bandwidths approaching those achievable in a PIM system, and thus they remain unsuitable for accurately simulating PIM architectures in a manner with sufficient performance to do native code development and performance benchmarking. In addition, almost all of them integrate a relatively low-performance, long-latency interface to a host, such as SCSI or PCI, for host and peer-to-peer interactions, also limiting their usefulness in simulating a large, multi-node system.

Approach: We start by first considering the DRAM performance of a typical merged DRAM-logic process: a typical macro in a mid-range merged DRAM-logic process (0.25u) is about 512 bits wide with a 166 MHz cycle time and roughly 10-20ns access latency [6]. This is an equivalent of 10.6 GBytes/s for a single macro. Thus, to simulate a single processor-memory macro as an SSRAM-FPGA pair, one would require a system with roughly equivalent bandwidth.

A modern FPGA such as the Xilinx Virtex-II, coupled with DDR (double data rate) SSRAM from a manufacturer such as Micron, gives us a system with a chunk of two or three million gates of logic and an SSRAM interface that is 512 bits wide that runs in excess of 300 MHz. This yields an overall bandwidth of 19.2 GBytes/s, at an access latency of about 10-20 ns. One can see that the integrated PIM process and the discrete FPGA-SSRAM solution are comparable performance solutions.

In addition, a modern FPGA has sufficient performance to directly simulate a PC-100 DIMM-style SDRAM interface: this is perhaps the highest-bandwidth, lowest-latency socket available on a modern PC today, short of a proprietary processor or DRDRAM socket. While new versions of the PCI bus, such as PCI-X and PCI-64 can provide comparable bandwidths, their multi-drop, bus-arbitrated architecture can lead to serious latency concerns. By incorporating a high-bandwidth, low-latency PC-100 SDRAM interface into the PIM simulator, one can quickly fill and empty the memory arrays, simulating the effects of “the rest of the system” more accurately than a PCI bus can.

The first-generation incarnation of our hardware simulator (the “Moore Board”) uses only single-data-rate SSRAM parts with half the bus width, due to parts availability and cost restrictions, but it does incorporate the high performance PC-100 SDRAM interface and high-performance FPGAs. As a result, our proof-of-concept platform contains two 1-million gate FPGAs, a 288-bit wide SSRAM interface, and a PC-100 interface. One FPGA is coupled with the SSRAM and the other is used to drive the PC-100 interface. The interface between the FPGAs is a high-performance LVDS solution that provides sufficient bandwidth to emulate, for example, a network connection between processor-memory pairs. While the performance of the SSRAM memory interface is slightly slower than that found in a potential target PIM process, it is still high enough to provide an interesting simulation target.

Impact: We are currently developing prototype processor architectures that run on the Moore Board. The short-term goal is to prove that the Moore Board is a viable simulation platform, and also to prove that our processor architecture ideas can effectively leverage the extremely wide word-accesses that we get using merged DRAM-logic style memory macros. We are also taking steps to ensure that the codebase developed for the Moore Board is sufficiently portable so that future revisions of the board and eventually the final silicon for the project can leverage the work being done today.

A collateral impact of doing this design is the realization that even higher performance could be gained by designing custom DRAM macros instead of using the drop-in DRAM macros provided by most merged logic-DRAM process providers. We estimate that a custom DRAM macro optimized for access bandwidth and latency can provide a 4 to 8 times performance boost over existing DRAM macros for about a roughly 2-3x area penalty. In addition, one can integrate small amounts of custom logic into the sense amps of the DRAM macro that can provide a huge amount of leverage for key computations such as associative lookups and searches through memory. Unfortunately, this design approach once again brings us into the realm of risky and expensive semiconductor process-dependent designs that are difficult to emulate with our FPGA-SSRAM system.

Future Work: A long-term goal of the work involving the Moore Board is to run large-scale multiprocessing benchmarks on a system consisting of several networked boards similar to the Moore Board. This will help us evaluate the impact of network performance, memory management and programming models on overall system performance and usability.

The Moore Board can also be adapted to perform a number of other tasks. In order to assist the migration of other work into our platform, it is a goal of ours to make the core primitives–the PC-100 interface and the SSRAM interfaces–simple and flexible, yet easy to use as a drop-in synthesizeable core.

Finally, future revisions of this board will be made as FPGA and SSRAM technologies progress. One of the advantages of committing to a discrete PWB-based FPGA/conventional SSRAM based technology mix is the relatively low cost of keeping up with the furious pace of Moore’s law. This is in stark contrast to designs that are committed to a particular IC process, where every 18 months a factor of two is lost in perceived performance. To make matters worse, it is very difficult to port designs between merged logic-DRAM processes, even within the same foundry.

Research Support: Support for this research was provided by the Air Force Research Laboratory, agreement number F30602-98-1-0172, “Active Database Technology”.

References:

[1] F. T. Chong, M. Oskin, T. Sherwood. Active Pages: A Computation Model for Intelligent Memory *ISCA 1998, Barcelona, Spain*

[2] C. E. Kozyrakis, S. Perissakis, D. Patterson, T. Anderson, K. Asanovic, N. Cardwell, R. Fromm, J. Golbus, B. Grbistad, K. Keeton, R. Thomas, N. Treuhaft, K. Yelick. Scalable Processors in the Billion-Transistor Era: IRAM *IEEE Computer*, September, 1997, pp. 75-78.

[3] N. Margolus. An Embedded DRAM Architecture for Large-Scale Spatial-Lattice Computations *ISCA 2000, Vancouver, Canada*. Proceedings, pp. 149-160.

[4] S. Perissakis, Y. Joo, J. Ahn, A. DeHon, J. Wawrzynek. Embedded DRAM for a Reconfigurable Array. *Proceedings of the 1999 Symposium on VLSI Circuits*, June 1999.

[6] TC240DC/DE Embedded DRAM SLI ASIC 0.25um dRAMASIC Product Brief http://www.toshiba.com/taec/components/ProdBrief/TC240.pdf