

Abstract Machines for OS/R Research

Ron Sass* Andrew G. Schmidt Matthew French
UNC-Charlotte USC/ISI USC/ISI
rsass@uncc.edu aschmidt@isi.edu mfrench@isi.edu

October 4, 2012

Description

A fundamental problem facing researchers studying the system software for extreme scale computing is that much of the target hardware technology and architecture is unsettled. In fact, some of the technology may not even be available for years. This poses a very significant challenge because historically operating and runtime system research has been very empirical. Some questions can only be answered through parameter studies on running prototypes or by observing systemic effects of large ensembles of prototypes. But clearly, waiting for the hardware to appear before starting any Operating System/Runtime (OS/R) software investigation is unacceptable.

This paper advocates for a research regime that would conduct experiments by implementing OS/R research prototypes on collections of Abstract Machines (AMs) which, in turn, are realized on a cluster of FPGA nodes. The AMs would be assembled from range of architectural components representative of future Exascale technologies currently under investigation by the hardware community.

There are significant advantages to this approach.

- Emulation is fast and scalable, decreasing the time-to-discovery.
- As the future of extreme scale computing unfolds, OS/R researchers can narrow the range of candidate AMs and refine the relevant AMs.
- By its very nature, this approach has a strong potential for hardware/software co-design of the system software.
- In this regime, AMs can be automatically instrumented (and reinstrumented) to collect

performance information at run-time. It is ripe with opportunities to automate and aggregate the collection of such information.

- This approach has the ability to inject environmental effects at multiple levels — from minute bit flips to stopping whole cores. It has the ability to inject noise (without necessarily causing faults) into smaller scale ensembles to recreate systemic effects found in very large systems.

Many of the weaknesses of this approach can be mitigated. Certainly, the development of AMs on FPGAs is not trivial. However, with standard interfaces and the use of components, the number of AMs can grow by “mixing and matching” components to create new architectures. Likewise, a significant body of existing work can be leveraged to form an emerging library of AM components. Another legitimate concern is the emulation’s significant of the loss of fidelity. (We are referring to the loss of fidelity in terms of the component’s ability to model the cycles to completion, not its functional correctness.) Positive results under this regime will require additional, follow-on validation studies.

To make this approach more concrete, consider three examples. RedSharc (REconfigurable Data-Stream Hardware software ARCHitecture) is an example of an abstract machine that was directly realized on an FPGA. This system implemented a DARPA(PCA)-developed Streaming Virtual Machine (SVM).¹ It is an excellent example of an unconventional architecture executing at a substantial fraction of its theoretical speed. *However, this would not have been possible had the AM been developed for another target, say custom CMOS or*

¹RedSharc was introduced in [Schmidt et al., 2010] and discussed in this context at a recent DOE workshop [Booth et al., 2011]

*Contact author.

ASIC. This is because the HDL for these other targets would have been grossly inefficient when synthesized for an FPGA fabric.

Likewise, it is extremely simple to implement a multicore AM of soft processors that run Linux [Sass and Schmidt, 2010] and open source MPI implementations. With the right tools [Rajasekhar et al., 2012, 2008] research assistants can focus on the OS/R aspects and not the development process.

Finally, to give a sense of what is possible, a system currently under development will implement an AM of a multi-core chip with differentiated processor cores (similar to the Fresh Breeze project [Dennis et al., 2011]). Although we have not done this, it would be straight forward to directly implement an AM that emulates Hybrid Memory Core (ie. 3D memory sitting upon a collection of cores) system by mapping individual cores to physical memory channels on the FPGA.

Assessment

Below we address the call for paper's specific rubric questions, one per paragraph.

The ability to effectively study OS/R issues by direct observation is the primary challenge addressed. OS/R experiments will execute on hardware, albeit much slower than the ultimate target technology. Other approaches (such as software simulation) are intractable due to the sheer number of interacting components required to make meaningful systemic observations.

Although using FPGAs for emulation is not novel, applying it to OS/R research is. For this reason, we argue that the **core idea is mature and well-established.**

If not for the scale of the problem (number of components, threads, hierarchies involved), other approaches would be suitable. For other programs that can use these simpler approaches (embedded or RTOS research, for example), the proposed approach is overkill. **This makes it less likely to be funded by other research programs.**

Unlike using FPGAs for CMOS chip design verification or cycle-accurate simulation of processors, the proposed AMs are intended to be realized on FPGAs. This is a subtle but critical difference. It sacrifices some component fidelity (cycle accuracy not functional correctness) to increase component density yielding much larger systems. **The novelty of this approach lies in its ability to observe the ef-**

fects of OS/R design decisions on systemic performance.

That said, as the broader Exascale research agenda advances, one could introduce cycle-accurate components with value to the architecture community. This is especially true **if a hardware/software co-design experiment that emerged from an OS/R regime resulted in data that informed the architecture community.**

Presently there are small systems (64 FPGA nodes) that demonstrate the ability to implement AMs. The system software required to manage many node systems has been developed. Advanced research demonstrates the feasibility of the automated instrumentation and performance monitoring. Many examples of hardware/software co-design of the OS/R exist. (See bibliography.) However, **to be effective,**

- ✓ **much larger collections of FPGAs are needed**
- ✓ **more automation of experiments**
- ✓ **additional AM innovation**

Related Work

Industry has, for decades, used FPGAs for Design Verification (DV) to vet custom chips for functionality before developing masks. More recently, university-driven research has begun using FPGAs to replace cycle-accurate software simulations with cycle-accurate hardware emulations [RAMP, 2012]. Both of these approaches focus narrowly on (single core) node performance which was paramount in the past but, unfortunately, does not capture systemic effects that emerge in extreme scale systems.

Recent endeavors have begun to use multi-level simulation and emulation techniques that incorporate FPGAs and software simulation of specific components [Rodrigues, 2012, Shalf, 2012, Shalf et al., 2011]. This is likely to be a huge boon for architecture research but its focus is application-driven.

Fortunately, there has been significant interest in using FPGAs to study system resilience [Sass et al., 2009, Schmidt et al., 2011, Mendon et al., 2012], OS/R hardware/software co-design [Gao et al., 2010, Mendon et al., 2009], and future system (on/off-chip) interconnects [Schmidt et al., 2010, Kritikos et al., 2011, Schmidt et al., 2012], and automated performance monitoring [Huang et al., 2010].

References

- Stephen Booth, Dan Campbell, Andrew Chien, , Richard Lethin, Lenore Mullin, Arun Rodrigues, Ron Sass, John Shalf, Mark Snir, and Tom Sterling. DOE 2011 workshop on architectures II: Exascale, and beyond: Configuring, reasoning, and scaling. Technical report, Department of Energy, Office of Advanced Scientific Computing (ASCR), August 2011.
- Jack B. Dennis, Guang R. Gao, and Xiao X. Meng. Experiments with the fresh breeze tree-based memory model. *Comput. Sci.*, 26(3-4):325–337, June 2011. ISSN 1865-2034. doi: 10.1007/s00450-011-0165-1.
- Shanyuan Gao, Andrew G. Schmidt, and Ron Sass. Impact of reconfigurable hardware on accelerating MPI.Reduce. In *2010 International Conference on Field-Programmable Technology (FPT'10)*, pages 29–36, December 2010. doi: <http://dx.doi.org/10.1109/FPT.2010.5681537>.
- Bin Huang, Andrew G. Schmidt, Ashwin A. Mendon, and Ron Sass. Investigating resilient high performance reconfigurable computing with minimally-invasive system monitoring. In *Fourth International Workshop on High-Performance Reconfigurable Computing Technology and Applications (HPRCTA'10) [Held in conjunction with SC10]*, pages 1–8, November 2010. **Won OpenFPGA ‘Best Paper’ award.**
- William V. Kritikos, Yamuna Rajasekhar, Andrew G. Schmidt, and Ron Sass. A radix tree router for scalable fpga networks. In *FPL '11: Proceedings of the 19th International Conference on Field-Programmable Logic and Applications*, pages 76–81, Crete, Greece, August 2011. doi: <http://dx.doi.org/10.1109/FPL.2011.24>.
- Ashwin A. Mendon, Andrew G. Schmidt, and Ron Sass. A hardware filesystem implementation with multi-disk support. *International Journal of Reconfigurable Computing*, page 13 pages, 2009. doi: doi:10.1155/IJRC.
- Ashwin A. Mendon, Justin L. Tripp, Zachary K. Baker, and Ron Sass. Design and implementation of a hardware checkpoint/restart core. In *Fault-Tolerance at Extreme Scale (FTXS) workshop held at DSN2012: The 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, Boston, MA, June 2012.
- Yamuna Rajasekhar, William V. Kritikos, Andrew G. Schmidt, and Ron Sass. Teaching FPGA system design via a remote laboratory facility. In *18th Annual Conference on Field Programmable Logic and Applications (FPL)*, pages 687–690, Heidelberg, Germany, September 2008.
- Yamuna Rajasekhar, Rahul R. Sharma, and Ron Sass. An extensible and portable tool suite for managing multi-node fpga systems. pages 117–120, Los Alamitos, CA, USA, 2012. IEEE Computer Society. ISBN 978-0-7695-4699-5. doi: <http://doi.ieeecomputersociety.org/10.1109/FCCM.2012.29>.
- RAMP. RAMP — research accelerator for multiple processors, 2012. URL: <http://ramp.eecs.berkeley.edu/index.php?/index>.
- Arun Rodrigues. Sst: The structural simulation toolkit, 2012. URL: <http://sst.sandia.gov/>.
- Ron Sass and Andrew G. Schmidt. *Embedded Systems Design with Platform FPGAs: Principles & Practices of System Design*. Morgan-Kaufmann, an imprint of Elsevier, San Francisco, CA, USA, August 2010.
- Ron Sass, Rahul Sharma, and Nathan DeBardleben. Towards a hardware fault-injection testbed to support reproducible resiliency experiments. In *Resiliency'09 Workshop in Proceedings International ACM Symposium on High Performance Distributed Computing*, Munich, Germany, June 2009.
- Andrew G. Schmidt, William V. Kritikos, Ron Sass, Erik K. Anderson, and Matthew French. Merging programming models and on-chip networks to meet the programmable and performance needs of multi-core systems on a programmable chip. In *2010 IEEE International Conference on Reconfigurable Computing and FPGA's (ReConFig 2010)*, pages 334–339, December 2010. doi: <http://dx.doi.org/10.1109/ReConFig.2010.55>. **Best paper award.**
- Andrew G. Schmidt, Bin Huang, and Ron Sass. Checkpoint/restart and beyond: Resilient high performance computing with fpgas. In *Proceedings of 19th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM 2011)*, pages 162–169, Salt Lake City, UT, USA, April 2011. IEEE Computer Society. doi: <http://dx.doi.org/10.1109/FCCM.2011.22>.
- Andrew G. Schmidt, William V. Kritikos, Shanyuan Gao, and Ron Sass. An evaluation of an integrated on-chip/off-chip network for high performance reconfigurable computing. *International Journal of Reconfigurable Computing*, 2012(Article ID 564704):15 pages, 2012. doi: doi:10.1155/2012/564704.
- John Shalf. CoDEx: Co-design for exascale, 2012. URL: <https://sites.google.com/a/lbl.gov/codex/>.
- John Shalf, Dan Quinlan, and Curtis Janssen. Rethinking hardware-software codesign for exascale systems. *Computer*, 44(11):22–30, November 2011. doi: 10.1109/MC.2011.300.