Microarchitecture Optimization for Embedded Systems

David V. Schuehler, Benjamin C. Brodie, Roger D. Chamberlain, Ron K. Cytron, Scott J. Friedman, Jason Fritts, Phillip Jones, Praveen Krishnamurthy, John W. Lockwood, Shobana Padmanabhan, and Huakai Zhang

David V. Schuehler, Benjamin C. Brodie, Roger D. Chamberlain, Ron K. Cytron, Scott J. Friedman, Jason Fritts, Phillip Jones, Praveen Krishnamurthy, John W. Lockwood, Shobana Padmanabhan, and Huakai Zhang, "Microarchitecture Optimization for Embedded Systems," *8th High Performance Embedded Computing Workshop*, September 2004.

Dept. of Computer Science and Engineering
Washington University
Campus Box 1045
One Brookings Dr.
St. Louis, MO 63130-4899

Microarchitecture Optimization for Embedded Systems [*]

David V. Schuehler, Benjamin C. Brodie, Roger D. Chamberlain, Ron K. Cytron, Scott J. Friedman, Jason Fritts, Phillip Jones, Praveen Krishnamurthy, John W. Lockwood, Shobana Padmanabhan, and Huakai Zhang
Department of Computer Science and Engineering
Washington University, St. Louis, Missouri

Abstract

Embedded applications are held to a higher standard than desktop applications along a number of dimensions. Not only must the functionality of the application be correct, it often must meet strict time constraints and function with restrictive resource limitations (e.g., memory size, power, weight). Conversely, the hardware systems used to execute embedded applications are often dedicated to that particular application, alleviating the need to be as general purpose as desktop systems. As a result, there is significant ongoing interest in the ability to build custom hardware platforms for which the design of the hardware is closely matched to the needs of the application. In addition, many applications can be characterized into distinct "phases" during which the application characteristics are often significantly different. A hardware platform that can closely match the needs of the application, even as the application characteristics change, is the goal of our research group.

We are investigating the concept of a *liquid architecture*, in which the physical microarchitecture of a processor has the ability to be altered to adapt to the needs of a particular application, even so far as to be altered during application execution when the needs of the application change [1]. This can be realized in a number of ways: (1) via a soft-core processor deployed on an FPGA [2], and (2) via a configurable processor that is extensible at execution time [3,4]. Candidate architectural alternatives include ISA extensions (e.g., new instructions for frequent operations), cache system alterations (e.g., size, line length, associativity, write-back vs. write-through, specialized cache structures), co-processors, etc. While many architectural alternatives have been proposed in the past, and ideas for new architectural features can readily be proposed both today and into the future, the challenge we currently face is the need to quantitatively evaluate these alternatives with sufficient depth of understanding to decide whether a particular application (or phase of an application) is well matched to a particular candidate architecture. In short, given the ability to be flexible in the architectural choice at runtime, what architecture should be chosen for a given application?

Answering the above question requires a quantitative assessment of the application's performance as well as the resources consumed by the architecture (e.g., area, power). While traditionally answered via discrete-event simulation techniques, simulation is both time-consuming and has severe fidelity/completeness tradeoffs associated with it. For example, it is rare for a cycle-accurate simulation of a processor to model the activity of the operating system as well as the target application. We have built a system that enables the quantitative architectural assessments required to match architecture and application in a reasonable timeframe. In addition, the performance measurements do not impact the actual performance of the system under observation, as is the case with classical software-based performance monitoring techniques.

The liquid architecture system we have constructed is based upon the LEON soft-core processor [2]. The LEON is a SPARC-compatible, synthesizable processor that has a reasonable level of configurability built into the original distribution. For example, the cache size,

[*] This material is based upon work supported by the NSF under grant ITR–0313203.

associativity, and line length are all settable via a menu-based configuration script prior to synthesis. In addition, full VHDL source is provided, enabling significantly greater levels of configurability to those willing to specify architectural alternatives at the HDL level. At the system level, the processor core is connected to memory via the AMBA High-speed Bus (AHB) and to other I/O devices (e.g., console UART) via the AMBA Peripheral Bus (APB) [5].

Figure 1 shows the overall system, as ported to the FPX platform [6]. The FPX supports high-bandwidth (GigE) network connectivity to/from the on-board FPGA (a VirtexE series part from Xilinx). Our locally-developed control packet processor supports not only control of the LEON processor itself but also the ability to independently communicate with other resources in the system (e.g., the off-chip external memory and the on-chip statistics module).

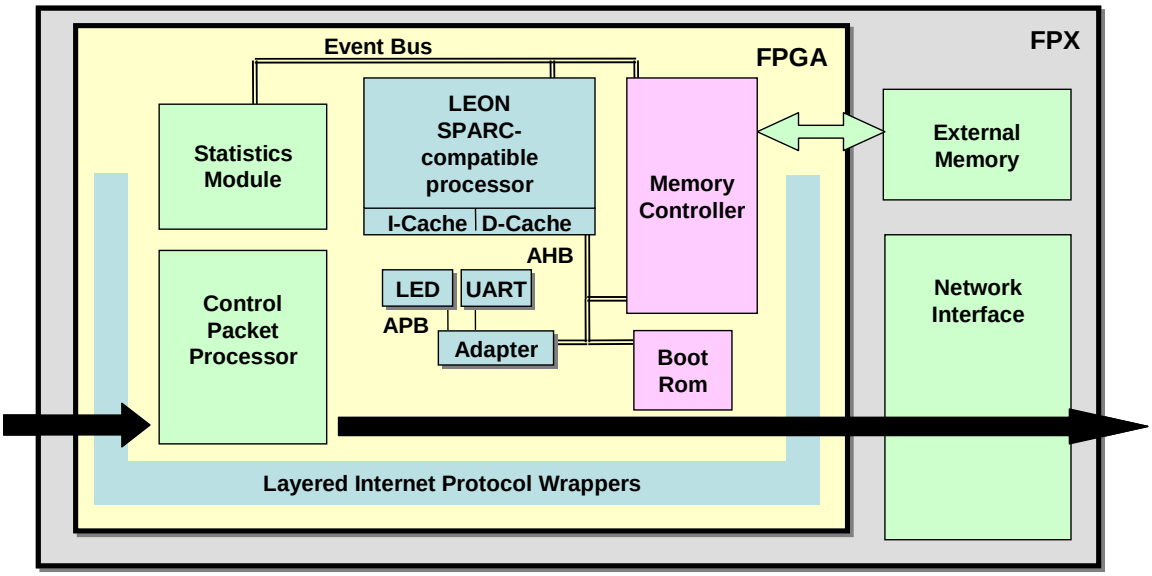

Figure 1: High-level diagram of the liquid architecture system

The statistics module and associated event bus enable detailed performance monitoring of any aspect of the system without perturbing the performance of the system itself. Currently supported statistics include monitoring of cache hit/miss rates as well as cycle-accurate profiling of time spent in user-specified method invocations. The ability to make measurements without perturbing the system under observation is due to the fact that independent hardware logic is used to both make the measurements and accumulate the intermediate results. The user specification of what statistics are to be collected, and over what portion of the application, is made via a web interface that interacts with the statistics module via the control packet processor. The point here is that the system under observation (i.e., the processor, bus, memory, peripherals, etc.) is not included in any way in the performance monitoring task.

To enable the evaluation of the application/architecture pairing in a realistic execution environment, it is necessary to include the impact of the operating system as well as the application itself. To support this need, we have installed the μClinux OS [7] on the liquid architecture system. The μClinux OS was developed for embedded processor platforms that do not include a memory management unit. The boot sequence screen shot is illustrated in Figure 2.

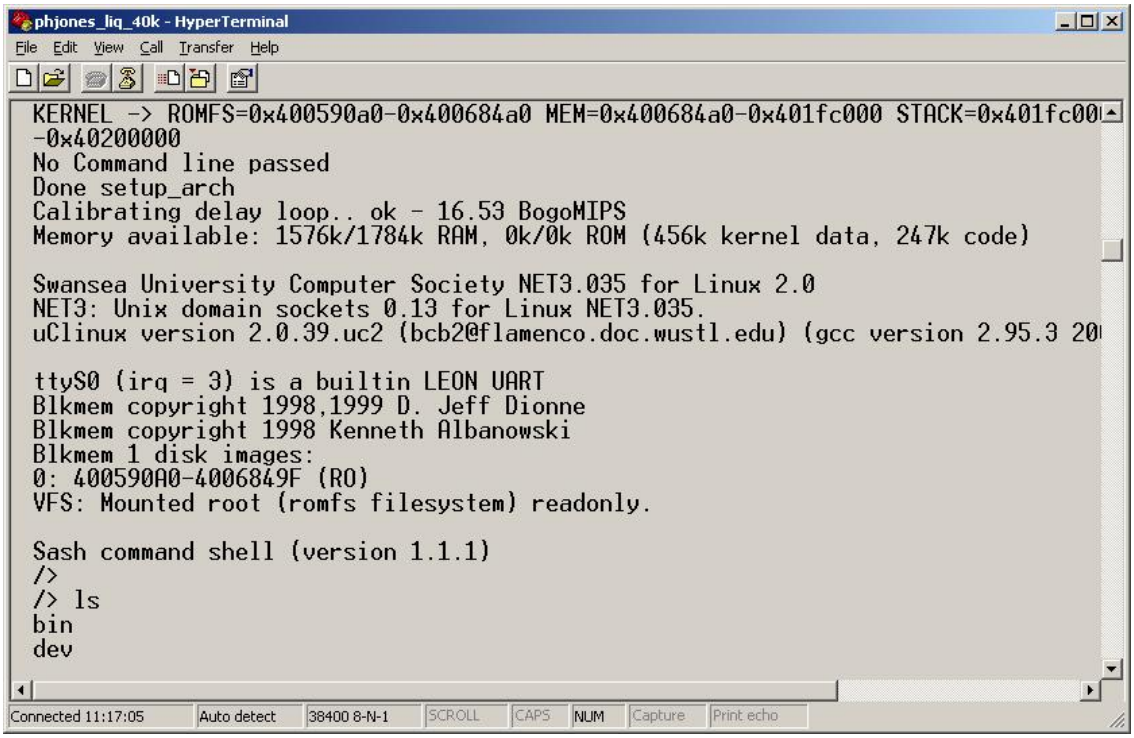

Figure 2: Screen image of the μClinux OS boot sequence

Our current activities include the porting of a number of benchmark applications to the system, including SPECint2000 [8], CommBench [9], and MediaBench [10]. We will present quantitative performance measurements on these applications, illustrating the performance impact of various architectural alternatives.

[1] Phillip Jones, Shobana Padmanabhan, Daniel Rymarz, John Maschmeyer, David V. Schuehler, John W. Lockwood, and Ron K. Cytron. Liquid architecture. In *Proc. of Workshop on Next Generation Software* (at IPDPS), April 2004.

[2] Free Hardware and Software Resources for System on Chip. `http://www.leox.org`

[3] S. Hauck, T. W. Fry, M. M. Hosler, and J. P. Kao. The Chimaera reconfigurable functional unit. In *Proc. IEEE Symp. on FPGAs for Custom Computing Machines*, pages 87–96, 1997.

[4] Stretch, Inc. `http://www.stretchinc.com`

[5] AMBA Specification (Rev 2.0). ARM, Ltd. `http://www.arm.com/products/solutions/AMBA_Spec.html`

[6] J.W. Lockwood, N. Naufel, J.S. Turner, and D.E. Taylor. Reprogrammable Network Packet Processing on the Field Programmable Port Extender (FPX). In *Proc. ACM Int'l Symp. on Field Programmable Gate Arrays*, pages 87–93, February 2001.

[7] μClinux – Embedded Linux/Microcontroller Project. `http://www.uClinux.org`

[8] J.L. Henning. SPEC CPU2000: Measuring CPU Performance in the New Millennium. *IEEE Computer*, **33**(7):28–35, July 2000.

[9] Tilman Wolf and Mark Franklin. CommBench – a telecommunication benchmark for network processors. In *Proc. of IEEE Int'l Symp. on Performance Analysis of Systems and Software*, pages 154–162, April 2000.

[10] Chunho Lee, M. Potkonjak, W.H. Mangione-Smith. MediaBench: a tool for evaluating and synthesizing multimedia and communications systems. In Proc. of IEEE Int'l Symp. on Microarchitecture, pages 330–337, December 1997.