

Tortola: Addressing Tomorrow’s Computing Challenges through Hardware/Software Symbiosis

Kim Hazelwood
University of Virginia and Intel Corporation

Abstract

Until recently, the vast majority of research efforts in optimizing computer systems have targeted a single logical “layer” in isolation: application code, operating systems, virtual machines, microarchitecture, or circuits. However, we are reaching the limits of the solutions than we can provide by targeting a single design layer in isolation. The Tortola project explores a symbiotic relationship between a virtual machine and the host microarchitecture to solve crosscutting concerns in the areas of power, reliability, security, and performance using both hardware and software extensions. We have demonstrated the effectiveness of our approach on the well-known dI/dt problem, where we successfully stabilized the voltage fluctuations of the CPU’s power supply by transforming the source code of the executing application using feedback from hardware.

This paper and accompanying talk will motivate our notion of symbiotic program optimization, discuss various applications, and detail our experiences in solving the dI/dt problem using a holistic approach.

1 Overview

Modern computer system designers must consider many more factors than just raw performance. Thermal output, power consumption, reliability, testing, and security are quickly becoming first-order concerns. Yet, the vast majority of research efforts in optimizing computer systems have targeted a single logical “layer” in isolation: application code, operating systems, virtual machines, microarchitecture, or circuits. There are several reasons to believe we are reaching the limits of the solutions we can provide by targeting a single layer in isolation.

An important class of computing challenges exist that are better suited for more holistic approaches. Many challenges can be solved much more easily using “reactive” techniques, whereby the hardware can detect a

problem, and a virtual machine can use its global knowledge about the executing workload to correct the problem. In fact, this solution has the potential to outperform each of its constituent hardware-only or software-only solutions.

2 A Virtual Interface

In order to explore such solutions, the Tortola project introduces a virtual interface between the application software and the underlying machine architecture. The unique aspect of this interface is that it facilitates communication between the microprocessor and the virtual layer, which allow us to investigate combined hardware-software techniques for solving many of the future computing challenges.

In essence, this solution *virtualizes* the ISA, allowing solutions to be developed which span the hardware-software divide, as shown in Figure 1. As the figure indicates, the virtual machine can use hardware feedback to detect various machine-specific events, such as voltage fluctuations in the power supply, temperature and power problems, as well as performance-related events, such as cache misses or resource contention. The VM can then factor in its knowledge about the executing workload, such as the specific instructions selected, their instruction schedule, and the control-flow graph. Finally, the VM can develop a holistic solution to the problem at hand which accounts for both hardware and software inputs; we call this technique *symbiotic optimization*.

3 Our Approach

As Figure 1 indicated, our solution requires changes to the hardware in order to provide feedback to our virtual layer. Rather than building custom hardware, we began our investigations using simulation. We used SimpleScalar [1] for x86 to simulate our modified hardware. We also incorporated the Watch [2] power extensions to SimpleScalar in order to enable power results in addition to performance results. For our virtual machine

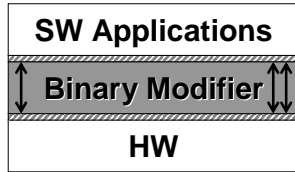


Figure 1. The Tortola architecture including a virtual layer to support HW-SW communication channels.

layer, we used the x86/Linux version of the Pin dynamic instrumentation system [5].

To explore symbiotic optimization, we execute our applications on top of Pin, which is running on top of SimpleScalar. Not surprisingly, a great deal of implementation effort was required in order to get SimpleScalar to the point where it could successfully emulate all of the system calls that Pin requires. The end result is a holistic simulation environment that allows us to explore collaborative solutions to existing and future computing challenges.

4 A Motivating Example

Many system design problems exist that can benefit greatly from the holistic design approach we present, including issues related to power, performance, temperature, reliability, and security. As a motivating example, we present one such problem and symbiotic solution.

A Symbiotic di/dt Solution We have demonstrated the effectiveness of our approach on the well-known di/dt problem. The di/dt problem is a side-effect of modern techniques in low-power processor design. In order to reduce overall power consumption, idle portions of a processor are turned off. If one feature is repeatedly turned on and off, reliability problems can arise.

Hardware-based sensor/actuator mechanisms have been proposed to detect and react to these problematic current variations [4], but they do so at a performance cost to the running application. Our approach keeps these hardware solutions in tact, but simply communicates to the virtual machine in real time when the problem occurs. The virtual machine then modifies and caches the currently executing instructions in an attempt to avoid the problem in the future. The code transformations can be fairly straightforward. In fact, we were able to apply standard compiler optimizations (loop unrolling and software pipelining) to remedy future recurrences of the di/dt problem. Figure 2 shows the result of applying software pipelining to a hand-coded *power virus* described by Joseph et al. [4].

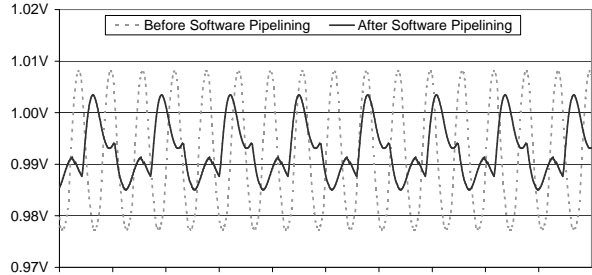


Figure 2. The effect of software pipelining on a voltage fluctuation stressmark.

Our symbiotic solution was therefore able to provide the safety of the hardware-only technique with the performance improvements possible from a software-only technique. More detailed information about this particular di/dt solution is available in our ISLPED paper [3].

5 Acknowledgments

Several others have contributed to this project. David Brooks provided a great deal of brainstorming effort in developing the di/dt component of this project, as well as with the Watch power extensions. Greg Humphreys helped to develop the project name and logo, and also, together with Dan Williams, provided invaluable software development of SimpleScalar/x86 to get it to the point where it could successfully run Pin. We thank Brad Calder for granting us early access to the new SimpleScalar for x86. Finally, Russ Joseph's power virus and power supply extensions to Watch have proven to be extremely useful in this research.

References

- [1] T. Austin, E. Larson, and D. Ernst. SimpleScalar: An infrastructure for computer system modeling. *IEEE Computer*, pages 59–67, February 2002.
- [2] D. Brooks, V. Tiwari, and M. Martonosi. Watch: A framework for architectural-level power analysis and optimizations. In *International Symposium on Computer Architecture (ISCA-27)*, 2000.
- [3] K. Hazelwood and D. Brooks. Eliminating voltage emergencies via microarchitectural voltage control feedback and dynamic optimization. In *International Symposium on Low-Power Electronics and Design*, pages 326–331, Newport Beach, CA, August 2004.
- [4] R. Joseph, D. Brooks, and M. Martonosi. Control techniques to eliminate voltage emergencies in high performance processors. In *High-Performance Computer Architecture (HPCA-9)*, 2003.
- [5] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. J. Reddi, and K. Hazelwood. Pin: Building customized program analysis tools with dynamic instrumentation. In *Programming Language Design and Implementation*, pages 190–200, Chicago, IL, June 2005.