Validating a Modern Microprocessor

Extended Abstract

Bob Bentley

2111 N.E. 25th Avenue, Hillsboro, Oregon 97124, U.S.A.
bob.bentley@intel.com

1 Introduction

The microprocessor presents one of the most challenging design problems known to modern engineering. The number of transistors in each new process generation continues to follow the growth curve outlined by Gordon Moore 40 years ago. Microarchitecture complexity has increased immeasurably since the introduction of out-of-order speculative execution designs in the mid-90s; and subsequent enhancements such as Hyper-Threading (HT) Technology, Extended Memory 64 Technology and ever-deeper pipelining indicate that there are no signs of a slowdown any time soon. Power has become a first-order concern thanks to a 20x increase in operating frequencies in the past decade and leakier transistors at smaller geometries, and the various schemes for managing and reducing power while retaining peak performance have added their own dimensions of complexity.

2 Microprocessor Design

Microprocessor design teams vary widely in size and organizational structure. Within Intel, implementing a new microarchitecture for the IA-32 product family typically requires a peak of more than 500 design engineers across a wide range of disciplines - logic and circuit design, physical design, validation and verification, design automation, etc. - and takes upwards of 2 years from the start of RTL coding to initial tape-out. Many of the same engineers, plus specialists in post-silicon debug, test, product engineering, etc. are needed to get from first silicon to production, which typically takes between 9 and 12 months. The product then has to be ramped into high-volume manufacturing, at a run rate of tens of millions of units per quarter, and sustained until end of life. The overall cycle is significantly longer than the time (roughly 24 months) between successive semiconductor process generations.

3 Microprocessor Validation

Microprocessor validation starts early in the design cycle - these days, even before the start of RTL coding. Validation engineers are involved in microarchitecture definition, helping to prevent architectural bugs and produce a more validatable design. On our most recent design, we have for the first time deployed formal tools and methods - built around Lamport's TLA - during the microarchitecture definition phase.

Most validation starts with the first release of RTL code. For a new microprocessor, code is typically created and released in a number of carefully planned phases, over the course of approximately one year. We rely heavily on Cluster Test Environments (CTEs) to allow us to do microarchitecture validation on logically related subsets of the design, which provides for a much greater degree of controllability than full chip and also decouples each cluster from the others. Even after a full chip model is available, we continue to focus much of our dynamic validation effort and cycles at the cluster level, since controllability in the later pipeline stages of an out-of-order machine will always be a significant issue. During the RTL development phase of the project, we have also started to deploy SAT checking as bug-finding tool.

4 Formal Verification

The Pentium® 4 processor was the first project of its kind at Intel where we deployed Formal Property Verification (FPV) as a mainstream validation technique during CPU development. Hitherto, FPV had only been applied retroactively, as was done for the FP divider of the Pentium® Pro processor. We focused on the areas of the design where we believed that FV could make a significant contribution – in particular, the floating-point execution units and the instruction decode logic. Bugs in these areas had escaped detection on previous designs, so this allowed us to apply FPV to some real problems with real payoff.

A major challenge for the FPV team was to develop the tools and methodology needed to handle a large number of proofs in a highly dynamic environment. The RTL model is constantly changing due to feature additions or modifications, bug fixes, timing-induced changes, etc. By the time we taped out we had over 10,000 proofs in our proof database, each of which had to be maintained and regressed as the RTL changed over the life of the project.

Though our primary emphasis was on proving correctness rather than bug hunting, FPV had found close to 200 logic bugs by the time we taped out. This was not a large number in the overall scheme of things (we found almost 8000 bugs total), but about 20 of them were “high quality” bugs that we do not believe would have been found by any other of our pre-silicon validation activities. Two of these bugs were classic floating-point data space problems:

- The FADD instruction had a bug where, for a specific combination of source operands, the 72-bit FP adder was setting the carryout bit to 1 when there was no actual carryout
- The FMUL instruction had a bug where, when the rounding mode was set to “round up”, the sticky bit was not set correctly for certain combinations of source operand mantissa values, specifically:

$$\text{src1}[67:0] := X * 2^{(i+15)} + 1 * 2^i$$

$$\text{src2}[67:0] := Y * 2^{(j+15)} + 1 * 2^j$$
 where $i+j = 54$, and $\{X, Y\}$ are any integers that fit in the 68-bit range

These bugs could easily have gone undetected, not just in the pre-silicon environment but in post-silicon testing also.

5 Future Challenges

Validating the next generation of microprocessors is going to be a real challenge. One area that we are exploring is the development of a more abstract level of microarchitectural specification to help us in this task - both to slow the rate of growth for bugs from its historical trend line and to enable us to find bugs earlier in the design cycle. We are already applying formal methods at a higher level of abstraction during the microarchitecture definition phase of the project.

We are also looking to increase the contribution of formal verification to the overall validation effort. We are developing combined FPV and dynamic verification plans whose implementation will be coordinated so that we apply the best approach to the problem at hand. We are counting on the next generation of FPV tools like gSTE to provide greater capacity, thus reducing the effort needed to decompose problems into a tractable form. In addition, we are applying SAT solver technology for bug hunting (falsification), especially in combination with dynamic verification.

References

1. Lamport, L: The Temporal Logic of Actions. *ACM Transactions on Programming Languages and Systems* 16, 3, May 1994.
2. Colwell, R.P. and Brennan, R: Intel's Formal Verification Experience on the Willamette Development. *Proceedings of the 13th International Conference on Theorem Proving in Higher Order Logics*, August 2000.
3. Bentley, B: Validating the Intel® Pentium® 4 Microprocessor. *Proceedings of the International Conference on Dependable Systems and Networks*, June-July 2001.
4. Burns, D.: Pre-Silicon Validation of Hyper-Threading Technology. *Intel Technology Journal*, Volume 6, Issue 1, February 2002.
5. Kaivola, R and Narasimhan, N: Verification of Pentium®4 Multiplier with Symbolic Simulation & Theorem Proving. *Proceedings of the Design, Automation and Test in Europe Conference*, March 2002
6. Bentley, B: High Level Validation of Next-Generation Microprocessors. *Seventh Annual IEEE International Workshop on High Level Design, Validation and Test*, October 2002.
7. Schubert, T: High Level Formal Verification of Next-Generation Microprocessors. *Proceedings of the 40th Design Automation Conference*, June 2003.