Automating the Verification of RTL-Level Pipelined Machines

Panagiotis Manolios
College of Computing
Georgia Institute of Technology
Atlanta, Georgia 30318
Email: manolios@cc.gatech.edu

I. Introduction

There are two major approaches to pipelined machine verification. The first is based on the use of theorem provers such as ACL2 [2]. This approach is quite general and can be used to reason about machines defined at the RTL level, but the cost in terms of expert human guidance is quite high. The second approach is based on decision procedures such as UCLID [1]. This approach is very automatic, *e.g.*, in a carefully constructed experiment, problems that took ACL2 days took seconds with UCLID [4]. Unfortunately, this approach has several problems. For example, pipelined machines that exceed the complexity threshold of the tools used, which happens rather easily, cannot be analyzed. Another serious limitation is that the pipelined machine models used are term-level models: they abstract away the datapath, implement a small subset of the instruction set, require the use of numerous abstractions, and are far from executable. To be industrially useful, we need both automation and a firm connection to the RTL level. In this paper, we outline possible solutions and challenges.

II. Refinement

We view pipelined machine verification as an instance of the refinement problem: given an abstract specification, the instruction set architecture (ISA), and a concrete specification, the pipelined machine, show that the pipelined machine refines (implements) the instruction set. The exact theory of refinement we use is based on WEB-refinement, a theory of refinement that is compositional and preserves safety and liveness properties [3]. A refinement proof is relative to a *refinement map*, a function from pipeline states to ISA states that shows one how to view a pipeline state as an ISA state. Refinement plays a key enabling role in reasoning about RTL-level designs, because it allows us to confidently and easily reason about multiple levels of abstraction, something which we consider to be required for any successful approach to RTL-level verification. We have exploited this aspect of refinement already, as we show in the next few sections.

III. Compositional Reasoning

We have developed a compositional reasoning framework based on refinement that consists of a set of convenient, easily-applicable, and complete compositional proof rules [5]. Our framework greatly extends the applicability of decision procedures, *e.g.*, we were able to verify, in seconds, a complex, deeply pipelined machine that state-of-the-art tools cannot currently handle. Our framework can be added to the design cycle, which is also compositional. In addition, one of the most important benefits of our approach over current methods is that the counterexamples generated tend to be much simpler, in terms of size and number of simulation steps involved and can be generated much more quickly.

This research was funded in part by NSF grants CCF-0429924, IIS-0417413, and CCF-0438871.

IV. Combining ACL2 and UCLID

As an initial step towards verifying RTL-level designs, we have combined ACL2 and UCLID in order to verify pipelined machine models with bit-level interfaces [6]. We use ACL2 to reduce the proof that an executable, bit-level machine refines its instruction set architecture to a proof that a term level abstraction of the bit-level machine refines the instruction set architecture, which is then handled automatically by UCLID. The amount of effort required is about 3-4 times the effort required to prove the term-level model correct using only UCLID. This allows us to exploit the strengths of ACL2 and UCLID to prove theorems that are not possible to even state using UCLID and that would require prohibitively more effort using just ACL2.

V. Increased Automation

We view our work on compositional reasoning and on combining ACL2 and UCLID as promising first steps. Clearly, compositional reasoning will be needed to allow us to handle the verification problems one component at a time. With regard to RTL-level reasoning, our work has shown that we can automate the problem to within a small constant factor of what can be done for term-level models, but it will be important to change the factor to $1 + \varepsilon$. Several ideas for doing this include: developing a pattern database that with some intelligent search that can automatically decompose verification problems, automating some of the simpler refinement steps we currently perform, using counter-example guided abstraction-refinement to automatically abstract RTL designs to term-level designs, automating the handling of memories and register files, improved decision procedures, and creating analyses that work directly on hardware description languages.

References

[1] R. E. Bryant, S. K. Lahiri, and S. Seshia. Modeling and verifying systems using a logic of counter arithmetic with lambda expressions and uninterpreted functions. In E. Brinksma and K. Larsen, editors, *Computer-Aided Verification–CAV 2002*, volume 2404 of *LNCS*, pages 78–92. Springer-Verlag, 2002.

[2] M. Kaufmann, P. Manolios, and J. S. Moore. *Computer-Aided Reasoning: An Approach*. Kluwer Academic Publishers, July 2000.

[3] P. Manolios. *Mechanical Verification of Reactive Systems*. PhD thesis, University of Texas at Austin, August 2001. See URL `http://-www.cc.gatech.edu/∼manolios/publications.html`.

[4] P. Manolios and S. Srinivasan. A suite of hard ACL2 theorems arising in refinement-based processor verification. In M. Kaufmann and J. S. Moore, editors, *Fifth International Workshop on the ACL2 Theorem Prover and Its Applications (ACL2-2004)*, November 2004. See URL `http://-www.cs.utexas.edu/users/moore/acl2/workshop-2004/`.

[5] P. Manolios and S. Srinivasan. A complete compositional reasoning framework for the efficient verification of pipelined machines. In *ICCAD-2005, International Conference on Computer-Aided Design*, 2005.

[6] P. Manolios and S. Srinivasan. Verification of executable pipelined machines with bit-level interfaces. In *ICCAD-2005, International Conference on Computer-Aided Design*, 2005.