On the Verification of Synthesized Designs Using Automatically Generated Transformational Witnesses

Elena Teica Rajesh Radhakrishnan Ranga Vemuri
Dept. of ECECS, University of Cincinnati, Cincinnati, OH 45221-0030
{eteica,rradhakr,ranga}@ececs.uc.edu

Abstract

This poster presents a new methodology for verifying the synthesized designs, and for debugging the software implementation of high-level synthesis algorithms. The methodology is based on a set of 7 RTL transformations which are able to emulate the effect of many scheduling and resource allocation algorithms.

The verification and debugging methodology presented here is based on the observation that it is possible to generate a sequence of elementary transformations that perform the same tasks as a traditional high-level synthesis tool (but in a transformational manner) by examining the binding data structures, without any knowledge of the synthesis algorithm used to perform the task. For example, a sequence of register instance substitutions can be generated by looking at the register binding without knowing whether a clique partitioning algorithm or a left-edge algorithm was used for register allocation. Similar observations were exploited in the past [1]. The novelty of our approach lies in the technique of *witness generation* for software debugging.

RTL Transformations. We identified and formally specified in PVS's [2] higher-order logic, the models for a set of 7 elementary RTL transformations which we found to be sufficient to emulate the effects of many existing high-level synthesis algorithms. These transformations are similar to those presented in [4]. We mechanically proved using PVS that each of the seven transformations is *correct* (it preserves the computational behavior of the design) provided a *certain set of preconditions* is satisfied. The information expressed by these preconditions is exploited in the debugging process.

Witness Generators. The *witness generators* analyze the binding information supplied by the synthesis system and automatically identify a sequence of elementary RTL transformations which when applied to the initial design achieves the same outcome as the synthesis software. Figure 1 shows a sketch of our approach. Each transformation identified is applied to an initial design implementation where it can be checked if its preconditions are satisfied, and if not, record the failed precondition. Failure of a precondition cannot reveal an error, but it can be recorded and interpreted for debugging purposes. During the proof exercise we tried to identify weaker preconditions by inserting weak hypothesis in the antecedent in an iterative manner. Using this methodology we were able to expose a seeded error in the scheduling algorithm implementation in the DSS system [3].

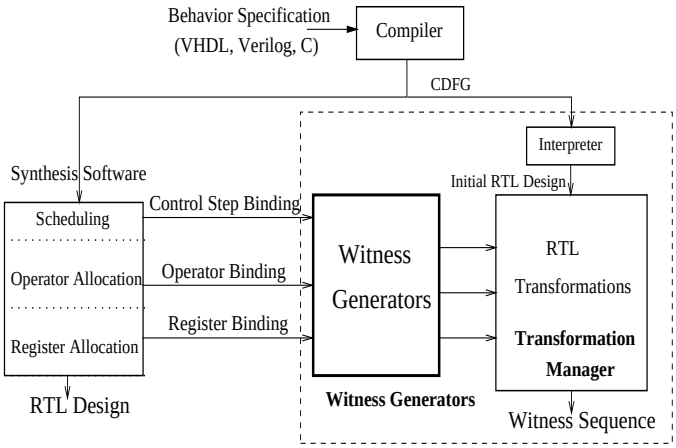

Figure 1. Debugging Methodology

References

[1] H. Eveking, H. Hinrichsen, and G. Ritter. Automatic Verification of Scheduling Results in High Level Synthesis. In *Design, Automation and Test in Europe (DATE)*, 1999.

[2] S. Owre, J. M. Rushby, and N. Shankar. PVS: A prototype verification system. In *(CADE'92)*, Lecture Notes in Artificial Intelligence, pages 748–752. Springer-Verlag.

[3] J. Roy, N. Kumar, R. Dutta, and R. Vemuri. DSS: A Distributed High Level Synthesis System. In *IEEE Design and Test of Computers*, volume 9, pages 18–32, 1992.

[4] R. Vemuri. *A Transformational Approach to Register-Transfer Level Design Space Exploration.* PhD thesis, Case Western Reserve University, 1989.