A Unified Approach for Combining Different Formalisms for Hardware Verification*

Klaus Schneider and Thomas Kropf

University of Karlsruhe, Department of Computer Science,
Institute for Computer Design and Fault Tolerance (Prof. D. Schmid),
P.O. Box 6980, 76128 Karlsruhe, Germany,
e-mail: {Klaus.Schneider, Thomas.Kropf}@informatik.uni-karlsruhe.de,
http://goethe.ira.uka.de/hvg/

Abstract. Model Checking as the predominant technique for automatically verifying circuits suffers from the well-known state explosion problem. This hinders the verification of circuits which contain non-trivial data paths. Recently, it has been shown that for those circuits it may be useful to separate the control and data part prior to verification. This paper is also based on this idea and presents an approach for combining various proof approaches like model checking and theorem proving in a unifying framework. In contrast to other approaches, special proof procedures are available to verify circuits with data sensitive controllers, where a bidirectional signal flow between controller and data path can be found. Generic circuits can be verified by induction or by model checking finite instantiations.

By giving the system 'proof hints', also the verification effort for model checking based proofs can be considerably reduced in many cases. The paper presents an introduction to the different proof strategies as well as an algorithm for their combination. The underlying CQS system also allows the efficiency evaluation of different approaches to verify the same circuits. This is shown in different case studies, demonstrating the trade-off between interaction and verifiable circuit size.

1 Introduction

Formal verification is the task of proving that a given specification holds for a certain design. Although more powerful than traditional simulation, a breakthrough in industrial use has been achieved only after the introduction of binary decision diagrams and symbolic state traversal algorithms [1, 2], leading to powerful techniques like symbolic model checking [3]. Using these approaches, a fully automated verification of significantly large circuits has become possible. However, as only finite state systems can be modeled, the size of the processable circuits is limited, due to the so-called state explosion problem. This restriction does not hold for theorem prover based approaches, where mostly higher-order

* This work has been financed by the DFG project Automated System Design, SFB No.358.
logic is used as the underlying formalism [4]. Unfortunately, these approaches require a considerable amount of manual interaction. Thus various approaches have been presented to partially automate the verification by incorporating automated reasoning procedures [5, 6] or by adding abstraction and compositional verification techniques to allow larger systems to be verified than by finite state approaches [7, 8, 9].

Many circuits are composed of a controller and a data path. Recently it has been shown, that it is useful to separate these parts prior to verification [10, 11, 12, 13, 14]. Proceeding this way, finite state approaches may be used for the controller part 'guiding' the verification of the data path, whereas the latter often requires theorem proving techniques. However, these approaches are also not able to provide full automation for circuits with heavy interaction between control and data. This paper presents a new approach which provides a general framework called C@S for exploring the combination of different proving techniques. The system aims at providing many different decision procedures which are invoked as soon as automizable proof goals are detected. Other verification goals are split up interactively into subgoals until they can be finally proved by decision procedures. C@S has been implemented on top of the HOL [15] system and has interfaces to the SMV [16] system and RRL [17]. As a result, C@S currently enriches HOL by the following decision procedures and proof methods:

- linear temporal logic theorem proving and model checking
- CTL model checking as implemented in SMV
- \( \omega \)-automata
- Pressburger arithmetic
- Büchi's monadic second-order theory of one successor
- inductionless induction as implemented in the RRL system
- invariant rules for eliminating goals with data dependent processing

In order to ease the adaption of proof goals to different proof approaches, a class of higher-order formulas called hardware formulas has been defined as a uniform representation for specifications and implementations [18]. In hardware formulas, time and data is represented separately. Once a proof goal has been transformed in hardware formulas, it may be converted according to the proof method to be used. We have provided an algorithm for automatically choosing the combination of proof procedures for a given circuit.

In case of data dependent loops, invariant based techniques may be used to cut the data dependency. The resulting proof goals without data dependency can then be proved by traditional inductive reasoning or simple rewriting. Thus it is possible to perform inductive proofs for generic circuits with data dependencies. Moreover, those techniques also allow to indicate 'proof hints' to the system that allow to significantly reduce the effort of model checking. Thus it is sometimes possible to verify a generic circuit with a concrete but large desired bit width, not being possible before.

The idea of enriching interactive theorem provers for higher order logic by powerful decision procedures can also be found in the PVS system. However, PVS