An Approach to Link HOL and MDG for Hardware Verification

V.K. Pisini[1], S. Tahar[1], O. Ait-Mohamed[2], P. Curzon[3] and X. Song[4]

[1] ECE Dept., Concordia University, Canada
[2]Cistel Technology, Inc., Canada
[3]School of Computing Science, Middlesex University, UK
[4] IRO Dept., Université de Montreal, Canada

ABSTRACT- In order to overcome the limitations of automated tools and the cumbersome proof process of interactive theorem proving, we adopt an hybrid approach for formal hardware verification by linking HOL and MDG. This approach uses the strengths of theorem proving (HOL) with its powerful mathematical tools such as induction and abstraction, and the advantages of automated tools (MDG) which support equivalence checking and model checking.

I. Introduction and Related Work

There are several approaches to formal hardware verification: theorem-proving, model checking, equivalence checking, symbolic simulation to name a few [1], each of which has its own strengths and weaknesses. In this paper, we present a methodology as to how equivalence checking of the automated MDG system [2] supports the proof process of the HOL theorem prover [3] thereby bringing in the advantages of both. The MDG system which allows equivalence checking and model checking is based on Multiway Decision Graphs—an extension of the traditional ROBDDs. In HOL, which is built on higher-order logic, hierarchical verification is possible wherein the modules are divided into sub-modules. In our hybrid approach, we enable the verification of sub-modules using the MDG system.

There exist a number of hybrid approaches such as combining theorem proving with model checking [4-5] and combining theorem proving and symbolic trajectory evaluation [6]. Rajan et. al [5] described an approach where a BDD-based model checker for the propositional mu-calculus has been used as a decision procedure within the framework of the PVS proof checker. Joyce and Seger [6] described an approach by means of an interface between the Voss system and HOL. Schneider et. al [4] proposed an approach of invoking model checking from within HOL where properties are translated from HOL to temporal logic.

II. Methodology

The work described in this paper is part of a larger project to link VHDL, HOL and MDG as shown in Figure 1. Here, the VHDL model is analyzed to get a data structure (Directed Acyclic Graph—DAG) of the model which is passed through an HOL Generator to get the HOL model. Within HOL, we use a function, MDG_TAC, to generate the required files for the MDG system to complete the verification. In the case of property verification, an LTL property description (L_MDG) is transformed into an equivalent VHDL or MDG-HDL circuit description that will either be fed into the Analyzer or directly to the MDG system, respectively.

In our hybrid approach, the verification starts in HOL with a goal to prove that an implementation implies a specification. Implementation and specification are described in HOL. In order to ease the use of the integrated MDG-HOL system, the specification is written in a table form of MDG [7]. Figure 2 shows the block diagram of our hybrid system. The interface block in Figure 2 takes the HOL description of the sub-goal, generates all required MDG files and returns the MDG verification result back to HOL. In the positive case (verification succeeded), it creates a theorem stating the proof of the sub-goal. In the above procedure, we make sure, however, that the expression of the HOL sub-goal is acceptable by the MDG system to be proved through equivalence checking. As part of the build-up of the interface, Curzon et. al [7] described a way to express MDG tables in HOL, which we used as a formal link.

For each sub-goal which is an assertion to be verified in MDG, the specification and implementation are translated by invoking MDG_TAC which is an ML function (tactic) that converts HOL expressions into MDG-HDL [8] circuit description, and also into MDG tables. Besides these two descriptions, the MDG system requires additional information such as the order of used variables and state encoding (for sequential verification). This information has to be supplied by the user as descriptions in HOL. Finally, the MDG system is called and the corresponding files are executed to get the verification result. The individual tasks of MDG_TAC are summarized in Figure 3.

Once the equivalence checking has succeeded, MDG returns "true" and this result is imported into HOL in the form of a theorem (using the *make_theorem* in HOL) and the main proof procedure continues in HOL with the next sub-goal to be proved. In case a sub-goal is not expressed in the MDG acceptable form or the MDG verification fails, then the regular HOL proof procedure is followed.

III. Conclusions and Future Work

Equivalence checking yields stronger result than implication. The verification with this hybrid tool is faster since it is partially automated. The behavioral description can easily be expressed in HOL using the table format of MDG [7]. The present compiler can be extended to accommodate model checking to be used as a decision procedure within HOL. We will know the limitations of this hybrid tool when we take up the verification of large designs including a 16 by 16 ATM switch fabric [9]. The described hybrid approach follows a top-down approach starting from HOL and proving sub-goals using MDG. We will also be working on bottom-up approach where a given circuit implementation is divided into smaller parts and the verification is carried with MDG first and later exported into HOL for higher level verifications.

References

[1] C. Seger, "An Introduction to Formal Hardware Verification," Tech. Rep. 92-13, Dept. of Computer Science, University of British Columbia, Vancouver, B.C., Canada, June 1992.

[2] F. Corella, Z. Zhou, X. Song, M. Langevin, and E. Cerny, "Multiway Decision Graphs for Automated Hardware Verification," *Formal Methods in System Design*, vol. 10, no. 1, pp. 7–46, 1997.

[3] M. Gordon and T. Melham, *Introduction to HOL: A Theorem Proving Environment for Higher-Order Logic.* Cambridge University Press, Cambridge, U.K., 1993.

[4] K. Schneider and T. Kropf, "Verifying Hardware Correctness by Combining Theorem Proving and Model Checking," Tech. Rep. SFB358-C2-5/95, University of Karlsruhe, Karlsruhe, Germany, December 1995.

[5] S. Rajan, N. Shankar, and M. Srivas, "An Integration of Model-checking with Automated Proof Checking," in *Computer Aided Verification* (P. Wolper, ed.), vol. 939 of *Lecture Notes in Computer Science*, (Liege, Belgium), pp. 84–97, Springer-Verlag, 1995.

[6] J. Joyce and C. Seger, "Linking BDD-based Symbolic Evaluation to Interactive Theorem Proving," in *Proceedings of the 30th Design Automation Conference*, Association for Computing Machinery, 1993.

[7] P. Curzon, S. Tahar, and O. Ait-Mohamed, "Verification of the MDG Components Library in HOL," in *Theorem Proving in Higher Order Logics: Emerging Trends* (J. Grundy and M. Newey, eds.), (Australian National University, Canberra, Australia), pp. 31–45, September 1998.

[8] Z. Zhou and N. Boulerice, *MDG Tools (V1.0) User's Manual.* Dept. of Computer Science, University of Montreal, Montreal, Canada, June 1996.

[9] I. Leslie and D. McAuley, "Fairisle: An ATM Network for the Local Area," *ACM Communication Review*, vol. 19(4), pp. 327–336, 1991.

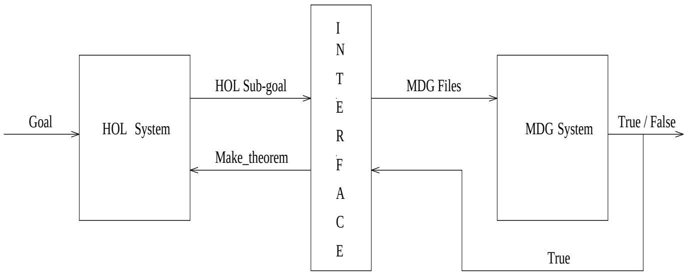

Fig. 2. Block Diagram of the Hybrid System

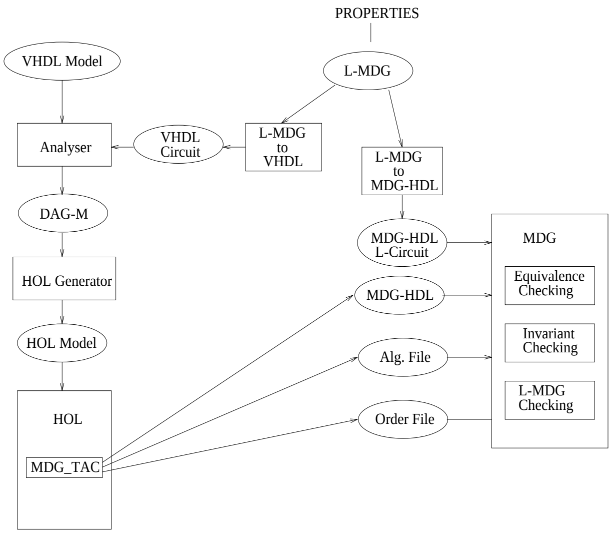

Fig. 1. Intended VHDL-HOL-MDG Project Skeleton

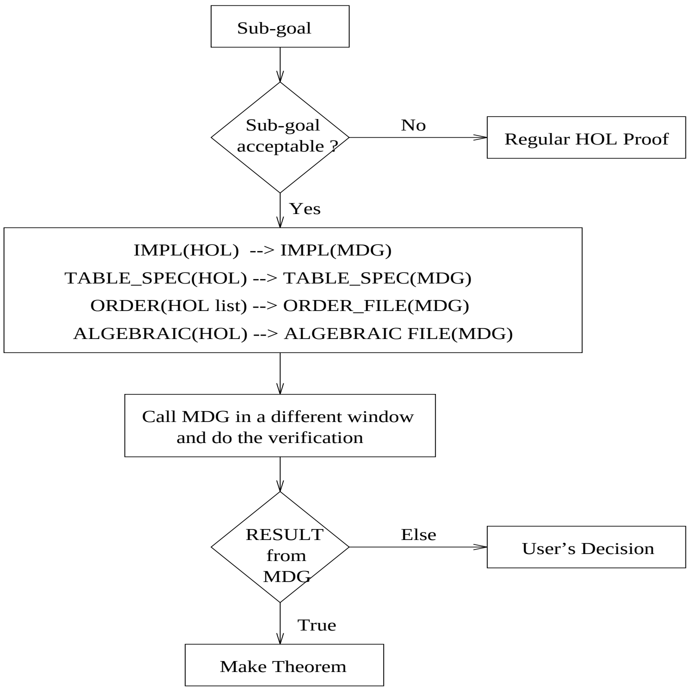

Fig. 3. Task of MDG_TAC as a Flow Diagram