# HOL-MDG : A Hybrid Tool for Formal Verification

S. Kort[1], S. Tahar[1], P. Curzon[2] and X. Song[3]
[1]ECE Dept., Concordia University, Montreal, Canada. Email: {tahar, kort}@ece.concordia.ca
[2] School of Computing Science, Middlesex University, U.K. Email: p.curzon@mdx.ac.uk
[3] IRO Dept., University of Montreal, Canada. Email: song@iro.umontreal.ca

**Abstract.** *We describe a hardware verification tool called HOL-MDG. This tool combines the HOL theorem prover with an automated verification package, namely MDG. The aim of such a combination is to bring together the strength of theorem proving and the automation of MDG. Moreover, the presented hybrid tool offers facilities for a hierarchical verification approach.*

## 1. Introduction

Formal verification methods fall in one of three categories: theorem proving, decision diagrams based methods and symbolic simulation. In this work, we focus on combining the first two categories. In theorem proving methods, the design's behavior as well as its structure are described in some formal logic. Then the design structure is proved to conform to the expected behavior using a set of axioms and inference rules. Theorem provers generally provide very powerful reasoning and abstraction mechanisms. This makes it possible to deal with complex designs. Nevertheless, theorem provers require a deep understanding of their underlying logic. They also involve a lot of interactions with the user. Decision diagrams based tools include equivalence and model checkers. These tools are easy to use since the verification is performed automatically. However, they fail to verify complex designs due to the state explosion problem. Therefore, combining both categories should enable verifying complex designs with much less interaction with the verification tool. Another way to cope with complex designs is to apply a hierarchical verification approach. In such an approach, the design consists of a block hierarchy. Individual blocks are verified separately, then their correctness results are combined to verify the next level blocks.

## 2. The Hybrid Tool

We describe in this paper a hybrid tool combining the HOL theorem prover [5] and a hardware verification package, namely MDG [2]. The integration of both tools was performed using the PROSPER toolkit [4].

Many integration methodologies have been investigated recently. In [1], a tool combining the ThmTac prover and the VOSS symbolic trajectory evaluation system has been presented. In [8], Schneider and Hoffmann have used PROSPER to link the SMV model checker with HOL. Hurd [6] also integrated the Gandalf prover and HOL using PROSPER.

Our hybrid tool is provided with a hierarchical structural specification of the design to be verified as well as a behavioral specification for every block. The structural specification is expressed using an embedding of the MDG built-in components in HOL. The behavioral specifications are expressed using an embedding of MDG tables in HOL [3]. MDG tables are a generalization of truth tables. The verification follows the HOL goal-oriented proof style [5]. Figure 1 shows a typical session with the hybrid tool. Initially, a goal stating that the design's structural specification implies its behavioral specification is set. This goal could be resolved in three different ways. The first way is to invoke the MDG equivalence checker using either the tactic `MDG_SEQ_TAC` or `MDG_COMB_TAC` [7]. In case of success, a theorem stating the correctness of the design is generated. Though, the equivalence checking may fail because of state explosion or because the structural specification is not equivalent to the behavioral specification (i.e. only the implication holds). As a second alternative, the user may apply hierarchical verification by invoking the tactic `HIER_VERIF_TAC`. This tactic generates a correctness subgoal for every sub-block. It also generates a proof for the whole design assuming the correctness of its sub-blocks. The third verification alternative is to perform a conventional HOL proof. Figure 2 shows the structure of the hybrid tool. The tool includes a parser (Parsing), a module for flattening hierarchical specifications (Extraction), a module to support hierarchical verification (HierarchicalVerificationSupport), a module to generate all the files needed by MDG and a module to manage the interaction between HOL and MDG (MDGInteraction). The tool was implemented in SML. The Parsing module was generated automatically from a grammar specification. The MDGInteraction module is based on the PROSPER plug-in interface [4].

## 3. Conclusions and Future Directions

We have described a hybrid tool combining the HOL theorem prover and the MDG system. The tool is intended to deliver the verification engineer from the cumbersome of theorem proving yet allowing him/her to deal with complex designs. Furthermore, the tool offers some facilities for a hierarchical verification approach. Indeed, it generates automatically the correctness sub-goals for lower-level sub-blocks as well as a correctness proof for the upper-level blocks assuming the correctness of their constituents. We are planning to use the tool in the verification of real world designs to assess the effectiveness of the suggested methodology. The tool can also be extended with a temporal property checker. Ongoing research focuses on interfacing the tool with VHDL in the aim of integrating the hybrid verification methodology in the design flow.

## References

[1] M. D. Aagaard, R. B. Jones and C-J. H. Seger. Lifted-FL: A Pragmatic Implementation of Combined Model Checking and Theorem Proving. In *TPHOLs'99*, LNCS 1690, France, 1999.

[2] F. Corella, et al. Multiway Decision Graphs for Automated Hardware Verification. *Formal Methods in System Design*, Vol. 10, 1997.

[3] P. Curzon, et al. Verification of the MDG Components Library in HOL. In *Theorem Proving in Higher Order Logics: Emerging Trends*, Canberra, Australia, 1998.

[4] L. A. Dennis, et al. The PROSPER Toolkit. In *TACAS'00*, Berlin, Germany, 2000.

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

[6] J. Hurd. Integrating Gandalf and HOL. In *TPHOLs'99*, LNCS 1690, Nice, France, 1999.

[7] V. K. Pisini, et al. Formal Hardware Verification by Integrating HOL and MDG. In *GLS-VLSI'00*, Chicago, USA, 2000.

[8] K. Schneider and D. W. Hoffmann. A HOL Conversion for Translating Linear Time Temporal Logic to *w*- Automata. In *TPHOLs'99*, LNCS 1690, Nice, France, 1999.

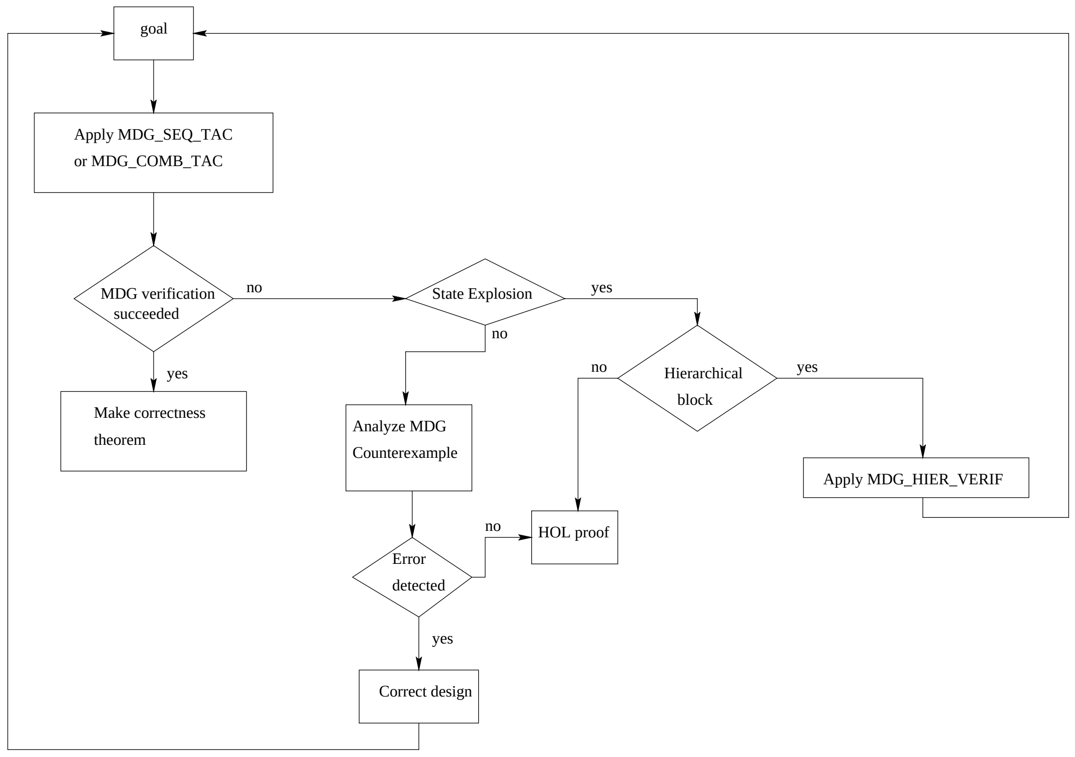

**Figure 1. The Verification Methodology**

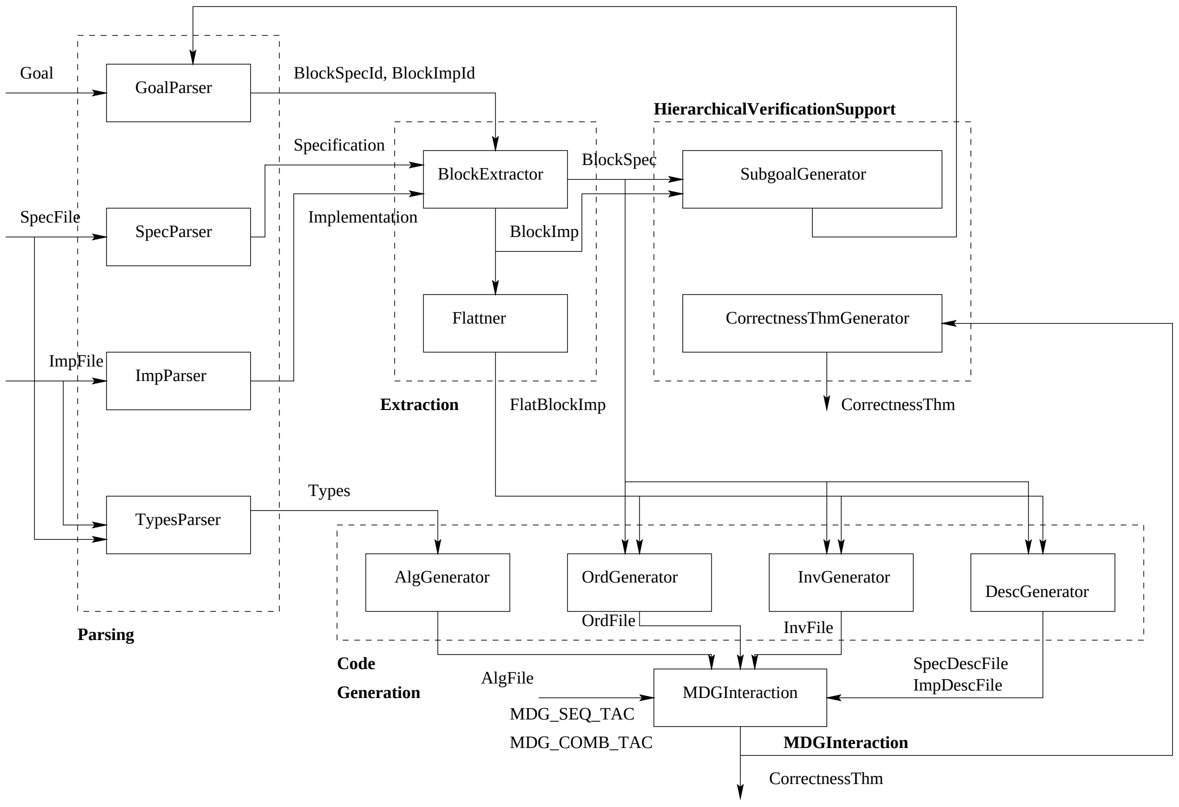

**Figure 2. The Hybrid Tool's Structure**