A Tool for Verifying ASM Models Using Multiway Decision Graphs

Amjad Gawanmeh, Sofiène Tahar, and Kirsten Winter
Concordia University
Address: Electrical & Computer Engineering Department, 1455 Maisonneuvve, Montreal, Quebec. H3H 1MG.
Tel: (514) 848-3124 Email: amjad@ece.concordia.ca

Abstract. *In this paper we present a formal hardware verification tool linking ASM with MDG. ASM (Abstract State Machines) is a state based language for describing transition systems. MDG (Multiway Decision Graphs) provides symbolic representation of transition systems with support of abstract sorts and functions. We implemented a transformation tool that automatically generates MDG models from ASM specifications, then formal verification techniques provided by the MDG tool, such as model checking or equivalence checking, can be applied on the generated models. We support this work with a case study.*

1. Introduction

There has been a recent surge of interest in formal verification and tool support recently, this is because of the increasing complexity of digital VLSI systems, and as a result, it is becoming impossible to simulate large designs adequately. In this work we describe an approach to interface Abstract State Machines (ASM) [4] with Multiway Decision Graphs (MDG) [1] to enable tool support for the formal verification of ASM descriptions. ASM is a formal specification method for software and hardware modeling and provides a powerful means of modeling various kinds of systems. An ASM model describes the state space of the system by means of universes or functions, and the state transitions by means of transition rules. MDGs are decision diagrams based on abstract representation of data and are used for modeling hardware systems in first place. The MDG tool provides equivalence checking and model checking applications based on MDG. The modeling language for the MDG tool is the hardware description language MDG-HDL.

2. ASM-MDG Tool

We chose to interface ASM with the MDG tool for three reasons: first, both notions, ASM and MDGs, are closely related to each other since they are both based on a subset of many-sorted first-order logic and the abstract representation of data. They also both support uninterpreted functions which is not available in many hardware modeling languages. Second, MDGs as data structure for representing transition systems provide a powerful means for abstraction in order to fit large models into the model checking process. Finally, the need to provide the MDG tool with a high-level modeling language, namely ASM, would allow MDG users to model a wide range of applications in an acceptable fashion [3].

Our tool consists of two parts in order to treat both structural and behavioral models of hardware: the first part generates MDG-HDL behavioral description from ASM specifications, while the second part generates MDG-HDL structural description. For each of these models, variable order and algebraic specifications [1] are also generated. We developed the interface for behavioral designs in two steps as shown in Figure 1: in the first step a model in Specification Language ASM-SL [2] is transformed into a flat, simple transition system, called the Intermediate Language (ASM-IL) [5]. The second step provides a transformation from ASM-IL into the syntax of the MDG-HDL.

For structural designs, we adapten an ASM-MDG direct interface, which works in three steps (Figure 2), first a parser is used to check the input model and validate its syntax, it collects ASM domains, ASM functions and transition rules in the model, then an analyzer is used to treat the data structures produced by the parser in order to construct design components, variables, functions and sorts that represent the design. Finally, a generator is used to produce MDG-HDL models based on the information collected in the previous step, these models are variable order, algebraic speifications and MDG-HDG structural design.

3. Application

We have applied this transformation schema on a case study of an Island Tunnel Controller by specifying behavioral and structural models in ASM-SL, then using our ASM-MDG interface to generate the corresponding MDG-HDL models, and verify them with the MDG tool. A set of properties have been verified for both models with MDG model checking, then the implementation was shown to be equivalent to the specification using MDG equivalence checking.

4. Conclusions

In this paper, we introduced an interface from ASM to the MDG tool, called “ASM-MDG”. The interface automatically transforms the ASM specification language, ASM-SL, into the MDG hardware description language, MDG-HDL. This enables the verification of ASM designs using the MDG tool, and supports the MDG tool with a high level modeling language, ASM-SL.

References

[1] 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, February 1997, pp. 7-46.

[2] G. Del Castillo. The ASM Workbench: A Tool Environment for Computer-Aided Analysis and Validation of Abstract State Machine Models. Ph.D. Thesis, Heinz Nixdorf Institute, Paderborn, Germany, 2000.

[3] A. Gawanmeh, S. Tahar and K. Winter: Interfacing ASMs with the MDG Tool; Abstract State Machines - Advances in Theory and Applications, Lecture Notes in Computer Science 2589, Springer Verlag,. 2003. pp 278-292.

[4] Y. Gurevich. Evolving Algebras 1993: Lipari Guide. In E. Borger, editor, Specification and Validation Methods. Oxford University Press, 1995.

[5] K. Winter. Model Checking Abstract State Machines, Ph.D. thesis, Technical University of Berlin, 2001.

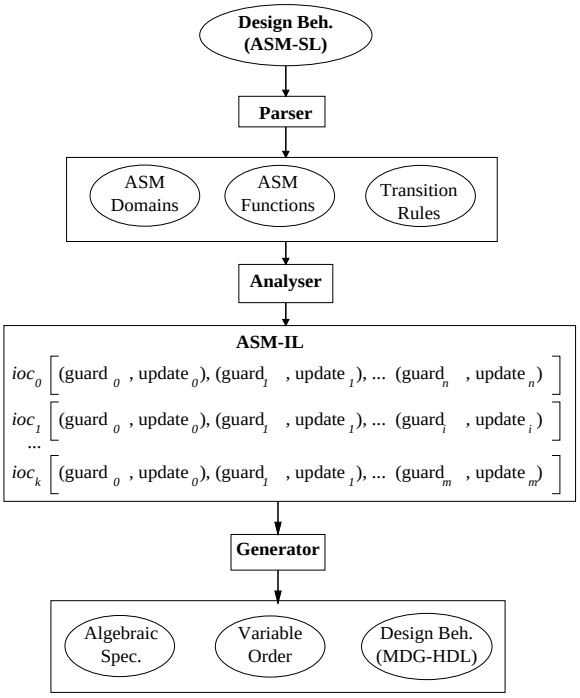

Figure 1. ASM-MDG Interface via ASM-IL for Behavioral Designs.

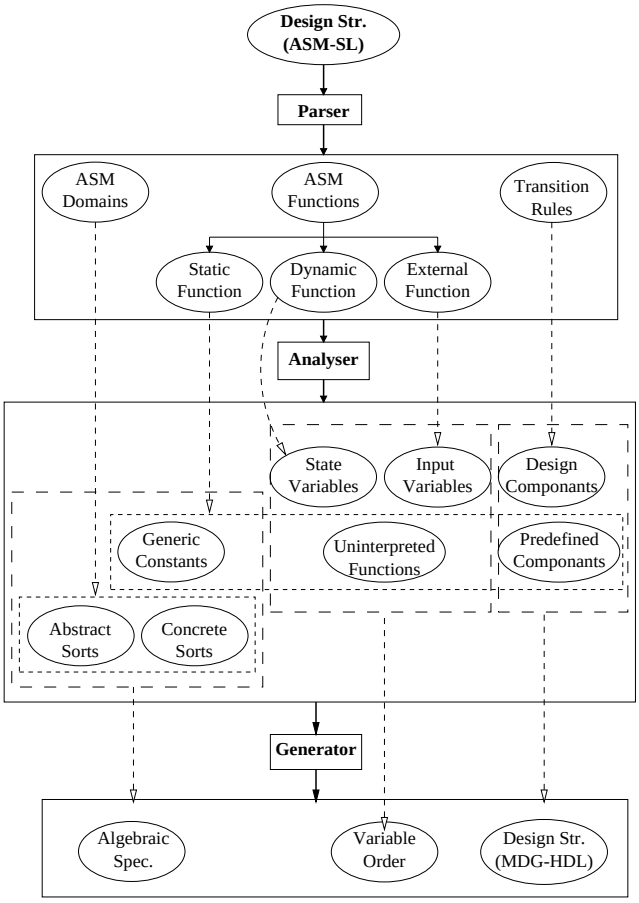

Figure 2. ASM-MDG Direct Interface for Structural Designs.