SCALABLE HARDWARE VERIFICATION WITH SYMBOLIC SIMULATION

SCALABLE HARDWARE VERIFICATION WITH SYMBOLIC SIMULATION

VALERIA BERTACCO
University of Michigan

Valeria Bertacco
The University of Michigan
Advanced Computer Architecture Lab
Department of EE & CS
1301 Beal Avenue, Room 2224
Ann Arbor, MI 48109
U.S.A.

Scalable Hardware Verification with Symbolic Simulation

Cover design by S. Alexander Garcia

Library of Congress Control Number: 20055934803

ISBN-10: 0-387-24411-5
ISBN-13: 9780387244112
ISBN-10: 0-387-29906-8 (e-book)
ISBN-13: 9780387299068 (e-book)

Printed on acid-free paper.

© 2006 Springer Science+Business Media, Inc.
All rights reserved. This work may not be translated or copied in whole or in part without the written permission of the publisher (Springer Science+Business Media, Inc., 233 Spring Street, New York, NY 10013, USA), except for brief excerpts in connection with reviews or scholarly analysis. Use in connection with any form of information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed is forbidden.
The use in this publication of trade names, trademarks, service marks and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

Printed in the United States of America.

9 8 7 6 5 4 3 2 1 SPIN 11325031

springeronline.com

To Roberta, Bruno,
Livio and Todo.

Contents

List of Figures

List of Tables

Preface

In the past 40 years, electronic systems have become a pervasive force in modern society. Digital integrated circuits (ICs) are at the heart of a large majority of these systems. These digital ICs are complex systems comprised of millions of interconnected transistors in a very small area. Moreover, the underlying semiconductor fabrication technology used to manufacture these ICs allows for the doubling of the number of transistors in the same area approximately every 18 months.

The design of digital systems is an intricate and time consuming process that progresses through various phases and levels of abstraction relying heavily on CAD (Computer-Aided Design) software tools. Within this context, ensuring the correctness of these digital systems is a critical consideration, especially because failure costs are becoming increasingly high. One of the most famous, recent examples of the importance of correct design is the Intel Pentium flaw in the floating point divide unit in 1994 that eventually forced Intel to replace many of the Pentium chips that were already in the market. In many cases, the possibility of failure is plainly unacceptable. Examples of these applications are transportation systems, medical applications and financial systems. Driven by the importance of correct design, the cost of verification in modern computing systems has grown to dominate the cost of system design in terms of the time and human resources dedicated to it. In contrast, even though guaranteeing the correctness of a design is such a central aspect of its development, current verification methodologies are still inadequate to tackle the complex systems that are being developed nowadays. Hardware design companies try to compensate for mediocre CAD tools by dedicating the majority of their resources to verification, yet are still unable to guarantee correct functionality over the entire design space.

In industry, the scalability, flexibility and predictable run-time behavior of logic simulation makes it the most widely accepted technique for ensuring the correctness of digital ICs. The technique is based on verifying a digital system

by providing sequences of binary values for each of the inputs of the system and checking that the corresponding outputs are correct, based on what the design team expected to see or described in a specification document. However, logic simulation can usually visit only a small fraction of all the possible configurations of a system - also called the state space - and, thus, the discovery of bugs heavily relies on the expertise of the designer to select a few crucial configurations to verify.

Symbolic simulation is another verification method that is attracting increasing interest because it allows the verification engineer to explore all, or a major portion, of a circuit's state space without the need to design time-consuming test stimuli. However, this approach poses a high demand on the resources of the simulating host, and in particular, on the memory system, because of the complexity of the algorithms involved and their unpredictable runtime behavior. Thus, the scalability of this approach has been the main limiting factor to its mainstream deployment, with the consequence that, thus far, its scope has been limited to small systems.

About this book

This book presents recent advancements in symbolic simulation-based solutions which radically improve scalability. We overview current verification techniques, both based on logic simulation and on formal verification methods, and we describe in detail the baseline technique of symbolic simulation. The core of this book focuses on new techniques that narrow the performance gap between the complexity of digital systems and the limited ability to verify them. In particular we cover a range of solutions that exploit approximation and parametrization methods in order to achieve this goal. In the direction of approximation techniques, we comprehensively cover quasi-symbolic simulation – an aggressive technique aiming at simulating only the portion of the design necessary for the verification goal at hand – and cycle-based symbolic simulation – a unique combination of formal methods and logic simulation that can stimulate a circuit with a very large number of input combinations and sequences in parallel. Cycle-based symbolic simulation is a hybrid solution that uses both approximation and parametrization to attain its scalability goal. Its key concept is the use of a parametric form to represent the set of states visited during simulation. This approach maintains a high degree of scalability, in line with current logic simulation techniques, while achieving better efficiency.

In the realm of parametric solutions, we discuss a range of approaches, including various applications of parametric symbolic simulation to industrial microprocessor designs. An in-depth analysis is dedicated to another solution that we recently proposed, disjoint-support decomposition-based symbolic simulation, where the parametrization makes use of the disjoint-support decomposition properties of a Boolean function. This simulation technique is

rooted on a novel algorithm that exposes the disjoint decomposition properties of a Boolean function by restructuring its BDD representation. The new algorithm is very efficient in the sense that it has worst-case complexity that is only quadratic in the size of the initial BDD, compared to that of previous solutions which had exponential complexity in the number of input variables of the function. We deploy this algorithm to decompose of the state functions in symbolic simulation. Then, by restructuring the next-state functions using their disjoint components, it is possible to transform them into a simpler parametric form without sacrificing simulation accuracy. Results show that this technique is effective in reducing the memory requirements of symbolic simulation while maintaining exact state exploration. When the design complexity becomes overwhelming, it can trade-off search breadth for performance, and proceed to simulate very large trace sets in parallel, thus maintaining a simulation speed and memory profile that are close to logic simulation.

In structuring this book, the hope was to provide an interesting reading for a broad range of readers. Chapters 1, 2 and 3 constitute a panoramic flight over the world of digital systems' design and, in particular, verification. Chapter 3 reviews some of the mainstream symbolic techniques in formal verification, dedicating most of the focus to symbolic simulation.

We use Chapter 4 to cover the necessary principles of parametric forms and disjoint-support decompositions. In particular, we attempt to keep the material at a level that facilitates understanding, but without too many formal details. While there is a range of resources discussing parametric forms and parametrizations for Boolean functions, we felt that the topic of disjoint-support decompositions was not as readily available. For that reason Appendix A complements Chapter 4 in providing a more formal presentation of the topic and derivation of the theoretical results.

Chapters 5 and 6 focus on a range of recent symbolic simulation techniques, which we grouped in approximate solutions, and exact parametrizations. Finally, Chapter 7 wraps up the presentation and outlines possible future research directions.

Acknowledgments

I would like to acknowledge several people who were critical in making this book a reality through their work, advice and support. The main solutions discussed by this book were developed during my Ph.D. work at Stanford, with the supervision of Kunle Olukotun, who had always been prompt and available in supporting whatever direction of research, and of life, I pursued. In our technical interactions, he would always move straight to the results of my work and challenge me on their practical contribution to improve the quality of verification in industrial designs. David Dill has been the person I could always go to bounce ideas off of and to have illuminating technical discussions. When my ideas could survive his dissecting analysis, I knew I could publish them.

My years at Synopsys have played a central role in shaping my understanding of design verification as an industrial challenge first and a research area later. My colleagues have been crucial in providing me with invaluable opportunities: Ghulam Nurie, Swami Venkat and the Vera Group team, who gave me early opportunities to interact with customers. Those customer meetings have always been enlightening in my quest towards understanding the needs of hardware designers; Pei-Hsin Ho, my manager in the Advanced Technology Group of Synopsys, showed me how to efficiently achieve technology transfers, by taking academic research and deploying it in software solutions for the hardware-design community. My undergraduate advisor, Maurizio Damiani, first introduced me to research and to the area of Computer-Aided Design for integrated circuits. I would like to thank him for the numerous interactions and collaborations that lasted long after my undergraduate studies and spurred many of the publications that led to this research work.

On a personal level, I would like to thank my parents for teaching me the first concepts of mathematics and logic and for introducing me, early in my life, to the pursuit of both education and industry experience. I also want to thank my family for supporting my choices in my path through life. My brother, Livio, provided all sorts of technical support and advice and solved many sys-

tem crashes, most often connecting from some remote location in Europe. My colleagues and my students at the University of Michigan have been an important source of support in continuing this research and in helping shuffle all the work and deadlines that are always overlapping: Kai-hui Chang, Steve Plaza, Smitha Shyam and Ilya Wagner. I would also like to thank Todd Austin for the numerous technical discussions and for taking the time to review this book multiple times. In addition, I would like to thank Gloria Cadavid, Alex Garcia, Tim Wright and Azita Emami.

Finally, I would like to express gratitude to my editors Michael Hackett and Alex Greene and their collaborators, Rose Antonelli, Melissa Guasch and Rebecca Olson at Springer for their help in preparing the final manuscript and keeping me on task, particularly during a year of transition and growth in the company.