Stephan Eggersglüß · Rolf Drechsler

High Quality Test Pattern Generation and Boolean Satisfiability

Springer

High Quality Test Pattern Generation and Boolean Satisfiability

Stephan Eggersglüß • Rolf Drechsler

High Quality Test Pattern Generation and Boolean Satisfiability

Springer

Stephan Eggersglüß
University of Bremen
Bremen, Germany

German Research Center for Artificial Intelligence (DFKI) – Cyber-Physical Systems, Bremen, Germany

Rolf Drechsler
University of Bremen
Bremen, Germany

German Research Center for Artificial Intelligence (DFKI) – Cyber-Physical Systems, Bremen, Germany

ISBN 978-1-4419-9975-7 e-ISBN 978-1-4419-9976-4
DOI 10.1007/978-1-4419-9976-4
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2012930124

© Springer Science+Business Media, LLC 2012
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, LLC, 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 on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The content of the book describes work that has been carried out in the Group of Computer Architecture at the University of Bremen, Germany over the last 5 years. Therefore, we would like to thank all the members of the group for their valuable help. Special thanks go to Daniel Tille and Görschwin Fey for many helpful discussions and support. Both are also co-authors of our previously published book *Test Pattern Generation using Boolean Proof Engines* which describes among other things the basics principles on which the content of this book is based on.

Various chapters of this book are based on scientific papers. Therefore, we would like to acknowledge the work of the co-authors of these papers Görschwin Fey, Hoang M. Le, Juergen Schloeffel and Daniel Tille. Since large parts of the work has been done in collaboration, our special thanks go to the Mentor Graphics Development group in Hamburg, Germany, especially to René Krenz-Bååth (now Hochschule Hamm-Lippstadt). Finally, we would like to thank Lisa Jungmann for her help with the cover design as well as Robert Wille, Judith End and Tom Gmeinder for proof-reading.

Parts of this research work were supported by the German Federal Ministry of Education and Research (BMBF) in the Project MAYA under contract number 01M3172B, by the German Research Foundation (DFG) under contract number DR 287/15-1 and by the Central Research Promotion (ZF) of the University of Bremen under contract number 03/107/05. The authors like to thank these institutions for their support.

Bremen

Stephan Eggersglüß
Rolf Drechsler

Contents

Part III High Quality Delay Test Generation

List of Figures

List of Tables

List of Acronyms

ARAP	As-Robust-As-Possible
ATE	Automatic Test Equipment
ATPG	Automatic Test Pattern Generation
BCP	Boolean Constraint Propagation
BDD	Binary Decision Diagram
CNF	Conjunctive Normal Form
CUT	Circuit Under Test
DCA	Dynamic Clause Activation
DFT	Design-For-Test
DTPG	Deterministic Test Pattern Generation
GDFM	Gate Delay Fault Model
IC	Integrated Circuit
IG	Implication Graph
ISAT	Incremental SAT
LC	Logic Class
LWL	Learned Watch List
ODC	Observability Don't Care
PBO	Pseudo-Boolean Optimization
PB-SAT	Pseudo-Boolean SAT
PDF	Path Delay Fault
PDFM	Path Delay Fault Model
PI	Primary Input
PO	Primary Output
PPI	Pseudo Primary Input
PPO	Pseudo Primary Output
RTPG	Random Test Pattern Generation
s-a-0	Stuck-at-0
s-a-1	Stuck-at-1
SAFM	Stuck-At Fault Model
SAT	Boolean Satisfiability, Satisfiable

SDD	Small Delay Defect
SWL	Structural Watch List
TF	Transition Fault
TFM	Transition Fault Model
UNSAT	Unsatisfiable

List of Symbols

$\downarrow$	falling transition
$\uparrow$	rising transition
$\cdot$	Boolean AND operator
$+$	Boolean OR operator
$\odot$	resolution operator
$\oplus$	Boolean XOR operator
$\bar{\cdot}$	Boolean NOT of $\cdot$
$\rightarrow$	implies
$\leftrightarrow$	equivalence
Φ	CNF, set of clauses
Φ_{dyn}	dynamically extended CNF
Φ_F	fault-specific constraints
Ψ	set of pseudo-Boolean constraints
ψ	pseudo-Boolean constraint of Ψ
η	Boolean encoding
κ	conflict
λ	arbitrary literal
ω	clause of Φ
ω_C	conflict clause
$@x$	at decision level x
$\mathbb{B}$	set of Boolean values $\{0,1\}$
$\mathscr{C}$	circuit
$\mathscr{F}$	set of flip-flops
F	fault
f	flip-flop $\in \mathscr{C}$, faulty line of gate
$\mathscr{F}(g)$	transitive fanin of g
$\mathscr{G}$	set of gates
g	gate $\in \mathscr{C}$
h	successor gate of g
$\mathscr{I}$	set of primary inputs
$i_1, \ldots, i_n$	primary inputs of $\mathscr{C}$

$\mathscr{J}$	J-stack
$\mathscr{L}_x$	multiple-valued logic with x values
$\mathscr{O}$	set of primary outputs
$o_1, \ldots, o_m$	primary outputs of $\mathscr{C}$
$\mathscr{P}$	structural path of $\mathscr{C}$
$\mathscr{S}$	set of signal lines or connections
s	signal line, connection
t_i	initial (current) time frame
t_{i+1}	final (next) time frame
V	vector
v	(Boolean) value
X	set of Boolean variables, don't care value
$x_1, \ldots, x_n$	Boolean variables

Chapter 1
Introduction

The *Integrated Circuit* (IC) was invented in the 1950s. At the beginning, ICs were mainly used in computers. With the advancing miniaturization of the components, the significance of ICs as part of our daily life grows. Many consumer products such as mobile music players or cell phones use ICs (or "chips") as core engines. A failure of these devices usually results in problems of lesser extent for the owner. But today, ICs also control safety critical applications. For example, chips are responsible for the correct mode of operation in car control systems, avionics or medical equipment. A failure of a chip can be life-threatening in the worst-case. Consequently, the correct mode of operation of a fabricated chip is crucial.

Due to the ever shrinking component sizes of today's designs, the vulnerability of chips to flaws in the manufacturing process increases. The IC manufacturers put much effort in guaranteeing the integrity of their products. A large part of the manufacturing costs is spent for the detection of defects caused by the manufacturing process. Every fabricated chip is subjected to a *post-production test* (or *manufacturing test*) to avoid that defective chips are delivered to customers (and by this could cause failures in operation mode). Thus, the purpose of such a post-production test is to detect any defects caused by the manufacturing process.

Stimuli are applied to the inputs of the *Circuit Under Test* (CUT) during this test. The output responses are monitored. If one or more of the output responses are inconsistent with the specification, the chip will be rejected as erroneous. However, the complexity of modern designs does not allow for a complete test of all possible input stimuli. A complete test for each fabricated chip would be far too time-consuming or costly, since the number of possible tests is exponential in the number of inputs. Instead, a test set is pre-computed that covers a large range of possible defects. Logical fault models are used to abstract from physical defects. The fault model most widespread is the *Stuck-at Fault Model* (SAFM) [Eld59].

This test set is applied to each fabricated chip by *Automatic Test Equipment* (ATE). Because the memory and bandwidth of an ATE is limited, the applied

S. Eggersglüß and R. Drechsler, *High Quality Test Pattern Generation and Boolean Satisfiability*, DOI 10.1007/978-1-4419-9976-4_1,
© Springer Science+Business Media, LLC 2012

test set has to be as small as possible. A large test set size signifies not only a long test application time but also immense test costs. The computation of the test set, which is known as *Automatic Test Pattern Generation* (ATPG), is the main subject of this book. Due to the large number of potential faults, ATPG is a computationally intensive task and fast algorithms are needed for obtaining a test set in acceptable run time.

Classical ATPG algorithms are mostly based on the D-algorithm proposed by Roth in 1966 [Rot66]. These algorithms work directly on the circuit structure, i.e. a flat gate-level netlist, and typically benefit significantly from their knowledge about the structure of the problem. Several techniques and powerful heuristics have been proposed over the years to improve the effectiveness of ATPG. At the turn of the millennium, the ATPG problem was considered to be solved. The existing algorithms were fast enough and provided sufficient fault coverage. This has changed in the last years.

The size of new designs doubles every 18 months according to Moore's law [Moo65]. Today's circuits consist of multi-million gates and the classical structural ATPG algorithms reach their limits. Tests for a large number of faults can still be generated very quickly. But the size of the set of faults for which no test can be generated in acceptable run time increases significantly. As a result, the high fault coverage demands of the chip manufacturers can barely be met. Thus, the overall quality of the test set decreases due to the lower fault coverage. As a consequence, there is a need for new robust ATPG algorithms especially when considering future design sizes.

Furthermore, another crucial point has been emerged in the field of manufacturing test. Due to the increased operation speed beyond the GHz mark and the small manufacturing technologies, the number of timing-related defects which affects the product quality has increased. This trend is expected to grow with the process technology scaling down towards very deep sub-micron devices. Therefore, *delay testing* has become mandatory to filter out defective devices and to assure that the desired performance specifications are met.

Test generation for delay faults generally needs more computational effort than ATPG for stuck-at faults because test patterns have to be computed over at least two time frames. A large number of delay faults remain unclassified during ATPG for modern designs. Moreover, in contrast to the SAFM, test patterns for delay faults can be classified in different quality levels like robust and non-robust test patterns [KC98]. The quality of a test can be – simply spoken – defined as the probability of fault detection. High quality test patterns are more desirable but usually harder to obtain. As a result, the need for new robust ATPG algorithms is even more urgent in the field of delay test generation. In the last years, *Small Delay Defects* (SDDs) have become more and more serious. Therefore, the ability to detect SDDs has become an important indicator for judging the quality of a delay test. However, ATPG approaches struggle with the generation of tests dedicated to detect SDDs due to the high complexity.

A promising solution to close the gap between test quality requirements and ATPG effectiveness is the application of solvers for *Boolean Satisfiability* (SAT).