

DIGITAL SYSTEMS TESTING AND TESTABLE DESIGN

Revised Printing

MIRON ABRAMOVICI, *AT&T Bell Laboratories, Murray Hill*
MELVIN A. BREUER, *University of Southern California, Los Angeles*
ARTHUR D. FRIEDMAN, *George Washington University*



The Institute of Electrical and Electronics Engineers, Inc., New York



A JOHN WILEY & SONS, INC., PUBLICATION

This is the IEEE revised printing of the book previously published by W. H. Freeman and Company in 1990 under the title *Digital Systems Testing and Testable Design*.

© 1990 by AT&T. All rights reserved.

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.
3 Park Avenue, 17th Floor, New York, NY 10016-5997

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning, or otherwise, except as permitted under Section 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923, 978-750-8400, fax 978-750-4470, or on the web at www.copyright.com. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, e-mail: permcoordinator@wiley.com.

For general information on our other products and services please contact our Customer Care Department within the U.S. at 877-762-2974, outside the U.S. at 317-572-3993 or fax 317-572-4002.

ISBN 0-7803-1062-4

Printed in the United States of America

10 9 8

To our families, who were not able to see their spouses and fathers for many evenings and weekends, and who grew tired of hearing "leave me alone, I'm working on the book." Thank you Gaby, Ayala, Orit, Sandy, Teri, Jeff, Barbara, Michael, and Steven. We love you.

CONTENTS

PREFACE	xiii
How This Book Was Written	xvii
1. INTRODUCTION	1
2. MODELING	9
2.1 Basic Concepts	9
2.2 Functional Modeling at the Logic Level	10
2.2.1 Truth Tables and Primitive Cubes	10
2.2.2 State Tables and Flow Tables	13
2.2.3 Binary Decision Diagrams	17
2.2.4 Programs as Functional Models	18
2.3 Functional Modeling at the Register Level	20
2.3.1 Basic RTL Constructs	20
2.3.2 Timing Modeling in RTLs	23
2.3.3 Internal RTL Models	24
2.4 Structural Models	24
2.4.1 External Representation	24
2.4.2 Structural Properties	26
2.4.3 Internal Representation	29
2.4.4 Wired Logic and Bidirectionality	31
2.5 Level of Modeling	32
REFERENCES	35
PROBLEMS	36
3. LOGIC SIMULATION	39
3.1 Applications	39
3.2 Problems in Simulation-Based Design Verification	41
3.3 Types of Simulation	42
3.4 The Unknown Logic Value	43
3.5 Compiled Simulation	46
3.6 Event-Driven Simulation	49
3.7 Delay Models	52
3.7.1 Delay Modeling for Gates	52
3.7.2 Delay Modeling for Functional Elements	54
3.7.3 Delay Modeling in RTLs	55
3.7.4 Other Aspects of Delay Modeling	55
3.8 Element Evaluation	56
3.9 Hazard Detection	60
3.10 Gate-Level Event-Driven Simulation	64

3.10.1	Transition-Independent Nominal Transport Delays	64
3.10.2	Other Logic Values	71
3.10.2.1	Tristate Logic	71
3.10.2.2	MOS Logic	72
3.10.3	Other Delay Models	74
3.10.3.1	Rise and Fall Delays	74
3.10.3.2	Inertial Delays	76
3.10.3.3	Ambiguous Delays	76
3.10.4	Oscillation Control	77
3.11	Simulation Engines	79
	REFERENCES	84
	PROBLEMS	86
4.	FAULT MODELING	93
4.1	Logical Fault Models	93
4.2	Fault Detection and Redundancy	95
4.2.1	Combinational Circuits	95
4.2.2	Sequential Circuits	103
4.3	Fault Equivalence and Fault Location	106
4.3.1	Combinational Circuits	106
4.3.2	Sequential Circuits	108
4.4	Fault Dominance	109
4.4.1	Combinational Circuits	109
4.4.2	Sequential Circuits	110
4.5	The Single Stuck-Fault Model	110
4.6	The Multiple Stuck-Fault Model	118
4.7	Stuck RTL Variables	122
4.8	Fault Variables	122
	REFERENCES	123
	PROBLEMS	126
5.	FAULT SIMULATION	131
5.1	Applications	131
5.2	General Fault Simulation Techniques	134
5.2.1	Serial Fault Simulation	134
5.2.2	Common Concepts and Terminology	134
5.2.3	Parallel Fault Simulation	135
5.2.4	Deductive Fault Simulation	139
5.2.4.1	Two-Valued Deductive Simulation	140
5.2.4.2	Three-Valued Deductive Simulation	145
5.2.5	Concurrent Fault Simulation	146
5.2.6	Comparison	154
5.3	Fault Simulation for Combinational Circuits	155
5.3.1	Parallel-Pattern Single-Fault Propagation	156
5.3.2	Critical Path Tracing	157
5.4	Fault Sampling	167

5.5	Statistical Fault Analysis	169
5.6	Concluding Remarks	172
	REFERENCES	173
	PROBLEMS	177
6.	TESTING FOR SINGLE STUCK FAULTS	181
6.1	Basic Issues	181
6.2	ATG for SSFs in Combinational Circuits	182
6.2.1	Fault-Oriented ATG	182
6.2.1.1	Common Concepts	189
6.2.1.2	Algorithms	196
6.2.1.3	Selection Criteria	213
6.2.2	Fault-Independent ATG	220
6.2.3	Random Test Generation	226
6.2.3.1	The Quality of a Random Test	227
6.2.3.2	The Length of a Random Test	227
6.2.3.3	Determining Detection Probabilities	229
6.2.3.4	RTG with Nonuniform Distributions	234
6.2.4	Combined Deterministic/Random TG	235
6.2.5	ATG Systems	240
6.2.6	Other TG Methods	246
6.3	ATG for SSFs in Sequential Circuits	249
6.3.1	TG Using Iterative Array Models	249
6.3.2	Simulation-Based TG	262
6.3.3	TG Using RTL Models	264
6.3.3.1	Extensions of the <i>D</i> -Algorithm	265
6.3.3.2	Heuristic State-Space Search	269
6.3.4	Random Test Generation	271
6.4	Concluding Remarks	272
	REFERENCES	274
	PROBLEMS	281
7.	TESTING FOR BRIDGING FAULTS	289
7.1	The Bridging-Fault Model	289
7.2	Detection of Nonfeedback Bridging Faults	292
7.3	Detection of Feedback Bridging Faults	294
7.4	Bridging Faults Simulation	297
7.5	Test Generation for Bridging Faults	301
7.6	Concluding Remarks	302
	REFERENCES	302
	PROBLEMS	302
8.	FUNCTIONAL TESTING	305
8.1	Basic Issues	305
8.2	Functional Testing Without Fault Models	305
8.2.1	Heuristic Methods	305

8.2.2	Functional Testing with Binary Decision Diagrams	309
8.3	Exhaustive and Pseudoexhaustive Testing	313
8.3.1	Combinational Circuits	313
8.3.1.1	Partial-Dependence Circuits	313
8.3.1.2	Partitioning Techniques	314
8.3.2	Sequential Circuits	315
8.3.3	Iterative Logic Arrays	317
8.4	Functional Testing with Specific Fault Models	323
8.4.1	Functional Fault Models	323
8.4.2	Fault Models for Microprocessors	325
8.4.2.1	Fault Model for the Register-Decoding Function	327
8.4.2.2	Fault Model for the Instruction-Decoding and Instruction-Sequencing Function	328
8.4.2.3	Fault Model for the Data-Storage Function	329
8.4.2.4	Fault Model for the Data-Transfer Function	329
8.4.2.5	Fault Model for the Data-Manipulation Function	329
8.4.3	Test Generation Procedures	330
8.4.3.1	Testing the Register-Decoding Function	330
8.4.3.2	Testing the Instruction-Decoding and Instruction-Sequencing Function	332
8.4.3.3	Testing the Data-Storage and Data-Transfer Functions	336
8.4.4	A Case Study	337
8.5	Concluding Remarks	337
	REFERENCES	338
	PROBLEMS	341
9.	DESIGN FOR TESTABILITY	343
9.1	Testability	343
9.1.1	Trade-Offs	344
9.1.2	Controllability and Observability	345
9.2	Ad Hoc Design for Testability Techniques	347
9.2.1	Test Points	347
9.2.2	Initialization	351
9.2.3	Monostable Multivibrators	351
9.2.4	Oscillators and Clocks	353
9.2.5	Partitioning Counters and Shift Registers	354
9.2.6	Partitioning of Large Combinational Circuits	355
9.2.7	Logical Redundancy	356
9.2.8	Global Feedback Paths	358
9.3	Controllability and Observability by Means of Scan Registers	358
9.3.1	Generic Boundary Scan	363
9.4	Generic Scan-Based Designs	364
9.4.1	Full Serial Integrated Scan	365

9.4.2	Isolated Serial Scan	366
9.4.3	Nonserial Scan	368
9.5	Storage Cells for Scan Designs	368
9.6	Classical Scan Designs	374
9.7	Scan Design Costs	382
9.8	Board-Level and System-Level DFT Approaches	382
9.8.1	System-Level Busses	383
9.8.2	System-Level Scan Paths	383
9.9	Some Advanced Scan Concepts	385
9.9.1	Multiple Test Session	385
9.9.2	Partial Scan Using I-Paths	386
9.9.3	BALLAST — A Structured Partial Scan Design	390
9.10	Boundary Scan Standards	395
9.10.1	Background	395
9.10.2	Boundary Scan Cell	398
9.10.3	Board and Chip Test Modes	399
9.10.4	The Test Bus	401
9.10.5	Test Bus Circuitry	402
9.10.5.1	The TAP Controller	402
9.10.5.2	Registers	407
	REFERENCES	408
	PROBLEMS	412
10.	COMPRESSION TECHNIQUES	421
10.1	General Aspects of Compression Techniques	421
10.2	Ones-Count Compression	423
10.3	Transition-Count Compression	425
10.4	Parity-Check Compression	428
10.5	Syndrome Testing	429
10.6	Signature Analysis	432
10.6.1	Theory and Operation of Linear Feedback Shift Registers	432
10.6.2	LFSRs Used as Signature Analyzers	441
10.6.3	Multiple-Input Signature Registers	445
10.7	Concluding Remarks	447
	REFERENCES	448
	PROBLEMS	452
11.	BUILT-IN SELF-TEST	457
11.1	Introduction to BIST Concepts	457
11.1.1	Hardcore	458
11.1.2	Levels of Test	459
11.2	Test-Pattern Generation for BIST	460
11.2.1	Exhaustive Testing	460
11.2.2	Pseudorandom Testing	460
11.2.3	Pseudoexhaustive Testing	461

11.2.3.1	Logical Segmentation	462
11.2.3.2	Constant-Weight Patterns	463
11.2.3.3	Identification of Test Signal Inputs	466
11.2.3.4	Test Pattern Generators for Pseudoexhaustive Tests	471
11.2.3.5	Physical Segmentation	476
11.3	Generic Off-Line BIST Architectures	477
11.4	Specific BIST Architectures	483
11.4.1	A Centralized and Separate Board-Level BIST Architecture (CSBL)	483
11.4.2	Built-In Evaluation and Self-Test (BEST)	483
11.4.3	Random-Test Socket (RTS)	484
11.4.4	LSSD On-Chip Self-Test (LOCST)	486
11.4.5	Self-Testing Using MISR and Parallel SRSG (STUMPS)	488
11.4.6	A Concurrent BIST Architecture (CBIST)	490
11.4.7	A Centralized and Embedded BIST Architecture with Boundary Scan (CEBS)	490
11.4.8	Random Test Data (RTD)	492
11.4.9	Simultaneous Self-Test (SST)	493
11.4.10	Cyclic Analysis Testing System (CATS)	495
11.4.11	Circular Self-Test Path (CSTP)	496
11.4.12	Built-In Logic-Block Observation (BILBO)	501
11.4.12.1	Case Study	510
11.4.13	Summary	513
11.5	Some Advanced BIST Concepts	514
11.5.1	Test Schedules	515
11.5.2	Control of BILBO Registers	517
11.5.3	Partial-Intrusion BIST	520
11.6	Design for Self-Test at Board Level	523
	REFERENCES	524
	PROBLEMS	532
12.	LOGIC-LEVEL DIAGNOSIS	541
12.1	Basic Concepts	541
12.2	Fault Dictionary	543
12.3	Guided-Probe Testing	549
12.4	Diagnosis by UUT Reduction	554
12.5	Fault Diagnosis for Combinational Circuits	556
12.6	Expert Systems for Diagnosis	557
12.7	Effect-Cause Analysis	559
12.8	Diagnostic Reasoning Based on Structure and Behavior	564
	REFERENCES	566
	PROBLEMS	568

13. SELF-CHECKING DESIGN	569
13.1 Basic Concepts	569
13.2 Application of Error-Detecting and Error-Correcting Codes	570
13.3 Multiple-Bit Errors	577
13.4 Checking Circuits and Self-Checking	578
13.5 Self-Checking Checkers	579
13.6 Parity-Check Function	580
13.7 Totally Self-Checking m/n Code Checkers	581
13.8 Totally Self-Checking Equality Checkers	584
13.9 Self-Checking Berger Code Checkers	584
13.10 Toward a General Theory of Self-Checking Combinational Circuits	585
13.11 Self-Checking Sequential Circuits	587
REFERENCES	589
PROBLEMS	590
14. PLA TESTING	593
14.1 Introduction	593
14.2 PLA Testing Problems	594
14.2.1 Fault Models	594
14.2.2 Problems with Traditional Test Generation Methods	597
14.3 Test Generation Algorithms for PLAs	597
14.3.1 Deterministic Test Generation	598
14.3.2 Semirandom Test Generation	599
14.4 Testable PLA Designs	600
14.4.1 Concurrent Testable PLAs with Special Coding	600
14.4.1.1 PLA with Concurrent Error Detection by a Series of Checkers	600
14.4.1.2 Concurrent Testable PLAs Using Modified Berger Code	601
14.4.2 Parity Testable PLAs	603
14.4.2.1 PLA with Universal Test Set	603
14.4.2.2 Autonomously Testable PLAs	605
14.4.2.3 A Built-In Self-Testable PLA Design with Cumulative Parity Comparison	606
14.4.3 Signature-Testable PLAs	608
14.4.3.1 PLA with Multiple Signature Analyzers	609
14.4.3.2 Self-Testable PLAs with Single Signature Analyzer	609
14.4.4 Partitioning and Testing of PLAs	610
14.4.4.1 PLA with BILBOs	611
14.4.4.2 Parallel-Testable PLAs	614
14.4.4.3 Divide-and-Conquer Strategy for Testable PLA Design	614
14.4.5 Fully-Testable PLA Designs	615
14.5 Evaluation of PLA Test Methodologies	618

14.5.1	Measures of TDMs	618
14.5.1.1	Resulting Effect on the Original Design	619
14.5.1.2	Requirements on Test Environment	619
14.5.2	Evaluation of PLA Test Techniques	620
REFERENCES		627
PROBLEMS		630
15.	SYSTEM-LEVEL DIAGNOSIS	633
15.1	A Simple Model of System-Level Diagnosis	633
15.2	Generalizations of the PMC Model	638
15.2.1	Generalizations of the System Diagnostic Graph	638
15.2.2	Generalization of Possible Test Outcomes	640
15.2.3	Generalization of Diagnosability Measures	641
REFERENCES		644
PROBLEMS		645
INDEX		647

PREFACE

This book provides a comprehensive and detailed treatment of digital systems testing and testable design. These subjects are increasingly important, as the cost of testing is becoming the major component of the manufacturing cost of a new product. Today, design and test are no longer separate issues. The emphasis on the quality of the shipped products, coupled with the growing complexity of VLSI designs, require testing issues to be considered early in the design process so that the design can be modified to simplify the testing process.

This book was designed for use as a text for graduate students, as a comprehensive reference for researchers, and as a source of information for engineers interested in test technology (chip and system designers, test engineers, CAD developers, etc.). To satisfy the different needs of its intended readership the book (1) covers thoroughly both the fundamental concepts and the latest advances in this rapidly changing field, (2) presents only theoretical material that supports practical applications, (3) provides extensive discussion of testable design techniques, and (4) examines many circuit structures used to realize built-in self-test and self-checking features.

Chapter 1 introduces the main concepts and the basic terminology used in testing. Modeling techniques are the subject of Chapter 2, which discusses functional and structural models for digital circuits and systems. Chapter 3 presents the use of logic simulation as a tool for design verification testing, and describes compiled and event-driven simulation algorithms, delay models, and hardware accelerators for simulation. Chapter 4 deals with representing physical faults by logical faults and explains the concepts of fault detection, redundancy, and the fault relations of equivalence and dominance. The most important fault model — the single stuck-fault model — is analyzed in detail. Chapter 5 examines fault simulation methods, starting with general techniques — serial, parallel, deductive, and concurrent — and continuing with techniques specialized for combinational circuits — parallel-pattern single-fault propagation and critical path tracing. Finally, it considers approximate methods such as fault sampling and statistical fault analysis.

Chapter 6 addresses the problem of test generation for single stuck faults. It first introduces general concepts common to most test generation algorithms, such as implication, sensitization, justification, decision tree, implicit enumeration, and backtracking. Then it discusses in detail several algorithms — the *D*-algorithm, the 9V-algorithm, PODEM, FAN, and critical path test generation — and some of the techniques used in TOPS, SOCRATES, RAPS, SMART, FAST, and the subscripted *D*-algorithm. Other topics include random test generation, test generation for sequential circuits, test generation using high-level models, and test generation systems.

Chapter 7 looks at bridging faults caused by shorts between normally unconnected signal lines. Although bridging faults are a "nonclassical" fault model, they are dealt with by simple extensions of the techniques used for single stuck faults. Chapter 8 is concerned with functional testing and describes heuristic methods, techniques using binary decision diagrams, exhaustive and pseudoexhaustive testing, and testing methods for microprocessors.

Chapter 9 presents design for testability techniques aimed at simplifying testing by modifying a design to improve the controllability and observability of its internal signals. The techniques analyzed are general ad hoc techniques, scan design, board and system-level approaches, partial scan and boundary scan (including the proposed JTAG/IEEE 1149.1 standard).

Chapter 10 is dedicated to compression techniques, which consider a compressed representation of the response of the circuit under test. The techniques examined are ones counting, transition counting, parity checking, syndrome checking, and signature analysis. Because of its widespread use, signature analysis is discussed in detail. The main application of compression techniques is in circuits featuring built-in self-test, where both the generation of input test patterns and the compression of the output response are done by circuitry embedded in the circuit under test. Chapter 11 analyzes many built-in self-test design techniques (CSBL, BEST, RTS, LOCST, STUMPS, CBIST, CEBS, RTD, SST, CATS, CSTP, and BILBO) and discusses several advanced concepts such as test schedules and partial intrusion built-in self-test.

Chapter 12 discusses logic-level diagnosis. The covered topics include the basic concepts in fault location, fault dictionaries, guided-probe testing, expert systems for diagnosis, effect-cause analysis, and a reasoning method using artificial intelligence concepts.

Chapter 13 presents self-checking circuits where faults are detected by a subcircuit called a checker. Self-checking circuits rely on the use of coded inputs. Some basic concepts of coding theory are first reviewed, followed by a discussion of specific codes — parity-check codes, Berger codes, and residue codes — and of designs of checkers for these codes.

Chapter 14 surveys the testing of programmable logic arrays (PLAs). First it reviews the fault models specific to PLAs and test generation methods for external testing of these faults. Then it describes and compares many built-in self-test design methods for PLAs.

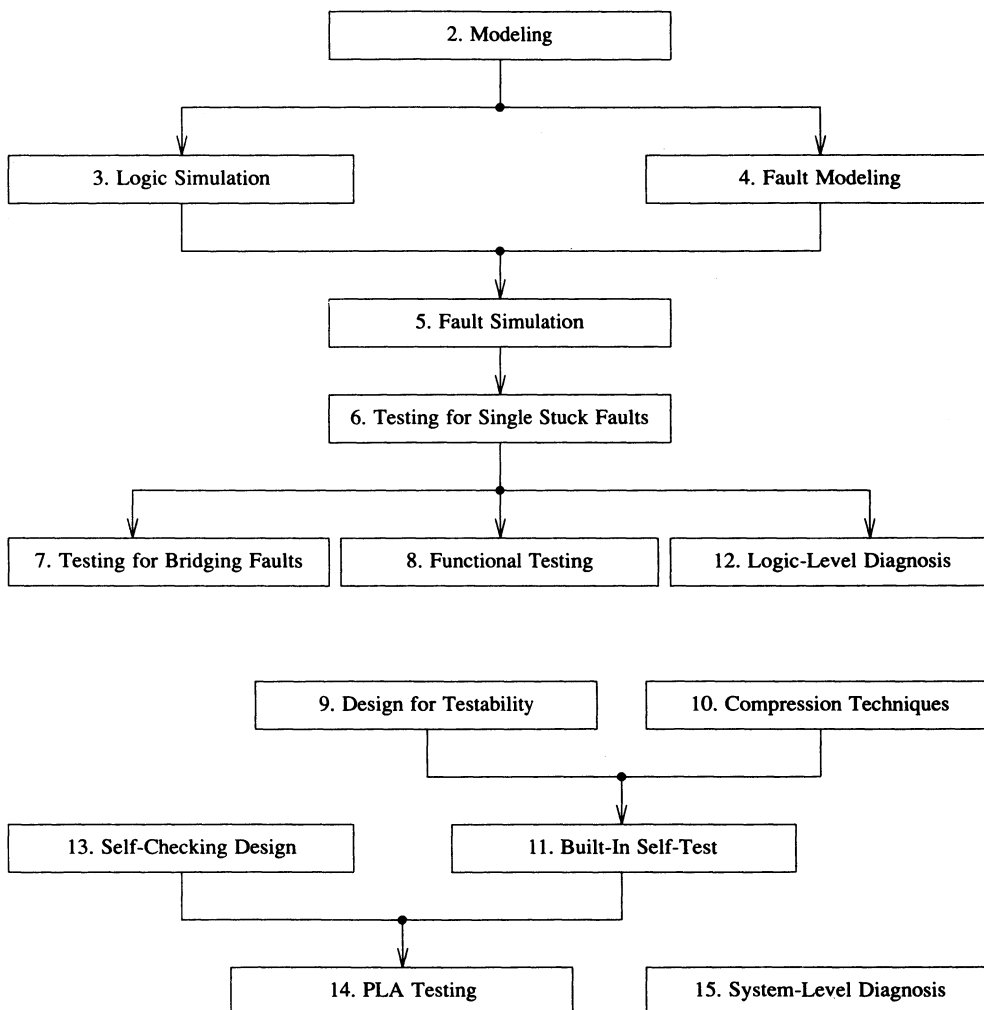
Chapter 15 deals with the problem of testing and diagnosis of a system composed of several independent processing elements (units), where one unit can test and diagnose other units. The focus is on the relation between the structure of the system and the levels of diagnosability that can be achieved.

In the Classroom

This book is designed as a text for graduate students in computer engineering, electrical engineering, or computer science. The book is self-contained, most topics being covered extensively, from fundamental concepts to advanced techniques. We assume that the students have had basic courses in logic design, computer science, and probability theory. Most algorithms are presented in the form of pseudocode in an easily understood format.

The progression of topics follows a logical sequence where most chapters rely on material presented in preceding chapters. The most important precedence relations among chapters are illustrated in the following diagram. For example, fault simulation (5) requires understanding of logic simulation (3) and fault modeling (4). Design for

testability (9) and compression techniques (10) are prerequisites for built-in self-test (11).



Precedence relations among chapters

The book requires a two-semester sequence, and even then some material may have to be glossed over. For a one-semester course, we suggest a "skinny path" through Chapters 1 through 6 and 9 through 11. The instructor can hope to cover only about half of this material. This "Introduction to Testing" course should emphasize the fundamental concepts, algorithms, and design techniques, and make only occasional forays into the more advanced topics. Among the subjects that could be skipped or only briefly discussed in the introductory course are Simulation Engines (Section 3.11),

The Multiple Stuck-Fault Model (4.6), Fault Sampling (5.4), Statistical Fault Analysis (5.5), Random Test Generation (6.2.3), Advanced Scan Techniques (9.9), and Advanced BIST Concepts (11.5). Most of the material in Chapter 2 and the ad hoc design for testability techniques (9.2) can be given as reading assignments.

Most of the topics not included in the introductory course can be covered in a second semester "Advanced Testing" course.

Acknowledgments

We have received considerable help in developing this book. We want to acknowledge Xi-An Zhu, who contributed to portions of the chapter on PLA testing. We are grateful for the support provided by the Bell Labs managers who made this work possible — John Bierbauer, Bill Evans, Al Fulton, Hao Nham, and Bob Taylor. Special thanks go to the Bell Labs word processing staff — Yvonne Anderson, Deborah Angell, Genevieve Przeor, and Lillian Pilz — for their superb job in producing this book, to David Hong for helping with troff and related topics, to Jim Coplien for providing the indexing software, and to John Pautler and Tim Norris for helping with the phototypesetter. We want to thank our many colleagues who have been using preprints of this book for several years and who have given us invaluable feedback, and especially S. Reddy, S. Seth, and G. Silberman. And finally, many thanks to our students at the University of Southern California and the Illinois Institute of Technology, who helped in "debugging" the preliminary versions.

Miron Abramovici
Melvin A. Breuer
Arthur D. Friedman

How This Book Was Written

Don't worry. We will not begin by saying that "because of the rapid increases in the complexity of VLSI circuitry, the issues of testing, design-for-test and built-in-self-test, are becoming increasingly more important." You have seen this type of opening a million times before. Instead, we will tell you a little of the background of this book.

The story started at the end of 1981. Miron, a young energetic researcher (at that time), noted that Breuer and Friedman's *Diagnosis & Reliable Design of Digital Systems* — known as the "yellow book" — was quickly becoming obsolete because of the rapid development of new techniques in testing. He suggested co-authoring a new book, using as much material from the yellow book as possible and updating it where necessary. He would do most of the writing, which Mel and Art would edit. It all sounded simple enough, and work began in early 1982.

Two years later, Miron had written less than one-third of the book. Most of the work turned out to be new writing rather than updating the yellow book. The subjects of modeling, simulation, fault modeling, fault simulation, and test generation were reorganized and greatly expanded, each being treated in a separate chapter. The end, however, was nowhere in sight. Late one night Miron, in a state of panic and frustration, and Mel, in a state of weakness, devised a new course of action. Miron would finalize the above topics and add new chapters on bridging faults testing, functional testing, logic-level diagnosis, delay-faults testing, and RAM testing. Mel would write the chapters dealing with new material, namely, PLA testing, MOS circuit testing, design for testability, compression techniques, and built-in self-test. And Art would update the material on self-checking circuits and system-level diagnosis.

The years went by, and so did the deadlines. Only Art completed his chapters on time. The book started to look like an encyclopedia, with the chapter on MOS testing growing into a book in itself. Trying to keep the material up to date was a continuous and endless struggle. As each year passed we came to dread the publication of another proceedings of the DAC, ITC, FTCS, or ICCAD, since we knew it would force us to go back and update many of the chapters that we had considered done.

Finally we acknowledged that our plan wasn't working and adopted a new course of action. Mel would set aside the MOS chapter and would concentrate on other, more essential chapters, leaving MOS for a future edition. Miron's chapters on delay fault testing and RAM testing would have the same fate. A new final completion date was set for January 1989.

This plan worked, though we missed our deadline by some 10 months. Out of love for our work and our profession, we have finally accomplished what we had set out to do. As this preface was being written, Miron called Mel to tell him about a paper he just read with some nice results on test generation. Yes, the book is obsolete already. If you are a young, energetic researcher — don't call us.