FAULT TOLERANT SYSTEMS
“Fault attacks have recently become a serious concern in the smart card industry. “Fault Tolerant Systems” provides the reader with a clear exposition of these attacks and the protection strategies that can be used to thwart them. A must read for practitioners and researchers working in the field.”

David Naccache, Ecole normale supérieure

“Understanding the fundamentals of an area, whether it is golf or fault tolerance, is a prerequisite to developing expertise in the area. Krishna and Koren’s book can provide a reader with this underlying foundation for fault tolerance. This book is particularly timely because the design of fault-tolerant computing components, such as processors and disks, is becoming increasingly important to the mainstream computing industry.”

Shubu Mukherjee, Director, FACT-AMI Group, Intel Corporation

“Professors Koren and Krishna, have written a modern, dual purpose text that first presents the basics fault tolerance tools describing various redundancy types both at the hardware and software levels followed by current research topics. It reviews fundamental reliability modeling approaches, combinatorial blocks and Markov chain techniques. Notably, there is a complete chapter on statistical simulation methods that offers guidance to practical evaluations as well as one on fault-tolerant networks. All chapters, which are clearly written including illuminating examples, have extensive reference lists whereby students can delve deeper into almost any topic. Several practical and commercial computing systems that incorporate fault tolerance are detailed. Furthermore, there are two chapters introducing current fault tolerance research challenges, cryptographic systems and defects in VLSI designs.”

Robert Redinbo, UC Davis

“The field of Fault-Tolerant Computing has advanced considerably in the past ten years and yet no effort has been made to put together these advances in the form of a book or a comprehensive paper for the students starting in this area. This is the first book I know of in the past 10 years that deals with hardware and software aspects of fault tolerant computing, is very comprehensive, and is written as a text for the course.”

Kewal Saluja, University of Wisconsin, Madison
FAULT TOLERANT SYSTEMS

Israel Koren
C. Mani Krishna
Contents

Foreword xi
Preface xiii
Acknowledgements xvii
About the Authors xix

1 Preliminaries 1
   1.1 Fault Classification 2
   1.2 Types of Redundancy 3
   1.3 Basic Measures of Fault Tolerance 4
       1.3.1 Traditional Measures 5
       1.3.2 Network Measures 6
   1.4 Outline of This Book 7
   1.5 Further Reading 9
   References 10

2 Hardware Fault Tolerance 11
   2.1 The Rate of Hardware Failures 11
   2.2 Failure Rate, Reliability, and Mean Time to Failure 13
   2.3 Canonical and Resilient Structures 15
       2.3.1 Series and Parallel Systems 16
       2.3.2 Non-Series/Parallel Systems 17
       2.3.3 M-of-N Systems 20
       2.3.4 Voters 23
       2.3.5 Variations on N-Modular Redundancy 23
       2.3.6 Duplex Systems 27
   2.4 Other Reliability Evaluation Techniques 30
       2.4.1 Poisson Processes 30
       2.4.2 Markov Models 33
3 Information Redundancy 55

3.1 Coding 56
   3.1.1 Parity Codes 57
   3.1.2 Checksum 64
   3.1.3 M-of-N Codes 65
   3.1.4 Berger Code 66
   3.1.5 Cyclic Codes 67
   3.1.6 Arithmetic Codes 74

3.2 Resilient Disk Systems 79
   3.2.1 RAID Level 1 79
   3.2.2 RAID Level 2 81
   3.2.3 RAID Level 3 82
   3.2.4 RAID Level 4 83
   3.2.5 RAID Level 5 84
   3.2.6 Modeling Correlated Failures 84

3.3 Data Replication 88
   3.3.1 Voting: Non-Hierarchical Organization 89
   3.3.2 Voting: Hierarchical Organization 95
   3.3.3 Primary-Backup Approach 96

3.4 Algorithm-Based Fault Tolerance 99

3.5 Further Reading 101
3.6 Exercises 102
References 106

4 Fault-Tolerant Networks 109

4.1 Measures of Resilience 110
   4.1.1 Graph-Theoretical Measures 110
   4.1.2 Computer Networks Measures 111

4.2 Common Network Topologies and Their Resilience 112
   4.2.1 Multistage and Extra-Stage Networks 112
   4.2.2 Crossbar Networks 119
   4.2.3 Rectangular Mesh and Interstitial Mesh 121
   4.2.4 Hypercube Network 124
5 Software Fault Tolerance  147
  5.1 Acceptance Tests  148
  5.2 Single-Version Fault Tolerance  149
      5.2.1 Wrappers  149
      5.2.2 Software Rejuvenation  152
      5.2.3 Data Diversity  155
      5.2.4 Software Implemented Hardware Fault Tolerance (SIHFT)  157
  5.3 N-Version Programming  160
      5.3.1 Consistent Comparison Problem  161
      5.3.2 Version Independence  162
  5.4 Recovery Block Approach  169
      5.4.1 Basic Principles  169
      5.4.2 Success Probability Calculation  169
      5.4.3 Distributed Recovery Blocks  171
  5.5 Preconditions, Postconditions, and Assertions  173
  5.6 Exception-Handling  173
      5.6.1 Requirements from Exception-Handlers  174
      5.6.2 Basics of Exceptions and Exception-Handling  175
      5.6.3 Language Support  177
  5.7 Software Reliability Models  178
      5.7.1 Jelinski–Moranda Model  178
      5.7.2 Littlewood–Verrall Model  179
      5.7.3 Musa–Okumoto Model  180
      5.7.4 Model Selection and Parameter Estimation  182
  5.8 Fault-Tolerant Remote Procedure Calls  182
      5.8.1 Primary-Backup Approach  182
      5.8.2 The Circus Approach  183
  5.9 Further Reading  184
  5.10 Exercises  186
  References  188
## 6 Checkpointing 193

6.1 What is Checkpointing? 195
   6.1.1 Why is Checkpointing Nontrivial? 197

6.2 Checkpoint Level 197

6.3 Optimal Checkpointing—An Analytical Model 198
   6.3.1 Time Between Checkpoints—A First-Order Approximation 200
   6.3.2 Optimal Checkpoint Placement 201
   6.3.3 Time Between Checkpoints—A More Accurate Model 202
   6.3.4 Reducing Overhead 204
   6.3.5 Reducing Latency 205

6.4 Cache-Aided Rollback Error Recovery (CARER) 206

6.5 Checkpointing in Distributed Systems 207
   6.5.1 The Domino Effect and Livellock 209
   6.5.2 A Coordinated Checkpointing Algorithm 210
   6.5.3 Time-Based Synchronization 211
   6.5.4 Diskless Checkpointing 212
   6.5.5 Message Logging 213

6.6 Checkpointing in Shared-Memory Systems 217
   6.6.1 Bus-Based Coherence Protocol 218
   6.6.2 Directory-Based Protocol 219

6.7 Checkpointing in Real-Time Systems 220

6.8 Other Uses of Checkpointing 223

6.9 Further Reading 223

6.10 Exercises 224

References 226

## 7 Case Studies 229

7.1 NonStop Systems 229
   7.1.1 Architecture 229
   7.1.2 Maintenance and Repair Aids 233
   7.1.3 Software 233
   7.1.4 Modifications to the NonStop Architecture 235

7.2 Stratus Systems 236

7.3 Cassini Command and Data Subsystem 238

7.4 IBM G5 241

7.5 IBM Sysplex 242

7.6 Itanium 244

7.7 Further Reading 246

References 247

## 8 Defect Tolerance in VLSI Circuits 249

8.1 Manufacturing Defects and Circuit Faults 249
8.2 Probability of Failure and Critical Area 251
8.3 Basic Yield Models 253
  8.3.1 The Poisson and Compound Poisson Yield Models 254
  8.3.2 Variations on the Simple Yield Models 256
8.4 Yield Enhancement Through Redundancy 258
  8.4.1 Yield Projection for Chips with Redundancy 259
  8.4.2 Memory Arrays with Redundancy 263
  8.4.3 Logic Integrated Circuits with Redundancy 270
  8.4.4 Modifying the Floorplan 272
8.5 Further Reading 276
8.6 Exercises 277
References 281

9 Fault Detection in Cryptographic Systems 285
9.1 Overview of Ciphers 286
  9.1.1 Symmetric Key Ciphers 286
  9.1.2 Public Key Ciphers 295
9.2 Security Attacks Through Fault Injection 296
  9.2.1 Fault Attacks on Symmetric Key Ciphers 297
  9.2.2 Fault Attacks on Public (Asymmetric) Key Ciphers 298
9.3 Countermeasures 299
  9.3.1 Spatial and Temporal Duplication 300
  9.3.2 Error-Detecting Codes 300
  9.3.3 Are These Countermeasures Sufficient? 304
  9.3.4 Final Comment 307
9.4 Further Reading 307
9.5 Exercises 307
References 308

10 Simulation Techniques 311
10.1 Writing a Simulation Program 311
10.2 Parameter Estimation 315
  10.2.1 Point Versus Interval Estimation 315
  10.2.2 Method of Moments 316
  10.2.3 Method of Maximum Likelihood 318
  10.2.4 The Bayesian Approach to Parameter Estimation 322
  10.2.5 Confidence Intervals 324
10.3 Variance Reduction Methods 328
  10.3.1 Antithetic Variables 328
  10.3.2 Using Control Variables 330
  10.3.3 Stratified Sampling 331
  10.3.4 Importance Sampling 333
10.4 Random Number Generation 341
   10.4.1 Uniformly Distributed Random Number Generators 342
   10.4.2 Testing Uniform Random Number Generators 345
   10.4.3 Generating Other Distributions 349
10.5 Fault Injection 355
   10.5.1 Types of Fault Injection Techniques 356
   10.5.2 Fault Injection Application and Tools 358
10.6 Further Reading 358
10.7 Exercises 359
References 363

Subject Index 365
Systems used in critical applications such as health, commerce, transportation, utilities, and national security must be highly reliable. Ubiquitous use of computing systems and other electronic systems in these critical areas requires that computing systems have high reliability. High reliability is achieved by designing the systems to be fault-tolerant. Even though the high reliability requirements of computing systems gave the original impetus to the study of the design of fault-tolerant systems, trends in manufacturing of VLSI circuits and systems are also requiring the use of fault-tolerant design methods to achieve high yields from manufacturing plants. This is due to the fact that with reduced feature sizes of VLSI circuit designs and shortcomings of lithographic techniques used in fabrication the characteristics of the manufactured devices are becoming unpredictable. Additionally small sizes of devices make them susceptible to radiation induced failures causing run time errors. Thus it may be necessary to use fault tolerance techniques even in systems that are used in non-critical applications such as consumer electronics.

This book covers comprehensively the design of fault-tolerant hardware and software, use of fault-tolerance techniques to improve manufacturing yields and design and analysis of networks. Additionally it includes material on methods to protect against threats to encryption subsystems used for security purposes. The material in the book will help immensely students and practitioners in electrical and computer engineering and computer science in learning how to design reliable computing systems and how to analyze fault-tolerant computing systems.

Sudhakar M. Reddy  
Distinguished Professor of Electrical and Computer Engineering  
University of Iowa Foundation  
Iowa City, Iowa
Preface

The purpose of this book is to provide a solid introduction to the rich field of fault-tolerant computing. Its intended use is as a text for senior-level undergraduate and first-year graduate students, as well as a reference for practicing engineers in the industry. Since it would be impossible to cover in one book all the fault-tolerance techniques and practices that have been developed or are currently in use, we have focused on providing the basics of the field and enough background to allow the reader to access more easily the rapidly expanding fault-tolerance literature. Readers who are interested in further details should consult the list of references at the end of each chapter. To understand this book well, the reader should have a basic knowledge of hardware design and organization, principles of software development, and probability theory.

The book has 10 chapters; each chapter has a list of relevant references and a set of exercises. Solutions to the exercises are available on-line and access to them is provided by the publisher upon request to instructors who adopt this book as a textbook for their class. Powerpoint slides for instructors are also available.

The book starts with an outline of preliminaries, in which we provide introductory information. This is followed by a set of six chapters that form the core of what we believe should be covered in any introduction to fault-tolerant systems.

Chapter 2 deals with hardware fault-tolerance; this is the discipline with the longest history (indeed, the idea of using hardware redundancy for fault-tolerance goes back to the very pioneers of computing, most notably von Neumann). We also include in this chapter an introduction to some of the probabilistic tools used in analyzing reliability measures.

Chapter 3 deals with information redundancy with the main focus on error detecting and correcting codes. Such codes, like hardware fault-tolerance, go back a very long way, and were motivated in large measure by the need to counter errors in information transmission. The same, or similar, techniques are being used today in other applications as well, principally in contemporary memory circuits. We have sought to provide a survey of only the more important coding techniques,
and it was not intended to be comprehensive: indeed, a comprehensive survey of coding would require multiple volumes. Following this, we turn to the issue of managing information redundancy in storage, and end with algorithm-based fault-tolerance.

Chapter 4 covers fault-tolerant networks. With processors becoming cheaper, distributed systems are becoming more commonplace; we look at some key network topologies and consider how to quantify and enhance their fault-tolerance.

Chapter 5 describes techniques for software fault-tolerance. It is widely believed that software accounts for a majority of the failures seen in today’s computer systems. As a field, software fault-tolerance is less mature than fault-tolerance using either hardware or information redundancy. It is also a much harder nut to crack. Software is probably the most complex artificial construct that people have created, and rendering it fault-tolerant is an arduous task. We cover such techniques as recovery blocks and N-version programming, together with a discussion of acceptance tests and ways to model software failure processes analytically.

In Chapter 6, we cover the use of time-redundancy through checkpointing. The majority of hardware failures are transient in nature; in other words, they are failures which simply go away after some time. An obvious response to such failures is to roll back the execution and re-execute the program. Checkpointing is a technique which allows us to limit the extent of such re-executions.

Chapter 7, which contains several case studies, rounds off the core of the book. There, we describe several real-life examples of fault-tolerant systems illustrating the usage of the various techniques presented in the previous chapters.

The remaining three chapters of the book deal with more specialized topics. In Chapter 8, we cover defect-tolerance in VLSI. As die sizes increase and feature sizes drop, it is becoming increasingly important to be able to tolerate manufacturing defects in a VLSI chip without affecting its functionality. We discuss the key approaches being used, as well as the underlying mathematical models.

In Chapter 9, we focus on cryptographic devices. The increasing use of computers in commerce, including smart cards and Internet shopping, has motivated the use of encryption in everyday applications. It turns out that injecting faults into cryptographic devices and observing the outputs is an effective way to attack secure systems and obtain their secret key. We present in this chapter the use of fault-detection to counter these types of security attacks.

Chapter 10, which ends the book, deals with simulation and experimental techniques. Simulating a fault-tolerant system to measure its reliability is often computationally very demanding. We provide in this chapter an outline of basic simulation techniques, as well as ways in which simulation can be accelerated. Also provided are basic statistical tools by which simulation output can be analyzed and an outline of experimental fault-injection techniques.

A companion web site (www.ecs.umass.edu/ece/koren/FaultTolerantSystems/) includes additional resources for the book such as lecture slides, the inevitable list of errors, and, more importantly, a link to an extensive collection of
educational tools and simulators that can be of great assistance to the readers of the book. Elsevier also maintains an instructor web site that will house the solutions for those who adopt this book as a textbook for their class. The website can be found at http://textbooks.elsevier.com.
Acknowledgements

Many people have assisted us in putting this book together. Pride of place in these acknowledgments must go to Zahava Koren, who read through the manuscript in detail and provided many incisive comments. While the authors are responsible for any errors that still remain in this book, she is responsible for the absence of very many that do not. We also had very valuable feedback from the reviewers of this manuscript. Some of them chose to remain anonymous, so we cannot thank them individually. However, those who can be named are: Wendy Bartlett from HP Labs, Doug Blough from Georgia Institute of Technology, Mark Karpovski from Boston University, Cetin Kaya Koc from Oregon State University, Shubu Mukherjee from Intel, David Naccache from École normale supérieure, Nohpill Park from Oklahoma State University, Irith Pomeranz from Purdue University, Mihaela Radu from Rose Hulman Institute of Technology, Robert Redinbo from UC Davis, Kewal Saluja from University of Wisconsin at Madison, Jean-Pierre Seifert from Applied Security Research Group, Arun Somani from Iowa State University, and Charles Weinstock from Carnegie Mellon University.

We would like to thank the staff at Morgan Kaufman for their efforts on behalf of this project. In particular, Denise Penrose and Kim Honjo spent many hours in meetings and discussions with us on many issues ranging from the technical content of this book to its look and feel.
Israel Koren is a Professor of Electrical and Computer Engineering at the University of Massachusetts, Amherst. Previously, he held positions with the University of California at Santa Barbara, the University of Southern California at Los Angeles, the Technion at Haifa, Israel, and the University of California at Berkeley. He received a BSc (1967), an MSc (1970), and a DSc (1975) in electrical engineering from the Technion in Haifa, Israel. His research interests include fault-tolerant systems, VLSI yield and reliability, secure cryptographic systems, and computer arithmetic. He publishes extensively and has over 200 publications in refereed journals and conferences. He is an Associate Editor of the IEEE Transactions on VLSI Systems, the VLSI Design Journal, and the IEEE Computer Architecture Letters. He served as General Chair, Program Chair and Program Committee member for numerous conferences. He is the author of the textbook Computer Arithmetic Algorithms, 2nd edition, A.K. Peters, Ltd., 2002, and an editor and co-author of Defect and Fault-Tolerance in VLSI Systems, Plenum, 1989. Dr. Koren is a fellow of the IEEE Computer Society.

C. Mani Krishna is a Professor of Electrical and Computer Engineering at the University of Massachusetts, Amherst. He received his PhD in Electrical Engineering from the University of Michigan in 1984. He previously received a BTech in Electrical Engineering from the Indian Institute of Technology, Delhi, in 1979, and an MS from the Rensselaer Polytechnic Institute in Troy, NY, in 1980. Since 1984, he has been on the faculty of the Department of Electrical and Computer Engineering at the University of Massachusetts at Amherst. He has carried out research in a number of areas: real-time, fault-tolerant, and distributed systems, sensor networks, and performance evaluation of computer systems. He coauthored a book, Real-Time Systems, McGraw-Hill, 1997, with Kang G. Shin. He has also been an editor on volumes of readings in performance evaluation and real-time systems, and for special issues on real-time systems of IEEE Computer and the Proceedings of the IEEE.