- Foreword p. xv
- Preface p. xix
- 1 Introduction p. 1
- 1.1 Register Transfer Level p. 2
- 1.1.1 What is It? p. 2
- 1.1.2 Verifiable RTL p. 3
- 1.1.3 Applying Design Discipline p. 4
- 1.2 Assumptions p. 4
- 1.3 Organization of This Book p. 5
- 2 The Verification Process p. 9
- 2.1 Specification Design Decomposition p. 10
- 2.1.1 High-Level Design Requirements p. 11
- 2.1.2 Block-Level Specification and Design p. 13
- 2.1.3 RTL Implementation p. 13
- 2.1.4 Synthesis and Physical Design p. 13
- 2.2 Functional Test Strategies p. 14
- 2.2.1 Deterministic or Directed Test p. 15
- 2.2.2 Random Test p. 16
- 2.2.3 Transaction Analyzer Verification p. 18
- 2.2.4 Chip Initialization Verification p. 19
- 2.2.5 Synthesizable Testbench p. 20
- 2.3 Transformation Test Strategies p. 20
- 2.4 Summary p. 21
- 3 Coverage, Events and Assertions p. 23
- 3.1 Coverage p. 24
- 3.1.1 Ad-hoc Metrics p. 25
- 3.1.2 Programming Code Metrics p. 25
- 3.1.3 State Machine and Arc Coverage Metrics p. 27
- 3.1.4 User Defined Metrics p. 27
- 3.1.5 Fault Coverage Metrics p. 27
- 3.1.6 Regression Analysis and Test Suite Optimization p. 28
- 3.2 Event Monitors and Assertion Checkers p. 28
- 3.2.1 Events p. 29
- 3.2.2 Assertions p. 31
- 3.2.3 Assertion Monitor Library Details p. 36
- 3.2.4 Event Monitor and Assertion Checker Methodology p. 37
- 3.2.4.1 Linting Strategy p. 39
- 3.2.4.2 Implementation Considerations p. 39
- 3.2.4.3 Event Monitor Database and Analysis p. 40
- 3.3 Summary p. 41
- 4 RTL Methodology Basics p. 43
- 4.1 Simple RTL Verifiable Subset p. 44
- 4.2 Linting p. 48
- 4.2.1 Linting in a design project p. 49

- 4.2.2 Lint description p. 50
- 4.2.2.1 Project Oriented p. 50
- 4.2.2.2 Linting Message Examples p. 51
- 4.3 Object-Based Hardware Design p. 53
- 4.3.1 OBHD and Simulation p. 56
- 4.3.2 OBHD and Formal Verification p. 59
- 4.3.3 OBHD and Physical Design p. 60
- 4.3.3.1 OBHD Synthesis p. 60
- 4.3.3.2 OBHD Scan Chain Hookup p. 62
- 4.3.4 A Text Macro Implementation p. 64
- 4.4 Summary p. 67
- 5 RTL Logic Simulation p. 69
- 5.1 Simulation History p. 71
- 5.1.1 First Steps p. 71
- 5.1.2 X, Z and Other States p. 73
- 5.1.3 Function and Timing p. 73
- 5.1.4 Gate to RTL Migration p. 74
- 5.1.5 Acceleration and Emulation p. 74
- 5.1.6 Language Standardization p. 76
- 5.2 Project Simulation Phases p. 78
- 5.2.1 Debugging Phase p. 79
- 5.2.2 Performance Profiling Phase p. 80
- 5.2.3 Regression Phase p. 81
- 5.2.4 Recreating Hardware Problems p. 82
- 5.2.5 Performance p. 82
- 5.3 Operation p. 83
- 5.3.1 Sequencing p. 84
- 5.3.1.1 Event-Driven p. 84
- 5.3.1.2 Rank-Ordered p. 86
- 5.3.2 Evaluation p. 88
- 5.3.2.1 Interpreted p. 88
- 5.3.2.2 Compiled code p. 88
- 5.3.2.3 RTL Methods p. 89
- 5.4 Optimizations p. 89
- 5.4.1 Flattening p. 90
- 5.4.2 Rank-Ordering p. 90
- 5.4.3 Bus Reconstruction p. 90
- 5.4.3.1 Concatenation p. 91
- 5.4.3.2 Expression Simplification p. 91
- 5.4.4 OBHD-based Optimization p. 92
- 5.4.4.1 Common Sub-expression Consolidation p. 92
- 5.4.4.2 Common if-else Control Consolidation p. 92
- 5.4.5 Partitioning p. 92
- 5.4.5.1 Branch Partitioning p. 93

- 5.4.5.2 Clock Partitioning p. 94
- 5.4.5.3 Chip Partitioning p. 94
- 5.5 Random Two-State Simulation Methods p. 95
- 5.5.1 Start Up State p. 95
- 5.5.1.1 Design Method p. 95
- 5.5.1.2 Zero Initialization p. 96
- 5.5.1.3 Random Initialization p. 97
- 5.5.1.4 Verification Test p. 98
- 5.5.2 Tri-state Buses p. 98
- 5.5.3 Assertion Monitors p. 99
- 5.5.4 Two-State in the Design Process p. 100
- 5.6 Summary p. 100
- 6 RTL Formal Verification p. 103
- 6.1 Formal Verification Introduction p. 106
- 6.2 Finite State Machines p. 107
- 6.2.1 Machine Equivalence p. 110
- 6.2.2 FSM Property Verification p. 111
- 6.3 Formal Transformation Verification p. 112
- 6.3.1 Equivalence Checking p. 112
- 6.3.1.1 Equivalence Checking Flow p. 112
- 6.3.1.2 Closing the Verification Loop p. 114
- 6.3.2 Cutpoint Definition p. 115
- 6.3.3 Equivalence Checking RTL Coding Style p. 117
- 6.3.3.1 Isolating Functional Complexity p. 117
- 6.3.3.2 Test Expressions within Case Statements p. 118
- 6.3.3.3 Equivalence Checking OBHD Practices p. 118
- 6.3.4 RTL Static Sign-off p. 120
- 6.3.5 Effective Equivalence Checking Methodology p. 122
- 6.4 Formal Functional Verification p. 124
- 6.4.1 Specification p. 124
- 6.4.2 Model Checking and Parameterized Modules p. 126
- 6.4.3 Model Checking OBHD Practices p. 127
- 6.5 Summary p. 128
- 7 Verifiable RTL Style p. 131
- 7.1 Design Content p. 132
- 7.1.1 Asynchronous Logic p. 132
- 7.1.2 RTL Races p. 133
- 7.1.3 Combinational Feedback p. 135
- 7.1.4 Case Statements p. 137
- 7.1.4.1 Fully-Specified case Statements p. 138
- 7.1.4.2 Test Signal and Constant Widths p. 143
- 7.1.5 Tri-State Buses p. 144
- 7.2 Organization p. 146
- 7.2.1 System Organization p. 146

- 7.2.1.1 Compiler Options p. 146
- 7.2.1.2 Design Hierarchy p. 148
- 7.2.1.3 Files p. 149
- 7.2.1.4 Libraries p. 150
- 7.2.2 Module Organization p. 151
- 7.2.2.1 Overall Organization p. 151
- 7.2.2.2 Connections p. 153
- 7.2.3 Expression Organization p. 153
- 7.3 Naming Conventions p. 155
- 7.3.1 General Considerations p. 155
- 7.3.1.1 Consistency p. 155
- 7.3.1.2 Upper/Lower Case p. 156
- 7.3.1.3 Hierarchical Name References p. 157
- 7.3.1.4 Global/Local Name Space p. 158
- 7.3.1.5 Profiling Support p. 159
- 7.3.2 Specific Naming Conventions p. 161
- 7.3.2.1 Constants p. 161
- 7.3.2.2 File Names p. 161
- 7.3.2.3 Instances p. 161
- 7.3.2.4 Modules p. 162
- 7.3.2.5 Port Names p. 163
- 7.3.2.6 Signal Names p. 164
- 7.3.2.7 User Tasks / Functions and Program Libraries p. 165
- 7.4 Naming In Verilog Library Modules p. 166
- 7.5 Editing Practices p. 166
- 7.5.1 Indentation p. 167
- 7.5.2 Comments p. 167
- 7.5.2.1 Header p. 168
- 7.5.2.2 Declarations p. 168
- 7.5.2.3 end Identification p. 169
- 7.5.2.4 Tool Controls p. 169
- 7.5.2.5 Embedded Comments p. 171
- 7.5.3 Line Length p. 171
- 7.6 Summary p. 172
- 8 The Bad Stuff p. 173
- 8.1 In-Line Storage Element Specification p. 174
- 8.2 RTL X State p. 175
- 8.2.1 RTL X-State Problems p. 176
- 8.2.1.1 RTL X-State Pessimism p. 176
- 8.2.1.2 RTL X-State Optimism p. 177
- 8.2.1.3 Impractical p. 177
- 8.3 Visits p. 180
- 8.3.1 Bit Visits p. 180
- 8.3.2 Configuration Test Visits p. 181

- 8.3.3 for Loops p. 182
- 8.4 Simulation vs. Synthesis Differences p. 184
- 8.4.1 Explicit Differences p. 185
- 8.4.1.1 Full and Parallel Case p. 185
- 8.4.1.2 X Assignment p. 188
- 8.4.1.3 Other Forms of State Machines p. 189
- 8.4.1.4 Initial blocks p. 190
- 8.4.2 Inadvertent Coding Errors p. 191
- 8.4.2.1 Incomplete Sensitivity List p. 191
- 8.4.2.2 Latch Inference in functions p. 192
- 8.4.2.3 Incorrect Procedural Statement Ordering p. 192
- 8.4.3 Timing p. 193
- 8.4.3.1 Delays p. 193
- 8.4.3.2 Race Conditions p. 196
- 8.5 Problematic RTL Verilog p. 198
- 8.5.1 Linting and Problematic RTL Verilog p. 198
- 8.5.2 Simulation and Problematic RTL Verilog p. 199
- 8.5.3 Formal Verification and Problematic Verilog p. 199
- 8.6 EDA Tool Vendors p. 200
- 8.6.1 Tool Library Function Naming p. 201
- 8.6.2 Command Line Consistency p. 202
- 8.6.3 Vendor Specific Properties p. 203
- 8.7 Design Team Discipline p. 204
- 8.8 Language Elements p. 205
- 8.8.1 Keywords p. 205
- 8.8.2 Parameters p. 205
- 8.8.3 User-Defined Primitives p. 206
- 8.9 Summary p. 207
- 9 Verifiable RTL Tutorial p. 209
- 9.1 Module p. 210
- 9.1.1 Specification p. 210
- 9.1.2 Comments p. 211
- 9.1.3 Instantiation p. 212
- 9.1.4 Interconnection p. 212
- 9.2 Adding Behavior p. 215
- 9.3 Multi-bit Interconnect and Behavior p. 216
- 9.4 Parameters p. 217
- 9.5 Expressions p. 218
- 9.5.1 Operators p. 218
- 9.5.1.1 Binary operators p. 218
- 9.5.1.2 Unary operators p. 219
- 9.5.1.3 Miscellaneous operators p. 220
- 9.5.2 Operator precedence p. 221
- 9.6 Procedural Blocks p. 222

- 9.6.1 Combinational Logic p. 222
- 9.6.1.1 Procedural Assignments p. 222
- 9.6.1.2 Functions p. 224
- 9.6.1.3 If-else Statement p. 225
- 9.6.1.4 Case, casex Statements p. 225
- 9.6.2 Storage Elements p. 227
- 9.6.2.1 Flip-flops p. 227
- 9.6.2.2 Latches p. 228
- 9.6.2.3 Memories p. 228
- 9.6.3 Debugging p. 229
- 9.6.3.1 $Display and $write Statements p. 230
- 9.6.3.2 $Finish p. 231
- 9.7 Testbench p. 231
- 9.8 Verilog Compilation p. 235
- 9.8.1 Compiler directives p. 235
- 9.8.1.1 Constants p. 235
- 9.8.1.2 Code Inclusion p. 236
- 9.8.1.3 Command Line p. 237
- 9.9 Summary p. 238
- 10 Principles of Verifiable RTL Design p. 239
- 10.1 Principles p. 239
- 10.1.1 Disciplined User Principle p. 240
- 10.1.2 Fundamental Verification Principle p. 240
- 10.1.3 Retain Useful Information Principle p. 240
- 10.1.4 Orthogonal Verification Principle p. 240
- 10.1.5 Functional Observation Principle p. 241
- 10.1.6 Verifiable Subset Principle p. 241
- 10.1.7 Project Linting Principle p. 241
- 10.1.8 Object-Based Hardware Design Principle p. 242
- 10.1.9 Fast Simulation Principle p. 242
- 10.1.10 Visit Minimization Principle p. 242
- 10.1.11 Cutpoint Identification Principle p. 242
- 10.1.12 Numeric Value Parameterization Principle p. 243
- 10.1.13 Consistency Principle p. 243
- 10.1.14 Asynchronous Principle p. 244
- 10.1.15 Combinational Feedback Principle p. 244
- 10.1.16 Property Principle p. 244
- 10.1.17 Faithful Semantics Principle p. 244
- 10.1.18 Good Vendor Principle p. 245
- 10.2 Summary p. 245
- Bibliography p. 247
- A. Comparing Verilog Construct Performance p. 255
- B. Quick Reference p. 259
- C. Assertion Monitors p. 265