Contents

LIST OF FIGURES .... **XIX**

LIST OF TABLES .... **XXVII**

1 DESIGN VERIFICATION CHALLENGES **1**

1.1 INTRODUCTION 1
1.2 SIMULATION-BASED VERIFICATION 1
1.3 FORMAL VERIFICATION 2
1.3.1 Model Checking 3
1.4 OVERVIEW 5
1.5 VERIFICATION TASKS 6
1.6 VERIFICATION CHALLENGES 8
1.6.1 Design Features 8
1.6.2 Verification Techniques 9
1.6.3 Verification Methodology 11
1.7 ORGANIZATION OF BOOK 13

2 BACKGROUND **17**

2.1 MODEL CHECKING 17
2.1.1 Correctness Properties 18
2.1.2 Explicit Model Checking 19
2.1.3 Symbolic Model Checking 19
2.2 NOTATIONS 20
2.3 BINARY DECISION DIAGRAMS 22
2.4 BOOLEAN SATISFIABILITY PROBLEM 23

2.4.1 Decision Engine 25
2.4.2 Deduction Engine 26
2.4.3 Diagnosis Engine 28
2.4.4 Proof of Unsatisfiability 29
2.4.5 Further Improvements 30
2.5 SAT-based Bounded Model Checking (BMC) 32
2.5.1 BMC formulation: Safety and Liveness Properties 33
2.5.2 Clocked LTL Specifications 36
2.6 SAT-based Unbounded Model Checking 37
2.7 SMT-based BMC 39
2.8 Notes 40

PART I: BASIC INFRASTRUCTURE 41

3 EFFICIENT BOOLEAN REPRESENTATION 43
3.1 Introduction 43
3.2 Brief Survey of Boolean Representations 45
3.2.1 Extended Boolean Decision Diagrams (XBDDs) 45
3.2.2 Boolean Expression Diagrams (BEDs) 45
3.2.3 AND/INVERTER Graph (AIG) 46
3.3 Functional Hashing (Reduced AIG) 49
3.3.1 Three-Input Case 50
3.3.2 Four-Input Case 52
3.3.3 Example 54
3.4 Experiments 57
3.5 Simplification using External Constraints 60
3.6 Comparing Functional Hashing with BDD/SAT Sweeping 61
3.7 Summary 62
3.8 Notes 62

4 HYBRID DPLL-STYLE SAT SOLVER 63
4.1 Introduction 63
4.2 BCP on Circuit 65
4.2.1 Comparing CNF- and Circuit-based BCP Algorithms 67
4.3 Hybrid SAT Solver 68
4.3.1 Proof of Unsatisfiability 69
4.3.2 Comparison with Chaff 69
4.4 Applying Circuit-based Heuristics 71
4.4.1 Justification Frontier Heuristics 71
4.4.2 Implication Order 72
4.4.3 Gate Fanout Count 73
4.4.4 Learning XOR/MUX Gates 74
4.5 Verification Applications of Hybrid SAT Solver 75

4.6 Summary 75
4.7 Notes 76

PART II: FALSIFICATION 77

5 SAT-BASED BOUNDED MODEL CHECKING 79
5.1 Introduction 79
5.2 Dynamic Circuit Simplification 81
5.2.1 Notation 82
5.2.2 Procedure Unroll 83
5.2.3 Comparing Implicit with Explicit Unrolling 84
5.3 SAT-based Incremental Learning and Simplification 86
5.4 BDD-based Learning 90
5.4.1 Basic Idea 90
5.4.2 Procedure: BDD_learning_engine 91
5.4.3 Seed Selection 92
5.4.4 Creation of BDDs 93
5.4.5 Generation of Learned Clauses 94
5.4.6 Integrating BDD Learning with a Hybrid SAT Solver 95
5.4.7 Adding Clauses Dynamically to a SAT Solver 95
5.4.8 Heuristics for Adding Learned Clauses 96
5.4.9 Application of BDD-based Learning 97
5.5 Customized Property Translation 98
5.5.1 Customized Translation for *F(p)* 100
5.5.2 Customized Translation of *G(q)* 102
5.5.3 Customized Translation of *F(p∧G(q))* 103
5.6 Experiments 104
5.6.1 Comparative Study of Various Techniques 105
5.6.2 Effect of Customized Translation and Incremental Learning 108
5.6.3 Effect of BDD-based Learning on BMC 109
5.6.4 Static BDD Learning 109
5.6.5 Dynamic BDD Learning 110
5.7 Summary 112
5.8 Notes 112

6 DISTRIBUTED SAT-BASED BMC 113
6.1 Introduction 113
6.2 Distributed SAT-based BMC Procedure 114
6.3 Topology-cognizant Distributed-BCP 116
6.3.1 Causal-effect Order 117
6.4 Distributed-SAT 118
6.4.1 Tasks of the Master 119
6.4.2 Tasks of a Client C_i 120

6.5 SAT-BASED DISTRIBUTED-BMC 120
6.6 OPTIMIZATIONS 121
6.6.1 Memory Optimizations in Distributed-SAT 121
6.6.2 Tight Estimation of Communication Overhead 121
6.6.3 Performance Optimizations in Distributed-SAT 123
6.6.4 Performance Optimization in SAT-based Distributed-BMC 124
6.7 EXPERIMENTS 124
6.8 RELATED WORK 128
6.9 SUMMARY 129
6.10 NOTES 129

7 EFFICIENT MEMORY MODELING IN BMC 131

7.1 INTRODUCTION 131
7.2 BASIC IDEA 132
7.3 MEMORY SEMANTICS 134
7.4 EMM APPROACH 135
7.4.1 Efficient Representation of Memory Modeling Constraints 136
7.4.2 Comparison with ITE Representation 139
7.4.3 Non-uniform Initialization of Memory 140
7.4.4 EMM for Multiple Memories, Read, and Write Ports 141
7.4.5 Arbitrary Initial Memory State 143
7.5 EXPERIMENTS ON A SINGLE READ/WRITE PORT MEMORY 144
7.6 EXPERIMENTS ON MULTI-PORT MEMORIES 149
7.6.1 Case Study on Quick Sort 150
7.6.2 Case Study on Industry Design (Low Pass Filter) 151
7.7 RELATED WORK 151
7.8 SUMMARY 152
7.9 NOTES 153

8 BMC FOR MULTI-CLOCK SYSTEMS 155

8.1 INTRODUCTION 155
8.1.1 Nested Clock Specifications 155
8.1.2 Verification Model for Multi-clock Systems 156
8.1.3 Simplification of Verification Model 156
8.1.4 Clock Specification on Latches 157
8.2 EFFICIENT MODELING OF MULTI-CLOCK SYSTEMS 158
8.3 REDUCING UNROLLING IN BMC 160
8.4 REDUCING LOOP-CHECKS IN BMC 161
8.5 DYNAMIC SIMPLIFICATION IN BMC 162
8.6 CUSTOMIZATION OF CLOCKED SPECIFICATIONS IN BMC 163
8.7 EXPERIMENTS 166
8.7.1 VGA/LCD Controller 167
8.7.2 Tri-mode Ethernet MAC Controller 168

8.8 RELATED WORK 169
8.9 SUMMARY 170
8.10 NOTES 171

PART III: PROOF METHODS 173

9 PROOF BY INDUCTION 175

9.1 INTRODUCTION 175
9.2 BMC PROCEDURE FOR PROOF BY INDUCTION 176
9.3 INDUCTIVE INVARIANTS: REACHABILITY CONSTRAINTS 177
9.4 PROOF OF INDUCTION WITH EMM 179
9.5 EXPERIMENTS 180
9.5.1 Use of Reachability Invaraints 180
9.5.2 Case Study: Use of Induction proof with EMM 181
9.6 SUMMARY 182
9.7 NOTES 183

10 UNBOUNDED MODEL CHECKING 185

10.1 INTRODUCTION 185
10.2 MOTIVATION 187
10.3 CIRCUIT COFACTORING APPROACH 188
10.3.1 Basic Idea 188
10.3.2 The Procedure 189
10.3.3 Comparing circuit cofactoring with cube-wise enumeration 190
10.4 COFACTOR REPRESENTATION 191
10.5 ENUMERATION USING HYBRID SAT 192
10.5.1 Heuristics to Enlarge the Satisfying State Set 193
10.6 SAT-BASED UMC 197
10.6.1 SAT-based Existential Quantification using Circuit Cofactor 198
10.6.2 SAT-based UMC for $F(p)$ 198
10.6.3 SAT-based UMC for $G(q)$ 199
10.6.4 SAT-based UMC for $F(p \wedge G(q))$ 202
10.7 EXPERIMENTS FOR SAFETY PROPERTIES 203
10.7.1 Industry Benchmarks 203
10.7.2 Public Verification Benchmarks 206
10.8 EXPERIMENTS FOR LIVENESS PROPERTIES 207
10.9 RELATED WORK 209
10.10 SUMMARY 211
10.11 NOTES 212

PART IV: ABSTRACTION/REFINEMENT 213

11 PROOF-BASED ITERATIVE ABSTRACTION 215

11.1 INTRODUCTION 215
11.2 PROOF-BASED ABSTRACTION (PBA): OVERVIEW 218
11.3 LATCH-BASED ABSTRACTION 219
11.4 PRUNING IN LATCH INTERFACE ABSTRACTION 222
11.4.1 Environmental Constraints 223
11.4.2 Latch Interface Propagation Constraints 224
11.5 ABSTRACT MODELS 225
11.6 IMPROVING ABSTRACTION USING LAZY CONSTRAINTS 226
11.6.1 Making Eager Constraints Lazy 227
11.7 ITERATIVE ABSTRACTION FRAMEWORK 228
11.7.1 Inner Loop of the Framework 228
11.7.2 Handling Counterexamples 229
11.7.3 Lazy Constraints in Iterative Framework 230
11.8 APPLICATION OF PROOF-BASED ITERATIVE ABSTRACTION 231
11.9 EMM WITH PROOF-BASED ABSTRACTION 232
11.10 EXPERIMENTAL RESULTS OF LATCH-BASED ABSTRACTION 233
11.10.1 Results for Iterative Abstraction 233
11.10.2 Results for Verification of Abstract Models 235
11.11 EXPERIMENTAL RESULTS USING LAZY CONSTRAINTS 236
11.11.1 Results for Use of Lazy Constraints 236
11.11.2 Proofs on Final Abstract Models 239
11.12 CASE STUDY: EMM WITH PBIA 240
11.13 RELATED WORK 242
11.14 SUMMARY 243
11.15 NOTES 243

PART V: VERIFICATION PROCEDURE 245

12 SAT-BASED VERIFICATION FRAMEWORK 247

12.1 INTRODUCTION 247
12.2 VERIFICATION MODEL AND PROPERTIES 248
12.3 VERIFICATION ENGINES 250
12.4 VERIFICATION ENGINE ANALYSIS 254
12.5 VERIFICATION STRATEGIES: CASE STUDIES 256
12.6 SUMMARY 261
12.7 NOTES 261

13 SYNTHESIS FOR VERIFICATION 263

13.1 INTRODUCTION 263
13.2 CURRENT METHODOLOGY 265
13.3 SYNTHESIS FOR VERIFICATION PARADIGM 267
13.4 HIGH-LEVEL VERIFICATION MODELS 269
13.4.1 High-level Synthesis (HLS) 269

13.4.2 Extended Finite State Machine (EFSM) Model 269
13.4.3 Flow Graphs 271
13.5 "BMC-FRIENDLY" MODELING ISSUES 272
13.6 SYNTHESIZING "BMC-FRIENDLY" MODELS 273
13.7 EFSM LEARNING 274
13.7.1 Extraction: Control State Reachability (CSR) 274
13.7.2 On-the-Fly Simplification 275
13.7.3 Unreachablility of Control States 277
13.8 EFSM TRANSFORMATIONS 277
13.8.1 Property-based EFSM Reduction 278
13.8.2 Balancing Re-convergence 278
13.8.3 Balancing Re-convergence without Loops 280
13.8.4 Balancing Re-convergence with Loops 282
13.9 HIGH-LEVEL BMC ON EFSM 285
13.9.1 Expression Simplifier 286
13.9.2 Incremental Learning in High-level BMC 287
13.10 EXPERIMENTS 287
13.10.1 Controlled Case Study 287
13.10.2 Experiments on Industry Software bc-1.06 289
13.10.3 Experiments on Industry Embedded System Software 292
13.10.4 Experiments on System-level Model 293
13.11 SUMMARY AND FUTURE WORK 294
13.12 NOTES 295

REFERENCES 297

GLOSSARY 309

INDEX 317

ABOUT THE AUTHORS 325