

# Contents

<b>Certificate of Approval</b>	<b>vii</b>
<b>Certificate</b>	<b>ix</b>
<b>Declaration</b>	<b>xi</b>
<b>Acknowledgments</b>	<b>xiii</b>
<b>List of Abbreviations</b>	<b>xv</b>
<b>List of Symbols</b>	<b>xvii</b>
<b>Abstract</b>	<b>xix</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation and Objectives . . . . .	2
1.2 Summary of Contributions . . . . .	5
1.2.1 Trace Assisted Formal Methods for the Verification of Bug Fixes	5
1.2.2 Formal Methods for Ranking Counterexamples . . . . .	8
1.2.3 Reusing Component Invariants in Equivalence Checking . . . . .	10
1.3 Organization of the Thesis . . . . .	12
<b>2 Background and Related Work</b>	<b>15</b>
2.1 Specification Development . . . . .	15
2.1.1 Formal Representation of the design . . . . .	15
2.1.2 Temporal Logics . . . . .	16
2.1.3 Linear Temporal Logic . . . . .	16
2.1.4 Specification Languages Used in Industry . . . . .	17
2.1.5 SystemVerilog Assertions (SVA) . . . . .	18
2.1.5.1 SVA Sequences . . . . .	18
2.1.5.2 Property Declarations . . . . .	20
2.1.5.3 Concurrent Assertions . . . . .	21
2.2 Pre-Silicon Verification: Major Paradigms . . . . .	23
2.2.1 Formal Property Verification . . . . .	23
2.2.1.1 LTL Model Checking . . . . .	24
2.2.2 Sequential Equivalence Checking . . . . .	25
2.2.3 Advanced Techniques for FPV and SEC . . . . .	26
2.2.3.1 Symbolic BDD-based Approach . . . . .	26
2.2.3.2 Symbolic SAT-Based Approach . . . . .	28
2.2.3.3 Use of Invariants . . . . .	29

2.2.3.4	Counterexample Guided Abstraction Refinement . . .	30
2.2.4	Dynamic Assertion-Based / Simulation-Based Verification . . .	31
2.2.5	Semi-formal Verification . . . . .	32
2.2.6	Existing Tool Suites . . . . .	32
2.3	Post-Silicon Validation and Debugging . . . . .	33
2.3.1	Debugging Methodologies . . . . .	33
2.3.2	Bug Trace Minimization and Bug Localization . . . . .	34
2.3.3	Formal Methods for Post-Silicon Debug . . . . .	35
2.4	Specification Mining . . . . .	36
2.4.1	Current Practices . . . . .	36
2.4.2	Existing Tool Suites . . . . .	37
2.5	Bayesian Networks . . . . .	40
2.6	Concluding Remarks . . . . .	41
<b>3</b>	<b>Trace Assisted Formal Methods for the Verification of Bug Fixes</b>	<b>43</b>
3.1	Bug fix classification: The OpenSPARC story . . . . .	46
3.2	Bug fix taxonomy . . . . .	49
3.3	Methodology Outline . . . . .	52
3.4	Detailed Methodology . . . . .	55
3.4.1	Slice computation . . . . .	55
3.4.2	Weakest pre-condition (WP) along a slice . . . . .	60
3.5	Bug Fix Classification Methodology . . . . .	62
3.5.1	Type-1 Fix . . . . .	62
3.5.2	Type-2 Fix . . . . .	64
3.5.3	Type-3 Fix . . . . .	65
3.6	Implementation Details . . . . .	66
3.7	Experimental Results . . . . .	66
3.7.1	Cache Coherence Test Case of Pentium Pro . . . . .	67
3.8	Conclusion . . . . .	69
<b>4</b>	<b>Formal Methods for Ranking Counterexamples</b>	<b>71</b>
4.1	Ranking using Global Simulation Dump . . . . .	72
4.1.1	Running Example . . . . .	73
4.1.2	Counterexample Ranking Metric . . . . .	75
4.1.2.1	Confidence computation . . . . .	76
4.1.2.2	Counterexample Families . . . . .	77
4.1.3	Assumption Mining and Confidence Metric . . . . .	78
4.1.3.1	Assumption Mining . . . . .	79
4.1.3.2	Belief Computation . . . . .	80
4.1.4	GENRANKCEX Tool Flow and Experimental Results . . . . .	82
4.2	Ranking using Scattered Dumps . . . . .	84
4.2.1	Illustration on a Toy Example . . . . .	86
4.2.2	Counterexample Confidence Computation . . . . .	89
4.2.2.1	Construction and Querying of the Bayesian network . . . . .	89
4.2.3	Filtering of Modules and Signals . . . . .	91
4.3	Experimental Results on scattered traces . . . . .	91
4.4	Concluding Remarks . . . . .	93

<b>5</b>	<b>A Case Study on OpenSPARC</b>	<b>95</b>
5.1	Major Functional Units of OpenSPARC T1 . . . . .	95
5.2	The L2 Cache of OpenSPARC . . . . .	96
5.3	Normal Working of L2 Cache Bank . . . . .	99
5.4	The Bug Scenario . . . . .	100
5.5	Bug Sensitization . . . . .	101
5.6	The Bug Fixes . . . . .	103
5.7	Experimental Results on OpenSPARC . . . . .	104
5.7.1	Trace Assisted Formal Methods for the Verification of Bug Fixes	104
5.7.2	Formal Methods for Ranking Counterexamples . . . . .	106
5.8	Concluding Remarks . . . . .	108
<b>6</b>	<b>Reusing Component Invariants in Equivalence Checking</b>	<b>109</b>
6.1	Definitions and Background . . . . .	112
6.1.1	Target Enlargement . . . . .	112
6.1.2	Constraint or Care Set of a Boolean Function . . . . .	112
6.1.3	Existing BDD Simplification Techniques . . . . .	112
6.1.3.1	BDD Restrict Operation . . . . .	113
6.1.3.2	BDD Constrain Operation . . . . .	113
6.1.3.3	Relative Merits and Demerits . . . . .	114
6.2	Proposed Methodology . . . . .	115
6.2.1	Reuse of Component SEC Results . . . . .	115
6.2.2	The Formalisms involved . . . . .	117
6.2.3	Some Rare Pathological Cases . . . . .	120
6.2.4	A Case Study on a Small Example . . . . .	122
6.2.5	Enhanced SEC Algorithm . . . . .	125
6.3	Tool Flow and Experimental Results . . . . .	127
6.3.1	Experimentations with ISCAS 89 and ITC 99 benchmark circuits	128
6.3.2	Experimentations on HWMCC 08 benchmark circuits . . . . .	132
6.3.3	Comparison with ABC . . . . .	133
6.4	Concluding Remarks . . . . .	135
<b>7</b>	<b>Conclusions</b>	<b>137</b>
	<b>Bibliography</b>	<b>143</b>