

Contents

- 1 Introduction.** 1
 - 1.1 Reliability Threats 2
 - 1.2 Increasing Trends for Soft Errors. 5
 - 1.3 Research Challenges for Enabling Cross-Layer Software Reliability 9
 - 1.4 Contributions of This Manuscript 11
 - 1.4.1 Cross-Layer Software Program Reliability Modeling and Estimation 12
 - 1.4.2 Cross-Layer Software Program Reliability Optimization . . 13
 - 1.5 Orientation of This Work in the SPP 1500 Priority Research Program on Dependable Embedded Systems 16
 - 1.6 Outline of the Manuscript 18
- 2 Background and Related Work** 23
 - 2.1 Soft Error 23
 - 2.1.1 Transistor Structure 23
 - 2.1.2 Soft Errors in Transistors 24
 - 2.1.3 Masking Sources for Soft Errors 26
 - 2.2 NBTI-Induced Aging 27
 - 2.3 Manufacturing-Induced Process Variations and Other Variability Sources 29
 - 2.3.1 Process Variation Model 31
 - 2.4 State-of-the-Art Soft Error Estimation Techniques 32
 - 2.4.1 Circuit-Level Techniques 32
 - 2.4.2 Architecture-Level Techniques 34
 - 2.4.3 Software Program-Level Techniques 35
 - 2.4.4 Fault Injection Methodologies 36
 - 2.5 State-of-the-Art Soft Error Mitigation Techniques 37
 - 2.5.1 Hardware-Level Soft Error Mitigation Techniques 37
 - 2.5.2 Software-Level Soft Error Mitigation Techniques 43
 - 2.6 Summary of Related Work 48

3	Cross-Layer Reliability Analysis, Modeling, and Optimization	51
3.1	Cross-Layer Reliability: System Overview	53
3.2	Software Program-Level Reliability Analysis	56
3.2.1	Error Characterization	57
3.2.2	Error Distribution Analysis	59
3.2.3	Instruction Classification: Critical and Noncritical Instructions	61
3.2.4	Spatial and Temporal Vulnerability	62
3.2.5	Summary of Software Program Reliability Analysis and Relevant Parameters	65
3.3	Cross-Layer Reliability Modeling: A Soft Error Perspective	66
3.3.1	Estimating the Probability of Fault for Different Processor Components Through Gate-Level Soft Error Masking Analysis	68
3.3.2	Estimating Reliability at the Instruction Granularity as a Function of Vulnerability, Error Masking, and Propagation	69
3.3.3	Function-Level Reliability Models	70
3.3.4	Consideration for Timing Correctness	71
3.4	Consideration of Aging Faults	71
3.5	Reliability-Driven Compilation Flow	72
3.5.1	Reliability-Driven Software Transformations	73
3.5.2	Reliability-Driven Instruction Scheduling	74
3.5.3	Reliability-Driven Selective Instruction Protection	75
3.5.4	Generating Multiple Function Versions Providing Tradeoff Between Performance and Reliability	75
3.6	Reliability-Driven System Software	76
3.6.1	Reliability-Driven Offline System Software	77
3.6.2	Reliability-Driven Adaptive Run-Time System Software . .	77
3.6.3	Comparing Cross-Layer vs. Single-Layer Reliability Optimizing Techniques	78
3.7	Chapter Summary	79
4	Software Program-Level Reliability Modeling and Estimation	81
4.1	Instruction Vulnerability Index	83
4.1.1	Estimation of Vulnerable Periods	84
4.1.2	Estimation of Vulnerable Bits	86
4.1.3	Estimation of Component-Level Fault Probabilities	87
4.1.4	IVI Results for Different Applications	90
4.2	Instruction Error Masking Index	92
4.2.1	Parameter Identification	92
4.2.2	An Example	93
4.2.3	Parameter Estimation	93
4.2.4	Instruction Error Masking Index Results for Different Applications	97

4.3	Instruction Error Propagation Index.	98
4.4	Function-/Task-Level Reliability Estimation Models	100
4.4.1	Function Vulnerability Index	101
4.4.2	Reliability-Timing Penalty.	102
4.5	Chapter Summary.	102
5	Software Program-Level Reliability Optimization for Dependable Code Generation	105
5.1	Reliability-Driven Software Transformation	107
5.1.1	Reliability-Driven Data Type Optimization.	108
5.1.2	Reliability-Driven Loop Unrolling.	111
5.1.3	Reliability-Driven Common Expression Elimination and Operation Merging	114
5.1.4	Reliability-Driven Online Table Value Computation.	118
5.1.5	Impact of Reliability-Driven Transformations on Error Distributions	120
5.1.6	Selection of Transformations.	124
5.1.7	Impact on Critical ALU Instructions	124
5.1.8	Impact on Performance Overhead When Employed Together with Error Detection and Recovery Techniques . .	127
5.1.9	Impact on FVI Reductions	128
5.1.10	Summary of Reliability-Driven Transformations	128
5.2	Reliability-Driven Instruction Scheduling	128
5.2.1	Soft Error-Driven Instruction Scheduling	131
5.2.2	Formal Problem Modeling	132
5.2.3	Lookahead Instruction Scheduling Heuristic.	133
5.2.4	Results for the Reliability-Driven Instruction Scheduling	135
5.2.5	Summary of Reliability-Driven Instruction Scheduling . .	137
5.3	Reliability-Driven Selective Instruction Protection	138
5.3.1	Reliability Profit Function for Choosing Instructions for Protection	138
5.3.2	Flow of the Selective Instruction Protection Heuristic . . .	140
5.3.3	Results for Selective Instruction Protection.	140
5.3.4	Summary of Selective Instruction Protection.	143
5.4	Multiple Function Version Generation and Selection	143
5.5	Chapter Summary.	145
6	Dependable Code Execution Using Reliability-Driven System Software	147
6.1	Reliability-Driven Offline System Software	148
6.1.1	Optimization Objective	151
6.1.2	Optimizing for the Reliability-Timing Penalty	151
6.2	Reliability-Driven Function Scheduling for Single Core Processors.	155

6.3	Reliability-Driven System Software for Multi-/Manycores	160
6.3.1	Soft Error Resilience in the Presence of Process Variations and Aging Effects	160
6.3.2	Dependability Tuning System for Soft Error Resilience under Variations	162
6.3.3	Dynamic RMT Adaptations and Core Allocation	164
6.3.4	Dynamic Reliable Code Version Selection	165
6.4	Chapter Summary	168
7	Results and Discussion	171
7.1	Processor Synthesis and Performance Variation Estimation	171
7.1.1	Processor Synthesis	171
7.1.2	Processor Aging Estimation	172
7.1.3	Process Variation Maps	173
7.2	Benchmark Applications	174
7.3	Comparison Partners and Evaluation Parameters	174
7.3.1	Comparison Partners	175
7.3.2	Parameters Considered for the Evaluation	176
7.3.3	An Overview of the Comparison Results	177
7.4	Overview of Savings Compared to State-of-the-Art	178
7.5	Detailed Comparison Results	182
7.6	Detailed Analysis of Comparison Results for Two Chips for a 6×6 Processor	185
7.7	Chapter Summary	188
8	Summary and Conclusions	189
	Appendix A: Simulation Infrastructure	193
A.1:	Reliability-Aware Manycore Instruction Set Simulator and Fault Injection	194
A.2:	ArchC Architecture Description Language	195
A.3:	Reliability-Aware Simulation and Analysis Methodology	197
	Appendix B: Function-Level Resilience Modeling	205
B.1:	Definition	205
B.2:	Modeling Function Resilience	205
B.3:	Results	209
	Appendix C: Algorithms	211
C.1:	Algorithm for Computing the Error Masking Probability PDP(I, p)	211
C.2:	Algorithm for Computing the Instruction Error Propagation Index	212
C.3:	Algorithm for FVI-Driven Data Type Optimization	213
C.4:	Algorithm for FVI-Driven Loop Unrolling	214
C.5:	Algorithm for Applying Common Expression Elimination	216
C.6:	Algorithm for Soft-Error-Driven Instruction Scheduler	217
C.7:	Algorithm for Selective Instruction Protection Technique	218
C.8:	Algorithm for Offline Table Construction	219

Contents	xiii
C.9: Algorithm for Hybrid RMT Tuning	221
C.10: Algorithm for Reliable Code Version Tuning.	222
C.11: Algorithm for Core Tuning and Version Update.	223
Appendix D: Notations and Symbols	225
Bibliography	229

<http://www.springer.com/978-3-319-25770-9>

Reliable Software for Unreliable Hardware

A Cross Layer Perspective

Rehman, S.; Shafique, M.; Henkel, J.

2016, XXVIII, 237 p. 121 illus., 83 illus. in color.,

Hardcover

ISBN: 978-3-319-25770-9