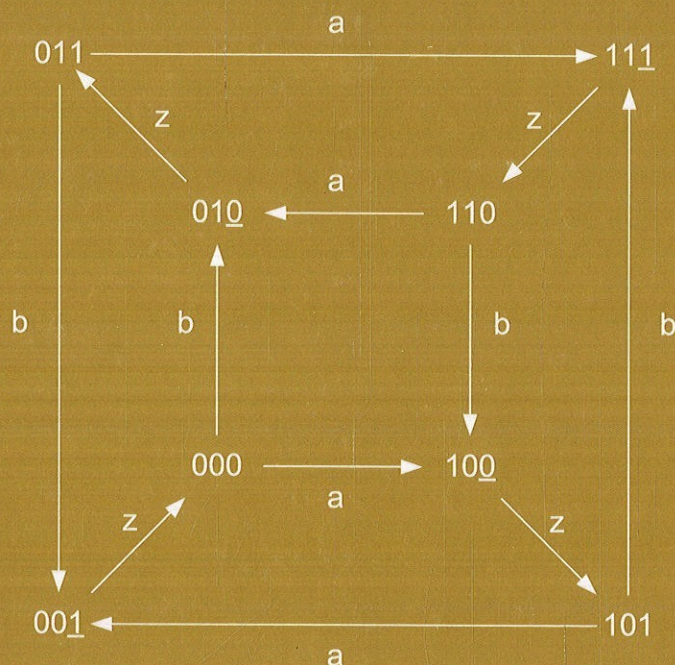


VERIFICATION OF SYSTEMS AND CIRCUITS

USING LOTOS, PETRI NETS, AND CCS



MICHAEL YOELI • RAKEFET KOL

Wiley Series on Parallel and Distributed Computing
Albert Y. Zomaya, Series Editor

CONTENTS

1. Introduction	1
1.1 Event-Based Approach	2
1.2 Event-Based Systems	2
1.3 Types of Verification	2
1.4 Toolsets Used	3
1.5 Level-Based Approach	3
1.6 Overview of the Book	3
1.7 References	5
 2. Processes	 7
2.1 Introduction	7
2.2 Examples of Processes and Basic Concepts	7
2.3 About Prefixing	10
2.4 Process Graphs	10
2.5 Choice Operator	11
2.6 Another Process Example	13
2.7 Equivalences	13
2.7.1 Strong Equivalence	13
2.7.2 Observation Equivalence	14
2.7.3 Some Additional Laws	15
2.8 Labeled Transition Systems (LTSs)	15
2.9 Parallel Operators	16
2.9.1 Parallel Composition	16
2.9.2 Synchronization Operator $\parallel$ (Blot Version)	16
2.9.3 Examples of Parallel Compositions	17
2.9.4 More Laws	17

2.9.5	Sample Proof	18
2.9.6	Interleaving Operator $\parallel$	18
2.10	Sequential Composition	18
2.11	Further Reading	19
2.12	Selected Solutions	20
2.13	References	21
3.	From Digital Hardware to Processes	23
3.1	The C-Element	23
3.1.1	The 2-Input CEL-Circuit	23
3.1.2	The 3-Input CEL-Circuit	25
3.1.3	The 4-Input CEL-Circuit	26
3.2	The XOR-Gate	26
3.2.1	The 2-Input XOR-Gate	26
3.2.2	The 3-Input XOR-Gate	27
3.3	TOGGLES	29
3.4	Modulo- N Transition Counters	30
3.4.1	Modulo- N Transition Counter Specification	30
3.4.2	Modulo- N Transition Counter Implementations	30
3.4.2.1	The Cases $N = 3$ and $N = 4$	31
3.4.2.2	The $N > 4$ Case	31
3.5	Modular Networks	31
3.6	Propositional Logic: A Review of Known Concepts	33
3.6.1	Logical Operators	34
3.6.2	Proving Logical Equivalences	35
3.6.3	Tautologies and the EQUIV Operator	36
3.7	Selected Solutions	36
3.8	References	37
4.	Introducing LOTOS	39
4.1	From Blot to Basic LOTOS	39
4.1.1	Recursion	40
4.2	Some Semantics	41
4.3	From LTS to LOTOS	42
4.4	Comparing Parallel Operators	43
4.5	Sequential Composition	44
4.6	Hiding	44
4.7	Equivalences and Preorders	44
4.8	About CADP	45
4.8.1	Getting Started with CADP	45
4.8.2	Verifying Equivalences and Preorders Using CADP	46

4.8.2.1	Verifying Equivalences Using CADP	46
4.8.2.2	Verifying Preorders Using CADP	47
4.8.3	Generating LTS of Choice Using CADP	47
4.8.4	Generating LTS of Recursion Using CADP	48
4.9	Full LOTOS—An Introduction	49
4.9.1	The Full-LOTOS NOT-Gate Example	49
4.9.1.1	The Full LOTOS NOT-File	49
4.9.1.2	Applying CADP to Derive LTS for the NOT-Gate	50
4.9.2	The Non-Terminating NOT-Gate	51
4.9.3	The Max Specifications	52
4.9.3.1	Max2 Specification	52
4.9.3.2	Max3 Specification	52
4.10	The Regular Mu-Calculus (RMC)	53
4.10.1	Introducing RMC by Examples	53
4.11	Further Reading	55
4.12	Selected Solutions	56
4.13	References	57

5. Introducing Petri Nets 59

5.1	About Petri Nets	59
5.1.1	Petri Graphs and Petri Nets	59
5.1.2	Enabling and Firing	60
5.1.3	Another Definition of Petri Nets	61
5.2	About Languages	61
5.3	About PETRIFY	62
5.4	Illustrating Petri Nets	64
5.5	Labeled Nets	66
5.6	Bounded Nets	68
5.7	Observation Equivalence of LPNs	70
5.8	From Blot to Petri Nets	70
5.9	Liveness and Persistence	72
5.10	Simple Reduction Rules	72
5.11	Marked Graphs	74
5.12	A Simple Net Algebra	75
5.12.1	The Prefix Operator	75
5.12.2	The Choice Operator	77
5.12.3	The Star Operator	77
5.12.4	Parallel Compositions	79
5.12.4.1	The Basic Approach	79
5.12.4.2	The Multiple-Labeled Case	79

5.13	Arc-Weighted Nets	80
5.13.1	Enabling and Firing in Arc-Weighted Nets	80
5.13.2	Arc-Weighted Versus Non-Labeled Nets	82
5.14	Readers–Writers System	83
5.14.1	A Readers–Writers System Net Representation	83
5.14.2	Verification of a Readers–Writers System	84
5.15	Inhibitor Nets	85
5.15.1	Introduction to Inhibitor Nets	85
5.15.2	The Expressive Power of Inhibitor Nets	85
5.16	True Concurrency	86
5.16.1	The Pi-Language	87
5.16.2	Pi-Equivalence	87
5.16.3	Concurrency-Preserving Synthesis	88
5.17	Further Reading	89
5.18	Selected Solutions	89
5.19	References	93
6.	Introducing CCS	95
6.1	About CCS	95
6.2	Operators ‘Prefix’ and ‘Sum’	95
6.2.1	Semantics	96
6.3	Recursion	97
6.3.1	Semantics	97
6.4	Concurrency Operator	97
6.5	Equivalences	98
6.6	Restriction	98
6.7	CTL	99
6.7.1	Introducing CTL	99
6.8	The Concurrency Workbench (CWB)	100
6.8.1	The ‘sim’ and ‘compile’ Commands	100
6.8.2	Checking Equivalences	102
6.8.3	Checking Restrictions	103
6.8.4	HML Formulas	103
6.8.5	Equivalences—Counterexamples	104
6.8.6	More Equivalence Checking	105
6.8.7	Using the mu-Calculus	106
6.8.8	Using CTL	107
6.9	CCS and CWB Application Examples	109
6.9.1	The CCS XCEL-Circuit Example	109

6.9.1.1	The CCS Approach	109
6.9.1.2	Comparing the CCS Approach with the LOTOS Approach	111
6.9.2	The CCS CEL3-Circuit Example	112
6.10	Further Reading	113
6.11	Selected Solutions	114
6.12	References	115
7.	Verification of Modular Asynchronous Circuits	117
7.1	About Asynchronous Circuits	117
7.1.1	Modular Asynchronous Circuits	117
7.1.2	Edge-Based (Dynamic) Versus Level-Based Behavior	118
7.2	XOR-Gates	118
7.2.1	LOTOS Representation of XOR-Gate	118
7.2.2	Petri Net Representation of XOR-Gate	119
7.2.3	CCS Representation of XOR-Gate	119
7.3	CEL-Circuit	119
7.3.1	LOTOS Representation of CEL-Circuit	120
7.3.2	Petri Net Representation of CEL-Circuit	120
7.3.3	CCS Representation of CEL-Circuit	120
7.4	Other Modules	121
7.4.1	Inverter	121
7.4.2	ICEL-Element	121
7.4.3	TOGGLE	122
7.4.4	CALL	122
7.5	Module Extensions	123
7.5.1	XOR k ($k > 2$) Modules	123
7.5.2	CEL k ($k > 2$) Modules	123
7.5.3	TOG k ($k > 2$)	124
7.6	Modular Networks	124
7.7	Realizations	125
7.7.1	Introduction to Realization	125
7.7.2	Type-A Realization	125
7.7.2.1	Type-A1 Realization	126
7.7.3	Type-B Realization	126
7.7.4	Type-C Realization	128
7.7.5	Type-D Realization	130
7.7.5.1	Extended Type-D Realizations	131
7.7.6	DI Realization	131

7.8	Verification of Extended Modules	131
7.8.1	Verification of XOR_k ($k > 2$) Modules	132
7.8.1.1	Implementation of XOR_k	132
7.8.1.2	Verification of XOR_k Using Petri Nets and PETRIFY	132
7.8.1.3	Verification of XOR_k Using LOTOS and CADP	134
7.8.1.4	Verification of XOR_k Using CCS and CWB-NC	135
7.8.2	Verification of CEL_k ($k > 2$) Modules	135
7.8.2.1	Implementation of CEL_k	135
7.8.2.2	Verification of CEL_k Using Petri Nets and PETRIFY	135
7.8.2.3	Verification of CEL_k Using LOTOS and CADP	136
7.8.2.4	Verification of CEL_k Using CCS and CWB-NC	136
7.8.3	Verification of TOG_k ($k > 2$) Modules	137
7.9	Verification of Parallel Control Structures	137
7.10	Further Reading	140
7.11	Selected Solutions	140
7.12	References	146

8. Verification of Communication Protocols 147

8.1	Introduction	147
8.2	Two Simple Communication Protocols	147
8.2.1	Simple Communication Protocol SCP	148
8.2.2	Simple Communication Protocol SCP1	148
8.3	The Alternating Bit (AB) Protocol	149
8.3.1	Introduction	149
8.3.2	The Reliable Channel Case	149
8.3.2.1	A LOTOS Description of the Reliable Channel Case	150
8.3.3	The Unreliable Channel Case	151
8.3.3.1	A LOTOS Verification of the Unreliable Channel Case	151
8.3.3.2	A CCS Verification of the Unreliable Channel Case	154
8.4	Further Reading	156
8.5	Selected Solutions	156
8.6	References	157

9. Verification of Arbiters	159
9.1 Introduction	159
9.2 A Random Arbiter (RGDA)	159
9.2.1 Verifying RGDA Using LOTOS	160
9.2.1.1 Blot and LOTOS Representation of RGDA	160
9.2.1.2 Verification of RGDA Using LOTOS	161
9.2.1.2.1 Verifying Mutual Exclusion	161
9.2.1.2.2 Verifying Grant Only on Request	163
9.2.1.2.3 Verifying Fairness	163
9.2.2 Verifying RGDA Using Petri Nets	163
9.2.2.1 Petri Net Representation of RGDA	163
9.2.2.2 Verification of RGDA Using Petri Net	164
9.2.2.2.1 Verifying Mutual Exclusion	164
9.2.2.2.2 Verifying Grant Only on Request	165
9.2.2.2.3 Verifying Fairness	165
9.2.3 Verifying RGDA Using CCS	165
9.3 A Token-Ring Arbiter	167
9.3.1 A Petri Net Representation of a Token-Ring Arbiter	167
9.3.2 Verification of a Token-Ring Arbiter Using Petri Net	168
9.4 Further Reading	168
9.5 Selected Solutions	169
9.6 References	171
 10. More Verification Case Studies	 173
10.1 Verification of Combinational Logic	173
10.1.1 The AND Gate	173
10.1.2 Composite Gates	175
10.2 Verification of Asynchronous Pipeline Controllers	177
10.2.1 Introduction	177
10.2.1.1 Transition Signaling	178
10.2.1.2 The Bundled Data Interface	178
10.2.2 Latch Control Unit	178
10.2.2.1 A Blot Specification of LCU	178
10.2.2.2 A LOTOS Specification of LCU	179
10.2.2.3 A LOTOS Implementation of LCU	179
10.2.2.4 Latch Problems	180
10.2.3 Phase Converters	181
10.2.3.1 2-Phase to 4-Phase Converter (PC24)	181
10.2.3.2 4-Phase to 2-Phase Converter (PC42)	182

10.3	Verification of Producer–Consumer Systems	183
10.3.1	Introduction	183
10.3.2	Verifying Producer–Consumer Systems Using Petri Nets	183
10.3.3	Occurrence Counts	184
10.3.4	Verifying the Producer–Consumer System Using Occurrence Counts	184
10.3.4.1	Verifying Liveness	184
10.3.4.2	Verifying Boundedness	185
10.3.4.3	Verifying Overflow/Underflow	185
10.3.5	Verifying Producer–Consumer Systems Using LOTOS	185
10.3.6	Verification of Multiple-Producer Multiple-Consumer Systems	186
10.3.6.1	A 2-Producer–2-Consumer System with Priority	186
10.4	Verification Based on Design Approaches	188
10.4.1	Synthesis Approach #1	188
10.4.1.1	Realization	188
10.4.1.2	Formal Verification of the CXC-Circuit Example	188
10.4.1.3	Another Synthesis Verification Example	189
10.4.2	Synthesis Approach #2	189
10.4.3	Extending the Synthesis Method by Adding XOR Modules	190
10.4.4	A Decomposition Approach	191
10.5	Verification of Toggles and Transition Counters	193
10.5.1	Verification of k -Toggles	193
10.5.2	Verification of Counters without Outputs	194
10.5.2.1	Direct Verification of udcout	194
10.5.2.2	Verification of udcout Using Petri Net	195
10.5.3	Verification of Up–Down Counters with Output	196
10.5.4	Verification of Modulo- N Transition Counters	196
10.5.4.1	Specification of Modulo- N Transition Counter	197
10.5.4.2	Decompositions of Modulo- N Counters	197
10.5.4.3	Verifying a Modulo-3 Counter	197
10.5.4.4	Verifying a Modulo-12 Counter	198
10.6	Vending Machines Verification—Revisited	199
10.6.1	Verifying Vending Machines VeMa Using CCS	199
10.6.2	Verifying Vending Machines VeMa Using LOTOS	200

10.7	Pi-Realizations	201
10.7.1	More on Modular Networks	202
10.7.2	Introducing Circuits	203
10.7.3	Concurrent Behavior of Circuits	204
10.7.4	Pi-Specifications of Circuits	205
	10.7.4.1 Pi-Processes	205
	10.7.4.2 Specifying Circuits by Pi-Processes	205
10.7.5	Simple Verification Examples	206
10.7.6	Applying Net Algebra	206
10.7.7	Another Verification Example	207
10.7.8	Some Pi Propositions	209
10.8	A Comparison of Equivalence Relations	210
10.8.1	An Equivalence Theorem	210
10.8.2	An Application of the Equivalence Theorem	211
10.9	Selected Solutions	211
10.10	References	223

11. Guide to Further Studies 225

11.1	Verification of Telecommunication Systems	225
11.1.1	Plain Old Telephone System (POTS)	225
11.1.2	Advanced Telephone Systems	226
11.1.3	ISDN Telephony	226
11.2	Verification Using Colored Petri Nets	226
11.3	Verification of Traffic Signal Control Systems	226
11.4	References	227