

# Contents

<b>1. Introduction</b>	1
1.1 A Little History	1
1.2 Advantages of Asynchronous Logic	3
1.2.1 Modularity	3
1.2.2 Power Consumption and Electromagnetic Interference	4
1.2.3 Performance	6
1.3 Asynchronous Control Circuits	7
1.3.1 Delay Models	10
1.3.2 Operating Modes	11
<b>2. Design Flow</b>	13
2.1 Specification of Asynchronous Controllers	13
2.1.1 From Timing Diagrams to Signal Transition Graphs	14
2.1.2 Choice in Signal Transition Graphs	15
2.2 Transition Systems and State Graphs	16
2.2.1 State Space	16
2.2.2 Binary Interpretation	17
2.3 Deriving Logic Equations	19
2.3.1 System Behavior	19
2.3.2 Excitation and Quiescent Regions	19
2.3.3 Next-state Functions	20
2.4 State Encoding	21
2.5 Logic Decomposition and Technology Mapping	23
2.6 Synthesis with Relative Timing	25
2.7 Summary	27
<b>3. Background</b>	29
3.1 Petri Nets	29
3.1.1 The Dining Philosophers	31
3.2 Structural Theory of Petri Nets	37
3.2.1 Incidence Matrix and State Equation	37
3.2.2 Transition and Place Invariants	38
3.3 Calculating the Reachability Graph of a Petri Net	39
3.3.1 Encoding	41

3.3.2	Transition Function and Reachable Markings .....	42
3.4	Transition Systems .....	44
3.5	Deriving Petri Nets from Transition Systems .....	45
3.5.1	Regions .....	45
3.5.2	Properties of Regions .....	47
3.5.3	Excitation Regions .....	47
3.5.4	Excitation-Closure .....	48
3.5.5	Place-Irredundant and Place-Minimal Petri Nets .....	49
3.6	Algorithm for Petri Net Synthesis .....	52
3.6.1	Generation of Minimal Pre-regions .....	53
3.6.2	Search for Irredundant Sets of Regions .....	54
3.6.3	Label Splitting .....	55
3.7	Event Insertion in Transition Systems .....	57
<b>4.</b>	<b>Logic Synthesis .....</b>	<b>61</b>
4.1	Signal Transition Graphs and State Graphs .....	62
4.1.1	Signal Transition Graphs .....	62
4.1.2	State Graphs .....	64
4.1.3	Excitation and Quiescent Regions .....	65
4.2	Implementability as a Logic Circuit .....	66
4.2.1	Boundedness .....	66
4.2.2	Consistency .....	67
4.2.3	Complete State Coding .....	69
4.2.4	Output Persistency .....	70
4.3	Boolean Functions .....	73
4.3.1	ON, OFF and DC Sets .....	73
4.3.2	Support of a Boolean Function .....	73
4.3.3	Cofactors and Shannon Expansion .....	74
4.3.4	Existential Abstraction and Boolean Difference .....	74
4.3.5	Unate and Binate Functions .....	74
4.3.6	Function Implementation .....	74
4.3.7	Boolean Relations .....	75
4.4	Gate Netlists .....	75
4.4.1	Complex Gates .....	76
4.4.2	Generalized C-Elements .....	76
4.4.3	C-Elements with Complex Gates .....	78
4.5	Deriving a Gate Netlist .....	79
4.5.1	Deriving Functions for Complex Gates .....	79
4.5.2	Deriving Functions for Generalized C-Elements .....	81
4.6	What is Speed-Independence? .....	82
4.6.1	Characterization of Speed-Independence .....	85
4.6.2	Related Work .....	85
4.7	Summary .....	86

<b>5. State Encoding</b>	87
5.1 Methods for Complete State Coding	91
5.2 Constrained Signal Transition Event Insertion	94
5.2.1 Speed-Independence Preserving Insertion	95
5.3 Selecting SIP-Sets	101
5.4 Transformation of State Graphs	103
5.5 Completeness of the Method	107
5.6 An Heuristic Strategy to Solve CSC	115
5.6.1 Generation of I-Partitions	115
5.6.2 Exploring the Space of I-Partitions	116
5.6.3 Increasing Concurrency	117
5.7 Cost Function	118
5.7.1 Estimation of Logic	119
5.7.2 Examples of CSC Conflict Elimination	119
5.8 Related Work	122
5.9 Summary	123
<b>6. Logic Decomposition</b>	125
6.1 Overview	126
6.2 Architecture-Based Decomposition	132
6.3 Logic Decomposition Using Algebraic Factorization	134
6.3.1 Overview	134
6.3.2 Combinational Decomposition	135
6.3.3 Hazard-Free Signal Insertion	137
6.3.4 Pruning the Solution Space	138
6.3.5 Finding a Valid Excitation Region	139
6.3.6 Progress Analysis	141
6.3.7 Local Progress Conditions	142
6.3.8 Global Progress Conditions	145
6.4 Logic Decomposition Using Boolean Relations	146
6.4.1 Overview	148
6.4.2 Specifying Permissible Decompositions with BRs	150
6.4.3 Functional Representation of Boolean Relations	154
6.4.4 Two-Level Sequential Decomposition	155
6.4.5 Heuristic Selection of the Best Decomposition	160
6.4.6 Signal Acknowledgment and Insertion	160
6.5 Experimental Results	161
6.5.1 The Cost of Speed Independence	163
6.6 Summary	164
<b>7. Synthesis with Relative Timing</b>	167
7.1 Motivation	167
7.1.1 Synthesis with Timing	169
7.1.2 Why Relative Timing?	169
7.1.3 Abstraction of Time	170

7.1.4	Design Flow .....	171
7.2	Lazy Transition Systems and Lazy State Graphs .....	172
7.3	Overview and Example .....	173
7.3.1	First Timing Assumption .....	173
7.3.2	Second Timing Assumption .....	174
7.3.3	Logic Minimization .....	176
7.3.4	Summary .....	177
7.4	Timing Assumptions .....	177
7.4.1	Difference Assumptions .....	179
7.4.2	Simultaneity Assumptions .....	179
7.4.3	Early Enabling Assumptions .....	181
7.5	Synthesis with Relative Timing .....	182
7.5.1	Implementability Properties .....	182
7.5.2	Synthesis Flow with Relative Timing .....	184
7.5.3	Synthesis Algorithm .....	185
7.6	Automatic Generation of Timing Assumptions .....	186
7.6.1	Ordering Relations .....	188
7.6.2	Delay Model .....	190
7.6.3	Rules for Deriving Timing Assumptions .....	190
7.7	Back-Annotation of Timing Constraints .....	193
7.7.1	Correctness Conditions .....	196
7.7.2	Problem Formulation .....	197
7.7.3	Finding a Set of Timing Constraints .....	198
7.8	Experimental Results .....	201
7.8.1	Academic Examples .....	201
7.8.2	A FIFO Controller .....	203
7.8.3	RAPPID Control Circuits .....	206
7.9	Summary .....	206
<b>8.</b>	<b>Design Examples .....</b>	<b>209</b>
8.1	Handshake Communication .....	210
8.1.1	Handshake: Informal Specification .....	210
8.1.2	Circuit Synthesis .....	211
8.2	VME Bus Controller .....	217
8.2.1	VME Bus Controller Specification .....	217
8.2.2	VME Bus Controller Synthesis .....	219
8.2.3	Lessons to be Learned from the Example .....	225
8.3	Controller for Self-timed A/D Converter .....	225
8.3.1	Top Level Description .....	226
8.3.2	Controller Synthesis .....	227
8.3.3	Decomposed Solution for the Scheduler .....	232
8.3.4	Synthesis of the Data Register .....	235
8.3.5	Quality of the Results .....	236
8.3.6	Lessons to be Learned from the Example .....	237
8.4	“Lazy” Token Ring Adapter .....	237

8.4.1	Lazy Token Ring Description .....	238
8.4.2	Adapter Synthesis .....	239
8.4.3	A Model for Performance Analysis .....	242
8.4.4	Lessons to be Learned from the Example .....	243
8.5	Other Examples .....	243
<b>9.</b>	<b>Other Work .....</b>	<b>245</b>
9.1	Hardware Description Languages .....	245
9.2	Structural and Unfolding-based Synthesis .....	247
9.3	Direct Mapping of STGs into Asynchronous Circuits .....	249
9.4	Datapath Design and Interfaces .....	250
9.5	Test Pattern Generation and Design for Testability .....	251
9.6	Verification .....	252
9.7	Asynchronous Silicon .....	253
<b>10.</b>	<b>Conclusions .....</b>	<b>255</b>
	<b>References .....</b>	<b>257</b>
	<b>Index .....</b>	<b>269</b>

Logic Synthesis for Asynchronous Controllers and  
Interfaces

Cortadella, J.; Kishinevsky, M.; Kondratyev, A.; Lavagno,  
L.; Yakovlev, A.

2002, XIII, 273 p., Hardcover

ISBN: 978-3-540-43152-7