

# Contents

<b>1</b>	<b>Introduction</b>	1
1.1	Contributions	3
1.1.1	(A) Decentralized Application Mapping	3
1.1.2	(B) Hybrid Application Mapping	4
1.1.3	(C) Nonfunctional Properties	5
1.2	Outline of this Book	5
	References	6
<b>2</b>	<b>Invasive Computing</b>	9
2.1	Principles of Invasive Computing	9
2.2	Invasive Programming Language	11
2.2.1	Invade, Infect, Retreat, and Claims	12
2.2.2	Communication-Aware Programming	13
2.2.3	Actor Model and Nonfunctional Properties	15
2.3	Overhead Analysis of Invasive Computing	19
2.3.1	Invasive Speedup and Efficiency Analysis	21
2.4	Invasive Hardware Architectures	24
2.4.1	Invasive Tightly Coupled Processor Arrays	25
2.4.2	The Invasive Core— <i>i</i> -Core	27
2.4.3	Dynamic Many-Core <i>i</i> -let Controller—CiC	27
2.5	Invasive Network on Chip— <i>i</i> -NoC	28
2.5.1	Router	30
2.5.2	Invasive Network Adapter— <i>i</i> -NA	31
2.5.3	Control Network Layer	33
2.6	Invasive Run-Time and Operating System	34
2.7	Related Work	35
	References	39

<b>3</b>	<b>Fundamentals</b>	45
3.1	Application Model	45
3.2	System Architecture	47
3.3	Application Mapping	50
3.4	Composability	50
3.5	Predictability	51
3.5.1	*-Predictability	52
	References	55
<b>4</b>	<b>Self-embedding</b>	57
4.1	Self-embedding Algorithm	59
4.2	Incarnations of Embedding Algorithms	63
4.2.1	Path Load and Best Neighbor	64
4.2.2	Random Walk	65
4.2.3	Discussion	67
4.3	Seed-Point Selection	67
4.4	Hardware-Based Acceleration for Self-embedding	68
4.4.1	Application Graph Preprocessing	69
4.4.2	Serialization	70
4.4.3	Protocol	72
4.4.4	Implementation	73
4.5	Experimental Results	74
4.5.1	Simulation Setup	74
4.5.2	Evaluation Metrics	75
4.5.3	Scalability	76
4.5.4	Random Walk with Weighted Probabilities	77
4.5.5	Hardware-Based Self-embedding	79
4.6	Related Work	80
4.7	Summary	81
	References	82
<b>5</b>	<b>Hybrid Application Mapping</b>	85
5.1	HAM Methodology	86
5.2	Static Performance Analysis	91
5.2.1	Composable Communication Scheduling	92
5.2.2	Composable Task Scheduling	94
5.3	Design Space Exploration	96
5.3.1	Generation of Feasible Application Mappings	98
5.3.2	Optimization Objectives and Evaluation	99
5.4	Run-Time Constraint Solving	101
5.4.1	Constraint Graphs	101
5.4.2	Run-Time Mapping of Constraint Graphs	102
5.4.3	Backtracking Algorithm	105
5.4.4	Run-Time Management and System Requirements	106

5.5	Experimental Results . . . . .	111
5.5.1	Comparison Run-Time Management . . . . .	112
5.5.2	MMKP-Based Run-Time Heuristic . . . . .	114
5.5.3	Considering Communication Constraints . . . . .	117
5.5.4	Objectives Related to Embeddability and Communication . . . . .	119
5.5.5	Temporal Isolation Versus Spatial Isolation . . . . .	121
5.5.6	Execution Time . . . . .	123
5.5.7	Case Study . . . . .	125
5.6	Related Work . . . . .	127
5.6.1	Techniques for Static, Dynamic, and Hybrid Application Mapping . . . . .	127
5.6.2	Communication Models in Hybrid Application Mapping . . . . .	128
5.7	Summary . . . . .	132
	References . . . . .	133
<b>6</b>	<b>Hybrid Mapping for Increased Security . . . . .</b>	<b>137</b>
6.1	Hybrid Mapping for Security . . . . .	138
6.1.1	Attacker Model . . . . .	140
6.1.2	Design Methodology . . . . .	141
6.2	Shape-Based Design-Time Optimization . . . . .	142
6.3	Run-Time Mapping . . . . .	145
6.3.1	First-Fit Mapping Heuristic . . . . .	146
6.3.2	SAT-Based Run-Time Mapping . . . . .	147
6.4	Experimental Results . . . . .	148
6.5	Region-Based Run-Time Mapping in the <i>i</i> -NoC . . . . .	151
6.6	Related Work . . . . .	153
6.7	Summary . . . . .	154
	References . . . . .	154
<b>7</b>	<b>Conclusions and Future Work . . . . .</b>	<b>157</b>
7.1	Conclusions . . . . .	157
7.2	Future Research Directions . . . . .	159
	References . . . . .	160
	<b>Index . . . . .</b>	<b>163</b>

Invasive Computing for Mapping Parallel Programs to  
Many-Core Architectures

Weichslgartner, A.; Wildermann, S.; Glaß, M.; Teich, J.

2018, XXII, 164 p. 80 illus., 77 illus. in color., Hardcover

ISBN: 978-981-10-7355-7