

# Contents

<b>Acknowledgements</b>	<b>ix</b>
<b>Contents</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Background . . . . .	4
1.2 Scope . . . . .	6
1.3 Problem Statement and Research Questions . . . . .	6
1.4 Outline of this thesis . . . . .	7
<b>2 Generating Synchronization Statements</b>	<b>11</b>
2.1 Introduction . . . . .	12
2.2 The Satin programming model . . . . .	13
2.3 Problem description . . . . .	16
2.4 Implementation . . . . .	19
2.4.1 Basic algorithm . . . . .	19
2.4.2 Analysis phase . . . . .	20
2.5 Evaluation . . . . .	26
2.5.1 Evaluation per application . . . . .	29
2.6 Discussion . . . . .	31
2.6.1 Precision of the Alias-Analysis . . . . .	32
2.6.2 Improving Sync Generation with Programmer Support . . . . .	34
2.6.3 Cilk . . . . .	35
2.6.4 Futures . . . . .	36
2.7 Programming model design considerations . . . . .	36
2.8 Related Work . . . . .	37
2.9 Conclusion . . . . .	38

<b>3</b>	<b>Stepwise-refinement for performance</b>	<b>41</b>
3.1	Introduction . . . . .	42
3.2	Related work . . . . .	44
3.2.1	Programming Many-cores . . . . .	44
3.2.2	Other related Work . . . . .	47
3.3	Stepwise-refinement for Performance . . . . .	48
3.3.1	Philosophy . . . . .	49
3.3.2	Methodology . . . . .	50
3.4	Design of MCL . . . . .	51
3.4.1	Overview . . . . .	52
3.4.2	Hardware Description Language HDL . . . . .	54
3.4.3	Programming Language MCPL . . . . .	57
3.4.4	Compiler . . . . .	59
3.5	Example: Matrix Multiplication . . . . .	61
3.5.1	GTX480 . . . . .	65
3.5.2	Xeon Phi . . . . .	72
3.5.3	Summary . . . . .	75
3.6	Implementation . . . . .	76
3.6.1	Translation between Abstraction Levels . . . . .	76
3.6.2	Operation Statistics . . . . .	79
3.6.3	Data Reuse Analysis . . . . .	80
3.6.4	Cache Analysis . . . . .	82
3.6.5	Performance Feedback Functions . . . . .	83
3.7	Evaluation . . . . .	84
3.8	Discussion . . . . .	91
3.9	Conclusion . . . . .	92
<b>4</b>	<b>Cashmere: Heterogeneous many-core computing</b>	<b>95</b>
4.1	Introduction . . . . .	96
4.2	Cashmere Programming Model . . . . .	98
4.2.1	Satin . . . . .	98
4.2.2	MCL . . . . .	100
4.2.3	Cashmere programming model . . . . .	100
4.3	Implementation . . . . .	103
4.3.1	MCL . . . . .	103
4.3.2	Cashmere . . . . .	104
4.4	Methodology . . . . .	106
4.5	Evaluation . . . . .	109
4.5.1	Kernel performance . . . . .	109

---

4.5.2 Scalability . . . . .	110
4.5.3 Heterogeneity . . . . .	111
4.6 Related Work . . . . .	117
4.7 Conclusion . . . . .	120
<b>5 Conclusions</b>	<b>123</b>
5.1 Summary . . . . .	123
5.2 Future Directions . . . . .	126
5.2.1 Conclusions . . . . .	128
<b>A Many-Core Levels Language Descriptions</b>	<b>129</b>
A.1 Hardware Description Language HDL . . . . .	129
A.2 Programming Language . . . . .	134
<b>B Translating to a lower level of abstraction</b>	<b>139</b>
B.1 The top-level functions . . . . .	140
B.2 Finding equivalent ParGroups . . . . .	141
B.3 Translating memory spaces . . . . .	142
B.4 Translating ForEach statements . . . . .	144
<b>References</b>	<b>151</b>
<b>List of Publications</b>	<b>163</b>
<b>Samenvatting</b>	<b>165</b>