
Contents

1	Introduction	1
1.1	Hardware Description Languages	1
1.2	Hardware Synthesis	7
1.2.1	High-Level Synthesis	8
1.3	Motivation for Higher Level Tools	14
1.3.1	Lack of Structuring Support	14
1.3.2	Limitations of Static Scheduling	15
1.4	Structure of the Monograph	16
2	Related Work	19
2.1	Verilog and VHDL	19
2.2	The Olympus Synthesis System	23
2.2.1	The HardwareC Language	23
2.2.2	Hercules	25
2.2.3	Hebe	25
2.3	Functional Languages	26
2.3.1	μ FP: An Algebra for VLSI Specification	26
2.3.2	Embedding HDLs in General-Purpose Functional Languages	28
2.4	Term Rewriting Systems	30
2.5	Occam/CSP-Based Approaches	31
2.5.1	Handel and Handel-C	31
2.5.2	Tangram and Balsa	31
2.6	Synchronous Languages	33
2.7	Summary	34
3	The SAFL Language	35
3.1	Motivation	35
3.2	Language Definition	36
3.2.1	Static Allocation	37
3.2.2	Integrating with External Hardware Components	37
3.2.3	Semantics	38

3.2.4	Concrete Syntax	38
3.3	Hardware Synthesis Using SAFL	41
3.3.1	Automatic Generation of Parallel Hardware	42
3.3.2	Resource Awareness	42
3.3.3	Source-Level Program Transformation	44
3.3.4	Static Analysis and Optimisation	47
3.3.5	Architecture Independence	48
3.4	Aside: Dealing with Mutual Recursion	48
3.4.1	Eliminating Mutual Recursion by Transformation	49
3.5	Related Work	50
3.6	Summary	50
4	Soft Scheduling	51
4.1	Motivation and Related Work	52
4.1.1	Translating SAFL to Hardware	54
4.2	Soft Scheduling: Technical Details	55
4.2.1	Removing Redundant Arbiters	56
4.2.2	Parallel Conflict Analysis (PCA)	56
4.2.3	Integrating PCA into the FLaSH Compiler	58
4.3	Examples and Discussion	58
4.3.1	Parallel FIR Filter	58
4.3.2	Shared-Memory Multi-processor Architecture	59
4.3.3	Parallel Tasks Sharing Graphical Display	61
4.4	Program Transformation for Scheduling and Binding	62
4.5	Summary	63
5	High-Level Synthesis of SAFL	65
5.1	FLaSH Intermediate Code	66
5.1.1	The Structure of Intermediate Graphs	67
5.1.2	Translation to Intermediate Code	71
5.2	Translation to Synchronous Hardware	73
5.2.1	Compiling Expressions	73
5.2.2	Compiling Functions	75
5.2.3	Generated Verilog	79
5.2.4	Compiling External Functions	80
5.3	Translation to GALS Hardware	81
5.3.1	A Brief Discussion of Metastability	81
5.3.2	Interfacing between Different Clock Domains	83
5.3.3	Modifying the Arbitration Circuitry	85
5.4	Summary	86
6	Analysis and Optimisation of Intermediate Code	87
6.1	Architecture-Neutral versus Architecture-Specific	87
6.2	Definitions and Terminology	88
6.3	Register Placement Analysis and Optimisation	88
6.3.1	Sharing Conflicts	89

6.3.2	Technical Details	91
6.3.3	Resource Dependency Analysis	92
6.3.4	Data Validity Analysis	93
6.3.5	Sequential Conflict Register Placement	95
6.4	Extending the Model: Calling Conventions	97
6.4.1	Caller-Save Resource Dependency Analysis	97
6.4.2	Caller-Save Permanisation Analysis	99
6.5	Synchronous Timing Analysis	99
6.5.1	Technical Details	100
6.5.2	Associated Optimisations	101
6.6	Results and Discussion	104
6.6.1	Register Placement Analysis: Results	104
6.6.2	Synchronous Timing Optimisations: Results	109
6.7	Summary	110
7	Dealing with I/O	113
7.1	SAFL+ Language Description	113
7.1.1	Resource Awareness	115
7.1.2	Channels and Channel Passing	115
7.1.3	The Motivation for Channel Passing	117
7.2	Translating SAFL+ to Hardware	118
7.2.1	Extending Analyses from SAFL to SAFL+	120
7.3	Operational Semantics for SAFL+	121
7.3.1	Transition Rules	124
7.3.2	Semantics for Channel Passing	124
7.3.3	Non-determinism	126
7.4	Summary	126
8	Combining Behaviour and Structure	129
8.1	Motivation and Related Work	129
8.2	Embedding Structural Expansion in SAFL	130
8.2.1	Building Combinatorial Hardware in Magma	130
8.2.2	Integrating SAFL and Magma	134
8.3	Aside: Embedding Magma in VHDL/Verilog	136
8.4	Summary	138
9	Transformation of SAFL Specifications	141
9.1	Hardware Software CoDesign	142
9.1.1	Comparison with Other Work	142
9.2	Technical Details	143
9.2.1	The Stack Machine Template	144
9.2.2	Stack Machine Instances	144
9.2.3	Compilation to Stack Code	146
9.2.4	The Partitioning Transformation	148
9.2.5	Validity of Partitioning Functions	148
9.2.6	Extensions	149

XII Contents

9.3	Transformations from SAFL to SAFL+	151
9.4	Summary	153
10	Case Study	155
10.1	The SAFL to Silicon Tool Chain	155
10.2	DES Encrypter/Decrypter	160
10.2.1	Adding Hardware VGA Support	162
10.3	Summary	167
11	Conclusions and Further Work	169
11.1	Future Work	170
Appendix		
A	DES Encryption/Decryption Circuit	171
B	Transformations to Pipeline DES	177
C	A Simple Stack Machine and Instruction Memory	181
References		185
Index		193



<http://www.springer.com/978-3-540-21306-2>

Higher-Level Hardware Synthesis

Sharp, R.

2004, XVI, 196 p., Softcover

ISBN: 978-3-540-21306-2