Contents

Preface vii

1 Controlling complexity 1
  1.1 Hierarchical design flow 1
  1.2 Designing hardware with software 4
  1.3 Summary 6

2 A Verilogical place to start 7
  2.1 My Veri first description 7
  2.2 A more formal introduction to the basics 9
  2.3 Behavioral and structural models 16
  2.4 Functions and tasks 28
  2.5 Summary 30

3 Defining the instruction set architecture 32
  3.1 Instruction set design 32
  3.2 Defining the VeSPA instruction set 35
  3.3 Specifying the VeSPA ISA 48
  3.4 Summary 56

4 Algorithmic behavioral modeling 58
  4.1 Module definition 59
  4.2 Instruction and storage element definitions 59
  4.3 Fetch-execute loop 64
  4.4 Fetch task 68
  4.5 Execute task 71
  4.6 Condition code tasks 77
  4.7 Tracing instruction execution 79
  4.8 Summary 81

5 Building an assembler for VeSPA 82
  5.1 Why assembly language? 82
  5.2 The assembly process 83
Table of Contents

5.3 VASM – the VeSPA assembler 88
5.4 Linking and loading 92
5.5 Summary 92

6 Pipelining 94
6.1 Instruction partitioning for pipelining 94
6.2 Pipeline performance 96
6.3 Dependences and hazards 97
6.4 Dealing with pipeline hazards 103
6.5 Summary 104
Further reading 104

7 Implementation of the pipelined processor 105
7.1 Pipelining VeSPA 105
7.2 The hazard detection unit 106
7.3 Overview of the pipeline structure 108
7.4 A detailed description of the pipeline stages 109
7.5 Timing considerations 115
7.6 Summary 117
Further reading 117

8 Verification 118
8.1 Component-level test benches 118
8.2 System-level self-testing 127
8.3 Formal verification 130
8.4 Summary 131
Further reading 131

A The VeSPA instruction set architecture (ISA) 132
A.1 Notational conventions 132
A.2 Storage elements 133
A.3 The instruction specifications 133

B The VASM assembler 147
B.1 Notational conventions 147
B.2 Assembler directives 148
B.3 Example program 149
B.4 Modifying the assembler 149
Further reading 152

Index 153