Advanced Digital Design with the Verilog HDL

Michael D. Ciletti
Department of Electrical and Computer Engineering
University of Colorado at Colorado Springs

Prentice Hall
Contents

Preface xiii
Simplify, Clarify, and Verify xiii

1 Introduction to Digital Design Methodology 1
1.1 Design Methodology—An Introduction 2
  1.1.1 Design Specification 4
  1.1.2 Design Partition 4
  1.1.3 Design Entry 4
  1.1.4 Simulation and Functional Verification 5
  1.1.5 Design Integration and Verification 6
  1.1.6 Presynthesis Sign-Off 6
  1.1.7 Gate-Level Synthesis and Technology Mapping 6
  1.1.8 Postsynthesis Design Validation 7
  1.1.9 Postsynthesis Timing Verification 8
  1.1.10 Test Generation and Fault Simulation 8
  1.1.11 Placement and Routing 8
  1.1.12 Physical and Electrical Design Rule Checks 9
  1.1.13 Parasitic Extraction 9
  1.1.14 Design Sign-Off 9
1.2 IC Technology Options 9
1.3 Overview 11
References 11

2 Review of Combinational Logic Design 13
2.1 Combinational Logic and Boolean Algebra 13
  2.1.1 ASIC Library Cells 13
  2.1.2 Boolean Algebra 16
  2.1.3 DeMorgan's Laws 18
2.2 Theorems for Boolean Algebraic Minimization 18
2.3 Representation of Combinational Logic 21
  2.3.1 Sum-of-Products Representation 23
  2.3.2 Product-of-Sums Representation 26
2.4 Simplification of Boolean Expressions 27
  2.4.1 Simplification with Exclusive-Or 36
  2.4.2 Karnaugh Maps (SOP Form) 36
  2.4.3 Karnaugh Maps (POS Form) 39
  2.4.4 Karnaugh Maps and Don't-Cares 40
  2.4.5 Extended Karnaugh Maps 41
2.5 Glitches and Hazards 42
  2.5.1 Elimination of Static Hazards (SOP Form) 45
  2.5.2 Summary: Elimination of Static Hazards in Two-Level Circuits 48
  2.5.3 Static Hazards in Multilevel Circuits 49
  2.5.4 Summary: Elimination of Static Hazards in Multilevel Circuits 52
  2.5.5 Dynamic Hazards 52
2.6 Building Blocks for Logic Design 55
  2.6.1 NAND-NOR Structures 55
  2.6.2 Multiplexers 60
  2.6.3 Demultiplexers 61
  2.6.4 Encoders 62
  2.6.5 Priority Encoder 63
  2.6.6 Decoder 64
  2.6.7 Priority Decoder 66
References 67
Problems 67

3 Fundamentals of Sequential Logic Design 69
3.1 Storage Elements 69
  3.1.1 Latches 70
  3.1.2 Transparent Latches 71
3.2 Flip-Flops 71
  3.2.1 D-Type Flip-Flop 71
  3.2.2 Master–Slave Flip-Flop 73
  3.2.3 J-K Flip-Flops 75
  3.2.4 T Flip-Flop 75
3.3 Busses and Three-State Devices 76
3.4 Design of Sequential Machines 80
3.5 State-Transition Graphs 82
3.6 Design Example: BCD to Excess-3 Code Converter 84
3.7 Serial-Line Code Converter for Data Transmission 90
  3.7.1 Design Example: A Mealy-Type FSM for Serial Line-Code Conversion 92
  3.7.2 Design Example: A Moore-Type FSM for Serial Line-Code Conversion 93
## Contents

3.8 State Reduction and Equivalent States 95  
References 99  
Problems 100  

4 Introduction to Logic Design with Verilog 103  
4.1 Structural Models of Combinational Logic 104  
4.1.1 Verilog Primitives and Design Encapsulation 104  
4.1.2 Verilog Structural Models 107  
4.1.3 Module Ports 108  
4.1.4 Some Language Rules 108  
4.1.5 Top-Down Design and Nested Modules 109  
4.1.6 Design Hierarchy and Source-Code Organization 111  
4.1.7 Vectors in Verilog 113  
4.1.8 Structural Connectivity 114  
4.2 Logic System, Design Verification, and Test Methodology 118  
4.2.1 Four-Value Logic and Signal Resolution in Verilog 119  
4.2.2 Test Methodology 120  
4.2.3 Signal Generators for Testbenches 123  
4.2.4 Event-Driven Simulation 125  
4.2.5 Testbench Template 125  
4.2.6 Sized Numbers 126  
4.3 Propagation Delay 126  
4.3.1 Inertial Delay 129  
4.3.2 Transport Delay 131  
4.4 Truth Table Models of Combinational and Sequential Logic with Verilog 131  
References 138  
Problems 138  

5 Logic Design with Behavioral Models of Combinational and Sequential Logic 141  
5.1 Behavioral Modeling 141  
5.2 A Brief Look at Data Types for Behavioral Modeling 143  
5.3 Boolean Equation-Based Behavioral Models of Combinational Logic 143  
5.4 Propagation Delay and Continuous Assignments 146  
5.5 Latches and Level-Sensitive Circuits in Verilog 148  
5.6 Cyclic Behavioral Models of Flip-Flops and Latches 150  
5.7 Cyclic Behavior and Edge Detection 152  
5.8 A Comparison of Styles for Behavioral Modeling 154  
5.8.1 Continuous Assignment Models 154  
5.8.2 Dataflow/RTL Models 156  
5.8.3 Algorithm-Based Models 160  
5.8.4 Naming Conventions: A Matter of Style 161  
5.8.5 Simulation with Behavioral Models 162  
5.9 Behavioral Models of Multiplexers, Encoders, and Decoders 162
5.10 Dataflow Models of a Linear-Feedback Shift Register 171
5.11 Modeling Digital Machines with Repetitive Algorithms 173
  5.11.1 Intellectual Property Reuse and Parameterized Models 178
  5.11.2 Clock Generators 180
5.12 Machines with Multicycle Operations 182
5.13 Design Documentation with Functions and Tasks: Legacy or Lunacy? 183
  5.13.1 Tasks 184
  5.13.2 Functions 185
5.14 Algorithmic State Machine Charts for Behavioral Modeling 187
5.15 ASMD Charts 191
5.16 Behavioral Models of Counters, Shift Registers, and Register Files 195
  5.16.1 Counters 195
  5.16.2 Shift Registers 202
  5.16.3 Register Files and Arrays of Registers (Memories) 206
5.17 Switch Debounce, Metastability, and Synchronizers for Asynchronous Signals 208
5.18 Design Example: Keypad Scanner and Encoder 214
References 223
Problems 223

6 Synthesis of Combinational and Sequential Logic 235
6.1 Introduction to Synthesis 236
  6.1.1 Logic Synthesis 237
  6.1.2 RTL Synthesis 245
  6.1.3 High-Level Synthesis 246
6.2 Synthesis of Combinational Logic 247
  6.2.1 Synthesis of Priority Structures 252
  6.2.2 Exploiting Logical Don't-Care Conditions 253
  6.2.3 ASIC Cells and Resource Sharing 258
6.3 Synthesis of Sequential Logic with Latches 260
  6.3.1 Accidental Synthesis of Latches 262
  6.3.2 Intentional Synthesis of Latches 266
6.4 Synthesis of Three-State Devices and Bus Interfaces 269
6.5 Synthesis of Sequential Logic with Flip-Flops 272
6.6 Synthesis of Explicit State Machines 275
  6.6.1 Synthesis of a BCD-to-Excess-3 Code Converter 276
  6.6.2 Design Example: Synthesis of a Mealy-Type NRZ-to-Manchester Line Code Converter 281
  6.6.3 Design Example: Synthesis of a Moore-Type NRZ-to-Manchester Line Code Converter 283
  6.6.4 Design Example: Synthesis of a Sequence Recognizer 284
6.7 Registered Logic 292
6.8 State Encoding 300
8.2.3 Erasable ROMs 421
8.2.4 ROM-Based Implementation of Combinational Logic 423
8.2.5 Verilog System Tasks for ROMs 423
8.2.6 Comparison of ROMs 426
8.2.7 ROM-Based State Machines 426
8.2.8 Flash Memory 430
8.2.9 Static Random Access Memory (SRAM) 430
8.2.10 Ferroelectric Nonvolatile Memory 452
8.3 Programmable Logic Array (PLA) 454
  8.3.1 PLA Minimization 457
  8.3.2 PLA Modeling 459
8.4 Programmable Array Logic (PAL) 463
8.5 Programmability of PLDs 464
8.6 Complex PLDs (CPLDs) 465
8.7 Field-Programmable Gate Arrays 466
  8.7.1 The Role of FPGAs in the ASIC Market 467
  8.7.2 FPGA Technologies 469
  8.7.3 XILINX Virtex FPGAs 470
8.8 Embeddable and Programmable IP Cores for a System-on-a-Chip (SoC) 470
8.9 Verilog-Based Design Flows for FPGAs 472
8.10 Synthesis with FPGAs 473
References 476
Related Web Sites 476
Problems and FPGA-Based Design Exercises 476

9 Algorithms and Architectures for Digital Processors 515
9.1 Algorithms, Nested-Loop Programs, and Data Flow Graphs 516
9.2 Design Example: Halftone Pixel Image Converter 519
  9.2.1 Baseline Design for a Halftone Pixel Image Converter 522
  9.2.2 NLP-Based Architectures for the Halftone Pixel Image Converter 526
  9.2.3 Minimum Concurrent Processor Architecture for a Halftone Pixel Image Converter 532
  9.2.4 Halftone Pixel Image Converter: Design Tradeoffs 547
  9.2.5 Architectures for Dataflow Graphs with Feedback 547
9.3 Digital Filters and Signal Processors 554
  9.3.1 Finite-Duration Impulse Response Filter 557
  9.3.2 Digital Filter Design Process 558
  9.3.3 Infinite-Duration Impulse Response Filter 563
9.4 Building Blocks for Signal Processors 566
  9.4.1 Integrators (Accumulators) 566
  9.4.2 Differentiators 570
  9.4.3 Decimation and Interpolation Filters 570
Contents ix

9.5 Pipelined Architectures 576
  9.5.1 Design Example: Pipelined Adder 579
  9.5.2 Design Example: Pipelined FIR Filter 583
9.6 Circular Buffers 586
9.7 Asynchronous FIFOs—Synchronization across Clock Domains 589
  9.7.1 Simplified Asynchronous FIFO 590
  9.7.2 Clock Domain Synchronization for an Asynchronous FIFO 599
References 619
Problems 620

10 Architectures for Arithmetic Processors 627
  10.1 Number Representation 627
    10.1.1 Signed Magnitude Representation of Negative Integers 628
    10.1.2 Ones Complement Representation of Negative Integers 629
    10.1.3 Twos Complement Representation of Positive and Negative Integers 630
    10.1.4 Representation of Fractions 632
  10.2 Functional Units for Addition and Subtraction 632
    10.2.1 Ripple-Carry Adder 632
    10.2.2 Carry Look-Ahead Adder 633
    10.2.3 Overflow and Underflow 638
  10.3 Functional Units for Multiplication 638
    10.3.1 Combinational (Parallel) Binary Multiplier 639
    10.3.2 Sequential Binary Multiplier 642
    10.3.3 Sequential Multiplier Design: Hierarchical Decomposition 644
    10.3.4 STG-Based Controller Design 646
    10.3.5 Efficient STG-Based Sequential Binary Multiplier 652
    10.3.6 ASMD-Based Sequential Binary Multiplier 658
    10.3.7 Efficient ASMD-Based Sequential Binary Multiplier 664
    10.3.8 Summary of ASMD-Based Datapath and Controller Design 669
    10.3.9 Reduced-Register Sequential Multiplier 670
    10.3.10 Implicit-State-Machine Binary Multiplier 675
    10.3.11 Booth’s Algorithm Sequential Multiplier 687
    10.3.12 Bit-Pair Encoding 702
  10.4 Multiplication of Signed Binary Numbers 710
    10.4.1 Product of Signed Numbers: Negative Multiplicand, Positive Multiplier 710
    10.4.2 Product of Signed Numbers: Positive Multiplicand, Negative Multiplier 710
    10.4.3 Product of Signed Numbers: Negative Multiplicand, Negative Multiplier 710
  10.5 Multiplication of Fractions 711
    10.5.1 Signed Fractions: Positive Multiplicand, Positive Multiplier 714
    10.5.2 Signed Fractions: Negative Multiplicand, Positive Multiplier 714
    10.5.3 Signed Fractions: Positive Multiplicand, Negative Multiplier 714
    10.5.4 Signed Fractions: Negative Multiplicand, Negative Multiplier 715
10.6 Functional Units for Division 715
   10.6.1 Division of Unsigned Binary Numbers 716
   10.6.2 Efficient Division of Unsigned Binary Numbers 724
   10.6.3 Reduced-Register Sequential Divider 734
   10.6.4 Division of Signed (2s Complement) Binary Numbers 739
   10.6.5 Signed Arithmetic 739

References 742
Problems 742

11 Postsynthesis Design Tasks 749
   11.1 Postsynthesis Design Validation 749
   11.2 Postsynthesis Timing Verification 753
      11.2.1 Static Timing Analysis 755
      11.2.2 Timing Specifications 757
      11.2.3 Factors That Affect Timing 760
   11.3 Elimination of ASIC Timing Violations 766
   11.4 False Paths 767
   11.5 System Tasks for Timing Verification 769
      11.5.1 Timing Check: Setup Condition 770
      11.5.2 Timing Check: Hold Condition 770
      11.5.3 Timing Check: Setup and Hold Conditions 771
      11.5.4 Timing Check: Pulsewidth Constraint 773
      11.5.5 Timing Check: Signal Skew Constraint 773
      11.5.6 Timing Check: Clock Period 774
      11.5.7 Timing Check: Recovery Time 774
   11.6 Fault Simulation and Manufacturing Tests 775
      11.6.1 Circuit Defects and Faults 776
      11.6.2 Fault Detection and Testing 780
      11.6.3 D-Notation 782
      11.6.4 Automatic Test Pattern Generation for Combinational Circuits 786
      11.6.5 Fault Coverage and Defect Levels 788
      11.6.6 Test Generation for Sequential Circuits 788
   11.7 Fault Simulation 792
      11.7.1 Fault Collapsing 793
      11.7.2 Serial Fault Simulation 793
      11.7.3 Parallel Fault Simulation 794
      11.7.4 Concurrent Fault Simulation 794
      11.7.5 Probabilistic Fault Simulation 794
   11.8 JTAG Ports and Design for Testability 794
      11.8.1 Boundary Scan and JTAG Ports 795
      11.8.2 JTAG Modes of Operation 796
11.8.3 JTAG Registers 798
11.8.4 JTAG Instructions 800
11.8.5 TAP Architecture 801
11.8.6 TAP Controller State Machine 803
11.8.7 Design Example: Testing with JTAG 807
11.8.8 Design Example: Built-In Self-Test 830

References 845
Problems 845

A Verilog Primitives 851
A.1 Multiinput Combinational Logic Gates 851
A.2 Multioutput Combinational Gates 853
A.3 Three-State Logic Gates 854
A.4 MOS Transistor Switches 855
A.5 MOS Pull-Up/Pull-Down Gates 860
A.6 MOS Bidirectional Switches 860

B Verilog Keywords 863

C Verilog Data Types 865
C.1 Nets 865
C.2 Register Variables 866
C.3 Constants 870
C.4 Referencing Arrays of Nets or Regs 871

D Verilog Operators 873
D.1 Arithmetic Operators 873
D.2 Bitwise Operators 875
D.3 Reduction Operators 875
D.4 Logical Operators 876
D.5 Relational Operators 877
D.6 Shift Operators 878
D.7 Conditional Operator 878
D.8 Concatenation Operator 879
D.9 Expressions and Operands 880
D.10 Operator Precedence 880
D.11 Arithmetic with Signed Data Types 881
D.12 Signed Literal Integers 882
D.13 System Functions for Sign Conversion 882
2.1.1 Assignment Width Extension 883

E Verilog Language Formal Syntax 885