# CONTENTS

**Preface**  
*page xv*

**Acknowledgments**  
*xx*

## Part I Introduction

1. **The digital abstraction**  
*page 3*

   1.1 Digital signals  
   *page 3*

   1.2 Digital signals tolerate noise  
   *page 5*

   1.3 Digital signals represent complex data  
   1.3.1 Representing the day of the year  
   *page 10*

   1.3.2 Representing subtractive colors  
   *page 11*

   1.4 Digital logic functions  
   *page 12*

   1.5 Verilog description of digital circuits and systems  
   *page 14*

   1.6 Digital logic in systems  
   *page 15*

   Summary  
   *page 16*

   Bibliographic notes  
   *page 17*

   Exercises  
   *page 17*

2. **The practice of digital system design**  
*page 21*

   2.1 The design process  
   *page 21*

   2.1.1 Specification  
   *page 22*

   2.1.2 Concept development and feasibility  
   *page 23*

   2.1.3 Partitioning and detailed design  
   *page 26*

   2.1.4 Verification  
   *page 26*

   2.2 Digital systems are built from chips and boards  
   *page 27*

   2.3 Computer-aided design tools  
   *page 32*

   2.4 Moore’s law and digital system evolution  
   *page 33*

   Summary  
   *page 35*

   Bibliographic notes  
   *page 36*

   Exercises  
   *page 36*
# Table of Contents

## Part II Combinational logic

3 **Boolean algebra**  
3.1 Axioms 41  
3.2 Properties 42  
3.3 Dual functions 44  
3.4 Normal form 45  
3.5 From equations to gates 47  
3.6 Boolean expressions in Verilog 49  
Summary 51  
Bibliographic notes 52  
Exercises 52

4 **CMOS logic circuits**  
4.1 Switch logic 55  
4.2 Switch model of MOS transistors 59  
4.3 CMOS gate circuits 66  
\hspace{1em} 4.3.1 Basic CMOS gate circuit 66  
\hspace{1em} 4.3.2 Inverters, NANDs, and NORs 67  
\hspace{1em} 4.3.3 Complex gates 69  
\hspace{1em} 4.3.4 Tri-state circuits 72  
\hspace{1em} 4.3.5 Circuits to avoid 74  
Summary 75  
Bibliographic notes 75  
Exercises 76

5 **Delay and power of CMOS circuits**  
5.1 Delay of static CMOS gates 79  
5.2 Fan-out and driving large loads 82  
5.3 Fan-in and logical effort 84  
5.4 Delay calculation 87  
5.5 Optimizing delay 89  
5.6 Wire delay 92  
5.7 Power dissipation in CMOS circuits 95  
\hspace{1em} 5.7.1 Dynamic power 96  
\hspace{1em} 5.7.2 Static power 97  
\hspace{1em} 5.7.3 Power scaling 97  
Summary 98  
Bibliographic notes 99  
Exercises 99
## 6 Combinational logic design

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Combinational logic</td>
<td>103</td>
</tr>
<tr>
<td>6.2 Closure</td>
<td>103</td>
</tr>
<tr>
<td>6.3 Truth tables, minterms, and normal form</td>
<td>104</td>
</tr>
<tr>
<td>6.4 Implicants and cubes</td>
<td>105</td>
</tr>
<tr>
<td>6.5 Karnaugh maps</td>
<td>108</td>
</tr>
<tr>
<td>6.6 Covering a function</td>
<td>112</td>
</tr>
<tr>
<td>6.7 From a cover to gates</td>
<td>114</td>
</tr>
<tr>
<td>6.8 Incompletely specified functions</td>
<td>115</td>
</tr>
<tr>
<td>6.9 Product-of-sums implementation</td>
<td>116</td>
</tr>
<tr>
<td>6.10 Hazards</td>
<td>118</td>
</tr>
<tr>
<td><strong>Summary</strong></td>
<td>120</td>
</tr>
<tr>
<td><strong>Bibliographic notes</strong></td>
<td>122</td>
</tr>
<tr>
<td><strong>Exercises</strong></td>
<td>123</td>
</tr>
</tbody>
</table>

## 7 Verilog descriptions of combinational logic

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1 The prime number circuit in Verilog</td>
<td>128</td>
</tr>
<tr>
<td>7.1.1 A Verilog module</td>
<td>129</td>
</tr>
<tr>
<td>7.1.2 The case statement</td>
<td>130</td>
</tr>
<tr>
<td>7.1.3 The casex statement</td>
<td>132</td>
</tr>
<tr>
<td>7.1.4 The assign statement</td>
<td>133</td>
</tr>
<tr>
<td>7.1.5 Structural description</td>
<td>134</td>
</tr>
<tr>
<td>7.1.6 The decimal prime number function</td>
<td>136</td>
</tr>
<tr>
<td>7.2 A testbench for the prime number circuit</td>
<td>137</td>
</tr>
<tr>
<td>7.3 Example: a seven-segment decoder</td>
<td>141</td>
</tr>
<tr>
<td><strong>Summary</strong></td>
<td>146</td>
</tr>
<tr>
<td><strong>Bibliographic notes</strong></td>
<td>146</td>
</tr>
<tr>
<td><strong>Exercises</strong></td>
<td>147</td>
</tr>
</tbody>
</table>

## 8 Combinational building blocks

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1 Multi-bit notation</td>
<td>150</td>
</tr>
<tr>
<td>8.2 Decoders</td>
<td>150</td>
</tr>
<tr>
<td>8.3 Multiplexers</td>
<td>155</td>
</tr>
<tr>
<td>8.4 Encoders</td>
<td>161</td>
</tr>
<tr>
<td>8.5 Arbiters and priority encoders</td>
<td>165</td>
</tr>
<tr>
<td>8.6 Comparators</td>
<td>170</td>
</tr>
<tr>
<td>8.7 Shifters</td>
<td>173</td>
</tr>
<tr>
<td>8.8 Read-only memories</td>
<td>173</td>
</tr>
<tr>
<td>8.9 Read-write memories</td>
<td>177</td>
</tr>
<tr>
<td>8.10 Programmable logic arrays</td>
<td>180</td>
</tr>
<tr>
<td>8.11 Data sheets</td>
<td>181</td>
</tr>
<tr>
<td>8.12 Intellectual property</td>
<td>182</td>
</tr>
</tbody>
</table>
## Contents

Summary 183  
Bibliographic notes 184  
Exercises 184  

9 Combinational examples 187  
9.1 Multiple-of-3 circuit 187  
9.2 Tomorrow circuit 189  
9.3 Priority arbiter 192  
9.4 Tic-tac-toe 193  
Summary 201  
Exercises 202  

Part III Arithmetic circuits

10 Arithmetic circuits 207  
10.1 Binary numbers 207  
10.2 Binary addition 210  
10.3 Negative numbers and subtraction 216  
10.4 Multiplication 223  
10.5 Division 226  
Summary 230  
Exercises 230  

11 Fixed- and floating-point numbers 236  
11.1 Representation error: accuracy, precision, and resolution 236  
11.2 Fixed-point numbers 238  
\hspace{1em} 11.2.1 Representation 238  
\hspace{1em} 11.2.2 Operations 241  
11.3 Floating-point numbers 243  
\hspace{1em} 11.3.1 Representation 243  
\hspace{1em} 11.3.2 Denormalized numbers and gradual underflow 245  
\hspace{1em} 11.3.3 Floating-point multiplication 245  
\hspace{1em} 11.3.4 Floating-point addition/subtraction 247  
Summary 250  
Bibliographic note 251  
Exercises 251  

12 Fast arithmetic circuits 255  
12.1 Carry look-ahead 255  
12.2 Booth recoding 260  
12.3 Wallace trees 265  
12.4 Synthesis notes 269
## Contents

Summary  
Bibliographic notes  
Exercises

### 13 Arithmetic examples

13.1 Complex multiplication  
13.2 Converting between fixed- and floating-point formats  
13.2.1 Floating-point format  
13.2.2 Fixed- to floating-point conversion  
13.2.3 Floating- to fixed-point conversion  
13.3 FIR filter  
Summary  
Bibliographic note  
Exercises

### Part IV Synchronous sequential logic

### 14 Sequential logic

14.1 Sequential circuits  
14.2 Synchronous sequential circuits  
14.3 Traffic-light controller  
14.4 State assignment  
14.5 Implementation of finite-state machines  
14.6 Verilog implementation of finite-state machines  
Summary  
Bibliographic notes  
Exercises

### 15 Timing constraints

15.1 Propagation and contamination delay  
15.2 The D flip-flop  
15.3 Setup- and hold-time constraints  
15.4 The effect of clock skew  
15.5 Timing examples  
15.6 Timing and logic synthesis  
Summary  
Bibliographic notes  
Exercises

### 16 Datapath sequential logic

16.1 Counters  
16.1.1 A simpler counter
## Contents

16.1.2 Up/down/load counter 333  
16.1.3 A timer 336  
16.2 Shift registers 338  
16.2.1 A simple shift register 338  
16.2.2 Left/right/load (LRL) shift register 339  
16.2.3 Universal shifter/counter 340  
16.3 Control and data partitioning 342  
16.3.1 Example: vending machine FSM 342  
16.3.2 Example: combination lock 348  
Summary 356  
Exercises 357  

17 Factoring finite-state machines 360  
17.1 A light flasher 360  
17.2 Traffic-light controller 367  
Summary 378  
Exercises 378  

18 Microcode 383  
18.1 Simple microcoded FSM 383  
18.2 Instruction sequencing 387  
18.3 Multi-way branches 394  
18.4 Multiple instruction types 397  
18.5 Microcode subroutines 400  
18.6 Simple computer 403  
Summary 406  
Bibliographic notes 408  
Exercises 411  

19 Sequential examples 414  
19.1 Divide-by-3 counter 414  
19.2 SOS detector 415  
19.3 Tic-tac-toe game 422  
19.4 Huffman encoder/decoder 422  
19.4.1 Huffman encoder 423  
19.4.2 Huffman decoder 427  
Summary 430  
Bibliographic note 430  
Exercises 430
Part V Practical design

20 Verification and test

20.1 Design verification

20.1.1 Verification coverage

20.1.2 Types of tests

20.1.3 Static timing analysis

20.1.4 Formal verification

20.1.5 Bug tracking

20.2 Test

20.2.1 Fault models

20.2.2 Combinational testing

20.2.3 Testing redundant logic

20.2.4 Scan

20.2.5 Built-in-self-test (BIST)

20.2.6 Characterization

Summary

Exercises

Part VI System design

21 System-level design

21.1 System design process

21.2 Specification

21.2.1 Pong

21.2.2 DES cracker

21.2.3 Music player

21.3 Partitioning

21.3.1 Pong

21.3.2 DES cracker

21.3.3 Music synthesizer

Summary

Exercises

22 Interface and system-level timing

22.1 Interface timing

22.1.1 Always valid timing

22.1.2 Periodically valid signals

22.1.3 Flow control
## Contents

22.2 Interface partitioning and selection 465  
22.3 Serial and packetized interfaces 465  
22.4 Isochronous timing 468  
22.5 Timing tables 469  
  22.5.1 Event flow 470  
  22.5.2 Pipelining and anticipatory timing 471  
22.6 Interface and timing examples 471  
  22.6.1 Pong 471  
  22.6.2 DES cracker 472  
  22.6.3 Music player 473  
Summary 476  
Exercises 476  

### 23 Pipelines

23.1 Basic pipelining 479  
23.2 Example pipelines 482  
23.3 Example: pipelining a ripple-carry adder 484  
23.4 Pipeline stalls 488  
23.5 Double buffering 489  
23.6 Load balance 494  
23.7 Variable loads 494  
23.8 Resource sharing 499  
Summary 500  
Bibliographic notes 500  
Exercises 501  

### 24 Interconnect

24.1 Abstract interconnect 504  
24.2 Buses 505  
24.3 Crossbar switches 507  
24.4 Interconnection networks 510  
Summary 512  
Bibliographic notes 512  
Exercises 513  

### 25 Memory systems

25.1 Memory primitives 515  
  25.1.1 SRAM arrays 515  
  25.1.2 DRAM chips 517  
25.2 Bit-slicing and banking memory 519  
25.3 Interleaved memory 521  
25.4 Caches 524  
Summary 528
Part VII  Asynchronous logic

26  Asynchronous sequential circuits  
26.1 Flow-table analysis  
26.2 Flow-table synthesis: the toggle circuit  
26.3 Races and state assignment  
Summary  
Bibliographic notes  
Exercises  

27  Flip-flops  
27.1 Inside a latch  
27.2 Inside a flip-flop  
27.3 CMOS latches and flip-flops  
27.4 Flow-table derivation of the latch  
27.5 Flow-table synthesis of a D-flip-flop  
Summary  
Bibliographic notes  
Exercises  

28  Metastability and synchronization failure  
28.1 Synchronization failure  
28.2 Metastability  
28.3 Probability of entering and leaving an illegal state  
28.4 Demonstration of metastability  
Summary  
Bibliographic notes  
Exercises  

29  Synchronizer design  
29.1 Where are synchronizers used?  
29.2 Brute-force synchronizer  
29.3 The problem with multi-bit signals  
29.4 FIFO synchronizer  
Summary  
Bibliographic notes  
Exercises  
# Contents

**Appendix A: Verilog coding style**

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.1 Basic principles</td>
<td>592</td>
</tr>
<tr>
<td>A.2 All state should be in explicitly declared registers</td>
<td>593</td>
</tr>
<tr>
<td>A.3 Define combinational modules so they are easy to read</td>
<td>594</td>
</tr>
<tr>
<td>A.4 Assign all variables under all conditions</td>
<td>596</td>
</tr>
<tr>
<td>A.5 Keep modules small</td>
<td>597</td>
</tr>
<tr>
<td>A.6 Large modules should be structural</td>
<td>599</td>
</tr>
<tr>
<td>A.7 Use descriptive signal names</td>
<td>599</td>
</tr>
<tr>
<td>A.8 Use symbolic names for subfields of signals</td>
<td>599</td>
</tr>
<tr>
<td>A.9 Define constants</td>
<td>600</td>
</tr>
<tr>
<td>A.10 Comments should describe intention and give rationale, not state the obvious</td>
<td>601</td>
</tr>
<tr>
<td>A.11 Never forget you are defining hardware</td>
<td>602</td>
</tr>
<tr>
<td>A.12 Read and be a critic of Verilog code</td>
<td>602</td>
</tr>
</tbody>
</table>

References

Index of Verilog modules

Subject index