CONTENTS

Preface xxv
Acknowledgments xxxi

PART I BASIC DIGITAL CIRCUITS DEVELOPMENT

1 Gate-Level Combinational Circuit 1
   1.1 Overview of VHDL 1
   1.2 General description 2
       1.2.1 Basic lexical rules 3
       1.2.2 Library and package 3
       1.2.3 Entity declaration 3
       1.2.4 Data type and operators 3
       1.2.5 Architecture body 4
       1.2.6 Code of a 2-bit comparator 5
   1.3 Structural description 6
   1.4 Top-level signal mapping 8
   1.5 Testbench 9
   1.6 Bibliographic notes 11
   1.7 Suggested experiments 11
       1.7.1 Code for gate-level greater-than circuit 11
       1.7.2 Code for gate-level binary decoder 11
2 Overview of FPGA and EDA Software

2.1 FPGA
2.1.1 Overview of a general FPGA device
2.1.2 Overview of the Xilinx Artix-7 devices
2.2 Overview of the Digilent Nexys 4 DDR board
2.3 Development flow
2.4 Xilinx Vivado Design Suite
2.5 Bibliographic notes
2.6 Suggested experiments
2.6.1 Gate-level greater-than circuit
2.6.2 Gate-level binary decoder

3 RT-Level Combinational Circuit

3.1 RT-level components
3.1.1 Relational operators
3.1.2 Arithmetic operators
3.1.3 Other synthesis-related VHDL constructs
3.1.4 Summary
3.2 Routing circuit with concurrent assignment statements
3.2.1 Conditional signal assignment statement
3.2.2 Selected signal assignment statement
3.3 Modeling with a process
3.3.1 Process
3.3.2 Sequential signal assignment statement
3.4 Routing circuit with if and case statements
3.4.1 If statement
3.4.2 Case statement
3.4.3 Comparison to concurrent statements
3.4.4 Unintended memory
3.5 Constants and generics
3.5.1 Constants
3.5.2 Generics
3.6 Replicated structure
3.6.1 Loop statements
3.6.2 Example
3.7 Design examples
3.7.1 Hexadecimal digit to seven-segment LED decoder
3.7.2 Sign-magnitude adder
3.7.3 Barrel shifter
3.7.4 Simplified floating-point adder
3.8 Bibliographic notes
4 Regular Sequential Circuit 61

4.1 Introduction 61
   4.1.1 D FF and register 62
   4.1.2 Basic block system 62
   4.1.3 Code development 63
   4.1.4 Sequential circuit coding style 63

4.2 HDL code of the FF and register 64
   4.2.1 D FF 64
   4.2.2 Register 67

4.3 Simple design examples 68
   4.3.1 Shift register 68
   4.3.2 Binary counter and variant 69

4.4 Testbench for sequential circuits 72

4.5 Case study 75
   4.5.1 LED time-multiplexing circuit 76
   4.5.2 Stopwatch 83

4.6 Timing and clocking 87
   4.6.1 Timing of FF 87
   4.6.2 Maximum operating frequency 88
   4.6.3 Clock tree 89
   4.6.4 GALS system and CDC 90

4.7 Bibliographic notes 91

4.8 Suggested experiments 91
   4.8.1 Programmable square wave generator 91
   4.8.2 PWM and LED dimmer 91
   4.8.3 Rotating square circuit 91
   4.8.4 Heartbeat circuit 92
   4.8.5 Rotating LED banner circuit 92
   4.8.6 Enhanced stopwatch 92

5 FSM 93

5.1 Introduction 93
CONTENTS

5.1.1 Mealy and Moore outputs 94
5.1.2 FSM representation 94
5.2 FSM code development 97
5.3 Design examples 100
5.3.1 Rising-edge detector 100
5.3.2 Debouncing circuit 104
5.3.3 Testing circuit 108
5.4 Bibliographic notes 110
5.5 Suggested experiments 110
5.5.1 Dual-edge detector 110
5.5.2 Early detection debouncing circuit 110
5.5.3 Parking lot occupancy counter 110

6 FSMD 113
6.1 Introduction 113
6.1.1 Single RT operation 114
6.1.2 ASMD chart 114
6.1.3 Decision box with a register 116
6.2 Code development of an FSMD 119
6.2.1 Debouncing circuit based on RT methodology 119
6.2.2 Code with explicit data path components 119
6.2.3 Code with implicit data path components 122
6.2.4 Comparison 123
6.3 Design examples 125
6.3.1 Fibonacci number circuit 125
6.3.2 Division circuit 128
6.3.3 Binary-to-BCD conversion circuit 131
6.3.4 Period counter 134
6.3.5 Accurate low-frequency counter 137
6.4 Bibliographic notes 141
6.5 Suggested experiments 141
6.5.1 Early detection debouncing circuit 141
6.5.2 BCD-to-binary conversion circuit 141
6.5.3 Fibonacci circuit with BCD I/O: design approach 1 141
6.5.4 Fibonacci circuit with BCD I/O: design approach 2 142
6.5.5 Auto-scaled low-frequency counter 142
6.5.6 Reaction timer 143
6.5.7 Babbage difference engine emulation circuit 143

7 RAM and Buffer of FPGA 145
7.1 Embedded memory of FPGA device 145
7.1.1 Memory of an Artix device 146
7.1.2 Memory available in a Nexys 4 DDR board 146
7.2 General description for a RAM-like component 147
  7.2.1 Register file 147
  7.2.2 Dynamic array indexing operation 149
  7.2.3 Key aspects of a RAM module 151
  7.2.4 Genuine ROM 151
7.3 FIFO buffer 153
  7.3.1 FIFO read configuration 153
  7.3.2 Circular queue implementation 154
7.4 HDL templates for memory inference 158
  7.4.1 Methods to incorporate memory modules 158
  7.4.2 Synchronous dual-port RAM 159
  7.4.3 “Simple” synchronous dual-port RAM 160
  7.4.4 Synchronous single-port RAM 161
  7.4.5 Synchronous ROM 162
  7.4.6 BRAM-based FIFO buffer 163
  7.4.7 Design considerations 164
7.5 Overview of the memory controller 164
7.6 Bibliographic notes 166
7.7 Suggested experiments 166
  7.7.1 ROM-based sign-magnitude adder 166
  7.7.2 ROM-based temperature conversion 166
  7.7.3 FIFO with data width conversion 167
  7.7.4 Standard FIFO to FWFT FIFO conversion circuit 167
  7.7.5 FIFO buffer with extended status 167
  7.7.6 Stack 167

PART II EMBEDDED SOC I:
VANILLA FPRESS SYSTEM

8 Overview of Embedded SoC Systems 171
  8.1 Embedded SoC 171
    8.1.1 Overview of embedded systems 171
    8.1.2 FPGA-based SoC 172
    8.1.3 IP cores 172
  8.2 Development flow of the embedded SoC 173
    8.2.1 Hardware–software partition 173
    8.2.2 Hardware development flow 173
    8.2.3 Software development flow 175
    8.2.4 Physical implementation and test 175
    8.2.5 Custom IP core development 175
8.3 FPro SoC platform 176
  8.3.1 Motivations 176
  8.3.2 Platform hardware organization 177
  8.3.3 Platform software organization 179
  8.3.4 Modified development flow 180
8.4 Adaptation on the Digilent Nexys 4 DDR board 180
8.5 Portability 182
  8.5.1 Processor module and bridge 182
  8.5.2 MMIO subsystem 183
  8.5.3 Video subsystem 183
8.6 Organization 184
8.7 Bibliographic notes 184

9 Bare Metal System Software Development 187
9.1 Bare metal system development overview 187
  9.1.1 Desktop-like system versus bare metal system 187
  9.1.2 Basic embedded program architecture 188
9.2 Memory-mapped I/O 189
  9.2.1 Overview 189
  9.2.2 Memory alignment 190
  9.2.3 I/O register map 190
  9.2.4 I/O address space of the FPro system 190
9.3 Direct I/O register access 191
  9.3.1 Review of C pointer 191
  9.3.2 C pointer for I/O register 192
9.4 Robust I/O register access 193
  9.4.1 chu_io_map.h and chu_io_map.vhd 193
  9.4.2 inttypes.h 194
  9.4.3 chu_io_rwlock.h 195
9.5 Techniques for low-level I/O operations 197
  9.5.1 Bit manipulation 197
  9.5.2 Packing and unpacking 198
9.6 Device drivers 199
  9.6.1 Overview 199
  9.6.2 GPO and GPI drivers 199
  9.6.3 Timer driver 201
  9.6.4 UART driver 203
9.7 FPro utility routines and directory structure 204
  9.7.1 Minimal hardware requirements 204
  9.7.2 Utility routines 204
  9.7.3 Directory structure 207
9.8 Test program 207
9.8.1 IP core verification routine 207
9.8.2 Programming with limited memory 208
9.8.3 Test function integration 208
9.8.4 Test program for the vanilla FPro system 209
9.8.5 Implementation 210
9.9 Bibliographic notes 211
9.10 Suggested experiments 211
9.10.1 Chasing LEDs 211
9.10.2 Collision LEDs 211
9.10.3 Pulse width modulation 212
9.10.4 System time display 212

10 FPro Bus Protocol and MMIO Slot Specification 213

10.1 FPro bus 213
  10.1.1 Overview of the bus 213
  10.1.2 SoC interconnect 214
  10.1.3 FPro bus protocol specification 215

10.2 Interface with the bus 216
  10.2.1 Introduction 216
  10.2.2 Write interface and decoding 217
  10.2.3 Read interface and multiplexing 219
  10.2.4 FIFO buffer as an I/O register 220
  10.2.5 Timing consideration 221

10.3 MMIO I/O core 222
  10.3.1 MMIO slot interface specification 222
  10.3.2 Basic MMIO I/O core construction 224
  10.3.3 GPO and GPI cores 225

10.4 Timer core development 227
  10.4.1 Custom logic 227
  10.4.2 Register map 227
  10.4.3 Wrapping circuit for the slot interface 228

10.5 MMIO controller 229
  10.5.1 chu_io_map.vhd file 229
  10.5.2 HDL code 231
  10.5.3 Vanilla MMIO subsystem 232

10.6 MCS I/O bus and bridge 235
  10.6.1 Overview of Xilinx MicroBlaze MCS 235
  10.6.2 MicroBlaze MCS I/O bus 235
  10.6.3 MCS-to-FPro bridge 237

10.7 Vanilla FPro system construction 238
10.8 Bibliographic notes 240
10.9 Suggested experiments 241
11 UART Core

11.1 Introduction
11.1.1 Overview of serial communication
11.1.2 Overview of UART
11.1.3 Oversampling procedure

11.2 UART Construction
11.2.1 Conceptual design
11.2.2 Baud rate generator
11.2.3 UART receiver
11.2.4 UART transmitter
11.2.5 Top-level HDL codes

11.3 UART core development
11.3.1 Register map
11.3.2 Wrapping circuit for the slot interface

11.4 UART driver
11.4.1 Class definition
11.4.2 Basic methods
11.4.3 ASCII code
11.4.4 Display methods
11.4.5 Test

11.5 Additional project ideas
11.5.1 Original serial port
11.5.2 Emulated serial port
11.5.3 Direct connection
11.5.4 USB-to-UART adaptor
11.5.5 Wireless adaptor

11.6 Bibliographic notes

11.7 Suggested experiments
11.7.1 UART-controlled chasing LEDs
11.7.2 Alternative read configuration
11.7.3 UART controller with a parity bit
11.7.4 UART core with an error status
11.7.5 Configurable UART core
11.7.6 UART core with automatic baud rate detection
11.7.7 UART core with enhanced automatic baud rate detection
11.7.8 UART core with an automatic baud rate and a parity detection circuit

PART III EMBEDDED SOC II: BASIC I/O CORES

12 Xilinx XADC Core

12.1 Overview of XADC
  12.1.1 Block diagram
  12.1.2 Configuration

12.2 XADC core development
  12.2.1 XADC instantiation
  12.2.2 Basic wrapping circuit design
  12.2.3 Register map
  12.2.4 HDL code

12.3 XADC core device driver
  12.3.1 Class definition
  12.3.2 Class implementation
  12.3.3 Testing for the XADC core

12.4 Sampler FPro system
  12.4.1 Testing procedure of an FPro core
  12.4.2 System configuration
  12.4.3 Hardware derivation
  12.4.4 Software verification program

12.5 Additional project ideas
12.6 Bibliographic notes
12.7 Suggested experiments
  12.7.1 Real-time voltage display
  12.7.2 Potentiometer-controlled chasing LEDs
  12.7.3 Potentiometer-controlled LED dimmer
  12.7.4 Enhanced wrapping circuit I
  12.7.5 Enhanced wrapping circuit II

13 Pulse Width Modulation Core

13.1 Introduction
  13.1.1 PWM as analog output
  13.1.2 Main characteristics

13.2 PWM design
  13.2.1 Basic design
  13.2.2 Enhanced design

13.3 PWM core development
  13.3.1 Register map
  13.3.2 Wrapped PWM circuit
13.4 PWM driver
   13.4.1 Class definition 304
   13.4.2 Class implementation 304
13.5 Testing 305
13.6 Project ideas 306
13.7 Suggested experiments 307
   13.7.1 Police dash light 307
   13.7.2 Rainbow night light 307
   13.7.3 Enhanced PWM core: part I 307
   13.7.4 Enhanced PWM core: part II 308
   13.7.5 Enhanced GPIO core 308
   13.7.6 Servo motor driver 308

14 Debouncing Core and LED-Mux Core 309
   14.1 Debouncing core 309
      14.1.1 Multi-bit debouncing circuit 309
      14.1.2 Register map and the slot wrapping circuit 312
      14.1.3 Driver 313
      14.1.4 Test 314
   14.2 LED-Mux core 315
      14.2.1 Eight-digit seven-segment LED display multiplexing circuit 315
      14.2.2 Register map and the slot wrapping circuit 316
      14.2.3 Driver 318
      14.2.4 Test 320
   14.3 Project ideas 321
   14.4 Suggested experiments 322
      14.4.1 Area comparison of two debouncing circuits 322
      14.4.2 Enhanced debouncing core: part I 323
      14.4.3 Enhanced debouncing core: part II 323
      14.4.4 Rotating square pattern revisited 323
      14.4.5 Heartbeat pattern revisited 323
      14.4.6 Stopwatch 323
      14.4.7 Enhanced LED-mux core 323

15 SPI Core 325
   15.1 Overview 325
      15.1.1 Conceptual architecture 326
      15.1.2 Multiple device configuration 326
      15.1.3 Basic timing 328
      15.1.4 Operation modes 329
      15.1.5 Undefined aspects 330
15.2 SPI controller 331
  15.2.1 Basic design 331
  15.2.2 FSMD construction 332
  15.2.3 HDL implementation 332
15.3 SPI core development 336
  15.3.1 Register map 336
  15.3.2 Wrapping circuit for the slot interface 336
15.4 SPI Driver 338
  15.4.1 Class definition 338
  15.4.2 Class implementation 339
15.5 Test 340
  15.5.1 ADXL362 accelerometer 341
  15.5.2 Test program 342
15.6 Project ideas 343
  15.6.1 SD card 343
  15.6.2 TFT LCD module 343
15.7 Bibliographic notes 344
15.8 Suggested experiments 344
  15.8.1 Inclination sensing 344
  15.8.2 “Tapping” detection 345
  15.8.3 ADXL362 C++ class 345
  15.8.4 Enhanced SPI controller: part I 345
  15.8.5 Enhanced SPI controller: part II 345
  15.8.6 Automatic-read ADXL362 wrapper: part I 346
  15.8.7 Automatic-read ADXL362 wrapper: part II 346
  15.8.8 Flash memory access 346
  15.8.9 SPI slave controller: part I 346
  15.8.10 SPI slave controller: part II 347

16 I^2C Core 349
16.1 Overview 349
  16.1.1 Electrical characteristics 350
  16.1.2 Basic bus protocol 350
  16.1.3 Basic timing 351
  16.1.4 Additional features 352
16.2 I^2C controller 353
  16.2.1 Basic design 353
  16.2.2 Conceptual FSMD construction 353
  16.2.3 Output control logic 356
  16.2.4 I^2C bus clock generation 356
  16.2.5 HDL implementation 357
16.3 I^2C core development 362
16.3.1 Register map 362
16.3.2 Wrapping circuit for the slot interface 362

16.4 I²C driver 364
  16.4.1 Class definition 364
  16.4.2 Class implementation 364

16.5 Test 367
  16.5.1 ADT7420 temperature sensor 367
  16.5.2 Test program 368

16.6 Project idea 369

16.7 Bibliographic notes 369

16.8 Suggested experiments 370
  16.8.1 Thermometer 370
  16.8.2 ADT7420 C++ class 370
  16.8.3 Enhanced I²C core 370
  16.8.4 Automatic-read ADT7420 wrapper 370
  16.8.5 I²C slave controller: part I 371
  16.8.6 I²C slave controller: part II 371

17 PS2 Core 373

17.1 Introduction 373
  17.1.1 PS2-device-to-host communication protocol and timing 374
  17.1.2 Host-to-PS2-device communication protocol and timing 374

17.2 PS2 controller 375
  17.2.1 Conceptual design 375
  17.2.2 PS2 receiving subsystem 375
  17.2.3 PS2 transmitting subsystem 379
  17.2.4 Complete PS2 system 384

17.3 PS2 core development 385
  17.3.1 Register map 385
  17.3.2 Wrapping circuit for the slot interface 385

17.4 PS2 driver 387
  17.4.1 Class definition 387
  17.4.2 Lower layer methods 387
  17.4.3 PS2 initialization routine 388
  17.4.4 Keyboard routine 390
  17.4.5 Mouse routine 393

17.5 Test 395

17.6 Bibliographic notes 396

17.7 Suggested experiments 396
  17.7.1 PS2 receiving subsystem with watchdog timer 396
  17.7.2 Keyboard-controlled LED flashing circuit 396
  17.7.3 Enhanced keyboard driver routine I 397
17.7.4 Enhanced keyboard driver routine II 397
17.7.5 Remote-mode mouse driver 397
17.7.6 Scroll-wheel mouse driver 397

18 Sound I: DDFS Core 399
18.1 Introduction 399
18.2 Design and implementation 400
  18.2.1 Direct synthesis of a digital waveform 400
  18.2.2 Direct synthesis of an unmodulated analog waveform 401
  18.2.3 Direct synthesis of a modulated analog waveform 402
18.3 Fixed-point arithmetic 403
18.4 DDFS Construction 404
18.5 DAC (digital-to-analog converter) 407
  18.5.1 Conceptual design 407
  18.5.2 HDL implementation 408
18.6 DDFS core development 409
  18.6.1 Register map 409
  18.6.2 Wrapping circuit for the slot interface 410
18.7 DDFS driver 412
  18.7.1 Class definition 412
  18.7.2 Class implementation 412
18.8 Testing 414
18.9 Bibliographic notes 415
18.10 Suggested experiments 415
  18.10.1 Quadrature phase carrier generation 416
  18.10.2 Reduced-size phase-to-amplitude lookup table 416
  18.10.3 Additive harmonic synthesis 416
  18.10.4 Simple function generator 416
  18.10.5 Arbitrary waveform generator 416
  18.10.6 Sample-based synthesis 417

19 Sound II: ADSR Core 419
19.1 Introduction 419
19.2 ADSR envelope generator 420
  19.2.1 Conceptual FSMD design 421
  19.2.2 ASMD chart 421
  19.2.3 HDL implementation 423
19.3 ADSR core development 425
  19.3.1 Register map 425
  19.3.2 Wrapped ADSR circuit 426
19.4 ADSR driver 428
## CONTENTS

19.4.1 Class definition 428  
19.4.2 Configuration methods 429  
19.4.3 calc_note_freq() method 431  
19.4.4 play_note() method 432  

19.5 Testing 433  
19.6 Project idea 434  
19.7 Bibliographic notes 435  
19.8 Suggested experiments 435  
  19.8.1 RTTTL music player 435  
  19.8.2 ADSR envelope testing 435  
  19.8.3 Pushbutton piano 435  
  19.8.4 Keyboard piano 435  
  19.8.5 Keyboard recorder 435  
  19.8.6 Real-time mode ADSR generator 436  
  19.8.7 Real-time mode button piano 436  
  19.8.8 Merged DDS and ADSR core 436  
  19.8.9 ADSR core with an automatic play FIFO buffer 436  
  19.8.10 ADSR core for frequency modulation 436  

### PART IV  EMBEDDED SOC III: VIDEO CORES

20 Introduction to the Video System 439  
  20.1 Introduction to a video display 439  
  20.1.1 Conceptual video display 439  
  20.1.2 VGA interface 440  
  20.2 Stream interface 441  
    20.2.1 Random-access interface versus stream interface 441  
    20.2.2 Flow control of the stream interface 441  
  20.3 VGA synchronization 443  
    20.3.1 Basic operation of a CRT monitor 443  
    20.3.2 Horizontal synchronization 444  
    20.3.3 Vertical synchronization 446  
    20.3.4 Pixel clock rate 447  
    20.3.5 VGA synchronization circuit 448  
  20.4 Bar test-pattern generator 452  
  20.5 Color-to-grayscale conversion circuit 454  
  20.6 Demo video system 455  
  20.7 Advanced video standards 457  
  20.8 Bibliographic notes 458  
  20.9 Suggested experiments 458  
    20.9.1 Horizontal bar test-pattern generator 458  
    20.9.2 Color channel selection circuit 458
20.9.3 Enhanced color-to-grayscale conversion circuit 458
20.9.4 Square test-pattern generator: part I 458
20.9.5 Square test-pattern generator: part II 459
20.9.6 Square test-pattern generator: part III 459
20.9.7 Square test-pattern generator: part IV 459

21 FPro Video Subsystem 461
21.1 Organization of the video subsystem 461
21.1.1 Overview 461
21.1.2 Video controller 463
21.1.3 HDL of the video controller 464
21.2 FPro video IP core 465
21.2.1 Basic functionality 465
21.2.2 Blending operation 466
21.2.3 Core architecture 468
21.2.4 Alternative core partition 470
21.3 Example video cores 470
21.3.1 Bar test-pattern generator core 470
21.3.2 Color-to-grayscale conversion core 473
21.3.3 “Dummy” core 474
21.4 FPro video synchronization core 475
21.4.1 Line buffer 476
21.4.2 Enhanced video synchronization circuit 479
21.4.3 HDL code 482
21.5 Daisy video subsystem 483
21.5.1 Subsystem overview 483
21.5.2 Interface to the video synchronization core 484
21.5.3 HDL code 485
21.5.4 Timing and performance considerations 489
21.6 Vanilla daisy FPro system 490
21.6.1 Clock management core 490
21.6.2 Updated chu_io.map VHDL package 491
21.6.3 HDL code 491
21.7 Video driver and testing program 494
21.7.1 Updated chu_io.map.h and chu_io_rw.h files 494
21.7.2 GPV core driver 495
21.7.3 Testing program 495
21.8 Bibliographic notes 497
21.9 Suggested experiments 497
21.9.1 Color channel selection core 497
21.9.2 Enhanced color-to-grayscale conversion core 497
21.9.3 Square test-pattern generator core 498


<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>21.9.4</td>
<td>Alpha blending circuit</td>
<td>498</td>
</tr>
<tr>
<td>21.9.5</td>
<td>“Highlight” core</td>
<td>498</td>
</tr>
<tr>
<td>21.9.6</td>
<td>SVGA synchronization core</td>
<td>498</td>
</tr>
<tr>
<td>21.9.7</td>
<td>Configurable video synchronization core</td>
<td>499</td>
</tr>
<tr>
<td>21.9.8</td>
<td>Pipelined video subsystem</td>
<td>499</td>
</tr>
</tbody>
</table>

**22 Sprite Core**

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>22.1</td>
<td>Introduction</td>
<td>501</td>
</tr>
<tr>
<td>22.2</td>
<td>Basic design</td>
<td>502</td>
</tr>
<tr>
<td>22.2.1</td>
<td>Sprite RAM</td>
<td>502</td>
</tr>
<tr>
<td>22.2.2</td>
<td>In-region comparison circuit</td>
<td>503</td>
</tr>
<tr>
<td>22.3</td>
<td>Mouse pointer core</td>
<td>504</td>
</tr>
<tr>
<td>22.3.1</td>
<td>Pointer sprite RAM</td>
<td>504</td>
</tr>
<tr>
<td>22.3.2</td>
<td>Pixel generation circuit</td>
<td>505</td>
</tr>
<tr>
<td>22.3.3</td>
<td>Top-level design</td>
<td>507</td>
</tr>
<tr>
<td>22.4</td>
<td>“Ghost” character core</td>
<td>509</td>
</tr>
<tr>
<td>22.4.1</td>
<td>Multiple images and animation</td>
<td>509</td>
</tr>
<tr>
<td>22.4.2</td>
<td>Overview of the palette scheme</td>
<td>510</td>
</tr>
<tr>
<td>22.4.3</td>
<td>Ghost sprite RAM and the palette circuit</td>
<td>510</td>
</tr>
<tr>
<td>22.4.4</td>
<td>Animation timing circuit</td>
<td>512</td>
</tr>
<tr>
<td>22.4.5</td>
<td>Pixel generation circuit</td>
<td>512</td>
</tr>
<tr>
<td>22.4.6</td>
<td>Top-level design</td>
<td>515</td>
</tr>
<tr>
<td>22.5</td>
<td>Sprite core driver and testing program</td>
<td>517</td>
</tr>
<tr>
<td>22.5.1</td>
<td>Sprite core driver</td>
<td>517</td>
</tr>
<tr>
<td>22.5.2</td>
<td>Testing program</td>
<td>518</td>
</tr>
<tr>
<td>22.6</td>
<td>Bibliographic notes</td>
<td>520</td>
</tr>
<tr>
<td>22.7</td>
<td>Suggested experiments</td>
<td>520</td>
</tr>
<tr>
<td>22.7.1</td>
<td>Mouse pointer control with a PS2 core</td>
<td>520</td>
</tr>
<tr>
<td>22.7.2</td>
<td>Emulated ghost core</td>
<td>520</td>
</tr>
<tr>
<td>22.7.3</td>
<td>Palette circuit for the mouse pointer sprite</td>
<td>520</td>
</tr>
<tr>
<td>22.7.4</td>
<td>Sprite scaling circuit</td>
<td>520</td>
</tr>
<tr>
<td>22.7.5</td>
<td>Portrait mode display</td>
<td>520</td>
</tr>
<tr>
<td>22.7.6</td>
<td>Multiple object generation</td>
<td>521</td>
</tr>
<tr>
<td>22.7.7</td>
<td>Animation speed control</td>
<td>521</td>
</tr>
<tr>
<td>22.7.8</td>
<td>Imitated blinking LED: part I</td>
<td>521</td>
</tr>
<tr>
<td>22.7.9</td>
<td>Imitated blinking LED: part II</td>
<td>522</td>
</tr>
<tr>
<td>22.7.10</td>
<td>Imitated blinking LED: part III</td>
<td>522</td>
</tr>
</tbody>
</table>

**23 On-Screen-Display Core**

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>23.1</td>
<td>Introduction to tile graphics</td>
<td>523</td>
</tr>
<tr>
<td>23.2</td>
<td>Basic OSD design</td>
<td>525</td>
</tr>
</tbody>
</table>
23.2.1 Text-mode display 525
23.2.2 Font ROM 526
23.2.3 Tile RAM 526
23.2.4 Basic organization 527
23.3 OSD core 528
23.3.1 Font ROM 528
23.3.2 Pixel generation circuit 529
23.3.3 Top-level design 532
23.4 OSD core driver and testing program 534
23.4.1 OSD core driver 534
23.4.2 Testing program 535
23.5 Bibliographic notes 536
23.6 Suggested experiments 536
23.6.1 Rotating banner 536
23.6.2 Text console 536
23.6.3 Underline for the cursor 537
23.6.4 Portrait-mode display 537
23.6.5 Font scaling circuit: part I 537
23.6.6 Font scaling circuit: part II 537
23.6.7 Extended font 537
23.6.8 Tile-based ghost core 538

24 VGA Frame Buffer Core 539

24.1 Overview 539
24.2 Frame buffer core 540
24.2.1 FPGA memory consideration 540
24.2.2 Video memory module 540
24.2.3 Address translation 542
24.2.4 Pixel generation circuit 542
24.3 Register map 544
24.3.1 Top-level HDL code 545
24.4 Driver and the testing program 546
24.4.1 Frame buffer core driver 546
24.4.2 Geometrical modeling 547
24.4.3 Testing program 549
24.5 Project ideas 549
24.6 Bibliographic notes 550
24.7 Suggested experiments 551
24.7.1 Virtual prototyping board panel 551
24.7.2 Virtual analog wall clock 551
24.7.3 Geometrical model functions 551
24.7.4 Simulated “Etch A Sketch” toy 551
24.7.5 Frame buffer core with 3-bit color depth 551
24.7.6 Frame buffer core with 1-bit color depth 552
24.7.7 QVGA frame buffer core 552
24.7.8 Line drawing hardware accelerator 552
24.7.9 Bidirectional frame buffer access: part I 552
24.7.10 Bidirectional frame buffer access: part II 552

PART V EPILOGUE

25 What’s Next 557

References 561

Appendix A: Tutorials 565
A.1 Overview of the Xilinx Vivado IDE 565
A.2 Short tutorial on Vivado hardware development 569
  A.2.1 Create a design project 570
  A.2.2 Add or create Xilinx IP core instances 571
  A.2.3 Add or create HDL design files 571
  A.2.4 Add a constraint file 572
  A.2.5 Perform synthesis, implementation, and bitstream generation 573
  A.2.6 Program an FPGA device 573
A.3 Short tutorial on Vivado simulation 574
  A.3.1 Add or create an HDL testbench 576
  A.3.2 Perform initial simulation 577
  A.3.3 Customize waveform display 577
A.4 Tutorial on IP instantiation 578
  A.4.1 Dual-clock FIFO core via HDL templates 579
  A.4.2 IP catalog utility 579
  A.4.3 Generate a MicroBlaze MCS component 580
  A.4.4 XADC IP core 581
  A.4.5 Clock management IP core 583
A.5 Short tutorial on FPro system development 584
  A.5.1 Derive FPro system hardware 585
  A.5.2 Export hardware configuration 585
  A.5.3 Derive software 586
  A.5.4 Embed elf file and regenerate bitstream 590
  A.5.5 Set up a terminal emulator program 590
  A.5.6 Program an FPGA device 591
A.6 Bibliographic notes 591

Topic Index 593