

---

# **FPGA PROTOTYPING BY VHDL EXAMPLES**

**Xilinx Spartan™-3 Version**

---

**Pong P. Chu**

Cleveland State University

 **WILEY-  
INTERSCIENCE**

A JOHN WILEY & SONS, INC., PUBLICATION

# CONTENTS

---

Preface	xix
Acknowledgments	xxv

## PART I BASIC DIGITAL CIRCUITS

<b>1 Gate-level combinational circuit</b>	<b>1</b>
1.1 Introduction	1
1.2 General description	2
1.2.1 Basic lexical rules	2
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 Testbench	8
1.5 Bibliographic notes	9
1.6 Suggested experiments	10
1.6.1 Code for gate-level greater-than circuit	10
1.6.2 Code for gate-level binary decoder	10
<b>2 Overview of FPGA and EDA software</b>	<b>11</b>

2.1	Introduction	11
2.2	FPGA	11
2.2.1	Overview of a general FPGA device	11
2.2.2	Overview of the Xilinx Spartan-3 devices	13
2.3	Overview of the Digilent S3 board	13
2.4	Development flow	15
2.5	Overview of the Xilinx ISE project navigator	17
2.6	Short tutorial on ISE project navigator	19
2.6.1	Create the design project and HDL codes	21
2.6.2	Create a testbench and perform the RTL simulation	22
2.6.3	Add a constraint file and synthesize and implement the code	22
2.6.4	Generate and download the configuration file to an FPGA device	24
2.7	Short tutorial on the ModelSim HDL simulator	27
2.8	Bibliographic notes	32
2.9	Suggested experiments	33
2.9.1	Gate-level greater-than circuit	33
2.9.2	Gate-level binary decoder	33
<b>3</b>	<b>RT-level combinational circuit</b>	<b>35</b>
3.1	Introduction	35
3.2	RT-level components	35
3.2.1	Relational operators	37
3.2.2	Arithmetic operators	37
3.2.3	Other synthesis-related VHDL constructs	38
3.2.4	Summary	40
3.3	Routing circuit with concurrent assignment statements	41
3.3.1	Conditional signal assignment statement	41
3.3.2	Selected signal assignment statement	44
3.4	Modeling with a process	46
3.4.1	Process	46
3.4.2	Sequential signal assignment statement	46
3.5	Routing circuit with if and case statements	47
3.5.1	If statement	47
3.5.2	Case statement	49
3.5.3	Comparison to concurrent statements	50
3.5.4	Unintended memory	52
3.6	Constants and generics	53
3.6.1	Constants	53
3.6.2	Generics	54
3.7	Design examples	56
3.7.1	Hexadecimal digit to seven-segment LED decoder	56
3.7.2	Sign-magnitude adder	59

3.7.3	Barrel shifter	62
3.7.4	Simplified floating-point adder	63
3.8	Bibliographic notes	69
3.9	Suggested experiments	69
3.9.1	Multi-function barrel shifter	69
3.9.2	Dual-priority encoder	69
3.9.3	BCD incrementor	69
3.9.4	Floating-point greater-than circuit	70
3.9.5	Floating-point and signed integer conversion circuit	70
3.9.6	Enhanced floating-point adder	70
<b>4</b>	<b>Regular Sequential Circuit</b>	<b>71</b>
4.1	Introduction	71
4.1.1	D FF and register	71
4.1.2	Synchronous system	72
4.1.3	Code development	73
4.2	HDL code of the FF and register	74
4.2.1	D FF	74
4.2.2	Register	77
4.2.3	Register file	78
4.2.4	Storage components in a Spartan-3 device <i>Xilinx specific</i>	79
4.3	Simple design examples	79
4.3.1	Shift register	79
4.3.2	Binary counter and variant	81
4.4	Testbench for sequential circuits	84
4.5	Case study	88
4.5.1	LED time-multiplexing circuit	88
4.5.2	Stopwatch	96
4.5.3	FIFO buffer	100
4.6	Bibliographic notes	104
4.7	Suggested experiments	105
4.7.1	Programmable square wave generator	105
4.7.2	PWM and LED dimmer	105
4.7.3	Rotating square circuit	105
4.7.4	Heartbeat circuit	106
4.7.5	Rotating LED banner circuit	106
4.7.6	Enhanced stopwatch	106
4.7.7	Stack	106
<b>5</b>	<b>FSM</b>	<b>107</b>
5.1	Introduction	107

5.1.1	Mealy and Moore outputs	107
5.1.2	FSM representation	108
5.2	FSM code development	111
5.3	Design examples	114
5.3.1	Rising-edge detector	114
5.3.2	Debouncing circuit	118
5.3.3	Testing circuit	122
5.4	Bibliographic notes	124
5.5	Suggested experiments	124
5.5.1	Dual-edge detector	124
5.5.2	Alternative debouncing circuit	124
5.5.3	Parking lot occupancy counter	125
<b>6</b>	<b>FSMD</b>	<b>127</b>
6.1	Introduction	127
6.1.1	Single RT operation	127
6.1.2	ASMD chart	128
6.1.3	Decision box with a register	129
6.2	Code development of an FSMD	131
6.2.1	Debouncing circuit based on RT methodology	132
6.2.2	Code with explicit data path components	134
6.2.3	Code with implicit data path components	136
6.2.4	Comparison	137
6.2.5	Testing circuit	138
6.3	Design examples	140
6.3.1	Fibonacci number circuit	140
6.3.2	Division circuit	143
6.3.3	Binary-to-BCD conversion circuit	147
6.3.4	Period counter	150
6.3.5	Accurate low-frequency counter	153
6.4	Bibliographic notes	156
6.5	Suggested experiments	157
6.5.1	Alternative debouncing circuit	157
6.5.2	BCD-to-binary conversion circuit	157
6.5.3	Fibonacci circuit with BCD I/O: design approach 1	157
6.5.4	Fibonacci circuit with BCD I/O: design approach 2	157
6.5.5	Auto-scaled low-frequency counter	158
6.5.6	Reaction timer	158
6.5.7	Babbage difference engine emulation circuit	159

**PART II I/O MODULES**

<b>7</b>	<b>UART</b>	<b>163</b>
7.1	Introduction	163
7.2	UART receiving subsystem	164
7.2.1	Oversampling procedure	164
7.2.2	Baud rate generator	165
7.2.3	UART receiver	165
7.2.4	Interface circuit	168
7.3	UART transmitting subsystem	171
7.4	Overall UART system	174
7.4.1	Complete UART core	174
7.4.2	UART verification configuration	176
7.5	Customizing a UART	178
7.6	Bibliographic notes	180
7.7	Suggested experiments	180
7.7.1	Full-featured UART	180
7.7.2	UART with an automatic baud rate detection circuit	181
7.7.3	UART with an automatic baud rate and parity detection circuit	181
7.7.4	UART-controlled stopwatch	181
7.7.5	UART-controlled rotating LED banner	182
<b>8</b>	<b>PS2 Keyboard</b>	<b>183</b>
8.1	Introduction	183
8.2	PS2 receiving subsystem	184
8.2.1	Physical interface of a PS2 port	184
8.2.2	Device-to-host communication protocol	184
8.2.3	Design and code	184
8.3	PS2 keyboard scan code	188
8.3.1	Overview of the scan code	188
8.3.2	Scan code monitor circuit	189
8.4	PS2 keyboard interface circuit	191
8.4.1	Basic design and HDL code	192
8.4.2	Verification circuit	194
8.5	Bibliographic notes	196
8.6	Suggested experiments	196
8.6.1	Alternative keyboard interface I	196
8.6.2	Alternative keyboard interface II	196
8.6.3	PS2 receiving subsystem with watchdog timer	197
8.6.4	Keyboard-controlled stopwatch	197
8.6.5	Keyboard-controlled rotating LED banner	197
<b>9</b>	<b>PS2 Mouse</b>	<b>199</b>

9.1	Introduction	199
9.2	PS2 mouse protocol	200
9.2.1	Basic operation	200
9.2.2	Basic initialization procedure	200
9.3	PS2 transmitting subsystem	201
9.3.1	Host-to-PS2-device communication protocol	201
9.3.2	Design and code	202
9.4	Bidirectional PS2 interface	206
9.4.1	Basic design and code	206
9.4.2	Verification circuit	208
9.5	PS2 mouse interface	210
9.5.1	Basic design	210
9.5.2	Testing circuit	212
9.6	Bibliographic notes	214
9.7	Suggested experiments	214
9.7.1	Keyboard control circuit	214
9.7.2	Enhanced mouse interface	214
9.7.3	Mouse-controlled seven-segment LED display	214
<b>10</b>	<b>External SRAM</b>	<b>215</b>
10.1	Introduction	215
10.2	Specification of the IS61LV25616AL SRAM	216
10.2.1	Block diagram and I/O signals	216
10.2.2	Timing parameters	216
10.3	Basic memory controller	220
10.3.1	Block diagram	220
10.3.2	Timing requirement	221
10.3.3	Register file versus SRAM	222
10.4	A safe design	222
10.4.1	ASMD chart	222
10.4.2	Timing analysis	223
10.4.3	HDL implementation	224
10.4.4	Basic testing circuit	226
10.4.5	Comprehensive SRAM testing circuit	228
10.5	More aggressive design	233
10.5.1	Timing issues	233
10.5.2	Alternative design I	234
10.5.3	Alternative design II	236
10.5.4	Alternative design III	237
10.5.5	Advanced FPGA features <i>Xilinx specific</i>	237
10.6	Bibliographic notes	240
10.7	Suggested experiments	240

10.7.1	Memory with a 512K-by-16 configuration	240
10.7.2	Memory with a 1M-by-8 configuration	240
10.7.3	Memory with an 8M-by-1 configuration	240
10.7.4	Expanded memory testing circuit	241
10.7.5	Memory controller and testing circuit for alternative design I	241
10.7.6	Memory controller and testing circuit for alternative design II	241
10.7.7	Memory controller and testing circuit for alternative design III	241
10.7.8	Memory controller with DCM	241
10.7.9	High-performance memory controller	241
<b>11</b>	<b>Xilinx Spartan-3 Specific Memory</b>	<b>243</b>
11.1	Introduction	243
11.2	Embedded memory of Spartan-3 device	243
11.2.1	Overview	243
11.2.2	Comparison	244
11.3	Method to incorporate memory modules	244
11.3.1	Memory module via HDL component instantiation	245
11.3.2	Memory module via Core Generator	245
11.3.3	Memory module via HDL inference	246
11.4	HDL templates for memory inference	246
11.4.1	Single-port RAM	246
11.4.2	Dual-port RAM	249
11.4.3	ROM	251
11.5	Bibliographic notes	254
11.6	Suggested experiments	254
11.6.1	Block-RAM-based FIFO	254
11.6.2	Block-RAM-based stack	254
11.6.3	ROM-based sign-magnitude adder	255
11.6.4	ROM based $\sin(x)$ function	255
11.6.5	ROM-based $\sin(x)$ and $\cos(x)$ functions	255
<b>12</b>	<b>VGA controller I: graphic</b>	<b>257</b>
12.1	Introduction	257
12.1.1	Basic operation of a CRT	257
12.1.2	VGA port of the S3 board	259
12.1.3	Video controller	259
12.2	VGA synchronization	260
12.2.1	Horizontal synchronization	260
12.2.2	Vertical synchronization	262
12.2.3	Timing calculation of VGA synchronization signals	263
12.2.4	HDL implementation	263



12.2.5	Testing circuit	266
12.3	Overview of the pixel generation circuit	267
12.4	Graphic generation with an object-mapped scheme	268
12.4.1	Rectangular objects	269
12.4.2	Non-rectangular object	273
12.4.3	Animated object	275
12.5	Graphic generation with a bit-mapped scheme	282
12.5.1	Dual-port RAM implementation	282
12.5.2	Single-port RAM implementation	287
12.6	Bibliographic notes	287
12.7	Suggested experiments	287
12.7.1	VGA test pattern generator	287
12.7.2	SVGA mode synchronization circuit	288
12.7.3	Visible screen adjustment circuit	288
12.7.4	Ball-in-a-box circuit	288
12.7.5	Two-balls-in-a-box circuit	289
12.7.6	Two-player pong game	289
12.7.7	Breakout game	289
12.7.8	Full-screen dot trace	289
12.7.9	Mouse pointer circuit	290
12.7.10	Small-screen mouse scribble circuit	290
12.7.11	Full-screen mouse scribble circuit	290
<b>13</b>	<b>VGA controller II: text</b>	<b>291</b>
13.1	Introduction	291
13.2	Text generation	291
13.2.1	Character as a tile	291
13.2.2	Font ROM	292
13.2.3	Basic text generation circuit	294
13.2.4	Font display circuit	295
13.2.5	Font scaling	297
13.3	Full-screen text display	298
13.4	The complete pong game	302
13.4.1	Text subsystem	302
13.4.2	Modified graphic subsystem	309
13.4.3	Auxiliary counters	310
13.4.4	Top-level system	312
13.5	Bibliographic notes	317
13.6	Suggested experiments	317
13.6.1	Rotating banner	317
13.6.2	Underline for the cursor	317
13.6.3	Dual-mode text display	317

13.6.4	Keyboard text entry	317
13.6.5	UART terminal	317
13.6.6	Square wave display	318
13.6.7	Simple four-trace logic analyzer	318
13.6.8	Complete two-player pong game	319
13.6.9	Complete breakout game	319
<b>PART III PICOBLAZE MICROCONTROLLER</b> <i>XILINX SPECIFIC</i>		
<b>14</b>	<b>PicoBlaze Overview</b>	<b>323</b>
14.1	Introduction	323
14.2	Customized hardware and customized software	324
14.2.1	From special-purpose FSM to general-purpose microcontroller	324
14.2.2	Application of microcontroller	326
14.3	Overview of PicoBlaze	326
14.3.1	Basic organization	326
14.3.2	Top-level HDL modules	328
14.4	Development flow	329
14.5	Instruction set	329
14.5.1	Programming model	331
14.5.2	Instruction format	332
14.5.3	Logical instructions	332
14.5.4	Arithmetic instructions	333
14.5.5	Compare and test instructions	334
14.5.6	Shift and rotate instructions	335
14.5.7	Data movement instructions	336
14.5.8	Program flow control instructions	338
14.5.9	Interrupt related instructions	341
14.6	Assembler directives	342
14.6.1	The KCPSM3 directives	342
14.6.2	The PBlazeIDE directives	342
14.7	Bibliographic notes	343
<b>15</b>	<b>PicoBlaze Assembly Code Development</b>	<b>345</b>
15.1	Introduction	345
15.2	Useful code segments	345
15.2.1	KCPSM3 conventions	345
15.2.2	Bit manipulation	346
15.2.3	Multiple-byte manipulation	347
15.2.4	Control structure	348
15.3	Subroutine development	350
15.4	Program development	351

15.4.1	Demonstration example	352
15.4.2	Program documentation	356
15.5	Processing of the assembly code	358
15.5.1	Compiling with KCSPM3	358
15.5.2	Simulation by PBlazeIDE	359
15.5.3	Reloading code via the JTAG port	362
15.5.4	Compiling by PBlazeIDE	362
15.6	Syntheses with PicoBlaze	363
15.7	Bibliographic notes	364
15.8	Suggested experiments	365
15.8.1	Signed multiplication	365
15.8.2	Multi-byte multiplication	365
15.8.3	Barrel shift function	365
15.8.4	Reverse function	365
15.8.5	Binary-to-BCD conversion	365
15.8.6	BCD-to-binary conversion	365
15.8.7	Heartbeat circuit	365
15.8.8	Rotating LED circuit	366
15.8.9	Discrete LED dimmer	366
<b>16</b>	<b>PicoBlaze I/O Interface</b>	<b>367</b>
16.1	Introduction	367
16.2	Output port	368
16.2.1	Output instruction and timing	368
16.2.2	Output interface	369
16.3	Input port	371
16.3.1	Input instruction and timing	371
16.3.2	Input interface	371
16.4	Square program with a switch and seven-segment LED display interface	373
16.4.1	Output interface	374
16.4.2	Input interface	375
16.4.3	Assembly code development	376
16.4.4	VHDL code development	384
16.5	Square program with a combinational multiplier and UART console	386
16.5.1	Multiplier interface	387
16.5.2	UART interface	387
16.5.3	Assembly code development	389
16.5.4	VHDL code development	398
16.6	Bibliographic notes	402
16.7	Suggested experiments	402
16.7.1	Low-frequency counter I	402
16.7.2	Low-frequency counter II	402

16.7.3	Auto-scaled low-frequency counter	402
16.7.4	Basic reaction timer with a software timer	403
16.7.5	Basic reaction timer with a hardware timer	403
16.7.6	Enhanced reaction timer	403
16.7.7	Small-screen mouse scribble circuit	403
16.7.8	Full-screen mouse scribble circuit	403
16.7.9	Enhanced rotating banner	403
16.7.10	Pong game	404
16.7.11	Text editor	404
<b>17</b>	<b>PicoBlaze Interrupt Interface</b>	<b>405</b>
17.1	Introduction	405
17.2	Interrupt handling in PicoBlaze	405
17.2.1	Software processing	406
17.2.2	Timing	407
17.3	External interface	408
17.3.1	Single interrupt request	408
17.3.2	Multiple interrupt requests	408
17.4	Software development considerations	409
17.4.1	Interrupt as an alternative scheduling scheme	409
17.4.2	Development of an interrupt service routine	410
17.5	Design example	410
17.5.1	Interrupt interface	410
17.5.2	Interrupt service routine development	411
17.5.3	Assembly code development	411
17.5.4	VHDL code development	413
17.6	Bibliographic notes	417
17.7	Suggested experiments	417
17.7.1	Alternative timer interrupt service routine	417
17.7.2	Programmable timer	417
17.7.3	Set-button interrupt service routine	417
17.7.4	Interrupt interface with two requests	417
17.7.5	Four-request interrupt controller	418
<b>Appendix A:</b>	<b>Sample VHDL templates</b>	<b>419</b>
A.1	General VHDL constructs	419
A.1.1	Overall code structure	419
A.1.2	Component instantiation	420
A.2	Combinational circuits	421
A.2.1	Arithmetic operations	421
A.2.2	Fixed-amount shift operations	422

A.2.3	Routing with concurrent statements	422
A.2.4	Routing with if and case statements	423
A.2.5	Combinational circuit using process	424
A.3	Memory Components	425
A.3.1	Register template	425
A.3.2	Register file	426
A.4	Regular sequential circuits	427
A.5	FSM	428
A.6	FSMD	430
A.7	S3 board constraint file (s3.ucf)	433
References		437
Topic Index		439