

FPGAs

Einstieg, Schaltungen, Projekte

DAS INHALTS- VERZEICHNIS

» Hier geht's
direkt
zum Buch

Inhalt

Materialien zum Buch	13
----------------------------	----

1 Los geht's: FPGAs für Maker und Kreative 15

1.1 Über dieses Buch: Was Sie lernen werden und was Sie schon können sollten	15
1.1.1 Für wen ist dieses Buch geeignet?	16
1.1.2 Was dieses Buch nicht ist	16
1.1.3 Was ist der Inhalt dieses Buches?	16
1.1.4 Das Lernkonzept dieses Buches	17
1.2 Ihr Weg durch dieses Buch	18
1.3 Was sind FPGAs?	20
1.4 Sprachen, Tools und Konzepte	23
1.4.1 Sprache: Warum Verilog?	24
1.4.2 Tools: Open-Source	24
1.4.3 Konzepte: Praktische Beispielprojekte und Learning by Doing	25

TEIL I Einstieg in die FPGA-Programmierung

2 FPGAs verstehen 29

2.1 Grundlagen und Überblick: Die FPGA-Technologie verstehen	30
2.1.1 Was ist ein FPGA genau?	30
2.1.2 Das Grundkonzept eines FPGAs	32
2.1.3 Geschichte und Entwicklung von FPGAs	34
2.1.4 Vorteile und Grenzen von FPGAs	36
2.1.5 Wo werden FPGAs verwendet?	38
2.1.6 Die Grundkomponenten aller FPGAs	40
2.1.7 Spezifische Komponenten unterschiedlicher FPGAs	46
2.1.8 Verarbeitungsgeschwindigkeit und Echtzeitfähigkeit	49
2.1.9 Der Vergleich im Überblick	51
2.1.10 Die vielfältige Namensgebung in FPGA-Designs	53
2.1.11 FPGA vs. Mikrocontroller: Stärken und Schwächen	54
2.2 Die FPGAs, die in diesem Buch verwendet werden	59
2.2.1 Der iCE40HX-FPGA von Lattice Semiconductor im Detail	60
2.2.2 Der GW1NR-9C-FPGA von Gowin Semiconductor im Detail	63
2.2.3 Überblick über die Möglichkeiten und Einsatzbereiche der beiden FPGAs	67

2.3	Was ist der richtige FPGA für Ihr Projekt?	68
2.4	Andere FPGAs und Regulatorien	73
2.5	Open-Hardware- und EDU-Boards	75
2.5.1	F4PGA	78
2.5.2	IceStorm	78
2.5.3	Board- vs. Chip-Unterstützung	80
2.5.4	Stolperfallen bei der Board-Auswahl	81
2.5.5	Ein nicht direkt unterstütztes Board in der Toolchain einrichten	82
2.5.6	Den openFPGALoader für das Programmieren verwenden	83

3 Offener Quellcode, offene Hardware 85

3.1	Kein Ablaufdatum und zeitlos gültig: Open-Source-Software und -Hardware	85
3.2	Die Open-Source-Toolchain und die Freiheit der Entwicklung	87
3.3	Das IceZero-Board für den Raspberry Pi	88
3.4	Der Tang Nano 9K	91
3.5	Die Vor- und Nachteile der Anschlüsse beider FPGA-Boards	92
3.6	Die Open-Source-Toolchain für das IceZero- und das Tang-Nano-Board	95
3.7	Die Hersteller-IDE für den Tang Nano 9K	95
3.8	Beide FPGAs gleichzeitig am Raspberry Pi nutzen	96

4 Software-Entwicklung leicht gemacht 99

4.1	Die Hardware-Beschreibung mit Verilog	100
4.2	Hardware-Beschreibung vs. Programmierung	101
4.3	Die Idee und die FPGAs im Rampenlicht, KI im Support	102
4.3.1	Generative KI	105
4.3.2	Copilot und ChatGPT	108
4.3.3	Welche KI nutzen?	110
4.4	Einen FPGA mit Prompts designen	111
4.5	Tipps und Tricks beim Prompten	115
4.5.1	Weisen Sie der KI eine Rolle zu	115
4.5.2	Konkret und spezifisch	115
4.5.3	Der Kontext ist wichtig	116
4.5.4	Viele Kommentare und gute Namen	116
4.5.5	Recherche und Zeitersparnis	117

5	Offene Tools und Setup	119
5.1	Unsere Steuerzentrale: Der Raspberry Pi	120
5.1.1	Das richtige Modell	120
5.1.2	Den Raspberry Pi einrichten	123
5.2	Die Entwicklungsumgebung einrichten	125
5.2.1	Projektstruktur	125
5.2.2	Versionsverwaltung mit Git	126
5.2.3	Make und Makefiles	136
5.2.4	Best Practices bei der Arbeit in der Entwicklungsumgebung	139
5.2.5	Backup der Entwicklungsumgebung und der eigenen Projekte	140
6	Erste Schritte mit FPGAs und dem Raspberry Pi	143
6.1	Der iCE40HX auf dem IceZero-Board	147
6.1.1	Die Toolchain	147
6.1.2	Das Installationsskript für die IceZero-Toolchain	151
6.1.3	Das Blinky-Projekt zum Einstieg	153
6.2	Der Tang Nano 9K	158
6.2.1	Das Installationsskript für die Tang-Nano-9K-Toolchain	160
6.2.2	Das Lauflicht-Projekt zur Funktionsüberprüfung des Tang Nano 9K	165
6.3	Sind die Toolchain-Komponenten austauschbar?	171
6.4	Quellcode-Editoren für Verilog	172
6.4.1	Visual Studio Code (VS Code)	172
6.4.2	Sublime Text	174
6.4.3	Atom und sein Nachfolger Pulsar	174
6.4.4	Die Klassiker	175
6.5	Die Ausgabeinformationen der Toolchain verstehen	177
7	Die Grundlagen von Verilog	183
7.1	Aufbau und Struktur	183
7.2	Ein erstes Code-Beispiel	186
7.3	Datentypen	187
7.3.1	Wertebereich	187
7.3.2	Netz-Datentypen	189
7.3.3	Variablen-Datentypen	191
7.3.4	Arrays	192

7.4	Kontrollstrukturen	194
7.4.1	Bedingte Anweisungen	194
7.4.2	Schleifen	195
7.5	Von der Software zur Hardware: Automatische Codegenerierung für FPGAs	196
7.5.1	Software-Tools zur automatischen Codegenerierung	197
7.5.2	Hochsprachen für Testbenches	199
7.5.3	Durchgängiges Testen komplexer eingebetteter Systeme mit Python	199
8	Ablauf eines FPGA-Designs	201
8.1	Entwicklungsschritte	201
8.2	Aufbau und Ablauf der FPGA-Projekte in diesem Buch	204

TEIL II Praxis und Projekte

9	Projekte mit dem IceZero	209
9.1	I/O: Interaktion mit der Außenwelt	210
9.1.1	P1 Pmod – eine vielseitige I/O-Schnittstelle	211
9.1.2	P2 Pmod – fokussiert auf Hochgeschwindigkeits-I/O	212
9.1.3	P3 Pmod – erweiterte I/O-Optionen	212
9.1.4	P4 Pmod – spezialisierte I/O-Fähigkeiten	212
9.1.5	J1 – eine Schnittstelle für allgemeine Aufgaben	214
9.1.6	LEDs und Taster	215
9.1.7	Der Takt des IceZero-FPGAs	216
9.1.8	J3 – Die FTDI-Schnittstelle	217
9.1.9	Die GPIO-Schnittstelle	218
9.1.10	J2 – die Spannungsversorgung	220
9.1.11	Der SRAM-Speicher des IceZero-Boards	221
9.2	Einen Takt ausgeben	224
9.2.1	Die ersten Schritte	224
9.2.2	Die Constraints-Datei	225
9.2.3	Der Verilog-Code und der Prompt	225
9.2.4	Das Makefile	229
9.2.5	Konfiguration des FPGAs	233
9.3	Einen Zähler implementieren	233
9.3.1	Die ersten Schritte	234
9.3.2	Die Constraints-Datei	234
9.3.3	Der Verilog-Code und der Prompt	234

9.4	RGB-LEDs ansteuern	237
9.4.1	Was sind Neopixel und RGB-LEDs?	238
9.4.2	Die ersten Schritte	240
9.4.3	Die Constraints-Datei	240
9.4.4	Das WS2812B-Protokoll	240
9.4.5	Timing	241
9.4.6	Versorgungsspannung und Level-Shifter	243
9.4.7	Der Verilog-Code und der Prompt	245
9.4.8	Das Makefile	251
9.5	Serielle Daten übertragen	252
9.5.1	Die Constraints-Datei	253
9.5.2	Der Verilog-Code und der Prompt	254
9.5.3	Erklärung des Quellcodes	259
9.5.4	Eine Testbench für das UART-Programm	261
9.5.5	Mit GTKWave VCD-Dateien analysieren	268
9.5.6	Konfiguration des FPGAs	270
9.5.7	Die serielle Verbindung einrichten	271
9.5.8	Testbench-Techniken und die konditionale Kompilation in Verilog	274
9.6	Web-Control-Server	276
9.6.1	Die Steuerungsebene und die Datenebene	277
9.6.2	Die Constraints-Datei	278
9.6.3	Der Verilog-Code	278
9.6.4	Der Python-Code	279
9.6.5	Konfiguration des FPGAs	281
9.6.6	Erweiterte Steuerungsmöglichkeiten	282
9.7	Retrospektive: Erlernte Grundlagen	283
9.7.1	Was bedeutet dies alles?	285
9.7.2	Grundlagen verinnerlichen und Wissen ausbauen	286
9.8	Mehr zu Testbench-Techniken: Simulation und Test von digitalen Signalverarbeitungssystemen	287
9.8.1	Testsignale erstellen	288
9.8.2	Externe Programme zur Signalgenerierung nutzen	288
9.8.3	Automatisierung der Testprozesse	288
9.8.4	Alternative I/O-Methoden für DSP-Tests	288
9.8.5	Einsatz von cocotb für DSP-Simulationen	289
9.8.6	Das Einlesen der generierten Testsignale	290
9.9	Mehr zu Makefiles: Automatisierung	293
10	Projekte mit dem Tang Nano 9K	295
10.1	I/O: Die 48 GPIO-Pins des Tang Nano 9K	295
10.2	Die Taster des Tang Nano 9K	301

10.3	Der Takt des Tang Nano 9K	301
10.4	Der Speicher eines FPGAs	301
10.5	Eine UART-Schnittstelle	310
10.5.1	Die Datei »Constraints«	312
10.5.2	Der Verilog-Code	314
10.5.3	Erklärung des Quellcodes	321
10.5.4	Das Makefile	326
10.5.5	Konfiguration des FPGAs	329
10.5.6	Die serielle Verbindung	330
10.6	Ein SPI-LCD ansteuern	331
10.6.1	Erklärung des Quellcodes	332
10.6.2	Portierung auf einen anderen FPGA	334
10.6.3	Konfiguration des FPGAs	336
10.6.4	Übung: Das Kombinieren von Modulen	336
10.7	Ein OLED-Display ansteuern	337
10.7.1	Das Funktionsprinzip des SSD1306-Controllers und SPI-Protokolls	339
10.7.2	Die Constraints-Datei	345
10.7.3	Verkabelung	348
10.7.4	Die Zustandsmaschine zur Ansteuerung des Displays	351
10.7.5	Verilog-Code	352
10.7.6	Das Makefile	360
10.7.7	Konfiguration des FPGAs	363
10.7.8	Was haben Sie gelernt?	363
10.8	Ein OLED-Display mit Bildanzeige	364
10.8.1	Ein Bild auf das Display laden	364
10.8.2	Bilddaten umwandeln	365
10.8.3	Konfiguration des FPGAs	369
10.8.4	Was haben Sie gelernt?	369
10.9	Ein OLED-Display mit Text-Engine	370
10.9.1	Buchstaben und Zeichen auf dem Display anzeigen	370
10.9.2	Buchstaben in Pixeldaten umwandeln	373
10.9.3	Konvertierung der Schriftartdatei	374
10.9.4	Das Design der Text-Engine	380
10.9.5	Die Zuordnungslogik der Adressen	382
10.9.6	Der Verilog-Code	387
10.9.7	Das Makefile	402
10.9.8	Den FPGA konfigurieren	402
10.9.9	Was haben Sie gelernt?	402
10.10	Ein OLED-Display mit Bargraphen (und die Frage, wann Zufall wirklich zufällig ist)	403
10.10.1	Ein LFSR	404
10.10.2	Pseudo Random	407

10.10.3	Der Verilog-Code	410
10.10.4	Generalisierung des LFSR-Moduls	412
10.10.5	Den Graphen ausgeben	417
10.10.6	Den Graphen zeichnen	420
10.10.7	Die Constraints-Datei	424
10.10.8	Das Makefile	424
10.10.9	Konfiguration des FPGAs	426
10.10.10	Was haben Sie gelernt?	426
10.11	Ein OLED-Display mit ADC-Ausgabe	427
10.11.1	Aufbau des I ² C-Protokolls	428
10.11.2	Der ADS1115	430
10.11.3	Die Kommunikation	434
10.11.4	Sub-Tasks	437
10.11.5	Der Verilog-Code	442
10.11.6	Die I ² C-Zustände	445
10.11.7	Das ADC-Modul	450
10.11.8	Die Constraints-Datei	470
10.11.9	Die Werte auf dem Display anzeigen	471
10.11.10	Das Makefile	474
10.11.11	Den FPGA konfigurieren	476
10.11.12	Was haben Sie gelernt?	476
10.12	Einen Servo steuern	477
10.12.1	Die Constraints-Datei	477
10.12.2	Der Verilog-Code	478
10.12.3	Fehlersuche und Debugging	482
10.12.4	Was haben Sie gelernt?	482
10.13	Retrospektive: Wie passt das alles zusammen?	483
11	Spezialthemen – kurz angerissen	487
11.1	Schleifen	487
11.1.1	Schleifen in Software	487
11.1.2	Schleifen, die im FPGA geschrieben werden	488
11.1.3	Die Alternative: Zustandsmaschinen	489
11.2	Generate-Anweisungen	490
11.3	Pipelining	492
11.4	Taktdomänen (Clock Domains)	496
11.4.1	Datenübertragung: Langsame Quelle zu schneller Senke	496
11.4.2	Datenübertragung: Schnelle Quelle zu langsamer Senke	496
11.4.3	Herausforderungen und Lösungen	496
11.4.4	Best Practices	497

11.5	Crossing Clock Domains	499
11.5.1	Synchronizer	499
11.5.2	Puffer und FIFOs	500
11.6	Die Bezeichnungen »größer« und »breiter«	502
11.7	Die Simulation stoppen	504
11.8	IP-Cores	505
11.8.1	Spezielle IP-Cores	505
11.8.2	Open-Source- und Community-Ressourcen	506
11.9	Grundlagen zu SerDes in FPGA-Systemen	507
11.10	Phase-Locked Loops (PLL)	509
11.11	Delay-Locked Loops (DLL)	510
11.12	Multiplexer und Demultiplexer	512
11.13	Das Shift-Register	513
11.14	RISC-V-Softcore-CPU	514
11.15	Retrospektive: Gut, mal gehört zu haben	518
12	Die Community und weiterführende Ressourcen	519
12.1	Machen Sie bei Open-Source-Communitys für Maker mit	519
12.2	Retro-Messen: Entdecken Sie 8-Bit-Computer und Emulatoren	521
12.3	Der MEGA65	522
12.3.1	Back to the Future!	523
12.3.2	Artix 7 200T vs. Tang Nano 9K	525
12.4	Im Gespräch mit einem FPGA-Entwickler	530
13	Ihr Traumprojekt: Legen Sie los!	533
	Danke und bis bald!	535
	Index	537