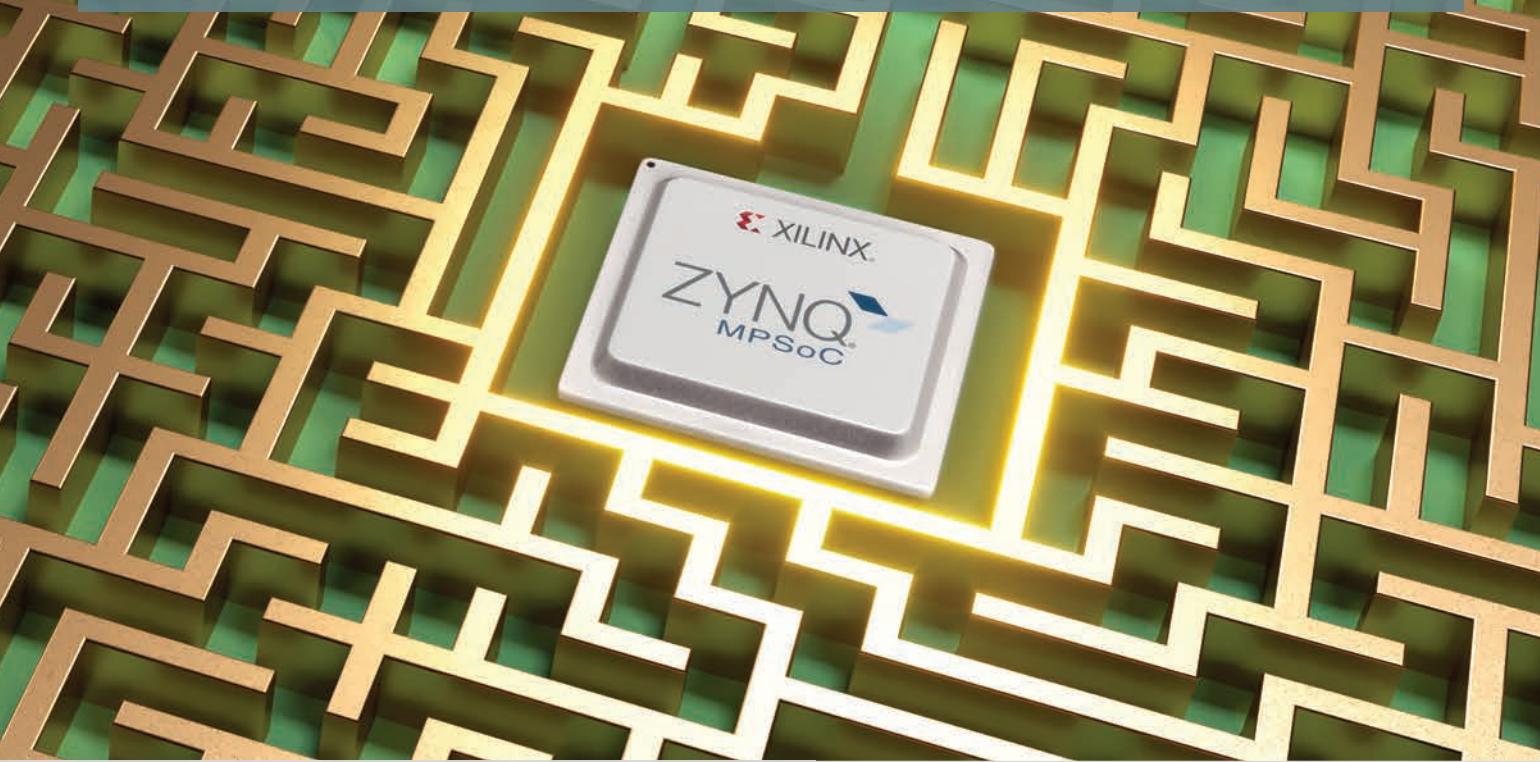


Exploring Zynq® MPSoC

With PYNQ and
Machine Learning Applications



Exploring Zynq® MPSoC

With PYNQ and Machine Learning Applications

Louise H. Crockett

David Northcote

Craig Ramsay

Fraser D. Robinson

Robert W. Stewart

Department of Electronic & Electrical Engineering

University of Strathclyde

Glasgow, Scotland, UK.

April 2019

Table of Contents

Foreword	xvii
Acknowledgements	xix
1) Introduction	1
1.1) Why Should I be Interested?	2
1.2) The Evolution of Xilinx SoCs — Very Briefly!	3
1.3) Design Methods	4
1.4) How to Use this Book	6
1.4.1) Organisation of the Book	6
1.4.2) Further Sources of Information	7
1.4.3) Suggestions for Beginners	7
1.5) What Next?	8
1.6) References	8
Part A: Getting to Know Zynq MPSoC	9
2) FPGAs, Zynq, and Zynq MPSoC!	11
2.1) Technology Timeline	12
2.2) The Zynq-7000 SoC	12
2.2.1) Zynq Architecture and Features	12
2.2.2) Zynq Devices	14
2.2.3) Zynq Use Models	14
2.3) The Xilinx Zynq MPSoC	15
2.3.1) The Release of Zynq MPSoC	15
2.3.2) Zynq MPSoC Architecture and Features	16
2.4) FPGAs	18
2.4.1) What is an FPGA?	18

Table of Contents

2.4.2)	The Development of FPGAs	18
2.4.3)	The Modern FPGA Architecture: UltraScale+	21
2.5)	Comparison and Discussion: FPGA v Zynq v Zynq MPSoC	24
2.5.1)	Architectures	24
2.5.2)	Power Consumption and Performance	25
2.5.3)	Embedded Systems Implementation	27
2.5.4)	Applications	28
2.6)	Chapter Summary	29
2.7)	References	29
3)	An Overview of the Zynq MPSoC Architecture	31
3.1)	Zynq MPSoC Device Families	31
3.1.1)	Arm Versus Xilinx Documentation	33
3.2)	Processing System	33
3.2.1)	Application Processing Unit	35
3.2.2)	Real-Time Processing Unit	37
3.2.3)	Graphics Processing Unit	38
3.2.4)	Connectivity	39
3.3)	Platform Management	42
3.3.1)	Power Modes	42
3.3.2)	Power Domains	43
3.3.3)	Platform Management Unit	44
3.4)	Programmable Logic	45
3.4.1)	The Logic Fabric	46
3.4.2)	Storage and Signal Processing Resources	47
3.4.3)	PL Peripherals	48
3.4.4)	Video Codec Unit	48
3.4.5)	General Purpose Input/Output	48
3.4.6)	High-Speed Connectivity	49
3.4.7)	JTAG Interface	49
3.5)	Processing System and Programmable Logic Interfaces	49
3.5.1)	The Arm AMBA Open Standard	50
3.5.2)	PS to PL Interconnects and Interfaces	52
3.5.3)	EMIO Interfaces	54
3.5.4)	Other PS to PL Interfaces	55
3.6)	Security and Configuration	55

3.6.1) Configuration Security Unit	56
3.7) Chapter Review	58
3.8) References	58
4) Design Tools & Methods for the Zynq MPSoC	63
4.1) Anatomy of a Zynq MPSoC Design	63
4.1.1) Zynq MPSoC Development Boards	64
4.1.2) The Hardware System	65
4.1.3) The Software Stack	68
4.2) The Design Process	69
4.2.1) Requirements and Specification	69
4.2.2) Project Decisions and Planning	70
4.2.3) Design Flow	71
4.2.4) Testing	72
4.3) Getting Ready: Design Tools and Development System Setup	72
4.3.1) Host System	72
4.3.2) Xilinx Software Components	73
4.3.3) Hardware Requirements	80
4.4) Development Boards and Supporting Resources	81
4.5) Resources and Support	82
4.5.1) Information about Zynq MPSoC Devices	82
4.5.2) Software Tools Support	83
4.5.3) Support for System Design	84
4.6) The Wider Ecosystem	84
4.6.1) Hardware System Design Tools and Components	85
4.6.2) Software System Design Tools and Components	87
4.6.3) Algorithm Development and Interfacing	89
4.6.4) Hardware and Peripherals	89
4.7) Chapter Summary	91
4.8) References	91
5) Candidate Application Areas for Zynq MPSoC	97
5.1) What Makes my System a ‘Zynq MPSoC System?’	97
5.2) Application Areas for the Zynq MPSoC (and RFSoC)	99
5.3) Drones	100
5.3.1) Flying	100

Table of Contents

5.3.2)	Video Processing	102
5.3.3)	Sensing	102
5.3.4)	Communications and Navigation	103
5.3.5)	Advanced Drones	104
5.4)	Smart and Autonomous Vehicles	105
5.4.1)	What Does ‘ADAS’ Cover?	106
5.4.2)	Sensing Requirements and Implementation	106
5.4.3)	Autonomous Vehicles	108
5.4.4)	Connectivity and Security	108
5.5)	Interfacing to the RF Analogue World — the Evolution to Zynq RFSoC	109
5.5.1)	The Zynq RFSoC Device Family Compared to MPSoC	110
5.5.2)	RF Sampling: RF-ADCs and RF-DACs, and Single Chip SDR	113
5.5.3)	Direct Sampling Zynq RFSoC Data Converter Subsystem	114
5.5.4)	4GLTE Multicarrier Solutions	116
5.5.5)	5G Mobile and Wireless Implementations	117
5.5.6)	5G mmWave Implementations using IF Architectures	117
5.6)	Chapter Summary	119
5.7)	References	119

Part B: The Zynq MPSoC Architecture in Detail 123

6)	The Application Processing Unit	125
6.1)	The Cortex-A53 MPCore Processor	125
6.2)	Fundamentals of Armv8-A	126
6.2.1)	64-Bit/32-Bit Execution States	127
6.2.2)	Instruction Sets and Programming Languages	128
6.2.3)	Exception Levels	129
6.2.4)	Processor Modes	131
6.2.5)	Armv8 Security Model	133
6.3)	Signals and Interfaces	137
6.3.1)	The Master Memory Interface	138
6.3.2)	ACP Slave Interface	139
6.3.3)	Cross Trigger Interface (CTI)	141
6.3.4)	Advanced Peripheral Bus (APB) Debug	142
6.3.5)	GIC Interface	142
6.3.6)	Trace Interface	142

6.3.7) Other Signals	143
6.4) Memory Management Unit	144
6.5) Memory Systems	144
6.5.1) Level 1 Memory System	145
6.5.2) Level 2 Memory System	146
6.6) Processor Extensions	146
6.6.1) Cryptography (Crypto)	146
6.6.2) NEON Media Processing Engine	146
6.6.3) Vector Floating Point Unit	148
6.7) Interrupts	148
6.7.1) GICv2 Interrupt Types	149
6.7.2) GICv2 Interrupt Priority and States	150
6.7.3) GIC-400 Overview	150
6.8) Power Management	153
6.8.1) Power Domains	153
6.8.2) Power Modes	154
6.9) System Virtualisation	155
6.9.1) Address Translation in a Virtual Environment	157
6.9.2) Interrupt Virtualisation	157
6.10) Chapter Review	159
6.11) References	159
7) The Real-Time Processing Unit	161
7.1) Introduction	161
7.1.1) What is Real-Time Processing?	161
7.1.2) Why a Different Architecture for Real-Time Processing?	163
7.2) Overview	163
7.3) Determinism and Responsiveness	165
7.3.1) Tightly Coupled Memories	165
7.3.2) Interrupt System	166
7.4) Safety within the RPU	170
7.4.1) Memory Protection	171
7.4.2) Split/Lock Modes	174
7.5) Chapter Review	176
7.6) References	176

8) Security in Zynq MPSoC	179
8.1) Information Assurance for Configuration Security	180
8.1.1) Configuration Security Unit Introduction	181
8.1.2) Crypto Blocks	183
8.1.3) Key Management	190
8.2) Anti-Tampering	200
8.2.1) Monitoring	201
8.2.2) Response	202
8.2.3) Precautions	204
8.3) Security Through Isolation	205
8.3.1) Isolating Software with Virtualisation and Armv8	206
8.3.2) Extending Isolation to a Complete System	209
8.3.3) Isolation Summary	213
8.4) Chapter Summary	213
8.5) References	214
9) Safety Features & Techniques	217
9.1) An Introduction to Safety	217
9.1.1) Safety and Dependability	218
9.1.2) Safety Application Examples	219
9.2) Functional Safety	219
9.2.1) What is Functional Safety?	219
9.2.2) Errors, Faults and Failures!	220
9.2.3) Handling of Faults, Errors and Failures	221
9.2.4) Functional Safety Standards	225
9.3) Design Principles and Architectural Support	227
9.3.1) Redundancy	227
9.3.2) Diversity	230
9.3.3) Isolation Design Flow	232
9.3.4) Real-Time Processors: Dual-Lockstep Mode	233
9.3.5) CCFs and System Monitoring	234
9.3.6) Error-Correcting Code (ECC) Memory	235
9.3.7) Fault Injection and Testing	236
9.4) Chapter Summary	237
9.5) References	237

10) Platform Management Features	239
10.1) Power Modes	240
10.1.1) Battery Powered Mode	240
10.1.2) Low-Power Mode	241
10.1.3) Full-Power Mode	241
10.1.4) Deep-Sleep Mode	241
10.2) Power Domains	242
10.2.1) Battery Power Domain	243
10.2.2) Low-Power Domain	244
10.2.3) Full-Power Domain	245
10.2.4) PL Power Domain	246
10.2.5) Other Power Domains	247
10.3) Platform Management Unit	247
10.3.1) PMU Processor	249
10.3.2) Interfaces and AXI Interconnect	250
10.3.3) Local and Global Registers	251
10.3.4) Programmable Interval Timers	251
10.3.5) Interrupts	251
10.4) Error Management	253
10.4.1) Error Status and Enable Registers	253
10.4.2) Interrupt Error Registers	254
10.4.3) Error Handling	254
10.5) PMU Firmware	255
10.5.1) Composition of the PMU Firmware	255
10.5.2) Default Modules	256
10.5.3) Execution of the PMU Firmware	257
10.5.4) Custom Modules	258
10.6) Chapter Review	258
10.7) References	259
Part C: Zynq MPSoC Systems Development	261
11) Hardware System Development	263
11.1) Heterogeneous Computing with Zynq MPSoC	264
11.2) Hardware System Overview	266

Table of Contents

11.2.1) Interfaces and Signals	266
11.2.2) The Interconnect	269
11.2.3) Memories	272
11.3) PL Interfacing	273
11.3.1) Snooping	274
11.3.2) AXI Coherency Extension (ACE) Interface	274
11.3.3) The AXI FIFO Interface	276
11.3.4) PL-FPD AXI Masters	277
11.3.5) PL-LPD AXI Master	277
11.3.6) PL-PS AXI Slaves	278
11.3.7) Selecting a PL Interface	278
11.4) The Interrupt System	280
11.4.1) Interrupt System Overview	280
11.4.2) Types of Interrupts	281
11.4.3) Interrupt Prioritisation, States, and Handling	283
11.4.4) The Inter-Processor Interrupt	284
11.5) Memories	285
11.5.1) The Global Address Space	285
11.5.2) On-Chip Memory	286
11.5.3) DDR Memory Interface	287
11.6) Data Movement Fundamentals	289
11.6.1) Direct Memory Access	289
11.6.2) The AXI Interconnect	291
11.6.3) DMA Controllers in the PS	291
11.6.4) Simple AXI Communication	293
11.6.5) The AXI DMA	294
11.6.6) The AXI Video DMA	296
11.7) Chapter Review	298
11.8) References	298
12) Software Stacks	301
12.1) Bare-Metal Software Stack	301
12.1.1) What does Bare-Metal give us?	302
12.1.2) C Standard Libraries	303
12.1.3) Standalone Libraries	304
12.1.4) Standalone Drivers	305

12.1.5) How can we use it?	305
12.2) FreeRTOS Software Stack	307
12.2.1) What does FreeRTOS give us?	307
12.2.2) Tasks	308
12.2.3) Inter-task Synchronisation	309
12.2.4) Other Utilities	311
12.2.5) How can we use it?	312
12.3) Linux Software Stack	313
12.3.1) What does Linux give us?	314
12.3.2) The Kernel Space and User Space Divide	315
12.3.3) Memory Management	317
12.3.4) Device Drivers	319
12.3.5) Process management	322
12.3.6) Inter-Process Communication	324
12.3.7) How can we use it?	325
12.3.8) Multimedia Software Stack	325
12.4) Chapter Review	326
12.5) References	327
13) Multiprocessor Development	331
13.1) Introduction to Heterogeneous Processing	332
13.1.1) Processor Sets on Zynq MPSoC	332
13.1.2) Splitting Software Tasks	332
13.1.3) Heterogeneous Computing Concepts	334
13.2) Symmetric Multiprocessing with Linux	336
13.3) Asymmetric Multiprocessing	339
13.3.1) Unsupervised AMP with OpenAMP	339
13.3.2) Supervised AMP with the Xen Hypervisor	344
13.4) A Hybrid Example for Software Defined Radio Applications	348
13.5) Chapter Summary	350
13.6) References	351
14) System Booting	353
14.1) Introduction to System Booting	353
14.2) Non-Secure Boot Process	356
14.2.1) Boot Process Overview	356

Table of Contents

14.2.2) Boot Media Options	360
14.2.3) Boot Image Search	361
14.2.4) Boot Image Format	361
14.2.5) Recap with a Linux/OpenAMP Use-Case	363
14.3) Secure Boot Process	364
14.3.1) Secure Boot Fundamentals	364
14.3.2) Secure Boot Cryptography	365
14.3.3) Secure Boot Image Format	369
14.4) Practical Non-Secure Device Configuration	370
14.4.1) Generating Boot Images	370
14.4.2) Programming the Boot Images	371
14.5) Practical Secure Device Configuration	371
14.5.1) Generating Boot Images	372
14.5.2) Device Configuration for Secure Boot	374
14.5.3) Maintaining the Chain of Trust Beyond the FSBL	374
14.6) Chapter Summary	375
14.7) References	375

Part D: System Design with Xilinx SDx Development Environment 377

15) Introduction to System Design with SDx	379
15.1) Motivation for using SDx	379
15.2) About SDx	381
15.2.1) The Development Environment	381
15.2.2) Introducing the Full-System Optimising Compiler	383
15.2.3) Data Motion Networks	383
15.2.4) Directing the SDx Tool-Chain	386
15.2.5) SDx API	389
15.2.6) SDx Tool-Chain Settings	390
15.3) The Design Flow	392
15.3.1) SDx Platform	394
15.3.2) Selecting Initial Candidate Functions for Hardware	395
15.3.3) Estimating System Performance	396
15.3.4) Optimising System Performance	396
15.3.5) Analysing System Performance	397
15.4) SDx Project Hierarchy	398

15.5) Chapter Review	402
15.6) References	403
16) System Profiling and Acceleration with SDx	405
16.1) System Profiling	405
16.1.1) Software-Only System Profiling	405
16.1.2) Estimating Relative Performance	411
16.1.3) Measuring System Performance	412
16.2) Software Acceleration using Programmable Logic	414
16.2.1) IP Core Optimisation	414
16.2.2) Optimising IP Core Integration	416
16.3) Chapter Review	421
16.4) References	422
17) Reusing Existing IP in SDx	423
17.1) Creating a C-Callable Library	423
17.1.1) IP Core Requirements	424
17.1.2) IP Configuration Parameters	425
17.1.3) Function Definition	425
17.1.4) Function Argument Mapping	426
17.1.5) The <i>sdslib</i> Utility	427
17.1.6) Library Header File	428
17.2) Using a C-Callable Library	429
17.3) Chapter Review	430
17.4) References	430
18) Debugging and Performance Monitoring with SDx	431
18.1) System Emulation	432
18.2) Software Debugging	434
18.3) Hardware Debugging	436
18.4) Performance Monitoring	439
18.4.1) Event Tracing	439
18.4.2) AXI Performance Monitor	441
18.4.3) Processing System Performance Monitoring	444
18.5) Chapter Review	445
18.6) References	446

Table of Contents

19) Custom SDx Platforms	447
19.1) Why might we want to develop a custom SDx Platform?	447
19.2) SDx Platform Structure and Components	448
19.2.1) Metadata Files	450
19.2.2) Sample Applications	450
19.3) SDx Platform Hardware Component	451
19.3.1) Developing the Platform Hardware Design	451
19.3.2) Platform Hardware Metadata File	453
19.4) SDx Platform Software Component	457
19.4.1) Boot Files	457
19.4.2) PetaLinux Tools	465
19.4.3) Boot Image Format Files	465
19.4.4) Libraries	466
19.5) SDx Platform Sample Applications	467
19.6) The sdspfm Utility	469
19.7) Pre-Built Hardware	471
19.8) Testing and Using a Custom SDx Platform	473
19.9) Chapter Review	474
19.10) References	474
Part E: PYNQ, and Machine Learning Applications.....	479
20) Deep Learning	481
20.1) Machine Learning is Everywhere (or will be!)	481
20.1.1) Machine Learning versus Deep Learning	481
20.1.2) Machine Learning Overview	483
20.1.3) Established Applications	485
20.1.4) Emerging Applications	486
20.1.5) Platforms for Machine Learning	489
20.2) Common Neural Network Architectures	489
20.2.1) The Neuron	489
20.2.2) Multilayer Perceptron	491
20.2.3) Convolutional Neural Networks (CNNs)	493
20.2.4) Recurrent Neural Networks (RNNs)	495
20.3) Training Neural Networks	497

20.3.1) Loss Functions	498
20.3.2) Backpropagation	499
20.3.3) Stochastic Gradient Descent (SGD)	501
20.3.4) Regularisation	502
20.4) Deep Learning Tools and Frameworks	503
20.5) The Need for Acceleration	504
20.5.1) Cloud and Edge Applications	504
20.5.2) Compression Techniques	505
20.6) Chapter Summary	505
20.7) References	506
21) Reduced Precision Neural Networks	509
21.1) Reduced Precision Neural Networks	509
21.1.1) What is ‘Reduced Precision’?	510
21.1.2) Motivation	511
21.1.3) The Impact on Scaling	511
21.1.4) Neural Network Implementation Options	513
21.2) Introducing FINN-R	514
21.2.1) Origins of FINN-R	514
21.2.2) Reduced Precision Optimisations	514
21.2.3) Architectural Optimisations	514
21.2.4) FINN-R Design Flow	515
21.2.5) Working with FINN-R	516
21.3) Neural Networks on Programmable Logic	516
21.3.1) A Neuron in Programmable Logic	517
21.3.2) Creating Layers	518
21.3.3) Design Methods and Tools	520
21.4) FINN-R Case Study: An Inference Accelerator for Zynq MPSoC	521
21.5) Chapter Summary	522
21.6) References	523
22) PYNQ	525
22.1) Introducing PYNQ	525
22.1.1) PYNQ as a Framework	525
22.1.2) What is PYNQ?	526
22.1.3) Why Python?	529

Table of Contents

22.1.4) What PYNQ Is Not!	530
22.2) An Example PYNQ Application	531
22.2.1) Python Notebook (Software Only)	534
22.2.2) PYNQ Notebook (Hardware Accelerated)	534
22.2.3) System Architecture (Hardware Accelerated Version)	534
22.2.4) Key Observations	537
22.3) Components of a PYNQ System	537
22.3.1) PYNQ Image	537
22.3.2) File Handling	538
22.3.3) Overlays	539
22.3.4) Jupyter Notebooks	542
22.3.5) Files and Installation	544
22.3.6) Advanced User Customisation	545
22.4) The PYNQ Project	546
22.4.1) PYNQ Goals	546
22.4.2) PYNQ Influences	547
22.5) Resources	549
22.5.1) How Do I Get PYNQ?	549
22.5.2) Documentation	549
22.5.3) Files and Source Code	550
22.5.4) Hardware	550
22.5.5) Community Resources	550
22.6) Chapter Summary	551
22.7) References	552
23) Case Study: Prototyping CNNs on Zynq for Space Applications, Using PYNQ.	555
23.1) Design Concept	556
23.2) Cloud Detection with Neural Networks	556
23.2.1) Algorithm Selection	556
23.2.2) Algorithm Development	558
23.3) Hardware Selection	558
23.3.1) Selecting the FPGA	558
23.3.2) Discovering PYNQ	560
23.4) Cloud Detection with PYNQ	560
23.4.1) Developing a Prototype with PYNQ	560
23.4.2) Results	561

23.5) Reflections and Ruminations	562
23.6) References	562
Part F: And Finally...	565
24) Academic Case Studies	567
24.1) Embedded Computer Vision	567
24.1.1) Advanced Driver Assistance Systems	571
24.2) Spaceflight Systems	573
24.3) Machine Learning	574
24.4) Chapter Summary	578
24.5) References	578
Postscript	581
Appendix: BSD 3-Clause Licence	583
List of Acronyms	585
Index	605