

# Preface

Field-Programmable Gate Arrays (FPGAs) were invented by Xilinx in 1985, i.e., less than 30 years ago. The influence of FPGAs on many directions in engineering is growing continuously and rapidly. There are many reasons for such progress and the most important are the inherent configurability of FPGAs and their relatively cheap development cost. Forecasts suggest that the impact of FPGAs will continue to grow and the range of applications will increase considerably in the future. Recent field-configurable microchips incorporate multicore processors and reconfigurable logic appended with a number of frequently used devices such as digital signal processing slices and block memories. FPGA-based systems can be synthesized and implemented in general-purpose computers using integrated design environments. Experiments and explorations of such systems are commonly based on prototyping boards linked to the same environment.

It is widely known and proven that FPGAs can be applied efficiently in a vast variety of engineering applications. One reason for this is that growing system complexity makes it very difficult to ship designs without errors. Hence, it is essential to be able to fix errors after fabrication, which can be done significantly easier with customizable devices.

The complexity of contemporary chips is increasing exponentially with time and the number of available transistors grows faster than the ability to design meaningfully with them. This situation is a well-known design productivity gap, which is increasing continuously. Therefore, design productivity will be the real challenge for future systems. Although in unit production volumes and revenue, Application-Specific Integrated Circuits (ASICs) and Application-Specific Standard Products (ASSPs) surpass FPGAs, forecasts of FPGA design start numbers are currently ahead of ASIC/ASSP design starts. Thus, the high involvement of FPGAs in new designs of circuits and systems and the need for better design productivity undoubtedly require huge engineering resources, which are the major output of technical universities, and this book is intended to provide assistance for the relevant courses.

FPGAs still operate at lower clock frequencies than general-purpose computers and ASICs. The cost of the most advanced devices is high. Cheaper microchips operate at clock frequencies that are lower than in inexpensive computers that are widely used. One of the most important applications of FPGAs is improving the

performance of implemented systems. To achieve acceleration with devices that are generally slower, parallelism needs to be applied extensively.

The book pursues two main objectives and is composed of two parts. The **first part** with appendices A and B (written by Valery Sklyarov and Ioulia Skliarova) introduces the concepts of the design of digital systems using contemporary Field-Programmable Gate Arrays and presents the recent results of the authors in FPGA-based high-performance accelerators. This part is composed of five chapters with the main objective of extending topics that are traditionally included within digital systems in a way that enables FPGA-based design to be discussed, illustrated by examples, and supported by experiments with relatively cheap prototyping boards that are widely available. The **second part** of the book (written by Alexander Barkalov and Larisa Titarenko) includes four chapters and covers more theoretical aspects of finite state machines (FSMs) with the main objective of reducing FPGA basic resources (slices or look-up tables), minimizing delays in the circuits, and achieving greater optimization of fundamental components in FPGAs.

The following features set the book apart from others in the field:

1. It provides easily understandable introductory sections (appropriate even for the first-year students in the area) that are gradually extended to more advanced topics covering the novel techniques that are proposed and disseminated by the authors and demonstrated in numerous examples from practical applications.
2. Fully synthesizable hardware-description language specifications (VHDL, in particular) for the majority of the circuits and systems described are presented ready to be tested and incorporated in practical engineering designs, which is indispensable for both undergraduate and postgraduate university students.
3. A number of practical designs based on the proposed models and methods for complete applications are discussed from areas such as data processing, combinatorial search, and computations relying on the model of a hierarchical finite state machine.
4. Exploring models and methods that involve not only core reconfigurable logical elements but also a number of embedded blocks (e.g., memories and digital signal processing slices) and template-based circuits.

The book provides the following additional features:

1. The design examples have been tested in three prototyping boards with Xilinx and Altera FPGAs. The latest Nexys-4 board from Digilent with the recent Artix-7 FPGA from the Xilinx 7 series and the well-known Digilent Atlys board with the Xilinx Spartan-6 FPGA were used for the majority of the examples. Many projects were also tested in the DE2-115 board with the Altera Cyclon-IVe FPGA that was developed especially for education and is popular in university courses.
2. All the VHDL examples from the book are available online at <http://sweet.ua.pt/skl/Springer2014.html>. The website also provides the latest updated projects. These projects can be downloaded and tested and evaluated immediately. Each example includes a brief description, the VHDL code, a user constraints file, and a bitstream for the selected FPGA.

The chapters in the book contain the following material:

**Chapter 1** introduces FPGA architectures by presenting the general structure of modern devices and explaining the core elements and the most important embedded blocks, such as memory and digital signal processing slices. A few typical FPGA-based design scenarios are discussed that cover the phases of specification, supplying physical constraints, implementation, configuration, and finally, testing. In this introductory chapter, design specifications are presented at the schematic level, where a circuit is constructed either from components available in vendor-specific libraries, user-defined blocks, or from properly customized intellectual property cores. A number of simple examples are given that are ready to be tested in FPGA-based prototyping boards. The three prototyping boards used in the book are characterized briefly and the general ideas for interaction with circuits and systems implemented in FPGAs are introduced. All the processing steps are explained through numerous examples.

**Chapter 2** presents a concise introduction to synthesizable VHDL that is sufficient for the design methods and examples given in subsequent chapters to be understood without much background knowledge. The main objective of this chapter is to explain the basis of VHDL modules and their specification capabilities without going into detail. There are many excellent books dedicated to VHDL that may be used to complement this book. Our primary target is the synthesis and optimization of FPGA-based circuits and systems and VHDL is just an instrument that is used in the book to describe the desired functionalities and structures. Thus this chapter only provides the minimum necessary to allow subsequent chapters to be read without additional material, and to enable all the proposed examples to be understood and tested with the FPGA-based prototyping boards.

**Chapter 3** begins with a brief description of widely used simple combinational and sequential circuits. Many examples are given with implementations of the circuits in FPGAs. Next, various optimization techniques are discussed with special emphasis on broad parallelism, which is very important for FPGA-based applications. More complicated digital circuits and systems are introduced, such as parallel networks for sorting and searching, hamming weight counters/comparators, concurrent vector processing units, and advanced finite state machines. The circuits are designed so that operations over multiple data items can be executed concurrently. Network-based solutions, such as sorting and counting networks in particular, and the efficient mapping of circuits to FPGA primitives (look-up tables) are examples. A number of alternative competing methods are discussed and evaluated. All the circuits and systems are described in VHDL, implemented and tested in FPGAs, and finally evaluated by applying various criteria. Many of the novel solutions proposed are parameterized, which permits very complex projects to be developed in FPGAs for solving advanced problems in several areas, such as data processing and combinatorial search.

**Chapter 4** begins with examples that demonstrate how commercially available intellectual property cores can be embedded in different designs. In particular, arithmetical circuits constructed from digital signal processing slices, and parameterized memory blocks that provide support for data buffering (such as

FIFO—first input first output), are described. More details on digital signal processing slices are then given and it is shown how these may be used efficiently in practical circuits such as hamming weight counters/comparators. The major part of this chapter is dedicated to interactions between a host computer and FPGA-based prototyping boards through the Digilent enhanced parallel port and the UART (Universal Asynchronous Receiver and Transmitter) interfaces. Complete details of the communication modules are described, including both software for general-purpose computers that was developed in the C++ language, and hardware for FPGAs. The next section makes use of the designed modules for projects that involve such interactions for different purposes. A more complicated design for a network-based iterative data sorter from Chap. 3 is implemented and tested in this way as a complete fully functioning example. The chapter concludes with a brief description of programmable systems-on-chip (PSoC) that combine an embedded processing system with a reconfigurable logic, which can lead to more efficient implementations of the applications. Proposals for mapping the designs from Chap. 3 to the PSoC are given and discussed.

Chapter 5 gives an overview of the design techniques based on hierarchical and parallel specifications. First, hierarchical graph-schemes (HGSs) are introduced that enable complex digital control algorithms to be decomposed and described efficiently. A module, described by an HGS, is the fundamental entity that provides the basis for the technique, and is an autonomous, complete, and potentially reusable component. A module has to be designed such that: (1) it can be verified independently of other modules; (2) it possesses a well-defined external interface so it can be reused in different specifications. It is shown that a set of HGSs (modules) can be implemented in a hierarchical finite state machine (HFSM) with a stack memory. Many VHDL examples are given that demonstrate that HFSMs permit the execution of hierarchical algorithms and provide support for recursion if required. Various types of HFSMs are described and synthesizable VHDL templates for these are given that can be customized for particular problems. Parallel specifications and parallel HFSMs are also discussed. Many fully functioning VHDL examples for all the types of HFSMs above are presented and evaluated. It is also shown how software programs can be mapped to hardware with the aid of HFSM models. Finally, a variety of HFSM optimization techniques are proposed.

Chapter 6 is devoted to the problems of optimization of Moore FSM's logic circuits implemented with FPGAs. The general characteristic is given for methods of functional and structural decomposition. Distinctive features of FPGA are analyzed allowing the number of look-up table (LUT) elements in logic circuits of Moore FSMs to be decreased. The classification of optimization methods are given for Moore FSM including: (1) the transformation of state codes into codes of the classes of pseudo-equivalent states (PES); (2) presentation of state codes as concatenations of codes of PES and collections of microoperations; (3) replacement of logical conditions (input variables of FSM) with additional variables. All discussed methods are illustrated by examples.

Chapter 7 deals with design of Moore FSMs based on using embedded memory blocks (EMB). The methods of trivial EMB-based implementation of logic circuits

of both Moore and Mealy FSMs are discussed. In this case, only one EMB is enough for implementing the circuit. Next, the optimization methods are discussed based on replacement of logical conditions as well as encoding of the collections of microoperations. The considered methods are based on encoding the rows of FSM's structure table. All these methods lead to two-level models of Mealy FSMs and to three-level models of Moore FSMs. Next, these methods are combined together for further optimization of hardware in FSM logic circuits. The last section considers applying PES-based methods in EMB-based Moore FSMs. All discussed methods are illustrated by examples.

**Chapter 8** is devoted to optimization of logic circuits of EMB-based FSMs. First of all, the design methods based on the replacement of logical conditions are discussed for both Moore and Mealy FSMs. Next, the proposed optimization methods are presented. These methods are based on splitting the set of logical conditions. This approach allows decreasing the number of LUTs in the circuit of the block of replacement of logical conditions. In the case of Moore FSM, the optimization methods are based on optimal state assignment, as well as the transformation of state codes into codes of the classes of PES. All discussed methods are illustrated by examples.

**Chapter 9** is devoted to using the datapath for decreasing the number of LUTs in logic circuits of FPGA-based Moore FSMs. Firstly, the principle of operational implementation of interstate transitions is proposed. It is based on the usage of operational elements (adders, counters, shifters, and so on) for calculating codes the states of transitions. Next, the organization of FSM with operational implementation of interstate transitions is discussed. An example is given for application of the proposed method. Next, the base structure of synthesis process is proposed for Moore FSM with operational implementation of interstate transitions. The structure of the synthesis process depends on initial conditions such as set of operations or codes of FSM states. The typical structures are discussed for the operational automaton executing the transitions. Next, the method is shown based on mixture of traditional and proposed approaches for calculation of the codes of states of transitions. The last part of the chapter discusses the efficiency of the proposed solutions.

Appendix A contains a short description used in the book synthesizable VHDL constructions and reserved words in alphabetical order with examples.

Appendix B offers a number of synthesizable VHDL specifications that provide support for many projects in the part I of the book. All the examples are presented so that they can be tried out and examined directly.

The book can be used as supporting material for university courses that involve FPGA-based design, such as "Digital design," "Computer architecture," "Electronics," "Embedded systems," "Reconfigurable computing," "Communications," and "FPGA-based systems." It will also be helpful in engineering practice and research activity in areas where FPGA-based circuits and systems are planned to be designed and investigated. It is important to note that the presented fully functioning VHDL projects (that are also available online at <http://sweet.ua.pt/skl/Springer2014.html>) can be used directly in many research and engineering applications.

Synthesis and Optimization of FPGA-Based Systems

Sklyarov, V.; Skliarova, I.; Barkalov, A.; Titarenko, L.

2014, XIX, 432 p. 235 illus., Hardcover

ISBN: 978-3-319-04707-2