

# Preface

Field Programmable Gate Arrays constitute one of the technologies at hand for developing electronic systems. They form an attractive option for small production quantities as their non-recurrent engineering costs are much lower than those corresponding to ASIC's. They also offer flexibility and fast time-to-market. Furthermore, in order to reduce their size and, hence, the unit cost, an interesting possibility is to reconfigure them at run time so that the same programmable device can execute different predefined functions.

Complex systems, generally, are made up of processors executing programs, memories, buses, input-output interfaces, and other peripherals of different types. Those components are available under the form of Intellectual Property (IP) cores (synthesizable Hardware Description Language descriptions or even physical descriptions). Some systems also include specific components implementing algorithms whose execution on an instruction-set processor is too slow. Typical examples of such complex algorithms are: long-operand arithmetic operations, floating-point operations, encoding and processing of different types of signals, data ciphering, and many others. The way those specific components can be developed is the central topic of this book. So, it addresses to both, FPGA users interested in developing new specific components—generally for reducing execution times—and, IP core designers interested in extending their catalog of specific components.

This book distinguishes itself with the following aspects:

- The main topic is circuit synthesis. Given an algorithm executing some complex function, how can it be translated to a synthesizable circuit description? Which are the choices that the designer can make in order to reduce the circuit cost, latency, or power consumption? Thus, this is not a book on algorithms. It is a book on “how to efficiently translate an algorithm to a circuit” using, for that purpose, techniques such as parallelism, pipeline, loop unrolling, and others. In particular, this is not a new version of a previous book by two of the authors.<sup>1</sup>

---

<sup>1</sup> Deschamps JP, Bioul G, Sutter G (2006) *Synthesis of Arithmetic Circuits*. Wiley, New York.

It is a complement of the cited book; its aim is the generation of efficient implementations of some of the most useful arithmetic functions.

- All throughout this book, numerous examples of FPGA implementation are described. The circuits are modeled in VHDL. Complete and synthesizable source files are available at the authors' web site [www.arithmetic-circuits.org](http://www.arithmetic-circuits.org).
- This is not a book on Hardware Description Languages, and the reader is assumed to have a basic knowledge of VHDL.
- It is not a book on Logic Circuits either, and the reader is assumed to be familiarized with the basic concepts of combinational and sequential circuits.

## Overview

This book is divided into sixteen chapters. In the first chapter the basic building blocks of digital systems are briefly reviewed, and their VHDL descriptions are presented. It constitutes a bridge with previous courses, or books, on Hardware Description Languages and Logic Circuits.

[Chapters 2–4](#) constitute a first part whose aim is the description of the basic principles and methods of algorithm implementation. [Chapter 2](#) describes the breaking up of a circuit into Data Path and Control Unit, and tackles the scheduling and resource assignment problems. In [Chaps. 3](#) and [4](#) some special topics of Data Path and Control Unit synthesis are presented.

[Chapter 5](#) recalls important electronic concepts that must be taken into account for getting reliable circuits and [Chap. 6](#) gives information about the main Electronic Design Automation (EDA) tools that are available for developing systems on FPGAs.

[Chapters 7–13](#) are dedicated to the main arithmetic operations, namely addition ([Chap. 7](#)), multiplication ([Chap. 8](#)), division ([Chap. 9](#)), other operations such as square root, logarithm, exponentiation, trigonometric functions, base conversion ([Chap. 10](#)), decimal arithmetic ([Chap. 11](#)), floating-point arithmetic ([Chap. 12](#)), and finite-field arithmetic ([Chap. 13](#)). For every operation, several configurations are considered (combinational, sequential, pipelined, bit serial or parallel), and several generic models are available, thus, constituting a library of virtual components.

The development of Systems on Chip (SoC) is the topic of [Chaps. 14–16](#). The main concepts are presented in [Chap. 14](#): embedded processors, memories, buses, IP components, prototyping boards, and so on. [Chapter 15](#) presents two case studies, both based on commercial EDA tools and prototyping boards. [Chapter 16](#) is an introduction to dynamic reconfiguration, a technique that allows reducing the area by modifying the device configuration at run time.

Guide to FPGA Implementation of Arithmetic Functions

Deschamps, J.-P.; Sutter, G.D.; Cantó, E.

2012, XVI, 472 p., Hardcover

ISBN: 978-94-007-2986-5