

Preface

This book presents fuzzy logic and LabVIEW FPGA for designing fuzzy logic controllers. This is a book for implementing fuzzy logic controllers in LabVIEW FPGAs.

Despite the FPGA's attractive features, their adoption by industrial control and signal processing engineers has been slower than processors and DSPs. This is due to several factors. First, these engineers traditionally programmed processors and DSPs using higher level languages, such as C. However, FPGAs possessed complex development tool chains that required designs to be specified using hardware description level (HDL) and register transfer level (RTL) semantics. Furthermore, traditional FPGA development tools lacked intellectual property (IP) blocks for common industrial applications, such as ADC and encoder interface logic, PWM and commutation logic, timing and triggering functions, PID control algorithms, memory management, and data transfer functions. In addition, FPGAs natively supported integer data types only which significantly increased development complexity for analog control and signal processing applications that required math, control, and digital signal processing algorithms, as opposed to floating point processors. Also, traditional FPGA simulation tools were operated at the digital design level and were not interoperable with the type of dynamic simulation tools used by control systems and signal processing engineers for modeling continuous time dynamic system response. Moreover, FPGAs compilation times were relatively long, as compared to processors and DSPs. For example, typical FPGA compilation times today range from 15 to 90 min, whereas processor and DSP compilations are typically completed in less than one minute. Finally, the sequential text-based semantics of traditional register level development tools made it relatively difficult to specify timing and concurrency among parallel processing tasks in a way that leverages the inherent parallel processing capability of FPGA devices.

Despite these traditional development challenges, the successful adoption of FPGAs in application areas such as consumer electronics, and the resulting drop in the price of FPGAs has spurred the interest of industrial control design and simulation vendors. Such vendors are creating the next generation FPGA development

tools that are designed for engineers with little or no digital design expertise. The goal of these next generation “system-level” graphical design tools is to empower control, simulation, and signal processing engineers to harness the full power of the FPGA technology. Graphical system design tools are intended to provide a more intuitive, high level programming paradigm that simplifies the creation of complex parallel processing and control applications. Also, they are intended to provide relatively competitive performance and resource usage, as compared to traditional HDL development tools.

Graphical data flow programming languages are a natural fit for FPGA development due to their inherent sense of parallelism and concurrency that naturally maps to hardware design. Also, recent technological advances are enabling designers to place their FPGA code within a high-level dynamic simulation environment. This ability to cross the boundary between the digital domain of the FPGA and the analog multi-physics domain of the system is facilitating a “true” mechatronics approach to development, in which the complex interplay between FPGA silicon logic, power electronic components, electric motor drives, and mechanical systems can all be simulated in a virtual environment without the need to wait for long FPGA compilations.

The ability to quickly iterate and optimize the FPGA logic design in a mechatronics simulation environment, combined with the new high-level programming tools for FPGAs is reducing dramatically the barriers that prohibited wide adoption of FPGAs in industrial control.

In addition to the improved design and simulation tools for FPGAs, the next generation tools are providing a rapidly growing library of IP blocks for common control and DSP algorithms through code sharing services. On the other hand, the number of books that present Fuzzy logic Control is big as Fuzzy logic control is one of the most important control techniques. However, several books are only mathematical descriptions and are not focused on implementation of fuzzy logic control. Moreover, there are not enough books that deal with implementing fuzzy logic controllers in FPGAs. There is still a lot of vagueness and misunderstanding around the implementation of fuzzy logic controllers implemented in LabVIEW™ FPGA.

Since this book presents a clear description of fuzzy logic control type 1 and 2 that are the most used fuzzy logic representations, the implementation in LabVIEW™ FPGA can be developed. Several experimental examples are presented in order to show the potential of Fuzzy Logic controllers implemented in FPGA.

Finally, a complete LabVIEW™ FPGA toolkit for fuzzy logic type 1 and type 2 is included in the book. This toolkit is based on fix point representation that LabVIEW™ FPGA needs. This toolkit is developed for working on LabVIEW™ real-time systems.

Pedro Ponce-Cruz
Arturo Molina
Brian MacCleery

Fuzzy Logic Type 1 and Type 2 Based on LabVIEW™
FPGA

Ponce-Cruz, P.; Molina, A.; MacCleery, B.

2016, XVI, 233 p. 203 illus., 52 illus. in color. With online
files/update., Hardcover

ISBN: 978-3-319-26655-8