HARDWARE-SOFTWARE CODESIGN OF FUZZY CONTROL SYSTEMS USING FPGAS

E. del Toro
Centro de Investigaciones de Microelectrónica (CIME), CUJAE, Ciudad de La Habana, Cuba
ernesto@imse-cnm.csic.es

S. Sánchez-Solano
Instituto de Microelectrónica de Sevilla (CNM-CSIC). Sevilla, España
santiago@imse-cnm.csic.es

M. Brox
Dpto. Arquitectura de Computadores, Electrónica y Tecnología Electrónica. Univ. Córdoba, España
mbrox@uco.es

A. J. Cabrera
Dpto. Automática y Computación. CUJAE. Ciudad de La Habana, Cuba
alex@electrica.cujae.edu.cu

Keywords: Fuzzy control, hardware/software codesign, FPGA.

Abstract: This paper describes a hardware/software codesign strategy for fuzzy control systems implementation using FPGAs. The main contribution of the paper consists of a methodology for joint development of hardware and software components intended for rapid and verifiable design of a fuzzy control system. The design flow combines specific tools for fuzzy inference systems included in the *XFuzzy* environment, simulation and modelling tools from Matlab and FPGA synthesis, and implementation tools provided by Xilinx. The advantages of this proposal are described in section 4 as it is used for the control system development of an autonomous vehicle.

1 INTRODUCTION

Fuzzy logic provides a mathematical framework to deal with the uncertainty and the imprecision typical of the human reasoning system. One of its main characteristics is the capability to describe the behaviour of a complex system in a linguistic way (Zadeh, 1973). Unlike classical logic systems, fuzzy logic aims to model approximated reasoning modes that play a significant role in the human ability to make rational decision without using precise mathematical models. These advantages have led to an increase in the number of applications using fuzzy logic controller (Ross, 2004).

A great number of different design proposals, which range from software implementation to complete hardware design, have been reported in the last year (Baturone et al., 2000). The level of complexity attained by many of the current applications of control systems requires designing the fuzzy inference modules (FIM) as components to be included in a bigger system that, not only will be able to apply the control responses, but also to interface to other systems, reconfigure itself to different states, and perform other tasks not related to the fuzzy inference process. In these systems, common tasks and configuration may be realized by the software part using a general purpose processor, while time consuming functions must be implemented by means of specialized hardware (Cabrera et al., 2004).

The progress in integrated circuits manufacturing technologies allows the integration of complex control systems on a single chip. Also, the resources

available in current FPGA families can be used to implement a system on a programmable chip (SoPC). However, to benefit from these technological advances, new design methodologies and powerful CAD tools must be developed to cut down the development cycle of new products and make them more competitive in market terms.

A fuzzy control system design methodology is described in this paper. The codesign strategy and the basic components of the control systems are introduced in Section 2. In Section 3, the design flow and the tools are described. An application of the proposed methodology is explained in Section 4. Finally, the main contributions and future goals are resumed in Section 5.

2 CONTROL SYSTEMS CODESIGN MODULES

The proposed HW/SW codesign methodology for development of a fuzzy control system as a SoPC combines the use of a general purpose processor, available as IP-module for FPGA, connected to specific fuzzy IP-modules that allow fuzzy inference acceleration.

The processor used is MicroBlaze, which is a 32-bit RISC processor soft core optimized for implementation in Xilinx FPGAs. The system architecture of MicroBlaze consists of several buses that allows using multiple interfaces to connect peripherals.

The main characteristics of the fuzzy module used in this design are the efficient use of resources, low power and high speed. In order to accomplish these characteristics it is important to remark the use of simplified defuzzification methods, the limited overlap degree of input membership functions and the implementation of a processing strategy that evaluates only the active rules (Sánchez-Solano et al., 2007).

3 DESIGN FLOW TOOLS

The proposed design flow combines the use of specific tools for development of fuzzy systems from the *XFuzzy* environment, modelling and simulation using Matlab, and Xilinx EDK for synthesis and implementation in FPGAs. According to the proposed methodology, the development of a fuzzy control system will be implemented at different stages which are described in next sections.

3.1 FIMs Design using Xfuzzy

The *Xfuzzy* environment has been developed to ease the design of fuzzy systems by starting from linguistic and/or numerical knowledge to final implementing them as hardware and/or software components. It provides a wide set of new featured tools which offer Graphical User Interfaces to ease the design flow at the stages of description, verification and synthesis. It can be also used for extracting fuzzy rules from numerical data and includes tuning and simplification facilities (López et al., 1998).

The first stage of the aforementioned methodology aims at functional description and verification of the fuzzy inference modules. The FIMs may be described in *Xfuzzy* using a hierarchical architecture that combines fuzzy modules (for implementation of fuzzy rules bases) and crisp modules (to perform arithmetic and logic functions).

Knowledge bases may be generated directly via *xfedit* or using identification and supervised learning tools, like *xfdm* and *xfsl*, with training data. *xfplot* may be used for functional verification. In addition, a closed loop simulation may be done with *xfsim*, using the fuzzy module in connection with a high-level model of the plant.

Once concluded the specification stage, a tool named *xfsg* is used to perform hardware synthesis. This tool generates the required files for the next stage.

3.2 Synthesis and verification using SysGen

Using the SysGen library (Xilinx, 2008b), *XFuzzyLib* is generated as a specific library that provides basic modules for implementation of fuzzy controllers. *XFuzzyLib* library contains basic building blocks and other module descriptions including different connectives and defuzzification methods. See Figure 1 for description of an inference module.

Automatic translation between the fuzzy inference description and the Simulink module is made using the files generated by the above mentioned *xfsg*. These files are a Simulink module describing the fuzzy system and a Matlab file that contains the definition of size and functionality of FIM components.

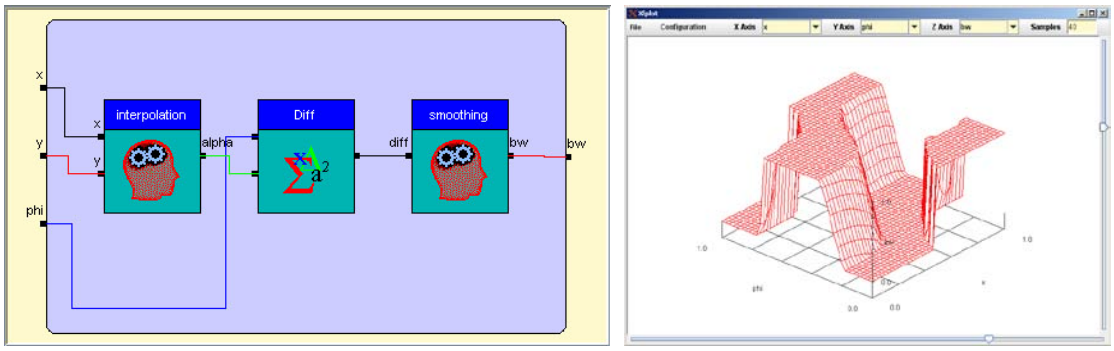

Figure 1: Description of a fuzzy inference module in *Xfuzzy* (left) and associated control surface (right).

Design correctness can be verified at this stage by means of the facilities provided by Matlab. The use of a System Generator Block allows hardware synthesis. Also, it is possible to perform functional verification in closed loop through hardware-software cosimulation as shown in Figure 2.

3.3 IP module construction and integration with MicroBlaze

SysGen has options to connect hardware design with MicroBlaze implementation in a smooth way. Basically this consists in defining *input* and *output* registers so they can be addressed by MicroBlaze in various forms, see (Xilinx, 2008a).

The import process in EDK adds interface (glue) logic according to the selected BUS for connection as well as basic drivers for software communication.

3.4 Control system implementation

MicroBlaze hardware synthesis connected with the fuzzy controller and with other IP modules is possible thanks to Xilinx XPS tool. According to the design needs and constraints, the designer follows basic steps in order to correctly implement the whole system. Numerous options can be used, including that where the system (MicroBlaze processor and peripherals including the fuzzy controller) can be taken again to Simulink in order to verify correctness of implementation and perform a much more complex simulation.

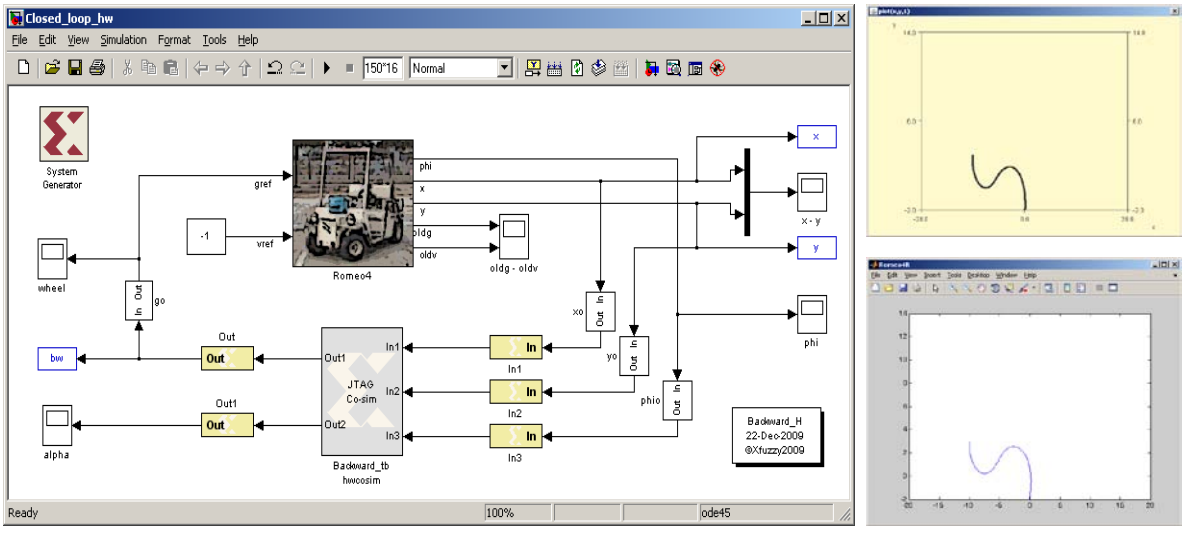

Figure 2: Left: Hardware/software cosimulation using the FPGA implementation of the fuzzy controller in a closed loop with a plant model. Top right: *Xfuzzy* simulation results of parking maneuvers, Bottom right: Simulink simulation.

4 CONTROL SYSTEM DEVELOPMENT OF AN AUTONOMOUS VEHICLE

Parking of autonomous vehicles in a constrained space is a typical control problem in robotics (Li et al., 2003). Starting from any given position and orientation (x, y, Φ), the autonomous mobile robot must drive forward and backward (as required) at speed v and with a wheel curvature γ in order to always arrive backward at target position (0, 0, 0).

The above methodology has been applied to the realization of a fuzzy control system for autonomous parking of an electric vehicle. The used mobile robot has been an autonomous electric vehicle called Romeo-4R. Romeo-4R is a four-wheeled car with standard Ackerman steering, DC traction and steering electrical motors. A digital signal processor (DSP) TMS-320LF provides support for motor control (encoder inputs and PWM outputs), A/D conversion, and communication links through serial ports, thus easing the low level control of the vehicle. The DSP acquires information from sensors (a gyroscope and encoders) and processes it by using a kinematical model usually employed for car-like robots in order to resolve the actual position (x, y) and orientation (Φ) (Cuesta et al., 2004).

The state of the vehicle is transmitted every 50 ms, thus determining the duration of the control cycle. The fuzzy high-level controller performs the parking control strategy and sends back to the DSP the new required values of speed and wheel curvature, so that the DSP controls the traction and direction motors. This hierarchical control structure allows developing different control strategies in the high-level controller and frees it from the low-level control task of Romeo-4R.

Once known the values of the current state (x, y, Φ, v, γ) of Romeo-4R, the DSP transmits them to the FPGA containing the fuzzy controller through a RS-232 serial interface and using a specific communication protocol (which is also implemented by the program running in the MicroBlaze processor).

In order to give physical support to the development platform, a Xilinx University Program Virtex2-Pro Development System Board has been employed. This board allows cosimulation to be carried out using Matlab.

Figure 2 (right) shows simulation results illustrating the trajectories of parking maneuvers.

5. CONCLUSIONS

A realization strategy for the development of hybrid HW/SW embedded fuzzy controllers on FPGA devices has been described. The design flow combines specific tools for development, simulation, synthesis and implementation using FPGAs. The main contribution of this paper is a methodology for the joint construction of hardware and software components in every stage of design. The proposed methodology is applied to solve a classic robotic problem.

ACKNOWLEDGEMENTS

Project TEC2008-04920 financed by "Ministerio de Ciencia e Innovación" and P08-TIC-03674 by "Junta de Andalucía". E. del Toro is a MAEC-AECID scholarship PhD. student.

REFERENCES

Baturone, I., Barriga, A., Sánchez-Solano, S., Jiménez, C.J., and López, D., 2000. *Microelectronic Design of Fuzzy Logic-Based Systems*. CRC Press.

Cabrera, A., Sánchez-Solano, S., Brox, P., Barriga, A., and Senhadji, R.. 2004. *Hardware/software codesign of configurable fuzzy control systems*. Applied Soft Computing, 4, 271-285.

Cuesta, F., Gómez-Bravo, F., and Ollero, A., 2004. *Parking maneuvers of industrial-like electrical vehicles with and without trailer*, IEEE Trans. on Industrial Electronics, 51, 2, 257-269.

Li, T.H.S., Shih-Jie, C., and Yi-Xiang, C. 2003. *Implementation of human-like driving skills by autonomous fuzzy behavior control on an FPGA-based car-like mobile robot*. IEEE Trans. Ind. Electron., 50, 867– 880.

López, D., Jiménez, C.J., Baturone, I., Barriga, A., and Sánchez-Solano, S. 1998. Xfuzzy: *A Design Environment for Fuzzy Systems*. IEEE International Conference on Fuzzy Systems, Anchorage.

Ross,, T. J. 2004. *Fuzzy Logic with Engineering Applications*, Wiley.

Sánchez-Solano, S., Cabrera, A., Baturone, I., Moreno-Velo, F.J., and Brox, M., 2007. FPGA *Implementation of Embedded Fuzzy Controllers for Robotic Applications*. IEEE Trans. on Industrial Electronics, 54, 1937-1945.

Xilinx. 2008a. *MicroBlaze Reference Guides*.

Xilinx. 2008b. *Xilinx System Generator v10.1 for Simulink. User Guide*.

Zadeh, L.A., 1973. 1973. *Outline of a new approach to the analysis of complex systems and decision processes*. IEEE Trans. on Systems, Man, and Cybernetics, 3, 28-44.