DEVELOPMENT OF IP MODULES OF FUZZY CONTROLLERS FOR THE DESIGN OF EMBEDDED SYSTEMS ON FPGAS

María Brox, S. Sánchez-Solano

Instituto de Microelectrónica de Sevilla, Centro Nacional de Microelectrónica, CSIC
Avda. Reina Mercedes s/n, 41012-Sevilla, Spain. maria@imse.cnm.es

ABSTRACT

The rapid evolution of integrated circuit technologies and the great complexity of current systems have led to an increasing difference between the resources provided by those technologies and the productivity obtained by designers. In order to avoid this “productivity gap” new design strategies based on the reuse of previously designed basic blocks, named intellectual properties (IP) modules, have been proposed [1][2]. These modules are easily adapted to different functionalities and to different integration technologies. The use of IP modules together with powerful CAD tools, which facilitate the development of controllers, has major economic advantages, because it reduces the time involved in the development cycle of new products and accelerates their introduction in the market.

In parallel, programmable logic devices have undergone considerable transformation in recent years. This has not only consisted of the increase in the number of general-purpose resources, but also of the introduction of specific elements (memory blocks, multipliers, clock signal generators, etc.), which allow very complex systems to be implemented on a FPGA. This has been reinforced by the existence of numerous system blocks available as IP modules or soft-cores (processors, I/O peripherals, memory controllers, etc) which facilitate the development of embedded processing systems adapted to a particular purpose. Examples include the MicroBlaze system from Xilinx and the Nios system from Altera.

This paper describes the design of fuzzy logic-based controllers as IP modules compatible with the processing systems available in the current families from Xilinx. These modules can be used as standard peripherals connectable to the OPB bus for the processors available in FPGAs, such as MicroBlaze or Power PC. The use of inference techniques based on fuzzy logic allows the development of complex control systems from the linguistic description of the knowledge of an expert operator, without any need to employ mathematical models, and provides good robustness in the face of changing operational conditions.

For the design of the processing system, a MicroBlaze processor has been used [3]. This processor is available as an IP module together with a wide variety of peripherals,

controllers and bus structures which allow the system to be configured according to the requirements of a particular application. MicroBlaze is a 32-bit RISC processor based on Harvard architecture, with separate buses for instructions and data and the possibility of using independent cache memories for both buses. This architecture is optimized for its implementation on FPGAs from Xilinx. Its instruction set includes 32-bit instructions with three operational elements and two direction modes. In Spartan 3, Virtex II and Virtex 4 families, the multiplication operations can be performed by hardware through the multipliers available in FPGAs. MicroBlaze uses a specific bus (LMB) in order to access the block memories available in the FPGA, whereas it employs IBM's standard OPB bus to connect external memory and peripherals. Finally, it has different FIFO canals for the connection of user functions.

The *Xfuzzy* design environment facilitates the different stages of development of a fuzzy system. This includes description, simulation, and simplification tools which allow the fuzzy system to be defined, verified, and optimized. The environment also includes synthesis tools which provide hardware or software implementations of the inference system [4]. The hardware implementation of fuzzy inference modules has been carried out using the synthesis tool of the *Xfuzzy* environment named *Xfvhdl* [5]. This tool provides a VHDL description of the system according to an architecture specially optimized to run fuzzy systems through dedicated hardware [6]. Important characteristics of this architecture, which increase the efficiency of its digital realization, are: the use of an active rule-driven inference process, the limitation of the overlapping degree of membership function inputs and the use of simplified defuzzification methods. When the designer runs the synthesis process, he/she can choose different architectural and implementation options according to the particular characteristics of the problem. Concerning architecture, the designer can select the type of membership function generator (using an arithmetical technique or one based on memory), the operator used as connective (minimum or product), and the defuzzification method employed. Regarding the implementation on the FPGA, the designer can select the particular device that is going to be used and can choose the implementation of the different components of the knowledge bases as

combinational logic or by storage in ROM or RAM memories. In this latter case it is possible to choose between using block type RAM or that associated to the FPGA CLBs [7]. As result, a VHDL description of the system is obtained, that can be synthesized with the tools from the Xilinx ISE environment.

The EDK (*Embedded Development Kit*) development environment includes a group of IP components and tools that facilitate the development of embedded processing systems on FPGAs from Xilinx. The XPS (*Xilinx Platform Studio*) graphical user interface provides a group of templates that facilitate the development of peripherals which connect to the OPB bus. These templates are VHDL codes with two components: IPIF (*Intellectual- property interconnect*), which performs the interface functions with the OPB bus; and User_logic which includes the logic developed by the user. In our case, this last file will include the VHDL description of the fuzzy controller given by *Xfuzzy* plus the code necessary to access the controller through the registers of the IP module. Both components communicate through the IPIC (*Intellectual property interconnect*) interface, which is independent of the peripheral bus. Different types of templates exist depending on the operational mode of the peripheral (master/slave) and the services provided by the IPIF block. The XPS graphical user interface also facilitates the generation of “.mdp” (microprocessor peripheral definition) and “.pao” (peripheral analyze order) files. These files are necessary for the use of the IP module in XPS as other peripherals of the system.

The design cycle of a fuzzy controller as an IP module includes the following steps. Firstly, through the XPS assistant for creating peripherals, the name and peripheral version are indicated, together with the place where it is stored. The following step is to select which bus the peripheral will be attached to. Thirdly, different services implemented by the IPIF block are configured (user logic master/slave support, user logic interrupt support, number of registers, direct memory access, etc.). The following step is the implementation of the user logic. Finally, the last step is to import the existing peripheral. The XPS assistant for importing peripherals requires the inclusion of VHDL templates and library elements, the assignment of ports and the definition of parameters. When these steps have been completed the IP module can be used like any other peripheral in EDK.

The proposed development technique is being employed in the design of a control system applied to the problem of parking autonomous vehicles. In this case the vehicle can be placed in a position close to the desired place of parking, in such a way that the trajectory to be taken must combine forward and backward movements. The proposed system uses a hierarchical knowledge base that includes modules for taking decisions (to decide if the vehicle must drive in one direction or another) and control

modules (to calculate the speed and the steering wheel angle). The vehicle employed has sensors (encoders, gyroscope, etc.) and actuators (traction and direction engines) which enable the vehicle's speed, curvature, position and orientation values to be calculated.

The global system has been implemented on a FPGA according to the hardware/software implementation strategy proposed in [8]. A general purpose processing system based on MicroBlaze deals with the communication with the robot actuators, the information acquisition from the sensors, and the processing of the speed, curvature, position and orientation of the vehicle. The inference mechanisms which perform the control heuristics are implemented by dedicated hardware according to the architecture previously mentioned. The experimental results shown in Figure 1 demonstrate that the proposed development technique is valid.

Fig. 1. Experimental results

REFERENCES

- [1] W. Savage, J. Chilton y R. Camposano, "IP reuse in the System on a Chip Era", *13th International Symposium on System Synthesis*, pp. 2-7, 2000
- [2] M. Keating. The reuse methodology manual, "Toward a reuse discipline. Design, Automation and Test in Europe", DATE'98, pp. 141-145, 1998
- [3] *MicroBlaze Reference Guide*, Xilinx Inc., 2002
- [4] F. J. Moreno-Velo, I. Baturone, S. Sánchez-Solano, A. Barriga, "Rapid Design of Complex Fuzzy Systems with Xfuzzy", *Proc. IEEE Int. Conf. on Fuzzy systems*, pp. 342-347, St. Louis, May, 2003
- [5] *Xfuzzy: Fuzzy Logic Design Tools* (<http://www.imse.cnm.es/Xfuzzy/>)
- [6] S. Sánchez-Solano, A. Barriga, C. J. Jiménez, J. L. Huertas, "Design and Applications of Digital Fuzzy Controllers", *IEEE Int. Conf. on Fuzzy Systems*, pp. 869-874, 1997
- [7] E. Lago, C. J. Jiménez, D. R. López, S. Sánchez-Solano, A. Barriga, "Xfvhdl: A tool for the Synthesis of Fuzzy Logic Controllers", *DATE'98*, pp. 102-107, Paris, Feb. 1998
- [8] A. Cabrera, S. Sánchez-Solano, P. Brox, A. Barriga, R. Senhadji, "Hardware/software codesign of configurable fuzzy control systems. *Applied Soft Computing*". Vol. 4, N°3, pp. 271-285, 2004.