PROTOTYPING OF FUZZY LOGIC–BASED CONTROLLERS USING STANDARD FPGA DEVELOPMENT BOARDS

S. Sánchez-Solano[1], R. Senhadji[1], A. Cabrera[2], I. Baturone[1], C. J. Jiménez[1], A. Barriga[1]

[1] Instituto de Microelectrónica de Sevilla, Centro Nacional de Microelectrónica,
Avda. Reina Mercedes s/n, E-41012-Sevilla, Spain.

[2] Dpto. Automática y Computación, Facultad de Ingeniería Eléctrica,
ISPJAE, Ciudad de la Habana, Cuba.

Proc. 13th IEEE International Workshop on Rapid System Prototyping (RSP'2002), pp. 25-32, Darmstadt, July 1-3, 2002.

Prototyping of Fuzzy Logic–Based Controllers Using Standard FPGA Development Boards

S. Sánchez-Solano[1], R. Senhadji[1], A. Cabrera[2], I. Baturone[1], C. J. Jiménez[1], A. Barriga[1]

[1] Instituto de Microelectrónica de Sevilla, Centro Nacional de Microelectrónica, Avda. Reina Mercedes s/n, 41012-Sevilla, Spain. <santiago@imse.cnm.es>

[2] Dpto. Automática y Computación, Facultad de Ingeniería Eléctrica, ISPJAE, Ciudad de la Habana, Cuba. <alex@electrica.ispjae.edu.cu>

Abstract

This paper describes a design methodology for fuzzy logic-based control systems. The methodology employs hardware/software codesign techniques according to an 'a priori' partition of the tasks assigned to the selected components. This feature makes it possible to tackle the control system prototyping as one of the design stages. In our case, the platform considered for prototyping has been a development board containing a standard microcontroller and an FPGA. Experimental results from an actual control application validate the efficiency of this methodology.

1 Introduction

The growing complexity of current electronic systems together with the need of decreasing the time-to-market when launching a new product have motivated very much the use and development of CAD tools that ease the description, verification and synthesis of electronic systems. This demand is becoming stronger as challenging applications aimed at integrated a whole system on a chip (SoC) are increasing. Taking into account the characteristics of integrated circuits fabrication processes such as cost, delivering time, etc., it is very useful the availability of mechanisms which permit the rapid and cheap prototyping of the system under development in order to verify its global functionality before its possible integration. This often means the use of general-purpose platforms.

This paper focuses on the design methodology of fuzzy logic-based control systems. Following the above commented ideas, one of the steps of our design flow will be the prototyping of the whole system on a development board that includes a general-purpose microcontroller and an FPGA, thus resorting to the use of hardware/software codesign techniques. The methodology employs specific tools for the development of fuzzy controllers and generic tools for a twofold task: the synthesis of components from hardware description languages, on one side, and the programming of standard microcontrollers, on the other side.

2 Fuzzy controllers

The capability of fuzzy controllers to describe with simple rules the linguistically expressed experience and knowledge of a human expert and to achieve that without the need of a mathematical model of the plant under control, have motivated a great increase in the number of control applications using fuzzy logic-based inference techniques [1].

A fuzzy controller employs the same input and output variables than its conventional counterpart. For instance, the output variation of a PI controller at each iteration is function of the error and the error variation at the previous iteration:

$$\Delta u(nT) = F(e(nT), \Delta e(nT)) \tag{1}$$

The difference between the conventional and the fuzzy approaches is that in the first one, the output is obtained as a lineal combination of the inputs,

$$\Delta u(nT) = k_I \cdot e(nT) + k_P \cdot \Delta e(nT) \tag{2}$$

while in the fuzzy approach, the control heuristic is defined by a rule set that employ linguistic variables represented by fuzzy sets (Figure 1).

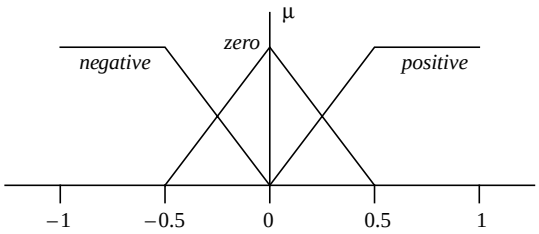
This is the reason why the control surface provided by a fuzzy controller is usually non lineal and can be modified locally by changing the rules that affect the corresponding universes of discourse. In addition, the use of fuzzy logic-based inference techniques allows the development of robust devices able to support greater perturbations than a conventional one and to provide soft output variations with a small number of rules. These features make it possible to overcome the automatization of complex control tasks that currently can be carried out only by human operators. Other reported advantages of fuzzy controllers are that they often reduce the development cost and maintenance of control systems [2].

1. **IF** error *e(k)* is *positive*
   AND error variation Δ*e(k)* is *near zero*
   THEN control variation Δ*u(k)* is *positive*.

2. **IF** error *e(k)* is *negative*
   AND error variation Δ*e(k)* is *near zero*
   THEN control variation Δ*u(k)* is *negative*.

3. **IF** error *e(k)* is *near zero*
   AND error variation Δ*e(k)* is *near zero*
   THEN control variation Δ*u(k)* is *near zero*.

(a)



(b)

Figure 1: Example of rule base (a) and fuzzy sets (b) used by a fuzzy PI controller.

3 Implementation of fuzzy control systems

Figure 2 illustrates the generic structure and tasks performed by a fuzzy logic-based controller. The variables defining the plant status are acquired by a set of sensors that translate them into electrical signals represented by currents or voltages. The signal conditioning input stage implements the A/D conversion and range adjustment together with several preprocessing algorithms required to obtain adequate input signals to the inference engine. This engine is in charge of evaluating the set of inputs (using a fuzzifier), calculating the new control action according to the strategy defined by the set of inference rules, and computing the non-fuzzy representative values of the control action (using a defuzzifier). The signal conditioning output stage postprocesses these values and adequates them to the actuators connected to the plant. In addition to these tasks, every control system usually includes elements to initialize the system, to define the control targets, and to monitor the achievement of the targets, as well as timing elements to ensure the correct behavior of the system.

Except for the A/D and D/A conversions, all the above commented tasks (including the inference mechanism) can be performed by software, so that the control system can be implemented with a programmable processor. Many applications to industrial or consume products resort to this type of implementation [3]. The problem appears whenever the application demands fuzzy control systems with a high inference speed and/or low size and power. Since standard software approaches are not able to meet these requirements, several solutions have been reported. Some of these solutions try to improve the response time of the controller by expanding, via software or hardware, the instruction set of an standard processor. Taking into account that many of the operations required to carry out the fuzzy inference limit to a great extend the operational speed of the fuzzy controller, the idea is to speed up the execution of operations such as the maximum, minimum, or defuzzification [4]. In the software approach, the processor microprogramming is modified to add functionality to the instruction set [5]-[6], while in the hardware approach, new circuitry is included in the processor architecture [7]-[9].

When the speed, size and/or power requirements are restrictive, the solution is to implement the fuzzy controller

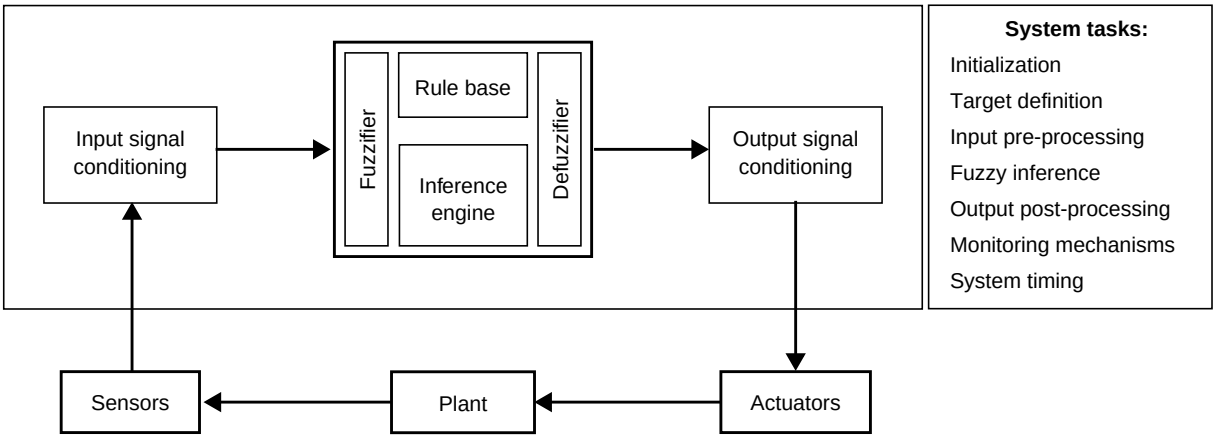


Figure 2: Block diagram and tasks of a fuzzy logic-based control system.

with specific analog or digital hardware that follows an efficient architecture [10]. Some key points to achieve low cost and high speed digital fuzzy controllers are to restrict the shapes of the fuzzy sets membership functions of the inference rules, to use simplified defuzzification methods, and to evaluate only the contribution of the active rules [11].

According to these considerations and thinking in challenging applications with restrictive requirements, the strategy described in this paper combines specific hardware for implementing the fuzzy inference with a general-purpose processor for carrying out the rest of the tasks by software.

4 Design methodology

The design process of an electronic system can be greatly accelerated by two mechanisms. One of them is the use of a design methodology which defines the different design stages and their interrelations. The second one is the use of CAD tools which ease the designer tasks at each stage. In our case, the *Xfuzzy* environment with its CAD tools (*xfc*, *xfsim*, *xflab*, and *xfvhdl*) is very helpful in the design flow of fuzzy logic-based control systems, as shown in Figure 3 [12]. Starting from the formal description of the system, this design flow is aimed at the realization of fuzzy systems on chip employing hardware/software codesign techniques. Between these extreme stages, three other stages are carried out: off-line simulation, on-line verification (whenever possible), and prototyping with low-cost development boards.

4.1 System specification

The hardware/software partitioning of the different tasks to be performed by the control system is done according to the considerations commented in Section 3. This has led us to implement the inference process by hardware while the rest of the tasks (such as pre- and post-processing) are carried out by software. The use of codesign techniques means that the control system specification requires the description of hardware as well as software components. The *XFL* language is employed for the specification of the fuzzy inference component [13]. This is the language shared by all the tools of the *Xfuzzy* environment. The C language is employed for specifying the rest of the tasks since it allows the automatic generation of programming code (there are many C compilers, assemblers and linkers for most of general-purpose processors) and it is very easy for *Xfuzzy* to integrate C code.

The *XFL* language allows the user to define comfortably not only the universes of discourse of the fuzzy controller variables, their associated membership functions, and the rule bases employed, but also the set of fuzzy operators that implement the antecedent connection, the implication function, the mechanism for rule aggregation, and the defuzzification method.

4.2 Off-line simulation

The *Xfuzzy* environment has a tool named *xfc* which permits obtaining a C description of the fuzzy inference engine from its *XFL* specification. The code generated by *xfc* together with the programming code associated to the rest of the control tasks can be combined with a C model of the plant under control to simulate the behavior of the whole closed-loop control system. This is possible within the *Xfuzzy* environment thanks to a tool named *xfsim*. This tool is very useful to allow a preliminary adjustment of the control system parameters. However, since the model used to represent the plant is usually a first-order approximation, the results of this simulation could not be reliable enough for control systems of a certain complexity. In these cases, the actual plant or process has to be analyzed to obtain a finer adjustment of the control system parameters. One solution is to include the actual plant into the control loop where the control system is implemented by a computer running a C program. This is the objective of the following design stage: an on-line verification. It is important to notice that

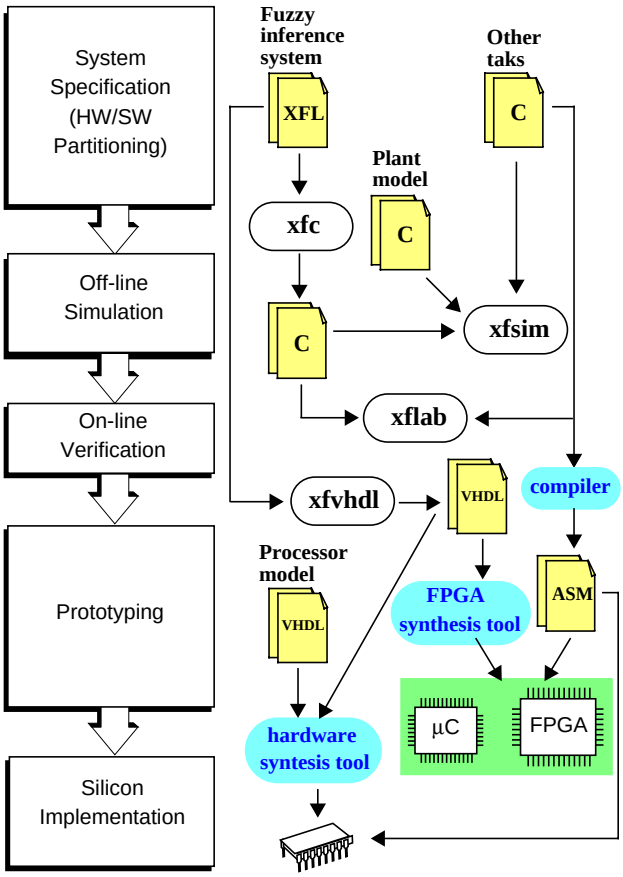


Figure 3: Design flow and CAD tools for fuzzy control systems.

this verification is possible whenever the response time of the computer meets the plant dynamic requirements.

4.3 On-line verification

The tool *xflab* included into the *Xfuzzy* environment provides the needed mechanisms for communicating the computer that runs the software implementation of the fuzzy controller with the plant under control via a data acquisition board connected to the internal bus of the PC [14]. The tool *xflab* contains drivers to read and write data from and in the board, so that the model of the plant used in the off-line verification can be replaced by a function that allows monitoring the true plant, thus obtaining the required information to execute the fuzzy control (input channels), as well as actuating on the plant providing the control action via the output channels.

Xflab eases the configuration of the data acquisition board (definition of the input and output addresses, assignation of the analog and digital channels, etc.), the codification of the required procedures (filtering, linearization, etc.), and the integration of these tasks with both the software implementation of the controller and the routines to access the actual plant. One of the most relevant advantages offered by *xflab* is the capability of monitoring the behavior of the whole system at real time, allowing the evaluation of a great variety of operation conditions as well as the influence of the different control parameters on the system efficiency. Figure 4-a shows one window of the *xflab* GUI where the different components of the control program are specified. The window where the system monitoring is configured is shown in Figure 4-b.

Once the control system has been adjusted conveniently, we have a software implementation of the control system which is quite operative. This will be the starting point for prototyping the system on a development board.

4.4 Prototyping

The availability of development boards that include a general-purpose microcontroller with an FPGA allows the implementation of a whole control system by partitioning the different tasks between them. As commented in Section 3, the fuzzy inference is assigned to an specific controller implemented on the FPGA, while the routines for processing the input and output variables of the fuzzy controller, together with all the other tasks are programmed in the microcontroller. The timing task is implemented by one of the timers available in the microcontroller. Additional circuitry is required to adapt the signals provided by the sensors to the development board as well as to adapt the output signals of the board to those employed by the actuators. This means the inclusion of circuits such as signal amplifiers and data converters.

Automatic synthesis tools are unavoidable if we want to complete this stage rapidly and efficiently. One of the most outstanding feature of *Xfuzzy* compared with other environments is that *Xfuzzy* includes tools that ease the hardware synthesis of the designed fuzzy controller. One of them, which is named *xfvhdl* [15], translates the XFL description of the fuzzy inference system into a VHDL description that is compatible with several hardware synthesis tools for FPGA or ASIC realizations.

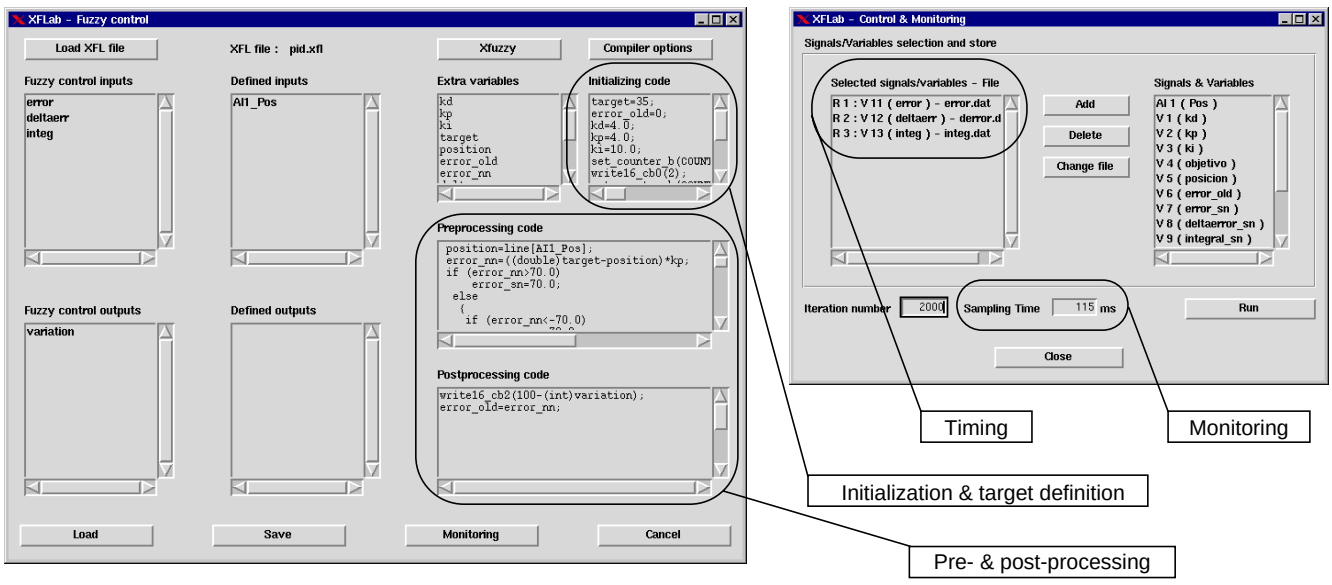


Figure 4: Xflab windows illustrating the task assignment carried out in the on-line verification stage.

On the other side, the use of development tools and program debuggers (assemblers, compilers, simulators, etc.) ease the programming of the different tasks to be performed by the microcontroller. The availability of C compilers for the chosen microcontroller is a factor to take into account since it reduces greatly the development time.

4.5 Silicon implementation

Once checked that the prototype performance is as desired, the designer can increase the integration level of the control system to achieve less area and power consumption. Thinking in an ASIC implementation, the good starting point is the VHDL description of the whole system. In our design flow, this means to combine the VHDL of the fuzzy inference engine with the VHDL description of the selected microcontroller core.

5 Application example

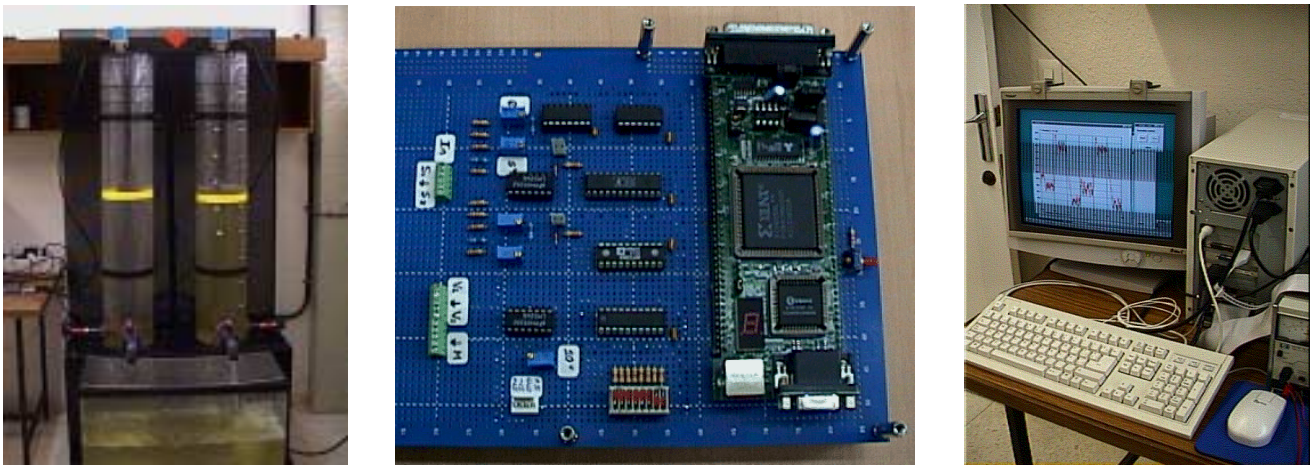
The above described methodology has been applied in the design of a level controller for a dosage system. As shown in Figure 5a, the dosage system is composed of two independent cylindrical tanks both with an electronic valve at its top to control the liquid injection, with a pressure sensor at its bottom to provide a measure of the liquid level, and with a manually-controlled valve also at its bottom to discharge liquid into a shared container [16]. A voltage-controlled water pump moves the liquid up to the tanks from the shared container.

The pressure sensors provide a current output signal between 4 and 20 mA, whereas the electronic valves and the water pump are controlled by voltage input signals between 0 and 10 V. The control strategy applied is the fuzzy version of a PI controller with an incremental output. This means that the fuzzy controller requires two input signals (the error and its variation) for each tank, and provides an output signal that represents the change of its corresponding valve aperture. The water pump is governed by the bounded sum of both controller outputs.

Due to the symmetry of the two tanks, their fuzzy controllers are identical, so that only one of them can be implemented if the inference processes are executed sequentially. This solution permits a great area reduction without increasing the response time significantly.

The control system prototyping was carried by using an XS40-005XL board from XESS Corporation [17] mounted on a board that also includes the A/D and D/A data converters together with the signal conditioning circuitry (Figure 5b). The XS40-005XL board has an Intel 8031 microcontroller, a Xilinx XC-4005 FPGA, 32 K of SRAM, a programmable clock circuit, a 7-segment LED display, and connectors for communicating the board with a PC via a parallel port, an VGA output, and a PS/2 input. The board is provided with software facilities to program the clock (GXSSETCLK), to download the configuration of the FPGA and the program of the microcontroller (GXSLOAD), and to test the board (GXSTEST).

The synthesis stage of the FPGA was carried out with the "Xilinx Foundation 3.1i" environment. On the other side, the VHDL description of the fuzzy controller was obtained from its XFL specification with the aid of the *xfvhdl* tool of *Xfuzzy*. Since the fuzzy inference engine implemented in the FPGA works as a coprocessor of the 8031 microcontroller, several interface circuits has to be added to the FPGA. As a result, the 98% of the FPGA CLBs are employed (193 de



(a) **(b)** **(c)**

Figure 5: Experimental set up: a) Dosage system. b) Control system implementation. c) Development system.

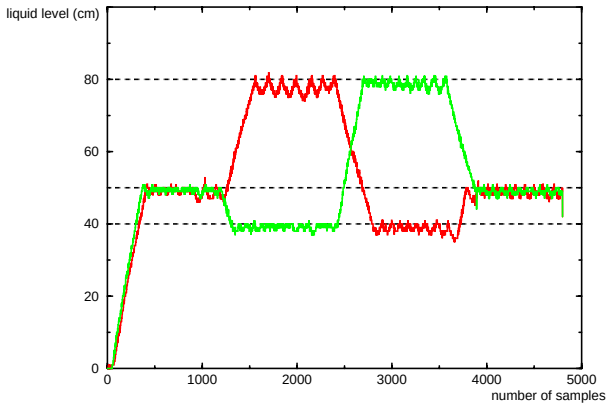


Figure 6: Experimental results.

196), and the maximum operational frequency reaches 20 MHz. At a 10 MHz frequency, the fuzzy inference cycle takes 700 ns, which is shorter than the execution time of a microcontroller instruction.

The software implementation in the 8031 was performed by using several development tools available for the MCS-51 family. The code downloaded to the microcontroller covers the typical tasks of a control cycle (acquisition and preprocessing, communication with the inference engine, postprocessing and output generation), and includes several interrupt service routines as well as routines to allow communication with a PC in order to evaluate the system performance (Figure 5c).

The designed prototype has been proved satisfactory in different experimental tests, providing a robust control which ensures a good behavior of the dosage system even under hard changes in the aperture of the manually controlled valves. As an example, Figure 6 illustrates how the liquid levels of the two tanks follow different target positions (dashed lines) with the typical ripple caused by the liquid dropping.

6 Conclusions

The use of codesign techniques according to a given partitioning of the tasks assigned to the hardware and software components has been proved to be an efficient strategy for designing high-speed and low-consumption fuzzy logic based control systems. According to this strategy, a design methodology has been described that employs several tools of the *Xfuzzy* environment and that includes, as one of its design stages, the prototyping of the system using a low cost development board with a microcontroller and an FPGA. The validity of the procedure is illustrated by its successful application to solve an actual dosage problem.

References

[1] Passino, K. M., Yurkovich, S., *Fuzzy Control*, Addison-Wesley, 1998.

[2] Terano, T., Asai, K., Sugeno, M., Eds., *Applied Fuzzy Systems*, Academic Press, 1994.

[3] Yen, J., Langari, R., Zadeh, L. A., Eds., *Industrial Applications of Fuzzy Logic and Intelligent Systems*, IEEE Press, 1995.

[4] Salapura, V. *A Fuzzy RISC Processor*, IEEE Transactions on Fuzzy Systems, Vol. 8, N. 6, Dec. 2000.

[5] Ungering A. P, Goser, K., *Architecture of a 64-bit fuzzy inference processor*, FUZZIEEE'94, pp. 1776-1780, Orlando, 1994.

[6] Von Altrock, C. *Adapting existing Hardware for Fuzzy Computation*, Institute of Physics Publishing, 1998.

[7] Patyra, M. J., Braun, E., *Fuzzy/Scalar RISC processor: architectural level design and modeling*, FUZZIEEE'96, pp. 1937-1943, New Orleans, 1996.

[8] Kim Y. D., Lee-Kwang, H., *High speed flexible fuzzy hardware for fuzzy information processing*, IEEE Trans. on Systems, Man, and Cybernetics, Vol. 27. N, 1, pp. 45-55, 1997.

[9] Pagni, A., *Digital approaches*, in *Handbook of Fuzzy Computation*, Institute of Physics Publishing, 1998.

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

[11] Sánchez-Solano, S., Barriga, A., Jiménez, C.J., Huertas, J.L, *Design and Applications of Digital Fuzzy Controllers*, FUZZIEEE'97, pp. 869-874, Barcelona, Jul. 1997.

[12] López, D.R., Jiménez, C.J., Baturone, I., Barriga, A. Sánchez-Solano, S., *Xfuzzy: A Design Environment for Fuzzy Systems*, FUZZIEEE'98, 1060-1065, Anchorage, May 1998.

[13] López, D.R., Moreno, F.J., Barriga, A. Sánchez-Solano, S., *XFL: A Language for the Definition of Fuzzy Systems*, FUZZ-IEEE'97, pp. 1581-1591, Barcelona, Jul. 1997.

[14] Senhadji, R., Sánchez-Solano, S., López, D.R. Barriga, A., *Xflab: An On-line Verification Tool for Fuzzy Controllers*, IPMU'2000, pp.. 44-49, Madrid, Jul. 2000.

[15] Lago, E., Jiménez C.J., López D.R., Sánchez-Solano S., Barriga A., *Xfvhdl: A Tool for the Synthesis of Fuzzy Logic Controllers*, DATE'98, pp. 102-107, Paris, Feb. 1998.

[16] Cabrera, A., Senhadji, R., Sánchez-Solano, S., Barriga, A., Jiménez, C.J., Llanes, O., *Development of Level Controllers Based on Fuzzy Logic*, NF'2002, Ciudad de la Habana, Jan. 2002.

[17] XS40, *XSP Board V1.4 User Manual*, XESS Corporation, USA, 1999.