Design of FPGA based PID-like Fuzzy Controller for Industrial Applications

Dr. Mohammed Y. Hassan, Member, IAENG Waleed F. Sharif, M. Sc.
Control and Systems Engineering Department
University of Technology
Baghdad, Iraq

Abstract**—This paper proposes the design of PID-like fuzzy logic controller (PIDFLC), on Field Programmable Gate Array (FPGA) device. The Fuzzy Inference System (FIS) used in the controller is aided with *Active Rules Selection Mechanism*. Developments were made to this FIS to make it able to manipulate signed numbers, (which is important issue in control system), then, it was blended with integral and derivative control components of tunable gains. These new features enable the controller to function as a PDFLC, a PIFLC, and a PIDFLC efficiently. The design utilizes 1394 slices of the target FPGA, and is able to produce an output at 0.421 µsec with maximum frequency of 40.295 MHz. Mathematical model of linear plants were used to test the controller. The simulation results using the proposed controller connected to these plants in unity feedback system were compared with simulation results of a similar system that uses a software-based controller. The plant responses controlled by the proposed controller were smooth and much similar to the plant responses when using software based controller.**

Index Terms—Industrial application , FPGA, Fuzzy logic, PID-like fuzzy controller.

I. INTRODUCTION

Fuzzy logic has rapidly become one of the most successful of today's technologies for developing sophisticated control systems. Fuzzy controllers are more robust than PID controllers because they can cover a much wider range of operating conditions than PID can, and can operate with noise and disturbances of different nature. Given the dominance of conventional PID control in industrial applications, it is significant both in theory and in practice if a controller can be found that is capable of outperforming the PID controller with comparable ease of use. Some of PID fuzzy controllers are quite close to this dream [1]. The simplest and most usual way to implement a fuzzy controller is to realize it as a computer program on a general purpose computer. However, a large number of fuzzy control applications require a real-time operation to interface high-speed constraints. Software implementation of fuzzy logic on general purpose computers can not be considered as a suitable design solution for this type of application, in such cases, design specifications can be matched by specialized fuzzy processors.

Higher density programmable logic devices such as FPGAs can be used to integrate large amounts of logic in a single IC. Semi-custom and full-custom application specific integrated circuit (ASIC) devices are also used for this purpose but FPGAs provide additional flexibility: they can be used with tighter time-to-market schedules. The Field-Programmable Gate Array (FPGA) places fixed logic cells on the wafer, and the FPGA designer constructs more complex functions from these cells. The term *field programmable* highlights the customizing of the IC by the user, rather than by the foundry manufacturing the FPGA.

Several researchers discussed the design of hardware fuzzy logic controller. Number of these works were specialized in control application [2,]-[3], and were aim to get better control responses. Others were concerned in developing general fuzzy logic processors [4]-[5]-[6]-[7]. Their searches were concern using new techniques in fuzzy algorithm, to get higher processing speed versus low utilization of chip resource. As a result, the proposed design in this paper is aim to employ the new techniques of fuzzy algorithm in controlling industrial application with the aid of conventional PID control to serve these applications efficiently.

II. THE PROPOSED PID-LIKE CONTROLLER

The general layout of the controller chip in a unity feedback control system is shown in Fig. 1. Generally, the proposed controller accept the output of the plant (y_p) and the desired output (y_d), both as digital signals, and deliver digital control action signal as an output. The design accepts also four 8-bit digital signals that represent the gain coefficients needed by the controller (proportional gain K_p, derivative gain K_d, integral gain K_i, and output gain K_o), and two one-bit signals to select the type of the controller (PDFLC, PIFLC, or PIDFLC).

III. STRUCTURE OF THE PROPOSED PIDFLC

In order to build a PIDFLC, it is required to design a fuzzy inference system with three inputs that represent the proportional, derivative, and integral components.

Manuscript received August 19, 2007. Paper title: Design of FPGA based PID-like Fuzzy Controller for Industrial Applications.

M. Y. Hassan is a lecturer with the Control and Systems Engineering Department, University of Technology, Iraq. (Corresponding author to provide e-mail: myhazawy@ yahoo.com.).

W. F. Sharif is an assistant lecturer with the Control and Systems Engineering Department, University of Technology, Iraq. (Corresponding author to provide E-mail: waleedfawwaz@ yahoo.com).

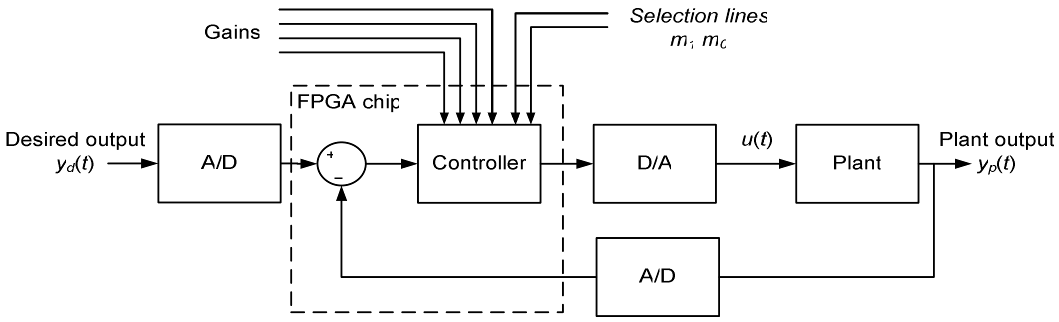


Fig. 1: Layout of the proposed controller in a unity feedback control system

A fuzzy controller with three inputs may not be preferred, because it needs large number of rules, instead, the PID fuzzy controller can be constructed as a parallel structure of a PD fuzzy controller and a PI fuzzy controller and the output of the PIDFLC is formed by algebraically adding the outputs of the two fuzzy control blocks. However, it is difficult to formulate control rules with the input variable sum of error ($\sum e$), as its steady-state value is unknown for most control problems. To overcome this problem, a PD controller may be employed to serve as PI controller in incremental form. Equation (1) shows a PD controller obtained in position form, while (2) shows a PI controller in incremental form:

$$u(n)=K_p e(\mathrm{n})+K_d r(n) \tag{1}$$

$$\Delta u(n)=K_p r(n)+K_i e(n) \tag{2}$$

where e(n) is sampled error signal, $r(n)$ rate of change of sampled error signal, and $a(n)$ is accelerated rate of change of sampled error signal.

Now by comparing (1) and (2), one sees that the PD controller in position form becomes the PI controller in incremental form if : **1)** $e(n)$ and $r(n)$ exchange positions, **2)** K_d is replaced by K_i, and **3)** $u(n)$ is replaced by $\Delta u(n)$ [1,8]. This modification is shown in Fig. 2, where a PDFLC, with summation at its output, is used instead of the PIFLC.

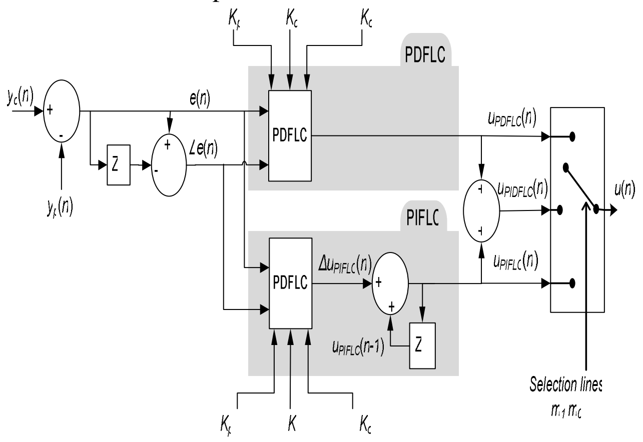


Fig. 2: Main structure of proposed controller.

The fuzzy inference system used in the each PDFLC is a two-inputs, one-output fuzzy system of Mamdani type that uses singleton membership functions for the output variable (it could also be considered as a Sugeno type with constant rule consequents). The first input is the error signal $e(n)$, and the second input is the rate of change of error signal. Before entering the fuzzy inference block, each one of these two inputs is multiplied by a gain coefficient inside the PDFLC, (K_p and K_d or K_p and K_i). In similar manner, the output of the fuzzy inference block is multiplied by the output gain coefficient inside the PDFLC, (K_o). The outputs of the PDFLC and PIFLC, (u_{PDFLC} and u_{PIFLC}) are summed together to form the PIDFLC output (u_{PIDFLC}). Since each PDFLC has its own gains and rules, the final design could act as a PDFLC, a PIFLC, or a PIDFLC depending on the two selection lines m_1 and m_0, as shown in Table I.

Table I: Selection lines setting

m_1	m_0	controller type
0	0	PDFLC
0	1	PIFLC
1	×	PIDFLC

As seen from Fig. 2 the basic block in the proposed controller is the PDFLC. The main components in the proposed PDFLC are: *Gain* block, *Fuzzifier* block, *inference engine* block, and *Defuzzifier* block, which will be discussed in the sections below.

A. Gain Block

The gain block resides at each of the two inputs and also at the output of each of the two PDFLC blocks. It receives two inputs: the variable to be scaled (input or output) and its related **gain coefficient,** and then multiply them. Each gain block contains an eight-bit latch to store the gain coefficient value received from one of the gain ports, depending on selection line values. Each **gain latch** is divided into two parts: 4 bits fraction and 4 bits integer. This limits the maximum scaling of a variable by 15 times (either expanding or compressing). Scaling the variables (or fuzzy sets) by large scale factors may cause fuzzy sets to get too far from its original meaning. If larger scaling factor is required, it would be better to assign new fuzzy sets. Details of **Gain** block are shown in Fig. 3. The gain blocks involve another process which is called **shifting**. **Shifting** process converts the range of the input variables from $[-128 \rightarrow 127\ ]$ to $[0 \rightarrow 255]$. This conversion is necessary because the error signal and rate of change of error signal signals can have positive and negative values, while the used fuzzy inference block in each PDFLC can handle positive values only. The **shift** process implies adding the number (2^7) to the input variable. This addition can be easily implemented by inverting the last bit (MSB) of input variable. Notice that, at the input, the gain process took place before the shift process, while at the output this

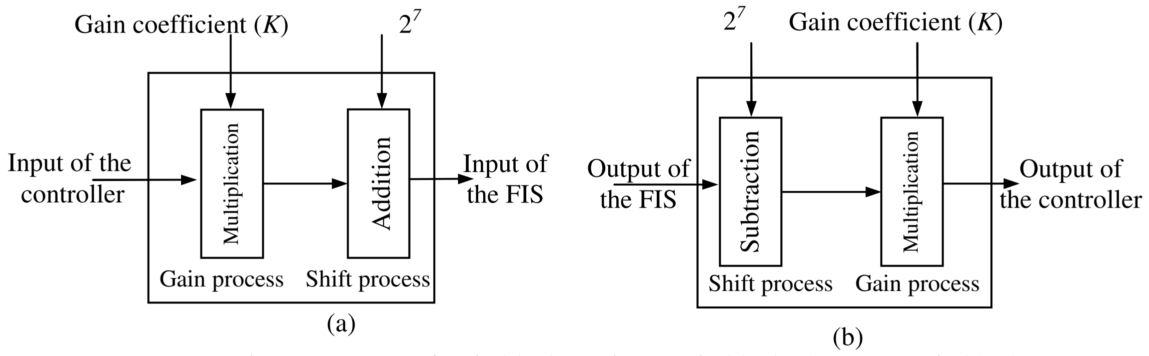


Fig. 3: Structure of Gain block: (a) input gain block, (b) output gain block.

sequence is reversed. Also, the shift process at the output implies subtraction, instead of addition, to convert the range of the output variable from $[0 \rightarrow 255]$ to $[-128 \rightarrow 127]$.

B. The Fuzzifier Block

Fuzzification process is performed using two fuzzifier blocks, one for each input variable. Each fuzzifier block takes the input variable and produces four output values represent the sequence numbers of the two active fuzzy sets, (i and $i+1$), and the membership degrees of the variable in each one of them, (μ_i and μ_{i+1}). Fuzzifier block consists of three elements: **memory module** (called Input fuzzy sets' memory), **inverter**, and **incrementer**, connected as shown in Fig. 4. The memory module is used as a lookup table that stores membership values and active fuzzy set number for each entry value of input. Membership functions of any shape could be implemented in this memory by choosing the right memory words that represent the desired membership functions accurately. The memory module was implemented using core utility provided by *Xilinx core generator system* as a read-only memory (ROM).

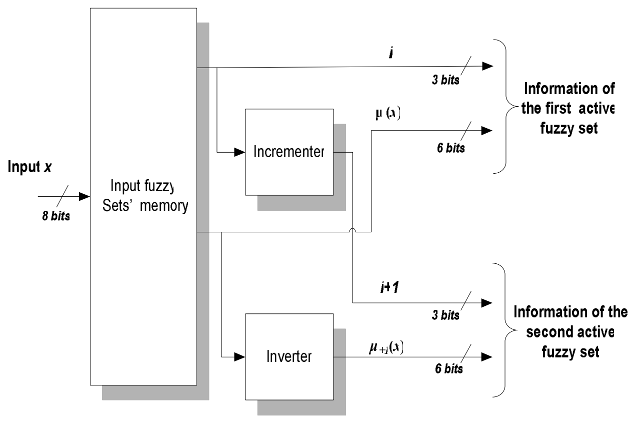


Fig. 4: Structure of Fuzzifier block

Each word in the **input fuzzy sets' memory** is divided into two parts. The first part is 3 bits data word represents the sequence number of the first active fuzzy set. The sequence number of the second active fuzzy set is obtained by adding one to the sequence number of the first active fuzzy set using the **incrementer**. Assigning 3 bits for the sequence number of the fuzzy set will restrict the maximum number of fuzzy sets for each input variable to 8 fuzzy sets.

The second part of memory word is 6 bits data word that represents the membership value of input in the first active fuzzy set. The membership value of input in the second active fuzzy set can be obtained by subtracting the membership value of the input in first active fuzzy set from one [4]. This dictates that the summation of membership values of two consecutive fuzzy set is always equal to one, as in the following equation:

$$\mu_i + \mu_{i+1} = 1 \tag{3}$$

This limits the changing of the shapes of fuzzy sets. However, this restriction is widespread in many fuzzy control systems.

C. Inference Engine Block

The Inference Engine Block used in the proposed design is based on *active rule selector mechanism.* Active rules selector block uses the information delivered from fuzzifier about active fuzzy sets, (have nonzero membership values), to launch only active rules. In this way, using an active rule selector, the number of rules to be processed will be reduced according to this equation:

$$\text{Number of active rules} = V^m \tag{4}$$

where m is the number of inputs, and V is the maximum number of overlapped fuzzy set. In the proposed design, it assumes that $m = 2$ and $V = 2$. Hence, the number of active rules at each time is: $V^m = 2^2 = 4$ rules.

In addition to active rules selector block, inference engine involve two other block: rule memory (contains rule consequent) and minimum circuit (circuit to calculate the applicability degree for each active rule). The memory was designed using core utility. These three blocks are shown in Fig. 5.

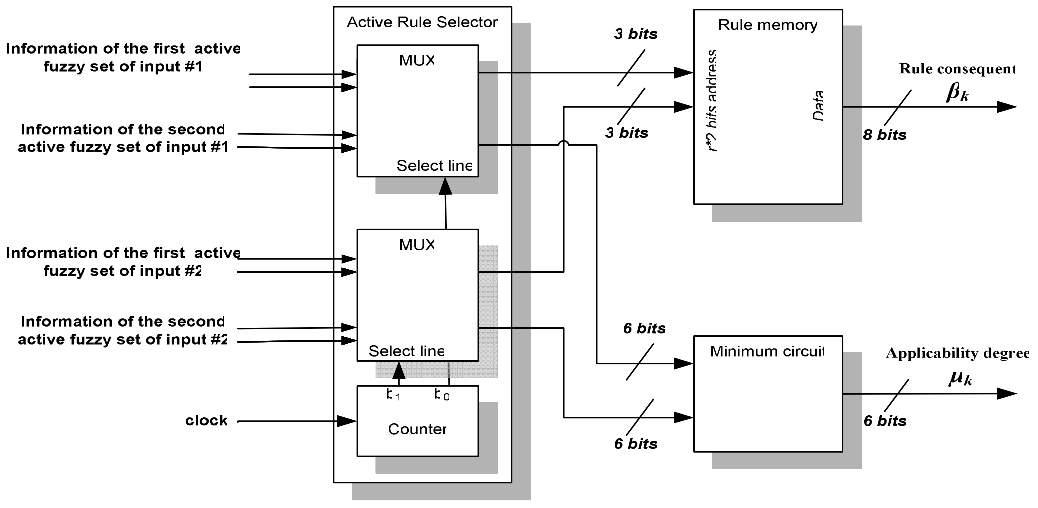

Active Rule Selector
Information of the first active fuzzy set of input #1
Information of the second active fuzzy set of input #1
Information of the first active fuzzy set of input #2
Information of the second active fuzzy set of input #2
clock
MUX
Select line
MUX
Select line
Counter
3 bits
3 bits
Rule memory
r*2 bits address
Data
Rule consequent
β_k
8 bits
6 bits
6 bits
Minimum circuit
Applicability degree
μ_k
6 bits

Fig. 5: Structure of Inference Engine block.

D. Defuzzifier Block

The defuzzification process is performed in the Defuzzifier block using the *Centroid* method defined by Equation below:

$$z = \frac{\sum_{k=1}^{N} \mu_k * \beta_k}{\sum_{k=1}^{N} \mu_k} \qquad (5)$$

where *N* represents the number of the rules, μ_k is the degree of the applicability of the *kth* rule, β_k is the defuzzified value of the output membership function of the *kth* rule [9]. The Defuzzifier involves two **accumulators**, one **multiplier**, and one **divider**. The defuzzifier block accepts four rules consequent and their membership degrees from the inference engine, (sequentially, in four clock cycles), and produces a crisp output to the output gain block, as shown in Fig. 6.

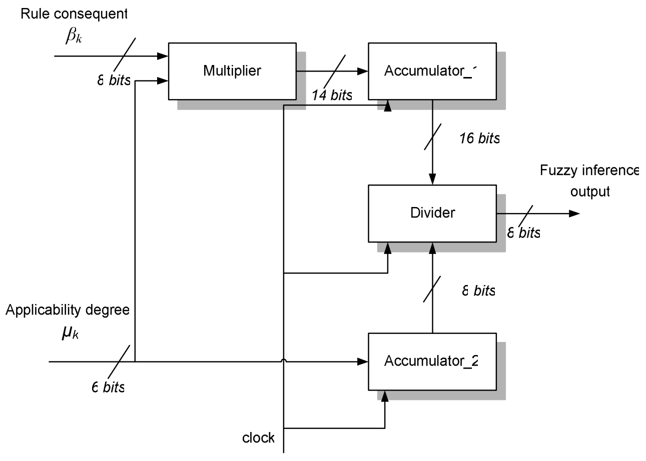


Figure (6) Structure of Defuzzifier block

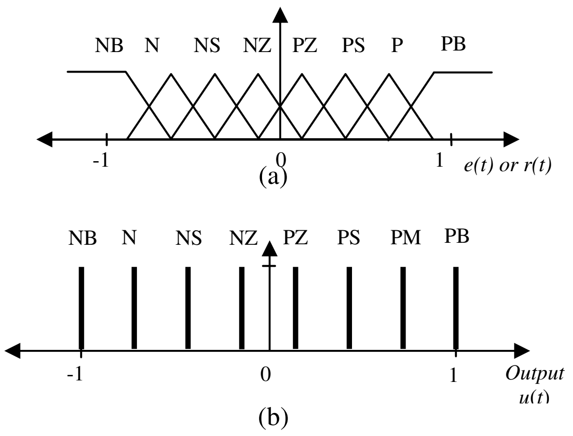
The membership degrees and rules consequents are delivered from the inference engine in a sequential manner in four consecutive clock cycles, instead of being produced in parallel in one clock cycle. This will enhance (reduce) the used area of the target FPGA, at the expense of increasing time interval between input latching and output producing.

IV. FPGA DESIGN CONSIDERATIONS AND SPECIFICATIONS

The chosen target device family in the proposed design is Virtex FPGAs family from Xilinx Company. Virtex FPGAs family offers a useful criterion to the proposed design, which is the internal RAM block. Virtex FPGAs incorporate several large block memories. This criterion is very useful because fuzzy system almost needs large storage element to store fuzzy sets information and rules table. The implementation of the design on FPGA chip is out of the work scope; hence the programming phase in Xilinx implementation tools was not carried out. In order to implement the proposed design, the selected **Target Device** is *xv150* (Xilinx Virtex device of 150 kilo gates), the **Target Package** is *bg256*, and the **Target Speed grade** is (-6). According to simulation reports, the design utilizes one clock net, 61 I/O blocks, and 1394 slices of the target device, with maximum frequency of 40.295 MHz.

V. SIMULATION ENVIRONMENTS AND RESULTS

The proposed controller is designed using ISE4.1 software tool, in addition to ModelSim5.5 software tool, which was used for simulation purposes. The same fuzzy controller is designed and simulated using MATLAB software tool. This Software-Based Controller (SBC) will be used to make a comparison with the proposed design. This comparison is important because it tells us to which extent our FPGA-Based Controller (FBC) is close to similar controller designed as a computer program. For the purpose of simulation symmetric triangular fuzzy sets and singleton fuzzy sets are used for input and output variable respectively, in addition to rule table of 64 fuzzy rules, (shown in Fig. 7).



		Error e(n)							
		NB	**NM**	**NS**	**NZ**	**PZ**	**PS**	**PM**	**PB**
Rate of error r(n)	**NB**	**NB**	**NB**	**NB**	**NM**	**NM**	**NS**	**NZ**	**PZ**
	NM	**NB**	**NB**	**NM**	**NM**	**NS**	**NZ**	**PZ**	**PZ**
	NS	**NB**	**NM**	**NM**	**NS**	**NZ**	**PZ**	**PZ**	**PS**
	NZ	**NM**	**NM**	**NS**	**NZ**	**PZ**	**PZ**	**PS**	**PM**
	PZ	**NM**	**NS**	**NZ**	**NZ**	**PZ**	**PS**	**PM**	**PM**
	PS	**NS**	**NZ**	**NZ**	**PZ**	**PS**	**PM**	**PM**	**PB**
	PM	**NZ**	**NZ**	**PZ**	**PS**	**PM**	**PM**	**PB**	**PB**
	PB	**NZ**	**PZ**	**PS**	**PM**	**PM**	**PB**	**PB**	**PB**

(c)

Fig. 7: Fuzzy sets and rule table: (a) fuzzy sets for $e(t)$ or $r(t)$, (b) fuzzy set for $u(t)$, (c) fuzzy rule table

During test, the controllers (FBC and SBC) are used in unity feedback control systems, as shown in Figure (1) and subjected to 0.5 step input. Mathematical models of two linear plants were used for this test. These two models were chosen in way that represents range of plants used in industrial applications. Many industrial processes, such as temperature, pressure, pH, and fluid-level controls, can be approximated by a first order models. The time delay occurs when a sensor (e.g., a thermocouple) and an actuator (e.g., a heater) are installed with a physical separation. Second order model may represent process such as position control of an ac motor [10]. Discrete transfer functions of the models were obtained using ZOH method, and the selected sampling period (T) is 0.1 second for the first model and 0.25 second for the second model. The discrete transfer functions (in *z-plane*) of models are listed below:

1. First order plant:

$$G_1(z) = \frac{0.1903\ z^{-1}}{1 - 0.9048\ z^{-1}} \quad , T = 0.1 \tag{6}$$

2. Second order plant with delay:

$$G_2(z) = z^{-2}\frac{0.02511\ z^{-1} + 0.01997\ z^{-2}}{1 - 1.48\ z^{-1} + 0.5028\ z^{-2}} \quad , \ T = 0.25 \tag{7}$$

Each one of these plants was designed in MATLAB (for simulation in MATLAB), and also in non-synthesizable VHDL code (for simulation in ModelSim). Since each controller could serve as PDFLC, PIFLC, or PIDFLC, therefore, a test is made for each one of these types using different plants. Fig. 8 shows step responses of the first order plant when controlled using the PDFLC, PIFLC and PIDFLC, while Fig. 9 shows step responses of the delayed second order plant when controlled using the PDFLC, PIFLC and PIDFLC. The values of K_p, K_d, K_i, *and* K_o used in this test were selected using trial and error.

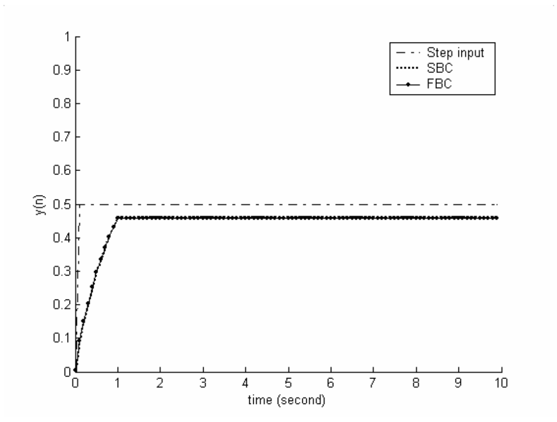
VII SIMULATION RESULT DISCUSSION

As seen in Fig. 8 and Fig. 9, the responses of the systems that use FBC are smooth and much similar to the SBC responses. The mean difference between the SBC results (step response) and the FBC results, shown in Fig. 8 and Fig. 9, is calculated, for each case, and listed in Table II. The table shows that the absolute mean of differences in the plant response, (for 0.5 step input), between the SBC and FBC is less than 0.01 for all test cases (less than 0.5% of the output range). The table also shows the mean of differences between the control action ($u(n)$) of the SBC and FBC for all test cases.

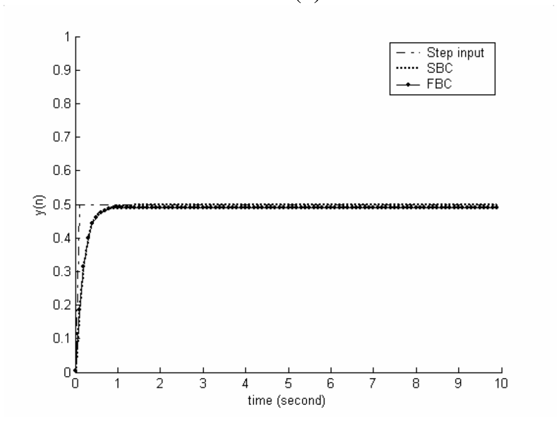
Table II: Mean differences between SBC and FBC results

Controller type	Plant type	Mean differences between **SBC** and **FBC**	
		Step response	Control action
PDFLC	G_1	0.0016	0.0039
	G_2	0.0001	0.0040
PIFLC	G_1	0.0072	0.0072
	G_2	0.0067	0.0081
PIDFLC	G_1	-0.0076	-0.0010
	G_2	0.0076	0.0086

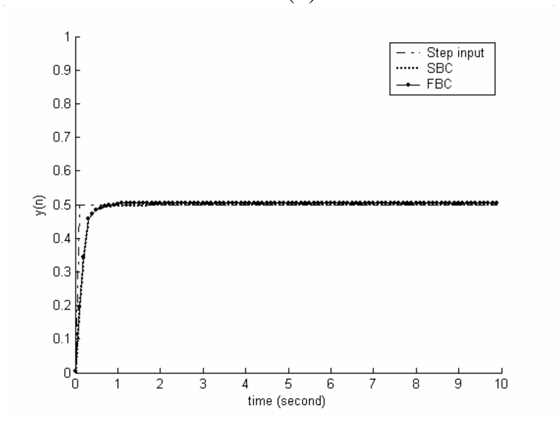
It is noticeable that some responses in figures (8) and (9), have large steady state error (e_{ss}) and/or slow response (long rise time (t_r)). Here we should emphasize that the aim of this test is to find to which extent the FBC responses are close to SBC responses, and not to find how to tune a PIDFLC to get better response.



(a)

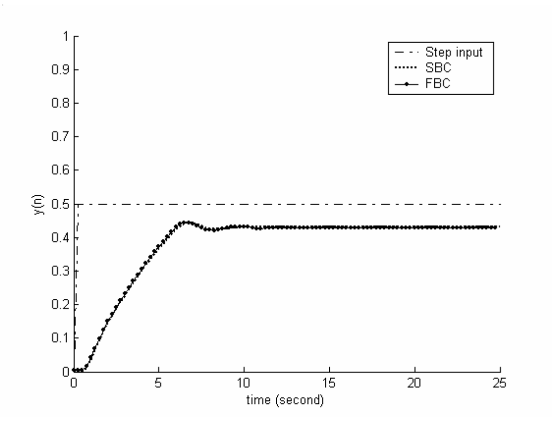


(b)

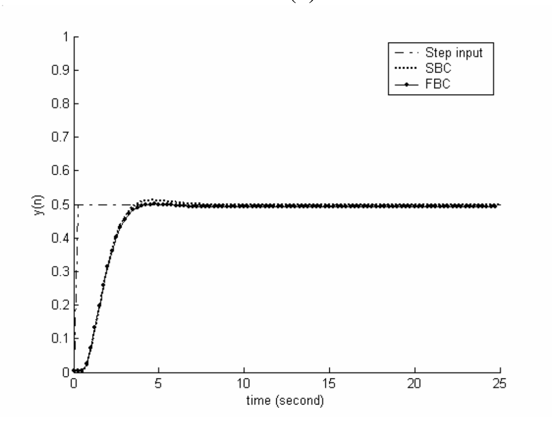


(c)

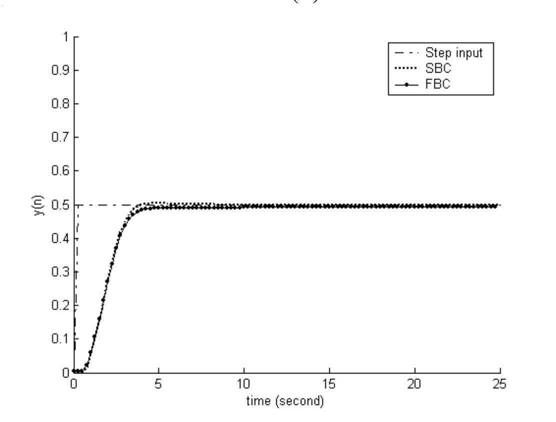
Fig. 8: First order plant controlled by (a) PDFLC (b) PIFLC (c) PIDFLC



(a)



(b)



(c)

Fig. 9: Delayed second order plant controlled by (a) PDFLC (b) PIFLC (c) PIDFLC

VIII. CONCLUSION

The design of a PID-like fuzzy logic controller based on fuzzy system with *active rule selection mechanism* and four tunable gains factor on FPGA chip is presented in this paper. Simulation results of applying the design on the target chip state that the design utilizes 1394 slices of the target device and needs 17 clock cycles per action. With the maximum clock frequency 40.295MHz, the controller was able to produce an output in less than 0.421 µs. Therefore, the proposed controller will be able to control many industrial applications with sampling time ranging from milliseconds, e.g. in pressure control, up to higher sampling time in the case of temperature control of larger installations (industrial furnaces). This small-size high-speed chip is able to offer adequate accuracy. The result of simulation shown that the step responses of first and second order linear models controlled by the proposed controller were very close to responses of the same models controlled by a software-based controller. The absolute mean of differences between the responses, was less than 0.5% of the output range.

REFERENCES

[1] H. Ying, *Fuzzy Control and Modeling, Analytical Foundations and Applications*. USA: Institute of Electrical and Electronic Engineers Inc., 2000.

[2] O. Karasakal, E. Yesil, M. Guzelkaya, and I. Eksin, " Implementation of a New Self-Tuning Fuzzy PID Controller on PLC," Turkish Journal of Electrical Engineering, Vol.13, No.2, pp.277-286, 2005.

[3] V. Tipsuwanpornm T. Runglimmawan, S. Intajag, and V. Krongratana, "Fuzzy Logic PID Controller Based on FPGA for Process Control," IEEE International Symposium on Industrial Electronics, Vol. 2, pp. 1495-1500, 4-7 May 2004.

[4] S. S. Solano, A. Barriga, C. J. Jiménez, and J. L. Huertas, "Design and Application of Digital Fuzzy Controllers," Sixth IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'97), Vol. 2, pp. 869-874, July 1-5, 1997, Barcelona-Spain.

[5] D. Falchieri, A. Gabrielli, and E. Gandolfi, "Very Fast Rate 2-Input Fuzzy Processor for High Energy Physics," Fuzzy Sets and Systems Vol. 132, Issue 2, pp. 261-272, December 2002.

[6] S. H. Huang and J. Y. Lai, "A High-Speed VLSI Fuzzy Inference Processor for Trapezoid-Shaped Membership Functions," Journal of Information Science and Engineering Vol. 21, No. 3, pp.607-626, May 2005.

[7] S. H. Huang, and J. Y. Lai, "A High Speed Fuzzy Inference Processor with Dynamic Analysis and Scheduling Capabilities," IEICE Transaction Information & System., Vol. E88-D, No.10 October 2005.

[8] G. K. Mann, B. G. Hu, and R. G. Gosine, "Analysis of Direct Action Fuzzy PID Controller Structures," IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 29, No. 3, pp. 371-388, June, 1999.

[9] S. S. Farinwata, D. Filev, and R. Langari, ***Fuzzy Control: Synthesis and Analysis***. John Wiley & Sons, Ltd., 2000.

[10] B. G. Hu, G. K. Mann, and R. G. Gosine, " New Methodology for Analytical and Optimal Design of Fuzzy PID Controllers," IEEE Transactions on Fuzzy Systems, Vol. 7, No. 5, pp. 521-539, October, 1999.

Mohammed Y. Hassan This author became a Member (2007) of IAENG. He was born in Baghdad, Iraq 1967. He received his B. Sc. in Electrical and Electronics Engineering from Al-Rasheed Collage of Engineering and Science, University of Technology, Iraq in 1989. Master Degree in Control Engineering from Al-Rasheed Collage of Engineering and Science, University of Technology, Iraq in 1995 and he received his Ph. D. in Control Engineering and Automation from the University of Technology, Iraq 2003. He is now a lecturer in the control and systems Engineering Department, University of Technology in Baghdad, Iraq.

He has several research publications in journals and conference proceedings. His areas of research interest are in Intelligent Control, Adaptive control, Modeling, Fuzzy logic,Neural network, Genetic Algorithm, Microcomputers and Microcontrollers.

Dr. Hassan has received in 2007 an Endeavour postdoctoral research Fellowship award from the department of Education, Training and Science in Australian government to do a research in the school of Engineering and Mathematic, Edith Cowan University in West Australia.

Waleed F. Sharif was born in Baghdad, Iraq, in 1982. He received his B. Sc. degree and M. Sc. degree from the University of Technology, Baghdad, Iraq, in 2004 and 2007, respectively, all in control and systems engineering.

His current research interests include fuzzy modeling and fuzzy control systems

He is currently working as an assistant lecturer in the same department.