IEICE Electronics Express, Vol.7, No.3, 132–137

Implementation of Multistructure PID-like fuzzy logic controller using FPGA

Zeyad Assi Obaid[a)], **Nasri Sulaiman, M. H. Marhaban, and M. N. Hamidon**

Department of Electrical & Electronic Engineering, Faculty of Engineering, University Putra Malaysia, 43400 UPM Serdang, Selangor Darul Ehsan, Malaysia

a) *eng.alhamdany@yahoo.com*

Abstract: This paper presents an implementation of Multistructure PIDFLC. Modification has been made to structure of the proposed PIDFLC in order to make it acts as PDFLC, PIFLC or PIDFLC depending on two external signals. Two versions of this controller have been designed using VHDL language for FPGA implementation. A new package has been designed in VHDL code to implement trigonometric functions and fourth-order Runge-Kutta method to test the proposed design with nonlinear systems. The controller was able to produce an output in 0.3 μsec for linear plants and 0.7 μsec for nonlinear plant. Therefore, the proposed controller will be able to control many systems with high sampling rate.

Keywords: PIDFLC, FPGA implementation, nonlinear systems, altera

Classification: Electronic instrumentation and control

References

[1] T. Jain, V. Patel, and M. J. Nigam, "Implementation of PID Controlled SIMO Process on FPGA Using Bacterial Foraging for Optimal Performance," *Int. J. Comput. Electr. Eng.*, vol. 1, no. 2, pp. 107–110, June 2009.

[2] V. Tipsuwanporn, S. Intajag, and V. Krongratana, "Fuzzy Logic PID controller based on FPGA for process control," *Proc. IEEE Int. Symp. Ind. Electron.*, Bangkok, Thailand, vol. 2, pp. 1495–1500, 4-7 May 2004.

[3] Z. A. Obaid, N. Sulaiman, and M. N. Hamidon, "FPGA-based Implementation of Digital Logic Design using Altera DE2 Board," *International Journal of Computer Science and Network Security (IJCSNS)*, vol. 9, no. 8, pp. 186–194, July 2009.

[4] Z. A. Obaid, N. Sulaiman, M. H. Marhaban, and M. N. Hamidon, "FPGA-Based Fuzzy Logic: Design and Applications – a Review," *Int. J. Eng. Technol.*, vol. 1, no. 5, pp. 491–502, Dec. 2009.

[5] L. Reznik, *Fuzzy controllers*, Newnes, first edition, 1997.

DOI: 10.1587/elex.7.132
Received October 19, 2009
Accepted January 04, 2010
Published February 10, 2010

1 Introduction

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 cannot be considered as a suitable design solution for this type of application higher density programmable logic devices such as FPGA 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 FPGA provide additional flexibility: they can be used with tighter time-to-market schedules [1, 2, 3, 4].

2 Layout of the Proposed Controller

Generally, this controller accept two types of outputs, the first one is the plant (Yp) and the second one is the desired output (Yd), both of them is digital signals, and deliver the control action signal as a digital output. It also accepts four 8-bit digital signals that represent the gain parameters needed by the controller (proportional gain Kp, derivative gain Kd, integral gain Ki, and output gain Ko), and other two one-bit signals to select the type of the controller (PD fuzzy logic controller, PI fuzzy logic controller, or PID fuzzy logic controller). Fig. 1 shows the general layout of the controller chip in a unity feedback control system. Fuzzy controller applications do not require high accuracy. Accuracy of 6–9 bits is enough and is quite sufficient for different applications. Many designed FIS chips use this range of bits [5], since two versions of the controller have been designed to make a comparison in which version is closest to Matlab-based design: the first one uses 6 bits for each input and output variables, and 4 bits for membership degree, while the other uses 8 bits and 6 bits respectively.

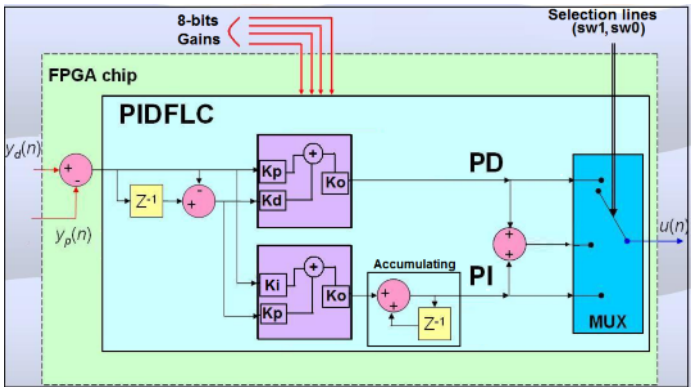


Fig. 1. Main Structure of Proposed PIDFLC

3 Structure of the Proposed PIDFLC

Generally, to represent PID fuzzy logic controller, it was required to design

DOI: 10.1587/elex.7.132
Received October 19, 2009
Accepted January 04, 2010
Published February 10, 2010

a fuzzy inference system with three inputs that represent the proportional, derivative, and integral components, and each one of them can have up to eight fuzzy sets. So that the maximum number of the required fuzzy rules to 83=512 rules. To avoid this huge number of rules, the proposed controller has been designed using two parallel PD fuzzy logic controllers to implement the PID fuzzy logic controller. The second PDFLC has been converted to PIFLC by accumulating its output. Fig. 1 shows the structure of proposed PID fuzzy logic controller. Both controllers, PD fuzzy logic controller and PI fuzzy logic controller, receive the same error signal. The error signal is calculated by subtracting plant output (yp) from the desired output (yd). The main block in the PD fuzzy logic controller is the fuzzy inference block. The proposed fuzzy inference block is two inputs, one output fuzzy system of Mamdani type that uses singleton membership functions for the output variable. The first input is the error signal $e(n)$, and the second input is the rate of change of error signal defined as the difference between two consecutive error values.

Before entering the fuzzy inference block, each one of these two inputs have been multiplied by a gain coefficient inside the PD fuzzy controller (K_p and K_d or K_p and K_i). In similar manner, the output of the fuzzy inference block is multiplied by a gain coefficient inside the PD fuzzy logic controller, (K_o). At the same time, the output of the fuzzy inference block in the second PD fuzzy controller is multiplied by a gain coefficient then accumulated to form the uPIFLC. Both outputs (uPD and uPI) are added together to form the PIDFLC output (uPID). Since each PDFLC has its own gains and rules, the final design could work as a PDFLC, PIFLC or a PIDFLC) depending on the two selection lines sw1 and sw0, where, sw1sw0= 00, gives PD fuzzy logic controller, sw1 sw0= 01 gives PI fuzzy logic controller, and sw1 sw0=0x gives PID fuzzy logic controller. The main components in the proposed PD fuzzy logic controller are: *Input/Output* block, *Fuzzifier* block, *inference engine* block, and *Defuzzifier* block.

4 Test Bench and Simulation Results

For the purpose of simulation symmetric triangular fuzzy sets and singleton fuzzy sets with 8 linguistic variables have been used for input and output variable respectively, in addition to rule table of 64 fuzzy rules. At first, a test is performed to make sure that the fuzzy inference system used inside the FPGA-based design is working properly This test is performed to make sure that the fuzzy inference system used inside the FPGA-based controller (6FBC or 8FBC) is working properly. This test involves generating control surface using fuzzy sets and rule table, this test has been used to make a comparison between both types of FBC with Matlab-based (MSBC), and shows that 8FBC is superior to 6FBC and it's much close to MSBC.

Case Study 1: Second order model may represent process such as position control of an ac motor [7] Equation (1) shows the mathematical plant

DOI: 10.1587/elex.7.132
Received October 19, 2009
Accepted January 04, 2010
Published February 10, 2010

model, discrete transfer functions of this model has been obtained using ZOH method, and the selected sampling period (T) is 0.52. The values of Kp, Kd, Ki, and Ko used in this test were selected using trial and error.

$$G(z) = \frac{0.02511z^{-1} + 0.01997z^{-2}}{1 - 1.48z^{-1} + 0.5028z^{-2}} \tag{1}$$

The controller gives action at 0.3 μs; when PIDFLC applied for this system, as shown in Fig. 2, 8FBC response is close to the responses using MSBC, with zero error and little overshot. The Mean differences between MSBC and 6FBC for Step response and control action are -0.0256 and -0.0009 respectively, and The Mean differences between MSBC and 8FBC for Step response and control action are -0.0030 and 0.0021 respectively, since the 8FBC is superior to 6FBC and its much close to MSBC.

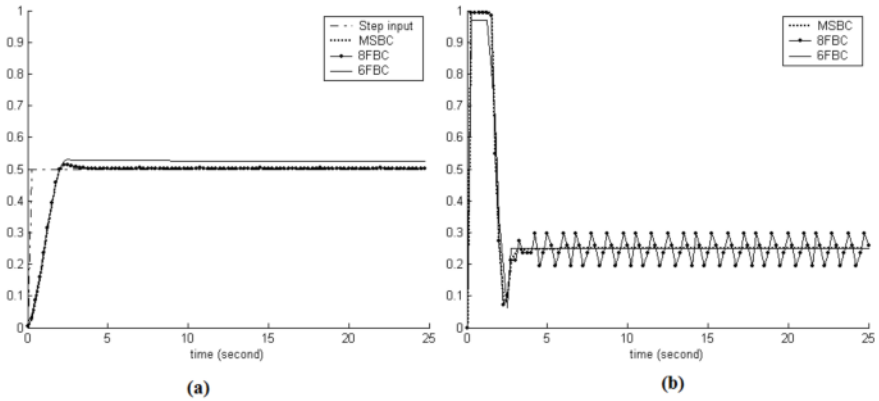


Fig. 2. Case study 1 controlled by PIDFLC (a) step response, (b) control action

Case Study 2: This case is considered as a special case with the proposed design, because of VHDL accepts four mathematical operation only, addition, subtraction, division and multiplication, since it's difficult to represent nonlinear elements like trigonometric functions. In this case, a mathematical model of nonlinear plant has been used to test the proposed controller with unity feedback control system; this model is characterized by Equation (2) and Equation (3).

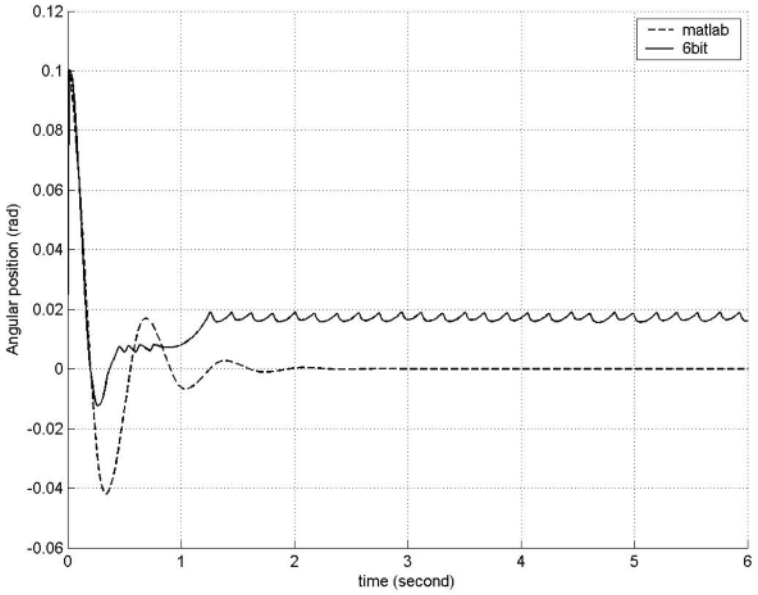
$$CS_5 = \ddot{y} = \frac{9.8\sin(y) + \cos(y)\left[\dfrac{-\bar{u} - 0.25\dot{y}^2\sin(y)}{1.5}\right]}{0.5\left[\dfrac{4}{3} - \dfrac{1}{3}\cos^2(y)\right]} \tag{2}$$

$$\dot{\bar{u}} = -100\bar{u} + 100u \tag{3}$$

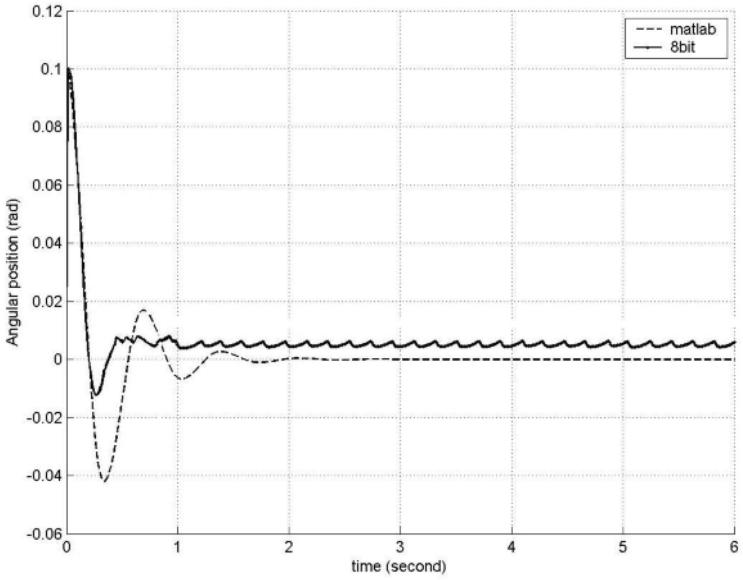
The first order filter on u to produce $\bar{u}$ represents an actuator. Assume the initial conditions $y(0) = 0.1$ radians ($= 5.73$ deg.), $\dot{y}(0) = 0$, and the initial condition for the actuator state is zero. For simulation of the fourth-order, Runge-Kutta method has been used with an integration step size of 0.01. Again, this plant has been designed using MATLAB software (for simulation in MATLAB), and in non-synthesizable VHDL code (for simulation

DOI: 10.1587/elex.7.132
Received October 19, 2009
Accepted January 04, 2010
Published February 10, 2010

in ModelSim). A special package was designed in VHDL code to implement trigonometric functions and fourth-order Runge-Kutta method which are not available in Quartus II (or in ISE) standard libraries. The values of K_p, K_d, K_i, and K_o used in this test were selected using trial and error. The controller gives action at 0.7 μs after the input latching. When using nonlinear system for test, both versions (6FBC and 8FBC) provide generally good responses though there is some oscillation. (one must not be deceived by the steady state error that appears in Figure (4), since it represents less than 1% of the output range in the case of 6FBC and less than 0.5% of the output range, in the case of 8FBC). The absolute mean difference between the nonlinear



(a)



(b)

Fig. 3. Nonlinear plant response controlled by (a) 6FBC (b) 8FBC

DOI: 10.1587/elex.7.132
Received October 19, 2009
Accepted January 04, 2010
Published February 10, 2010

plant response, using MSBC, and the nonlinear plant response, using 6FBC, is less than 0.0155. The absolute mean difference between the nonlinear plant response, using MSBC, and the nonlinear plant response, using 8FBC, is less than 0.0085 as shown in Fig. 3.

5 Implementation of the Proposed PIDFLC

The proposed PIDFLC has been implemented using Altera DE2 board, this board offers a rich set of features that make it suitable for use in a laboratory environment for university and college courses and can used for any design implementations, as well as for the development of sophisticated digital systems by using hardware description language (HDL). All connections are made through the Cyclone II 2C35 FPGA device in order to provide maximum flexibility for the user. Thus, the user can configure the FPGA to implement any system design.

6 Conclusion

Simulation environments have been built using non-synthesizable VHDL code for the purpose of simulation in ModelSim, and the same design is coded in Matlab for the purpose of simulation in Matlab (MSBC). Two version of the controller has been designed, the first one is 6-bits which uses 6-bits for each input/output variables (6FBC), while the second uses 8-bits each input/output variables (8FBC). Two case studies have been used in order to test this controller. From these results, 8FBC is superior to 6FBC and it's much close to MSBC. The controller was able to produce an output in 0.3 μsec (after input latching) for linear plants and 0.7 μsec for nonlinear plant. Therefore, the proposed controller will be able to control systems with high sampling rate.

Acknowledgments

The authors would like to thank firstly, our god, and all UPM staff and all friends who gave us any help related to this work. Finally, the most thank is to our families and to our countries which born us.

DOI: 10.1587/elex.7.132
Received October 19, 2009
Accepted January 04, 2010
Published February 10, 2010