Implementation of Optimized Fuzzy Logic Controller with VHDL

Meghana Hasamnis , Jayu Kalambe and Deepali Shelke

*Abstrac*t : **This paper proposes a new hardware architecture for fuzzy logic controller. In place of MAX-MIN inference module and COG defuzzifier the FLC is replaced by read-modify-write operation and Coarse-to-fine search algorithm respectively. The accuracy and cost of proposed fuzzy logic controller is obtained by read-modify-write operation and speed of operation is increased by Coarse-to-fine search algorithm that considers both membership values and span of membership function in calculating the crisp value and division operation is eli[1]minated which was present in defuzzifier module. The FLC is partitioned into many temporally independent functional modules, and each module is implemented individually on the FLC automatic design and implementation system, which is an integrated development environment for performing many subtasks. Simulation of the FLC is performed in VHDL.**

Keywords : **Fuzzy Logic Controller, Read-modify-Write Module, Coarse-to-fine Search Algorithm, VHDL Simulation.**

I. INTRODUCTION

FUZZY Logic Controller is widely applied to Industrial process controls. For most real world control and signal processing problems, the information concerning design, evaluation, realization etc., can be classified into two kinds: Numerical information obtained from sensor measurements and Linguistics information obtained from human experts.

A typical FLC consists of four principal units:

1) Fuzzifier which converts a crisp input to a fuzzy term set.

2) Fuzzy rule base, which stores fuzzy rules describing how the fuzzy systems performs.

3) Fuzzy inference engine ,which performs an approximate reasoning by associating input variable fuzzy rules.

4) Defuzzifier, which converts the FLC output to a crisp value for the actual system input.

The control performance is more or less influenced by the selection of the fuzzy sets of the linguistic variables, the shape of membership functions, the fuzzy rule base and the inference mechanism and there are many approaches to implement the FLC in hardware which can be categorized into three classes: 1) general-purpose fuzzy processor with specialized fuzzy computation 2) dedicated fuzzy hardware for specific application 3) dedicated fuzzy hardware for specific application. The general – purpose fuzzy processor can be implemented quickly but it provides lower performance, while the dedicated fuzzy hardware requires long time for development but it offers high performance.

The first consideration in the implementation of FLC is the choice of implementation mode, they are analog or digital implementation but we have chosen a digital implementation because it gives better design flexibility and easy programmability.

The second consideration in the implementation of dedicated FLC is getting higher control performance and

The third consideration in the implementation of dedicated FLC is reducing the hardware complexity for that we take the following schemes.

Firstly, the MIN and MAX module in the FLC are often implemented by some tree like structures in order to balance the hardware complexity and the operation speed. However , the hardware complexities in MIN and MAX modules are increased by $O(n.2^n)$ and $O(2^n)$respectively, where n is the number of input variables. We solve

This hardware explosion problem by using the read-modify-write operation based on the structure file at the cost of slower operation speed for that the read-modify-write operation needs $O (2^n)$.

Secondly Coarse-to-Fine search algorithm is designed and implemented to reduce the hardware complexity by eliminating the COG defuzzifier block as it contains division module which consumes space and time is consumed and hence speed is less.

II. AN ARCHITECTURE OF THE PROPOSED FLC

The below diagram explains the MAX-MIN Inference module. From practical application of fuzzy logic to control systems, we apply some constraints to reduce the hardware complexity of FLC and improve the control performance of the FLC.

1) The observed inputs of the FLC are crisp and quantized into a finite number of levels.
2) For each input variable, the overlapping degree of its MF is at most two.
3) MF's in the output variable have a symmetric triangular shape.
4) Both membership values and spans of constituent MF's are used to compute an accurate crisp value.

Ms.Meghana Hasamnis , Lecturer, S.R.K.N.E.C, Nagpur , Email: :meghanahasamnis@rediffmail.com
Ms .Jayu Kalambe, Lecturer, S.R.K.N.E.C, Nagpur, Email:jayu_kalambe@rediffmail.com
Ms. Deepali Shelke Lecturer, S.R.K.N.E.C, Nagpur
Email : :deepshelke@yahoo.co.in

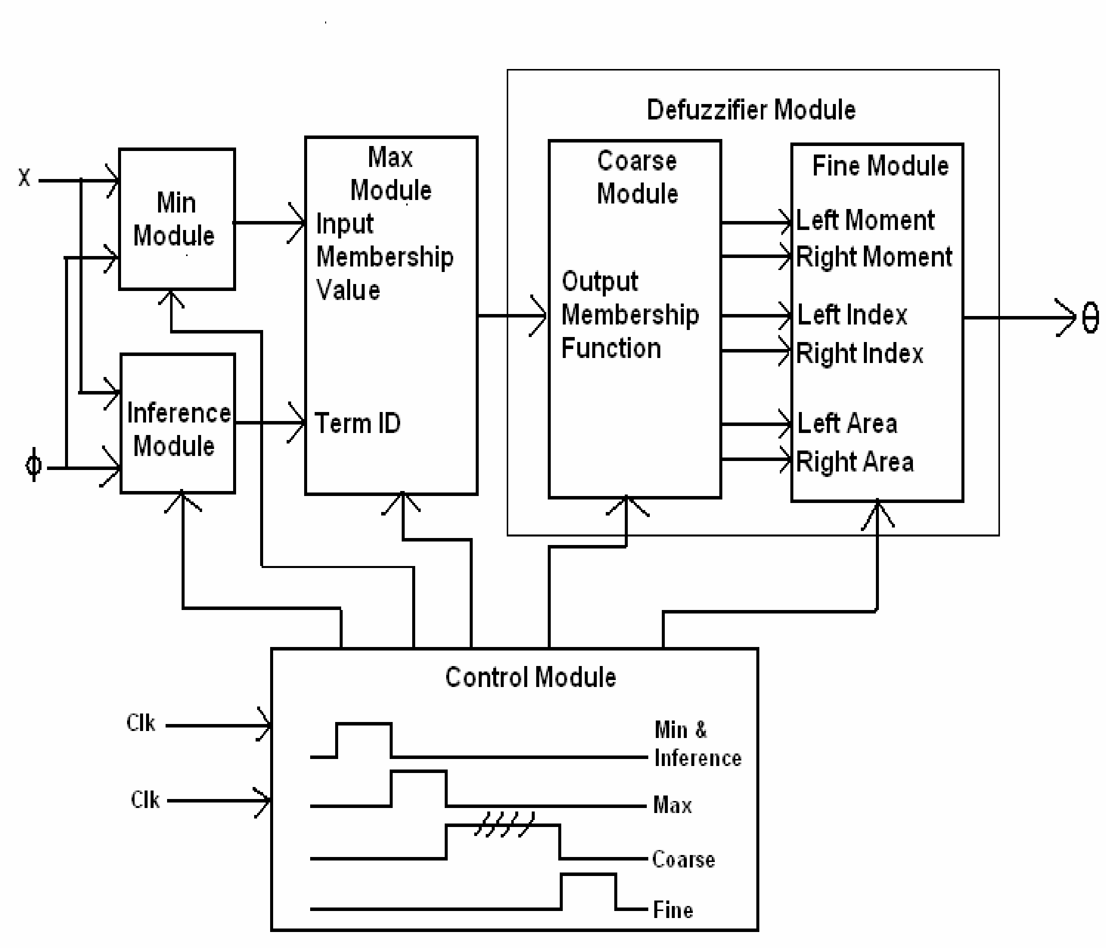


Fig.1 Architecture of the proposed FLC

With the First constraint, the MAX-MIN inference can be simplified as a lookup table operation. Both membership values and index numbers of input MF's are prestored in the input lookup tables, and the observed crisp inputs serve as the addresses for reading them. The second constraint implies that the n-dimensional input can activate 2^n control rules at most. So, the fuzzy rule base can be decomposed into 2n disjoint subrule bases that are operated concurrently. With the third constraint, each output MF is treated as a singleton located at the center of the MF. So, the computational complexity of the COG defuzzification is significantly reduced. The fourth constraint is needed for improving the control performance.

Architecture of the proposed FLC hardware consists of MIN module, inference module, and MAX module. The Defuzzifier module is divided into two modules coarse and fine and implemented using search algorithm.

A. *Min Module*

Assume that the universe of input variable xi is [lxi, uxi], where lxi and uxi are the minimum and maximum bound of xi respectively. Then all input are uniformly quantized into 2^p intervals accordingly. Similarly, the MF's of fuzzy terms in the input variables xi can be quantized into 2^q levels, so that each quantization level has a value of $1/2^q$. Let Nxi fuzzy terms of the input variable xi be{$T_x{}^1$,$T_x{}^2$,…….,$T_{xi}{}^{Nxi}$} .

Each term T_{xi}^j (j=1,2,……,N_{xi}) can be represented by a pair of the MF and index number as (M_{xi}^j,I_{xi}^j). Since the overlapping degree of input variable s limited to 2,N_{xi}, fuzzy terms of xi can be divided into two disjoint term sets, T_{xi}^o and T_{xi}^e as T_{xi}^o = {(M_{xi}^j,I_{xi}^j) | j=2.k-1, k=1,2,…, N_{xi}/2 } T_{xi}^e = {(M_{xi}^j,I_{xi}^j) | j=2.k, k=1,2,…, N_{xi}/2 }
So, the FLC with n input variables requires 2.n input lookup tables where each lookup table consists of two parts, MF table and ID table. The required bits for storing the membership values in the MF table are 2^p.q . The required bits for storing the index values in the ID table are 2p.$\log_2$(N_{xi}/2).q, since each fuzzy term in the individual term set can be identified with lod_2(N_{xi}/2_ bits) . In order to balance the hardware complexity as well as operation speed. However, the hardware complexities for the MIN and MAX operation are increased explosively by O (2^n) and O (2.2^n), respectively, where n is the number of the input variable. We solve this hardware explosion problem by using the read-modify-write operation. The read-modify-write operation needs O (2^n) registers to perform MIN and MAX operation.

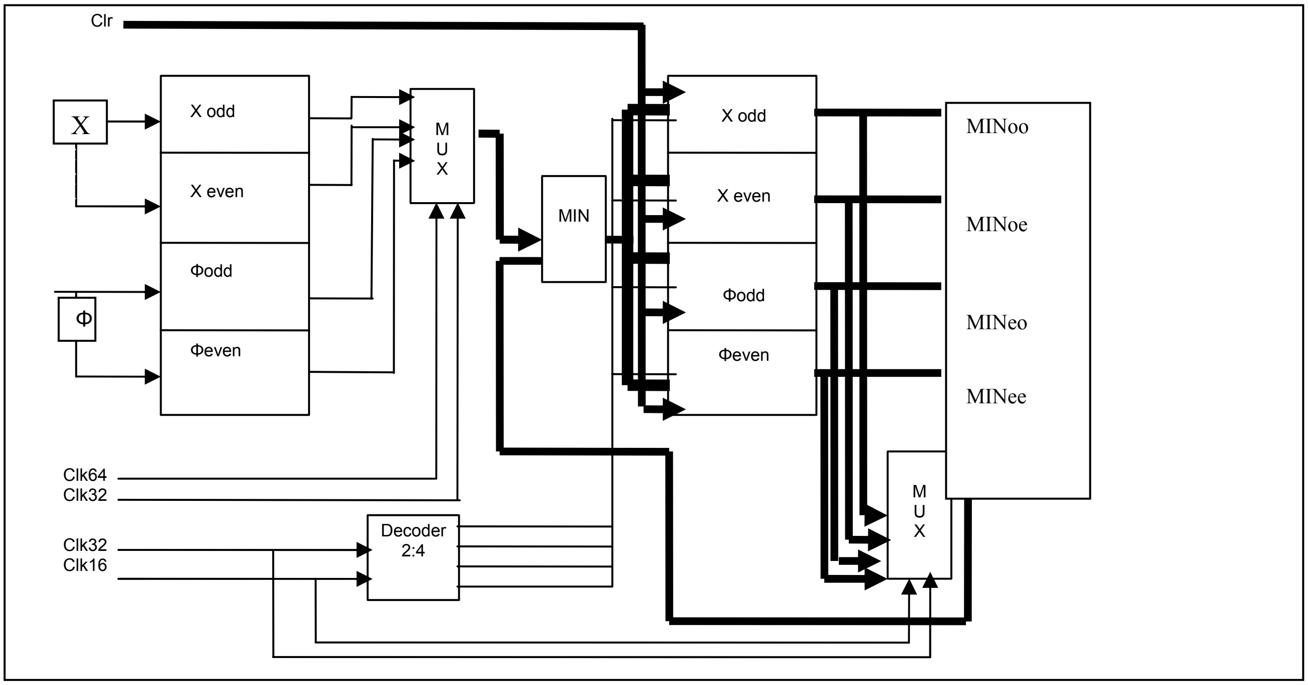


Fig 2. Circuit diagram of the MIN Module.

In order to reduce the hardware complexity, we propose the read-modify-write operation where modify means a MIN operation . It needs only 2^n registers to perform the read-modify-write operation . This implies the reduction of hardware by O(n) . The read-modify-write operation for the MIN operation is done as follows. Initially , the contents of 2^n registers set to the maximum value(FFH) that a membership value can have. We need n read-modify-write cycles to complete a MIN operation, where the input variable n is involved in the read-modify-write operation during the ith cycle. The ask which occurred during only the ith cycle is explained , since it is the same for all cycles. The content in the first register is read and compared with the clipped membership value of an input variable xi. The smaller one between two values is written back to the first register. This operation is repeated up to the 2^nth register. One point here to remember that the frequency of reading the odd and even MF lookup tables alternatively is 2^n-I during the ith cycle. Consider n=2 as an example. The order of referencing x1 is x_1^e,x_1^e,x_1^o,x_1^o during the first cycle. The order of referencing x_2^e,x_2^e,x_2^o,x_2^o during the second cycle. So, it takes n. 2^n the read-modify-write operation to compute a MIN operation.

B. INFERENCE MODULE

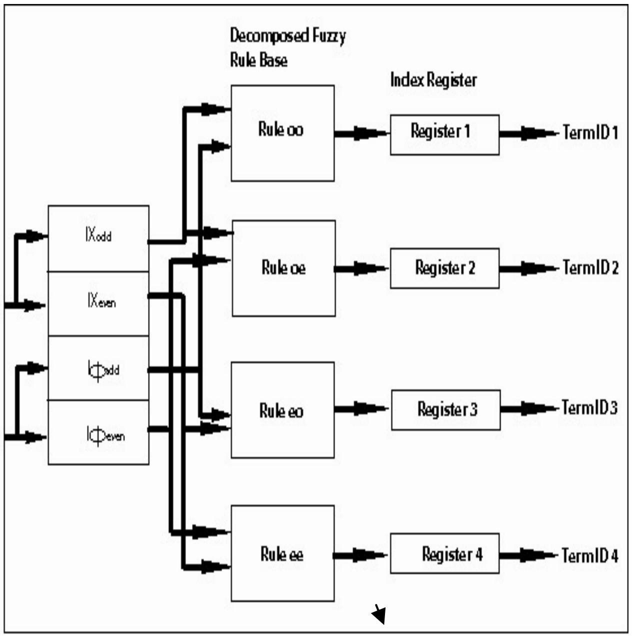


Fig.3 Circuit diagram of the Inference Module

In the case of n-input single-output FLC, a fuzzy rule base can be treated as the n-dimensional cell array where each cell $C(x_1,x_2,\ldots\ldots\ldots.x_n\ )$has the output index number $I_y(x_1,x_2,\ldots\ldots\ldots.x_n\ )$ corresponding to the consequence in the specific control rule R $(x_1,x_2,\ldots\ldots\ldots.x_n\ )$. Since overlapping among the fuzzy terms in the input variable are eliminated by separating the fuzzy terms into two disjoint fuzzy sets with the odd and even index numbers, respectively.

The fuzzy rule base can be decomposed into 2^n subrule bases as

Ro,o={ C(i,j)| i = 2.k-1Λ j = 2.l-1 }
Ro,e={ C(i,j)| i = 2.k-1Λ j = 2.l
Re,o={ C(i,j)| i = 2.k Λ j = 2.l-1 }
Re,e={ C(i,j)| i = 2.k Λ j = 2.l }

Fig.3 shows the circuit diagram of the inference module when the number of input variable is two. The inference module requires $(\Pi_{i=1}^{n} N_{xi}).\log_2 N_y$ bits to store the output index numbers of the consequences where N_{xi} and N_y are the number of the fuzzy terms of the input variables xi and output variables y, respectively. The decomposition of fuzzy rule base does not increase the memory space for storing the output index numbers, since the decomposition simply divides the fuzzy rule base into 2^n disjoint subrule bases where each subrule base requires $(\Pi_{i=1}^{n} N_{xi}/2).\log_2 N_y$ bits for storing its constituent output index numbers.

C. MAX Module

2^n disjoint subrule bases produce 2^n output index numbers that represent the consequences of their constituent control rules about an observed *n*-dimensional input. Since some subrule bases can provide the same output index numbers , we need a MAX operation to resolve these conflicts.

In order to reduce the hardware complexity, we propose another *read – modify – write* operation where modify means a MAX operation. It needs M registers to perform a *read – modify – write* operation, where *M* is the number of output fuzzy terms. This implies the reduction of hardware of the proposed *MAX* module when the number of input variables is two. Here the upper 2^n to 1 multiplexer is used to select the 2^n the minimum membership value and the lower 2^n to 1 multiplexer is used to select one of 2^n output index numbers .[2]

The read – modify – write operation for the MAX operation is done as follows. Initially, the contents of *M* register sets to the minimum value (00H) that the membership value can have. Firstly the content in the register accessed by an output index number is read and compared with the membership value . The greater one between two values is written back in the register. The *read – modify – write* operation continues until the final output index numbers is taken. Thus, it takes 2^n cycles to complete a MAX operation. This sequential processing slows down the MAX computation time by 2^n times. However this slowdown is not a serious problem, because most of the processing time of FLC is consumed in the defuzzier module.

III. DEFUZZIFIER MODULE

Coarse to fine search algorithm is implimented using the blocks below

A. Coarse module

In coarse module membership function, span of the membership function and term intervals are considerd using the components like multiplier,adder, latches left and right moments and area are calculated and compared and output of coarse module is obtained.

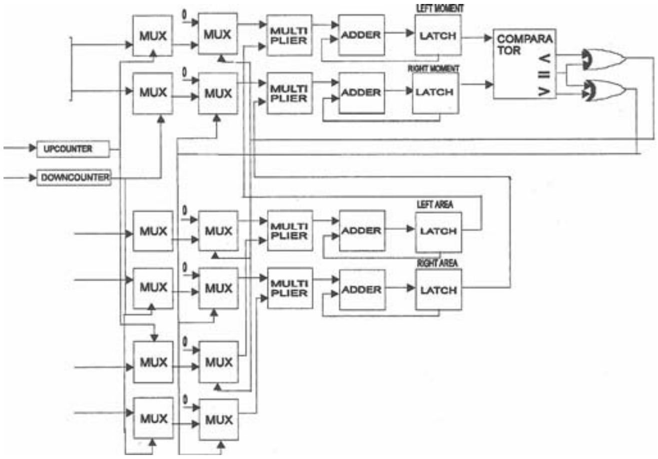


Fig 4 Block Diagram of the Coarse Module

B. Fine module

In Fine module membership function, span of the membership function and term intervals are considerd using the components like multiplier,adder, latches left and right moments and area are calculated and compared and output of Fine module is obtained.

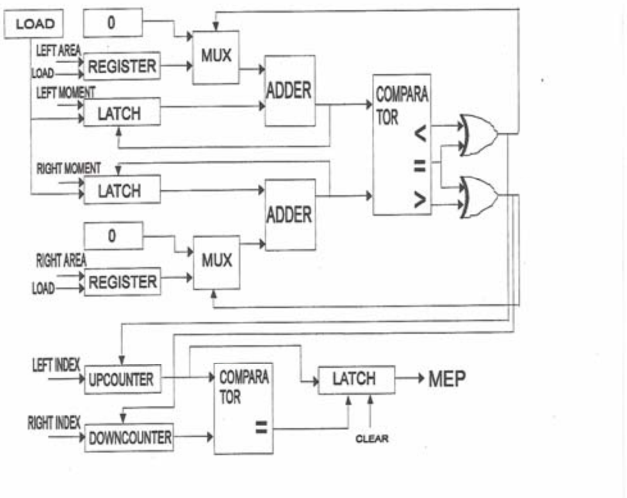
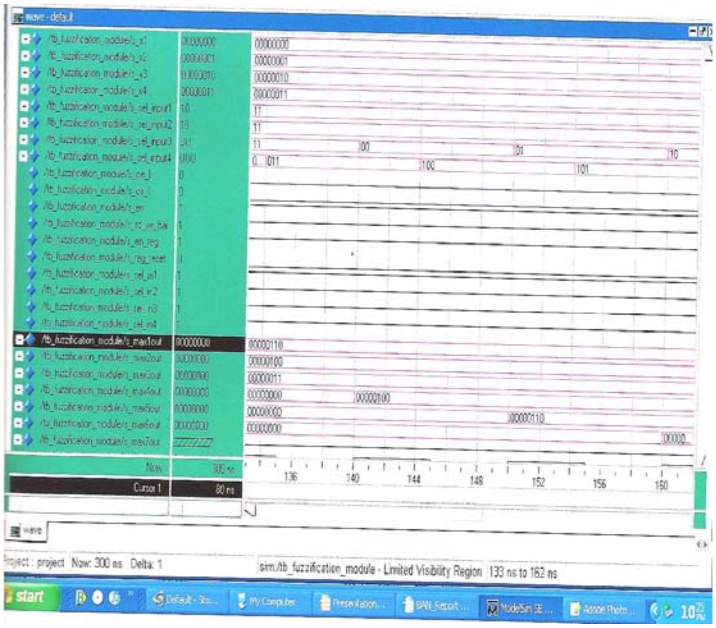


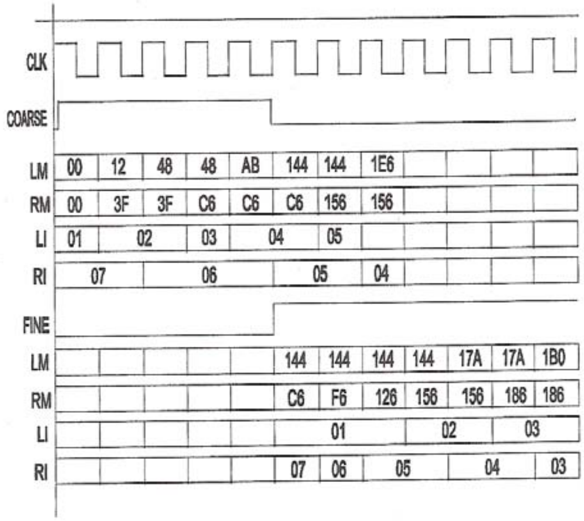
Fig 3 Block Diagram of the Proposed Fine Module

IV. Simulation Results

Results are obtained using VHDL


wave - default
/tb_fuzzification_module/t_x1 00000000
/tb_fuzzification_module/t_x2 00000001
/tb_fuzzification_module/t_x3 00000010
/tb_fuzzification_module/t_x4 00000011
/tb_fuzzification_module/t_sel_input1 10
/tb_fuzzification_module/t_sel_input2 10
/tb_fuzzification_module/t_sel_input3 UU
/tb_fuzzification_module/t_sel_input4 UUU
/tb_fuzzification_module/t_oe_l 0
/tb_fuzzification_module/t_cs_l 0
/tb_fuzzification_module/t_en 1
/tb_fuzzification_module/t_rd_wr_bar 1
/tb_fuzzification_module/t_en_reg 1
/tb_fuzzification_module/t_reg_reset 0
/tb_fuzzification_module/t_sel_in1 1
/tb_fuzzification_module/t_sel_in2 1
/tb_fuzzification_module/t_sel_in3 1
/tb_fuzzification_module/t_sel_in4 1
/tb_fuzzification_module/t_max1out 00000000
/tb_fuzzification_module/t_max2out 00000000
/tb_fuzzification_module/t_max3out 00000000
/tb_fuzzification_module/t_max4out 00000000
/tb_fuzzification_module/t_max5out 00000000
/tb_fuzzification_module/t_max6out 00000000
/tb_fuzzification_module/t_max7out ZZZZZZZZ
Now 300 ns
Cursor 1 80 ns
136
140
144
148
152
156
160
wave
Project : project Now: 300 ns Delta: 1
sim:/tb_fuzzification_module - Limited Visibility Region 133 ns to 162 ns
start
Default - Str...
My Computer
Presentation...
BAN_Report ...
ModelSim SE ...
Adobe Photo...

A . Simulation Results for fuzzifier block:



B. Simulation Results for Defuzzifier block:

V. Conclusion

The design of a typical fuzzy logic controller using VHDL is presented in this paper .This paper proposes a new FLC that is intended to be applied for accurate control and cost effectiveness. The hardware complexity of the tree like structure for the MIN and MAX operation was explosive as the number of input variables increases for that we propose a simple read-modify-write operation instead of the tree like structure at the cost of slower operation speed. Once the basic design of the fuzzy logic control system has been defined, the implementation of the fuzzy logic controller is very straight forward by coding each component of the fuzzy Inference system, MIN module, and MAX module in VHDL according to the design specifications. The availability of different synthesis tools for the programmable logic devices such as FPGA and CPLD have made it easier for the designers to experiment their design capabilities. By simply changing some parameters in the codes and design constraint on the specific synthesis tool, one can experiment with different design circuitry to get the best result in order to satisfy the system requirement.

VI. References

[1] Madni Asad M, Wan L. A, Hansen, Robert K, Vuong Jim B, “Adaptive Fuzzy Logic Based Control System for Rifle Stabilization”, *Proceedings, 1998 World Automation Congress (WAC ’98)*, Anchorage, Alaska, May 10-14, 1998, pp.103-112.

[2] Madni Asad M, Wan L.A, Vuong J. and Vuong P, “Fuzzy Logic Based Smart Controller for a Cryogenic Cooler”, *Proceedings 1996 World Automation Congress (WAC ’96), International Symposium of Soft Computing in Industry (ISSCI)*, May 27-30, 1996, Volume 5, pp.481-488.

[3] Madni Asad M, Wan L. A, Kuo D and Vuong J, “Sensor Based Microcontroller Unit with Built In Fuzzy Inference for an Endometrium Ablator”, *Proceedings 1995, 3rrd European Congress on Intelligent Techniques and Soft Computing (EUFIT ’95)*, August 28-31, 1995, pp. 1621-1624.

[4] Zadeh, L.A., "Fuzzy Sets," *Information and Control*, 8, 338-353, 1965.

[5] Brubaker, David I., "Fuzzy Logic Basics: Intuitive Rules Replace Complex Math," *EDN*, June 18, 1992, pp.111.

[6] Glenn A, “Fundamentals of Fuzzy Logic Part I & II”, *SENSORS*, April 1993.

[7] Earl Cox, *The Fuzzy Systems Handbook* (1994), ISBN 0-12-194270-8

[8] Sibigtroth J, “Implementing Fuzzy Expert Rules in Hardware”, *AI Expert*, April 1992.

[9] Sibigtroth J, “Creating Fuzzy Micros”, *Embedded Systems Programming*, Vol. 4, December 1991.

[10] Kevin Skahill, “VHDL for PROGRAMMABLE LOGIC”, *Cypress Semiconductor*, Addison-Wesley Publishing., Inc. 1996.

VII. Bibliographies

Ms. Meghana Hasamnis, born in Kolhapur in India on May 09, 1975. She graduated from Yeshwantrao Chavan College of Engineering, Nagpur (Maharashtra State) and studied her post graduation in Electronics Engineering from VNIT, Nagpur (Maharashtra State).

Her employment experience includes Seven years of teaching at graduate level. She is having to her credit many International and National Conference papers and received best paper award in one of the conference. Her special fields of interest include Embedded System and Fuzzy Logic.

Ms. Jayu Kalambe, born in Katol in India on November 17, 1977. She graduated from the Priyadarshini College of Engineering, Nagpur (Maharashtra State) and studied her post graduation in Electronics Engineering from VNIT, Nagpur (Maharashtra State).

Her employment experience includes Seven years of teaching at graduate level. She is having to her credit many International and National Conference papers. Her special fields of interest include Neural Network and embedded system.

MsDeepali Shelke, born in Yeotmal in India on November 15, 1979. She graduated from the Government College of Engineering, Amravati (Maharashtra State) and studied her post graduation in Electronics Engineering from YCCE, Nagpur (Maharashtra State).

Her employment experience includes five years of teaching at graduate level. Her special fields of interest include VLSI and embedded system.