# IMPLEMENTATION OF SPEED CONTROL OF MOTOR USING FUZZY LOGIC ON FPGA

**Mohd Ayub Khan[1], Manish Kumar Sharma[2], Sandhya Sharma[3], Sukrati Shukla[4], Shubhangi Singh[5], Shilpi Wakankar[6], Sadhvi Dwivedi[7]**

[1,2]*Department of Electronics and Communication, Anand Engineeering College, Agra, (India)*

[3,4,5,6,7]*Student, Department of Electronics and Communication, Anand Engineering College, Agra, (India)*

## ABSTRACT

*This paper describes the implementation for a basic fuzzy logic controller in Very High Speed Integrated Circuit Hardware Description Language(VHDL). Here we will see the implementation of a fuzzy logic controller through the use of the VHDL code. Use of the hardware description language (HDL)in the application is suitable for being implemented into an Application Specific Integrated Circuit(ASIC) and Field Programmable Gate Array (FPGA). The main advantages of using the HDL approach are rapid prototyping, and allowing usage of powerful synthesis tools such as Xilinx ISE,Synosys,Mentor Graphic, or Cadence to be targeted easily and efficiently. The paper also describes the hardware of our system implemented in the project.*

***Keywords:*** *FuzzyLogic Controller, VHDL, ASIC, FPGA, Implementation*

## I. INTRODUCTION

Fuzzy controller in VHDL was implemented due to the need for an inexpensive hardware implementation of a generic fuzzy controller for use in industrial and commercial applications. A very simple fuzzy controller is used to demonstrate this implementation. In the controller, an external device's information, such as that from a sensor, etc., is converted into an output control signal to drive a device(s) such as motors, actuators etc., via the process of fuzzification, rule evaluation and defuzzification. These processes are all based on a set of membership functions and the details of this process can be found in numerous publications.

## II. PROJECT DESCRIPTION

For the implementation of fan controlling system using fuzzy logic on FPGA kit is done by programming the kit according to software program having temperature and temperature rate as an input and output is the varying voltage which controls the speed of fan . Temperature input to the FPGA kit is given by LM35 temperature sensor but this is in analog form but FPGA works only with digital data . For converting it into digital form ADC0808 is employed ,from which we give temperature input to the FPGA. A software in FPGA process the temperature and temperature rate input according to software program . There are many combination according to which it decides whether the speed of fan is slow , fast and very fast. The temperature states we utilize in program are :

- cold

- Cool
- Mild
- Warm
- Hot

The temperature rate states , we utilize in program are :

- Slow
- Moderate
- Fast

After processing the inputs, the varying voltage in the digital form are available on the4 output pins of FPGA .These digital data is given to DAC0808 for converting this into analog form .This analog output of DAC is given to DC motor through npn transistor for amplification. The DC motor works as a fan . The fan speed varying according to the temperature change in the surrounding sense by LM35.

## III. FUZZY CONTROLLER FOR SPEED CONTROL OF FAN

### 3.1 Define Temperature Membership Functions

#### 3.1.1 Temperature Error (TE)

The membership functions for TE consists of seven fuzzy logic ranges that can be define using the linguistic terms as Cold, Cool, Mild, Warm, and Hot.

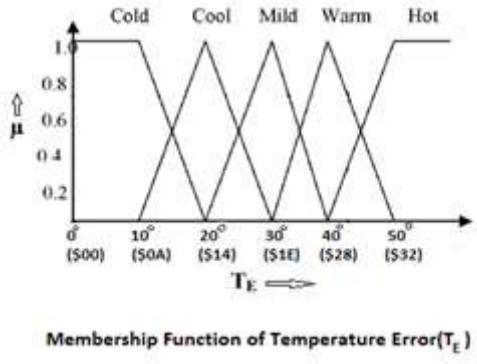



**Figure-1**

```
typete_membership_functions is array(1 to 6)of te_membership;
constantte_mfs:te_membership_functions:=
((term=>cold,point1=>x"00",slope1=>x"19",point2=>x"0a",slope2=>x"19",point3=>x"14"),
(term=>cool,point1=>x"0a",slope1=>x"19"
,point2=>x"14",slope2=>x"19",point3=>x"1e),
(term=>mild,point1=>x"14",slope1=>x"19",
point2=>x"1e",slope2=>x"19",point3=>x"28"),
(term=>warm,point1=>x"1e",slope1=>x"19",
point2=>x"28",slope2=>x"19",point3=>x"32"),
(term=>hot,point1=>x"28",slope1=>x"19",
point2=>x"32",slope2=>x"19",point3=>x"FF"),
(term=>none,point1=>x"FF",slope1=>x"19",
point2=>x"FF",slope2=>x"19",point3=>x"FF"));
```

**2.1.2. Rate of Temperature Change (TE/dt)**

$$rate = \frac{(present\ temperature\ reading) - (previous\ tempereture\ reading)}{time\ between\ temperature\ reading}$$

As usual, the rate of temperature TE/dtchange takes three linguistic terms (Slow, Moderate and Fast) as its Membership Function. The graphical representation of this membership function is shown in Figure-2

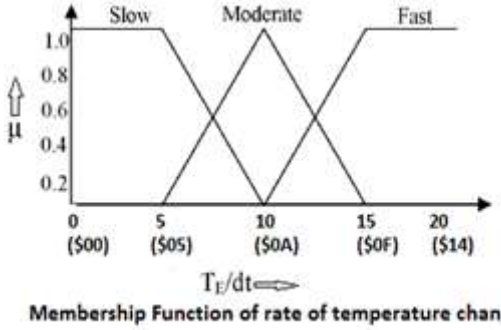



**Figure-2**

```
typetr_membership_functions is array(1 to 4)of tr_membership;
constanttr_mfs:tr_membership_functions:=
((term=>slow,point1=>x"00",slope1=>x"19",
point2=>x"05",slope2=>x"19",point3=>x"0a"),
(term=>moderate,point1=>x"05",slope1=>x"19",point2=>x"0a",slope2=>x"19",point3=>x"0f"),
(term=>fast,point1=>x"0a",slope1=>x"19",
point2=>x"0f",slope2=>x"19",point3=>x"ff"),
(term=>none,point1=>x"FF",slope1=>x"19",
point2=>x"FF",slope2=>x"19",point3=>x"FF"));
```

## IV. AIR CONDITIONING CONTROL OUTPUT (AC)

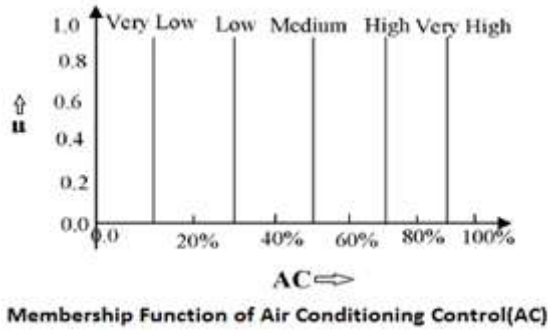
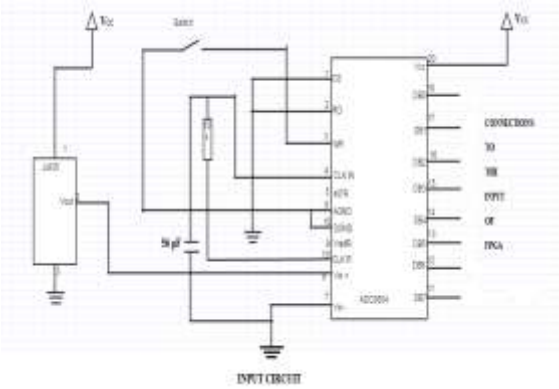



**Figure-3**

```
typesinglentons is array(1 to 5)of std_logic_vector(7 downto 0);
constantvery_low:std_logic_vector(7 downto 0):=x"2A";
constantlow:std_logic_vector(7 downto 0):=x"55";
constantmedium:std_logic_vector(7 downto 0):=x"7F";
constanthigh:std_logic_vector(7 downto 0):=x"A8";
constantvery_high:std_logic_vector(7 downto 0):=x"D2";
signal ac :singlentons:=(very_low,low,medium,high,very_high);
```

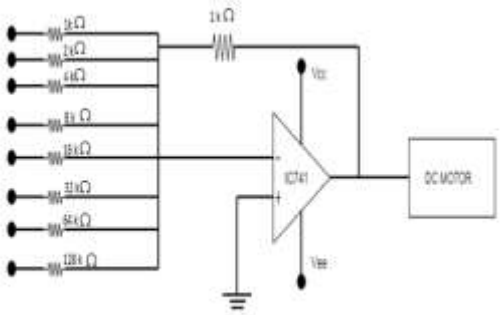
The output membership function of the fuzzy temperature control are singletons, Five different linguistic terms of the singleton output (Very Low, Low, Medium, High and Very High) are used to describe the heater control variation. Figure 3 shows the graphical representation of the singletons of these five variables. The assignment to the output membership function in VHDL is simply declared as an 8-bit constant value.

### 4.1. Input Module



INPUT CIRCUIT

### 4.2. Output Module




OUTPUT CIRCUIT

## V. FUZZY INFERENCE MODULE

In this step, each input value is applied to its membership function to find the fuzzy input value. In an application of two inputs, one having five membership functions and the other having three, there are totally eight degrees of membership functions needed to be calculated. Since a specific input value only intersects at most two membership functions to create the degree of membership function for the corresponding input, so most of them will be zero. As described previously, two points and two slopes are used to define the structure of a membership function. The following pseudo-code illustrates the procedure of this process:

## VI. RULE INFERENCE MODULE

After the degrees of membership function have been determined in the fuzzification stage, the next step is to use linguistic rules to decide what action is needed to be taken in response to a given set of degree of membership function. A technique called min-max inference is used to calculate numerical results of the linguistic rules based on the system input values. The numerical results from this calculation are called fuzzy outputs, since

different rules that can be true for different degrees of membership function variation, thus introducing composite results.

A system with two inputs, one having five linguistic terms and the other having three, and one output with five linguistic terms has a total of 5x3x5 75 different rules can be used to describe the complete fuzzy control strategy. Although there are totally 75 possible rule evaluations in this case, only eight of them are enough to be used in describing how the fuzzy control system should be operated and the remaining rule combinations can be discarded. The reason for choosing only the important rule to be evaluated is to mimic the actual behavior of human thinking. This will allow the ability of a fuzzy control system to approximate human thought using a simple description of the system behavior. Table below summarizes these eight important rules of evaluation for this application.

| IF | | | THEN |
|---|---|---|---|
| Temperature error (Te) | Logical operator | Rate change(Te/dt) | Required o/p |
| 1. cold | and | fast | very high |
| 2. cold | and | slow | high |
| 3. cool | and | moderate | medium |
| 4. mild | and | slow | medium |
| 5. warm | and | slow | low |
| 6. warm | and | moderate | low |
| 7. hot | and | slow | Very low |
| 8. hot | and | fast | Very low |

The IF part of the rule gets the degree of the input membership function and the THEN part has the degree of membership function that is determined by the result of the rule evaluation. Since the input grades are connected by a logical "AND" operator, one input is dependent upon the other, selecting the minimum value will satisfy both conditions. Therefore, taking the minimum of the two variables will be used as the output grade of membership function in each rule.

## VII. DEFUZZIFICATION MODULE

After the output grade of each rule has been determined, the next step is to combine all the output grades into a single value that can be used to control the heater operation. In the defuzzifier process, Center-of-Gravity (COG) deffuzification technique will be used to obtain the final system output. In this application, singleton variables are used as the output membership functions. Defuzzification involves taking the weighted average of all the fuzzy outputs. Each fuzzy output is multiplied by its corresponding singleton value, then the sum of these products divided by the sum of all the fuzzy outputs to obtain the final single output variable.

## VIII. RESULT

| | |
|---|---|
| **Membership1** | cold(00,19,0A,19,14)<br>cool(0A,19,14,19,1E)<br>mild(14,19,1E,19,28)<br>warm(1E,19,28,19,32)<br>hot(28,19,32,19,FF)<br>none(FF,19,FF,19,FF) |
| **Membership2** | slow(00,19,05,19,0A)<br>moderate(05,19,0A,19,0F)<br>fast(0A,19,0F,19,FF)<br>none(FF,19,FF,19,FF) |

| **Input1 (in hex)** (cold,cool,mild, Warm,hot) | **Input2 (in hex)** (slow,moderate, Fast) | **Output (in hex)** (very low,low, Medium,high, Very high) |
|---|---|---|
| 09 | 04 | 55 |
| 32 | 0F | D2 |
| 1E | 05 | 7F |
| 28 | 0A | A8 |
| 17 | 0E | 7F |
| 14 | 08 | 7F |
| 29 | 07 | A8 |

### 8.1 Waveform

**8.2 Final Working Model**

## IX. CONCLUSION

The design of a typical fuzzy logic controller using VHDL is presented in this paper. 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 in VHDL according to the design specifications. The availability of different synthesis tools for the programmable logic devices such as FPGA 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.

## 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, 3d 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,

[6]. 1992, pp.111.

[7]. Glenn A, "Fundamentals of Fuzzy Logic Part I & II", SENSORS, April 1993.

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

[9]. Sibigtroth J, "Implementing Fuzzy Expert Rules in Hardware", AI Expert, April 1992.

[10]. Sibigtroth J, "Creating Fuzzy Micros", Embedded Systems Programming, Vol. 4, December 1991.

[11]. Sibigtroth J, "Fuzzy Logic for Embedded Microcontrollers", Circuit Cellar, March 1995.

[12]. Kevin Skahill, "VHDL for PROGRAMMABLE LOGIC", Cypress Semiconductor, Addison-Wesley

[13]. Publishing., Inc. 1996.