Design and Synthesis of Temperature Controller Using Fuzzy for Industrial Application

Md. Shabiul Islam[1], Mukter Zaman[1], M.S. Bhuyan[1], Masuri Othman[2]
[1]Faculty of Engineering, [2]Dept. of Electrical and System Engineering
[1]Multimedia University, [2]University Kebangsaan Malaysia
[1]63100 Cyberjaya, Selangor, [2]43600 Bangi, Selangor Darul Ehsan
Malaysia

Abstract: - This paper describes the VHDL modeling of temperature controller based on fuzzy logic intended for industrial application. The system is built of four major modules called fuzzification, inference, implication and defuzzification. The composition method selected for the fuzzy model is the Max-Min composition while the Mamdani Min operator was chosen as the implication method. Each module is modeled individually using behavioral VHDL and combined using structural VHDL. Following a successful VHDL coding, timing analysis was done verifying the functionalities of the algorithm with all simulated conditions. Simulation results show that the model has been tested successfully. The verified model of VHDL has synthesized using synthesis tool to get gate levels. The inferred maximum operating frequency is 5MHz with a critical path of 199.3ns.

Key-Words: - VHDL prototyping, Temperature controller, Fuzzy logic, Synthesis

1 Introduction

The effective and efficient control of the surrounding environment is crucial to many technical processes. Ranging from IC fabrication to the production of chemical solutions, any changes in the ambient parameters can have a drastic effect in the outcome of a process, at the very least lowering the yield or quality of the product. Among the crucial parameters that merits close supervision is the temperature of the environment. As such, temperature control is critical to the quality, appearance and consumer acceptance of a manufacturer's products.

The processes that requiring temperature control has various unfavorable characteristics including non-linearity, dead zone time, external disturbances and so on. Conventional approximations do not produce satisfactory temperature controls for controlling complex processes, which is usually the case in the industry because they suffer from various drawbacks such as slow stabilization, overshooting and overall slow response.

A fuzzy system improves the relative performance of a temperature control process with respect to the conventional scheme. It compensates non-linear errors, accelerates the response and reduces the steady-state error. The fuzzy logic controller is also able to bring keep the temperature constant at the desired value regardless of changes in the load or environment. This project attempts to enable a fuzzy-based control of temperature employing VHDL as a mean of improving upon conventional methods.

Several works had been done in this area. Zhiqiang et al. [1] had developed a closed loop control system incorporating fuzzy logic for a class of industrial temperature control problems employing a unique FLC structure with an efficient realization and a small rule. Their works demonstrated in both software simulation and hardware test in an industrial setting that the fuzzy logic control is much more capable than the current temperature controllers. This includes compensating for thermo mass changes in the system, dealing with unknown and variable delays and operating at very different temperature setpoints without retuning. Thyagarajan et al. [2] presented four control schemes designed using advanced techniques for regulating the temperature of the Air Heat Plant. Four control schemes namely, PID, fuzzy logic control (FLC), FLC using genetic algorithms (FLC-GA) and neuro fuzzy control (NFC) are developed and presented. All these schemes are evaluated with respect to set-point tracking using performance indices. Their works highlighted superiority of FLC over PID, FLC-GA over FLC and NFC over other schemes. There are also two more works utilizing fuzzy logic to control temperature for specific applications which will not discussed here [3]-[4].

In this paper, the control system will be implemented using VHDL, aiming for FPGA implementation in future. There are several advantages for this approach. FPGA implementation allows immediate manufacturing realization and negligible prototype costs. FPGA also offer affordable and practical solutions to custom applications as well as allowing new vista in designing reconfigurable digital systems. In testing, FPGA allow designers the freedom to redesign portions of their circuit for optimization, not to mention performing additional iterations to improve their design [5].

One major benefit of hardware implementation over software is the simulation speed. Hardware-based simulation allows the simulation process to take advantage of the parallel execution of instructions. Other advantages of programmable hardware are the ability to perform bit-level operations on "unusual", i.e., not powers of two, word lengths and the possibility to allocate only a certain number of bits to represent internal variables. Hence, it can be seen that the FPGA combines the flexibility of software and the speed of hardware [6].

2 Design Overview

This project aims to implement a fuzzy logic temperature control in FPGA chip. At present, we are able to model the system on VHDL. In this system, there are two inputs into the fuzzy model namely Error which is the error signal and CError which is a signal representing the rate of change in error after a fixed period. Each input consists of 4 triangular membership functions over a normalized range from 0 to 1. Input Fuzzy Variable Error and CError are illustrated in Figures 1 and 2 respectively. The 4 fuzzy variables for both inputs are termed ZE (Zero), PS (Positive Small), PM (Positive Medium) and PL (Positive Large).

Based on these 2 inputs, the fuzzy logic model aims to determine the amplitude of the voltage signal (for example, from 0 to 5V) that is necessary to be sent to the heater in order to maintain a constant temperature in the industrial process. This is provided by an output signal from the fuzzy model termed Output, with a normalised range of 0 to 1. Similar to the inputs, the Output signal has 4 triangular membership functions spaced over this range, as can be seen from Figure 3, called ZE, PS, PM and PL as well.

The rule-base utilised in this fuzzy model consists of 16 rules where the output produced is based on various combinations of the two fuzzy inputs. Figure 4 shows the complete fuzzy rules of this project [7]. Here, the connective ELSE term is interpreted as an *intersection* (i.e., an OR operation) while the connective AND is given a *minimum* interpretation.

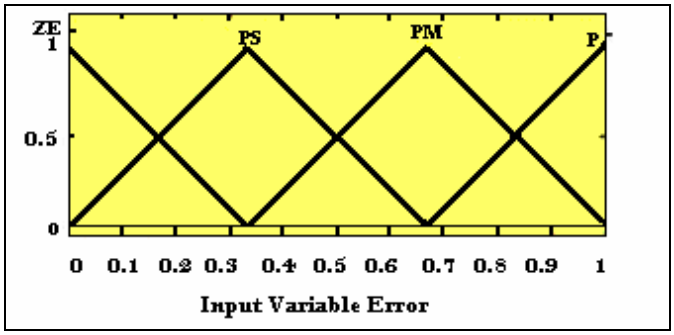

Fig.1 Membership Functions of Input Fuzzy Variable Error

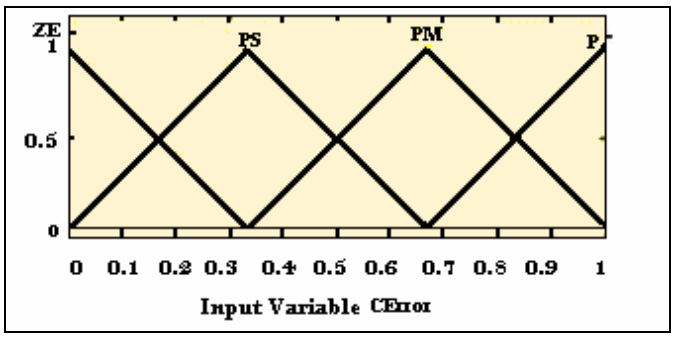

Fig.2 Membership Functions of Input Fuzzy Variable Cerror

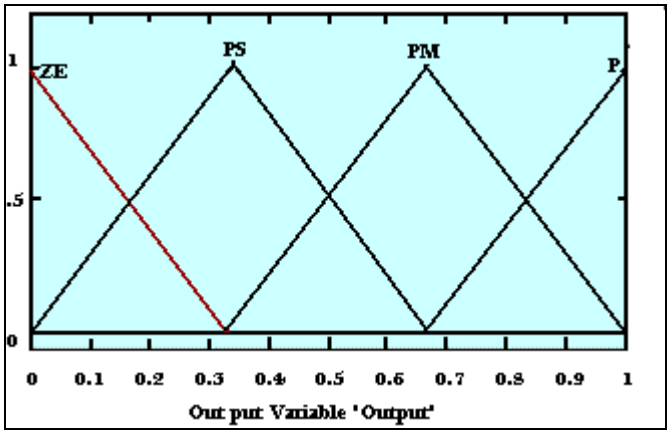

Fig.3 Membership Functions of Output Fuzzy Variable Output

The composition method selected for our fuzzy model is the Max-Min composition while the Mamdani Min operator was chosen as the implication method. The Max-Min composition is

favoured as it is used extensively in control applications of fuzzy logic [8]. The Mamdani Min implication was chosen for similar reasons.

The defuzzification method used in this model is the Centroid Centre of Area (COA) method, as it is a well known and commonly used method . The COA method takes into account the area of the resultant membership function as a whole and favours "central values" in the universe of discourse (or region).

1. IF ERROR is ZE AND CERROR is ZE THEN OUTPUT is ZE ELSE
2. IF ERROR is ZE AND CERROR is PS THEN OUTPUT is PS ELSE
3. IF ERROR is ZE AND CERROR is PM THEN OUTPUT is PM ELSE
4. IF ERROR is ZE AND CERROR is PL THEN OUTPUT is PL ELSE
5. IF ERROR is PS AND CERROR is ZE THEN OUTPUT is PS ELSE
6. IF ERROR is PS AND CERROR is PS THEN OUTPUT is PS ELSE
7. IF ERROR is PS AND CERROR is PM THEN OUTPUT is PM ELSE
8. IF ERROR is PS AND CERROR is PL THEN OUTPUT is PL ELSE
9. IF ERROR is PM AND CERROR is ZE THEN OUTPUT is PM ELSE
10. IF ERROR is PM AND CERROR is PS THEN OUTPUT is PM ELSE
11. IF ERROR is PM AND CERROR is PM THEN OUTPUT is PM ELSE
12. IF ERROR is PM AND CERROR is PL THEN OUTPUT is PL ELSE
13. IF ERROR is PL AND CERROR is ZE THEN OUTPUT is PL ELSE
14. IF ERROR is PL AND CERROR is PS THEN OUTPUT is PL ELSE
15. IF ERROR is PL AND CERROR is PM THEN OUTPUT is PL ELSE
16. IF ERROR is PL AND CERROR is PL THEN OUTPUT is PL

where:

ZE ? Zero; PS ? Positive Small; PM ? Positive Medium & PL ? Positive Large

Fig.4 Fuzzy IF/THEN rules used

Using MATLAB Package, the surface of the 16 fuzzy rules in the system can be seen in Figure 5. It can be observed that the surface changes in a gradual and smooth manner as either/both fuzzy variables Error or CError increases from 0 to 1. The smooth change in the surface indicates that the rules as a whole are consistent and hence, an accurate output should be produced by the system.

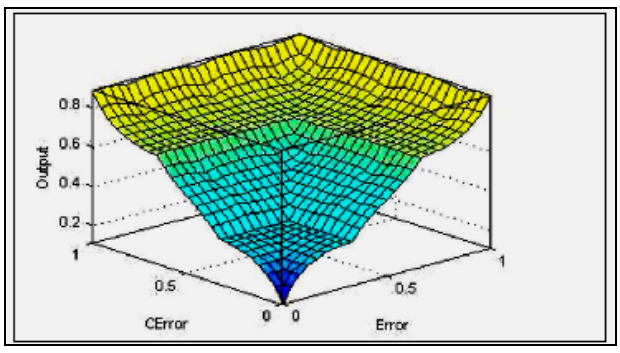

Fig.5 Surface of Fuzzy Rules

Notice that both inputs and the output range have been normalised to within 0 to 1. This gives the system a measure of flexibility in being adaptable to various input/output parameters, through the use of appropriate simple conversion circuits. Hence, the model is able to accommodate different processes and environments without major changes within the algorithm.

3 VHDL Implementation

In the VHDL implementation, the temperature controller system is built of four major modules called fuzzification, inference, implication and defuzzification. Each module is modeled ndividually using behavioral VHDL and combined using structural VHDL[9]- [10]. Figure 6 shows the overall VHDL model of the system.

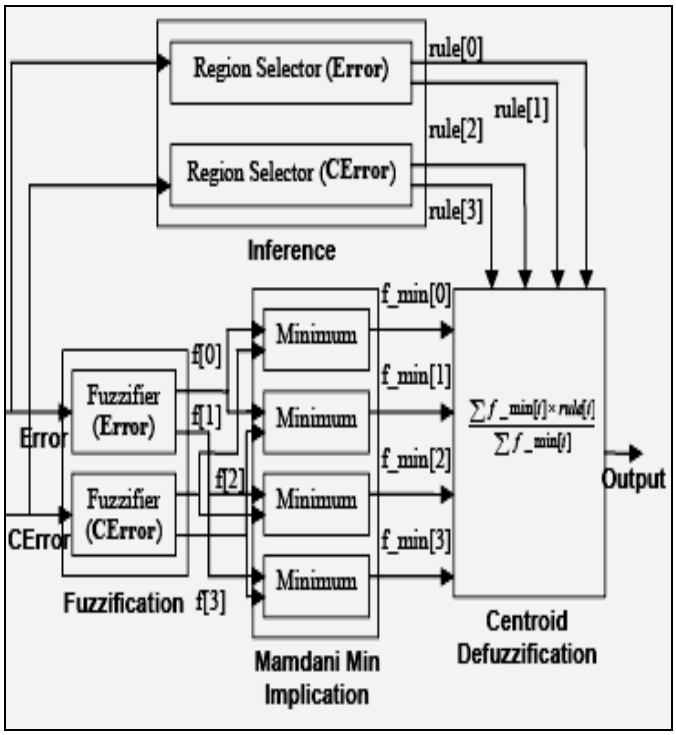

Fig.6 Overall model of Fuzzy Temperature Control System

The overall model accepts two 8 bit unsigned inputs, termed Err and C_error (representing input fuzzy variables Error and CError) and has one 8 bit unsigned output, naturally called Output (corresponding to the Output variable). Err and C_error are mapped to the inputs of components Fuzzification and Inference; the inputs of the two components are labelled as In1 and In2 respectively. Output on the other hand, is mapped to the output of the Defuzzification module, termed Out_put.

3.1 Fuzzification Model

The fuzzification module can be divided into two similar parts; both serving the same function. The module accepts two crisp (i.e. real worldsignals) and produces 4 fuzzified signals that will be sent to the implication module, 2 fuzzy values for each input. The two crisp signals are the Error and CError signal respectively, each normalised to within the range 0 to 1.

For each input signal's universe of discourse, there are 4 triangular membership functions. A triangular membership function was chosen as it can be easily modelled using the elementary line equation; Y = mX + c, where Y represents the fuzzy value, m represents the gradient of the membership function, X is the crisp input and c symbolises the intersection the membership curve makes with the Y-axis.

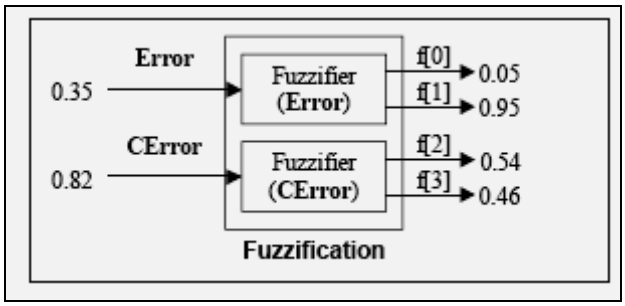

Fig.7 Fuzzification Example

An illustration in Figure 7 is provided to ease understanding. The Error signal is given a value of 0.35 and CError is assigned to 0.82. These values will be used in the following sections for additional illustrations in order to be consistent. It can be seen that Error=0.35 intersects with fuzzy variables PS and PM, where PS is taken as the first fuzzy variable f[0] and PM is called the second fuzzy variable, f[1]. Based on the membership functions, f[0] = 0.05 and f[1] = 0.95. Similarly, the value of CError=0.82 intersects with the fuzzy membership functions PM and PL, where PM is assigned to f[2] = 0.54 and f[3] = 0.46 corresponds to PL.

3.2 Inference Model

The inference module selects the rules to be fired based on the 2 crisp input values of Error and CError. The rules are selected in such a way that only two fuzzy variables are active at any given time, whereby each fuzzy variable will result in the firing of two rules resulting in a total of 4 fired rules from the module. This process is achieved by dividing the universe of discourse into 3 regions; with each region containing only two fuzzy variables. As the universe of discourse for both Error and CError are the same, the division process is simply duplicated. From Table 1, it can be seen that Error = 0.35 lies in Region B while CError = 0.82 lies in Region C.

The inference module functions by selecting the appropriate rules to be fired based on the fuzzy variables that are chosen according to the regions that the variables fall in. For example, if Region B of Error containing fuzzy variables PS and PM; and Region C of CError holding fuzzy variables PM and PL is chosen, then rules 7, 8, 11 and 12 will be selected.

Following normal practices, the Output value is represented using fuzzy singleton sets in place of membership functions like those of the two inputs. The use of singleton values allows faster inferencing as well speeding up the defuzzification process. The downside of singleton values is that a certain amount of accuracy is sacrificed as each output values now represents a range of input values. Figure 8 shows the four singleton values chosen to represent the fuzzy output variable. Hence, when rules 7, 8, 11 and 12 are fired, the following singleton values are assigned to the outputs of the module.

Table 1 - Regions of Error and Cerror

Input	Region	Range
Error / CError	A	0.000 to 0.3334
	B	0.3334 to 0.6666
	C	0.6666 to 1.0000

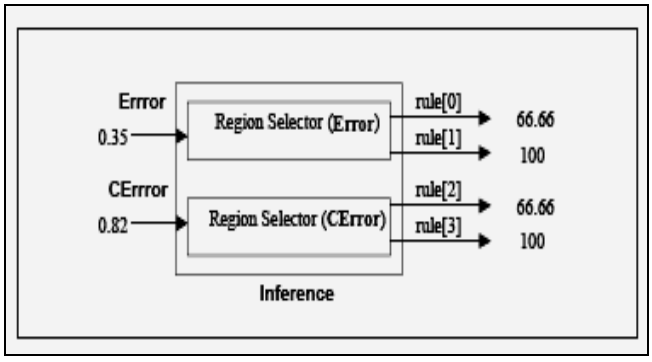

Fig.8 Interface Example

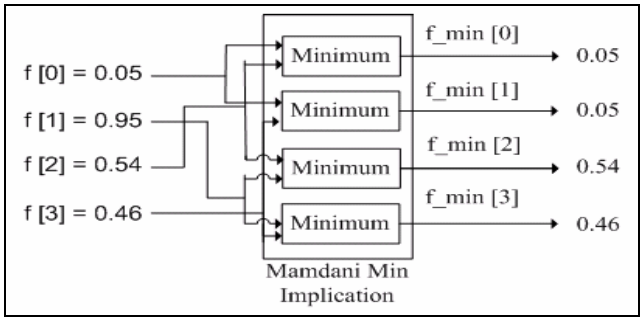

Fig.9 Implication Example

3.3 Implication Model

This module receives the 4 fuzzy variables from the Fuzzification Module and performs the Mamdani Min implication operation on the combination of these four variables. The Mamdani Min operation simply takes the minimum of two values, sometimes represented by the connective AND. A Max-Min composition is also carried out, as the fuzzy variables due to Error, namely f[0] and f[1], will be compared to CError's fuzzy variables of f[2] and f[3]. The four resulting output of the implication operation will then be fed into the Defuzzification module. Continuing the previous examples, the values of f[0], f[1], f[2] and f[3] respectively are 0.05, 0.95, 0.54 and 0.46. The implication and composition functions are illustrated in Figure 9.

3.4 Defuzzification Model

This module accepts four inputs each from the Implication module (i.e. f_min[0-3]) and the Inference module (i.e. rule[0-3]) and produces the defuzzified/crisp output signal, Output used to control the actuators of the control system. The defuzzification scheme chosen was the COA Centroid method as explain previously, expressed mathematically in equation 1.

$$\frac{\sum f\_\min[i] \times rule[i]}{\sum f\_\min[i]} \qquad \ldots \quad \ldots \quad (1)$$

There are a few arithmetic operations performed within the module, as can be seen from the equation 1 which are 2 summation, 1 division and 4 multiplication operations. The output of the module is in the form of a percentage value, which can be easily converted into a normalised form. Using the values from the previous example, the output of the defuzzification module = 82.113%

4 Result and Discussion

Upon completion of coding and the successful compilation of the codes, a simulation of the VHDL model is performed. This simulation is essential in determining whether the model is functioning properly and also to determine the deviation or tolerance parameters if they exist. The simulation as the coding and compilation is carried out using Aldec's Active-HDL version 3.5. For the simulation, a set of stimuli is needed as sample inputs. In this system the stimuli are set of functional vectors that changes with at fixed time duration.

The stimuli is added to the test bench coded generated by Active-HDL. In this paper, a specific test case is shown in Table 2 where the primary output, Output is the outputs of the Fuzzification module, f[0], f[1], f[2] and f[3]. The results also show the rules fired from the Inference module, namely rule[0], rule[1], rule[2] and rule[3].

The waveform of the simulation is shown in Figure 10. The waveform shows the 2 fuzzy inputs Error and Cerror, the fuzzy variable Output, the four outputs of the Fuzzification module labelled as S05, S06, S07 and S08; and the outputs of the Inference module labelled as S01, S02, S03 and S04. The results in Figure 10 shows the values of the inputs and the corresponding output in hex form at the various instances determined by the stimuli in the test bench. A manual check of these results shows that the output is correct. Thus, the VHDL model has been verified to be functioning as expected. Various other test cases representing different type of conditions had also been tested. The speed of the entire system is clocked at 5MHz with the critical path of 199.3ns.

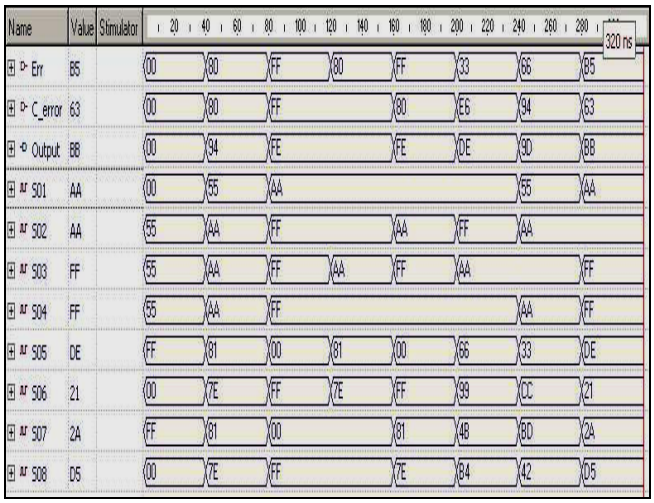

Fig.10 Waveform of VHDL Model Simulation

Table 2 - Test Outputs of VHDL Fuzzy Model

Error	CError	Output	f(0)	f(1)	f(2)	f(3)	Rule [0]	rule [1]	rule [2]	rule [3]
0	0	0	1	0	1	0	0	33.33	33.33	33.33
0.5	0.5	.5504	.5059	.4241	.5059	.4941	33.33	66.66	66.66	66.66
1	1	.9961	0	1	0	1	66.66	100	100	100
0.5	1	.9961	.5059	.4241	0	1	66.66	100	66.66	100
1	0.5	.9961	0	1	.5059	.4941	66.66	66.66	100	100
0.2	0.9	.8708	.4	.6	.2941	.7059	66.66	100	66.66	100
0.4	0.88	.6157	.2	.8	.7412	.2888	33.33	66.66	66.66	66.66
0.71	0.39	.7333	.8706	.1294	.1647	.8363	66.66	66.66	100	100

5 Synthesis

Synthesis is the process of transforming one representation in the design abstraction hierarchy to another representation. Synthesis process has performed using synplify tools for synthesizing the compiled VHDL design codes into gate level schematics. The synthesis tool is also used to optimize the gate level design for area by applying specified options. It initially processes the VHDL building blocks such as multiplier, registers, gates and flip-flops etc, for which it can determine whether logic blocks can be shared between the building blocks function for efficiency performance. While synthesizing the design with the synthesis tool, HDL library browser was used to synthesize the design in a hierarchical manner.

In this step, the VHDL codes are synthesized for converting into Register Transfer level (RTL) view of Fuzzy Temperature Controller architecture. The Technology mapping has chosen in this project from Altera's FLEK10K70 with RC240 package and a speed grade of -4. Then the technology view of the various modules for Fuzzy Temperature Controller chip has been carried out. As an example, the flattened technology view of overall Fuzzy Temperature Control System is given in the Figure 11.

Upon successful synthesis step, another simulation process is carried out to ensure the functionality of the synthesized codes prior to the final stage, namely the FPGA realization. The synthesized schematic is also simulated to ensure the synthesized design functions the same way as the validated VHDL model for VLSI implementation.

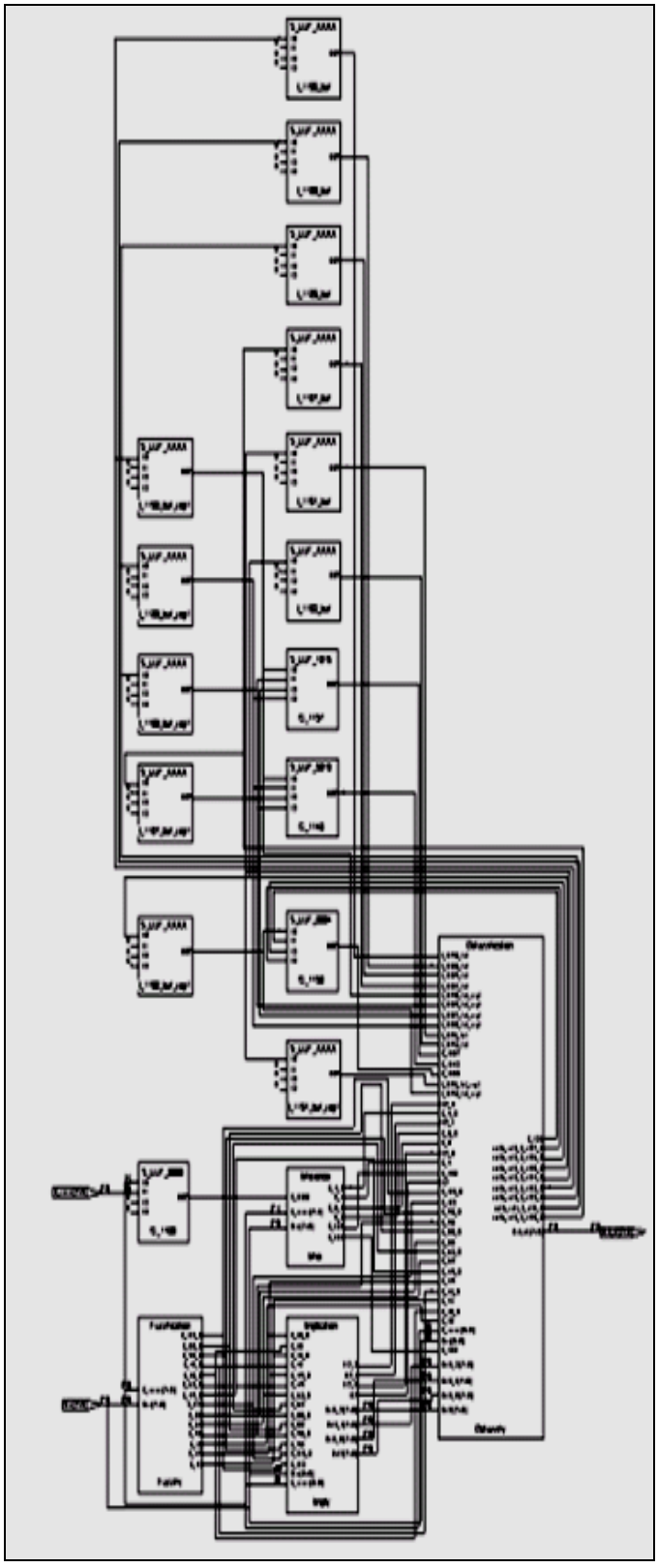

Figure 11- Flattened Technology View of Overall Fuzzy Temperature Control System.

Upon completion of coding and the successful compilation of the codes, a simulation of the VHDL model is performed.

6 Application

Temperature control is widely used in various processes. These processes, no matter it is a process of large industrial plant, or a process in home appliance, share several unfavourable features such as non-linearness, interference, dead time, and external disturbance, etc.

Conventional control approaches usually cannot achieve satisfactory results for this kind of processes. Besides this all other processes that requiring temperature control has various unfavorable characteristics including non-linearity, dead zone time, external disturbances and so on. Currently used conventional approximations do not produce satisfactory temperature controls for controlling complex processes, which is usually the case in the industry because they suffer from various drawbacks such as slow stabilization, overshooting and overall slow response.

This fuzzy system based temperature controller can be applied in any kind of environment by which we can get an improvement of relative performance with respect to the conventional scheme. It compensates non-linear errors, accelerates the response and reduces the steady-state error. The FLC is also able to bring keep the temperature constant at the desired value regardless of changes in the load or environment. Thus we can experience a great solve of the overshooting problem. It also can able to improve the slow stabilization problem. Thus it application can be implemented on almost all scale industry.

7 Conclusion

A fuzzy logic temperature controller has been designed with an industrial application in mind. The system has been coded, compiled and simulated in VHDL using EDA tools, specifically Aldec Active-HDL 3.5. The hardware implementation demonstrated complete, correct functionality and met all the initial system requirements. At present the system inferred maximum operating frequency is 5MHz with a critical path of 199.3ns. Further research will then be carried out for hardware implementation using FPGA. This could take advantage of the high speeds achievable using hardware, and as a result would be a beneficial and economic investment for designs requiring fuzzy logic.

References:

[1] Zhiqiang, Gao., Trautzsch, T.A., and Dawson, J.G., A stable self-tuning fuzzy logic control system for industrial temperature regulation, *IEEE Industry Applications Conference*, Vol.2, 2000, pp. 1232 – 1240.

[2] Thyagarajan, T., Shanmugam, J., Ponnavaikko, M., and Rao, P.G., Advanced control schemes for temperature regulation of air heat plant, *IEEE International Fuzzy Systems Conference Proceeding*, Vol.2, 1999, pp. 767 – 772.

[3] Ayala, I.L., and Solis, I.J. 1991. IEEE Transactions on Industry Applications. Volume: 27 , Issue: 1, Pages:108 – 111

[4] Coeyman, B., and Bowles, J.B., Fuzzy logic applied to reboiler temperature control, *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems*, Vol.1, 1996, pp.511 – 516.

[5] K.T. Tho, K.H. Yeow, F. Mohd-Yasin, M.S. Sulaiman, and M.I. Reaz, VHDL Modeling of Boolean Function Classification Schemes for Lossless Data Compression, *WSEAS Transactions on Computers*, Vol.3, No.2, 2004, pp. 365-368.

[6] Brown, S.D., Francis, R.J., Rose, J., and Vranesic, Z.G.,*Field-Programmable Gate Arrays*, Kluwer Academic Publishers,1996

[7] Mohd-Yasin, A. Tio, M.S. Islam, M.I. Reaz and M.S. Sulaiman, VHDL Prototyping of Temperature Controller Based on Fuzzy Logic for Industrial Application, *2nd International Conference on Artificial Intelligence in Engineering and Technology, Sabah, Malaysia*, 2004.

[8]Tsoukalas, L.H., Uhrig, R.E., *Fuzzy and Neural Approaches in Engineering*. Wiley-Interscience, 1997.

[9] Ashenden, P.J., *The Designer's Guide to VHDL*. San Francisco:Morgan Kaufmann.

[10] Hunter, R.D.M. *Introduction to VHDL*.1996