Design of Optimized Fuzzy Logic Controller for Area Minimisation and its FPGA Implementation

G.Sakthivel Dr.T.S.Anandhi Dr.S.P.Natarajan
Dept of E&I . Annamalai University, Chidambaram, INDIA

Abstract:
Area optimization is one of the important problems in reconfigurable systems. A Field-programmable gate array (FPGA) based optimised Fuzzy Logic Controller(FLC) has been developed for speed control of DC Motor by exploiting the basic features of Fuzzy logic. The aim of the proposed scheme is to reduce the area, as compared with conventional Fuzzy Logic Controller. The area of a design refers to sum of the area space of the circuit components in FPGA. To implement optimised Fuzzy Logic Controller certain modifications have to made, such that its functionality is identical to original design but requires a smaller area. Real time implementation of Optimised FLC and conventional FLC are made on Spartan-3A DSP FPGA (XC3SD1800A) reconfigurable FPGA for the speed control of DC motor. Results shows that area needed to implement the proposed scheme is reduce drastically and while maintaining the same performance.

Key words:
Field programmable gate array, VHDL, FLC, optimization

1. INTRODUCTION

Fuzzy Logic Controllers have been widely applied to both consumer products and industrial process controls. In particular, FLC's are very effective techniques for complicated and imprecise processes for which either no mathematical model exists or the mathematical model is severely nonlinear, because FLC's can easily approximate a human expert's control behaviours that work fine in such ill-defined environments [1]. Most of the fuzzy logic applications with the physical systems require a real-time operation to interface high speed constraints. The simple and usual way to implement these systems is to realize it as a software program on general purpose computers, these ways cannot be considered as a suitable design solution. Higher density programmable logic device such as FPGA can be used to integrate large amounts of logic in a single IC.

FPGA becomes one of the most successful of technologies for developing the systems which require a real time operation. For these systems [3], [4], [6], [7], FPGAs are more sufficient than the simple way because they can cover a much wider range of operating conditions. FPGA places fixed logic cells on the wafer, and the FPGA designer constructs more Complex functions from these cells [5].

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 systems. Numbers of these works were specialized in control application, and were aim to get better control responses, [8], [9]. FPGA are two dimensional arrays of logic blocks and flip-flops with an electrically programmable interconnection between logic blocks. The interconnections consist of electrically programmable switches which is why FPGA differs from Custom Integrated Chips, as Custom Integrated Chip is programmed using integrated circuit fabrication technology to form metal interconnections between logic blocks. In an FPGA logic blocks are implemented using multiple level low fan in gates, which gives it a more compact design compared to an implementation with two-level AND-OR logic.

FPGA provides its user a way to configure: The intersection between the logic blocks and the function of each logic block. Logic block of an FPGA can be configured in such a way that it can provide functionality as simple as that of transistor or as complex as that of a microprocessor. It can used to implement different combinations of combinational and sequential logic functions.

In this work, optimised fuzzy logic controller which reduces the area is designed by making certain modifications on conventional FLC, such that its functionality is identical to original design. The resources utilised used for optimised FLC is tabulated based on number of slice flip flop, number of 4 input LUTS, number of slices, number of bonded IOBS, number of bufgmuxs, number of gates.

This paper is organized as follows. Section 2 explains briefly the methodology of proposed. Section 3 explains a hardware implementation on the reconfigurable FPGA system Section 4 briefly explains architecture of the proposed optimised FLC. Section5 explains a overall experimental set up DC motor. Section6 will give a discussion on the synthesis results of the fuzzy logic controller for speed control of DC motor . Finally, a conclusion is drawn in Section 7.

Manuscript received August 5, 2010
Manuscript revised August 20, 2010

2. Optimized Fuzzy Logic Controller

In order to implement the designed fuzzy logic controller in FPGA there are still other issues to consider, particularly regarding *area optimisation*. This section looks at the issue of *area optimisation*. Modifications made here optimise the design such that it is functionally identical to the original design but requires a smaller area.

2.1 Fuzzifier optimisation

In the fuzzification process of the conventional FLC design, the two inputs, x_1 and x_2, are processed using two separate fuzzification blocks. Since both of the inputs are fuzzified in exactly the same manner, it is possible to create just one fuzzification block to be shared between the two input variables. This reduces the area.
To achieve this, there is also the need for memory elements to store the result of the first computation while the second set of data is being computed. At the end of the second computation, both sets of results are released simultaneously. In order to reduce the area size further, another method of optimization is devised. The fuzzification block has five outputs, one for each fuzzy value defined in the inputs universe of discourse. However, the fuzzification process entails that, for any single crisp value of the input x_i, only two adjacent fuzzy values are significant (with non-zero membership values). By ignoring the insignificant fuzzy values, the number of output signals can also be reduced from five to two. The possible combinations of significant fuzzy values for an arbitrary input are:
B_i^1 and B_i^2, B_i^2 and B_i^3, B_i^3 and B_i^4, B_i^4 and B_i^5
It is found that using just three variables, ADRi, Bi_A and Bi_B, all the combinations can be sufficiently represented for any value of *xi* as shown by the following table:

ADR_i	B_i A		B_i B	
00	B_i^1	NB	B_i^2	N
01	B_i^2	N	B_i^3	Z
10	B_i^3	Z	B_i^4	P
11	B_i^4	P	B_i^5	PB

Figure 1 illustrates how these conditions correspond with the universe of discourse. Management of the input and output signals is mainly controlled by a status bit, **R_sig**.

When **R_sig** is '1', the input **x1** is fuzzified but the output values remain unchanged.When **R_sig** is '0', the input **x2** is fuzzified and the new values of all the outputs are assigned. The variables **temp**, **temp_A** and **temp_B** represent the temporary registers used to store the fuzzy values of **x1** while the fuzzy values of **x2** are being computed and all the fuzzy values are simultaneously released.

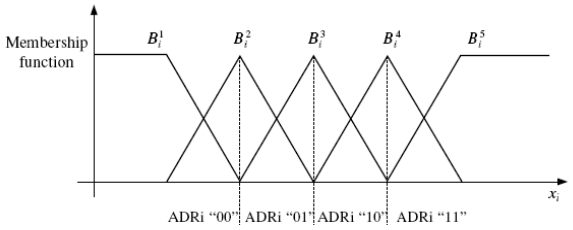

Fig 1 Input fuzzy values

2.2 'Mini' FAM tables

The FAM table of the conventional FLC is shown in Table 1. It was mentioned that the inference of the fuzzy rules is achieved using Mamdani's inference technique and inference engine triggers all 25 rules during every calculation.

ΔE / E	**NB** (B_i^1)	**N** (B_i^2)	**Z** (B_i^3)	**P** (B_i^4)	**PB** (B_i^5)
NB (B_i^1)	NVB R^1	NB R^2	N R^3	NS R^4	Z R^5
N (B_i^2)	NB R^6	N R^7	NS R^8	Z R^9	PS R^{10}
Z (B_i^3)	N R^{11}	NS R^{12}	Z R^{13}	PS R^{14}	P R^{15}
P (B_i^4)	NS R^{16}	ZS R^{17}	PS R^{18}	P R^{19}	PB R^{20}
PB (B_i^5)	Z R^{21}	PS R^{22}	P R^{23}	PB R^{24}	PVB R^{25}

Table 1 FAM table of the conventional FLC

This section describes an algorithm which is developed to reduce the amount of computation required by focusing only on the relevant rules and ignoring those which are irrelevant to the conditions only.
In the conventional design, for every set of inputs, only four fuzzy values (two for each input) are significant. This means that only four fuzzy rules are relevant at any one time. Therefore, instead of having to access 25 rules, the inference engine only has to access four rules during every computation. Index j is identified by as follows

Index j	ADR1			ADR2		
0	00	NB	N	00	NB	N
1	00	NB	N	01	N	Z
2	00	NB	N	10	Z	P
3	00	NB	N	11	P	PB
4	01	N	Z	00	NB	N
5	01	N	Z	01	N	Z
6	01	N	Z	10	Z	P
7	01	N	Z	11	P	PB
8	10	Z	P	00	NB	N
9	10	Z	P	01	N	Z
10	10	Z	P	10	Z	P
11	10	Z	P	11	P	PB
12	11	P	PB	00	NB	N
13	11	P	PB	01	N	Z
14	11	P	PB	10	Z	P
15	11	P	PB	11	P	PB

There are 16 possible MINI FAM TABLES along with rules which are considered relevant for the input conditions corresponding to the index *j*. It is observed that out of the 16 mini-FAM tables, there are only seven unique tables available(example j=1and j=4)If **WIN** is the index for the new set of tables, then the tables can be arranged using the following：

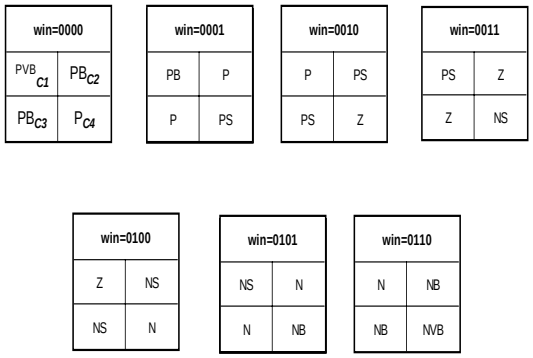

Figure 8 Identical MINI FAM table

Fig:2 MINI FAM table

This algorithm requires a considerable number of IF-THEN operations and is not necessarily an efficient way to implement the design into hardware. By observing the pattern in the original FAM table, it can be shown that the mini-FAM tables are identical when the sum of **ADR1** and **ADR2** is the same. Therefore, instead of using numerous IF-THEN operations, the arrangement of the mini-FAM tables is achieved using a single addition operation as shown by the following statement in the VHDL code

WIN <= (“00”& ADR1)+ADR2

In the conventional FLC, the inference engine contains 25 MIN-operations. The modified algorithm consists of only four MIN-operations, which is a notable reduction.

2.3 Defuzzification algorithm

The original algorithm for the MAX-operation and defuzzification process contains two important computations: the aggregation of 25 rule-consequents into nine output (fuzzy) values, and the multiplication of each output value by a constant weighting. By incorporating the modifications discussed in the previous sections, only four significant rule-consequents are considered. Therefore, the number of rule-consequents to be aggregated is reduced but the allocation of correct weightings for the significant output values becomes slightly more complicated. From the tables in Fig. 9. it is obvious that regardless of the **WIN** value, the consequents C2 and C3 always point to the same fuzzy value (e.g. when **WIN** = “0000”: C1→PVB, C2→**PB**, C3→**PB**, C4→P). This implies that only C2 and C3 have to be aggregated, hence:

DA=C1

DB=MAX[C1,C2]

DC=C4

where DA, DB and DC represent the membership function of the output fuzzy values. The actual fuzzy values referred to by DA, DB and DC are determined by the value of **WIN**. If E^i is the weighting while VA, VB and VC are the weighted values, then

WIN=”0000” : VA=DA * E^{PVB}, VA=DA * E^{PB}, VA=DA * E^{P}

WIN=”0001” : VA=DA * E^{PB}, VA=DA * E^{P}, VA=DA * E^{PS}

WIN=”0010” : VA=DA * E^{P}, VA=DA * E^{PS}, VA=DA * E^{Z}

WIN=”0011” : VA=DA * E^{PS}, VA=DA * E^{Z}, VA=DA * E^{NS}

WIN=”0100” : VA=DA * E^{Z}, VA=DA * E^{NS}, VA=DA * E^{N}

WIN=”0101” : VA=DA * E^{NS}, VA=DA * E^{N}, VA=DA * E^{NB}

WIN=”0110” : VA=DA * E^{N}, VA=DA * E^{NB}, VA=DA * E^{NVB}

Although the functions above can be implemented with seven IF-THEN statements it is preferable to adopt a less space consuming method based on recurring pattern .The new algorithm relies on the fact that the values of VA, VB and VC decrease steadily as **WIN** increases.

Then the defuzzified final output based on weighted average sum is defined as

$$y = \frac{VA + VB + VC}{DA + DB + DC}$$

Once the optimised structural-level design is successfully completed, it is implemented in FPGA..

3. Interfacing blocks

Figure 3 shows a block diagram demonstrating the implementation of the FLC in a Speed control of DC motor .The notation 'Z^{-1}' is used to mark a delay in the signal by one sampling period .In this application, the input interface converts the d.c. voltage at the output of the plant into *error* and *change of error* which are used as the two inputs to the FLC.

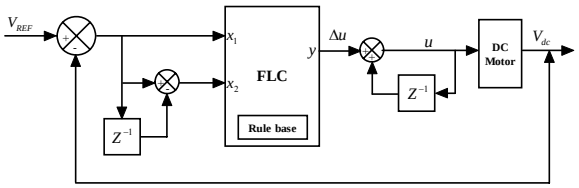

Fig 3 Overall block diagram of FLC based control system for DC motor

Another interface converts the output into the required value for the plant. The characteristics
of the interfacing blocks can be described by the following equations:

Input interface:
$e = V_{REF} - V_{dc}$
$x1 = e$
$x2 = x1 - x1\ z^{-1}$
Output interface:
$\Delta u = y$
$u = \Delta u + u\ z^{-1}$

4. FPGA Implementation

When the FPGA-based system is used for implementing the desired FLC, many possible designs can be tried due to the reusability of the FPGA. Often, a hardware implementation on the FPGA-based system is supported by many existing EDA tools for modelling, synthesis, verification, and implementation. The major advantage of using the EDA tools is that the same hardware description language code for modelling can be directly used for synthesis, verification, and implementation. Also, the general architecture of the FLC is invariant except for the change in the number of input and output variables, the number of fuzzy terms, the membership functions, the bit resolutions, and the control rule base according to the different applications.

The Target device in this case is Spartan®-3A DSP family (XC3SD1800A).The features of Spartan®-3A DSP – 2XC3SD1800A is tabulated in table2

System gates		1800k
Equivalent logic cells		37,440
CLB Array (one CLB = four slices)	Rows	88
	Columns	48
	Total CLB	4,160
	Total slices	16,640
Distributed RAM bits		260k
Block RAM bits		1512k
DSP48As		84
DCMs		8
Maximum user I/O		519
Maximum Differential I/O pairs		227

Table 2 Features of Spartan®-3A DSP

The FLC is divided in to 5 VHDL components. Figure 4 shows a diagram of the FLC architecture . Each component is depicted with its VHDL code. The functionality of components **Interface1**, **Interface2** and part of the component **Infer** will determine the characteristics of the FLC. Input and output variables are designed with a resolution of 8 bits.

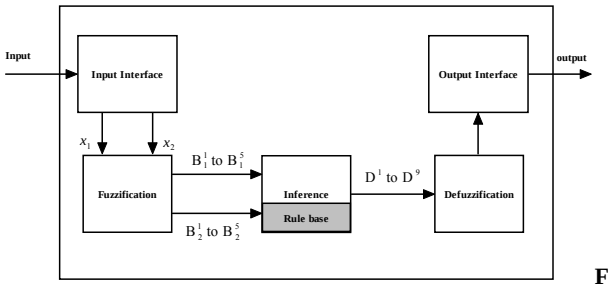

Fig 4: Components of Fuzzy logic controller

5. Experimental setup

The experimental studies are carried out to evaluate the performance of the controller. Configuration is the Process by which the bit streams of a design, as generated by the development software are loaded into the internal configuration memory of the FPGA.

Fig 5 Real time experimental setup

To verify the performance of the controller design on Hardware, the VHDL code (Bit file) is downloaded into the Target FPGA device (Spartan 3 family XC3S400) .the DC motor that is controlled in the work is a 1 HP, 1500rpm,220v machine. The complete real time setup is shown in Fig 5.

	Logic utilization	Available	Comparison	
			Conventional FLC	Optimised FLC
1	Number of Slice Flip Flop	33280	730	262
2	Total Number of 4 input LUTs	32280	5350	698
3	Number occupied Slices	16640	2851	357
4	Number Bonded IOBs	519	14	14
5	Number of BUFMUX	24	1	1
6	Number DSP48As	54	---	---
7	Total equivalent gate count for design	180000	50488	7160

The experimental set up for this system consists of a DC motor , FPGA kit(with on board ADC, DAC, PWM modules), Intelligent power module . The speed of the motor is measured by means of optical encoder and given as input to ADC(AD7266). The ADC used here is 12 bit, high speed, low power, successive approximation ADC and features throughput rates up to 2 MSPS. The set speed is assigned to motor by toggle switches according to the requirement. Once this is done the ADC data will be read and Fuzzy Logic Controller implemented will calculate output value and the output of controller in term changes the duty cycle of PWM to increase or decrease the speed.IPM Module consists of Switching power converters are used in most DC motor drives to deliver the required energy to the motor. The energy that a switching power converter delivers to a DC motor is controlled by Pulse Width Modulated (PWM) **Table 3 Resource utilization table**

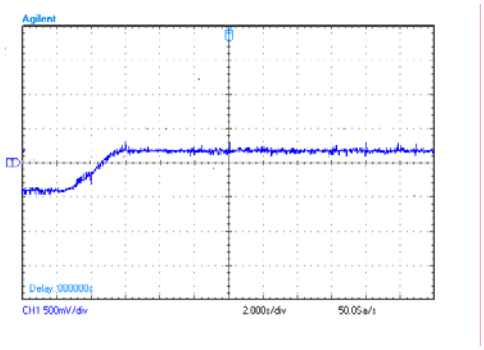

Fig 11 Servo response of optimized FLC

signal applied to the gate of a power transistor coming from PWM module FPGA kit.once the current speed equals the set speed, the motor starts running at the set speed. Again to change the set speed, the above procedure is repeated by changing the toggle switch position.

6 Results and Discussion

The optimized Fuzzy Logic Controller is implemented in FPGA for real time control of
speed in DC motor .Resource utilization of optimized FLC shown in table 3 .It provides the information regarding number slices, Flip Flops, Input output blocks, total number of gates required. Results shows that resources used for
optimized FLC is reduced drastically which suggests that design is so efficient, that means the FPGA chip can accommodate more controllers with adequate speed ,resulting in cost reduction of the controller hardware.Resource utilastion is reduced for the optimsed FLC while maintain same controller performance of conventional FLC. The FLC is run for a set point change from 550 rpm to 1100 rpm. The servo responses of the optimized FLC shown in Figure 10 .It is observed controller having good set point tracking with zero study state error and minimum overshoot.

7. Conclusion

In this paper, an novel optimized FLC is designed and implemented in FPGA for speed control of DC motor. Optimised FLC is also used in low cost FPGA which is having minimum resources. Experimental results of controller performance for both controllers show that response is smooth for set point change,. The optimized FLC uses very minimum resources utilized in FPGA with same controller performance when compared to conventional FLC. Also implementing FLC on FPGA features speed, accuracy, power compactness and cost improvement.

REFERENCES

[1] A. Ruiz, J. Gutiérrez, and J. Fernández, “A fuzzy controller with an optimized defuzzification algorithm,” IEEE Micro, pp. 1–10, Dec. 1995.
[2] M. J. Patyra, J. L. Grabtner, and K. Koster, “Digital fuzzy logic controller: Design and implementation,” IEEE Trans. Fuzzy Syst., vol. 4,pp. 439–459, Nov. 1996.
[3] Gerard0 aranguren, mariano barron, joseba l. Arroyabe and gonzalo garcia-carreira” a pipe-line fuzzy controller in FPGA”, IEEE conference in fuzzy systems (FUZZ-IEEE), 0-7803-3796-419, p- 635- 640, 1997.
[4] Valentinmure_san, dan and xiaojunwang” fromVHDL to FPGA. A case study of a fuzzy logic controller” proceedings of the international conference of young

lecturers and PHD students, pp. 83 - 90, miskolc, hungary, 11-17 august 1997.
[5] Philip t. Vuong, asad m. Madni and jim b. Vuong" VHDL implementation for a fuzzy logic controller", BEI technologies, inc.13100 telfair avenue, sylmar, california 91342 USA, 2006.http://www.wacong.org/wac2006/allpapers/isiac/isiac_248.pdf
[6] Oscar Montiel, Yazmin Maldonado, Roberto Sep´ulveda, and Oscar Castillo," Simple Tuned Fuzzy Controller Embedded into an FPGA", Annual Meeting of the North American, Fuzzy Information Processing Society, NAFIPS, ISBN: 978-1-4244-2351-4, p. 1-6, New York City, NY, 19-22 May 2008.
[7] Jacobo Alvarez, Alfonso Lago and Andres Nogueiras," FPGA Implementation of a Fuzzy Controller for Automobile DC-DC Converters", IEEE International Conference on Field Programmable Technology, ISBN: 0-7803-9729-0, p. 237-240, Dec. 2006
[8] Mohammed Y. Hassan and Waleed F. Sharif, " Design of FPGA based PID-like Fuzzy Controller for Industrial Applications", IAENG International Journal of Computer Science, 34:2, IJCS_34_2_05, 17 November 2007.
[9] Masmoudi n., hachicha m. and kamoun l." Hardware Design of Programmable Fuzzy Controller on FPGA", IEEE International Fuzzy Systems Conference Proceedings, Seoul, Korea, p. -1675- -1679, August 22-25, 1999.
[10] FPGA tutorial "Over view on FPGA", www.Tutorial-reports.com, 2008. http://www.tutorial-reports.com/computer-science/fpga/overview.php
[11] Daijin Kim, "An Implementation of Fuzzy Logic Controller on the Reconfigurable FPGA System", IEEE transactions on industrial electronics, vol. 47, no. 3, june 2000

G.Sakthivel obtained the B.E. (Electronics and Instrumentation) and M.E (Process Control and Instrumentation) degrees from Annamalai University in 1998 and 2005 respectively. He is presently a Selection grade Lecturer in the Department of Instrumentation Engineering, Annamalai University where he has put in a total service of 12 years. His research interests are intelligent control, adaptive control ,embedded system and VLSI design. . He is a life member of Instrument Society of India Education

T.S.Anandhi obtained the B.E. (Electronics and Instrumentation) and M.E (Process Control and Instrumentation) degrees from Annamalai University in 1996 and 1998 respectively. She is presently a Senior Lecturer in the Department of Instrumentation Engineering, Annamalai University where she has put in a total service of 8 years. Her research interests are in modeling and control of multiple connected DC-DC converters and multilevel inverters.

S.P.Natarajan was born in 1955 in Chidambaram. He has obtained B.E. (Electrical and Electronics) and M.E. (Power Systems) in 1978 and 1984 respectively from Annamalai University securing distinction and then Ph.D in Power Electronics from Anna University, Chennai in 2003. He is currently Professor and Head of Instrumentation Engineering Department at Annamalai University where he has put in 29 years of service. He has 15 publications in national journals and 22 international journals. His research interests are in modeling and control of DC-DC converters and multiple connected power electronic converters, control of PMBLDC motors, embedded control of multilevel inverters and matrix converters.