Progression in Science, Technology and Smart Computing

~ Executive Editor ~
Dr. Bhaskar V. Tamhankar

~ Editor ~
Dr. Neha Joshi

PRARUP PUBLICATIONS

ISBN – 978-81-948061-2-7

PROGRESSION IN SCIENCE, TECHNOLOGY AND SMART COMPUTING

Executive Editor

Dr. Bhaskar V. Tamhankar

Editors

Dr. Neha Joshi

Dr. Suresh Kumbhar

Dr. Narsing Padal

Shri. Amar Mane

Shri. Uttam Harale

Miss. Smruti Kulkarni

ISBN – 978-81-948061-2-7

PROGRESSION IN SCIENCE, TECHNOLOGY AND SMART COMPUTING

1st Edition 2021

Executive Editor:
Dr. Bhaskar V. Tamhankar
Principal,
Willingdon College, Sangli. (M.S.)

Publisher:
Prarup Publications
Kolhapur, Maharashtra.

ISBN 978-81-948061-2-7

Typeset and Printing
Mauli Computers & Printers, Kolhapur

ISBN – 978-81-948061-2-7

Sr. No.	Name of Authors	Title of Paper	Page No.
26	Hemant K Upadhyay, Tina	Analysis Of Issues And Challenges Of Online Teaching Using Fuzzy AHP Approach	184 – 191
27	Dr. Prakash Rajaram Chavan	Selection Of Most Informative Maintenance Approach By Using Fuzzy Multiple Criteria Decision Making Method	192 – 199
28	Jyoti Sabarad, Smita Kore, Najmin Mursal, Ashwini Patil, Neha Joshi	A Novel Approach of "5G Enabled IoT": A Review	200 – 209
29	Madhavi C. Bhosale, Dr. N.M.Kulkarni, Dr.A.D.Shaligram, Sharad Pustake	Comparative Study of RF Attenuation in Open Air with Closed Chamber	210 – 216
30	Uttam Harale, Jyoti Sabrad, Najmin Mursal, Ashwini Patil, Neha Joshi	Development of an Embedded System for Pesticide Residue Detection of Fruits and Vegetables	217 – 222
31	Ashwini Patil, Neha Joshi, Jyoti Sabarad, Najmin Mursal, Suresh Kumbhar	WSN Based Smart Industrial Pollution Monitoring System	223 – 228
32	Satyajit Ramesh Potdar	Modeling, Simulation and Validation of Greenhouse Microclimate	229 – 239
33	Poonam S. Jadhav, R. R. Mudholkar	FPGA Implementation of Fuzzy Logic Controller	240 – 245
34	K. G. Kharade, S. V. Katkar, S. K. Kharade, M. S. Sawant, R. K. Kamat	Review of Home Automation Strategies Analysis using Sensors	246 – 251
35	Sandip M. Dhavale, Nitin M. Kulkarni, Arvind D. Shaligram	A Comprehensive and Systematic Study of Energy Harvesting for Wearable Electronics	252 – 257
36	Asif Jamadar, Nishad Kumbhare, U. R. Harale, S. R. Kumbhar	Future of 7G Wireless Communication Technology and Evaluation	258 - 264
37	P.B.Sakhare, P.B.Dahikar	A Review on Internet of Things Pioneer for Recent COVID -19 and Post Pandemics	265 - 271
38	V. K. Lambe, G. M. Galgale, R. M. Chavan, J. V. Sabarad	IoT Based Home Automation System	272 – 278
39	Mrs. Akshata A. Nayak	Prognosis of Mental Health Disorder Among Population During Pandemic	279 - 286

ISBN – 978-81-948061-2-7

Progression in Science, Technology and Smart Computing

FPGA Implementation of Fuzzy Logic Controller

Poonam S. Jadhav
Department of Electronics,
Shivaji University, Kolhapur
Maharashtra, India-416004

R. R. Mudholkar
Department of Electronics,
Shivaji University, Kolhapur
Maharashtra, India-416004

ABSTRACT- This paper describes FPGA realization of Fuzzy Logic Controller (FLC) using hardware software Co-design methodology. In industrial application fuzzy logic controller is emerged as most promising method to improve the industrial control. The majority of Fuzzy Logic based real time operations require interfacing with high speed constraints. It leads to the need of finding efficient way to hardware implementation. FPGA becomes successful tool for developing the systems that requires a real time operation. It provides hardware rapidity and software flexibility.
Keywords- Fuzzy Logic, FPGA, Co-design.

INTRODUCTION:

Fuzzy logic control handle complex problem with ease. Fuzzy Logic can be customized and attuned effortlessly to get better or significantly alter system performance, because the Fuzzy Logic Controllers (FLC) processes user-defined rules ruling the target control system. Fuzzy systems are signifying good assurance in consumer products, industrial, commercial and decision support systems. Fuzzy Logic makes use of linguistic depictions to relate the input data with the output action. It has been implemented by various technologies such as digital and analog by using full custom or semi-custom techniques [1]. The reconfigurable FPGA offers hardware/software co-design. This methodology improves significantly the system performance by providing the less time delay among the simulation and response [2]. The fuzzy systems with dynamic reconfiguration, implemented with an FPGA. This co-evolutionary cooperation is used to increase the computing speed of the system [3].

IMPLEMENTATION OF FTC IN FPGA:

The hardware-software codesign approach on FPGA board can be applied for FLC implementation. The Fig.1. presents the design flow of the system.

ISBN – 978-81-948061-2-7

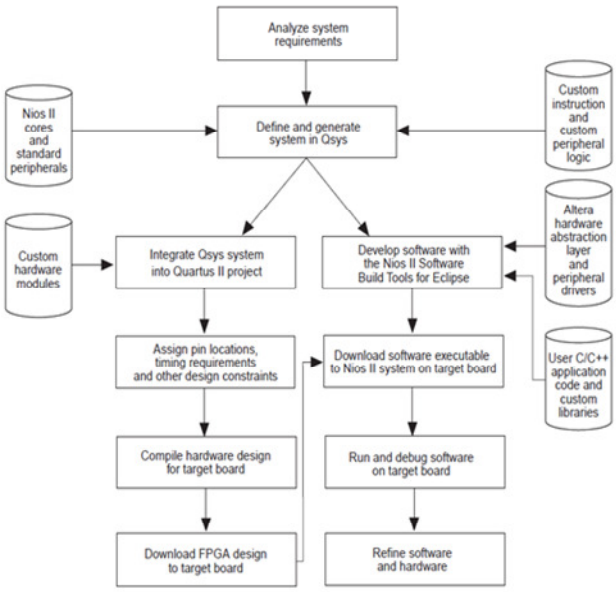

Fig. 1 The Development Flow of Co-design [4]

The development starts with analyzing and selecting the system component as per the system requirement. After that system integration tool Qsys used to specify Nios II processor core, memory and the other components. The hardware system is realized using SOPC builder tool, Qsys in combination with Quartus prime Edition 15.1. Qsys sanctions engendering a system predicated on Nios II.

Integrating Qsys system into Quartus II project, implements hardware modules into the FPGA design. Here pin locations, timing requirements are assigned and other design constraints are applied. Then the hardware design compiled for target board. After that the C application code is developed by using Software Build Tool (SBT). The low level hardware details can be written in Nios II programs with the drivers of component and HAL (hardware abstraction layer) provided by Altera. Along with application code user can design and reprocess custom libraries in Nios II SBT

In order to build a new application project based on C the SBT makes use of the hardware details containing '*.sopcinfo'* file. Previous to running the application FPGA board the system needs the '*.sof*' file. Refining hardware and software facility allows user to get better the software algorithm or can return to the hardware design steps to add acceleration logic.

FUZZY LOGIC CONTROLLER DESIGN AND IMPLEMENTATION:

The FLC designing involves fuzzification, inference engine, rule base, defuzzification and data pre and post processing. The software development task has been performed for Nios Processor system by using Nios II-SBT. The SBT is used to write C application code.

ISBN – 978-81-948061-2-7

A Nios II C application project is created by utilizing the *.sopcinfo* file and *.sof* file. The *.sopcinfo file* is required for getting hardware information of the system and *.sof* file is required to execute the program on the FPGA device.

A. Fuzzification Module **-** Inputs to the FLC are *"Error"* and *"Change-in-Error"*. In the present controller triangular and trapezoidal membership functions (MF's) are used. The values of fuzzy variables are presented in linguistic terms. The linguistic variables for inputs are displayed in Table1.

Table 1: Input Linguistic Variables

Sr. No	Input Variable Name	Crisp Input Range of "Error"	Crisp Input Range of "Change-in-Error (CE)"
01	**NM**	**[-6 -6 -4.6 -2.4]**	**[-5 -5 -2.5]**
02	**NS**	**[-3.5 -2.024 0]**	**[-3 -1.6 0]**
03	**Z**	**[-0.489 0 0.437]**	**[-0.5 0 0.5]**
04	**PS**	**[0 1.45 2.95]**	**[0 1.6 3]**
05	**PM**	**[1.9 3.76 6 6]**	**[2.5 5 5]**

Fig. 2 and Fig. 3 displays membership functions for input variables "Error" and "Change-in-error".

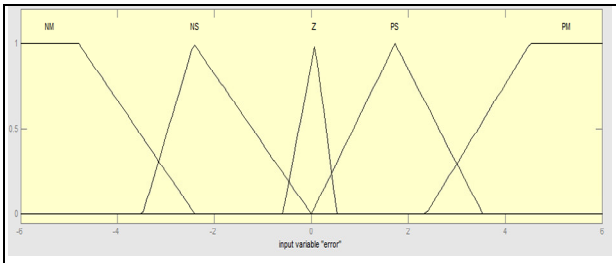

Fig. 2 MF for Input Variable "Error"

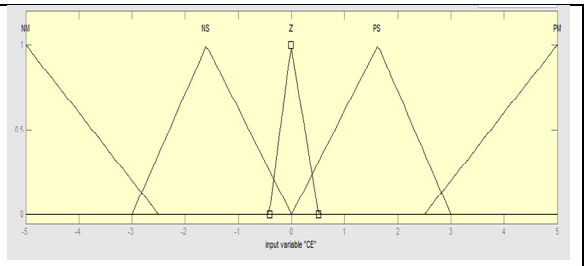

Fig. 3: MF for Input Variable *"Change-in-Error (CE)"*

The input variables *"Error"* and *"Change-in-Error"* have five membership functions as Negative Medium (NM), Negative Small(NS),Zero(Z),Positive Small(PS), and Positive Medium(PM). A coding of linguistic variables is necessary to include the linguistic variables into the FPGA.

B. Fuzzy MF's for Output

The output variables expressed linguistically are applied to the actuators to control. The linguistic variables for output are displayed in Table 2. Fig. 4 presents membership function for output variable control.

ISBN – 978-81-948061-2-7

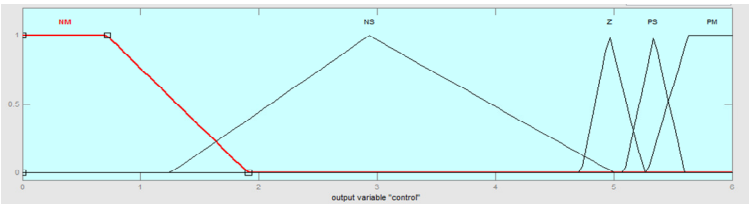

Fig. 4 MF for Output Variable "Control"

Table 2: Output Linguistic Variables

Sr. No	Input Variable Name	Crisp Output Range of "Control"
01	**NM**	**[0 0 0.7 1.9]**
02	**NS**	**[1.25 3 5]**
03	**Z**	**[4.7 5 5.26]**
04	**PS**	**[5 5.34 5.6]**
05	**PM**	**[5.28 5.63 6 6]**

```
#include"stdio.h"
#define MAXNAME 10   /* max number of characters in names */
#define UPPER_LIMIT 255   /* max number assigned as degree of membership */
#define MIN_TEMP 25
#define MAX_TEMP 100
#define System_Inputs 10
#define System_Outputs 10
#define MIN MIN_TEMP
#define MAX MAX_TEMP
#define RATIO 100/(MAX-MIN)
 signed int NM[] ={-50,-50, -35, -20,-4,-4,-3,-2,0,0,1,1 };
 signed int NS[] ={-40,-20, 0, -3,-1.5,0,1,2,3 };
 signed int Z[] ={-10,0, 10, -1,0,1,2.5,3,3.5 };
signed int PS[] ={0,20, 40, 0,1.5,3,3,4,5 };
signed int PM[] ={20,35,50,50,2,3,4,4,4,5,6,6 };
```

Fig. 5 C code for the input and output variables

The C code for the implementation of fuzzy membership functions has been displayed in Fig. 5.

C. Inference Module

A FIS processes input data and evaluates the proposed action in terms of their agreement with the knowledge base. An important task in an inference engine is to finding the matching degree among the fuzzified input MF and the antecedent MF [5]. The matching degree can be obtained from max-min operation. The C code in Fig. 6 is applied for the FIS in Nios- II Eclipse SBT.

```
struct rule_element_type {
int *value;                    /* pointer to antecedent/output strength value */
struct rule_element_type *next;     /* next antecedent/output element in rule */
};
struct rule_type{
struct rule_element_type *if_side;          /* list of antecedents in rule */
struct rule_element_type *then_side;        /* list of outputs in rule */
struct rule_type *next;                     /* next rule in rule base  */
};
struct rule_type *Rule_Base;
```

Fig. 6 C code for Fuzzy Inference Module

D. Knowledge Representation and Rule Block

Knowledge base provides the necessary information about the rules and data handling in fuzzy logic controller [6]. The logical rule base indicates the correlation input and output element. Once the current values of the inputs are fuzzified, the FLC starts to make decisions to perform the required operation. The rule consists of "IF-THEN" statement. If part indicates the condition for which it is designed and then part presents reaction of the fuzzy system in that situation. The fuzzy rules are presented in Table 3. It consists of 25 rules. The all 25 rules are executed. For example If the error is Negative Medium (NM) and Change-in-Error (CE) is Negative Medium (NM) then control output is Positive Medium (PM).

Table 3: Fuzzy Rules

Control		Change-in-Error (CE)				
		NM	**NS**	**Z**	**PS**	**PM**
Error (E)	**NM**	**PM**	**PM**	**PM**	**PS**	**Z**
	NS	**PM**	**PM**	**PS**	**Z**	**NS**
	Z	**PM**	**PS**	**Z**	**NS**	**NM**
	PS	**PS**	**Z**	**NS**	**NM**	**NM**
	PM	**Z**	**NS**	**NM**	**NM**	**NM**

The following C code (Fig.7) is applied to implement the fuzzy rules in Nios- II Eclipse SBT.

```
rule  evaluation ()
{
 struct rule_type *rule;
 struct rule_element_type *ip;                  /* pointer of antecedents  (if-parts)   */
 struct rule_element_type *tp;                  /* pointer to consequences (then-parts) */
 int strength;                                  /* strength of rule currently being evaluated */
 for (rule=Rule_Base; rule!= NULL; rule=rule->next)
 {
  strength = UPPER_LIMIT;                       /* max rule strength allowed */
   for (ip=rule->if_side; ip != NULL; ip=ip->next)   /* process if-side of rule to determine
strength */
  strength = min(strength,*(tp->value));  /* process then-side of rule to apply strength */
  for(tp=rule->then_side; tp != NULL; tp=tp->next)  *(tp->value) = max (strength,*(tp-
>value));
  }
```

Fig. 7 C code for Implementation of Fuzzy Rules

E. Defuzzification Module

The defuzzification process converts fuzzified value into crisp value to use fuzzy results in control application. The output of defuzzification is a numeric value that decides controls output. The "Centre-of-gravity" (COG) defuzzification technique is applied in the present system. This method is consisting of four steps [7].

```
defuzzification ( )
{
 struct io_type *so;                    /* system output pointer */
 struct mf_type *mf;                    /* output membership function pointer */
 int sum_of_products;                   /* sum of products of area & centroid */
 int sum_of_areas;                      /* sum of shortend trapezoid area */
 int area;
 int centroid;
 /* compute a defuzzified value for each system output */
for (so=System_Outputs; so != NULL; so=so->next){
  sum_of_products = 0;
  sum_of_areas = 0;
  for (mf=so->membership functions; mf != NULL; mf=mf->next){
   area = compute_area_of_trapezoid (mf);
  centroid = mf->point1 + (mf->point2 - mf->point1)/2;
  sum_of_products += area * centroid;
  sum_of_areas += area;
    }
  so->value = sum_of_products/sum_of_areas;        /* weighted average */
  }
}
```

Fig. 8: C code for defuzzification

CONCLUSION:

The FPGA realization of fuzzy logic controller technique on FPGA has been described. By changing parameter values in code and design constraints one can test for different design circuitry. Implementation of FLC in FPGA provides highest execution speed, less time to market use of suitable software tools, and significant integration density.

REFERENCES:

1) Kim, D. *An Implementation of Fuzzy Logic Controller on the Reconfigurable FPGA System*, IEEE Transactions on Industrial Electronics, Vol.47 (3), pp:703-715,2000.

2) Masmoudi, M. S., Song, I., Karray, F., Masmoudi, M., and Derbel, N. *Hardware/Software Approach for FPGA Implementation of Fuzzy Logic Controller*, International Conference on Design and Test of Integrated Systems in Nanoscale Technology, DTIS 2006, pp:419-423,2006.

3) Ramos-Arreguin, J. M., Guillen-Garcia, E., Canchola-Magdaleno, S., Pedraza-Ortega, J. C., Gorrostieta-Hurtado, E., Aceves-Fernández, M. A., and Ramos-Arreguin, C. A. *Fuzzy Logic Hardware Implementation for Pneumatic Control of One DOF Pneumatic Robot*, Mexican International Conference on Artificial Intelligence MICAI 2010, Advances in Soft Computing, pp: 500-511,2010.

4) Altera Nios II Hardware Development Tool, Browsed on February 2017 http://persoetis.ensea.fr/olivier.romain/Teaching_2A_IUT_UCP_files/tutorial.pdf

5) Lee, C. C. *Fuzzy Logic in Control Systems: Fuzzy Logic Controller-Part I,* IEEE Transactions on Systems, Man and Cybernetics, Vol:20 (2), pp:404-418,1990.

6) Ross, T. J. Membership Functions, Fuzzification and Defuzzification. In: Szczepaniak, P. S., Lisboa, P. J. G., Kacprzyk, J. (eds), *Fuzzy Systems in Medicine. Studies in Fuzziness and Soft Computing*, Physica, Heidelberg. Vol.41, pp:48-77,2000.

7) Bouselham, L., Hajji, M., Hajji, B., El Mehdi, A., and Hajji, H. *Hardware Implementation of Fuzzy Logic MPPT Controller on A FPGA Platform*, 3rd International Conference on Renewable and Sustainable Energy Conference, 1-6,2015.