American Journal of Applied Sciences 8 (7): 740-747, 2011
ISSN 1546-9239

Subway Train Braking System: A Fuzzy Based Hardware Approach

Mamun Bin Ibne Reaz, Jubayer Jalil, Hafizah Husain and Badariah Bais
Department of Electrical, Electronic and Systems Engineering,
University Kebangsaan Malaysia, 43600, UKM, Bangi, Selangor, Malaysia

Abstract: Problem statement: Automated subway train-braking system require perfection, efficiency and fast response. In order to cope with this concerns, an appropriate algorithm need to be developed which need to be implemented in hardware for faster response. **Approach:** In this research, the FPGA realization of fuzzy based subway train braking system has been presented on an Alter FLEX10K device to provide an accurate and increased speed of convergence of the network. The fuzzy based subway train braking system is comprised of fusilier, inference, rule selector and defuzzifier modules. Sixteen rules are identified for the rule selector module. After determining the membership functions and its fuzzy variables, the Max-Min Composition method and Madman-Min implication operator are used for the inference module and the Centre of Gravity method is used for the defuzzification module. Each module is modeled individually using behavioral VHDL. The layers are then connected using structural VHDL. Two 8-bit and one 8-bit unsigned digital signals are used for input and output respectively. Six ROMs are defined in order to decrease the chances of processing and increasing the throughput of the system. **Results:** Functional simulations were commenced to verify the functionality of the individual modules and the system as well. We have validated the hardware implementation of the proposed approach through comparison, verification and analysis. The design has utilized 2372 units of LC with a system frequency of 139.8MHz. **Conclusion:** In this research, the FPGA realization of fuzzy brake system of subway train has been successfully implemented with minimum usage of logic cells. The validation study with C model shows that the hardware model is appropriate and the hardware approach shows faster and accurate response with full automatic control.

Key words: Fuzzy logic, fuzzy controller, braking system, Field-Programmable Gate Arrays (FPGA), brake pressure, accurate response, defuzzification module, proposed approach, rule selector, subway train, rule selector, brake pressure

INTRODUCTION

Fuzzy techniques are becoming an attractive approach to handle uncertain, imprecise, or unmodeled data in solving control and intelligent decision-making problems (Zadeh, 1972; Al-Odienat, 2008). This is mainly due to their inherent ability to describe a complex system by means of a simple set of intuitive and ambiguous behavioral rules. The application of fuzzy technologies into real-time control problems demands the development of efficient hardware implementations of fuzzy inference mechanisms.

In this paper, fuzzy logic utilities for synthesis of control system are discussed in the context of an application to the subway train braking system. A fuzzy based subway train braking system has the advantage that it allows the intuitive nature of accurate prediction of brake pressure to be easily modeled using the number of linguistic terms, the respective membership functions for each linguistic variable, the inference mechanism, the rules, the fuzzification and defuzzification methods (Victor and Dourado, 1997; Zachary, 2010). Due to the relative computational simplicity of fuzzy rule-based system, intelligent decision can be made in real-time, thus allowing an accurate prediction of brake pressure.

The Field-Programmable Gate Arrays (FPGA) offers a potential alternative to speed up the hardware realization (Coussy *et al*., 2009; Reaz *et al*., 2007). From the perspective of computer-aided design, FPGA comes with the merits of lower cost, higher density and shorter design cycle (Choong *et al*., 2005; Akter *et al*., 2008). It comprises a wide variety of building blocks. Each block consists of programmable look-up table and storage registers, where interconnections among these blocks are programmed through the hardware description language (Reaz *et al*., 2004; 2005). This programmability and simplicity of FPGA made it

Corresponding Author: Mamun Bin Ibne Reaz, Department of Electrical, Electronic and Systems Engineering,
University Kebangsaan Malaysia, 43600, UKM, Bangi, Selangor, Malaysia

favorable for prototyping digital system. FPGA allows the users to easily and inexpensively realize their own logic networks in hardware. FPGA also allows modifying the algorithm easily and the design time frame for the hardware becomes shorter by using FPGA (Choong *et al*., 2006; Ibrahimy *et al*., 2006).

In this study, a unified framework for FPGA realization of fuzzy based subway train braking system is designed by means of using a standard hardware description language VHDL. The use of VHDL for modeling is especially appealing since it provides a formal description of the system and allows the use of specific description styles to cover the different abstraction levels (architectural, register transfer and logic level) employed in the design (Pang *et al*., 2006: Reaz *et al*., 2006; 2007). In the computation of method, the problem is first divided into small pieces, each can be seen as a submodule in VHDL. Following the software verification of each submodule, the synthesis is then activated. It performs the translations of hardware description language code into an equivalent netlist of digital cells. The synthesis helps integrate the design work and provides a higher feasibility to explore a far wider range of architectural alternative (Reaz *et al*., 2005; 2004). The method provides a systematic approach for hardware realization, facilitating the rapid prototyping of the fuzzy based subway train braking system.

MATERIALS AND METHODS

FPGA realization of subway train braking system using fuzzy technique is introduced to implement a dedicated hardware for faster and accurate response with full automatic control. The subway train braking system, which determines the amount of brake pressure with given two inputs: distance and speed. The fuzzy based subway train braking system is comprised of four major blocks: the fuzzifier, the rule selector, the inference and the defuzzifier. Three membership functions 'distance' and 'speed' as input and 'brake' as output are chosen. For 'distance', the range of 500m with fuzzy variables of Very Close, Close, Far and Very Far are chosen.

Table 1: Sixteen (16) rules of the rule selector module

Speed	Very slow	Slow	Fast	Very fast
Distance				
Very close	Light	Heavy	Very heavy	Very heavy
Close	Light	Very heavy	Heavy	Very heavy
Far	Light	Very light	Light	Heavy
Very far	Very light	Very light	Light	heavy

For 'speed', a range of 100km h^{-1} with fuzzy variables of Very Slow, Slow, Fast and Very Fast are chosen
For 'speed', a range of 100km h^{-1} with fuzzy variables of Very Slow, Slow, Fast and Very Fast are chosen Similarly, for 'brake' a range of 100 with fuzzy variables of Very Light, Light, Heavy and Very Heavy are chosen. The rule selector module contains 16 rules as shown in Table 1.

The inference inputs the truth-values generated from the fuzzification process. The inference module performs the fuzzy composition using Max-Min Composition and Mamdani-Min implication operator (Mamdani *et al*., 1947; Zimmermann *et al*., 1985). The inference engine goes through all the rules in the rule selector, applies a composition type to evaluate the firing strength of each rule-which in turn is proportional to the truth-value of the preconditions.

The defuzzification module converts the fuzzy output set, back to a crisp number by using the Centre of Gravity method. A block diagram illustrating the major blocks of the subway train braking system is shown in Fig. 1. The system consists of two inputs: X0 and X1, each corresponding to the values of distance and speed respectively. The output of the system gives the predicted brake pressure to be applied. In VHDL modeling of the fuzzy based subway train braking system, the entity declaration provides an "external" view of a component. It defines an interface to a component and describes the properties of the components' ports.

The overall structure consists of the main entity of the fuzzy logic controller, "Train_FLC" and four components namely, "Fuzzifier", "Inference", "Rule_Selector" and "Defuzzifier". All the operations done in the system are carried out with unsigned numbers and all the components are connected to one another through the VHDL structural description.

The inputs are directly mapped into the Fuzzifier and Rule_Selector. The outputs of the Fuzzifier are mapped into the Inference. Following that, the outputs of the Inference and the Rule_Selector components are mapped into the Defuzzifier.

Fuzzifier: The starting block, fuzzifier takes two inputs 'distance' and 'speed', which are crisp values and converts them to fuzzy values.

Each of the membership functions expressed as $U = mX + C$, where U is the membership function of the fuzzy variable and X is the crisp inputs with distance (X0) and speed (X1). The variable m is the gradient of the line equation while the variable C represents the intersection at the U axis (Donato and Barbieri, 1985; Alcalaa *et al*., 2007).

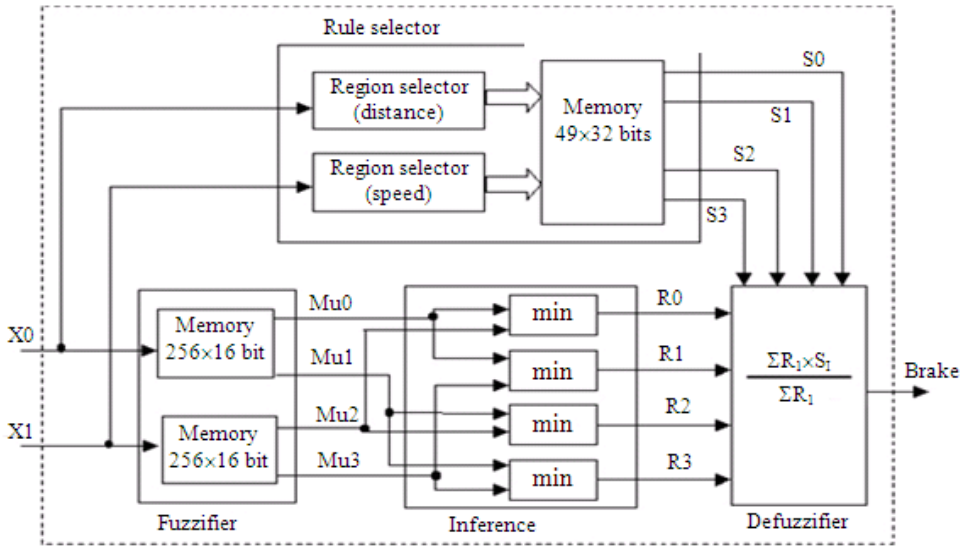

Fig. 1: Block diagram of the brake system

The fuzzifier has one fuzzifier-entity and four ROM components labeled as 0, 1, 2 and ROM_3. It has two 8-bit input ports and four 8-bit output ports. The input ports are labeled as 0 and X1, which takes the value of distance and speed while the output ports are labeled as 1, 2, 11and MU12. These output ports are mapped into the Inference component.

Initially, the crisp values of distance and speed are normalized and passed into the fuzzifier module, where the values are rounded and converted to hexadecimal. All the hexadecimal values are printed into text files and passed into the VHDL ROM codes. The 8-bit inputs of distance and speed are translated into memory locations in the ROM, which are actually the fuzzified membership function values, mapped at different values from the universe of discourse.

Each ROM has one 8-bit input and one 8-bit output as address and data respectively. The address ports take the value of 0-X1, while the data ports maps into the output ports 1, 2, 11 and MU12 of the fuzzifier. _0 - ROM_1 consist of fuzzified values for distance while 2 and ROM_3 consist of fuzzified values for speed.

Inference: The inference block accepts four inputs from the fuzzifier block, which are inferred using the Max-Min Composition and Mamdani-Min implication operator to produce four outputs.

It contains one Inference entity and one component outputs 0-R1, 2-R3. The inputs of inference are mapped from the outputs of fuzzifier and the outputs of the labeled as Comparator. The Inference entity consists of four 8-bit inputs, 1-MU02, 11-MU12 and four 8-bit inference are mapped into the defuzzifier as its input. The inference provides a minimum of the membership function values from the fuzzifier. The minimum value, found by mapping the inference inputs, is fed into the comparator as input.

Rule selector: The rule selector takes two crisp inputs, distance and speed. In the rule selector module, only two fuzzy variables exist in each region of comparison. Therefore, after analyzing the membership functions of both speed and distance, it is divided into seven regions. The division of regions for input 0 and X1 is shown in Table 2.

Once all the regions are determined, each of the regions in distance is compared to the regions in speed. At each comparison, a corresponding associated rule from the fuzzy rule selector is fired. For faster inferencing and defuzzification, singleton sets are applied in the system. The output of the rule selector takes the values of the singleton values indicated by the fired rule.

The entire process for all the comparison of regions considering inputs of 0 and X1 were drawn up and tabulated in an Active Rules Lookup Table. Thus for each comparison, the rule selector looks up the table and determines the rules to be fired and its corresponding singleton values. The rule selector has one Rule_selector entity and two components.

Table 2: Division of regions for input X0 and X1

Input	Regions	Range of values
Distance (X0) (m)	1	0-10
	2	10-20
	3	20-60
	4	60-120
	5	120-250
	6	250-350
	7	350-500
	1	0-3
	2	3-5
Speed (X1) (km h^{-1})	3	5-20
	4	20-30
	5	30-45
	6	45-60
	7	60-100

The Rule_selector entity has two 8-bit input ports and four 8-bit output ports. The input ports 0 and X1 take values of distance and speed. The output ports 0, 1, 2 and S3 are mapped into the defuzzifier. The component region functions as a region selector.

The entity Region has two 8-bit input ports, X0 and X1 and one 6-bit output port X. There are 49 region combinations to determine the rules to be fired.

The ROM_output entity has one 6-bit input port Y and four 8-bit output ports 0, 1, 2 and 3. The ROM_output contains the singleton values of S0-S3. The most significant 16 bits hold the binary values of S3 down to S2 while the 16 least significant bits hold the binary values of 1 and S0.

Deffuzifier: The defuzzifier module takes four inputs from the inference module and four inputs from the rule selector module. The output of the defuzzifier is the brake pressure. The defuzzifier converts the fuzzy values back into the crisp values. The Centre of Gravity method was chosen for defuzzification that favors "central" values in the universe of discourse and only takes into account the area of the resultant membership function. This method of defuzzification is expressed by the equation $\left(\sum R_i \times S_i\right) / \left(\sum R_i\right)$. The defuzzifier block contains one entity Defuzzifier and four components labeled as Adder8bit, Adder16bit, Multiplier and Divider. The Defuzzifier entity has eight 8-bit inputs: S0, S1, S2, S3, R0, R1, 2-R3 and one output 'brake'. The inputs S0-S3 are mapped from the rule selector, while the inputs 0-R3 are mapped from the inference unit.

An 8-bit adder, a 16-bit adder, four 8-bit multipliers and a divider are chosen to realize the Center of Gravity method. A total of four multipliers are used since there are four S inputs and four R inputs. The multiplier unit accepts two 8-bit signals and provides a 16-bit signal, which performs the multiplication of $R_i \times S_i$. The outputs of each multiplier are then added. Thus, the 16-bit output signal of the multiplier is fed into a 16-bit adder. The 16-bit adder has an output of 18-bit due to the carry bits. The output of the inference is fed into an 8-bit adder, which produces an output of 10-bits again due to the carry bits. The divider component performs the division operation by converting the divisor containing the output of the 8-bit adder to its inverse. The ROM values are then multiplied with the dividend that contains the outputs of the 16-bit adder. The multiplication produces an output of a 34-bit number. However, since the first 16-bits represent the decimal value of the number, the remaining bits are neglected. Therefore, the output would only consist of an 8-bit signal to the Brake port.

RESULTS

Simulation: In order to evaluate the performance of the VHDL model of subway train braking system, a set of input values of distance and speed are fed into the model. Singleton output values from the fuzzifier and the rule selector are taken to analyze the degree of fulfillment and the brake values are taken for verification of the overall output. The input values are converted into its binary form before being entered into the test-bench of the VHDL model. The generated simulated waveform is shown in Fig. 2 and the converted binary values of hexadecimal output are tabulated in Table 3.

The same input, which is a real value, is fed into a C model in order to compare the result. The C model was set as benchmark against the VHDL model. The corresponding output is shown in Table 4.

Synthesis: For the designated hardware realization, the VHDL code is synthesized by considering an Altera FLEX10K: EPF10K10LC84 FPGA chip on an LC84 package. The FLEX 10K family provides the density, speed and features to integrate the entire systems, including multiple 32-bit buses into a single chip. The top level RTL view and the top-level technology view are shown in Fig. 3-4 respectively.

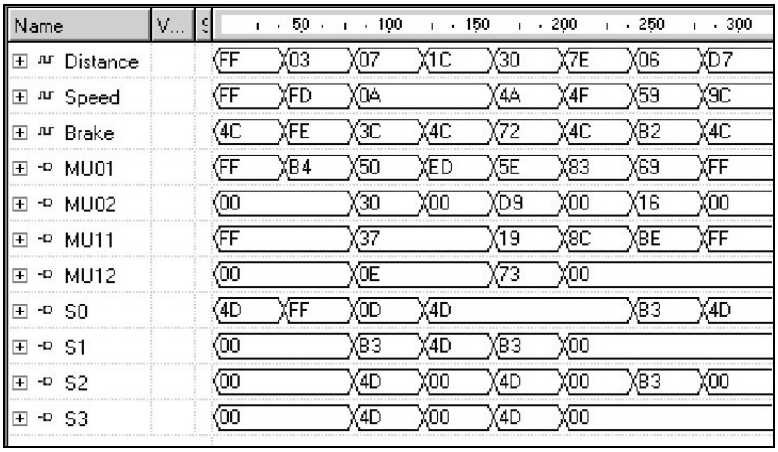

Fig. 2: Waveform simulated from the VHDL model

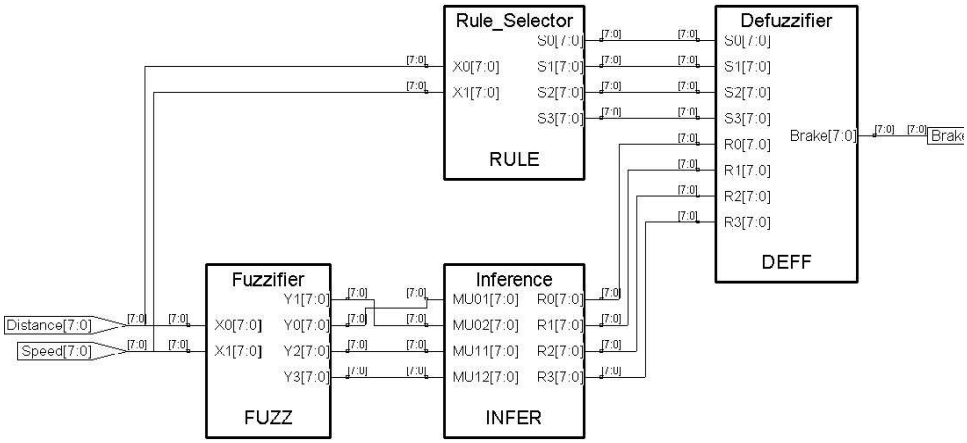

Fig. 3: Top level RTL view

Table 3: Data from the VHDL model simulation

Distance	Speed	U0	U1	U2	U3	S0	S1	S2	S3	Break
255	255	1.0000	0.0000	1.000	0.0000	30	0	0	0	29.8
3	253	0.7059	0.0000	1.000	0.0000	100	0	0	0	99.6
7	10	0.3137	0.1882	0.216	0.0549	5	70	30	30	23.5
28	10	0.9294	0.0000	0.216	0.0549	30	30	0	0	29.8
48	74	0.3686	0.8510	0.098	0.4510	30	70	30	30	44.7
126	79	0.5137	0.0000	0.549	0.0000	30	0	0	0	29.8
6	89	0.4118	0.0863	0.745	0.0000	70	0	70	0	69.8
215	156	1.0000	0.0000	1.000	0.0000	30	0	0	0	29.8

Table 4: Data from the C model simulation

Distance	Speed	U0	U1	U2	U3	S0	S1	S2	S3	Break
500	100	1. 0000	0.00	1.00	0.0000	30	0	0	0	30.0
5	99	0.7500	0.00	1.00	0.0000	100	0	0	0	100.0
13	4	0.3500	0.15	0.20	0.0588	5	70	30	30	24.34
55	4	0.9286	0.00	0.20	0.0588	30	30	0	0	30.00
94	29	0.3714	0.85	0.10	0.4500	30	70	30	30	44.55
247	31	0.5150	0.00	0.55	0.0000	30	0	0	0	30.00
11	35	0.4500	0.50	0.75	0.0000	70	0	70	0	70.00
421	61	1.0000	0.00	1.00	0.0000	30	0	0	0	30.00

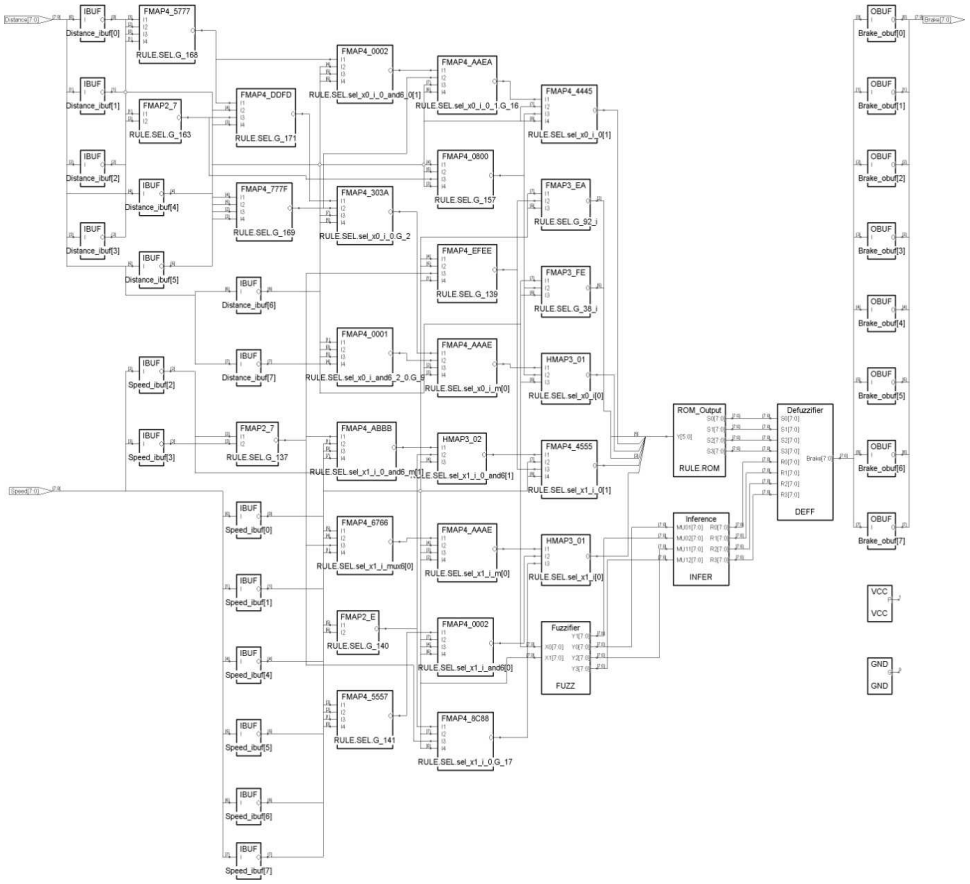

Fig. 4: Top level technology view

DISCUSSION

To verify the efficiency and perfectness, we have also developed the C model of our proposed design.
In the comparative analysis, the output results of the C model are compared with the output of VHDL model. The percentage of error was calculated as shown in equation (1):

$$\% \text{ of Error} = \frac{\text{Difference in VHDL and C output} \times 100}{\text{Output of C}} \quad (1)$$

Using the above equation, the errors for the brake outputs are calculated and shown in Table 5.

In Table 5, it is being shown that the maximum error is indicated as 3%. Most of the large errors occur at the fuzzified outputs. This is due to the fact that the crisp values have to be converted to fuzzy values in the fuzzifier.

Table 5: Percentage error of brake pressure

Brakes (C Model)	Brakes(VHDL Model)	% Braking error
30.00	29.80	0.67
100.00	99.60	0.40
24.34	23.52	3.36
30.01	29.80	0.67
44.55	44.71	0.37
70.00	69.80	0.29

As a result, some of the values are modified due to the variation in the slope of the membership function and thereby causes some information loss. Thus the largest errors occur in the fuzzifier kernel. The synthesis esults obtained from Quartus II showed that the whole system has taken up 2372 units of logic cells, which is about 47% utilization of the device (Altera EPF10K10LC84). Further optimization is still possible though the utilization of the logic cells are already very little. The clock frequency report showed the critical frequency is 139.8MHz. This frequency is fast enough in terms of faster response of the braking system.

CONCLUSION

The FPGA realization of fuzzy based subway train braking system has been proposed and its potentialities, as indicated by the good prediction of brake values, have been presented. By simulating and synthesizing with the values of distance and speed, the proposed approach has been successfully designed, implemented and tested. It is found that the error is almost negligible when the VHDL simulation result has been compared with the results from C model, which shows that hardware realization train braking system is appropriate. Moreover, hardware solution is faster and robust than software solution.

REFERENCES

Akter, M., M.B.I. Reaz, F. Mohd-Yasin and F. Choong, 2008. Hardware implementations of image compressor for mobile communications. J. Commun. Tech. Elect., 53: 899-910. DOI: 10.1134/S106422690808007X

Alcalaa, R., J. Alcala-Fdez, M.J. Gacto and F. Herreraa, 2007. Genetic Learning of Membership Functions for Mining Fuzzy Association Rules. IEEE International Fuzzy Systems Conference, 23-26 July, London, UK., pp. 1-6. DOI: 10.1109/FUZZY.2007.4295595.

Al-Odienat, A.I. and A.A. Al-Lawama, 2008. The advantages of pid fuzzy controllers over the conventional types. Am. J. Applied Sci., 5: 653-658. DOI: 10.3844/ajassp.2008.653.658

Cheong, C.W., L.H. Jie, M.C. Meng and A.L.H. Lan, 2008. Design and development of decision making system using fuzzy analytic hierarchy process. Am. J. Applied Sci., 5: 783-787. DOI: 10.3844/ajassp.2008.783.787

Choong, F., M.B.I. Reaz and F. Mohd-Yasin, 2005. Power quality disturbance detection using artificial intelligence: A hardware approach. Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05), Workshop, Apr. 4-8, Denver, USA., pp: 146a. DOI: 10.1109/IPDPS.2005.348

Choong, F., M.B.I. Reaz, T.C. Chin and F. Mohd-Yasin, 2006. Design and implementation of a data compression scheme: A partial matching approach. Proceedings of the Computer Graphics, Imaging and Visualisation: Techniques and Applications (CGIV'06), Jul. 26-28, Sydney, Australia, pp: 150-155. DOI: 10.1109/CGIV.2006.94.

Coussy, P., D.D. Gajski, M. Meredith and A.Takach, 2009. An Introduction to High-Level Synthesis. IEEE Des. Test Comput., 26: 8-17. DOI: 10.1109/MDT.2009.69

Donato, J.M., E. Barbieri, 1995. Mathematical Representation of Fuzzy Membership Functions. Proceedings of the 27th Southeastern Symposium on System Theory (SSST'95), pp: 290-294. DOI: 10.1109/SSST.1995.390567

Ibrahimy, M.I., M.B.I. Reaz, M.A. Mohd Ali, T.H. Khoon and A.F. Ismail, 2006. Hardware realization of an efficient fetal QRS complex detection algorithm. WSEAS Trans. Circ. Syst., 5: 575-581. http://www.worldses.org/journals/circuits/circuits-april2006.doc or http://www.scopus.com/record/display.url?src=sandorigin=ctoandctoId=CTODS_161278959andstateKey=CTOF_161278960andeid=2-s2.0-33744522404

Khan, S., S.F. Abdulazeez, L.W. Adetunji, A.H.M.Z. Alam and M.J.E. Salami, *et al*., 2008. Design and implementation of an optimal fuzzy logic controller using genetic algorithm. J. Comp. Sci., 4: 799-806. DOI: 10.3844/jcssp.2008.799.806

Mamdani, E.H. and N.S. Assilian, 1974. A case study on the application of fuzzy set theory to automatic control. Proceedings of IFAC Stochastic Control Symposium, Sept. 25-27, Budapest, Hungary, pp: 643-649. http://doc.utwente.nl/68046/1/Bagchi75symposium.pdf

Marufuzzaman, M., M.B.I. Reaz, M.S. Rahman, M.A.M. Ali, 2010. Hardware prototyping of an intelligent current dq PI controller for FOC PMSM drive. Proceedings of the 6th International Conference on Electrical and Computer Engineering, Dec. 18-20, Dhaka, Bangladesh, pp: 86-88. DOI: 10.1109/ICELCE.2010.5700559

Pang, W.L., M.B.I. Reaz, M.I. Ibrahimy, L.C. Low and F. Mohd-Yasin *et al*., 2006. Handwritten character recognition using fuzzy wavelet: A VHDL approach. WSEAS Trans. Syst., 5: 1641-1647. http://www.worldses.org/journals/systems/systems-july2006.doc or http://www.scopus.com/record/display.url?src=sandorigin=ctoandctoId=CTODS_161278959andstateKey=CTOF_161 278960andeid=2-s2.0-33746438467

Reaz, M.B.I., F. Choong and F. Mohd-Yasin, 2006. VHDL modeling for classification of power quality disturbance employing wavelet transform, artificial neural network and fuzzy logic. Simulation: Trans. Soc. Model. Simul. Int., 82: 867-88. DOI: 10.1177/0037549707077782

Reaz, M.B.I., F. Choong, M. S. Sulaiman and F. Mohd-Yasin, 2007. Prototyping of wavelet transform, artificial neural network and fuzzy logic for power quality disturbance classifier. J. Elect. Power Comp. Syst., 35: 1-17. DOI: 10.1080/15325000600815431

Reaz, M.B.I., F. Mohd-Yasin, M.S. Sulaiman, K.T. Tho and K.H. Yeow, 2004. Hardware prototyping of boolean function classification schemes for loss-less data compression. Proceedings of the Second IEEE International Conference on Computational Cybernetics, Aug-Sep. 30-01, Vienna, Austria, pp: 47-51. DOI: 10.1109/ICCCYB.2004.1437664

Reaz, M.B.I., F. Mohd-Yasin, S.L. Tan, H.Y. Tan and M.I. Ibrahimy, 2005. Partial encryption of compressed images employing FPGA. Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'05), pp: 2385-2388. DOI: 10.1109/ISCAS.2005.1465105

Reaz, M.B.I., Islam, M.T. Sulaiman, M.S.A. and M.A.M. Sarwar *et al*., 2003. FPGA realization of multipurpose FIR filter. Proceedings of the Parallel and Distributed Computing, Applications and Technologies (PDCAT), Aug. 27-29, Chengdu, China, pp: 912-915. DOI: 10.1109/PDCAT.2003.1236448

Reaz, M.B.I., M.I. Ibrahimy, F. Mohd-Yasin, C.S. Wei and M. Kamada, 2007. Single core hardware module to implement encryption in TECB mode. Inform. MIDEM, J. Microelect., Elect. Components Mater., 37: 165-171. http://www.midem-drustvo.si/VOL%2037%282007%293.pdf

Reaz, M.B.I., P.W. Leong, F. Mohd-Yasin and T.C. Chin, 2005. Modeling of data compression using partial matching: A VHDL approach. Proceedings of the 6th World Wireless Congress (WWC), May 25-27, Palo Alto, USA., pp: 411-416. http://delson.org/wwc05/study.htm or http://www.scopus.com/record/display.url?src=sandorigin=ctoandctoId=CTODS_161278959andstateKey=CTOF_161 278960andeid=2-s2.0-20444443530

Reaz, M.B.I., Sulaiman, M.S., Yasin, F.M. and T.A Leng, 2004. IRIS recognition using neural network based on VHDL prototyping. Proceedings of the 2004 International Conference on Information and Communication Technologies: From Theory to Applications, Apr. 19-23, Damascus, Syria, pp: 463-464. DOI: 10.1109/ICTTA.2004.1307832

Victor, J. and A. Dourado, 1997. Wuzzy: A Real-Time Fuzzy Control Tool and Its Application. Proceedings of the 4th. IFAC/IFIP Workshop of Algorithms and Architectures for Real-Time Control, Apr. 9-11, Vilamoura, Portugal, pp: 324-329. DOI: 10.1.1.33.3774, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.33.3774&rep=rep1&type=pdf

Zacharie, M., 2010. Adaptive fuzzy knowledge based controller for autonomous robot motion control. J. Comp. Sci., 6: 1048-1055. DOI: 10.3844/jcssp.2010.1048.1055

Zadeh, L.A., 1972. A rationale for fuzzy control. J. Dyn. Syst. Measu. Cont., 94: 3-4. DOI: 10.1115/1.3426540

Zimmermann, H.J., 1985. Applications of fuzzy set theory to mathematical programming. Inform. Sci., 36: 29-58. DOI: 10.1016/0020-0255