Research on the Hardware RBF Fuzzy Neural based on the FPGA

Bing Xu1,a, Fei Guo2,b

1 GD LONG YUAN Electrical CO.,LTD, Beijing, 100039, China
2 School of Information Engineering, Beijing Institute of Fashion Technology, Beijing 100029, China
aemail: lunwen95@163.com, bemail: iciclexu@sina.com

Abstract

To meet the real-time requirements of industrial field, the hardware FPGA RBF fuzzy neural network is designed and implemented based on the 250000 door Spartan-3E (XC3S250E) chip of Xilinx. First the structure and algorithm of RBF fuzzy neural network is introduced, and then a kind of improved hybrid excitation function approximation algorithm is put forward in order to overcome the difficulties in the hardware design of the neural network, and the hardware co-imitation and timing simulation is done on it. The experimental results demonstrates that, the method has better identification precision and speed, and it is a effective method of hardware implementation of RBF fuzzy neural network, and it lays the foundation for control and image processing based on hardware neural network.

Keywords: RBF Fuzzy Neural; FPGA; Excitation Function; Function Approximation

Introduction

Artificial Neural Network (ANN) with good nonlinear mapping capability has been increasingly applied to a variety of practical systems modeling and control, such as in industrial control, nonlinear modeling and forecasting. Neural network model has two implementations: hardware and software. The practical application of neural network is more than software implementation. he software form has the advantages of larger network size, easy realization, high precision, but unable to meet the requirements of high real-time field, its application is limited more. The hardware implementation of neural network has become a new hot direction with fast processing speed, small size. The reconfigurable computing of FPGA is well adapted to the characteristics of neural network, such as parallel, modular, dynamic adaptability, is the preferred choice of the hardware implementation of neural networks [1][2][3].

A hardware RBF fuzzy neural network based on FPGA is improved and implemented for engineering applications with the example of nonlinear function approximation in this dissertation. A improved hybrid representation method of excitation functions is used in this paper, which has better accuracy and speed.

RBF Fuzzy Neural Network Algorithm

Fuzzy neural network is the product of fuzzy theory and neural networks, it combines the advantages of two, which integrating learning, recognition, association, information processing [4][5][6]. RBF fuzzy neural network structure is shown in Figure 1, and the network is composed of antecedent network and consequent network, which matching antecedent and consequent of fuzzy rules.

The first layer of antecedent network is input layer, each node of this layer directly connected to components x_i of the input vector, n is the number of input variables.

$$f_i^{(1)} = x_i \qquad i = 1,2....n$$

(1)

The second layer of antecedent network is membership computing layer. If membership functions using Gaussian function, then

$$f_{ij}^{(2)} = \mu_i^j = \exp(-\frac{(f_i^{(1)} - c_{ij})^2}{\sigma_{ij}^2}) \qquad i = 1,2...n; j = 1,2...m_i$$

(2)

In Eq.2, m_i is fuzzy segmentation of x_i, c_{ij} and σ_{ij} are center and width of membership function.

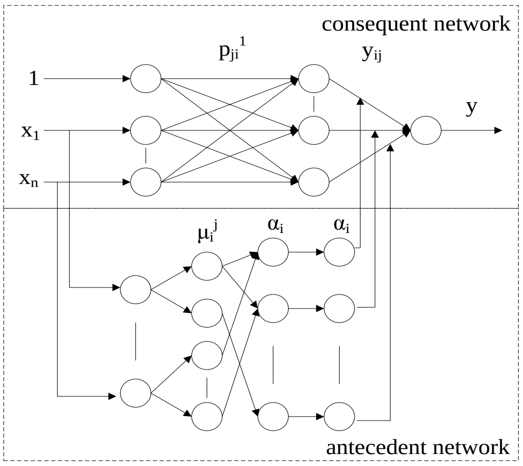


Fig.1 RBF Fuzzy Neural Network Structure

The third layer of antecedent network is fuzzy reasoning layer, to calculate the fitness of each rule.

$$f_l^{(3)} = \alpha_l = \mu_1^1 \mu_2^2 \text{......} \mu_n^l \qquad or$$

$$f_l^{(3)} = \alpha_l = \min(\mu_1^1 \mu_2^2 \text{......} \mu_n^l)$$

(3)

In Eq.3, $l = 1,2,....m, m = \prod_{i=1}^{n} m_i, j = 1,2....m_i$.

The fourth layer of antecedent network is normalized calculation.

$$f_l^{(4)} = \bar{\alpha} l = f_l^{(3)} \Big/ \sum_{l=1}^{m} f_l^{(3)} = \alpha_l / \sum_{l=1}^{m} a_l$$

(4)

Consequent network consists of three layers, consisting of 3 identical structure parallel sub networks, each sub network generates an output.

The first layer:

$$y^{(1)} = x_i$$

(5)

The second layer:

$$y_l^{(2)} = p_{l0}^i + p_{l1}^i x_1 + \text{....} p_{\ln}^i x_n = \sum_{k=0}^{n} p_{lk}^i x_k \qquad , \qquad i = 1,2...r \quad ; l = 1,2...m$$

(6)

The third layer:

$$y^{(3)} = \sum_{l=1}^{m} \bar{\alpha} l y_l^{(2)}$$

(7)

Set the error function as $E = \frac{1}{2} \sum_{i=1}^{r} (y_{ri} - y_i)^2$, where y_{ri} and y_i are respectively expressed the desired output and the actual output. The error back propagation algorithm to calculate the $\frac{\partial E}{\partial p_{li}}$, $\frac{\partial E}{\partial \sigma_{ij}}$, $\frac{\partial E}{\partial c_{ij}}$, then by a gradient optimization algorithm to adjust the $p_{li}, \sigma_{ij}, c_{ij}$.

$$\frac{\partial E}{\partial p_{li}} = -\left(y_{ri} - y_i\right) \bar{a} l x_i$$

(8)

$$\frac{\partial E}{\partial \sigma_{ij}} = -\left(y_{ri} - y_i\right) y_l^{(2)} \bar{\alpha}_l \frac{2\left(x_i - c_{ij}\right)}{\sigma_{ij}^3}$$
(9)

$$\frac{\partial E}{\partial c_{ij}} = -\left(y_{ri} - y_i\right) y_l^{(2)} \bar{\alpha}_l \frac{2\left(x_i - c_{ij}\right)}{\sigma_{ij}^2}$$
(10)

Hardware Design of RBF Fuzzy Neural Network based on FPGA

FPGA realization of neural networks is primarily the implementation of multiplier, accumulators, excitation function and precision of data storage. Studies have shown that fixed-point decimal representation is better than the floating in neural network based on FPGA[7], so 18 bit fixed-point has been used in this paper. FPGA realization of neural networks, multipliers and adders are the most used math module. The MAC operations of RBF neural network is implemented by FPGA own multiplier and adder in this paper[8][9], and set input and output fixed point digital wide of adder and multiplier according to the actual situation of project.

Excitation function is the key to affecting network performance in FPGA design of RBF fuzzy neural network. Sigmoid function has been used as excitation function in this paper, shown as Eq.11.

$$f(x) = \frac{1}{1+e^{-x}}$$
(11)

Excitation function approximation algorithm mainly includes: Taylor series approximation, coordinate rotation, look-up table method, segmentation method. Taylor series and segmentation share less resources but poor accuracy. Coordinate rotation and look-up table method have high precision but low utilization rate of resources [10][11][12].This paper presents a approximation method mixed with segmentation and LUT method to implement the excitation functions as shown follows:

$$f(x)=\begin{cases}-1 & x\le -6\\ LUT & -6<x<-3\\ \dfrac{\frac{1}{2}+\frac{x'}{4}}{2^{|(x)|}} & -3\le x\le 0\\ 1-\dfrac{\frac{1}{2}-\frac{x'}{4}}{2^{|(x)|}} & 0<x\le 3\\ LUT & 3<x<6\\ 1 & x\ge 6\end{cases}$$

(12)

The Sigmoid function is symmetric by point (0,0.5). $|(x)|$ is absolute value of the integer part of x, x' is the fractional part of x which has symbols bit. The output is 1 or -1 when input is greater than 6 or less than -6, the error is less than 0.000152 at this time. In (-3,-6) and (3,6) range, uses LUT method, the step is 0.01,because of the larger errors of liner approximation algorithm. In [-3,0] and (0,3) range uses liner approximation function, Sigmoid function can be implemented by simple shifts and adds shown as Eq,12.

Simulation of Hardware RBF Fuzzy Neural Network based on FPGA

Xilinx's Spartan-3E (XC3S250E) chip with 250000 gates was used in this paper. This chip has high ratio of performance to price in current FPGA chip with using 90nm technology, can provide 1.6M system gates and 376 user IO.

Combined development mode of software and hardware has been used in this paper, which combined by XilinxISE10.1, System Generator and Matlab, to speed up the development progress. System Generator is a designed tool for a digital system provided by Xilinx company. It took some modules embedded in the Simulink library, allowing users to fixed-point simulation in Simulink, generate HDL file etc. The generated file can be invoked, integration, implementation, validation and download in the ISE. The basic System Generator FPGA development and design process figure is shown as follow:

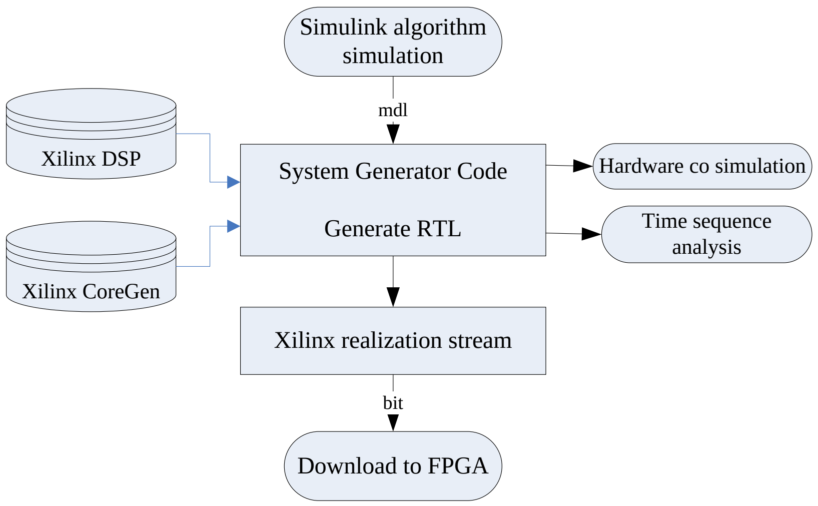


Fig.2 Development Flow Chart by System Generator

$y = x^2 \sin(x) - 5$ is used as the detection function. The hardware Co-imitation model is shown as Fig3.

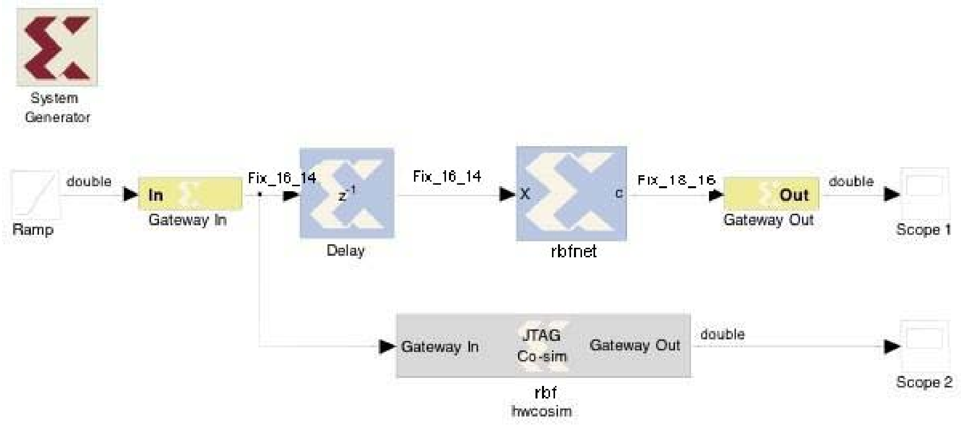
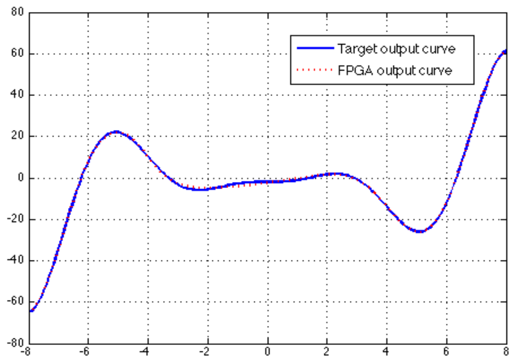


Fig.3 Hardware Co-imitation model

Waveform of RBF fuzzy neural both Hardware simulation and implemented in Matlab are shown in the Fig.4. There is no difference between the two curves, the most error is about 0.38(about 1.2%), That illustrates logic function was been designed correctly.



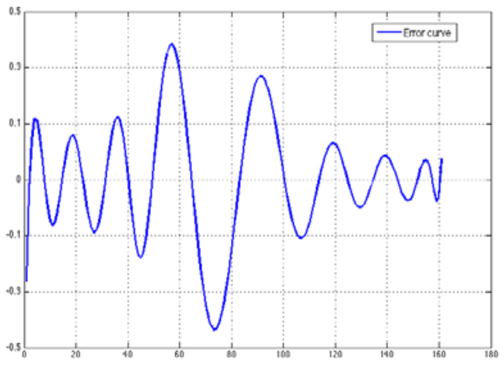


Fig.4 Simulation Curve of Matlab and FPGA

The chip resources occupancy rate after hardware co-imitation, design, integrated and realization of RBF fuzzy neural network is shown as Table1. FPGA configuration bit streams is downloaded to the FPGA after timing simulation, realization the hardware RBF fuzzy neural network based on FPGA.

Table 1 Chip Resources Occupancy Rate

Item	Occupancy Number	Occupancy Rate
Number of Slice Flip Flops	236 out of 4896	5%
Number of 4 input LUTs	1250 out of 4896	26%
Number of occupied Slices	760 out of 2448	31%
Number of Slices containing only related logic	760 out of 778	98%
Number of Slices	0 out of 778	0%

containing unrelated logic		
Total Number of 4 input LUTs	1296 out of 4896	27%
Number used as logic	1250	/
Number used as a route-thru	40	/
Number used as Shift registers	6	/
Number of bonded IOBs	40 out of 108	37%
Number of RAMB16s	3 out of 12	25%
Number of BUFGMUXs	2 out of 24	8%
Number of MULT18X18SIOs	10 out 12	83%

Conclusion

RBF fuzzy neural network is realized based on FPGA in this paper. In the premise of guaranteeing accuracy, through the improved Sigmoid hardware realization method of the activation function, resource utilization is increased. After the hardware and timing simulation, its implementation is downloaded to the FPGA. Through simulation of test functions it can be seen, that the design of FPGA hardware and software realization of the RBF fuzzy RBF networks has little error, and has a very good approximation accuracy. It lays the foundation for other FPGA implementation of neural network models, and also lays the groundwork for hardware-based neural network control and identification, this may be applied on the control of small devices such as handheld devices which has high real-time requirements and resource-constrained scenarios. Meanwhile, the validity and reliability of the hardware implementation of neural networks depends on the hardware realization of data accuracy and activation functions, and the performance can be improved from these two aspects.

Acknowledgement

In this paper, the research was sponsored by the General Program of Science and Technology Development Project of Beijing Municipal Commission of Education (Project No. KM201410012001) and Beijing Key Laboratory of Digital and Interactive Media Project (Project No. KF2013-13) and Youth Innovation Fund Project of science of Beijing Institute Of Fashion Technology (2014AL-29).

References

[1] Ang Li, Qin Wang, Zhancai Li, ET al. Neural networks hardware implementation based on FPGA[J]. Journal of University of Science and Technology Beijing, 2007 29(1), 90-95.
[2] Shoujue Wang, Zhaozhou Li, Xiangdong Chen ET al. Discussion on the Basic Mathematical Models of Neurons in General Purpose Neurocomputer[J]. Acta Electronica Sinica, 2001 29(5), 577-580.
[3] Song Pan, Jiye Huang, Yu Zeng. SOPC Technology Practical Course[M]. Beijing, Tsinghua University Press,2005.
[4] Jing Zhu. Fuzzy Control Theory and System Principle [M]. Beijing, China Machine Press, 2005.
[5] Jun Zhao, Jianjun Chen. Improving strategies on fuzzy neural network control for nonlinear object [J]. Control Theory & Applications, 2010 27(4), 466-472.
[6] Haijun Lin, Zhaosheng Teng, Jinbao Yang. Error compensation for digital temperature sensor based on RBF neural network ensembles-fuzzy weighing output [J]. Chinese Journal of Scientific Instrument, 2011 32(7), 1675-1680.
[7] Jordan L Holt, Jenq-Neng Hwang. Finite Precision Error Analysis of Neural Network Hardware Implementations [J]. IEEE Transactions on Computers, 1993 42(3), 281-290.
[8] A. Muthuramalingam, S. Himavathi, E. Srinivasan. Neural Network Implementation Using FPGA: Issues and Application[J]. International Journal of Information Technology, 2008 4(2),86-92.
[9] Fang Xu, Weiwei Jin, Hong Chen ET al. Hardware implementation method for model predictive control on a FPGA chip[J]. Journal of Jilin University(Engineering and Technology Edition), 2014 4,1042-1050.
[10] Javier Valls. Evaluation of CORDIC Algorithms for FPGA Design [J]. Journal of VLSISignal Processing, 2002 32, 207-222.
[11] Xin Xia, Yonggang Jia, Suzhen Wang. The Realization of Exponential Function ex in RBF Neural Network[J]. Microcomputer Information, 2005 10,145-146.
[12] Xing Wang. Realizing of RBF neural network based on FPGA[J]. Electronic Design Engineering, 2012 20(16),164-166.