International Journal of Electrical and Electronics Engineers
Vol. No.8 Issue 01, January-June 2016
www.arresearchpublication.com
IJEEE
ISSN 2321 - 2055

VLSI IMPLEMENTATION OF KNOWLEDGE-BASED NEURAL NETWORK BACK PROPAGATION

Mrs. B.PRIYANKA[1], Mrs. S.DIVYA[2]

M.E[1], M.E[2]

[1, 2, 3] *Department of Electronics and Communication,*

Ultra college of Engineering and Technology for women, Madurai (India)

ABSTRACT

Neural Networks have a wide range of applications in analog and digital signal processing. Nonlinear activation function is one of the main building blocks of artificial neural networks. Hyperbolic tangent and sigmoid are the most used nonlinear activation functions of Neural Network. This project proposes a knowledge-based neural network (KBNN) modeling approach with new hyperbolic tangent function. The Knowledge-based neural network (KBNN) embeds the existing FPGA analytical models (AMs) into an Neural Network .For fast computation of neuron in Neural Network(NN) we use new approximation scheme for hyperbolic tangent function calculation . The Neural Network (NN) can complement the Analytical Model (AM) according to their needs to provide further increased model accuracy, while maintaining the meaningful trends successfully captured in Analytical Model. The obtained Knowledge based neural network(KBNN) predicts the routing channel width required by circuit implementations on various FPGA architectures , which can be used by architects to quickly and accurately evaluate various FPGA architectures in early development stages. The proposed Knowledge-based neural network(KBNN) coded using Verilog HDL and simulated using Xilinx 12.1. The proposed Knowledge-based neural network (KBNN) with new activation function results in reduction in area, delay, and power in VLSI implementation of artificial neural networks with hyperbolic tangent activation function.

Keyword: Neural Network(NN),Artificial Neural Networks(ANN).

I. INTRODUCTION

1.1 Neural network

Neural networks are complex non-linear models, built from components that individually behave similarly to a regression model. They can be visualized as graph and some sub-graphs may exist with behavior similar to that of logic gates. Although the structure of a neural network is explicitly designed beforehand, the processing that the network does in order to produce a hypothesis (and therefore, the various logic gates and other processing structures within the network) evolves during the learning process. This allows a neural network to be used as a solver that “programs itself”, in contrast to typical
algorithms that must be designed and coded explicitly. Evaluating the hypothesis defined by a neural network may be achieved via feed-forward which amounts to setting the input nodes, then propagating the values through the connections in the network until all output nodes have been calculated completely. The learning can

be accomplished by using gradient descent, where the error in the output nodes is pushed back through the network via back-propagation, in order to estimate the error in the hidden nodes, which allows calculation of the gradient of the cost-function.

1.2 Uses of Neural Network

Neural networks have a wide range of applications in analog and digital signal processing. Hardware implementation of neural networks has been used in applications such as pattern recognition , optical character recognition , test of analog circuits , real-time surface discrimination , smart sensing and identification of heavy ions . Massive and hierarchical networking of the brain seems to be the fundamental precondition for the emergence of consciousness and complex behavior. So far, however, biologists and neurologists have concentrated their research on uncovering the properties of individual neurons. Today, the mechanisms for the production and transport of signals from one neuron to the other are well-understood physiological phenomena, but how these individual systems cooperate to form complex and massively parallel systems capable of incredible information processing feats has not yet been completely elucidated. Mathematics, physics, and computer science can provide invaluable help in the study of these complex systems. It is not surprising that the study of the brain has become one of the most interdisciplinary areas of scientific research in recent years.

However, we should be careful with the metaphors and paradigms commonly introduced when dealing with the nervous system. It seems to be a constant in the history of science that the brain has always been compared to the most complicated contemporary artifact produced by human industry . The nervous system of an animal is an information processing totality. The sensory inputs, i.e., signals from the environment, are coded and processed to evoke the appropriate response. Biological neural networks are just one of many possible solutions to the problem of processing information. The main difference between neural networks and conventional computer systems is the massive parallelism and redundancy which they exploit in order to deal with the unreliability of the individual computing units. Moreover, biological neural networks are self-organizing systems and each individual neuron is also a delicate self-organizing structure capable of processing information in many different ways.

1.3Working process in Neural Networks

The main building blocks needed for hardware implementation of neural networks are multiplier, adder, and nonlinear activation function. A lot of research has been done in digital implementation of multipliers and adders, which can be readily used leaving the nonlinear activation function as the most complex building block.

To solve the implementation problem, approximation methods are generally applied. These methods are based on piecewise linear approximation (PWL), piecewise nonlinear approximation, lookup table (LUT), bit-level mapping, and hybrid methods. Generally,

In PWL approximation methods, the function is divided into segments and linear approximation is used in each segment. This method is used for the hyperbolic tangent and sigmoid function implementation. To implement the neuron, various nonlinear activation functions, such as threshold, sigmoid, and hyperbolic tangent can be used. Hyperbolic tangent and sigmoid are mostly used because their differentiable nature makes them compatible with back propagation algorithm. Both activation functions have an s-shaped curve while their output range is different. Because of the exponentiation and division terms present in sigmoid and hyperbolic tangent activation function, it is hard to realize the hardware implementation of these functions directly.

ii) Another PWL approximation method is introduced. Unlike other PWL methods, the developed method is not based on input domain segmentation and exploits lattice algebra-based centered recursive interpolation (CRI) algorithm.

iii)The piecewise nonlinear approximation is similar to the PWL method with the difference that nonlinear approximation is used in each segment. This method is used to approximate the sigmoid function and scheme is proposed for approximating both sigmoid and hyperbolic tangent.

iv)In the LUT-based methods, input range is divided to equal sub-ranges and each sub-range is approximated by a value stored in LUT. This method is used to implement the hyperbolic tangent.

v)Hybrid methods use a combination of the aforementioned methods which have used a combination of PWL and LUT methods for hyperbolic tangent activation function implementation.

The approximation error present in all methods affects the neural network performance. Research shows that the nonlinear activation function implementation with higher accuracy improves the learning and generalization capabilities of neural networks. However, implementations with higher accuracy require more silicon area and decrease the network operation speed. Therefore, having nonlinear activation function hardware structures with lower area and higher speed for a specified accuracy becomes a key issue.

Artificial neural networks are an attempt at modeling the information processing capabilities of nervous systems. Thus, first of all, we need to consider the essential properties of biological neural networks from the viewpoint of information processing. This will allow us to design abstract models of artificial neural networks, which can then be simulated and analyzed. Although the models which have been proposed to explain the structure of the brain and the nervous systems of some animals are different in many respects, there is a general consensus that the essence of the operation of neural ensembles is “control through communication”. Animal nervous systems are composed of thousands or millions of interconnected cells. Each one of them is a very complex arrangement which deals with incoming signals in many different ways. However, neurons are rather slow when compared to electronic logic gates. These can achieve switching times of a few nanoseconds, Where as neurons need several milliseconds to react to a stimulus. Nevertheless the brain is capable of solving problems which no digital computer can yet efficiently deal with.

1.4 Hyperbolic functions

The hyperbolic functions have similar names to the trigonmetric functions, but they are defined in terms of the exponential function. In this unit we define the three main hyperbolic functions, and sketch their graphs. We also discuss some identities relating these functions, and mention their inverse functions and reciprocal functions.

* define the functions f(x) = cosh x and f(x) = sinh x in terms of the exponential function, and define the function f(x) = tanh x in terms of cosh x and sinh x,

* sketch the graphs of cosh x, sinh x and tanh x,

* recognize the identities $\cosh^2 x - \sinh^2 x = 1$ and sinh 2x = 2 sinh x cosh x,

* understand the meaning of the inverse functions $\sinh^{-1} x$, $\cosh^{-1} x$ and $\tanh^{-1} x$ and specify their domains,

* define the reprocal functions sech x, csch x and coth x.

f(x) = cosh x

The hyperbolic functions cosh x and sinh x are defined using the exponential function ex. We shall start with cosh x. This is defined by the formula

cosh x $=e^{x} + e^{-x}/2$

As x gets larger, e^{x} increases quickly, but e^{-x} decreases quickly. So the second part of the sum $e^{x}/2 + e^{-x}/2$ gets very small as x gets large. Therefore, as x gets larger, cosh x gets closer and closer to $e^{x}/2$. We write this as

cosh x $\approx e^{x}/2$ for large x.

The graph is symmetric about the y-axis, because cosh x = cosh(−x).

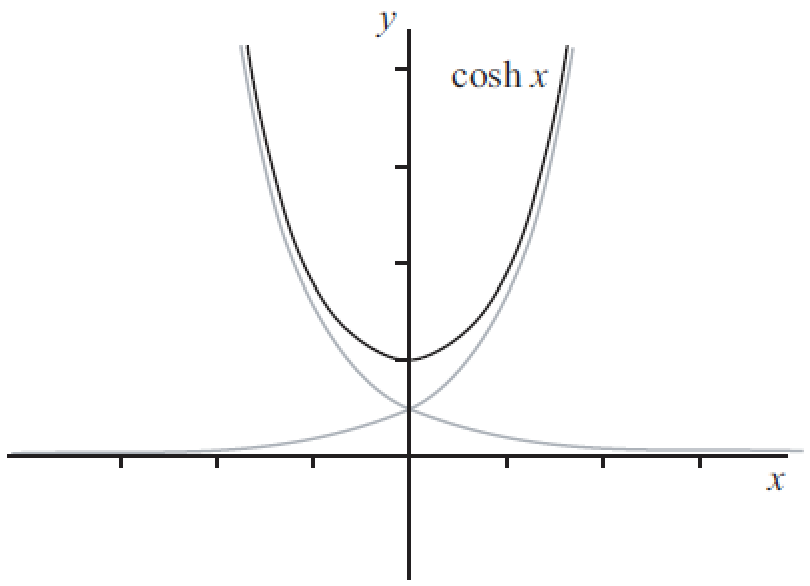

Fig 1: Graph of cosh X

f(x) = sinh x

The hyperbolic function f(x) = sinh x is defined by the formula

sinh x $=e^{x} - e^{-x}/2$

The function satisfies the conditions sinh 0 = 0 and sinh(−x) = −sinh x.

sinh x $=e^{x}/2-e^{-x}/2$

.If that sinh(−x) = −sinh x,then graph sketched as

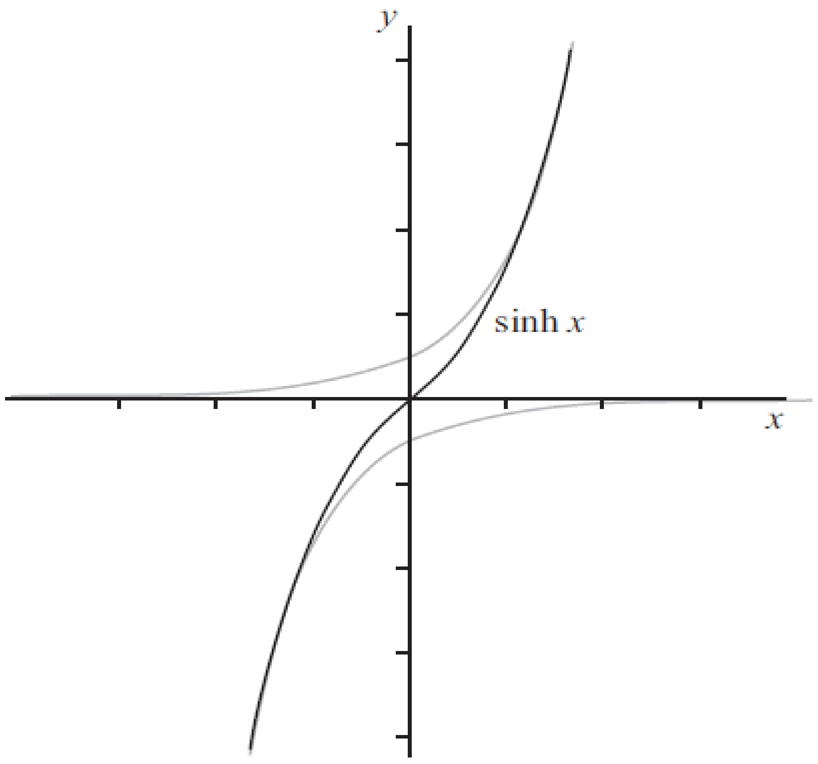

Fig 2: Graph of sinh x

f(x) = tanh x

the hyperbolic function tanh x. In speech, this function is pronounced as ‘tansh’, or sometimes as ‘than’. The function is defined by the formula

tanh x =sinh x/cosh x

on exponential,

tanh x $=e^{x} - e^{-x}/2 \div e^{x} + e^{-x}/2=e^{x} - e^{-x}/e^{x} + e^{-x}$

Take x = 0. We know that sinh 0 = 0 and cosh 0 = 1, so tanh 0 =sinh 0/ cosh 0=0/1= 0 .

As x gets large, sinh x ≈ cosh x, so tanh x gets close to 1:

tanh x ≈ 1 for large x.

That tanh(−x) = −tanh x.

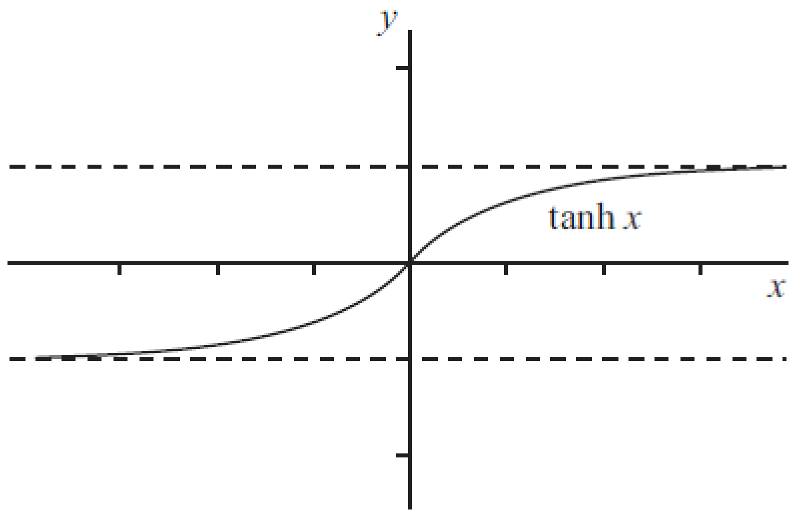

Fig3: Graph of tanh X

II. EXISTING SCHEME

KBNN embeds prior knowledge into the NNs to provide additional information about the problem under modeling .Therefore, a KBNN usually consists of two main parts: 1) a NN and 2) the knowledge in the form of empirical/analytical formulas as shown in Fig. 3. The knowledge formulas encode existing knowledge and need not be accurate and complete. The NN can complement the knowledge formulas by learning the fine bits of the problem. where a knowledge neuron and a three-layer MLP NN are combined.

There are five kinds of neurons in the KBNN structure: 1) the input neurons; 2) the hidden neurons; 3) the output neuron of the three-layer MLP; 4) the knowledge neuron; and 5) the output neuron of the KBNN. The connections between the input and the hidden neurons and between the hidden neurons and the output neuron in the three-layer MLP network have weights

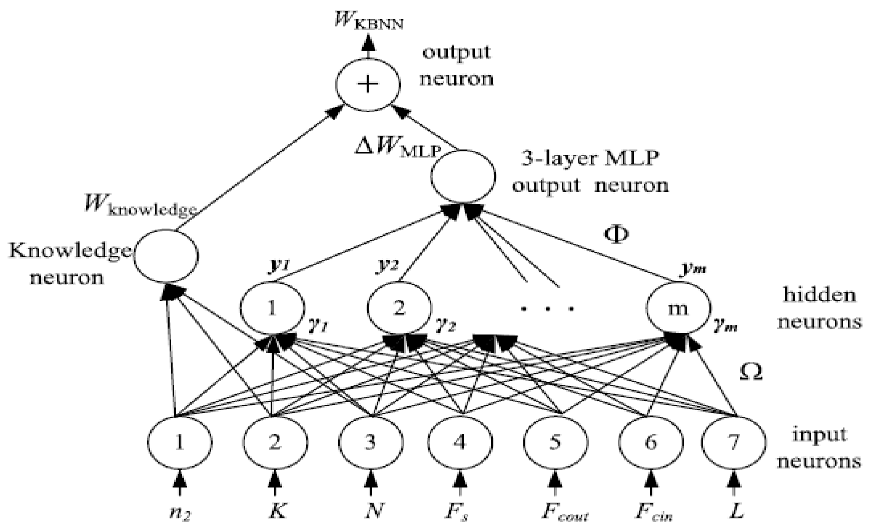

Fig 4: Knowledge based neural network

The KBNN structure for modeling the channel width is shown in Figure , where a knowledge neuron and a three-layer MLP NN are combined. There are five kinds of neurons in the KBNN structure: 1) the input neurons; 2) the hidden neurons; 3) the output neuron of the three-layer MLP4) the knowledge neuron; and 5) the output neuron of the KBNN. The connections between the input and the hidden neurons and between the hidden neurons and the output neuron in the three-layer MLP network have weights, The seven input neurons are for inputting the seven parameters. The input to each hidden neuron is a weighted sum of the input parameters. The final output of the KBNN is the sum of the outputs from the knowledge neuron and those from the MLP network

$W\text{KBNN} = W\text{knowledge} + \Delta W\text{MLP}$

where Wknowledge is the channel width estimated by the existing knowledge formulas. By predicting the difference, the three-layer MLP can complement the error of the existing channel width approximation formulas.

III.PROPOSED SYSTEM

An efficient approximation scheme for hyperbolic tangent function is proposed. The approximation is based on a mathematical analysis considering the maximum allowable error as design parameter. Hardware implementation of the proposed approximation scheme is presented, which shows that the proposed structure compares favorably with previous architectures in terms of area and delay. This structure requires less output bits for the same maximum allowable error when compared to the state-of-the-art. The number of output bits of the activation function determines the bit width of multipliers and adders in the network. Therefore, the tangent activation function results in reduction in area, delay, and power in VLSI implementation of artificial neural networks with hyperbolic tangent activation function.

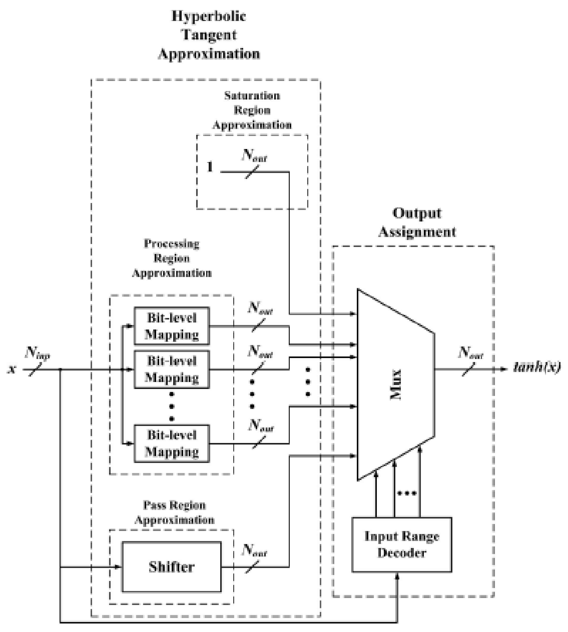

Fig 5: Tangent Activation Function

The hardware is composed of two main blocks, including hyperbolic tangent approximation and output assignment.

This block is composed of three main blocks to approximate the hyperbolic tangent function in all three regions, including saturation, processing, and pass region. General arithmetic operations in each region can be described as follows.

1) Pass Region: In this region, fractional part of input is passed to the output. Based on a shift to left by *N*out−*N* *f* bits before passing the input to output is required.

2) Processing Region*:* For inputs in the processing region, a bit-level input mapping is required. The number of bit-level mapping blocks required is equal to the number of input ranges in this region. For each input range in the processing region, log*N* 2 bits after *N*one bit of input should be mapped to output bits using the bit-level mapping. Using log*N* 2 bits after *N*one bit covers all sub-ranges. The number of sub-ranges (*N*) is calculated while the output of each sub-range is found . The bit-level mapping can be implemented using a combinational circuit.

3) Saturation Region Approximation*:* In this region, hyperbolic tangent function is approximated by the maximum value represented by output bits, and can be realized by setting all output bits to one.

A sigmoid function is a mathematical function having an "S" shape (sigmoid curve). Often, sigmoid function refers to the special case of the logistic function shown in the first figure and defined by the formula

$$S(t) = \frac{1}{1+e^{-t}}.$$

Other examples of similar shapes include the Gompertz curve (used in modeling systems that saturate at large values of t) and the ogee curve (used in the spillway of some dams). A wide variety of sigmoid functions have been used as the activation function of artificial neurons, including the logistic and hyperbolic tangent functions. Sigmoid curves are also common in statistics as cumulative distribution functions, such as the integrals of the logistic distribution, the normal distribution, and Student's *t* probability density functions.

The same bilinear transformation used to find out sigmoid,sin,cos and other elementary functions

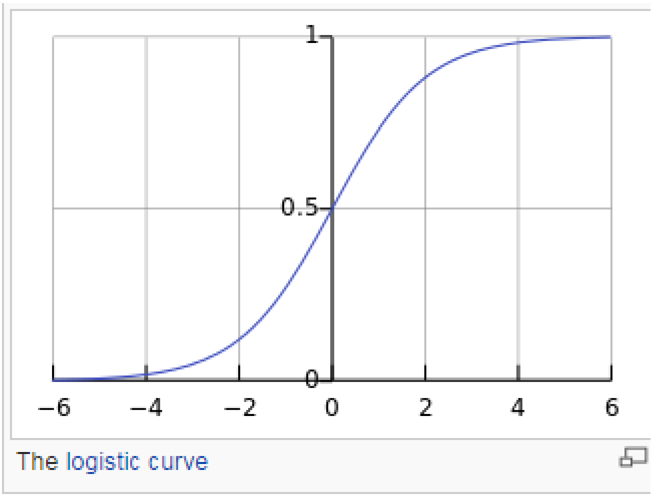

Fig 6: Logistic curve

IV.RESULT COMPARISONS

The proposed has been simulated and the synthesis report can be obtained by using Xilinx ISE 12.1i. The various parameters used for computing existing and proposed systems with Spartan-3 processor are given in the table .

S.No	Parameter	Existing	Proposed
1	Slice	46	6
2	LUT	32	12

Table 1: Comparison Table of Proposed with Existing system

The Figure given below is shown that there is a considerable reduction in time and area based on the implementation results which have been done by using Spartan-3 processor. The proposed algorithm significantly reduces consumption when compared to the existing system.

V. RTL SCHEMATIC

After performing the synthesize process, the RTL schematic has been created automatically based on the functionality. The routing between the different cells can be viewed clearly by this schematic

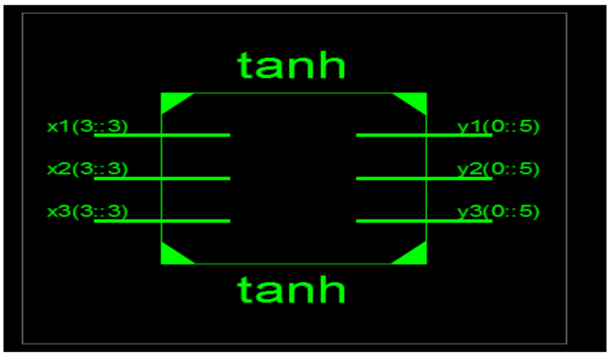

Fig 7a: Logic block module of tanh

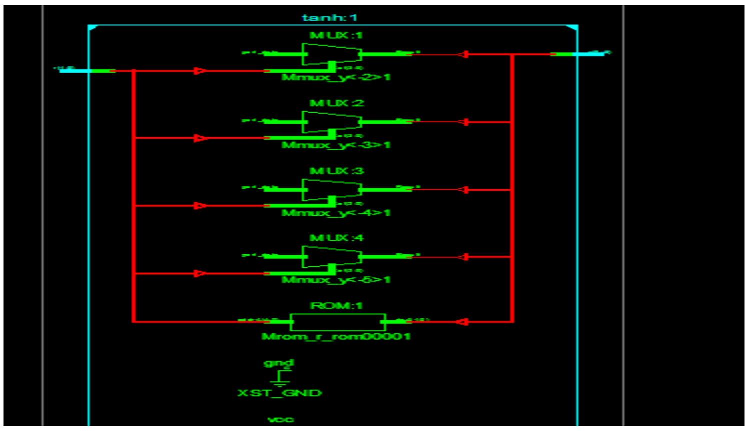

Fig 7b: RTL Schematic of neural network

VI. CONCLUSION

A new approximation scheme for hyperbolic tangent was proposed in this project. The proposed approximation scheme is based on a mathematical analysis considering maximum allowable error as a design parameter. Based on the proposed approximation scheme, a hybrid architecture for hardware implementation of hyperbolic tangent activation function was presented. The synthesis results showed that the proposed structure compares favorably to the previously developed architectures in terms of area, delay, and area × delay. Hashing trick and back propagation methods are to be used in the future work in order to reduce the computational complexity.

REFERENCES

[1] Qiang liu,Ming gao,Quijun zhang, "Knowledge based neural network model for FPGA logical architecture development," in 2015.

[2] W. M. Fang and J. Rose, "Modeling routing demand for early-stage FPGA architecture development," in Proc. 16th ACM/SIGDA Int. Symp.Field Program. Gate Arrays, Monterey, CA, USA, 2008, pp. 139–148.

[3] A. M. Smith, S. J. E. Wilton, and J. Das, "Wirelength modeling forhomogeneous and heterogeneous FPGA architectural development," inProc. ACM/SIGDA Int. Symp. Field Program. Gate Arrays, Monterey, CA, USA, 2009, pp. 181–190.

[4] J. Das, A. Lam, S. J. E. Wilton, P. H. W. Leong, and W. Luk, "Ananalytical model relating FPGA architecture to logic density and depth,"IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 19, no. 12, pp. 2229–2242, Dec. 2011.

[5] V. K. Devabhaktuni, B. Chattaraj, M. C. E. Yagoub, and Q.-J. Zhang,"Advanced microwave modeling framework exploiting automatic model generation, knowledge neural networks, and space mapping," IEEE Trans. Microw. Theory Techn., vol. 51, no. 7, pp. 1822–1833, Jul. 2003.

[6] H. Kabir, L. Zhang, M. Yu, P. H. Aaen, J. Wood, and Q.-J. Zhang,"Smart modeling of microwave devices," IEEE Microw. Mag., vol. 11,no. 3, pp. 105–118, May 2010.

[7] T. Zhang and M. Nakamura, "A hybrid neuro-inverse control approach with knowledge-based nonlinear separation for industrial nonlinear system,"IEEE Trans. Control Syst. Technol., vol. 13, no. 5, pp. 840–846, Sep. 2005.

[8] B.-Z. Wang, D. Zhao, and J. Hong, "Modeling stripline discontinuities by neural network with knowledge-based neurons," IEEE Trans. Adv.Packag., vol. 23, no. 4, pp. 692–698, Nov. 2000.

[9] A. Rahman, S. Das, A. P. Chandrakasan, and R. Reif, "Wiring requirementand three-dimensional integration technology for field programmable gate arrays," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 11, no. 1, pp. 44–54, Feb. 2003.

[10] I. Kuon and J. Rose, "Exploring area and delay tradeoffs in FPGAs witharchitecture and automated transistor design," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 19, no. 1, pp. 71–84, Jan. 2011.

[11] C. Chiasson and V. Betz, "COFFE: Fully-automated transistor sizing for FPGAs," in Proc. Int. Conf. Field-Program. Technol. (FPT), Kyoto,Japan, Dec. 2013, pp. 34–41.

[12] A. M. Smith, G. A. Constantinides, and P. Y. K. Cheung,"FPGA architecture optimization using geometric programming," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 29, no. 8, pp. 1163–1176, Aug. 2010.

[13] E. Ahmed and J. Rose, "The effect of LUT and cluster size on deepsubmicron FPGA Performance and density," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 12, no. 3, pp. 288–298, Mar. 2004.

[14] F. Li, Y. Lin, L. He, D. Chen, and J. Cong, "Power modelling and characteristics of field programmable gate arrays," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 24, no. 11, pp. 1712–1724, Nov. 2005.

[15] L. Cheng, F. Li, Y. Lin, P. Wong, and L. He, "Device and architecture cooptimization for FPGA power reduction," IEEE Trans. Computer-Aided Design Integr. Circuits Syst., vol. 26, no. 7, pp. 1211–1221, Jul. 2007.

[16] Q. Liu, J. Ma, and Q. Zhang, "Neural network based pre placement wirelength estimation," in Proc. Int. Conf. Field-Program. Technol. (FPT), Seoul, Korea, Dec. 2012, pp. 16–22.