A Novel Sigmoid Function Approximation Suitable For Neural Networks on FPGA

Peter W. Zaki[1], Ahmed M. Hashem[1], Emad A. Fahim[1], Mostafa A. Masnour[1], Sarah M. ElGenk[1], Maggie Mashaly[1] and Samar M. Ismail[2]

[1]Faculty of Information Engineering and Technology (IET), German University in Cairo (GUC), Egypt.

[2]Faculty of Engineering and Technology, Egyptian Chinese University (ECU), Egypt.

***Abstract*— Artificial Neural Networks (ANN) is invading a lot of practical applications in our life nowadays. One of the main blocks of ANN is the activation function block, which is based on the sigmoid function. The hardware implementation of sigmoid function is a challenging task; hence some approximation techniques were previously developed. In this paper, a novel sigmoid approximation technique is proposed and compared with previous techniques, on both simulation and hardware design levels. They are applied in a neural network application, where the proposed technique showed high accuracy compared to the original sigmoid function. Moreover, the different techniques are implemented on Virtex 7 FPGA using IEEE 754 Floating Point representation to achieve high precision, where the proposed approximation consumed the least hardware area utilization compared to previous works for clock frequency of 358.166 MHZ.**

Keywords*— *Neural Networks, Sigmoid, Floating Point, FPGA.

I. Introduction

Artificial neural networks (ANN) research had emerged and developed in the last few years. It had been used in different applications like speed estimation, imaging processing and pattern recognition [1, 2]. Software-based ANN research is most commonly popular than ANN hardware implementations as they are simpler and their time-to-market is less than their hardware counterparts. However, the software-based ANN has slower execution speed compared to the hardware implementation [3-5]. This directed researchers to hardware implementations of ANN, which can be designed in both the digital or the analog domain. Nevertheless, the digital system implementation is more preferable for its better accuracy, easier testability, higher flexibility and lower noise sensitivity compared to analog system design [4].

ANN is a way to imitate how the human brain works [6], an ANN network is shown Fig.1, where each node represents a neuron. Each neuron calculates the information that is passed to it by other neurons using mathematical activation functions following equation (1). The summation Y is the input of the activation function.

$$Y = \sum X * W \tag{1}$$

The most widely used activation function is the sigmoid function expressed as follows:

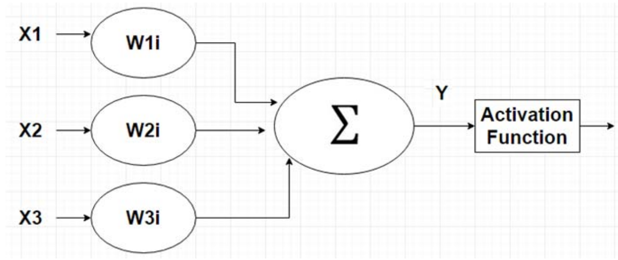

Fig. 1. Neural Network Structure.

$$f(x) = \frac{1}{1+e^{-x}} \tag{2}$$

The hardware implementation of the nonlinear sigmoid function is a challenging task since it consumes a large hardware area [5]. Hence, researchers were directed to use various approximations to reach the same output as the sigmoid function [3-7].

In order to gain high accuracy and precision in the hardware implementation, Floating point (FP) representation is being used in this work to represent data. Following the IEEE 754 FP, numbers can be represented as double precision (64 bits) or single-precision (32 bits) format. Figure 2 shows the single-precision format having a Sign bit (1 bit), Exponent (8 bits), and Mantissa (23bits) [6,7].

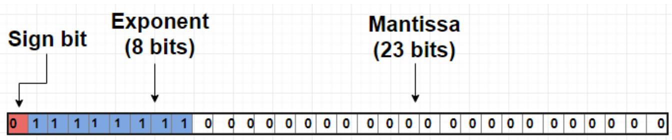

Fig. 2. Floating Point Single Precision representation

A novel approximation technique for the sigmoid function is proposed in this paper and compared with other techniques of previous works [8,9]. The comparisons are performed by applying all the techniques in an ANN application, an image recognition algorithm and calculate their accuracies. Moreover, all the techniques are hardware implemented on FPGA comparing their area utilization and complexity.

The paper is organized as follows: Section II discusses the different approximation techniques. Section III presents the Matlab simulation results of all the techniques, while the hardware implementations are detailed in Section IV, and Section V concludes this work.

II. SIGMOID APPROXIMATION TECHNIQUES

A. Technique A

An approximation technique of the sigmoid function was introduced in [5]. The approximation function used was expressed as follows:

$$f(x) = \frac{1}{2}\left(\frac{x}{1+|x|} + 1\right), \quad (3)$$

where x is the sum of product between the inputs of the network and the weight associated to each input.

This function was implemented on FPGA to decrease the running time of the learning process [5]. The functional units used in the hardware design were adder and divider.

B. Technique B

This technique was presented in [6], offering an activation function used in the back propagation process of the artificial Neural Networks. It was an attempt to improve generalization and reduce overtraining on mislabeled or irrelevant data. A network with the new activation function was implemented, trained and compared with the standard activation function in an attempt to improve the area under curve of the receiver operating characteristic in biological classification task. The original function followed this equation:

$$f(x) = \big(\sigma(x+b) + \sigma(x-b)\big) - 1 \quad (4)$$

Where $\sigma(x)$ is the same as equation (2) and b is a variable parameter which was then set to 2 [6].

Modified Technique B

In this work, a modification is applied for Technique B. the activation function is modified as follows to increase its simulation accuracy:

$$f(x) = \frac{(\sigma(x+b) + \sigma(x-b))}{2} \quad (5)$$

The reason behind the modification is that when Technique B was applied to the application presented in this work, which is image recognition, the algorithm of the neural network needs an activation function of range [0,1], but the original technique B is of range [-1,1].Thus, the accuracy of Technique B was very low as shown in Table I, while the modified technique gave higher accuracy. Figure 3 shows the difference between the approximated sigmoid function presented by Technique B [6] and the modified technique.

C. Proposed Logarithmic Technique

This function is a novel approximation technique for the original sigmoid (2), which is a piecewise function based on a logarithmic function and addition. The proposed Technique is expressed as follows:

For $x \geq 0$:

$$f(x) = \frac{\log_2(2x + 0.486)}{8} + 0.63 \quad (6)$$

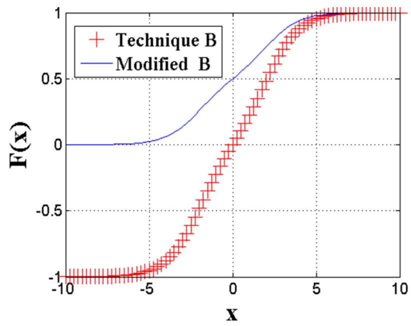

Fig. 3. Technique B vs Modified Technique B.

For $x < 0$:

$$f(x) = \frac{-\log_2(-0.5x + 0.008458)}{16} + 0.07 \quad (7)$$

Any other logarithmic function with base of powers of 2 as 4, 8, 16 can be applied instead of $\log_2 x$.

The advantage of this technique is in the simplicity of hardware utilization. The beauty of the log function is that is handled in a more suitable form for digital system taking the advantage of shifting and IEEE 754 representation. The functional units used by the circuit are just one adder and one shifter. The shifter is used to handle the divisions and the multiplication as it is always involved with numbers of power of 2.

The calculations of the log is based on the separation of the exponent from the mantissa $'m'$ then approximating the log of the mantissa as follows:

$$Log(m) = m - 0.92 \qquad m \in [1,1.84[\quad (8)$$

$$Log(m) = 0.5 * m \qquad m \in [1.84,2[\quad (9)$$

III. SIMULATION RESULTS

Figure 4 shows the responses of all the techniques explained in the previous section. They are drawn versus the actual sigmoid using Matlab.

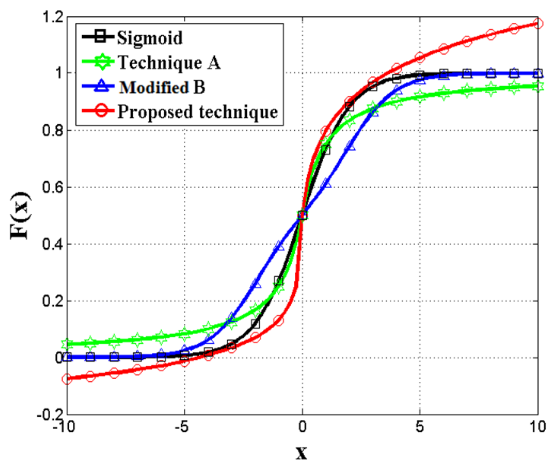

Fig. 4. All Techniques versus Sigmoid function.

To test the functionality and efficiency of these approximation techniques, they are all applied in an image recognition application based on back propagation algorithm for neural networks to detect a hand-written digit recognition. The training set consists of 4000 training examples, subset from the data set provided by [10], each training example is a 20x20 pixel grayscale image. The testing set was 1000 examples, used for testing the accuracy of the given algorithm using the different techniques as an activation function versus the actual sigmoid function. Table I presents a quantitative comparison of the accuracy of the presented application. The table shows that the modified technique B has higher accuracy than the others, Technique A is the least, and the proposed Technique is as accurate as the actual sigmoid proving its efficiency.

TABLE I. MATLAB ACCURACIES

Activation Function	Actual Sigmoid	Techniq. A [5]	Techniq. B [6]	Modified Techniq. B	Proposed Techniq.
Accuracy	91.8%	77.8 %	58.9%	92.4 %	**91.7 %**

IV. FPGA HARDWARE IMPLEMENTATION

The hardware design of all the techniques is presented in this section and implemented Virtex 7 FPGA kit. Single precision IEEE 754 floating point representation is used for all data representations.

A. Technique A Implementation

Figure 6 shows the hardware design block diagram of Technique A. The design consists of adders, dividers and absolute calculation block, realizing equation (3). The FPGA output is drawn versus the Matlab output shown in Fig. 6, where they are exactly the same with no errors.

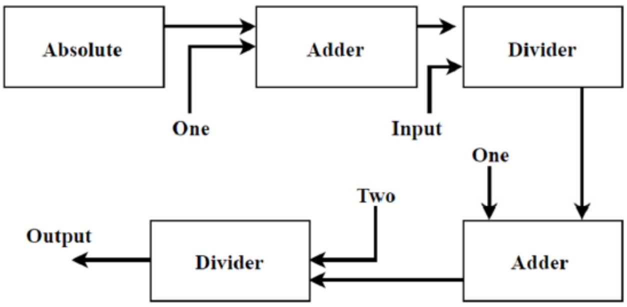

Fig. 5. Block Diagram for Technique A.

B. Modified Technique B Implementation

The design of this technique is presented Fig. 7, where adders, exponential and reciprocal blocks are used. The code starts first by implementing $\sigma(x+b)$ and $\sigma(x-b)$ then the division by 2 is done by checking the exponent if it's equal to 0 then the mantissa is shifted one unit to the right and if it's not equal to zero one is subtracted from the exponent. The design is fully pipelined. Figure 8 shows the FPGA output versus the Matlab output fitting on each other with no errors.

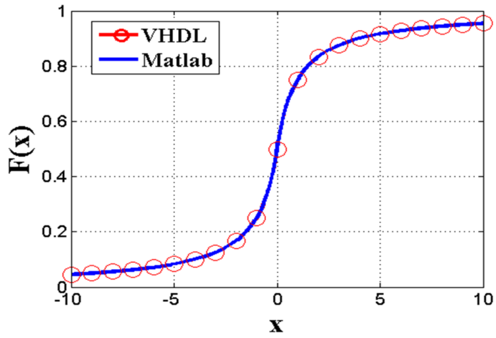

Fig. 6. Matlab versus Harware output for Technique A.

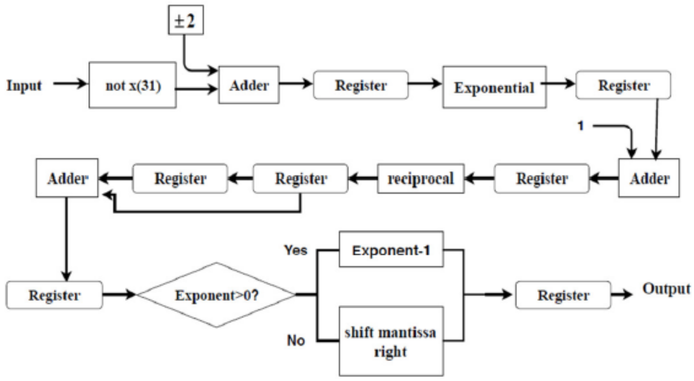

Fig. 7. Block Diagram for Modified Technique B.

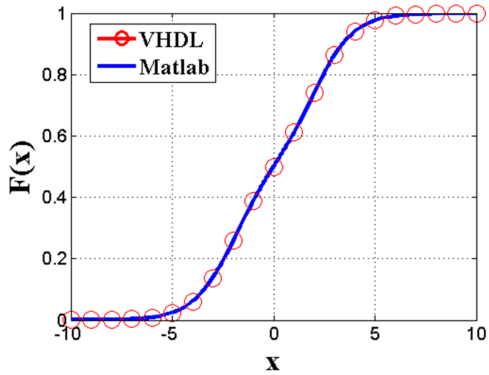

Fig. 8. Matlab versus Harware output for Modified Technique B.

C. Proposed Logarithmic Technique Implementation

Figure 9 shows the hardware design block diagram of the proposed Technique. It is a simple logarithmic algebraic function but on the other hand its VHDL implementation is designed avoiding the use of complex hardware units as detailed in Fig. 10. The main blocks used are the comparator, adder and shifter blocks. The VHDL output of the proposed technique is plotted versus the Matlab curve, as showin in Fig. 10 having a maximum error of 1%.

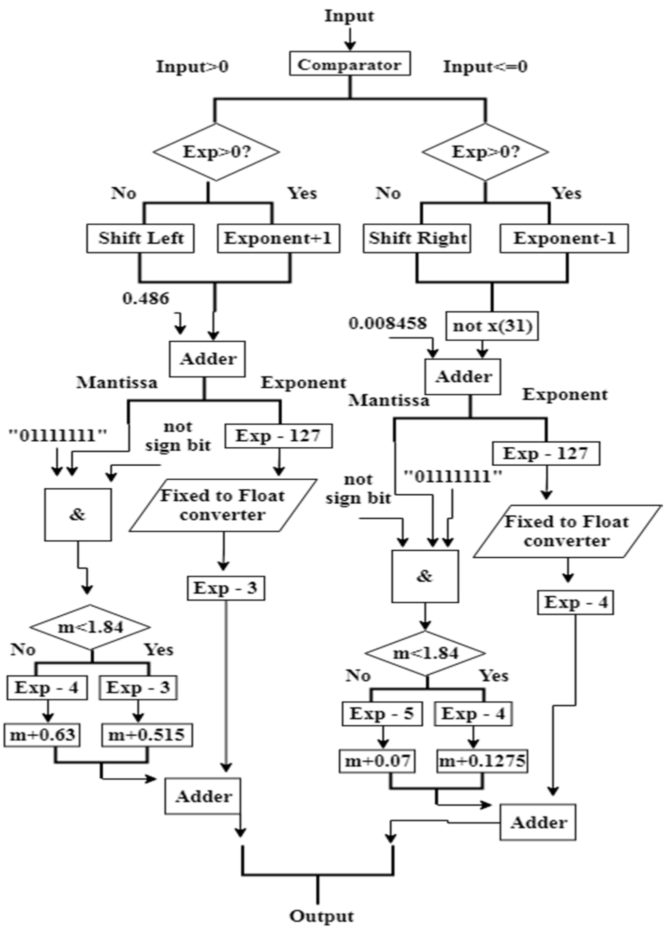

Fig. 9. Block Diagram of the proposed technique.

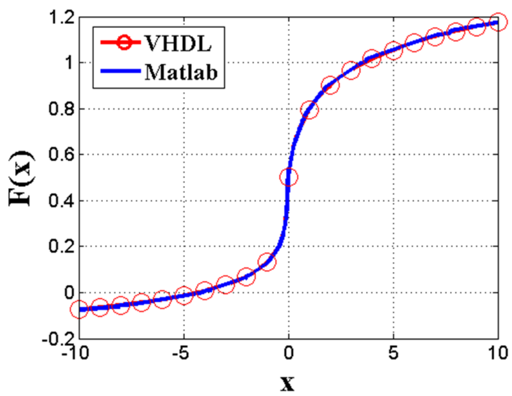

Fig. 10. Matlab versus Harware output for proposed technique.

Table II presents the hardware utilization comparison between the actual sigmoid, Technique A, Modified Technique B and the proposed Logarithmic Technique. It is obvious that the proposed technique is the best regarding hardware utilization as it consumed the least percentage of all the available units compared to the other techniques.

TABLE II. THE USED HARDWARE RESOURCES

Resources Available	**Actual Sigmoid**		**Technique A**		**Modified Technique B**		**Proposed Log Technique**	
	Used	%	Used	%	Used	%	Used	%
LUT 303600	1232	0.41	1034	0.34	1338	0.44	**363**	**0.12**
LUT RAM 130800	176	0.13	50	0.04	176	0.13	**16**	**0.01**
FF 607200	1282	0.21	1835	0.3	1476	0.24	**566**	**0.09**
DSP 2800	11	0.39	2	0.07	11	0.39	**2**	**0.07**
Freq. (MHz)	356.88		358.166		358.166		**358.166**	

V. CONCLUSION

A novel approximation technique for the sigmoid function, used as an activation function for neural networks applications is presented. The proposed technique is compared to previous works regarding simulation accuracy for an image recognition application based on back propagation algorithm. All the techniques were hardware implemented on Xilinx Virtex 7 FPGA, where the proposed technique excels with the least hardware area utilization with clock frequency of 358.166 MHz.

REFERENCES

[1] Da Silva, Ivan Nunes, et al. "Artificial neural networks." Cham: Springer International Publishing, 2017.

[2] Walczak, Steven. "Artificial neural networks." Advanced Methodologies and Technologies in Artificial Intelligence, Computer Simulation, and Human-Computer Interaction. IGI Global, 40-53, 2019

[3] H. K. Ali and E. Z. Mohammed, "Design Artificial Neural Network Using FPGA," IJCSNS International Journal of Computer Science and Network Security, vol. 10, no. 8, pp. 88-92, 2010.

[4] A. Muthuramalingam, S. Himavathi and E. Srinivasan, "Neural Network Implementation Using FPGA: Issues and Application," International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering, vol. 2, no. 12, pp. 2802-2808, 2008.

[5] A. Tisan, S. Oniga, D. Mic and A. Buchman, "Digital Implementation of the Sigmoid function for FPGA circuits," ACTA TECHNICA NAPOCENSIS Electronics and Telecommunications, vol. 50, no. 2, pp. 15-20, 2009.

[6] S. Sahin, Y. Becerikli and S. Yazici, "Neural Network Implementation in Hardware Using FPGAs," in In International Conference on Neural Information Processing, Berlin, 2006.

[7] M. Panicker and C.Babu, "Efficient FPGA Implementation of Sigmoid and Bipolar Sigmoid Activation Functions for Multilayer Perceptrons," IOSR Journal of Engineering, vol. 2, no. 6, pp. 1352-1356, 2012.

[8] T. M.Jamel and B. M. Khammas, " Implementation of a Sigmoid activation function for neural network using FPGA," 13th Scientific Conference of Al-Ma'moon University College, 2012.

[9] Bonnell, Jeffrey A., "Implementation of a New Sigmoid Function in Backpropagation Neural Networks."(2011). Electronic Theses and Dissertations. Paper 1342.https://dc.etsu.edu/etd/1342.

[10] The MINST database of handwritten digits Yann LeCun, Courant Institute, NYU Corinna Cortes, Google Labs, New York. Christopher J.C.Burges,MicrosoftResearch,Redmond,http://yann.lecun.com/exdb/mnist/