Evolution in Electrical and Electronic Engineering Vol. 2 No. 2 (2021) 951-958
© Universiti Tun Hussein Onn Malaysia Publisher's Office

EEEE

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/eeee
e-ISSN : 2756-8458

Performance Analysis of Different Neuron Activation Function Implementations on FPGA using Artificial Neural Network

Nurizzati Aishah Mohamed[1], Mohamad Hairol Jabbar[1]*

[1]Faculty of Electrical and Electronic Engineering,
University Tun Hussein Onn Malaysia, 86400 Parit Raja, Johor, MALAYSIA

*Corresponding Author Designation

DOI: https://doi.org/10.30880/eeee.2021.02.02.113
Received 04 July 2021; Accepted 17 October 2021; Available online 30 October 2021

Abstract: Artificial Neural Network (ANNs) is a part of a computing system that design to process data information in the same way as the human brain working. It has been trained to process a large amount of data in the network and successfully applied in many applications such as pattern recognition. The high computational technique and processing power required to develop Artificial Neural Network architecture allow it to be embedded into Field Programmable Array (FPGA). This allows for even more parallelism in the network implementation. Furthermore, the activation function selection is also significant in Artificial Neural Network as it influences the network's performance. This work aims to implement different neuron activation functions on an Artificial Neural Network architecture. The design has been chosen to implement on FPGA, specifically the Virtex-7 board based on Xilinx FPGA. Xilinx Vivado software was used for synthesizing and implementing the architecture to analyse the performance of each neuron activation function. At the end, the comparison of different activation functions reveals that there is not a significant difference between the activation function tested. Every activation function has almost given the same impact on a network once it has been trained.

Keywords: Artificial Neural Network, Activation Function, FPGA

1. Introduction

Over the last few decades, the neural network is getting popular and recognize for many applications including embedded systems. The evolution in the embedded system field has created a new era dimension in recognition method from the digital text of handwritten transform into security access systems and others [1]. In other words, this system can convert text into a documented format and making it more convenient without necessarily typing the text back into the system. Many parameters need to be optimized for having optimal architecture to meet the specification of application requirements. One of the main parameters in this project is the architecture of a neural network that

**Corresponding author: hairol@uthm.edu.my* comprises one or more neurons attached to each other and it can be said as one of the largest components inside the neural network [2]. Neuron is a function based on the inputs and excites activation function as the desired output. Those activation functions have different characteristics and give impact to the overall performance which is crucial in every stage of operation. Apart from that, with the growth of field-programmable gate array (FPGA) in an embedded system, it becomes essential and very helpful in the neural network to perform any calculation for embedded operation [3]. This project will analyse the hardware implementation of different neuron activation functions.

2. Research Methods

2.1 System Overview

This project was performed according to the flowchart in Figure 1. According to the flowchart, the workflow started with developing VHDL code for the neural network, proceeding with the code's simulation, running the synthesis and implementation, and finally obtaining the result. The workflow will continue to progress until all the activation function types are implemented in the software tools.

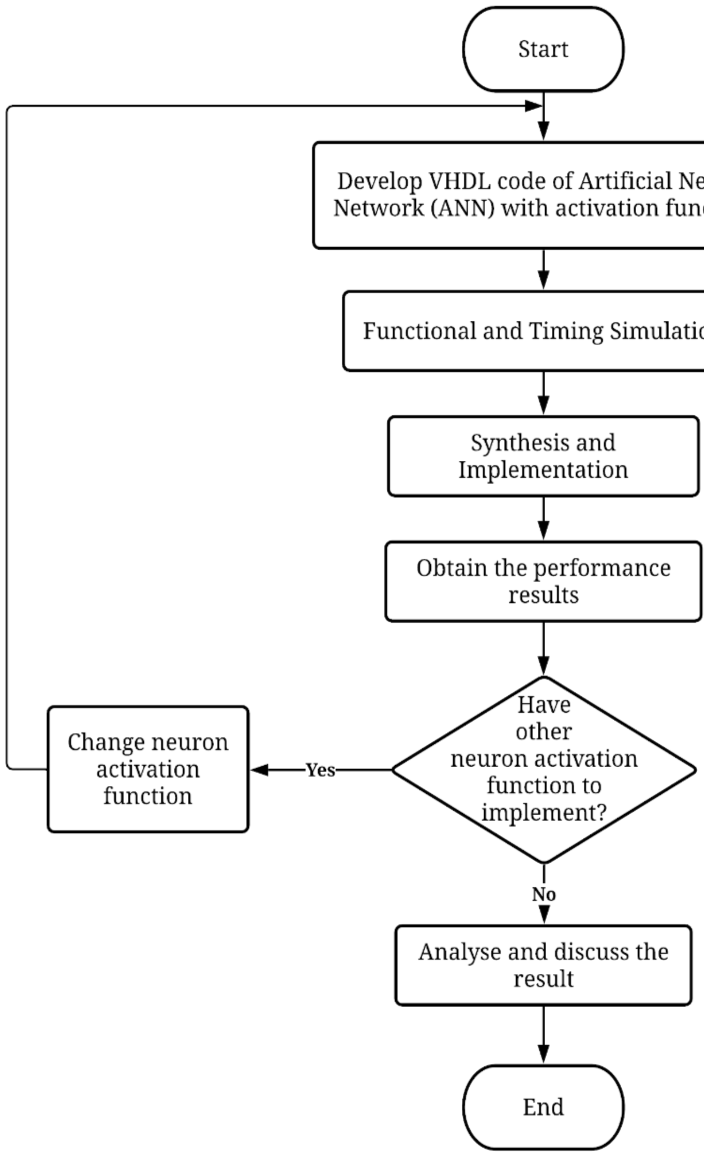
Start
Develop VHDL code of Artificial Neural Network (ANN) with activation function
Functional and Timing Simulation
Synthesis and Implementation
Obtain the performance results
Have other neuron activation function to implement?
Yes
Change neuron activation function
No
Analyse and discuss the result
End

Figure 1: Workflow of project

2.2 Design Tools

Xilinx Vivado Design Suite was used to carry out all the activities in this project. Xilinx was verified a long year ago for its ability to invent a field-programmable array gate (FPGA). The purpose of choosing this software is as stated in the objective: to synthesize and implement FPGA performance. Furthermore, this software provided a variety board of FPGA and easy to implement the design FPGA configuration. The Virtex-7 device is chosen as the type of FPGA board with high performance and low power consumption. Design tools and its function for Artificial Neural Network is listed in Table 1.

Table 1: Design tools and its function for Artificial Neural Network

No.	Component	Version	Function
1	Xilinx Vivado Design Suite	2020.1	To synthesis and implement the performance of activation functions in the neural network.
2	Xilinx Virtex-7 FPGA	XC7VX690T	The target FPGA device for implementation.

2.3 Implementation of Activation Function

The importance of applying activation function in the neural network is to introduce the non-linearity in the system. The proposed activation used for this project is based on Table 2.

Table 2: Variation of the activation function

No.	Type of Activation Function	Description
1.	Sigmoid Function	Produce output value in binary in-between '1' and '0'. However, the performance of the Sigmoid becomes slow when it approaches a negative value.
2.	Tanh Function	The output produce is similar to Sigmoid which is between '1' and '0'. The difference is that Tanh is more robust as it always centers at value zero, making the positive value stronger.
3.	ReLU Function	The output range produce is from zero towards infinity. But, when the value becomes negative, its immediately produce zero output.
4.	Threshold Function	One of the simplest activation functions. The produce output value is between 0 and 1.

2.4 Optimizing Network

In this work, the architecture of Artificial Neural Network is composed of hidden layers h1, h2, ..., hn, that interconnected to each other as demonstrated in Figure 2.

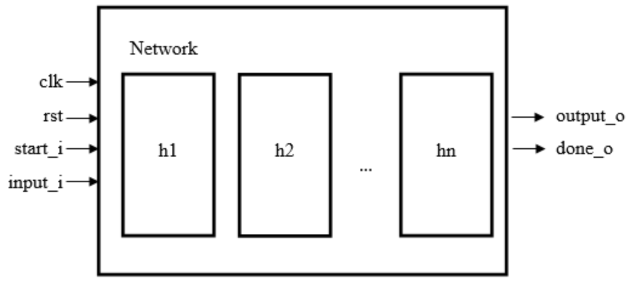

Figure 2: The design of neural network

Each hidden layer composed several neurons n1, n2, ..., nk, as shown in figure. The description of Figure 3 shows that hidden layers correspond only to the arrangement of neurons and do not constitute a component by themselves.

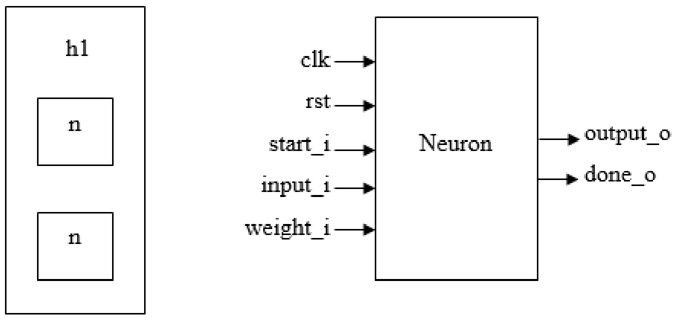

Figure 3: The arrangement of neurons in the hidden layers

2.5 VHDL Design for Artificial Neural Network

The development of Artificial Neural Network is fully using the code in hardware description language (VHDL) by expressing the descriptions of neuron, activation function and so on in either behavioural or structural. Artificial Neural Network required high computational power to distribute the information throughout the whole network structure. Meanwhile, FPGA is design to be parallelism. So, the high parallelism and computational power required numerical precision in the network design to produce more efficient and accurate design. In this case, numerical operations are performed by using fixed-point representation as the input data. The width of fixed-point data will be for about 8 to 16 bits length. The coding development of Artificial Neural Network in VHDL is listed in Table 3.

Table 3: The development of network in VHDL

Design Sources	Description
types.vhd	Configuration of the size of the integer part and fractional part used in the fixed point in all component. It also contains some useful functions for the conversion of the values in VHDL. Other than that, it responsible for setting the desired accuracy for the network.
act_func.vhd	Description about activation function architecture. It contains several activations to be implemented such as ReLu, Threshold, Tanh and Sigmoid.
neuron.vhd	Description about artificial neurons. A bit description about the activation function also shown in this file.
network.vhd	Description about the neural network architecture with its interconnection such as neuron. The input is directly connected to the input neurons. The network is controlled by the connections of the neurons in the hidden layers.

Furthermore, the neuron architecture for this project is controlled by a simple finite state machine (FSM) as shown in Figure 4. The description of the state machine is stated as Table 4:

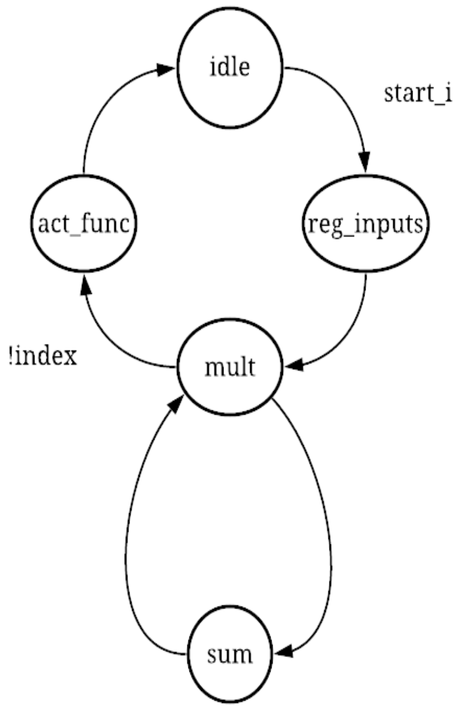

Figure 4: FSM responsible for controlling neuron

Table 4: The description of a simple state machine

State Machine	Description
idle	It is the initial state, and it waits for the start_i signal to indicate the start of computation.
reg_inputs	The input signals x0, x1, ..., xm; the bias b is added to the lump sum accumulator, the number of inputs m of the neuron is registered.
mult	Its function to check whether the current computation has already been executed for all m inputs. Then, its proceeds to the act_func state. Otherwise, it multiplies the input m by its respective synaptic weight wkm, where m>1. Therefore, $$(wkj \times j) + bk$$
sum	The result of the previous operation is added with the sum. The index is responsible for storing the input currently being operated. Therefore, $$\sum_{j=1}^{m} (wkj \times j) + bk$$
act_func	The activation function chosen for neuron is defined in the architecture of act_func.vhd. The result in the lump sum accumulator is used as input to the activation function and its output is instantiated to the main output of the neuron. Therefore, $$yk = \varphi\left(\sum_{j=1}^{m} (wkj \times j) + bk\right)$$

3. Results and Discussion

All the performance of each activation function was presented after being synthesized and implemented. The results are performed to study Artificial Neural Network's implementation with different activation function variations on the targeted FPGA board. The aspect performances are focusing on the operation speed, area utilization and power consumption.

3.1 Operation Speed

The comparison of operation speed for each activation function is done by calculating its maximum frequency. The maximum frequency operation involves the value of worst negative slack (WNS) in the timing summary. From the maximum frequency, it can analysis which network has a higher speed.

$$Maximum\ Frequency = \frac{1}{(T - WNS)} \qquad Eq. 1$$

Where, $T = target\ clock\ period;$

$WNS = worst\ negative\ slack$

Worst negative slack (WNS) describes all the values in timing paths for maximum delay analysis. The timing constraints are set at the period 10 ns for every activation function during implementation, so $T = 10ns$.

Table 5 shows that the value of maximum frequency for activation function Threshold, Tanh, and Sigmoid have almost the same value. From Table 5, the ReLU activation function has the highest speed among the other activation functions by having 894.45 MHz as the maximum frequency. Meanwhile, Sigmoid having the slowest speed compared with ReLU, Threshold and Tanh.

Table 5: Comparison of speed for each activation function in Artificial Neural Network

Type of Activation Function	Worst Negative Slack (WNS) (ns)	Maximum Frequency (MHz)	Ranking (speed)	Percentage Difference of Each Activation Function Performance
ReLU	8.882	894.45	1 (Fastest)	41.80%
Threshold	8.154	541.71	3	3.90%
Tanh	8.242	568.83	2	8.49%
Sigmoid	8.079	520.56	4 (Slowest)	0%

3.2 Area Utilization

Area utilization in FPGA is greatly influenced the performance of a program. Table 6 shows the analysis of the area utilization in terms of Lookup Tables (LUT), Flip Flop (FF), Block RAMs (BRAM), Input-Output buffers (IO) and Global Buffer (BUFG). Overall, this project network design for each activation function does not consume a lot of area utilization in the Xilinx Virtex-7. However, based on the comparison in Table 6, Tanh utilizes the most area in the FPGA followed by Sigmoid, ReLU and lastly Threshold. Thus, Threshold is the most efficient in designing a network as it implements low area utilization.

Table 6: Comparison of FPGA utilization

Activation Function	Resources	Utilization	Available
ReLu	LUT	930	433200
	FF	589	866400
	BRAM	12	3600
	IO	99	1000
	BUFG	10	32
Threshold	LUT	908	433200
	FF	529	866400
	BRAM	12	3600
	IO	99	1000
	BUFG	9	32
Tanh	LUT	1464	433200
	FF	687	866400
	BRAM	18	3600
	IO	99	1000
	BUFG	10	32
Sigmoid	LUT	1281	433200
	FF	556	866400
	BRAM	12	3600
	IO	99	1000
	BUFG	9	32

3.3 Power Consumption

The consumption of power in the network for each activation function is shown in Table 7. Generally, all different activation function has almost consumed the exact value of power. From this view, the Tanh activation function consumes a lot of power which is 0.395W. Meanwhile, the Threshold consumes the least power which is 0.344W.

Table 7: Comparison of Power Consumption

		ReLU Power (W)	Threshold Power (W)	Tanh Power (W)	Sigmoid Power (W)
Dynamic	Clocks	0.001	0.001	0.002	0.002
	Signals	0.013	0.008	0.020	0.011
	Logic	0.007	0.006	0.014	0.010
	DSP	0.005	0.004	0.010	0.003
	I/O	0.019	0.001	0.024	0.007
	sTotal	0.045	0.020	0.070	0.034
Static		0.0325	0.325	0.325	0.325
Total		**0.37**	**0.344**	**0.395**	**0.359**
Percentage Difference of Each Activation Function Performance		**7.03%**	**0%**	**12.91%**	**4.18%**

4. Conclusion

In a neural network, the activation function is one of the most critical parameters to consider. Comparing different activation functions reveals that there is no significant difference between the activation function tested. Every activation function has almost given the same impact on a network once it has been trained. However, it is crucial to select the best activation function to develop a network as each activation function has it owns characteristic.

Overall, the objective of this work was achieved, the network architecture has been successfully developed with the hardware description language (VHDL). The performance of the design has been accessed through synthesized and implemented on Xilinx Vivado. Lastly, the comparison of each activation function has been carried out in terms of operational speed, area utilization and power consumption.

Acknowledgment

The authors would like to thank the Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia for its support.

References

[1] J. Križaj, V. Štruc, S. Dobrišek, and W. Day, “Handwritten Digit Classification,” *Int. J. Adv. Robot. Syst.*, vol. 9, no. June 2014, 2012

[2] Z. Chen, “Handwritten digits recognition,” *Proc. 2009 Int. Conf. Image Process. Comput. Vision, Pattern Recognition, IPCV 2009*, vol. 2, pp. 690–694, 2009, doi: 10.31142/ijtsrd8384

[3] N. J. Cotton, B. M. Wilamowski, V. Vodyanoy, T. A. Roppel, and V. P. Nelson, “A neural network implementation on embedded systems,” thesis, 2010