International Journal of Computer Applications (0975 - 8887)
Volume 184 - No.12, May 2022

Design of Low Power Artificial Hybrid Adder using Neural Network Classifiers to Minimize Energy Delay Product for Arithmetic Application

C. Pakkiraiah
Research Scholar, SVUCE, S.V.University
Tirupati, Andhra Pradesh, India

R.V.S. Satyanarayana
Professor, SVUCE, S.V.University
Tirupati, Andhra Pradesh, India

ABSTRACT

The binary adder is a primary computational block in many arithmetic processors and digital signal processing applications. Artificial Neural Network (ANN)s validate a group of neuron particles to configure a feed-forward neural network, a perceptron that executes functionally accomplished basic logic gate operations and provides a re-programmable, re-configurable, extensible computing system and FPGA board to form ANNs and make analytical findings. Different methodologies are analyzed, such as ANNs, which are one of the most encouraging subsequent innovative designs, and researchers are exploring and enhancing different trade-off characteristics such as delay, dynamic power dissipation, and area. With the constraint of purposeful computational time, the use of the intended style of software implementation provides the advantages of easy programming and low cost. Hardware implementation can be used to control the limits of software perception in neural networks. The proposed neural network hybrid adder's major goal is to design a low-energy-delay device with a small footprint. In this paper, first consider the design of basic logic gates using neural networks followed by 1-bit hybrid full adder circuits, which are the primary components in computing. The hybrid adder designs are simulated and synthesized using Xilinx Vivado for the XC7Z020clg400-1 configurable device and implemented on the FPGA ZYBOZ7 board. The implementation findings reveal that, in comparison to Proposed Full Adder and single layer perceptron hybrid adder, the proposed multi-layer perceptron hybrid adder design achieved substantial refinement, with reductions of (60%, 60%) and (30.8%, 28.2%) in dynamic power dissipation and EDP, respectively.

Keywords

ANN hybrid adder, ANNs, Dynamic Power dissipation, EDP, Perceptron

1. INTRODUCTION

Field Programmable Gate Array (FPGA) based Neural Network (NN)s could potentially furnish tunable trade-offs between complex systems such as delay, area, and power.The design of low power full adder is discussed in [1].There has been a trend in recent techniques toward the expansion of programmable modules, which processors use to give flexibility and execution. The expansion of cyberspace automation led to the myriad and exponential growth of binary data. [2] Shows how neural networks can be used to create logic gates. User-friendly electronic providers have boosted demand for multi-operand adders with the shortest latency and lowest power consumption to be incorporated into today's portable systems. For Nano scale NNs, the nanoparticle computing architecture [3] was created. In particular, deep learning neural networks seem widely acquired as they manifest exactness for several analytical categorizations on tasks (e.g., ML, AI, speech, etc.). Exploring and building logic gates and half-adder modules [4] explains the concept of neural networks. In [5], neural networks are used to show the re-configurable constant coefficient multipliers. Models of NNs are gaining extensive recognition as future advanced architectures for computing systems. To date, the exploration of neural network models has primarily focused on conceptual research and computer simulations. The re-configurable threshold logic block described in [6] is used to create an arithmetic logic unit. The fast training [7] algorithm is shown to provide a Boolean logic minimization-based approach to the creation of deep neural networks. However, actual assurance for the utilization of the models lies in limited hardware in specific microelectronic systems. Simulations of macro scale networks on consecutive computers are unfortunately slow, and only with tailored hardware can we desire to realize neural network designs with momentum sufficient for applications. The concept of a neural network is thoroughly defined in [8]. The FPGA is used to implement the machine learning models [9].The most optimistic perspective for accomplishing an electronic neural net is to invent specifically intended VLSI chips. With today's unification compactness, macro scale numerous elementary processors can be integrated on a single chip. [10] Investigates time multiplexing and approximation computing. The arithmetic complexity is reduced using the quick winogral [11] technique. The desire to design quick-witted systems, accompanied by the benefit of fewer delays in computation, has manifested through simulation the competence of ANN to delineate, prototyping, and categorizes linear systems. [12] Describes an end-to-end FPGA-based convolution neural network accelerator with all layers mapped on a single chip. Deep neural network techniques are performed using advanced FPGA technologies [13] . Digital performance is more desired as it has the dominance of giant perfection, superior reiteration, lower sus-

Fig. 1. Design flow of ANN's on FPGA

ceptibility, better verifiable, higher pliability, and affinity with other types of processors. [14] Proposes a design process for a ring oscillator based on a re-configurable exclusive OR gate. [15] Describes the mapping of convolution nets onto re-programmable FPGAs. FPGA is an acceptable hardware for NN execution as it takes care of parallel processing of the neurons in each layer and provides pliability in re-programmable. Neural networks are relatively worthwhile for simulating digital systems. [16] Discusses the operation, training, and overview of ANNs. The process of behavioral conversion of digital circuits into neural networks is discussed in [17]. A digital system's output is an assembly of logic modules that can be arranged in any desired pattern using a layout. [18] Uses a complementary metal oxide semiconductor transistor and created ANNS to present a simplified technique. In [19], [20], It proposes a step-by-step method for employing ANNs to create logic gates and XOR gates. There is a contingency plan to build ANNs on FPGA in a cost-effective and competent manner using FPGA re-configurable. The goals of this study are to 1) provide a brief overview of subsystem ANN implementation in FPGA hardware. 2) Takeover characteristics and concerns that serve as guiding principles in such realisations. 3) Investigate the most effective ways to use FPGA reprogrammable to implement ANNs.[21], [22] As a first step, a basic node accelerator for convolution neural networks is implemented on a mid-range FPGA. [23] Describes the creation of fundamental logic gates and an XOR gate using perceptron's and threshold elements. The Figure 1 illustrates a systematic representation for imposing ANNs onto FPGA. The Maestro, who interprets supplied ANN specifications into Verilog codes, is the most important step in this representation. Verilog code is converted to a net list via HDL synthesis. The net list data is used in the configuration file to implement the design on the FPGA board. This study, which is a considerable extension of [1, 2], is organized as follows: The essential preliminaries required for current research happenings are given in Section II. Section III explores more of the use of neural networks to create logic gates. The development of three artificial neural network adders is discussed in Section IV. Section V contains the simulation results of three ANN adders. The performance metrics evaluation is discussed in Section VI, followed by the conclusion.

2. PRELIMINARIES

The basics of neuron structure, ANNs, and trade-offs are explained in this section.

2.1 Basics of Artificial Neural Network

The basic structure of neuron is shown in Figure.2.The cell body, also known as the soma, gives diagnostic information and aids in the development of neurons. Dendrites are an add-on to the cell body that is unique. They collect information from other cells and pass it on to the cell body. The axon, also known as a nerve fiber, is responsible for separating nerve impulses from the soma. A synapse is a structure that allows nerve impulses to pass between two nerve cells. Artificial neurons mimic the biological neuron's soma, dendrites, axon, and synapse, which allude to nodes, inputs, outputs, and weights, respectively.The block diagram of ANN is

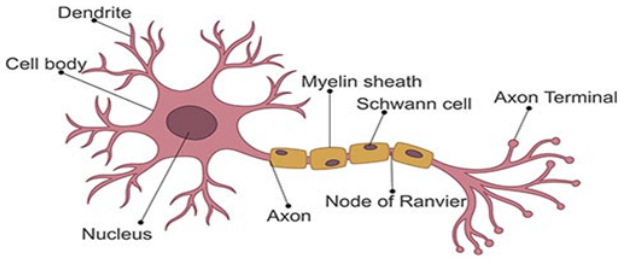

Fig. 2. Basic structure of neuron

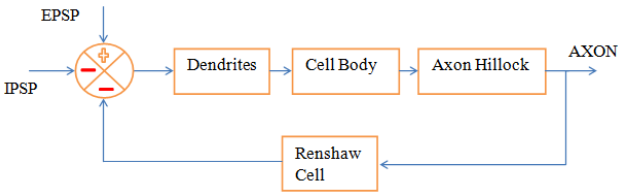

Fig. 3. Block diagram of ANN's

shown in Figure.3. An artificial neuron's mathematical model comprises three main components. (1) Input signals are given weights. 2) To combine weighted input signals, a summing block is employed. 3) A "squashing function" for limiting an output's range. In essence, whatever a neuron does is add up the components of numerous inputs, assign a weighted number to each, and display an output when the sum of the weighted inputs reaches a predetermined threshold. Excitatory (positive) or inhibitory (negative) inputs are possible in NNs. A negative feedback loop emerged in Renshaw's cell, which connects two nerve cells. As a result, it acts as a preventive against extreme agitation. The ANN mathematical equation is as follows:

$$Z = f(y) \tag{1}$$

Each artificial neuron contains inputs, weights, and bias values. The equation for each artificial neuron is as follows:

$$y = \sum_{i=1}^{n} X_i W_i + b \tag{2}$$

Where Xi denotes the input values, Wi denotes the input weights, and b denotes the bias value. Artificial neuron models are simplified versions of biological neurons at their core. A perceptron is the popular name for this artificial neuron. As shown in the diagram, a typical perceptron will have numerous inputs, each of which is individually weighted.The mathematical model of ANN is shown in Figure 4. There are various constraints to implementing the ANN on hardware.

1) Synapse

• Network wiring: As the number of neurons increases, so does the number of synapses.

• Synaptic weight: Weights must be defined precisely to ensure the algorithm's correct convergence.

2) Neuron

• Neuronal state: It is necessary to perform a weighted input summation.

• Activation function: Decisions based on a high threshold function.

From equ.3, if the sum of the inputs exceeds a certain threshold,

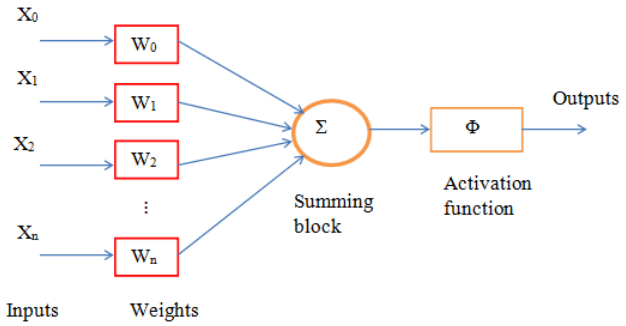

Fig. 4. ANN mathematical model

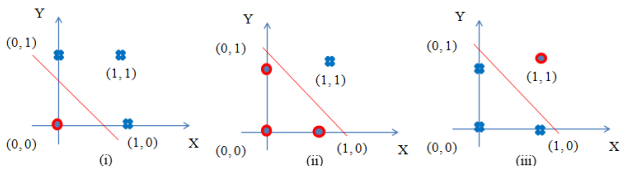

Fig. 5. NN classifiers for ((i) OR gate (ii) AND gate (iii) NAND gate

the activation function will generally output "1," otherwise "0."

$$Z = \begin{cases} f(y), & \text{if } y \geq \Phi \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

2.2 Power dissipation, delay, and cell count

A digital circuit's power dissipation is consists of two elements: a static component and a dynamic component. The leakage current flowing via the supply voltage causes static power consumption in MOS transistors (Vdd). The load capacitance, operating frequency, supply voltage, and switching activity factors leads to the dynamic power dissipation of MOS transistors. The time it takes for a signal to flow from input to output is known as the propagation delay of a digital circuit. The propagation delay of a digital circuit can be used to measure its speed. The number of cells necessary to create any logic circuit is defined as the area of this study.

3. DESIGN OF BASIC GATES USING NN

All electronic digital circuits and embedded systems are made possible by digital logic gates, which are the fundamental building blocks. In contrast to conventional based capabilities, the decades-long obsession with threshold components and logic has resulted in a wide range of functionally enhanced capabilities. The threshold logic elements can be used as the basis for ANN squashing functions. In terms of structure, ANNs are based on the feed-forward network principle, which involves layer-by-layer processing from the network's input to output. The neural network can categorize based on the conditions of AND, OR, and NAND gates using the straight line equation illustrated in the Figure 5. A neural network classifier for two input OR, AND, and NAND gates is shown in the Figure 5. In Figure 5, the red circle represents a false condition, while the blue cross represents a valid condition for fundamental logic gates. The NN-based output computations for all logic gates are built using the perceptron approach for weights and the NN classifier[2] for output logic functionality. When the threshold function is utilized as the neuron output function and binary input

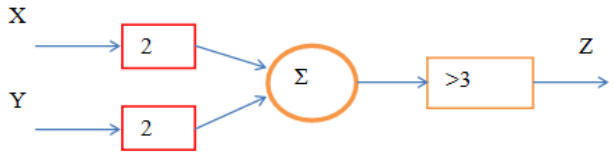

Fig. 6. Mathematical model of AND gate

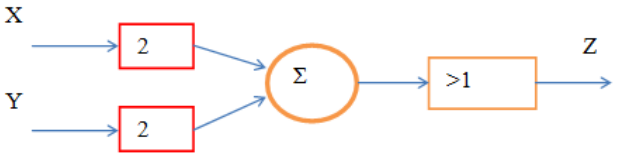

Fig. 7. Mathematical model of OR gate

values of 0 and 1 are assumed, the basic logic gates AND, OR, NAND, and NOT of two variables can be formed by choosing appropriate weights and threshold values.

3.1 Neural Network AND logic unit

The AND gate's functional table is used as training data for the neural network in this part. The neuron's X and Y inputs are connected. To design the AND function, the threshold on Z and the weights can be set according to the following requirements. To match the needs of its functional table, the NN AND logic unit[2] has a training data weight of 2 and a threshold value of 3. The Mcculloch-Pitts neuron model is the same as threshold logic. It calculates the weighted sum of its inputs and compares them to the threshold value provided in equation 4.The ANN model of equ.4 is shown in Figure 6.

$$Z = \begin{cases} 1, & \text{if } 2X + 2Y > 3 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

3.2 Neural Network OR logic unit

This section explains how to make a two-input OR gate with a NN classifier. The following is the condition for the NN OR logic unit with NN classifier:

$$Z = \begin{cases} 1, & \text{if } 2X + 2Y > 1 \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

Equ.5 defines the design connection for the NN OR logic unit, which is a fundamental gate. The NNOR logic unit has a training data weight of 2 and a threshold value of -1 to match the conditions of its functional table. The NN OR gate output equals one when at least one of the inputs equals one; otherwise, the output equals zero. The ANN model of a neural network OR logic unit is shown in Figure 7. Equ.5 is used to check the function table of the two-input OR gate in Table 1. The ANN model of a two-input OR gate based on equ.5 is shown in Figure 7.

3.3 Neural Network NAND logic unit

NN functional table of two input OR gate

$$Z = \begin{cases} 1, & \text{if } 2X + 2Y < 3 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

Table 1. NN functional table of two input OR gate

Inputs		Weights		Sum= X*W1+Y*W2	Threshold function (Sum > 1)	Output
X	Y	W1	W2			
0	0	2	2	0	false	0
0	1	2	2	2	true	1
1	0	2	2	2	true	1
1	1	2	2	4	true	1

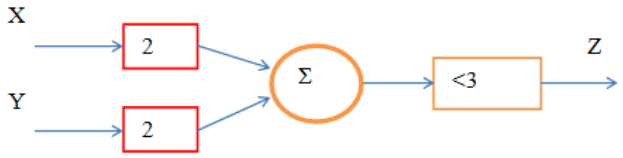

Fig. 8. Mathematical model of NAND gate

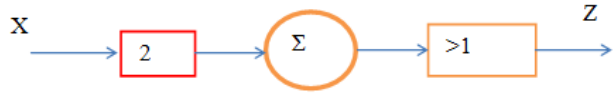

Fig. 9. Mathematical model of NOT gate

The two inputs of the NN NAND logic unit design have weights of -2 and -2, respectively, to match the requirements of its truth table, and a threshold value of 3. The ANN model for equ.6 is shown in Figure 8.

3.4 Neural Network NOT logic unit

The single input NOT gate is defined using the perceptron technique. The functional equation for the NN NOT gate is as follows:

$$Z = \begin{cases} 0, & \text{if } 2X > 1 \\ 1, & \text{otherwise} \end{cases} \tag{7}$$

If X equals zero, Z equals one; otherwise, it equals zero, and the outcome is the same as a single input NOT gate. The ANN model of the NN NOT gate is shown in Figure 9.

3.5 Neural Network MUX logic unit

A digital multiplexer is a combinational circuit that chooses binary data from one of several input lines based on a single input line and reroutes it to a single output. The output of 2×1 MUX is stated as the following Boolean equation:

$$Y = (\bar{S})I_0 + SI_1 \tag{8}$$

Where, I_0 I_1 are inputs. S is select line. The design connection for the NN MUX logic unit, which is a fundamental data selector, is defined by equ.8.

4. ANN HYBRID ADDER DESIGNS

One of the most commonly used component in computing processes is full adder. Multiplication, subtraction, and division are all handled by this module's logic operations. The 1-bit full adder circuit is a critical component in the design of application-specific integrated circuits.

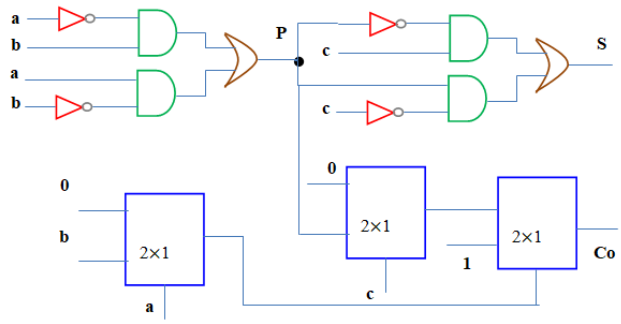

Fig. 10. Logic diagram of CHA

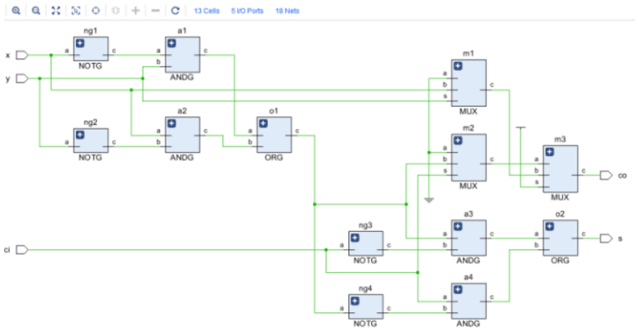

Fig. 11. RTL diagram of ANN_{CHA}

4.1 Conventional Hybrid Adder (CHA) using ANN

The conventional binary adder is divided into three sections. Module 1 is an exclusive OR between the two inputs a and b. Furthermore, the output of the first module is used as an input for modules 2 and 3. The three logical expressions that can be rewritten are as follows:

$$P = \bar{a}b + a\bar{b} \tag{9}$$

$$S = \bar{P}c_i + P\bar{c_i} \tag{10}$$

$$C_o = Pc_i + ab \tag{11}$$

The logic diagram of a conventional hybrid full adder formed from logical equ.9, 10, and 11 is shown in Figure 10. The conventional artificial neural network adder is made up of basic neural network logic units like AND, OR, and NOT. The conventional hybrid adder is thus named as the ANN Conventional Hybrid Adder (ANN_{CHA}). The ANN_{CHA} RTL diagram is shown in Figure 11. The ANN_{CHA} features thirteen cells, five input/output ports, and sixteen nets, as shown in Figure 11.

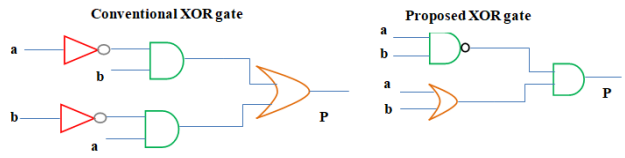

Fig. 12. Logic diagram of conventional and proposed XOR gate

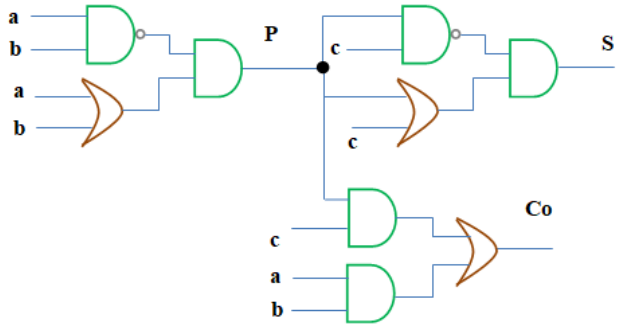

Fig. 13. Logic diagram of PFA[1]

4.2 Proposed Full Adder without Neural Network

The XOR module of the proposed binary adder circuit is responsible for the majority of the dynamic power consumption of the unified adder circuit. In adders and compressors, XOR gates help with huge area and dynamic power dissipation. As a consequence, the logical equation of the XOR module is rearranged using the logic decomposition method.

The logic diagrams of the conventional XOR gate and the proposed XOR gate are shown in Figure 12. Without affecting functionality, the proposed XOR gate uses only three logic gates instead of the typical XOR gate's five logic gates. The Boolean expression of a proposed full adder is determined using the full adder circuit's functional table. A three-input digital logic circuit that creates a sum and an output carry is known as a full adder.

$$P = (\bar{a} + \bar{b})(a + b) \tag{12}$$

$$S = (\bar{P} + \bar{c_i})(c_i + P) \tag{13}$$

$$C_o = Pc_i + ab \tag{14}$$

The logic diagram of a Proposed Full Adder based on Boolean equ.12,13 and 14 is shown in Figure 13.

4.3 Single Layer Perceptron Hybrid Adder (SLPHA) design

The proposed single layer perceptron hybrid adder circuit was built by using the basic logic gates NN classifiers on a conventional hybrid full adder circuit. Figure 14 illustrates the SLPHA logic diagram, which is based on Boolean equ.12, 13, and 14. The single layer perceptron hybrid adder is made up of basic neural network logic units like AND, OR, and NAND. The Proposed Full Adder[1] is thus defined as single layer perceptron hybrid adder. The RTL diagram of SLPHA is shown in Figure 15. The SLPHA features seven cells, five input/output ports, and ten nets, as shown in Figure 15.

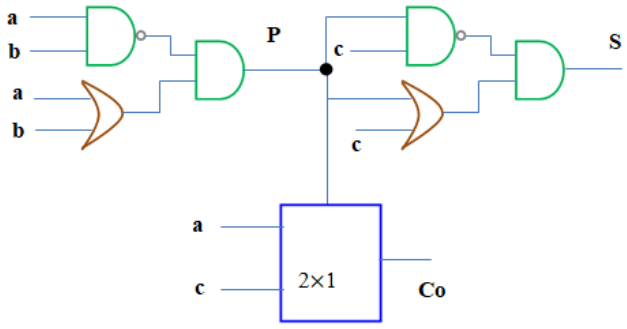

Fig. 14. Logic diagram of SLPHA

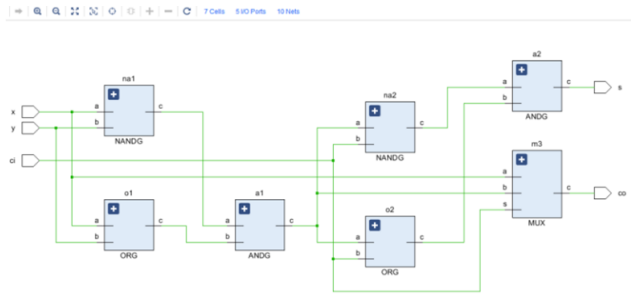

Fig. 15. RTL diagram of SLPHA

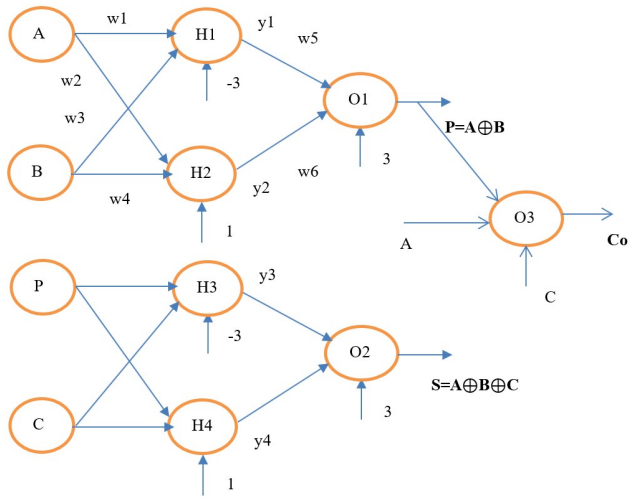

Fig. 16. Logic diagram of MLPHA

4.4 Multi Layer Perceptron Hybrid Adder (MLPHA) design

The Multi Layer Perceptron Hybrid Adder (MLPHA) is designed based upon the multi-layer perceptron [24] method. Figure 16 illustrates the MLPHA logic diagram, which is developed based on Boolean equ.12, 13, and 14. The RTL diagram of MLPHA is shown in Figure 17.

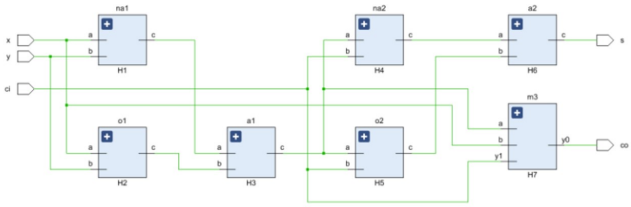

Fig. 17. RTL diagram of MLPHA

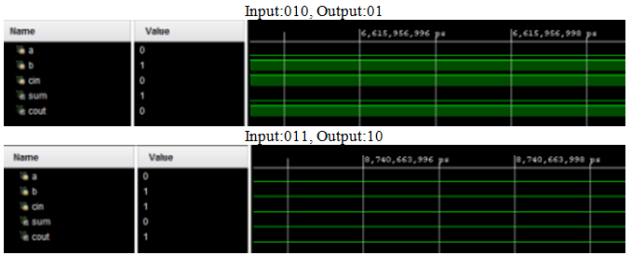

Fig. 18. Simulation result of ANN adder

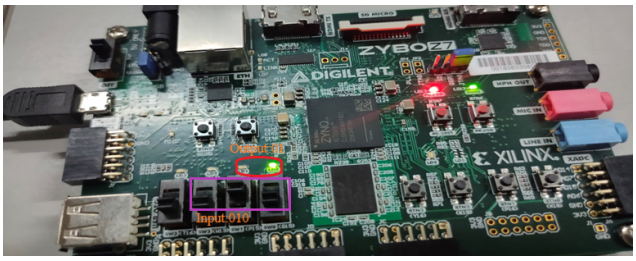

Fig. 19. Implementation of ANN adder on FPGA board when input=010

5. SIMULATION AND IMPLEMENTATION RESULTS

To build Verilog codes for three ANN Hybrid adders, we utilised the Xilinx Vivado SDK 2017.2 for Zynq-7000, device xc7z020, and package clg400 with speed grade-1. The Xilinx Vivado Zynq 7000 target device's behavioural simulation is used to show the functional verification of all three ANN adders. The behaviour of three ANN adders was validated for all conceivable test vectors, and here shown simulation results of the ANN adder for 010 and 011 combinations. The simulation results of ANN hybrid adder is shown in Figure 18. All three ANN Hybrid adders are implemented on the FPGA ZYBOZ7 board. On this board, the user switches G15, P15, and W13 are mapped to inputs (cin, b, and a). The user LEDs M14 and M15 are allocated to the outputs (sum, cout). We use logic 1 as an input when the switch is turned on, and logic 1 as an output when the LED glows, and vice versa. The LED M14 is activated and M15 is turned off when the input combination 010 is used, resulting in an output of 01 as shown in Figure 19. As seen in Figure 20, input 011 is likewise transmitted to 10. The implementation results of three ANN Hybrid adders are shown in Figure 21. Figure 21 demonstrates that the ANN_{CHA} has 213mW of power dissipation (dynamic and static) with a 6.99nsec delay, whereas from Figure 22 the SLPHA has 173mW with a 6.919nsec delay and MLPHA absorbs 156mW with a minor increase in delay shown in Figure 23. On the basis of implementation findings, Table 2 shows compar-

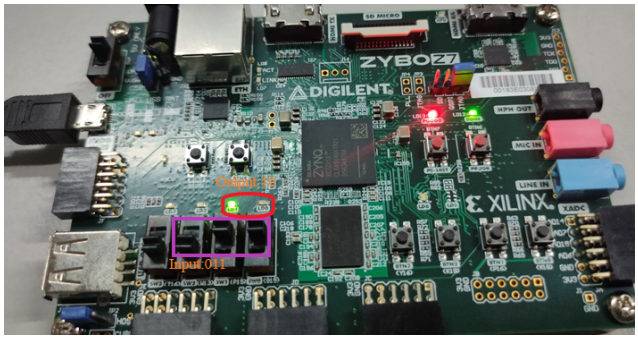

Fig. 20. Implementation of ANN adder on FPGA board when input=011

Fig. 21. Synthesis results of ANN_{CHA}

Fig. 22. Synthesis results of SLPHA

Fig. 23. Synthesis results of MLPHA

Table 2. Implementation results comparison among ANN adders

Type of ANN Adder	$P_{dynamic}$ (mW)			P_{static} (mW)	Delay (nsec)
	Signal	Logic	I/O		
CHA	2	1	89	121	6.969
PFA[1]	1	<1	89	121	7.106
SLPHA	1	<1	51	121	6.919
MLPHA	1	<1	**35**	120	7.051

Table 3. Comparison among ANN adders based on performance metrics

Type of ANN Adder	P_{avg} (mW)	$P_{dynamic}$ (mW)	Delay (nsec)	FOM	PDP(nJ)	EDP(10^{-18}×nJ)
CHA	106.5	92	6.969	0.742	0.641	4.47
PFA[1]	105.5	90	7.106	0.749	0.639	4.54
SLPHA	86.5	52	6.919	0.598	0.359	2.48
MLPHA	**78**	**36**	**7.051**	**0.549**	**0.235**	**1.78**

ison among three adders. Table 2 indicates that with a little delay change, the proposed MLPHA has minimum power dissipation.

6. EVALUATION OF PERFORMANCE METRICS

Performance indicators for ANN Hybrid adder designs, such as PDP, EDP, and FOM, can also be assessed. It is simple to determine the ideal design for advanced architecture based on these performance indicators. The dynamic power dissipation of digital circuits alone is no longer sufficient to determine the performance of digital systems in today's technology. As a result, the Power Delay Product (PDP) is defined as the product of dynamic power dissipation and delay in a digital circuit.

$$PDP = P_d * t_{pd} \tag{15}$$

A digital circuit's Figure of Merit (FOM) is also a crucial parameter for determining the optimum design. A digital logic circuit's FOM is determined by the average power dissipation and propagation latency.

$$FOM = P_{avg} * t_{pd} \tag{16}$$

However, in computation, a digital logic circuit with a low PDP may be slow. The Energy Delay Product is an important measure that should be used (EDP). The EDP is the sum of the PDP and the time it takes to complete specific processes.

$$EDP = PDP * t_{pd} \tag{17}$$

Based on equ.15, 16, and 17, Table 3 illustrates the estimated performance metrics for three ANN Hybrid adders. As shown in Table 3, the Proposed MLPHA outperforms in terms of FOM, PDP, and EDP, When contrasted to the PFA[1] and single layer perceptron hybrid adder designs. Figure 24 presents a graphical representation of the proposed ANN Hybrid adder in terms of percent improvement in performance trade-offs over PFA[1] and SLPHA.

7. CONCLUSION

In this paper, SLP and MLP approaches are introduced to design arithmetic circuits. The novel method proposes each neuron as a unique instance of a Boolean equation, including variables that can be employed to achieve solid realization. This is accomplished by a set of reprogrammable Verilog algorithms that can be easily translated into FPGA realizations using appropriate EDA tools. The use of NNs in this study is used to present a modern digital circuit design. The proposed architecture introduces a innovative approach to minimizing EDP of digital NN computing circuits. Finally, FPGA

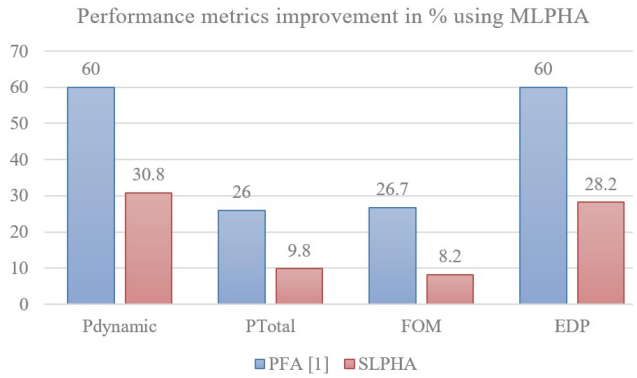

Fig. 24. Performance metrics comparison in terms of percentage w.r.t. MLPHA

realization approach is used to evaluate dynamic power dissipation, resulting in a (60%, 28.2%) improvement in dynamic power dissipation of the proposed method over PFA[1] and single layer perceptron hybrid adder.The estimation of dynamic power dissipation is important because it helps the user to identify the level of dynamic power dissipation occurs that is expected in real-time applications. Furthermore, the multi-operand computing architectures can be employed with the NN-based single bit hybrid adder. FPGA devices with artificial neural networks is more useful to design high performance computing architectures and processors. The proposed MLP hybrid adder is more suitable for present modern machine learning computations.

8. REFERENCES

[1] Pakkiraiah, C., Satyanarayana, D. R. (2022). An Innovative Design of Low Power Binary Adder based on Switching Activity. International Journal of Computing and Digital Systems, 11(1), 861-871.

[2] Ganesh R, Bhanu Prakash D. FPGA Realization of Logic Gates using Neural Network, CVR Journal of Science and Technology, Volume 20, June 2021, pp.61-66.

[3] Kim S, Kim N, Seo J, Park JE, Song EH, Choi SY, Kim JE, Cha S, Park HH, Nam JM. Nanoparticle-based computing architecture for nanoparticle neural networks Science advances 2020 Aug 1;6(35):eabb3348.

[4] Sabbaghi R, Akbari-Hasanjani R, Dehbozorgi L. New logic gates using neural network. International Journal of Smart Electrical Engineering. 2019 Jun 1;8(02):67-74.

[5] Faraone, Julian, et al, Addnet Deep neural networks using fpga-optimized multipliers. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 28.1 (2019): 115-128.

[6] Medina-Santiago A, Reyes-Barranca MA, Algredo-Badillo I, Cruz AM, Gutiérrez KA, Cortés-Barrón AE. Reconfigurable arithmetic logic unit designed with threshold logic gates. IET Circuits, Devices Systems. 2019 Jan 17;13(1):21-30.

[7] Nazemi M, Pasandi G, Pedram M. Energy-efficient, low-latency realization of neural networks through boolean logic minimization. InProceedings of the 24th Asia and South Pacific Design Automation Conference 2019 Jan 21 (pp. 274-279).

[8] Gurney, Kevin. An introduction to neural networks. CRC press, 2018.

[9] Duarte, Javier, et al, Fast inference of deep neural networks in FPGAs for particle physics. Journal of Instrumentation 13.07 (2018): P07027.

[10] Abdelouahab, Kamel, Maxime Pelcat, and Francois Berry. "The challenge of multi-operand adders in CNNs on FPGAs: how not to solve it!." Proceedings of the 18th International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation. 2018.

[11] Lu, Liqiang, et al. "Evaluating fast algorithms for convolutional neural networks on FPGAs." 2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM). IEEE, 2017.

[12] Zhao, Ritchie, et al. "Accelerating binarized convolutional neural networks with software-programmable fpgas." Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. 2017.

[13] Nurvitadhi, Eriko, et al. "Can fpgas beat gpus in accelerating next-generation deep neural networks?." Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. 2017.

[14] Zhang L, Wang C, Liu W, O'Neill M, Lombardi F. XOR gate based low-cost configurable RO PUF. In2017 IEEE International symposium on circuits and systems (ISCAS) 2017 May 28 (pp. 1-4). IEEE.

[15] Venieris, Stylianos I., and Christos-Savvas Bouganis. "fpgaConvNet: A framework for mapping convolutional neural networks on FPGAs." 2016 IEEE 24th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM). IEEE, 2016.

[16] Nurvitadhi, Eriko, et al. "Accelerating binarized neural networks: Comparison of FPGA, CPU, GPU, and ASIC." 2016 International Conference on Field-Programmable Technology (FPT). IEEE, 2016.

[17] Li, Huimin, et al. "A high performance FPGA-based accelerator for large-scale convolutional neural networks." 2016 26th International Conference on Field Programmable Logic and Applications (FPL). IEEE, 2016.

[18] Kiron, K., et al. "Performance Evaluation of Sequential Adder using Neural Networks." Indian Journal of Science and Technology 9.37 (2016).

[19] Mandal RK. Design of basic logic gates using CMOS and Artificial Neural Networks (ANN). Advances in Modelling and Analysis D. 2016;21(1):54-65.

[20] Zhang, Chen, et al. "Optimizing fpga-based accelerator design for deep convolutional neural networks." Proceedings of the 2015 ACM/SIGDA international symposium on field-programmable gate arrays. 2015.

[21] Ovtcharov, Kalin, et al "Accelerating deep convolutional neural networks using specialized hardware." Microsoft Research Whitepaper 2.11 (2015): 1-4.

[22] Abrol S, Mahajan R. Implementation of single artificial neuron using various activation functions and XOR gate on FPGA chip. In2015 Second International Conference on Advances in Computing and Communication Engineering 2015 May 1 (pp. 118-123). IEEE.

[23] Ele SI, Adesola WA. Artificial neuron network implementation of boolean logic gates by perceptron and threshold element as neuron output function. International Journal of Science and Research (IJSR). 2015;4(9):637-41.

[24] Sibanda, Wilbert, and Philip Pretorius. "Novel application of Multi-Layer Perceptrons (MLP) neural networks to model HIV in South Africa using Seroprevalence data from antenatal clinics." International Journal of Computer Applications 35.5 (2011): 26-31.