An Efficient Perforation Based Neuron Model for Digital Application

[1]B.Akilandaeswari, [2]S.Murugeswari,M.E

[1,2]*Associate Professor, Syed Ammal Engineeing College*

ABSTRACT:

Neural Network with perforation approach has emerged as a promising solution for Digital application in VLSI Technology. The performance of the Multipliers in Neural Network are degraded by complexity, more area and power consumption. In order to improve the performance neural networks perforation scheme is used. Using the partial product perforation method on multiplier architectures of Neural Network for weight and data product by exploiting the statistics of the input data. The perforation skips the generation of partial products and thus decreases the number of operands to be accumulated. The proposed perforation technique to neuron model edges to reduce a structural complexity. Our proposed Neural Network has been coded in Verilog HDL and simulated using Xilinx12.1.

Keyword: *Neural Networks (NNs) ,Field Programmable Gate Array (FPGA) .*

I INTRODUCTION

The term **neural network** was traditionally used to refer to a network or circuit of biological neurons. The modern usage of the term often refers to artificial neural networks, which are composed of artificial neurons or nodes. Thus the term has two distinct usages:

1. Biological neural networks are made up of real biological neurons that are connected or functionally related in the peripheral nervous system or the central nervous system. In the field of neuroscience, they are often identified as groups of neurons that perform a specific physiological function in laboratory analysis.

2. Artificial neural networks are made up of interconnecting artificial neurons (programming constructs that mimic the properties of biological neurons). Artificial neural networks may either be used to gain an understanding of biological neural networks, or for solving artificial intelligence problems without necessarily creating a model of a real biological system. The real, biological nervous system is highly complex and includes some features that may seem superfluous based on an understanding of artificial networks.

The brain, neural networks and computers:

Neural networks, as used in artificial intelligence, have traditionally been viewed as simplified models of neural processing in the brain, even though the relation between this model and brain biological architecture is debated, as little is known about how the brain actually works.[citation needed] A subject of current research in theoretical neuroscience is the question surrounding the degree of complexity and the properties that individual neural elements should have to reproduce something resembling animal intelligence. Historically, computers evolved from the von Neumann architecture, which is based on sequential processing and execution of explicit instructions. On the other hand, the origins of neural networks are based on efforts to model information processing in biological systems, which may rely largely on parallel processing as well as implicit instructions based on recognition of patterns of 'sensory' input from external sources. In other words, at its very heart a neural network is a complex statistical processor (as opposed to being tasked to sequentially process and execute). Neural coding is concerned with how sensory and other information is represented in the brain by neurons. The main goal of studying neural coding is to characterize the relationship between the stimulus and the individual or ensemble neuronal responses and the relationship among electrical activity of the neurons in the ensemble. It is thought that neurons can encode both digital and analog information.

Neural networks and artificial intelligence: A neural network (NN), in the case of artificial neurons called artificial neural network (ANN) or simulated neural network (SNN), is an interconnected group of natural or artificial neurons that uses a mathematical or computational model for information processing based on a connectionistic approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network. In more practical terms neural networks are non-linear statistical data modeling or decision making tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data. However, the paradigm of neural networks - i.e., implicit, not explicit , learning is stressed - seems more to correspond to some kind of natural intelligence than to the traditional Artificial Intelligence, which would stress, instead, rule-based learning.

An artificial neural network involves a network of simple processing elements (artificial neurons) which can exhibit complex global behavior, determined by the connections between the processing elements and element parameters. Artificial neurons were first proposed in 1943 by Warren McCulloch, a neurophysiologist, and Walter Pitts, an MIT logician. One classical type of artificial neural network is the recurrent Hopfield net. In a neural network model simple nodes, which can be called variously "neurons", "neurodes", "Processing Elements" (PE) or "units", are connected together to form a network of nodes — hence the term "neural network". While a neural network does not have to be adaptive per se, its practical use comes with algorithms designed to alter the strength (weights) of the connections in the network to produce a desired signal flow. In modern software implementations of artificial neural networks the approach inspired by biology has more or less been abandoned for a more practical approach based on statistics and signal processing. In some of these systems, neural networks, or parts of neural networks (such as artificial neurons), are used as components in larger systems that combine both adaptive and non-adaptive elements. The concept of a neural network appears to have first been proposed by Alan Turing in his 1948 paper "Intelligent Machinery".

*Applications of natural and of artificial neural networks:*The utility of artificial neural network models lies in the fact that they can be used to infer a function from observations and also to use it. This is particularly useful in applications where the complexity of the data or task makes the design of such a function by hand impractical. Real life applications The tasks to which artificial neural networks are applied tend to fall within the following broad categories: Function approximation, or regression analysis, including time series prediction and modeling. Classification, including pattern and sequence recognition, novelty detection and sequential decision making. Data processing, including filtering, clustering, blind signal separation and compression. Application areas of ANNs include system identification and control (vehicle control, process control), gameplaying and decision making (backgammon, chess, racing), pattern recognition (radar systems, face identification, object recognition, etc.), sequence recognition (gesture, speech, handwritten text recognition), medical diagnosis, financial applications, data mining (or knowledge discovery in databases, "KDD"), visualization and e-mail spam filtering.

II RELATD WORKS AND THE BOOTH MULTIPLIERS

The multiplier is produced by reducing the number of partial products generated. The Booth recording multiplier is one such multiplier; it scans the three bits at a time to reduce the number of partial products. These three bits are: the two bit from the present pair; and a third bit from the high order bit of an adjacent lower order pair. After examining each triplet of bits, the triplets are converted by Booth logic into a set of five control signals used by the adder cells in the array to control the operations performed by the adder cells.

To speed up the multiplication Booth encoding performs several steps of multiplication at once. Booth's algorithm takes advantage of the fact that an adder, subtractor is nearly as fast and small as a simple adder.

From the basics of Booth Multiplication it can be proved that the addition/subtraction operation can be skipped if the successive bits in the multiplicand are same. If 3 consecutive bits are same then addition/subtraction operation can be skipped. Thus in most of the cases the delay associated with Booth Multiplication are smaller than that with Array Multiplier. However the performance of Booth Multiplier for delay is input data dependant.

The method of Booth recording reduces the numbers of adders and hence the delay required to produce the partial sums by examining three bits at a time. The high performance of booth multiplier comes with the drawback of power consumption. The reason is large number of adder cells required that consumes large power .

ENCODING METHODS

Several methods of encoding the multiplicand are possible. These methods are used to reduce the number if summands that are needed to produce the final result.

Non – Booth

The first and simplest method for encoding is non Booth. This algorithm is simply a shift and add algorithm where the multiplicand is conditionally added to produce the final result. In this algorithm where the summand is selected from the set {0,M}. This algorithm's summand selection logic is a simple AND gate. Unfortunately, there is no reduction in the number of multiplicands that need to be summed to produce the final result.

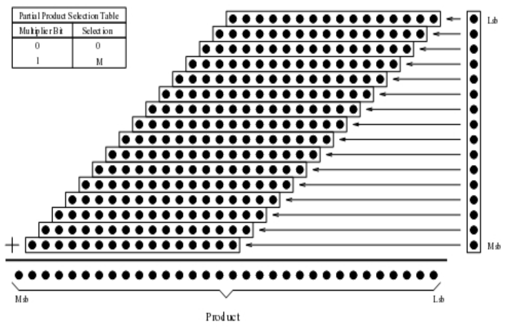


Fig.1 Non – Booth Encoding

Booth 2

A smaller number of multiplicand multipliers that need to be summed is better. The Booth algorithm attempts to reduce the number of the summands by recoding the multiplier so that groups of its bits select multipliers of the multiplicand. The Booth algorithm as it was originally proposed performed the encoding serially. Therefore, the Modified Booth Algorithm which performs the encoding in parallel is used. In this algorithm, the multiplier is partitioned into overlapping groups of 3bits. Each group is decoded in parallel to select a multiple of the multiplicand from the set {+2M, +M, 0}, as shown in figure 4. All of these multiples are obtainable by simple shifting and complementation.

Using this algorithm the number of summands is {n+1/2}. The extra 1 in the multiplicand. This is achieved by adding at least an extra zero to the left of the multipliers.

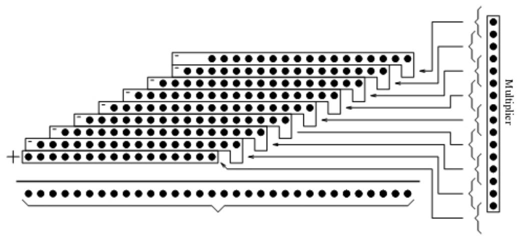


Fig.2 Booth Encoding

III PROPOSED SYSTEM

we target the design of power–error efficient multiplication circuits. We differ from the previous works by exploring approximation on the generation of the partial products. The proposed method can be easily applied in any multiplier architecture without the need for a special design, in contrast to related works. In addition, the error imposed by perforation depends only on the configuration parameters and, in contrast to existing work, can be analytically calculated without the need for exhaustive simulations.

$$A \times B = \sum_{i=0}^{n-1} Ab_i 2^i, \ \ b_i \in \{0, 1\}.$$

$$A \times B|_{j,k} = \sum_{\substack{i=0,\\ i\notin[j,j+k)}}^{n-1} Ab_i 2^i, \ \ b_i \in \{0, 1\}.$$

the partial product perforation method for the design of approximate hardware multipliers is described. Consider two *n*-bit numbers *A* and *B*. The result of their

multiplication *A* × *B* is obtained after summing all the partial products *Abi*, where *bi* is the *i* th bit of *B*. Thus The partial product perforation technique omits the generation of *k* successive partial products starting from the *j* th one. A perforated partial product is not inserted in the accumulation tree, and hence *n* full adders can be eliminated. Applying the product perforation with *j* and *k* configuration values on the multiplication, *A* × *B* produces the approximate result For each architecture, the dot diagrams of the accurate and the respective perforated tree are presented. The dots represent the bits of the partial products that have to be accumulated, while the stages represent the delay of the reduction process followed by each tree. The dashed boxes with four dots are 4:2 compressors, those with three are full adders and those with two are either full- or half-adders. Through the proposed approximation technique, the power, area, and delay of the multiplication circuit are decreased, making, though, the computation imprecise. The higher the order of a perforated partial product, the greater the error imposed at the final result. In addition, since the addition is an associative and commutative operation, when more than one partial products are perforated, the total error results from the addition of the errors produced from the perforation of each partial product separately For each architecture, the dot diagrams of the accurate and the respective perforated tree are presented. The dots represent the bits of the partial products that have to be accumulated, while the stages represent the delay of the reduction process followed by each tree. The dashed boxes with four dots are 4:2 compressors, those with three are full adders and those with two are either full- or half-adders. Through the proposed approximation technique, the power, area, and delay of the multiplication circuit are decreased, making, though, the computation imprecise. The higher the order of a perforated partial product, the greater the error imposed at the final result. In addition, since the addition is an associative and commutative operation, when more than one partial products are perforated, the total error results from the addition of the errors produced from the perforation of each partial product separately that produces specific least significant bits (LSBs) of the accumulation tree, while the perforation skips the generation of partial products and thus decreases the number of operands to be accumulated. For example, in an 8-bit array multiplier, perforating a partial product removes eight full adders from the accumulation tree and reduces its delay. In order to attain similar circuit reduction using truncation, 6 LSB have to be truncated. However, truncating 6 LSB does not offer any delay reduction. Moreover, in this example, the truncation delivers, in all the cases, incorrect results, whereas the outputs of perforation are 50% correct. Finally, perforating one partial product (out of eight) results in a 12.5% loss of information while truncating 6 LSB (out of 16) results in a 37.5% information loss

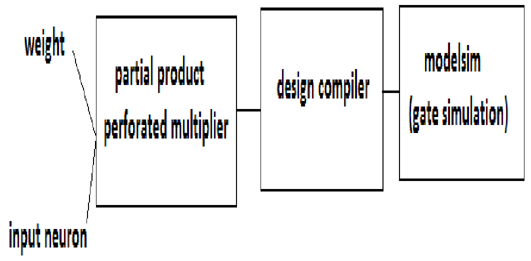


Fig.3 Partial Product

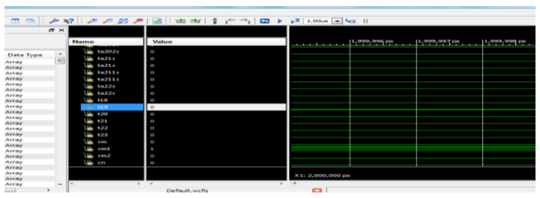
Above example for both multiplier and

```
                10100000
                01100100
               00000000
              00000000
             10100000
            00000000
           00000000
          10100000
         10100000
        00000000
        0 111111010000000
```

multiplicand more number of one is present in MSB when you truncate LSB bits in partial product do not affect output value majorly .based on statistical data in multiplier input number of half adders and full adders will be reduced.

IV EXPERIMENTAL RESULTS

The proposed circuit are simulated and synthesized by using modelsim and xilinx12.1 which occurs low area than the existing. The experimental results are given in Table 1 and the simulation results of layout and the waveforms are shown in the fig.4 and fig.5. Then the RTL schematic of the proposed are shown in fig.6.



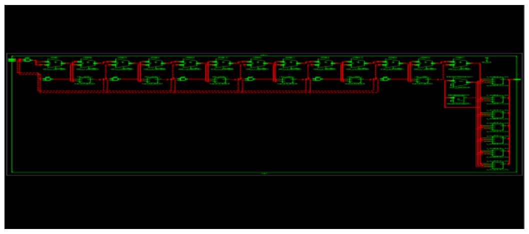
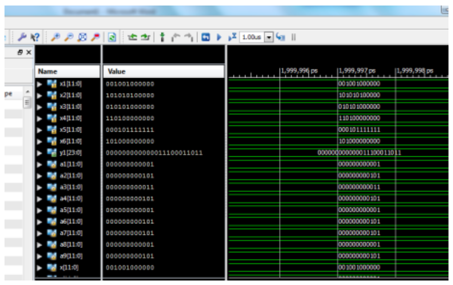


Fig.4simulation results



```
Name        Value
x1[11:0]    001001000000
x2[11:0]    101010100000
x3[11:0]    010101000000
x4[11:0]    110100000000
x5[11:0]    000101111111
x6[11:0]    101000000000
y1[23:0]    000000000000011100011011
a1[11:0]    000000000001
a2[11:0]    000000000101
a3[11:0]    000000000011
a4[11:0]    000000000101
a5[11:0]    000000000001
a6[11:0]    000000000101
a7[11:0]    000000000101
a8[11:0]    000000000001
a9[11:0]    000000000101
x[11:0]     001001000000
```

Fig.5 simulation results

	Slice	**lut**
Existing	**182**	**318**
Proposed	**158**	**275**

Table.1 comparison table

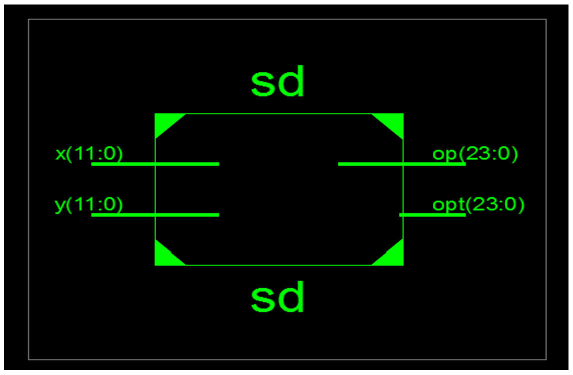


Fig.6 RTL schematic

Fig 7: gatelevl=el netlist

V **PERFORMANCE ANALYSIS**

The performance of the our proposed system with the existing scheme based on the area and error compensation which was given in Fig. 8

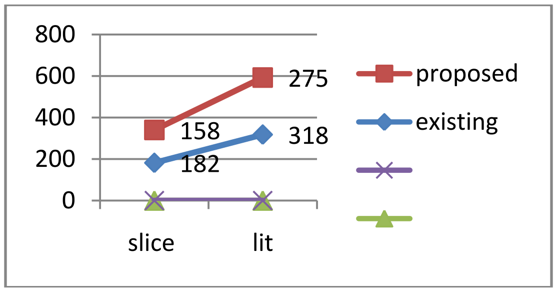


Fig.8 performance analysis of existing and proposed

VI CONCLUSION

A multiplierless model based on the Perforation targeting low cost digital implementation has been presented The proposed technique omits a number of partial products enabling high area and power savings while retaining high accuracy. Through a rigorous error analysis, we analytically characterized the induced error metrics proving that the error is bounded and predictable and we proposed two error correction methods that trade a small increase in power for high error reduction. We explored product perforation on a large set of multiplier architectures, evaluating its impact on different architectures and error bounds This proposed model has lower computational and hardware cost compared with the existing one .

REFERENCE

[1] Mohsen Hayati, Moslem Nouri, Derek Abbott,(2016) Digital Multiplierless Realization of Two Coupled Biological Hindmarsh-Rose Neuron Model IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS–II: vol.64, no. 7, pp. 135–141.

[2] M. Hayati, M. Nouri, S. Haghiri, and D. Abbott,(2015) “Digital multiplierless realization of two coupled biological Morris-Lecar neuron model,” IEEE Transactions on Circuits and Systems-I, vol. 62, no. 7, pp. 1805–1814.

[3] M. Hayati, M. Nouri, S. Haghiri, and D. Abbott, (2015)“A Hopf resonator for 2-D artificial cochlea: Piecewise linear model and digital implementation,”IEEE Transactions on Biomedical Circuits and Systems, Jun. 2015,

[4] A. Momeni, J. Han, P. Montuschi, and F. Lombardi, (2015)“Design and analysis of approximate compressors for multiplication,”IEEETrans.Comput., vol. 64, no. 4, pp. 984–994.

[5] S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, andS.Kim(2015),“Energy-efficient approximate multiplication for digital signal processing and classification applications,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 6, pp. 1180–*1184,*

[6] H. Soleimani, M. Bavandpour, A. Ahmadi, and D. Abbott (2014), “Digital implementation of a biological astrocyte model and its application,” IEEE Transactions on Neural Networks, vol. 26, no. 1, pp. 127–139,

[7] S. Gomar, A. Ahmadi, (2013) “Digital multiplierless implementation of biological adaptive-exponential neuron model,” IEEE Trans. Circuits Syst. I:Regular Papers, vol. 61, no. 4, pp. 1206–1219

[8] J. Liang, J. Han, and F. Lombardi , (2012)“New metrics for the reliability of approximate and probabilistic adders,” IEEE Trans. Comput., vol. 62, no. 9, pp. 1760–1771.

[9] T. Hishiki and H. Torikai, (2011)“A novel rotate-and-fire digital spiking neuron and its neuron-like bifurcations and responses,” IEEE Trans. Neural Netw., vol. 22, no. 5, pp. 752–767

[10] N. Zhu, W. L. Goh, W. Zhang, K. S. Yeo, and Z. H. Kong,(2010) “Design of low- power high-speed truncation-error-tolerant adder and its applicationin digital signal processing,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 18, no. 8, pp. 1225–1229,*

[11] T. Yu and G. Cauwenberghs,(2010) “Analog VLSI biophysical neurons and synapses with programmable membrane channel kinetics,” IEEE TrailS.Biomed. Circuits Syst., vol. 4, no. 3, pp. 139–148,

[12] J. Zhou, W. Yu, X. Li, M. Small, and J. A. Lu, (2010) “The correspondence between deterministic and stochastic digital neurons: Analysis and methodology,” IEEE Trans. Neural Netw., vol. 20, no. 10, pp. 1679–1684,