DOI: 10.21917/ijme.2022.0217

ANALYSIS AND IMPLEMENTATION OF MAC UNIT FOR DIFFERENT PRECISIONS

Vijay Pratap Sharma and Hemant Patidar
Department of Electronics and Communication Engineering, Oriental University, India

Abstract

This paper describes the design of the multiply- Accumulate unit and compares all parameters of the 4-bit, 8-bit, 12-bit, and 16-bit MAC unit. MAC is the basic unit that performs the multiplication operation and addition/accumulation operation. This MAC unit is designed on Vivado HLS software using LUTs at room temperature. These designs are analyzed and simulated by using the Vivado HLS tool and implemented on Zybo Evaluation and Development kit (xc7z020clg400-1).

Keywords:

MAC unit, LUTs, Power, Delay, and Utilization

1. INTRODUCTION

The MAC stands for the fundamental arithmetic tasks, the multiply-accumulate component is indeed the essential block of several digital signal processing (DSP) systems. For any real-time processing, great capacity and speed are required. The MAC unit is used to find out the energy conserved by the system and also used to find out the speed of the system. MAC unit is playing an important role in the signal processing unit. To use DSP in the future, it is essential in developing high-speed, low-power MAC. WSN (Wireless Sensor Network) is a network of television stations that broadcast. As we all know, essential processes in digital signal processing generally entail a lot of multiplies, adding, and accumulates. In order to accomplish high-performance digital signal processing, an essential part is the high-speed multiplier accumulator (MAC) unit for actual signal processing.

The multiply-accumulate unit (MAC) action adds the product of the two integers to an accumulator. The three layers that make up Convolutions Neural Networks (CNN) processing are a convolution layer, a pooling layer, and a fully connected layer. Basic MAC operations make up the convolution layer, which takes up the majority of the execution time. If we are going to study the MAC unit, we must first understand all of the basic ideas of ANN (Artificial Neural Network) and DNN (Deep Neural Network). The purpose of that work is to develop and build a multiplier accumulator (MAC) unit for high-speed digital signal processing.

2. LITERATURE SURVEY

This section provides an overview of different methodologies for classifying hyper-spectral images, as well as brief explanations of the algorithms adopted by the researcher.

In this paper, G. Raut et al. [1] proposed RECON for a neuron architecture, a valuable resource and adaptable CORDIC-based architecture. Configuration is possible with the CORDIC-based architecture; thus, a single block can produce both MAC and a number of activation functions. Vamsi and Ramesh [2] asserted that the MAC unit formed with the design multiplier might be employed in DSP applications to boost efficiency and pace. This idea has the potential to grow in the future by using reversible logic gates instead of multipliers to get even more power savings and improved performance.

Yuvaraj et al. [3] introduced the Sampoornam, a single integrated multiplier having a unique logic block that utilizes all of Vedic mathematic multiplying modules to produce better time delay performance.

Langer [4] explored the efficacy of deep neural networks (DNNs) using sigmoid activation function in this paper. DNNs have been demonstrated to estimate any d-dimensional, smooth function on a compressed set at a pace of order Wp/d, here, W denotes the number of nonzero weights and p presents smoothness of function. Consequently, only a subset of DNNs with sparse connections benefit from these rates.

Ma et al. [5], in this circumstance, cross-validation was found to be ineffective for fine-tuning algorithmic parameters for particular data sets. The machine learning community's automatic approaches for tweaking DNN parameters rely on validation performance being an accurate predictor of test performance. Before we can realize the value of DNNs, new ways that can better signal a DNN's predictive potential in a time-split test set must be created.

Ke-Lin Du and Swamy [6], suggested that machine learning algorithms can be significantly accelerated using hardware and parallel implementations, allowing them to be used in more places. Various circuit realizations for common neural network learning methods are first discussed in this paper. Then we'll tell you about systolic arrays of processors, parallel implementations on graphic processing units (GPUs), and computers in parallel. FINN is a framework for constructing efficient and configurable FPGA accelerators with a configurable heterogeneous programming architecture, according to Umuroglu et al. [7].

Simonyan and Zisserman [8] investigated the impact of the complexity of a convolutional network on its effectiveness in large-scale image identification. Bifet et al. [9] introduced a new Twitter streaming data feature allows any user to know about what is happening in the world at any given moment. Data stream analytics and analysis methods are perfect for the job since this Twitter Streaming API delivers a large number of tweets in real time, but they haven't been investigated before.

Nurvitadhi et al. [10], introduced a new hardware accelerator design for BNNs that provides higher performance while using less power. Abundance of logic, circuit efficiency, and energy extravagance for core architecture in 90-nm CMOS field-programmable gate array (FPGA) and 90-nm CMOS standard-cell application-specific integrated circuits (ASICs) have been tested scientifically by Kuon and Rose [11]. These evaluations will help framework developers make more informed conclusions about each of these two media to deploy, and FPGA manufacturers will be able to spot weaknesses in their products, allowing them to improve them.

Kastner et al. [12] created a tool flow for constructing layer-specific hardware components and modifying them in real time. Software is employed to perform segments of the inference pipeline during reconfiguration. They proposed a parameter parser that starts with CNN training using the Caffe framework. Layer reimplementation templates in C++ also make it easier to build layers in software and hardware using High-Level Synthesis based on layer particular properties.

Wu et al. [13] indicated that optimizing both running clock rates and computation efficiency is crucial for improving the execution of FPGA-based neural-network accelerators. Data transfer between memory and computation is being optimized. It will be required to improve these metrics. According to Apicella et al. [14], produced the idea about enhancing the efficiency of FPGA-based neural-network multipliers tends to boost both operating clock rates and computation performance. Streamlining data stream between storage and CPU is essential for raising such KPIs.

Wen Yan et al. [15] proposed and constructed an energy-efficient rapid array multiplier. The multiplier functions from the left side, allowing for a complete overlay across the carry-save elimination of partial multiplications and the final addition that produces by multiplication. The architecture utilizes the left-to-right carry-free (LRCF) multiplier. This varies from the LRCF multiplier in that it uses radix-4 complete adders in a significantly greatly reduced on-the-fly circuit of conversion of O(n) size. During the reduction process, the newly developed converter creates the most important portion half of the product. It gets rid of the most important portion of the final adder. During the reduction, a carry-ripple adder gets used to getting the least-significant half of the product. Antony et al. [16], suggested that the use of Verilog HDL to develop a fast-paced Vedic multiplier in light of the Urdhva Triyakbhyam Sutra. Partial product culmination is accomplished using high-speed MUX-based complete adders. In comparison to existing traditional Vedic multipliers, the proposed device has a significantly shorter delay. In the future, its efficiency in the MAC unit and ALU may be analyzed and compared to other traditional and Vedic designs.

Uma Maheswara Sainath and Sekhar [17] presented a revolutionary very fast speed design for multiplication of two 8-bit numbers in this work, which combines the benefits of compressor-based adders with the old Vedic arithmetic technique. A new 7:2 compressor architecture was also discussed, which was based on a 4:2 compressor architecture. We may observe that the Vedic math multiplier based on compressor is a preferable answer than traditional multipliers in certain expeditious and complicated VLSI circuits based on a comparison of the multiplier's area and speed with two other common multipliers. Ciminiera and Valenzano [18], discussed in this paper that four novel signed number multiplication and multiplication/addition arrays are presented in this study. The factors are assumed to be written in 2's complement, but the addend and outcomes are written in abbreviated form in these structures.

3. DIFFERENT TYPES OF MULTIPLIERS

3.1 VEDIC MULTIPLIER

The meaning of the term 'Veda' is an arcade of data. Religious text arithmetic is correlated with an ancient sort of arithmetic resurfaced from scriptures cited as Vedas. It is supported by sixteen sutras that interact with completely different branches of arithmetic as pure mathematics, geometry, arithmetic. Urdhva Tiryagbhyam is that the most generalized Sanskrit literature for the execution of religious text number styles as result of getting larger in the variety of-bits each space and holdup enhance tardily [2]. The peculiarity of religious text numbers resides in the fact that they are frequently solved verbally, which boosts the pace of operation [3]. The two Digit Multiplication Using UT Sutra is described in the Fig.1 the accumulating of twelve and forty-five victimization Urdhva Tiryagbhyam [4].

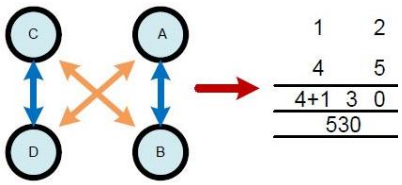

Fig.1. Two Digit Multiplication Using UT Sutra

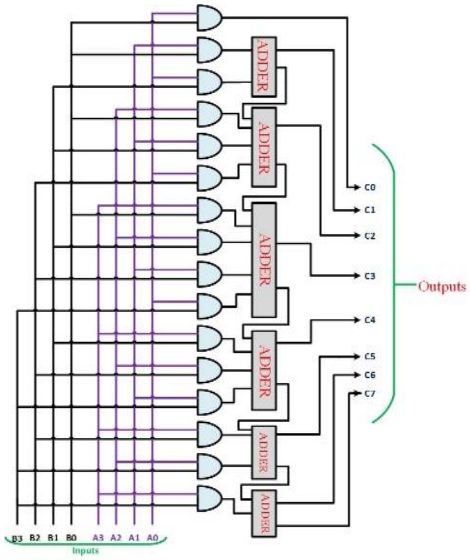

Fig.2 Circuit Diagram of Vedic Multiplier

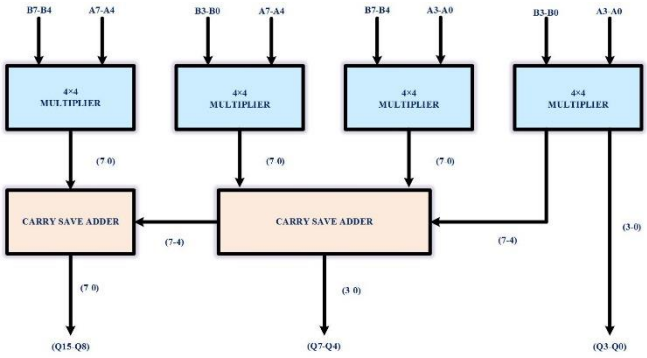

Fig.3. Block Diagram of Vedic Multiplier

The multiplier is an important component of arithmetic, logic, digital signal processing blocks, numbers, and accumulating units. It also checks the programmer performance and output. Nowadays, we prefer to use religious text numbers for computation and analysis since they are speedier.[5] Digital signal processors are utilized in various of applications, including wireless communication, sound or visual processing, corporate management, and mobile natural philosophy. The circuit and block diagram are presented in the Fig.2 and Fig.3 respectively. The following are the applications:

3.2 COLUMN BYPASSING MULTIPLIER

Column bypassing number helps to alienate the extra circuit. It skips the entire adder circuit and it consumes low power than the Braun number for a high frequency of operation. The combinations of lines of bring full adder makes up the column bypass multiplier. The advantage of this number is that it decreases the number of switching transistors required for calculation. Modified carry save full adder is described in the Fig.4, the adder cell is illustrated.

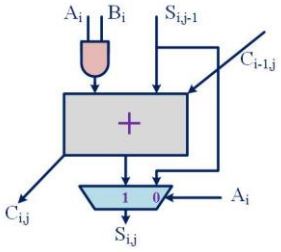

Fig.4. Modified carry save full adder

For better understanding, we consider the multiplication of two numbers 1010×1000. This number contains two zeros due to this the first and third columns are deactivated. Now we take another example like 1111×1000 since there is no zero so all the columns are activated. The diagram of Column bypassing multiplier is presented in Fig.5. The criteria of this methodology are that the column that becomes switched on is depends upon the number of ones. For example, if we discuss sixteen-bits then there are sixteen ones and then used all the adders are get switched on together. Due to this, they consume high power. If the numbers of zeros are maximum then there is less shift activity and uses less power

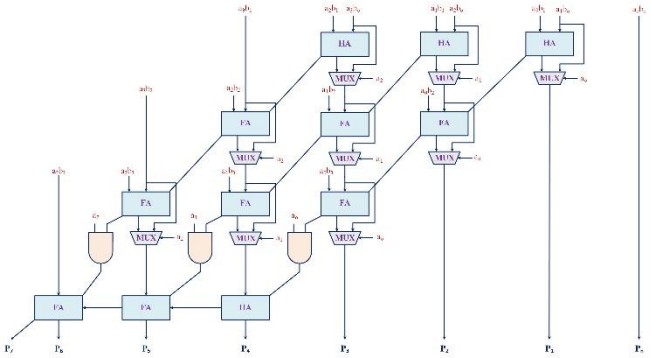

Fig.5. Diagram of Column bypassing multiplier

The limitation of this method is that variety of columns Switched depends on the number of ones within the number. As an example, if the number is sixteen-bit long as 1111111111111111 then all the total adders altogether the columns can get switched and consumes a lot of power. Less shift activity of the elements will be achieved if the number contains a lot of zeros than one.

3.3 WALLACE TREE MULTIPLIER

The initial step is that the formation of partial product by multiplying every-bit from the multiplier to same-bit position of number. Secondly, teams of 3 adjacent rows are collected. Every cluster of 3 rows is reduced by exploitation half adders and full adders. The Wallace Tree Multiplier's functioning is seen in Fig.6.

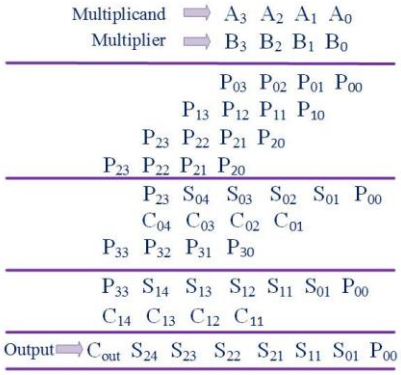

Fig.6. Operation of Wallace Tree Multiplier

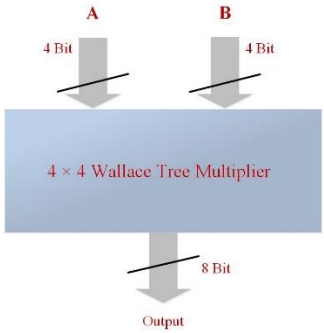

Fig.7. Wallace Tree Multiplier

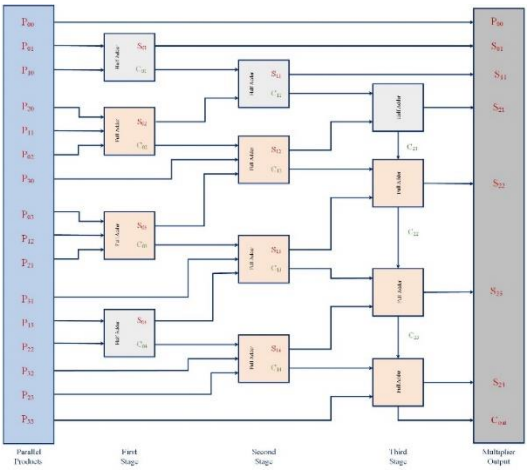

Fig.8. Wallace Tree Multiplier using full and half adders

Half adders square measure utilized in every column wherever their square measure 2-bits whereas full adders square measure used in every column wherever their square measure 3-bits, without any operation, any single-bit which is present in the column is sent to the next stage in the same column. This process is repeated until only two rows remain. Within the finish, the remaining 2 rows square measure value-added. Once finishing all the 3 stages we tend to get eight little-bits of output. The Fig.7 shows the Wallace Tree Multiplier architecture. And the Fig.8 described circuit diagram of Wallace Tree Multiplier using full and half adders.

4. MULTIPLIER BY USING VHDL CODE

MAC is the collection of an adder, multiplier, and accumulator. We get the input of the multiplier and accumulator from the memory location and then transfer it into the multiplier factor block, which does the multiplication operation. After that data send to the adder and then it accumulates all data and then the data is stored in the memory location. The entire process gets into a single clock. MAC unit architecture. The basic MAC unit architecture is presented in the Fig.9.

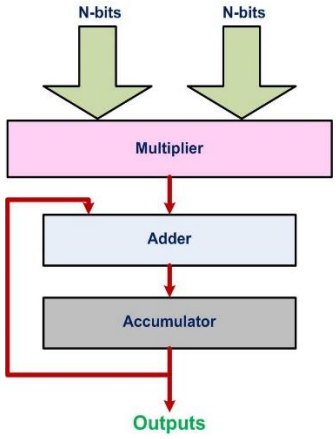

Fig.9. Multiply and Accumulate (MAC)

The design consists of eight-bit modified array multiplier designed using full adder and gate. The design of full adder and AND is victimization transmission gate (uses map entered logic). And to store the output it has one eight-bit accumulator resister and two 4-bit registers to achieve the data from memory. This mac unit designed reduces the standby power consumption and gives higher system performance. In this case, we send the feedback of the product of A_i and B_i to the accumulator and add again other products of A_i and B_i and accumulate them. This type of MAC is able to multiply and add with previous products consecutively as several times.

$$Operation: Output = \sum_i A_i B_i \quad (1)$$

The Eq.(1) presents output of MAC unit. For example, in FIR filter the output of the FIR filter is provided by

$$Y(m) = \sum_{m=0}^{k-1} x(k) h(m-k) \quad (2)$$

The Eq.(2) represents output of the FIR filter, here the filter's output is represented by $y(m)$., the impulse response of the filter is $h(m)$, and the input to the filter is $x(m)$. A finite impulses reaction filter's result has described the finite-length balanced sum of the filter's current and prior inputs. As a result, to perform filtering through as shown in the above equation, the very minimum need is to quickly multiply two values and combine the results, requiring the use of a particular low-power hardware mac in Digital Signal Processing.

The MAC operation is that the basis of the many DSP algorithms notably digital filtering. The term "digital filter" refers to an algorithmic rule by that a digital signal or cycle of numbers is changed into another order of numbers called the output digital signal. A mac speed is allocating for each finite impulse response (FIR) and infinite impulse response (IIR) filter. The amount of mac operations depends on the sophistication of the prototype filter. Digital filters are extensively used in applications like image processing technique, pattern matching, and spectrum analysis to process signals (discrete-time signals) in the digital domain. Generally, FIR filters are most well-liked in lower-order solutions. To use feedback, they exhibit naturally bounded responses. Each instruction needs one RAM space and one variable, making it simple to accomplish. The architecture of MAC unit is which comes automatically from the Xilinx Vivado tool is represented in the Fig.10.

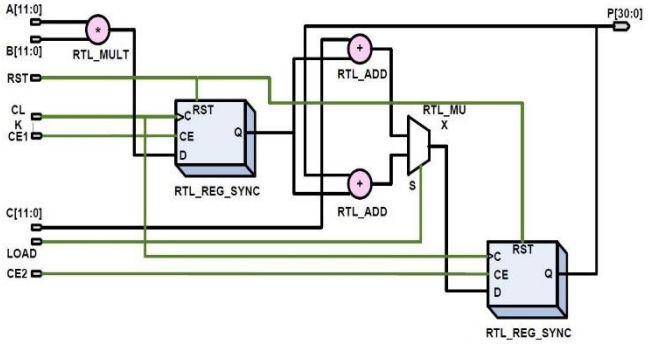

Fig.10. Schematic of MAC Unit

5. RESULTS

In this section, we have done several analyses like Utilization, Power and Delay. We have extracted all the results with the help of Xilinx Vivado tool. In the report utilization part, we will see what kind of resources are used in that MAC Unit. Other section will describe that how much power it will consume. Mainly we will go for the dynamic power. The last section of that result part is Delay. So, we will deeply explain all the results one by one in particular section. The Table.1 presented the comparison between the parameter of resource utilization report, on chip power reports and critical delay reports.

Table.1. Mac Unit Comparison Table for Different-bit-Precision

Hardware	**Precision**			
Physical Parameters	4-bit	8-bit	12-bit	16-bit
Resources Utilization Reports				
Logic Slices (68)	11	25	54	51
Slice LUTs (17600)	35	81	182	301
Slice Registers (35200)	20	35	58	70

DSPs (80)	18	34	51	67
On-chip Power Reports				
Logic (µW)	20	20	160	240
signal (µW)	70	120	240	320
I/O (µW)	870	1740	3520	3360
Static Power (mW)	91	91	91	91
Circuit's Critical Delay				
Path Delay (ns)	5.203	5.579	5.692	13.963
Input Delay (ns)	1.10	1.10	1.10	3.00
Logic Delay (ns)	3.134	3.161	3.155	4.732
Route Delay (ns)	2.046	2.418	2.516	9.231

6. CONCLUSION AND FUTURE SCOPE

The Table.1 presented the comparison between the parameter of resource utilization report, on chip power reports and critical delay reports. From the above table it is clear that as the number of precision increases by 4-bit, 8-bit, 12-bit and 16-bit the physical parameters of Resources Utilization Reports like Logic Slices, Slices LUTs, Slices Registers and DSPs are also increases. In the table, on chip power reports the Logic Power, Signal power and I/O power are increases as the number of decision increases. The circuit's critical delay report shows that Input delay is same for the 4bit, 8-bit and 12-bit precision but change (increase) for 16-bit precision. Although other parameters like Path Delay, Logic Delay and Route Delay have nearly same value for 4-bit, 8-bit and 12-bit but has different value for the 16-bit precision. This MAC unit is also verified on the Zybo board for the different precision and performs better as the results shown in the table. This multiplier performs with less delay, high computational speed, and consumption of low power. The resource utilization will decrease on the FPGA board if the number of components is reduced fatherly. If we will be able to use this design as an image processing unit, there will be an oversized scope to implement image processing algorithms. If machine learning techniques are included in the future, then these varieties of multiplication algorithms will be very helpful to implement a few specific tasks.

REFERENCES

[1] G. Raut, S. Rai, S.K. Vishvakarma and A. Kumar, "Recon: Resource Efficient Cordic-Based Neuron Architecture", *IEEE Open Journal of Circuits and Systems*, Vol. 2, pp. 170-181, 2021.

[2] A.S.K. Vamsi and S. Ramesh, "An Efficient Design of 16 Bit Mac Unit using Vedic Mathematics", *Proceedings of International Conference on Communication and Signal Processing*, pp. 319-322, 2019.

[3] M. Yuvaraj, B.J. Kailath and N. Bhaskhar, "Design of Optimized Mac Unit using Integrated Vedic Multiplier", *Proceedings of International Conference on Microelectronic Devices, Circuits and Systems*, pp. 1-6, 2017.

[4] S. Langer, "Approximating Smooth Functions by Deep Neural Networks with Sigmoid Activation Function", *Journal of Multivariate Analysis*, Vol. 182, pp. 104696-104699, 2021.

[5] J. Ma, R.P. Sheridan, A. Liaw, G.E. Dahl and V. Svetnik, "Deep Neural Nets as a Method for Quantitative Structure-Activity Relationships", *Journal of Chemical Information and Modeling*, Vol. 55, No. 2, pp. 263-274, 2015.

[6] K.L. Du and M. Swamy, "Neural Network Circuits and Parallel Implementations", *Neural Networks and Statistical Learning*, PP. 829-851, 2019.

[7] Y. Umuroglu, N.J. Fraser, G. Gambardella, M. Blott, P. Leong, M. Jahre and K. Vissers, "Finn: A Framework for Fast, Scalable Binarized Neural Network Inference", *Proceedings of ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pp. 65-74, 2017.

[8] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", *Proceedings of International Conference on Microelectronic Devices*, pp. 1-6, 2014.

[9] A. Bifet and E. Frank, "Sentiment Knowledge Discovery in Twitter Streaming Data", *Proceedings of International Conference on Discovery Science*, pp. 1-15, 2010.

[10] E. Nurvitadhi, D. Sheffield, J. Sim, A. Mishra, G. Venkatesh and D. Marr, "Accelerating Binarized Neural Networks: Comparison of FPGA, CPU, GPU, and ASIC", *Proceedings of International Conference on Field-Programmable Technology*, pp. 77-84, 2016.

[11] I. Kuon and J. Rose, "Measuring the Gap between FPGAs and ASICs", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 26, No. 2, pp. 203-215, 2007.

[12] F. Kastner, B. Janben, F. Kautz, M. Hubner and G. Corradi, "Hardware/Software Codesign for Convolutional Neural Networks Exploiting Dynamic Partial Reconfiguration on PYNQ", *Proceedings of IEEE International Symposium Workshops on Parallel and Distributed Processing*, pp. 154-161, 2018.

[13] E. Wu, X. Zhang, D. Berman, I. Cho and J. Thendean, "Compute Efficient Neural-Network Acceleration", *Proceedings of International Symposium on Field-Programmable Gate Arrays*, pp. 191-200, 2019.

[14] A. Apicella, F. Donnarumma, F. Isgro and R. Prevete, "A Survey on Modern Trainable Activation Functions", *Neural Networks*, Vol. 13, No. 2, pp. 1-17, 2021.

[15] W. Yan, M.D. Ercegovac and H. Chen, "An Energy-Efficient Multiplier with Fully Overlapped Partial Products Reduction and Final Addition", *IEEE Transactions on Circuits and Systems I: Regular Papers*, Vol. 63, No. 11, pp. 1954-1963, 2016.

[16] S.M. Antony, S.S.R. Prasanthi, S. Indu and R. Pandey, "Design of High-Speed Vedic Multiplier using Multiplexer Based Adder", *Proceedings of International Conference on Control Communication and Computing*, pp. 448-453, 2015.

[17] M.U. Maheswara Sainath and B. Sekhar, "High Speed Vedic Multiplier", *International Journal of Engineering Research*, Vol. 2, No. 3, pp. 1-15, 2014.

[18] L. Ciminiera and A. Valenzano, "Low Cost Serial Multipliers for High Speed Specialised Processors", *IEE Proceedings E (Computers and Digital Techniques)*, Vol. 135, No. 5, pp. 259-265, 1988.