accumulation units using modified vedic multiplier

Nithin
PG Student ECE.
Malla Reddy Engineering College
Jawaharlal Nehru
Technological University
Hyderabad, India

Dr M Jagadeesh Chandra Prasad
Professor & Head ECE
Malla Reddy Engineering College
Jawaharlal Nehru
Technological University
Hyderabad, India

***Abstract*—: Multiply and accumulation (MAC) units are the essential programmable logic blocks in microprocessors, microcontrollers. However, the conventional MAC units were developed by basic adders and multipliers, which resulted in higher area, delay and power consumption. Thus, this work is focused on implementation of Vedic-MAC using modified vedic multiplier and high-speed parallel adder (HSPA). Initially, N-bit HSPA is developed with the advanced carry-propagations, carry-generation concepts. Then, N-bit modified vedic multiplier is implemented with HSPA modules through fast carry calculation properties. Further, the proposed vedic-MAC is developed by introducing HSPA and modified vedic multiplier. The simulation results shows that the proposed vedic MAC-consumed lower area (look up tables-LUTs), power and delay consumptions as compared to other MAC methods.**

Keywords*—: Multiply and accumulation, high-speed parallel adder, modified vedic multiplier, look up tables

I. INTRODUCTION

In the current scenario everyone is dealing with electronic devices with bulky circuits in one or the other form as the gadgets, computers, TV, camera etc. Today electronics world has spread in all areas such as, healthcare, medical diagnosis, automobiles, etc. It has taken a situation and convinced everyone that it is impossible to work without electronics. A circuit is constructed from the logic gates [1], the basic electronic circuit that could be used to construct any combinational circuit. These combinational circuits function on the Boolean logic. A logic gate is created from one or more electrically controlled switches like transistors. Another form of digital MAC circuit is constructed from LUTs, termed as Programmable logic device. The LUTs can be configured to work as that of any logic circuit to arrive at logical or arithmetic values. This performs with pre-loaded values from a memory location [2]. This helps in the reprogramming of the circuit or error rectification easily without changing the internal wire arrangement. For small volume outputs these programmable logic devices are the preferred ones [3]. Electronic Digital MAC circuit design has evolved rapidly over the last decades. The earliest digital MAC circuits were designed with vacuum tubes and 2 transistors. The invention of integrated circuits with logic gates were placed on a single chip. The initial integrated circuit (IC) chips were Small Scale Integration chips [4]. In these the gate counts were very small. The advancement in the technologies has made the designers to place hundreds of gates in a single chip, which are termed as medium scale integration chips. The technology further developed to large scale integration [5], which was able to accommodate thousands of gates. With the advent of the VLSI, Very Large-Scale Integration, the design of circuits in this chip could have more than a lakh of transistors. Complexity is the ability to incorporate N number of transistors in a chip [6]. The Complexity for the particular IC increases with the number of transistors to be fabricated in it. In the fabrication process a huge circuit with more processing stages would require higher number of basic components [7]. In the current scenario, the ICs are to be designed in such a way that it could occupy less area and this certainly would lead to less power consumption. A VLSI circuit would be efficient only if it could utilize a smaller number of transistors, reduce the propagation delay and dissipates less power [8]. When the complexity of the circuit reduces then it would be certainly possible to accommodate a larger circuit in a specified die size of an IC. Hence compactness, reliability due to low interconnection and less power consumption can be achieved. The complexity of the circuits has made it impossible for the verification of the digital MAC circuits manually with the breadboards [9]. In this juncture electronic design automation tools were evolved for these processes. Some computer aided techniques were needed for the verification and layout formation [10]. The gate level digital MAC circuits were built manually by the designers till the numbers of the gates were less. Computer aided techniques became critical for verification and design of VLSI based digital MAC circuits [11]. This also became 3 popular for the circuit layout routing and automatic placement of components. Logic simulators came into existence to verify the functionality of these circuits before they were fabricated on chip. The design being more complex, logic simulation assumed an important role in the design process [12]. The functional bugs in the architecture were able to be addressed by the designers and sort out the flaws if any in the design before it is fabricated. Therefore, the major contributions of this work are as follows:

- Design of N-bit HSPA with the advanced carry-propagations, carry-generation concepts.
- Development of N-bit modified vedic multiplier is implemented with HSPA modules through fast carry calculation properties.
- Design and implementation of vedic-MAC by introducing HSPA and modified vedic multiplier.

Rest of the article is organized as follows: section 2 delas with literature survey, section 3 delas with the proposed vedic-MAC implementation, section 4 delas with analysis of results with performance comparison, section 5 concludes the article with possible future directions.

II. LITERATURE SURVEY

In [13], the authors advocated making use of an energy-efficient CMOS full adder in a reduced complexity Wallace Multiplier as part of the process to cut down on area and power

consumption while simultaneously increasing processing speed. The Reduced complexity reduction approach drastically cuts down on the amount of half adders, with a decrease of between 70 and 80 percent in the amount of half adders in comparison to the conventional Wallace multipliers. In [14], the authors described a way for reducing the latency in Wallace multipliers by applying parallel prefix adders, also known as fast adders, in the last part of the process. Parallel operations are carried out by Wallace multipliers, which leads to a significant increase in speed. In the phase when they do the reduction, it makes use of full adders and half adders. Carry propagating adders are used in the last step of both multipliers, which contributes to an increased latency in the process.

The authors of [15] compared the vedic multiplier against a traditional multiplier in order to demonstrate the speed of the vedic multiplier, which also lowered the amount of power that was used. Any marked (or unmarked) integers should be able to have their value increased using a multiplier if at all practicable. There is a vast variety of multiplier designs that are feasible, yet even the fastest one cannot do jobs any faster than the others. One such multiplier that may be realized is called the "Wallace" tree multiplier. This multiplier is able to complete jobs more quickly while still producing useful results for unsigned numbers.

The authors of [16] presented a rapid multiplier that was also conscious of its power consumption and was designed for error-resistant systems. The suggested estimate algorithm 17 is carried out with the assistance of an altered bit-width aware methodology in addition to a carry-in expectation method, whilst the proposed hybrid Wallace tree is realized with the assistance of high request counters. These suggested algorithms are realized with the help of the HDL programming language, synthesised using the Quartus and Modelsim tools, and displayed with the Modelsim software. The purpose of this article was to find ways to cut down on their power usage by lowering the required level of accuracy while simultaneously increasing their rate of operation.

In [17], the authors explored several adder tyspes and constructed the algorithm in every possible combination. A MAC unit is one of the designs that is used in the Digital Signal Processing (DSP) applications the most, and it is also used in a large number of FPGA layouts. It is one of the most often used designs. As a consequence of this, the reversible implementation of a 32-bit MAC unit—something that is sometimes used in the digital world—is carried out in the course of this study endeavor. Radix-16 Booth encoded vedic multiplier is taken into consideration in the construction of this MAC unit. This multiplier produces superior results.

In [18] authors presented an inexact 2-bit adder that was purposefully built for the purpose of computing the sum of the bits one and two of a binary value. This adder requires very little space, very little power, and a very short delay in the most basic form. Because it is easier to generate partial products using the radix-4 algorithm (also known as the adjusted Booth algorithm), a multiplier that uses this algorithm is very effective. In contrast, the radix-8 Booth multiplier [19] is less effective because it is more difficult to generate odd products using this algorithm. After that, the approximation multipliers are related to the construction of a low-pass FIR filter, and they show that their execution is chosen over that of various inexact Booth multipliers. In [20], the authors developed a High speed and Low power implementation of a 3-bit flash analog to digital converter. In this research, a thorough inquisition of a 3-bit flash ADC circuit that utilizes diode-based stacked power gating method and low leakage stacked power gating technique [21] is suggested. The total number of techniques used is 18. These methods of power gating are quite effective in lowering both the standard power and the leakage current. In order to evaluate the power gating strategies, a simulation was carried out making use of the cadence virtuoso tool at the arranged power supply provided by the 90nm technology. The authors of [22] presented a method of fine-grained power gating. This method involves gradually and favorably turning on or off the power supply for functional components in advance. In the deep submicron technology, the power leakage has evolved into a significant portion of the embedded processor's [23] overall power usage. According to the findings of the research, the method may reduce the dynamic power consumption of a LEON3 processor by 48% and the leaky power consumption by 39% while maintaining the same level of execution quality.

In [24], the authors presented a design and control plan for a chip with internal capacity units that are power gated at the instruction-by-instruction premise. Estimated aftereffects of the created chip in the 65nm CMOS technology showed that the methodology [25] reduces energy consumption by 21-35% within a temperature range of 25-85 degrees Celsius. When compared with the traditional fine-grain power gating methodology used in the same temperature range, the new method was able to minimize energy dispersion by up to 15%..

III. Proposed Method

In microprocessors and microcontrollers, the most important programmable logic blocks are referred to as MAC units. The standard MAC units, on the other hand, were created using simple adders and multipliers, which resulted in a greater need for space, a longer delay, and increased power consumption. As a result, the primary emphasis of this study is on the implementation of Vedic-MAC by making use of a modified vedic multiplier and HSPA, as is seen in Figure 1. The next step is to create an N-bit modified vedic multiplier using HSPA modules by utilizing their quick carry computation features. Here, the multiplier outcomes are applied as input to HSPA. The innovative carry-propagations and carry-generation principles are included into the development of N-bit HSPA. Then, the accumulator stores the adder output, which is stored and added back with adder circuit.

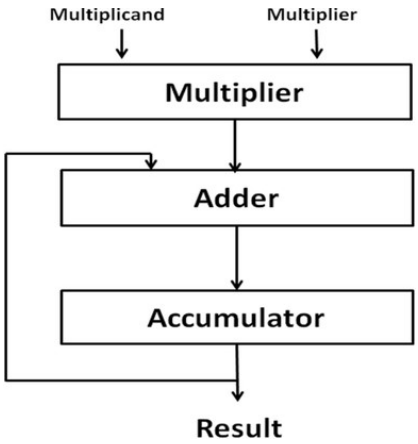

Fig. 1. Proposed Vedic-MAC block diagram.

A. Modified vedic multiplier

A N-bit Modified vedic multiplier has been built as part of MAC architecture as shown in Figure 2. The Modified vedic multiplier was selected because it produces a lower total number of partial products, which in turn reduces the total number of adders and HSPAs that are required for the accumulation of these products, which in turn reduces the total amount of space required as well as the amount of power required. AND gates are used in the operation of multiplication, which ultimately results in the production of partial products.

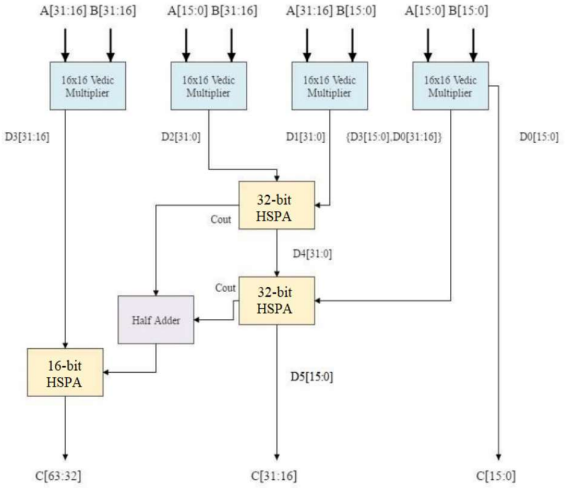

Figure 2. Block diagram of modified vedic multiplier.

The multiplication operation is executed by using "AND" gates for producing partial products. The Multiplication could also be achieved by shift operations of the multiplicand. It gives non regular PP for the further reduction stages while it could be a faster approach with less PPs. This technique of generation could best suited for the conventional array multipliers with CSA. The partial products are re arranged to form an inverted pyramid structure where all the columns are shifted upward till the first row. Then the partial products produced are grouped into multiple levels of seven rows each. These sub groups are then processed with the novel HSPAs as detailed above. The PPRs are processed in parallel with multiple HSPAs. As the overall die size is a major thing to be taken into consideration, the implementations of the gates with the latest CMOS techniques are capable to resolve these issues.

The first level in the Stage –I shall have the first seven rows for processing and the next level will process the next sub group of seven rows and so on till the group has less than seven rows, these rows are not processed in the first stage and passed over to the next stage. Each group is processed with the HSPAs, so the columns in the group with less than seven inputs are carried over to the next stage for compression. This almost eliminates first six and last six columns in the level one compression. Similar processing is done for all the groups. Every HSPA will deliver a sum and carry of one bit level higher as the output and two carry-outs to the adjacent stages of HSPA as intermediate carries. The sum and carry generated from each level of stage one is considered to be two rows of data for the next stage.

The number of levels in the first stage decides the rows for the next stage along with the left-out rows while grouping. The rows generated are 2n rows where n is the number of levels in stage one. These rows are again sub grouped along with few rows not considered in the first stage and taken for processing in the reduction sequence. These procedures are repeated till all the rows shrink to two rows. The Partial products that are left out unprocessed either as row or in a column are compressed with a lower level of HSPAs, the initial columns with one or two elements, are directly passed to the final adders. In our proposed 16 x16 Modified vedic multipliers, the entire PPs can be grouped to two levels with merely two rows left out for the next stage.

In order to accomplish a greater reduction in complexity and latency, the columns containing three data are reduced using a full adder, while columns containing four and five items are processed using HSPA. Same, HSPA is used for both the column of six elements and the column of seven elements itself. By using this strategy, it is possible to cut down on the total amount of delay as well as the area by a respective 23% and 12%. The outputs of the HSPAs each contain a sum and carry that carries them to the next 98 level. Aside from this, each stage of the HSPA provides two-bit data to the stage that comes after it for processing. Stage 2 is responsible for the addition of the output from the HSPAs located on levels one and two of stage 1. The first step has resulted in the generation of four rows. For the case of 16-bit multiplier, the three rows left out in the stage one and the four rows of the result of stage two are processed with various HSPAs depending on the corresponding column length as utilized in the stage one. In the second stage, only the center column has 6 bits which is processed with HSPAs. The preceding stages from this column are processed with HSPAs. The just first HSPA preceding the HSPA stages have a bit extra which pushes the reduction stage to have a row extra. Since this has only one element and to avoid an additional row for addition, the last HSPA's carries are added with a Half adder and passed to the next HSPAs. By having this additional half adder, the delay is maintained at the same level which would have otherwise increased by one gate level. The complexity of this adder is meagre and does not add a considerable net value in the circuit. The PPs reduced to two rows are added with a ripple adder and generates the final product outcome.

B. HSPA

In this study, HSPA is employed as the final adder. A straight implementation of this research would need an m-bit adder, which would be a 2N-bit adder with carry propagation. The implementation of different HSPAs for various column lengths will improve the efficiency on complexity. The efficiency factor increases to a greater level as the bit length of the multiplicand and the multiplier increases. With increase in the data length large numbers of rows are processed by HSPAs. The consecutive stages of the partial product reduction are the sum and carry generated from the previous stage levels. The percentages of HSPAs are larger when compared to the lower-level HSPAs which add for the efficiency.

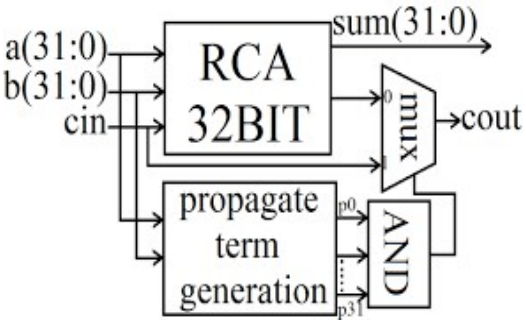

Fig. 3. Block diagram of HSPA.

The unprocessed columns that have a lower bit count are carried over to the subsequent step so that they might be added. The formula for dividing rows into levels in the PPs that are created as well as the rows that are not used is ri divided by seven, where ri is the number of rows in the PP matrix. Because of this, it is necessary to include a greater number of bits in the column being compressed whenever the degree of compression is increased. The requirement for Full adders or other HSPAs is contingent on the reduction of extra stages in the system; in the absence of this reduction, the number of stages in the system may rise, which would then result in an increase in latency and complexity.

IV. RESULTS AND DISCUSSION

Xilinx ISE software was used to create all of the Vedic-MAC designs. This software programmed gives two types of outputs: simulation and synthesis. The simulation results provide a thorough examination of the Vedic-MAC architecture in terms of input and output byte level combinations. Decoding procedure approximated simply by applying numerous combinations of inputs and monitoring various outputs through simulated study of encoding correctness. The use of area in relation to the LUT count will be accomplished as a result of the synthesis findings. In addition, a time summary will be obtained with regard to various path delays, and a power summary will be prepared utilizing the static and dynamic power consumption.

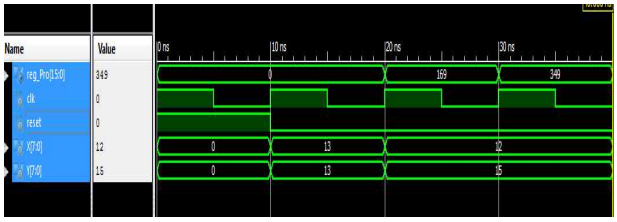

Figure 4. Simulation outcome.

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice LUTs	1626	17600	9%
Number of fully used LUT-FF pairs	0	1626	0%
Number of bonded IOBs	128	100	128%

Fig. 5. Design summary.

```
MUXCY:S->O            1    0.230   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<45> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<46> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<47> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<48> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<49> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<50> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<51> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<52> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<53> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<54> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<55> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<56> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<57> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<58> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<59> (
MUXCY:CI->O           1    0.013   0.000  dsr2/_i000006/CPA2/Madd_fsum_cy<60> (
XORCY:CI->O           3    0.251   0.438  dsr2/_i000006/CPA2/Madd_fsum_xor<61>
LUT3:I0->O            2    0.043   0.347  dsr2/_i000006/CPA5/aa[61].b1/CCC1/Mmu
LUT5:I3->O            1    0.043   0.550  dsr2/_i000006/CPA5/aa[62].b1/CCC2/Mmu
LUT6:I0->O            1    0.043   0.000  dsr2/_i000006/CPA5/Madd_fsum_lut<62>
MUXCY:S->O            0    0.230   0.000  dsr2/_i000006/CPA5/Madd_fsum_cy<62> (
XORCY:CI->O           1    0.251   0.279  dsr2/_i000006/CPA5/Madd_fsum_xor<63>
OBUF:I->O                 0.000          Product_63_OBUF (Product<63>)
----------------------------------------
Total                     6.410ns (2.362ns logic, 4.048ns route)
                                  (36.8% logic, 63.2% route)
```

Fig. 6. Time summary.

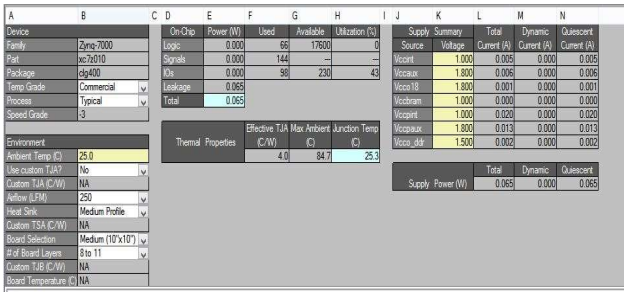

Fig. 8. Power summary.

TABLE I. PERFORMANCE EVALUATION

Metric	Standard MAC [22]	Booth-MAC [24]	Hybrid-MAC [25]	Proposed Vedic-MAC
LUTs	3467	2855	2442	1626
Time delay (ns)	20.927	13.837	10.735	6.410
Logic delay (ns)	9.927	7.837	5.735	2.362
Route delay (ns)	10.927	9.837	7.735	4.048
Power consumption (w)	2.61	1.41	0.26	0.065

CONCLUSION

Implementation of the Vedic-MAC algorithm utilizing the modified vedic multiplier and HSPA is the primary focus of this work. Initially, advanced carry-propagations and carry-generation concepts are incorporated into the development of N-bit HSPA. The next step is to implement an N-bit modified vedic multiplier with HSPA modules by utilizing their fast carry calculation properties. In addition, the HSPA and the modified vedic multiplier are incorporated into the development of the proposed vedic-MAC. When compared to other MAC methods, the proposed vedic MAC was found to have lower area (LUT) consumption, as well as lower power and delay consumptions. These findings were gleaned from the simulation. This work can be extended with hybrid MAC units with hybrid adders, and hybrid multipliers.

REFERENCES

[1] Bhuvaneswary, N., et al. "Efficient Implementation of Multiply Accumulate Operation Unit Using an Interlaced Partition Multiplier." *Journal of Computational and Theoretical Nanoscience* 18.4 (2021): 1321-1326.

[2] Chen, Jia, et al. "Multiply accumulate operations in memristor crossbar arrays for analog computing." *Journal of Semiconductors* 42.1 (2021): 013104.

[3] Wang, Yin, et al. "An in-memory computing architecture based on two-dimensional semiconductors for multiply-accumulate operations." *Nature communications* 12.1 (2021): 1-8.

[4] Leroux, Nathan, et al. "Radio-frequency multiply-and-accumulate operations with spintronic synapses." *Physical Review Applied* 15.3 (2021): 034067.

[5] Matsui, Chihiro, et al. "Energy-efficient reliable HZO FeFET computation-in-memory with local multiply & global accumulate array for source-follower & charge-sharing voltage sensing." *2021 Symposium on VLSI Technology*. IEEE, 2021.

[6] Schober, Peter, M. Hassan Najafi, and Nima TaheriNejad. "High-Accuracy Multiply-Accumulate (MAC) Technique for Unary Stochastic Computing." *IEEE Transactions on Computers* 71.6 (2021): 1425-1439.

[7] Cho, M., & Kim, Y. (2021). FPGA-Based Convolutional Neural Network Accelerator with Resource-Optimized Approximate Multiply-Accumulate Unit. *Electronics*, *10*(22), 2859.

[8] Christopher Vishal, J., Sai Sri Charan, S., Kumar, A., & Sivasankaran, K. (2021, February). Design of Reconfigurable Multiply-Accumulate Unit with Computational Optimization. In *International Conference on Microelectronic Devices, Circuits and Systems* (pp. 335-349). Springer, Singapore.

[9] Park, J., Lee, S., & Jeon, D. (2021). A neural network training processor with 8-bit shared exponent bias floating point and multiple-way fused multiply-add trees. *IEEE Journal of Solid-State Circuits*, *57*(3), 965-977.

[10] Kim, J. S., & Lee, J. W. (2021). 10T SRAM computing-in-memory macros for binary and multibit MAC operation of DNN edge processors. *IEEE Access*, *9*, 71262-71276.

[11] Chen, Ke, et al. "Efficient implementations of reduced precision redundancy (RPR) multiply and accumulate (MAC)." *IEEE Transactions on Computers* 68.5 (2018): 784-790.

[12] Chen, Wei-Hao, et al. "A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors." *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 2018.

[13] Nahmias, M. A., De Lima, T. F., Tait, A. N., Peng, H. T., Shastri, B. J., & Prucnal, P. R. (2019). Photonic multiply-accumulate operations for neural networks. *IEEE Journal of Selected Topics in Quantum Electronics*, *26*(1), 1-18.

[14] Bhuvaneswary, N., Prabu, S., Tamilselvan, K., & Parthiban, K. G. (2021). Efficient Implementation of Multiply Accumulate Operation Unit Using an Interlaced Partition Multiplier. *Journal of Computational and Theoretical Nanoscience*, *18*(4), 1321-1326.

[15] Camus, Vincent, Linyan Mei, Christian Enz, and Marian Verhelst. "Review and benchmarking of precision-scalable multiply-accumulate unit architectures for embedded neural-network processing." *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 9, no. 4 (2019): 697-711.

[16] Masadeh, M., Hasan, O., & Tahar, S. (2019). Input-conscious approximate multiply-accumulate (mac) unit for energy-efficiency. *IEEE Access*, *7*, 147129-147142.

[17] Tung, Che-Wei, and Shih-Hsu Huang. "A high-performance multiply-accumulate unit by integrating additions and accumulations into partial product reduction process." *IEEE Access* 8 (2020): 87367-87377.

[18] Camus, Vincent, Christian Enz, and Marian Verhelst. "Survey of precision-scalable multiply-accumulate units for neural-network processing." 2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS). IEEE, 2019.

[19] Zhang, Hao, Jiongrui He, and Seok-Bum Ko. "Efficient posit multiply-accumulate unit generator for deep learning applications." *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2019.

[20] Lyakhov, Pavel, et al. "High-performance digital filtering on truncated multiply-accumulate units in the residue number system." *IEEE Access* 8 (2020): 209181-209190.

[21] Hussain, S. U., Rouhani, B. D., Ghasemzadeh, M., & Koushanfar, F. (2018, June). Maxelerator: FPGA accelerator for privacy preserving multiply-accumulate (MAC) on cloud servers. In *Proceedings of the 55th Annual Design Automation Conference* (pp. 1-6).

[22] Ryu, S., Park, N., & Kim, J. J. (2018). Feedforward-cutset-free pipelined multiply–accumulate unit for the machine learning accelerator. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *27*(1), 138-146.

[23] Shin, D., Choi, W., Park, J., & Ghosh, S. (2019). Sensitivity-based error resilient techniques with heterogeneous multiply–accumulate unit for voltage scalable deep neural network accelerators. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *9*(3), 520-531.

[24] Garland, J., & Gregg, D. (2018). Low complexity multiply-accumulate units for convolutional neural networks with weight-sharing. *ACM Transactions on Architecture and Code Optimization (TACO)*, *15*(3), 1-24.

[25] Carmichael, Zachariah, et al. "Performance-efficiency trade-off of low-precision numerical formats in deep neural networks." *Proceedings of the conference for next generation arithmetic 2019*. 2019.