International Journal on Applications in Engineering and Technology
Volume 6 : Issue 3: November 2020, pp 1 –6 www.aetsjournal.com ISSN (Online) : 2455 - 0523

A New VLSI Architecture for Modified Booth Algorithm using Vedic Multiplier

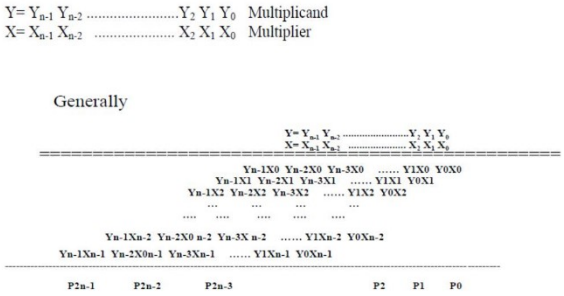
Dr.V. Jayaraj, Mrs.S.M.Deepa,Mrs.Jebapaulin

***Abstract*—** In this paper, we proposed a new architecture of Multiplier-and-accumulator (MAC) for high-speed arithmetic. By combining Vedic multiplier with accumulation and devising a carry save adder (CSA), the performance was improved. Since hybrid type of the accumulator that has the largest delay in MAC was merged into CSA, the overall performance was elevated. The proposed CSA tree uses the modified array for the sign extension in order to increase the bit density of the operands. The CSA propagates the carries to the least significant bits of the partial products and generates the least significant bits in advance to decrease the number of the input bits of the final adder. Also, the proposed MAC accumulates the intermediate results in the type of sum and carry bits instead of the output of the final adder, which made it possible to optimize the pipeline scheme to improve the performance. The proposed architecture was synthesized with 250, 180 and 130 nm, and 90 nm standard CMOS library. The proposed MAC showed the superior properties to the standard design in many ways and performance twice as much as the previous research in the similar clock frequency. The proposed MAC can be adapted to various fields requiring high performance such as the signal processing areas.

***Keywords*—** Vedic multiplier, carry save adder (CSA) tree, computer arithmetic, digital signal processing (DSP), multiplier and-accumulator (MAC).

I. INTRODUCTION

Multipliers play an important role in today's digital signal processing and various other applications. With advances in technology, many researchers have tried and are trying to design multipliers which offer either of the following design targets – high speed, low power consumption, regularity of layout and hence less area or even combination of them in one multiplier thus making them suitable for various high speed, low power and compact VLSI implementation. The common multiplication method is "add and shift" algorithm. In parallel multipliers number of partial products to be added is the main parameter that determines the performance of the multiplier. To reduce the number of partial products to be added, Modified Booth algorithm is one of the most popular algorithms. To achieve speed improvements Wallace Tree algorithm can be used to reduce the number of sequential adding stages. Further by combining both Modified Booth algorithm and Wallace Tree technique we can see advantage of both algorithms in one multiplier. However with increasing parallelism, the amount of shifts between the partial products and intermediate sums to be added will increase which may result in reduced speed, increase in silicon area due to irregularity of structure and also increased power consumption due to increase in interconnect resulting from complex routing. On the other hand "serial-parallel" multipliers compromise speed to achieve better performance for area and power consumption. The selection of a parallel or serial multiplier actually depends on the nature of application. In this lecture we introduce the multiplication algorithms and architecture and compare them in terms of speed, area, power and combination of these metrics. The multiplication algorithm for an N bit multiplicand by N bit multiplier is shown below:



The multiplier-and-accumulator (MAC) are the essential elements of the digital signal processing such as filtering, convolution, and inner products. Most digital signal processing methods use nonlinear functions such as discrete cosine transform (DCT) or discrete wavelet transform (DWT). Because they are basically accomplished by repetitive application of multiplication and addition, the speed of the multiplication and addition arithmetic's determines the execution speed and performance of the entire calculation. Because the multiplier requires the longest delay among the basic operational blocks in digital system, the critical path is determined by the multiplier.

In general, a multiplier uses Booth's algorithm and array of full adders (FAs), or Wallace tree instead of the array of FAs. This multiplier mainly consists of the three parts: Booth encoder, a tree to compress the partial products such as Wallace tree, and final adder. Because Wallace tree is to add the partial products from encoder as parallel as possible, its

Dr.V .Jayaraj, Professor, Department of Electronics and Communication Engineering , Nehru Institute of Engineering and Technology, Coimbatore, Tamilnadu.

Mrs.S.M.Deepa, Assistant Professor, Department of Electronics and Communication Engineering , Nehru Institute of Engineering and Technology, Coimbatore, Tamilnadu.

Mrs.Jebapaulin, Assistant Professor, Department of Electronics and Communication Engineering , Nehru Institute of Engineering and Technology, Coimbatore, Tamilnadu.

operation time is proportional to O (log2N), where N is number of inputs. It uses the fact that counting the number of 1's among the inputs reduces the number of outputs into log2N. In real implementation, many (3:2) or (7:3) counters are used to reduce the number of outputs in each pipeline step. The most effective way to increase the speed of a multiplier is to reduce the number of the partial products because multiplication proceeds a series of additions for the partial products. To reduce the number of calculation steps for the partial products, MBA Algorithm has been applied mostly where Wallace tree has taken the role of increasing the speed to add the partial products. As multiplier consumes more time unit, work is concentrated on the design of multiplier units for reducing the delay. Conventional multiplier consumes more time and power due to generation of more number of partial products. This leads to more number of adders thus increasing the hardware complexity. So work is concentrated on the design of efficient multiplication algorithms where there is a possibility for reducing the number of partial products. Use of Vedic multiplier reduces the number of partial products in which a group of multiplier bits are compared for partial product generation

Further by eliminating separate accumulation and implementing CSA with MAC there is a possibility for reducing hardware complexity and minimizing delay and power dissipation. Performance of the MAC unit is improved by either using high speed multipliers or by using improved fast adder architectures. So we go for a MAC unit combined with accumulation and devising a hybrid type of carry save adder (CSA), for high speed operation. The proposed MAC is designed using Verilog code and simulated using Xilinx. Simulation results are used for performance comparison of the proposed multiplier.

This paper is organized as follows: Section II describes the operation of modified MAC architecture. Section III deals with the proposed MAC where multiplication is do using Vedic multiplier. Section IV compares the simulated results with previous MAC architectures. Section V concludes this paper

II. MODIFIED PARALLEL MAC ARCHITECTURE

If an operation to multiply two N–bit numbers and accumulate into a 2N- bit number is considered, the critical path is determined by the 2N-bit accumulation operation. If a pipeline scheme is applied for each step in the standard design as shown in Fig 1, the delay of the last accumulator must be reduced in order to improve the performance of the MAC. The overall performance of the standard MAC is improved by eliminating the accumulator itself by combining it with the CSA function. If the accumulator has been eliminated, the critical path is then determined by the final adder in the multiplier.

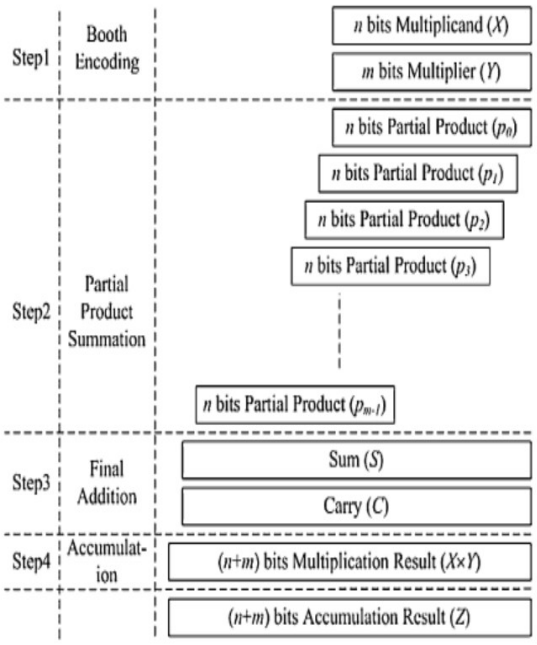


Fig 1: Basic Arithmetic steps of multiplication and accumulation

The basic method to improve the performance of the final adder is to decrease the number of input bits. In order to reduce this number of input bits, the multiple partial products are compressed into a summand a carry by CSA. The number of bits of sums and carries to be transferred to the final adder is reduced by adding the lower bits of sums and carries in advance within the range in which the overall performance will not be degraded. A 2-bit CLA is used to add the lower bits in the CSA. In addition, to increase the output rate when pipelining is applied, the sums and carry from the CSA are accumulated instead of the outputs from the final adder in the manner that the sum and carry from the CSA in the previous cycle are inputted to CSA. Due to this feedback of both sum and carry, the number of inputs to CSA increases, compared to the standard design. In order to efficiently solve the increase in the amount of data, CSA architecture is modified to treat the sign bit.

The modified MAC is organized into three steps. When compared with Fig 1, it is easy to identify the difference that the accumulation has been merged into the process of adding the partial products. Another big difference from Fig 1 is that the final addition process in step3 is not always run even though it does not appear explicitly in Fig 2. Since accumulation is carried out using the result from step2 instead of that from step3, step3 does not have to be run until the point at which the result for the final accumulation is needed.

The n-bit MAC inputs, X and Y, are converted into an (n+1) bit partial product by passing through the Booth encoder. In the CSA and accumulator, accumulation is carried out along with the addition of the partial products. As a result, n-bit S, C and

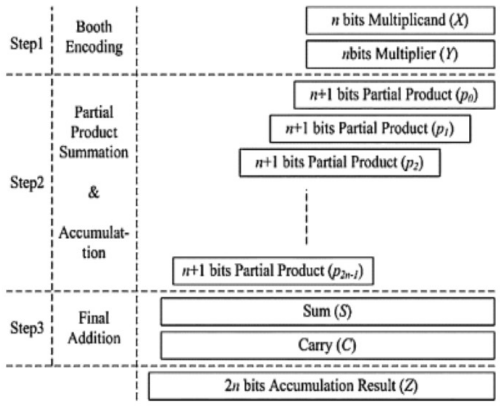


Fig 2 Modified Arithmetic operations of multiplication and accumulation.

Z (the result from adding the lower bits of the sum and carry) are generated. These three values are fed back and used for the next accumulation. If the final result for the MAC is needed, P [2n-1: n] is generated by adding S and C in the final adder and combined with P [n-1: n] that was already generated.

The architecture of hybrid type CSA that complies with the operation of the proposed MAC is shown Fig 3 which performs 8x8-bit operation. Si is to simplify the sign expansion and Ni is to compensate 1's complement number into 2's complement number. S[i] and C[i] correspond to the i-th bit of the feedback sum and carry. Z[i] is the i-th bit of the sum of the lower bits for each partial product that were added in advance and Z' [i] is the previous result.

In addition, Pj[i] corresponds to the i-th bit of the j-th partial product. Since the multiplier is for 8 bits, totally four partial products P0 [7: 0] ~ P3 [7:0] are generated from the Booth encoder. This CSA requires at least four rows of FAs for the four partial products. Thus, totally five FA rows are necessary since one more level of rows is needed for accumulation.

For an n x n-bit MAC operation, the level of CSA is (n/2). The white square in Fig 3 represents an FA and the grey square is a half adder (HA

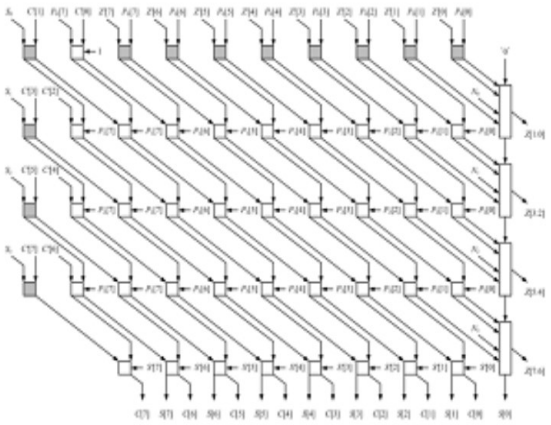


Fig 3 Architecture of the Modified CSA Tree.

The rectangular symbol with five inputs is a 2-bit CLA with a carry input. The critical path in this CSA is determined by the 2-bit CLA.

III. PROPOSED ARCHITECTURE

A. Vedic multiplier

Vedic Multiplier has become highly popular as a faster method for computation and analysis. Nikhilam Sutra is used in our proposed architecture to perform the multiplication operation. Larger the original number, lesser the complexity of the multiplication. We will illustrate this Sutra by considering the multiplication of two decimal numbers (96 × 93) where the chosen base is 100 which is nearest to and greater than both these two numbers. As shown in Fig. 4, we write the multiplier and the multiplicand in two rows followed by the differences of each of them from the chosen base, i.e., their compliments. We can write two columns of numbers, one consisting of the numbers to be multiplied (Column 1) and the other consisting of their compliments (Column 2).

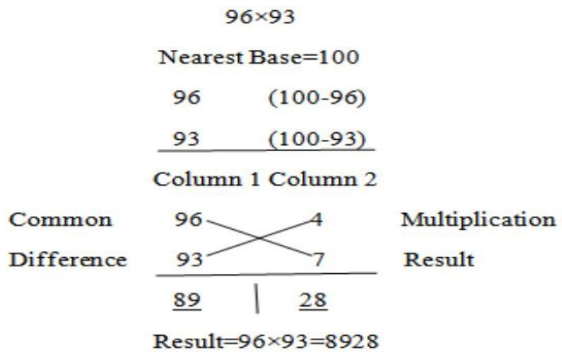


Fig 4. Multiplication of decimal numbers using Nikhilam Sutra

he product also consists of two parts which are distributed by a vertical line. The right hand side of the product will be obtained by simply multiplying the numbers of the Column 2 (7×4 = 28). The left hand side of the product will be found by cross subtracting the second number of Column 2 from the first number of Column 1 or vice versa, i.e., 96 - 7 = 89 or 93 - 4 = 89. The final result will be obtained by combining RHS and LHS (Answer = 8928).

B. Proposed MAC Architecture

The hardware architecture of the MAC is shown in Fig.5 and Fig.6. The n-bit MAC inputs, and, are converted into an (n+1) -bit partial product by passing through the Vedic multiplier. In the CSA and accumulator, accumulation is carried out along with the addition of the partial products. As a result, n-bit, S, C and Z (the result from adding the lower bits of the sum and carry) are generated. These three values are fed back and used for the next accumulation. If the final result for the MAC is needed [2n-1: n] is generated by adding S and C in the final adder and combined with P [n-1:0] that was

already generated.

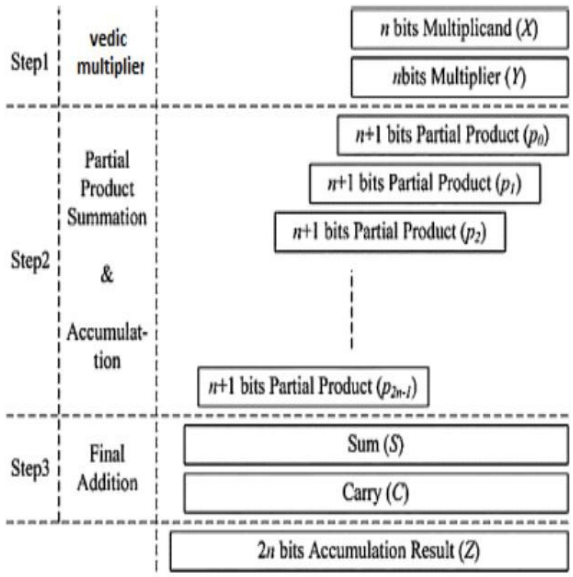


Fig. 5 Proposed arithmetic operation of MAC

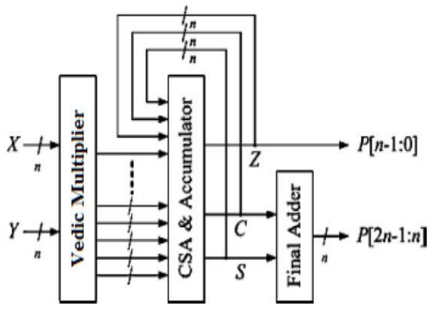


Fig 6 Hardware architecture of proposed MAC

C. Proposed CSA Architecture

The architecture of the hybrid-type CSA that complies with the operation of the proposed MAC is shown in Fig. 7, which performs 8x8-bit operation. It was formed based on Fig. 5, Si to simplify the sign expansion and Ni is to compensate 1's complement number into 2's complement number S[i] and C[i] correspond to the ith bit of the feedback sum and carry. Z[i] is the ith bit of the sum of the lower bits for each partial product that were added in advance and Z'[i] is the previous result. In addition, Pj[i] corresponds to the ith bit of the jth

partial product. Since the multiplier is for 8 bits, totally four partial products (P0[7:0] andP3 [7:0]) are generated from the Vedic Multiplier. In (11),d0Yand dN/2-12N-2Y correspond to P0[7:0] and P3[7:0], respectively. This CSA requires at least four rows of FA's for the four partial products. Thus, totally five FA rows are necessary since one more level of rows are needed for accumulation. For an nxn -bit MAC operation, the level of CSA is (n/2+1). The white square represents an FA and the gray square is a half adder (HA). The rectangular symbol with five inputs is a 2-bit CLA with a carry input. The critical path in this CSA is determined by the 2-bit CLA. It is also possible to use FAs to implement the CSA without CLA. However, if the lower bits of the previously generated partial product are not processed in advance by the CLAs, the number of bits for the final adder will increase. When the entire multiplier or MAC is considered, it degrades the performance.

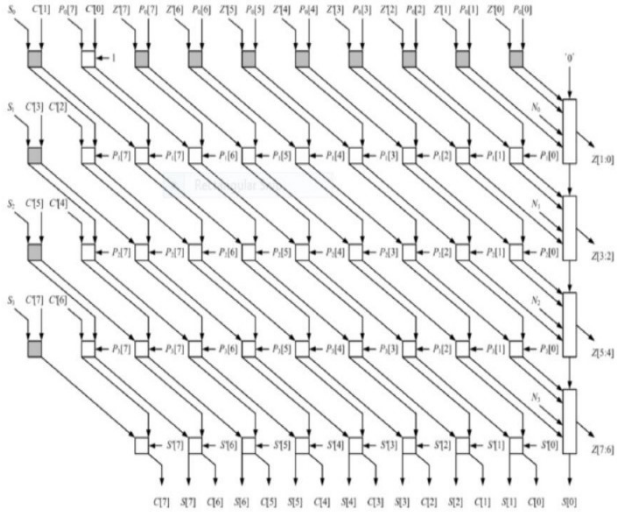


Fig 7 Architecture of CSA tree

IV. RESULTS AND COMPARISON

A. Simulated result

The Xilinx ISE Design Suite 14.5 is used to obtain the simulated output for MAC architecture using Vedic Multiplier. The simulated waveforms are obtained by assigning the input values at various levels of extraction and the corresponding outputs are obtained from the assigned inputs. The outputs obtained are complementary inputs. The simulated waveforms of the proposed work are shown here.

Partial Product Generation

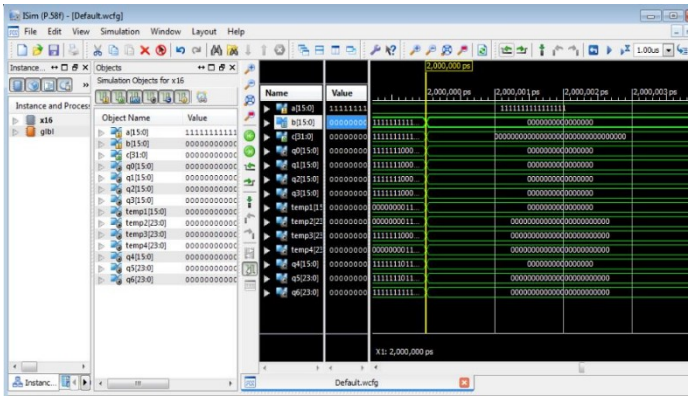


Fig 8 simulation output of Vedic multiplier

B. Hybrid Type CSA Architecture

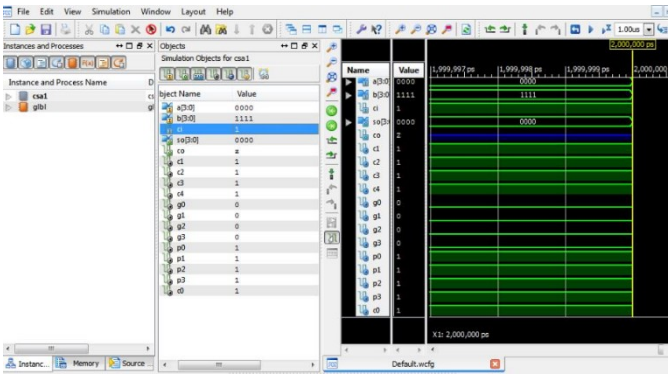


Fig 9 simulation output of CSA tree architecture

Fig 9 shows the simulated output of hybrid type Carry Save Adder architecture done by using Xilinx software.

C. Simulated Output of MAC

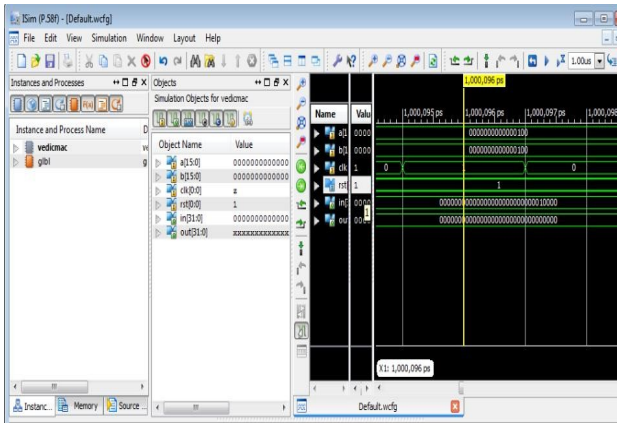


Fig 10 simulation output of proposed MAC unit

D. Performance Analysis

Simulation results shows that multiplier based on Vedic mathematics for multiplication of binary numbers is faster than multipliers based on Array and Booth multiplier. It also proves that as the number of bits increases to N, where N can be any number, the delay time is greatly reduced in Vedic Multiplier as compared to other multipliers. Vedic Multiplier has the advantages as over other multipliers also for power and regularity of structures.ther performance parameters of MAC unit using Vedic Multiplier can be compared with existing MAC in terms of power, delay, power delay product and throughput based on their simulation results.

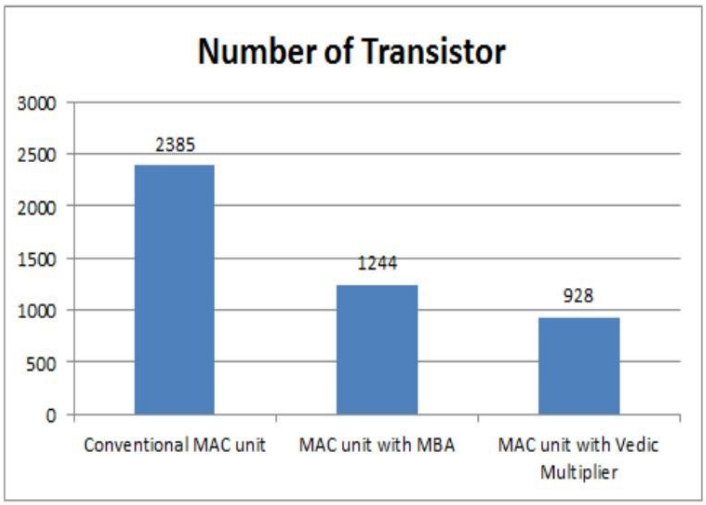


Fig 11(a) comparison of number of transistors

Fig 11(a) shows the number of transistors of MAC unit. The number of transistors in MAC unit with Vedic multiplier get decreased by 26%.

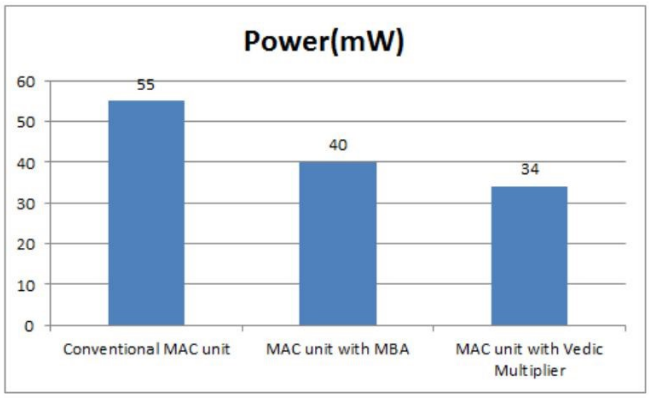


Fig 11(b) Comparison of Power

Fig 11(b) shows the comparison of power of MAC units.

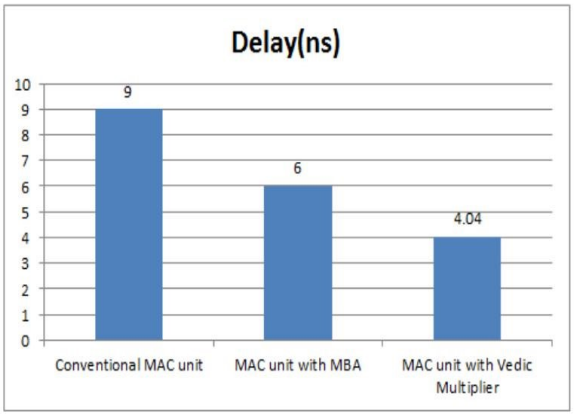


Fig 11(c) comparison of delay

Fig 11(c) shows the delay comparison of various MAC

units. By using Vedic multiplier the delay gets reduced by 33%.

V.CONCLUSION

In this paper, a new MAC architecture to execute the multiplication- accumulation operation, which is the key operation, for digital signal processing and multimedia information processing efficiently, was proposed. Use of Vedic multiplier reduces the number of partial products in which a group of multiplier bits are compared for partial product generation. Further by eliminating separate accumulation and implementing CSA with MAC there is a possibility for reducing hardware complexity and minimizing delay and power dissipation. By removing the independent accumulation process that has the largest delay and merging it to the compression process of the partial products, the overall MAC performance has been improved almost twice as much as in the previous work.

REFERENCES

[1] Akanksha Kant and Shobha Sharma (2015) "Applications of Vedic multiplier design". 978-1-4673-7231-2/15/$31.00 ©2015 IEEE.

[2] Hardik sangani, Tanay M.Modi and V.S Kanchana Bhaskaran,"Low power Vedic multiplier using Energy Recovery Logic", International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2014.

[3] Yogita Bansal, Charu Madhu and Pardeep kaur,"High Speed Vedic Multiplier Designs- A Review", proceedings of 2014 RAECS UIET Punjab University Chandigarh, 06-08 March, 2014.

[4] Aravind E Vijayan, A.John and D.Sen,"Efficient Implementation of 8-bit Vedic multipliers for Image Processing Application", International Conference on Contemporary Computing and Informatics (ICS), 2014.

[5] Rakshith Saligram, Rakshith T.R,"Optimised Reversible Vedic Multipliers for High Speed Low Power Operations", proceedings of 2013. IEEE conference on information and communication technologies (ICT 2013), 2013.

[6] Sumit Vaidya and Deepak Dandekar (2010) "Delay-power performance comparison of multipliers in VLSI circuit design". International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, July 2010