IMPLEMENTATION OF FIR FILTER BY USING MULTIPLIERS WITH OPTIMIZED PERFORMANCE ADDERS

BEERAKAYALA LATHASURESH 1, S. KANNAPPAN 2
1 PG SCHOLAR.DEPT OF ECE, GATES INSTITUTE OF TECHNOLOGY, AP, INDIA
2 ASSOC. PROFESSOR, DEPT OF ECE, GATES INSTITUTE OF TECHNOLOGY, AP, INDIA

Abstract— VLSI experiences a key position in many of the signal processing applications. Multiply and Accumulation process is one among the mostly used operation. Power, area and speed are the metrics used to determine the efficiency of a MAC unit. For certain cases each of these metrics plays a key role. In some cases, speed is only concentrated, so the other parameters are not given much priority in that case. This work focuses on two metrics. The power and area is concentrated for the better efficiency in this work. Through the deep analysis of adders, Ripple Carry Adder has shown less area and power consumption than other adders. The processes that are involved in MAC are multiplication, addition and accumulation. The addition of Vedic techniques in a MAC is always an added advantage. So, this work includes development of a 32-bit multiply and accumulate unit using Vedic sutra (Urdhva Tiryakbhyam sutra), accumulation unit involving ripple carry adder (RCA) and its implementation in a 4-tap FIR filter.

1.INTRODUCTION

Multiplication Accumulation Unit (MAC) performs a vital position in Digital Signal Processing applications, microprocessor, multimedia etc. The main process of the Multiplication Accumulation Unit (MAC) is to multiply two numbers and add the obtained product in the accumulator. So, in VLSI circuits computation plays a crucial role as it decides the power consumption of the design. Power, area and time decides the efficiency of a Multiplication Accumulation Unit (MAC).

The processes involved in the Multiplication Accumulation Unit (MAC) are multiplication and accumulation. Using Vedic's technique (Urdhva Tiryakbhyam sutra) the computation speed of multiplication is reduced far more than Wallace Tree multiplier and Booth's multiplier. Moreover they consume more power and area compared to Vedic multiplier when implemented inside a Multiplication Accumulation Unit (MAC) . So, the use of Vedic multiplier gives an added advantage for reduction of area and power in a MAC.

This work demonstrates a 32 X 32 Vedic multiplier designed from the initial 2 X 2 Vedic multiplier which is developed using Vedic technique (Urdhva Tiryakbhyam Sutra) and then this Vedic multiplier is modified to develop a 4 X 4 module. It is then modified to develop an 8 X 8 Vedic multiplier. Later the designed unit is modified to develop a 16 X 16 Vedic multiplier and finally modifying it to a 32 X 32 block (all using Vedic techniques). The adder that is used in the addition process is Ripple Carry Adder (RCA). The designed MAC is implemented in a 4 tap FIR filter. The authors in [2] have designed a low power multiplier. It also reports good improvement in speed. The concept of parallel multiplication is used to attain speed. Significant improvements in power are obtained by performing some architectural modifications.

High speed arithmetic operations are very important in many signal processing applications. Speed of the digital signal processor (DSP) is largely determined by the speed of its multipliers. In fact the multipliers are the most important part of all

digital signal processors; they are very important in realizing many important functions such as fast Fourier transforms and convolutions. Since a processor spends considerable amount of time in performing multiplication, an improvement in multiplication speed can greatly improve system performance.

Multiplication can be implemented using many algorithms such as array, booth, carry save, and Wallace tree algorithms. Booth algorithm reduces the number of partial products. However, large booth arrays are required for high speed multiplication and exponential operations which in turn require large partial sum and partial carry registers.

To improve the multiplication technique in processors like DSP or any other processors there is another technique taken from the ancient mathematics which is also called as Vedic mathematics. Vedic Mathematics hails from the ancient Indian scriptures called "Vedas" or the source of knowledge. This system of computation covers all forms of mathematics, be it geometry, trigonometry or algebra. The striking feature of Vedic Mathematics is the coherence in its algorithms which are designed the way our mind naturally works. This makes it the easiest and fastest way to perform any mathematical calculation mentally. Vedic Mathematics is believed to be created around 1500 BC and was rediscovered between 1911 to 1918 by Sri Bharti Krishna Tirthaji (1884-1960) who was a Sanskrit scholar, mathematician and a philosopher. He organized and classified the whole of Vedic Mathematics into 16 formulae or also called as sutras. These formulae form the backbone of Vedic mathematics. Great amount of research has been done all these years to implement algorithms of Vedic mathematics on digital processors. It has been observed that due to coherence and symmetry in these algorithms it can have a regular silicon layout and consume less area along with lower power consumption. The computational time required by the Vedic multiplier is less because the partial products are computed independently in parallel.

Arrangement of adders is another way of improving multiplication speed. Normally signal processing algorithms are developed using high level languages like C or Mat lab using floating point number representations. The algorithm to architecture mapping using floating point number representation consumes more hardware which tends to be expensive. Fixed point number representation is a good option to implement at silicon level. Hence our focus in this work is to develop optimized hardware modules for multiplication operation which is one of the most frequently used operation in signal processing applications like Fourier transforms, FIR and IIR filters, image processing systems, seismic signal processing, optical signal processing etc. Any attempt to come out with an optimized architecture for this basic block is advantageous during the product development stages. The advantage it provides over floating point multipliers is in the fact that Q format fraction multiplications can be carried out using integer multipliers which are faster and consume less die area.

2. LITERATURE SURVEY

Design & implementation of area efficient low power high speed MAC unit using FPGA by R. Pawar and S. S. Shriramwar

This paper illustrates the implementation of Multiply-Accumulate unit (MAC) to improve performance using the Ancient Indian Vedic Mathematics. Real-time signal processing needs high speed and high yield Multiplier-Accumulator (MAC) unit that consumes less power, which is always a key to achieve a high performance digital signal processing system. Speed of the multiplier is essential to MAC unit. Vedic Mathematics

is the ancient mathematics. There are sixteenth unique sutras in the Vedic mathematics. The Sutras saves effort and time required in solving the problems as compared to the formal methods. The Sutra "Urdhav Triyagbhyam" (Vertical and Crosswise) is used inside the MAC unit as a multiplier. The coding is done in VHDL, synthesis is done in Xilinx ISE series and the FPGA synthesis is done using Xilinx Spartan library. The results show that design of MAC unit using Vedic multiplication is much more efficient in terms of delay and speed compared to conventional multiplication.

Low power high speed multiplier using parallel multiplication by Madhav T hiyagarajan

This paper presents a low power and high speed row bypassing multiplier. The primary power reductions are obtained by tuning off MOS components through multiplexers when the operands of multiplier are zero. Analysis of the conventional DSP applications shows that the average of zero input of operand in multiplier is 73.8 percent. Therefore, significant power consumption can be reduced by the proposed bypassing multiplier. The proposed multiplier adopts ripple-carry adder with fewer additional hardware components. In addition, the proposed bypassing architecture can enhance operating speed by the additional parallel architecture to shorten the delay time of the proposed multiplier. Both unsigned and signed operands of multiplier are developed. Post-layout simulations are performed with standard TSMC 0.18 μm CMOS technology and 1.8 V supply voltage by Cadence Spectre simulation tools. Simulation results show that the proposed design can reduce power consumption and operating speed compared to those of counterparts. For a 16×16 multiplier, the proposed design achieves 17 and 36 percent reduction in power consumption and delay, respectively, at the cost of 20 percent increase of chip area in comparison with those of conventional array multipliers. In addition, the proposed design achieves averages of 11 and 38 percent reduction in power consumption and delay with 46 percent less chip area in comparison with those counterparts for both unsigned and signed multipliers. The proposed design is suitable for low power and high speed arithmetic applications.

3. EXISTING SYSTEM

The concept of the carry-select adder is to compute alternative results in parallel and subsequently selecting the correct result with single or multiple stages. In carry-select

adders both sum and carry bits are calculated for the two alternatives: carry "0" and "1". Once the carry-in is fired, the correct computation is chosen using multiplexers to produce the desired output. Therefore instead of waiting for the carry-in to calculate the sum, the sum is correctly output as soon as the carry-in gets there. The time taken to compute the sum is then avoided which results in a good improvement in speed.

Here, four 4X4 Vedic multiplier blocks and three carry select adders of 8 bits each are used. The arrangement of the carry select adders is made in a different way such that it requires less computation time. Some of the carry select adders are given with zero inputs, wherever required. The output of middle multipliers is added using first CSLA. The Output of first CSLA and first Vedic multiplier are added using second CSLA. Carry outputs from first two CSLAs are performed by OR operation and it is given as an input to the third CSLA to generate the final result

The design of 16×16 block is a similar arrangement of 8×8 blocks in an optimized

manner which is shown in Figure 5.The first step in the design of 16×16 block will be grouping the 8 bit (byte) of each 16-bit input.The LSB of two inputs will form vertical and crosswise product terms. Each input byte is handled by a separate 8×8 Vedic multiplier to produce sixteen partial product rows. These partial products rows are added in a 16-bit carry select adder optimally to generate final product bits. The schematic of a 16×16 block is designed by using the 8X8 Vedic multiplier. The partial products represent the Urdhva vertical and cross product terms. Then by using or gate, the final product is obtained

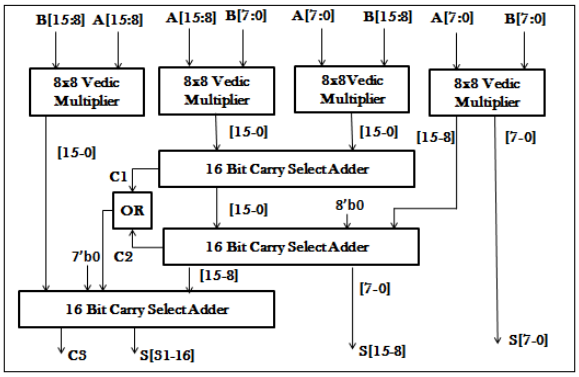

Figure 1. Block Diagram of 16X16 bit Vedic Multiplier

The use of carry select adder increases the hardware thereby more power consumtion which is twice that of rippe carry adder.The use of multiplexer increases the chip area.But consierable increase in the speed is achieved. So, optimisation of the carry select adder can be useful in increasing the speed of the mutiplier speed

4. PROPOSED SYSTEM

Urdhava Tiryakbhyam is a Sanskrit word which means vertically and crosswire in English. The method is a general multiplication formula applicable to all cases of multiplication. It is based on a novel concept through which all partial products are generated concurrently. Fig. 2demonstrates a 4 x 4 binary multiplication using this method. The method can be generalized for any N x N bit multiplication. This type of multiplier is independent of the clock frequency of the processor because the partial products and their sums are calculated in parallel. The net advantage is that it reduces the need of microprocessors to operate at increasingly higher clock frequencies. As the operating frequency of a processor increases the number of switching instances also increases. This results in more power consumption and also dissipation in the form of heat which results in higher device operating temperatures. Another advantage of Urdhva Tiryakbhyam multiplier is its scalability.

The processing power can easily be increased by increasing the input and output data bus widths since it has a regular structure. Due to its regular structure, it can be easily layout in a silicon chip and also consumes optimum area. As the number of input bits increase, gate delay and area increase very slowly as compared to other multipliers. Therefore Urdhava Tiryakbhyam multiplier is time, space and power efficient. The line diagram in fig. 2 illustrates the algorithm for multiplying two 4-bit binary numbers and . The procedure is divided into 7 steps and each step generates partial products. Initially as shown in step 1 of fig. 2, the least significant bit (LSB) of the multiplier is multiplied with least significant bit of the multiplicand vertical multiplication). This result forms the LSB of the product. In step 2 next higher bit of the multiplier is multiplied with the LSB of the multiplicand and the LSB of the multiplier is multiplied with the next higher bit of the multiplicand (crosswire multiplication).

In order to multiply two 8-bit numbers using 4-bit multiplier we proceed as follows. Consider two 8 bit numbers denoted as AHAL and BHBL where AH and BH corresponds to the most

significant 4 bits, AL and BL are the least significant 4 bits of an 8-bit number. When the numbers are multiplied according to Urdhava Tiryakbhyam (vertically and crosswire) method, we get,

AH AL
BH BL

(AH x BH) + (AH x BL + BH x AL) + (AL x BL).

Thus we need four 4-bit multipliers and two adders to add the partial products and 4-bit intermediate carry generated. Since product of a 4 x 4 multiplier is 8 bits long, in every step the least significant 4 bits correspond to the product and the remaining 4 bits are carried to the next step. This process continues for 3 steps in this case. Similarly, 16 bit multiplier has four 8 x 8 multiplier and two 16 bit adders with 8 bit carry. Therefore we see that the multiplier is highly modular in nature. Hence it leads to regularity and scalability of the multiplier layout.

Each block as shown above is 2x2 bit Vedic multiplier. First 2x2 bit multiplier inputs are A1A0 and B1B0. The last block is 2x2 bit multiplier with inputs A3 A2 and B3 B2. The middle one shows two 2x2 bit multiplier with inputs A3 A2 & B1B0 and A1A0 & B3 B2. So the final result of multiplication, which is of 8 bit, S7 S6S5S4 S3 S2 S1 S0. To understand the concept, the Block diagram of 4x4 bit Vedic multiplier is shown in Fig. 5. To get final product (S7 S6 S5 S4 S3 S2 S1 S0), four 2x2 bit Vedic multiplier (Fig. 3) and three 4-bit Ripple-Carry (RC) Adders are required. The proposed Vedic multiplier can be used to reduce delay. Early literature speaks about Vedic multipliers based on array multiplier structures. On the other hand, we proposed a new architecture, which is efficient in terms of speed. The arrangements of RC Adders shown in Fig. 5, helps us to reduce delay. Interestingly, 8x8 Vedic multiplier modules are implemented easily by using four 4x4 multiplier modules.

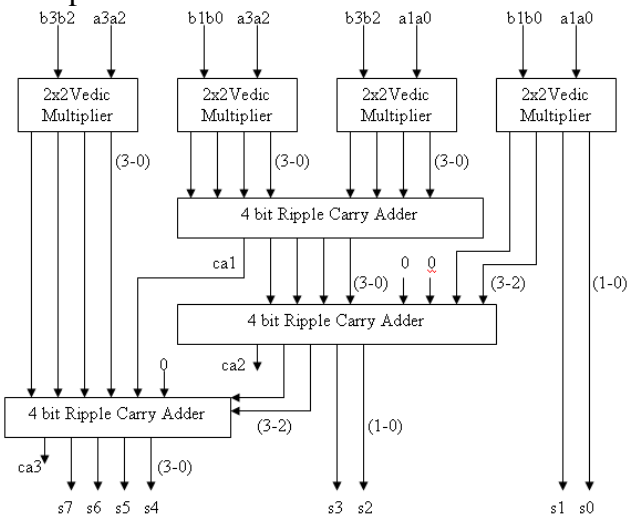

Fig. 2 Block Diagram of 4x4 bit Vedic Multiplier

Vedic Multiplier for 8x8 bit Module

The 8x8 bit Vedic multiplier module as shown in the block diagram in Fig. 6 can be easily implemented by using four 4x4 bit Vedic multiplier modules as discussed in the previous section. Let's analyze 8x8 multiplications, say A= A7 A6 A5 A4 A3 A2 A1 A0 and B= B7 B6 B5B4 B3 B2 B1B0. The output line for the multiplication result will be of 16 bits as – S15 S14 S13 S12 S11 S10 S9 S8 S7 S6S5S4 S3 S2 S1 S0. Let's divide A and B into two parts, say the 8 bit multiplicand A can be decomposed into pair of 4 bits AH-AL. Similarly multiplicand B can be decomposed into BH-BL. The 16 bit product can be written as:

Using the fundamental of Vedic multiplication, taking four bits at a time and using 4 bit multiplier block as discussed we can perform the multiplication. The outputs of 4x4 bit multipliers are added accordingly to obtain the final product. Here total three 8 bit Ripple-Carry Adders are required as shown in Fig. 6.

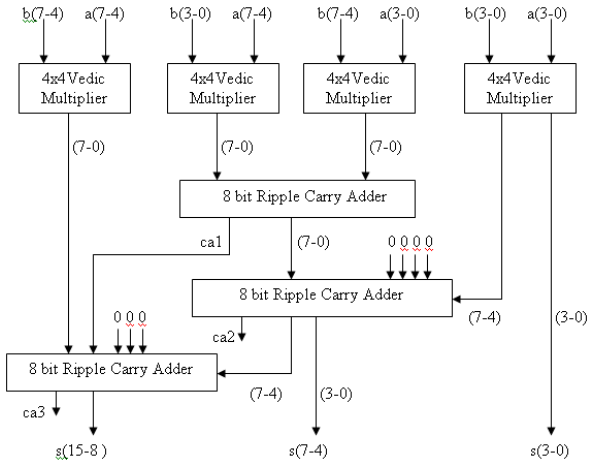

Fig 3 . Block Diagram of 8x8 bit Vedic Multiplier

Multiplication technique in ancient times includes multiplication of both smaller numbers and also for the larger numbers

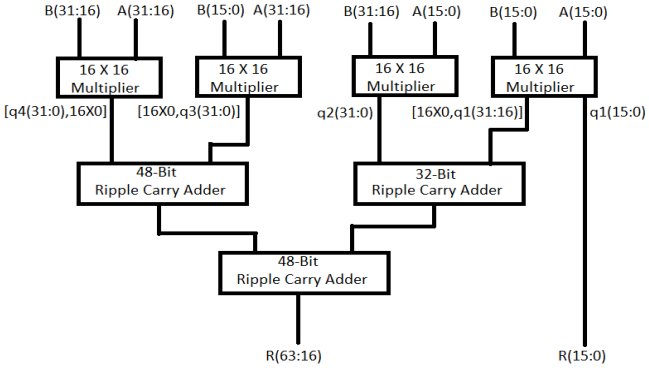

Fig.4.Proposed 32-Bit Vedic Multiplier Architecture

Fig.2 explains the multiplication architecture of the Urdhva Tiryakbhyam sutra for a 32 X 32-bit multiplication. This is developed using a 2x2 Vedic multiplier using ripple carry adders. So, here two 32-bit numbers A (31:0) and B (31:0) are taken

A (15:0) X B (15:0)
A (15:0) X B (31:16)
A (31:16) X B (15:0)
A (31:16) X B (31:16)

The result from the first multiplication i.e. A (15:0) X B (15:0) is stored in q1(31:0). Out of this q1(31:0), the q1(15:0) is directly stored in the result R (15:0). The other part of q1 i.e., q1(31:16) is appended with 16 zeros before and sent to 32-Bit Ripple Carry Adder (RCA) as input. The result from the second multiplication i.e. A (15:0) X B (31:16) is stored in q2(31:0) and given as other input to 32-Bit Ripple Carry Adder. The result from third multiplication i.e. A (31:16) X B (15:0) is stored in q3(31:0) and appended with 16 zeros before q3 and sent as input to 48-Bit Ripple Carry Adder. The result of fourth multiplication i.e. A (31:16) X B (31:16) is stored in q4(31:0) and 16 zeros are appended after q4 and sent as other input for 48-Bit Ripple Carry Adder.

Finally, result from 32-Bit RCA and 48-Bit RCA is added using a 48-Bit Ripple Carry Adder and the result R (63:16) is obtained. By combining R (63:16) and R (15:0) are combined to get the final result i.e., R (63:0).

5. ACCUMLATION UNIT

Basically, a Multiply Accumulation Unit (MAC) is made up of a multiplier, adders and accumulator. A multiplier is used to multiply two numbers and obtain their product. Addition task is carried out suitable adders. Accumulator will add the present product with the previously multiplied products. The input is given to the multiplier and the output is obtained from the accumulator. So, every time the input is given the product is stored in the accumulator.

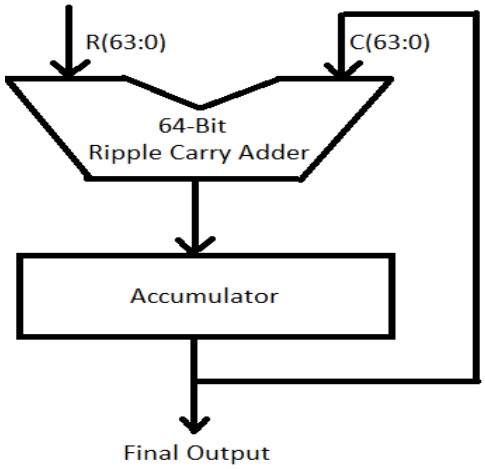

Fig.5.Accumulation of 64-Bit Number

The result obtained from the 32-bit Vedic multiplier is passed into accumulator and the accumulation process is done for every cycle and added to the previous value stored in the accumulator.

6.IMPLEMENTATION OF FIR FILTER USING PROPOSED MULTIPLIER

A filter is a device or process that removes some unwanted component or feature from a signal. Filtering is a class of signal processing, the defining feature of filter being the complete or partial suppression of some aspect of the signal. There are two main kinds of filter, analog and digital. Filters can be classified in several different groups, depending on what criteria are used for classification. The two major types of digital filters are finite impulse response digital filters (FIR filters) and infinite impulse response digital filters (IIR).

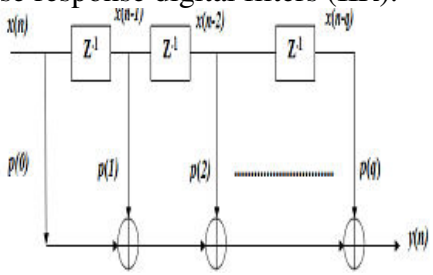

Fig4.7. Finite Impulse Response Filter Realization

FIR filters are one of the primary types of filters used in Digital Signal Processing. FIR filters are said to be finite because they do not have any feedback. Therefore, if we send an impulse through the system (a single spike) then the output will invariably become zero as soon as the impulse runs through the filter. A non-recursive filter has no feedback. The Finite Impulse Response Filter Realization is as shown in figure 10.

Finite Impulse response (FIR) digital filter is widely used in several digital signal processing applications, such as speech processing, loud speaker equalization, echo cancellation, adaptive noise cancellation, and various communication applications, including software-defined radio (SDR) and so on. Many of these applications require FIR filters of large order to meet the stringent frequency specifications. Very often these filters need to support high sampling rate for high-speed digital communication. The number of multiplications and additions required for each filter output, however, increases linearly with the filter order. Since there is no redundant computation available in the FIR filter algorithm, real-time implementation of a large order FIR filter in a resource constrained environment is a challenging task. Filter coefficients very often remain constant and known apiarian signal processing applications. Finite Impulse Response (FIR) filters are widely used in digital signal processing. An N-tap FIR filter is defined by the following input-output equation 1

$$out(n) = \sum_{i=0}^{N-1} x(n-i)\, h(i)$$

-→1

Where {h (i): i = 0… N-1} are the filter coefficients.

An FIR filter implements a convolution operation [1], which is often built on the assumption of infinite length signals. Finite length signals (e.g. images) on the other hand, have discontinuities at the boundaries. Thus emerges the problem of which values to use at these regions. A usually recommended solution is to extend each row by reflection at the signal edges. The number of extra samples introduced at the signal boundaries is equal to N-1. They can be partitioned unequally between the left and the right side signal. We will refer by α and μto the number of samples introduced respectively at the left side and the right side input signal (α+μ=N-1).

5. RESULTS

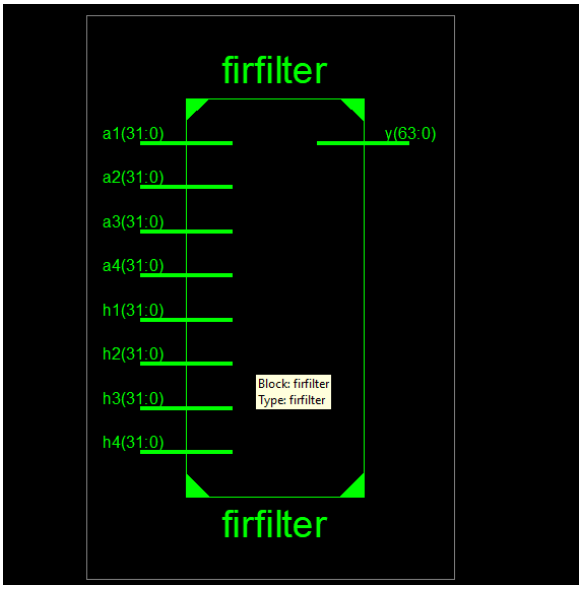

Fig 5 Block Diagram of fir filter

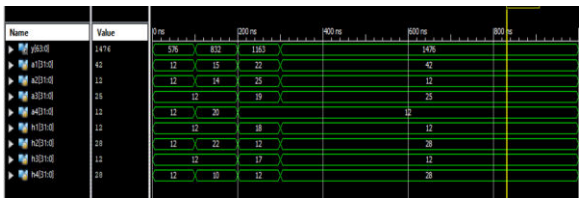

Fig 6 Simulation Wave from of fir filter

6. COMPRESSION TABLE

Table 1 compression of area-delay product and power

	AREA (no of LUTs)	DELAY (ns)	AREA-DELAY PRODUCT	POWER (mw)
8 BIT RCA MULTIPLIER	171	23.401	4001.571	0.305
8 BIT CSLA MULTIPLIER	177	21.582	3820.014	0.306
16 BIT RCA MULTIPLIER	756	41.305	31226.58	0.489
16 BIT CSLA MULTIPLIER	814	35.302	28735.828	0.494
32 BIT RCA MULTIPLIER	3186	72.501	230988.186	0.492
32 BIT CSLA MULTIPLIER	3475	53.377	185485.075	0.512
FIR FILTER USING RCA	13105	83.449	1093599.145	0.498
FIR FILTER USING CSLA	14538	75.162	1092705.156	0.516

Table 2 compression of power

POWER COMPARISION		USING RCA MULTIPLIER			
		8-BIT	16-BIT	32-BIT	4 TAP FIR FILTER
POWER	EXISTING SYSTEM	0.608	0.628	0.878	1.649
	PROPOSED SYSTEM	0.305	0.489	0.492	0.498

CONCLUSION & FUTURE SCOPE

A MAC unit is realized using Verilog and the synthesis is done in xilinx. A comparison is carried out with the MAC used in [1]. Power and area reductions have been reported. The power has been reduced by 9% and area by 5%. Also, the proposed MAC is utilized in 4 tap FIR filter. It highlights significant improvements in power and area. As a future scope the designed MAC can be implemented in an IIR filter. Also, implementation of MAC using other Vedic techniques is a suitable choice. This project presents a highly efficient method of multiplication – "Urdhva Tiryakbhyam Sutra" based on Vedic mathematics. It gives us method for hierarchical multiplier design and clearly indicates the computational advantages offered by Vedic methods. Hence our motivation to reduce delay is finely fulfilled.. Future work lies in the direction of introducing pipeline stages in the multiplier architecture for maximizing throughput and also we can implement for 64 – bit also.

REFERENCES

[1] Desu Sai Manikanta, Katakam Shanmukha Satya Ramakrishna "Hardware Realization of Low power and Area Efficient Vedic MAC in DSP Filters", 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)

[2] R. Pawar and S. S. Shriramwar, "Design & implementation of area efficient

low power high speed MAC unit using FPGA," 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), Chennai, India, 2017, pp. 2683-2687, doi: 10.1109/ICPCSI.2017.8392205.

[3] Madhav T hiyagarajan, "Low power high speed multiplier using parallel multiplication", International Journal of VLSI Design, Vol:2, Issue: 1, pp: 75-78, 2017.

[4] T asmiya Shaikh, Manjunatha Beleri, "FPGA Implementation of Multiply Accumulate (MAC) Unit based on Block Enable T echnique", International Journal of Innovative Research in Computer and Communication Engineering, (An ISO 3297: 2007 Certified Organization) Volume: 03 Issue: 04, pp.2785-2790, April 2015

[5] H.S. Dhillon and A. Mitra, "A Reduce Bit Multiplication Algorithm for Digital Arithmetic", International Journal of Computational and Mathematical Sciences, pp. 64-69, 2008.

[6] Lachireddy D., Ramesh S.R. (2020) Power and Delay Efficient ALU Using Vedic Multiplier. In: Sengodan T ., Murugappan M., Misra S. (eds) Advances in Electrical and Computer Technologies. Lecture Notes in Electrical Engineering, vol 672. Springer, Singapore.

[7] K. Paldurai, K. Hariharan, G. C. Karthikeyan and K. Lakshmanan, "Implementation of MAC using area efficient and reduced delay Vedic multiplier targeted at FPGA architectures," 2014 International Conference on Communication and Network T echnologies, Sivakasi, India, 2014, pp. 238-242, doi: 10.1109/CNT .2014.7062762.

[8] Prabhu, E., Mangalam, H. & Gokssul, P.R. A Delay Efficient Vedic Multiplier. Proc. Natl. Acad. Sci., India, Sect. A Phys. Sci. 89, 257–268 (2019).

[9] Pande, Kuldeep, AbhinavParkhi, ShashantJaykar, and AtishPeshattiwar. "Design and Implementation of Floating-Point Divide-Add Fused Architecture." In Communication Systems and Network T echnologies (CSNT), 2015 Fifth International Conference on, pp. 797-800. IEEE, 2015.

[10] A. S. Krishna Vamsi and S. R. Ramesh, "An Efficient Design of 16 Bit MAC Unit using Vedic Mathematics," 2019 International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 2019, pp. 0319-0322, doi: 10.1109/ICCSP.2019.8697985

[11] R. K. Kodali, L. Boppana and S. S. Yenamachintala, "FPGA implementation of Vedic floating point multiplier," 2015 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES), Kozhikode, India, 2015, pp. 1-4, doi: 10.1109/SPICES.2015.7091534.

[12] Sachin Raghav, Dr. Rinkesh Mitta, "Implementation of Fast and Efficient Mac Unit on FPFA", I.J. Mathematical Sciences and Computing, pp.24-33, 2016

[13] Y. Bansal, C. Madhu and P. Kaur, "High speed Vedic multiplier designs-A review," 2014 Recent Advances in Engineering and Computational Sciences (RAECS), Chandigarh, India, 2014, pp. 1-6, doi: 10.1109/RAECS.2014.6799502.

[14] Mothukuri Ramalakshmi Bala1, Chodisetti L S S P avanKumar, "Power Efficient Compressor based MAC Architecture for DSP Applications", International Journal of Engineering Science and Computing, (IRJET) Volume: 09 Issue: 01| January 2019 (19550) pp.19547-19550.

[15] Mehta Shantanu Sheetal, T . Vigneswaran, "High Speed and Efficient 4-T ap FIR Filter Design Using Modified ET A and Multipliers", 2014 International Journal of Engineering and

Technology, ISSN : 0975-4024, Vol 6 No 5 Oct-Nov 2014, pp. 2159-2170.
[16] Nusrat Jabeen M.Anees ,Ashish B. Kharate, "Design of area efficient digital FIR filter using MAC," International Research Journal of Engineering and T echnology (IRJET) Vol. 4, no.9, Sep.2017,pp.1140-1143.