FPGA-BASED RESOURCE-OPTIMIZED 64-BIT MULTIPLY-ACCUMULATE ARCHITECTURE USING HIERARCHICAL VEDIC ARITHMETIC AND REVERSIBLE DKG ADDERS

Tabassum Nousheen[1], Sameena Begum[2]
[1]PG Scholar, Department of ECE, Shadan Women's College of Engineering and Technology, Hyderabad,
tabassumnousheen18@gmail.com
[2]Asst Professor, Department of ECE, Shadan Women's College of Engineering and Technology
sameena@gmail.com

ABSTRACT

The design of Multiplier-Accumulator (MAC) unit can be implemented by using the Vedic multiplier along with the reversible logic gates. The designing of Vedic multiplier is designed by using the new sutra called "UrdhavaTriyagbhayam". The performance of the MAC operation depends on the multiplier unit and the adder units. Here the designing of a multiplier and an adder can be designed by using the reversible gates to get the high speed of operation and also a Vedic multiplier is used for the higher performance, lesser area and to reduce the partial products. Nowadays reversible computing will take a more preferable for low power dissipation, higher speed of operation. Here, we proposed an 8, 16, 32, 64-bit Vedic multiplier is designed by using the RCA based DKG adder and Vedic multiplier is designed by using the CSLA based DKG adder out of these the proposed DKG gate adder Based on CSLA is having the high speed of operation. The comparative analysis is carried out among the ripple carry adder (RCA), carry select adder. Finally, it has been proved that the proposed CSLA based DKG gate with Vedic multiplier- adder is havingthe high speed of operation. The overall Simulation and synthesis process is carried out with Xilinx ISE14.7 and is dumped on the FPGA vertex7 board.

1. 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, Vedic multiplier using carry look ahead adder is one of the most popular Karatsubamethod. 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 basic method of multiplier explains below

```
   123
 x 456
 =====
   738  (this is 123 x 6)
  615   (this is 123 x 5, shifted one position to the left)
+ 492    (this is 123 x 4, shifted two positions to the left)
 =====
 56088
```

The binary multiplication also happens in same way of digit multiplication as shown in below example here by getting partial products and gates are used and we are using adder (half adder ,fulladder)adding the columns
An example of 4-bit multiplication method is shown below:

```
        1 0 1 0 X 1 0 1
        ---------------
          1 0 1 0
        0 0 0 0
      1 0 1 0
      ---------------
      1 1 0 0 1 0
```

Although the method is simple as it can be seen from this example, the addition is done serially as well as in parallel. To improve on the delay and area the CRAs are replaced with Carry Save Adders, in which every carry and sum signal is passed to the adders of the next stage. Final product is obtained in a final adder by any fast adder (usually carry ripple adder). In array multiplication we need to add, as many partial products as there are multiplier bits. This arrangements are shown in the figure below

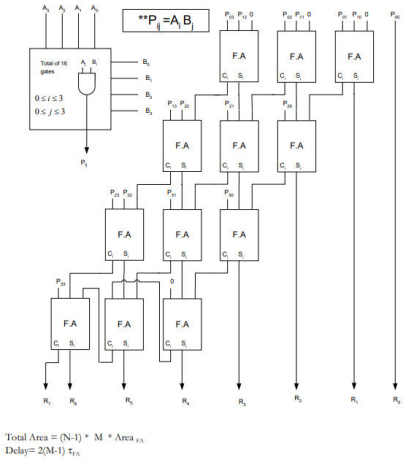


Fig 1.1: array multiplier

In applications like multimedia signal processing and data mining which can tolerate error, exact computing units are not always necessary. They can be replaced with their approximate counterparts. Research on approximate computing for error tolerant applications is on the rise. Adders and multipliers form the key components in these applications. In, approximate full adders are proposed at transistor level and they are utilized in digital signal processing applications.

IMPLEMENTATION OF WALLACE MULTIPLIER

The Wallace tree has three steps:

1.Multiply (that is - AND) each bit of one of the arguments, by each bit of the other, yielding results. Depending on position of the multiplied bits, the wires carry different weights, for example wire of bit carrying result is 32.

2.Reduce the number of partial products to two by layers of full and half adders.

3.Group the wires in two numbers, and add them with a conventional adder.

4.The second phase works as long as there are three or more wires with the same weight add a following layer:

•Take any three wires with the same weights and input them into a full adder. The result will be an output wire of the same weight and an output wire with a higher weight for each three input wires.

•If there are two wires of the same weight left, input them into a half adder.

•If there is just one wire left, connect it to the next layer.

a)Steps involved in WALLACE TREE multipliers Algorithm:

∟Multiply (that is - AND) each bit of one of the arguments, by each bit of the other, yielding N results. Depending on position of the multiplied bits, the wires carry different weights.

∟Reduce the number of partial products to two layers of full adders.

∟Group the wires in two numbers, and add them with a conventional adder.

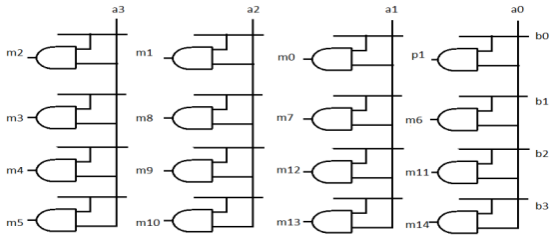


Fig 1.2 Product terms generated by a collection of AND gates

b)WALLACE TREE Multiplier Using Adder

Ripple Carry Adder is the method used to add a greater number of additions to be performed with the carry in sand carry outs that is to be chained. Thus, multiple adders are used in ripple carry adder. It is possible to create a logical circuit using several full adders to add multiple-bit numbers. Each full adder inputs a Cin, which is the Cout of the previous adder. This kind of adder is a ripple carry adder, since each carry bit "ripples" to the next full adder.The proposed architecture of WALLACE multiplier algorithm using RCA is shown in Fig

Take any 3 values with the same weights and give them as input into a full adder. The result will be an output wire of the same weight.

∟Partial product obtained after multiplication is taken at the first stage. The data are taken with 3 wires and added using adders and the carry of each stage is added with next two data in the same stage.

∟Partial products reduced to two layers of full adders with same procedure.

At the final stage, same method of ripple carry adder method is performed and thus product terms p1 to p8 isobtained .

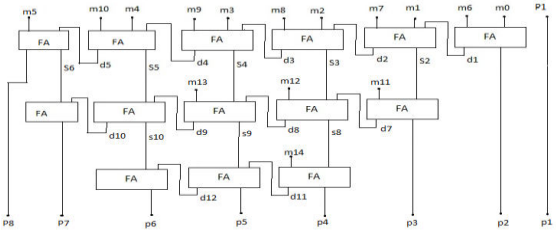


Fig 1.3:4x4 WALLACE Multiplier

3.Wallace tree multiplier operation

•A fast way to multiply two binary integers.

•Any multiplier has three stages.

Stage1: partial products.

Stage2:partial product addition.

Stage3:final addition.

Stage1 partial products

•Works exactly like "long hand" multiplication.Numbers are binary integers.Products(each blue square) are the result of a simple and gate.All products are done simultaneously.Fast (however long and gates takes).Trick is in adding up the columns

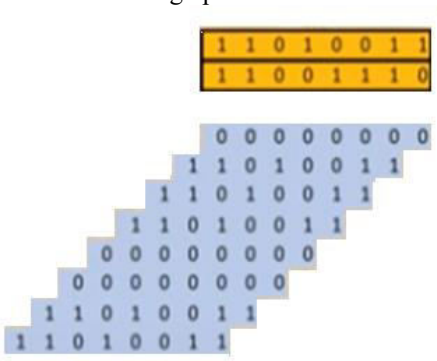


Fig 1.4: Stage1 partial products

Stage2:partial product addition ,step1

To add up columns, add up three rows at a time.The result for each set of three rows is a set of two rows.Each resulting set of two rows has a row for the

sum and a row for the carry-out.Odd rows are left alone.
red: full adder output.
yellow: half adder output.
green :left alone.

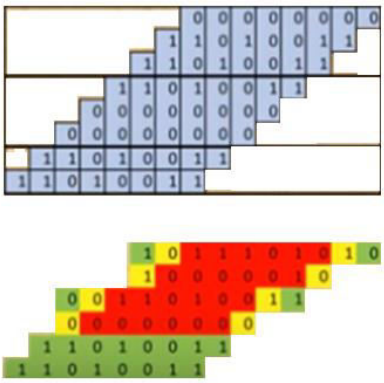


Fig 1.5: partial product addition ,step1

Stage2 :partial products addition,step2

•Repeat the same process. This time, three are two sets of three rows .Result in two set of two rows . Gray boxes indicate that summation bits have been moved down to the carryout row.

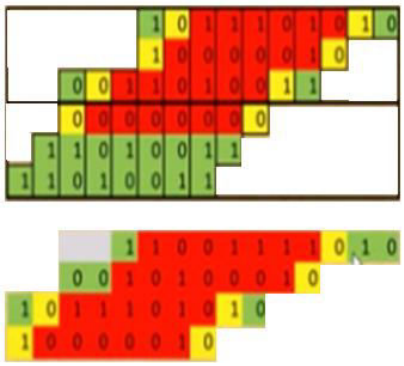


Fig 1.6 :partial products addition,step2

Stage2:partial products addition,step4

•Repeat the process one last time.Remaining three rows become two rows.In this example, stage2 has 4 steps ,and 4 full adder delays.The five LSB have already been calculated.

Fig 1.7: partial products addition,step4

Stage3:final addition

•Final result is calculated by adding the final two rows. In this example, the 5 LSBs do no need to be added. The saving from already having 5bits offsets the delay from doing stage2Result is that Wallace tree multiplication takes about the same amount of time as a 2N _bit ripple carry adder.

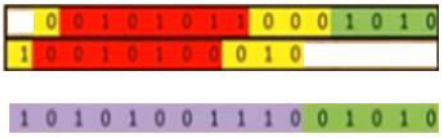


Fig 1.8: final addition

VEDIC MULTIPLIER USING CARRY SAVE ADDER

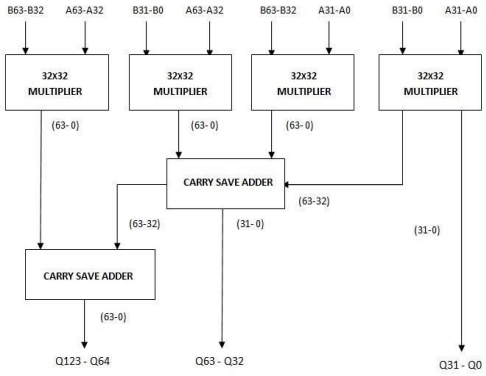


Fig.1.9: 64-bit Vedic multiplier using carry save adder

CARRY SAVE ADDER

The carry-save adderreduces the addition of 3 numbers to the addition of 2 numbers. The propagation delay is 3 gates regardless of the number of bits. The carry-save unit consists of n full adders, each of which computes a single sum and carries bit based solely on the corresponding bits of the three input numbers. The entire sum can then be computed by shifting the carry sequence left by one place and appending a 0 to the front (most significant bit) of the partial sum sequence and adding this sequence with RCA produces the resulting n + 1-bit value. This process can be continued indefinitely, adding an input for each stage of full adders, without any intermediate carry propagation. These stages can be arranged in a binary tree structure, with cumulative delay logarithmic in the number of inputs to be added, and invariant of the number of
bits per input. The main application of carry save algorithm is, well known for multiplier architecture is used for efficient CMOS implementation of much wider variety of algorithms for high-speed digital signal processing .CSA applied in the partial product line of array multipliers will speed up the carry propagation in the array. The design schematic of Carry Save Adder is shown in Figure (4).

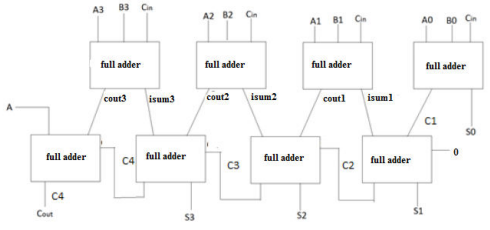


Fig1.10 : bit carry save adder

VEDIC MULTIPLIER USING KOGGE STONE ADDER:

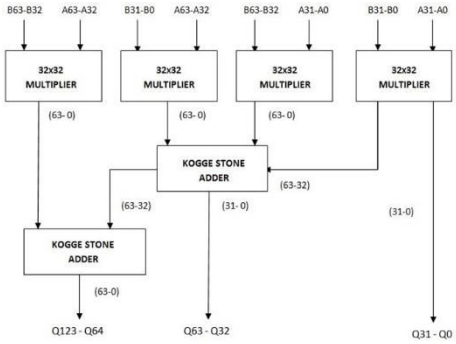


Fig 1.11 :64-bit Vedic multiplier using Kogge Stone Adder

KOGGE STONE ADDER

KSA is a parallel prefix form carry look ahead adder. It generates carry in O (logn) time and is widely considered as the fastest adder and is widely used in the industry for high performance arithmetic circuits. In KSA, carries are computed fast by computing the min parallel at the cost of increased area.

The complete functioning of KSA can be easily comprehended by analyzing It in terms of three distinct parts :

1. Pre processing

This step involves computation of generate and propagate signals corresponding too each pair o fbits in A and B.

$$pi = Ai \text{ xor } Bi$$
$$gi = Ai \text{ and } Bi$$

2 . Carry look ahead network

This block differentiates KSA from other adders and is the main force behind its high performance. This step involves computation of carries corresponding to each bit . It uses group propagate and generate as intermediate signals .

$$Pi:j = Pi:k+1 \text{ and } Pk:j$$
$$Gi:j = Gi:k+1 \text{ or } (Pi:k+1 \text{ and} Gk:j)$$

3. Post processing

This is the final step and is common to all adders of this family (carry look ahead). It involves computation of sum bits.

$$Si = pi \text{ xor } Ci-1$$

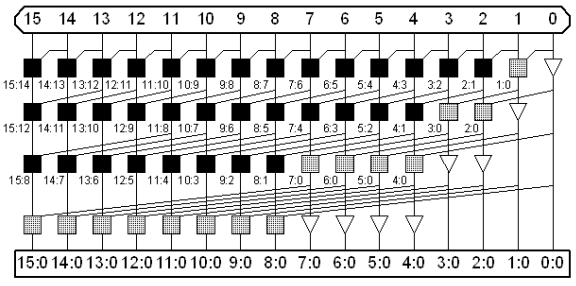


Fig 1.12 -bit Kogge Stone Adder

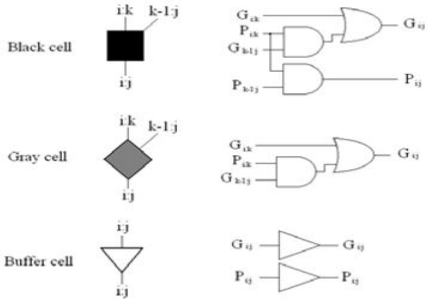


Fig 1.13: Complex logic cells inside the Prefix Carry Tree

2. LITERATURE VIEW

Vijay Kumar Reddy Modified High Speed Vedic Multiplier Design and Implementation The proposed research work specifies the modified version of binary Vedic multiplier using Vedic sutras of ancient Vedic mathematics. It provides modification in preliminarily implemented Vedic multiplier. The modified binary Vedic multiplier is preferable has shown improvement in the terms of the time delay and also device utilization. The proposed technique was designed and implemented in Verilog HDL. For HDL simulation, modelsim tool is used and for circuit synthesis, Xilinx is used. The simulation has been done for 4-bit, 8-bit,16-bit, multiplication operation. Only for 16-bit binary Vedic multiplier technique the simulation results are shown. This modified multiplication technique is extended for larger sizes. The outcomes of this multiplication technique are compared with existing Vedic multiplier techniques.

A. Momeni, J. Han, P. Montuschi, and F. Lombardi, “Design and Analysis of Approximate Compressors for Multiplication”, Inexact (or approximate) computing is an attractive paradigm for digital processing at nanometric scales. Inexact computing is particularly interesting for computer arithmetic designs. This paper deals with the analysis and design of two new approximate 4-2 compressors for utilization in a multiplier. These designs rely on different features of compression, such that imprecision in computation (as measured by the error rate and the so-called normalized error distance) can meet with respect to circuit-based figures of merit of a design (number of transistors, delay and power consumption). Four different schemes for utilizing the proposed approximate compressors are proposed and analyzed for a Dadda multiplier. Extensive simulation results are provided and an application of the multipliers to image processing is presented. The results show that the proposed designs accomplish significant reductions in power dissipation, delay and transistor count compared to an exact design; moreover, two of the proposed multiplier designs provide excellent capabilities for image multiplication with respect to average normalized error distance and peak signal-to noise ratio (more than 50 dB for the considered image examples).

C. Liu, J. Han, and F. Lombardi, “A Low-Power,

High-Performance Multiplier with Configurable Partial Error Recovery", Proc. of IEEE Design, Automation & Test in Europe Conference & Exhibition (DATE), [Approximate circuits have been considered for error-tolerant applications that can tolerate some loss of accuracy with improved performance and energy efficiency. Multipliers are key arithmetic circuits in many such applications such as digital signal processing (DSP). In this paper, a novel multiplier with a lower power consumption and a shorter critical path than traditional multipliers are proposed for high-performance DSP applications. This multiplier leverages a newly-designed approximate adder that limits its carry propagation to the nearest neighbors for fast partial product accumulation. Different levels of accuracy can be achieved through a configurable error recovery by using different numbers of most significant bits (MSBs) for error reduction. The multiplier has a low mean error distance, i.e., most of the errors are not significant in magnitude. Compared to the Wallace multiplier, a 16-bit multiplier implemented in a 28nm CMOS process shows a reduction in delay and power of 20% and up to 69%, respectively. It is shown that by utilizing an appropriate error recovery, the proposed multiplier achieves similar processing accuracy as traditional exact multipliers but with significant improvements in power and performance.

G. Zervakis, et al., "Design-Efficient Approximate Multiplication Circuits Through Partial Product Perforation" Approximate computing has received significant attention as a promising strategy to decrease power consumption of inherently error tolerant applications. In this paper, we focus on hardware-level approximation by introducing the partial product perforation technique for designing approximate multiplication circuits. We prove in a mathematically rigorous manner that in partial product perforation, the imposed errors are bounded and predictable, depending only on the input distribution. Through extensive experimental evaluation, we apply the partial product perforation method on different multiplier architectures and expose the optimal architecture-perforation configuration pairs for different error constraints. We show that, compared with the respective exact design, the partial product perforation delivers reductions of up to 50% in power consumption, 45% in area, and 35% in critical delay. In addition, the product perforation method is compared with the state-of-the-art approximation techniques, i.e., truncation, voltage over scaling, and logic approximation, showing that it outperforms them in terms of power dissipation and error.

T. Yang, T. Ukezono, and T. Sato "A Low-Power High-Speed Accuracy-Controllable Multiplier Design", Multiplication is a key fundamental function for many error-tolerant applications. Approximate multiplication is considered to be an efficient technique for trading off energy against performance and accuracy. This paper proposes an accuracy-controllable multiplier whose final product is generated by a carry-maskable adder. The proposed scheme can dynamically select the length of the carry propagation to satisfy the accuracy requirements flexibly. The partial product tree of the multiplier is approximated by the proposed tree compressor. An 8 × 8multiplier design is implemented by employing the carry maskable adder and the compressor. Compared with a conventional Wallace tree multiplier, the proposed multiplier reduced power consumption by between 47.3% and 56.2% and critical path delay by between 29.9% and 60.5%, depending on the required accuracy. Its silicon area was also 44.6% smaller. In addition, results from an image processing application demonstrate that the quality of the processed images can be controlled by the proposed multiplier design.

A. Cilardo, et al., "High-Speed Speculative Multipliers Based on Speculative Carry-Save Tree", Sacrificing exact calculations to improve digital circuit performance is at the foundation of approximate computing. In this paper, an approximate multiply-and-accumulate (MAC) unit is introduced. The MAC partial product terms are compressed by using simple OR gates as approximate counters; moreover, to further save energy, selected columns of the partial product terms are not formed. A compensation term is introduced in the proposed MAC, to reduce the overall approximation error. A MAC unit, specialized to
perform 2D convolution, is designed following the proposed approach and implemented in TSMC 40nm technology in four different configurations. The proposed circuits achieve power savings more than 60%, compared to standard, exact MAC, with tolerable image quality degradation.

J. Liang, et al., "New Metrics for The Reliability of Approximate and Probabilistic Adders", Approximate/inexact computing has become an attractive approach for designing high performance and low power arithmetic circuits. Floating-point (FLP) arithmetic is required in many applications, such as digital signal processing, image processing and machine learning. Approximate FLP multipliers with variable accuracy are proposed in this paper; the accuracy and the circuit requirements of these designs are analyzed and assessed according to different metrics. It is shown that the proposed approximate FLP multiplier designs further reduce delay, area, power consumption and power-delay product (PDP) while incurring about half of the normalized mean error distance (NMED) compared with the previous designs. The proposed IFLPM24–15 is the most efficient design when considering both PDP and NMED. Case studies with three error-tolerant applications show the validity of the proposed approximate designs.

3. IMPLEMENTATION OF A MAC BY 64x64 VEDIC MULTIPLIER USING DKG ADDERS

A MAC unit is a foreseeable element of a many digital signal processing (DSP) applications involving multiplications/accumulations. It is also used for performing the high-speed digital DSP systems. There are a number of applications in DSP including the convolution, filtering, and inner products. The discrete cosine transforms (DCT) or discrete wavelet transforms (DWT) are the nonlinear functions generally use in DSP methods. Because they are essentially accomplished by cyclic application of addition and multiplication, the overall speed of the addition and multiplication arithmetic calculations are determined by the speed of execution and the entire calculation performance.

Multiplication-and-accumulate operations are distinctive for digital filters. Then, the basic functionality of the MAC unit which enables the high-speed filtering as well as other distinctive processing units for DSP applications, while the MAC unit operates totally independent of the CPU, it individually processes the data for reducing the load of the CPU. Optical communication system is one of the best applications is completely based on the DSP, which requires enormous data for fast processing of digital data. The multiplication and addition operations are also required for Fast Fourier Transform (FFT). The 64-bit MAC unit which can deal a large number ofbits and it needs more amount of memory. Fundamentally it consists of the multiplier and an accumulator unit which contains the sum of previous successive product terms. The MAC unit inputs are getting from the corresponding memory location which is connected to the multiplier block.

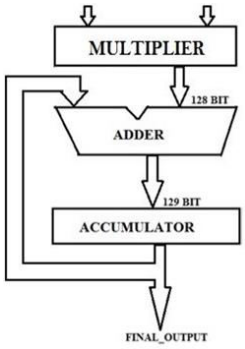


Fig 3.1 MAC – Basic building block diagram

MAC OPERATION

MAC operation is not only the key operations in DSP but also in the multi-media information processing applications and various other applications also. Already mentioned above that, the MAC [12] consists of multiplier, adder and register/accumulator. In this paper, we used a Vedic multiplier. The inputs of MAC are obtained from the corresponding memory location which is connected to the multiplier block. This is helpful for the 64-bit DSP.

From the 64-bit memory location, we can connect the input. The 64-bit input is given to the multiplier, after successful computation of the multiplier it will give the output of 128-bit data; these multiplier outputs of 128-bit data is given as the input to any one of the adders which perform addition. Here we are using the three different types of adders carry save adder, Kogge Stone Adder, and new DKG gate. Finally, it has

proved that the DKG adder is having the highest speed of operation.

The MAC unit function is represented by the following equation:

$$F = \sum P_i Q_i$$

We can get the final output of the adder unit is 129-bit that is the carry is another extra bit. The corresponding output data is connected to the accumulator register. The parallel in Parallel out (PIPO) register technique is used in the accumulator register. Because of this PIPO, the bits are enormous and it produces the corresponding adder output values are generated in parallel, PIPO referred as the input bits are received in parallel and the corresponding out bits are also generated in parallel mode. The accumulator register output is getting from any one of the inputs to a corresponding adder. The Basic building block diagram of the MAC unit is shown in Figure 1.

The Graphical representation of an Urdhwa Tiryakbhyam Sutra is shown in figure2. The basic design of 64×64 Vedic multiplier [3, 4] with CSA using a 32×32 Vedic multiplier is shown in figure 3 and we customized the stage of the adder with a KSA, it is even more sophisticated than the CSA which is shown in figure 4. The Reversible logic concept [5, 6] is a distinctive technique. Here Loss of data is ideally zero. In this, the numbers of inputs and outputs are equals to same.

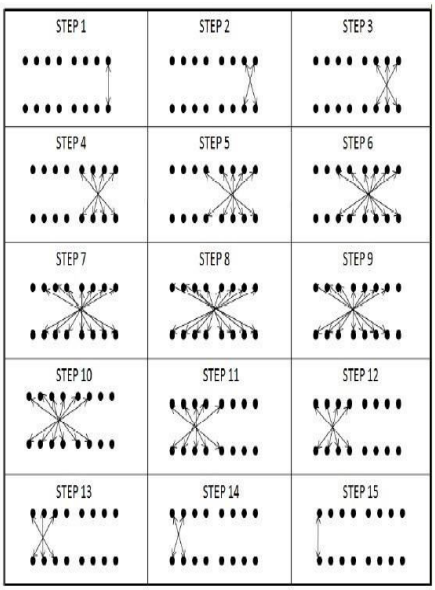


Fig 3.2: Graphical representation of Urdhwa Tiryakbhyam Sutra

4. IMPLEMENTATIOM OF MC BY 64 X 64 VEDIC MULTIPLIER USING DKG GATE

REVERSIBLE GATES

Reversible circuits provide a one-to-one relation between inputs and outputs; therefore, inputs can be recovered from outputs. This interesting feature results in significant power saving in digital circuits. Classical digital gates are not reversible, reversible gates should be designed as basic components to design logical reversible circuits. Well known reversible gates are Feynman, Peres and HNG. The Feynman or controlled not (CNOT) gate is frequently used in reversible circuits, since it can provide exclusive OR (XOR) as well as copy and complement of the input. Since reversible circuits do not take advantage of fan-out, this gate can be used to achieve two copies of the same input by setting the other input of the gate to the zero-logic level. Similarly, by setting the second input of the CNOT to one-logic level, we can achieve the complement of the other input.

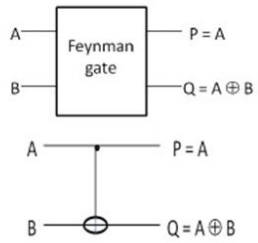
1 Feynman Gate

•Feynman gate is a 2*2 one through reversible gate as shown in figure 4.1.

•The input vector is I(A, B) and the output vector is O(P, Q).

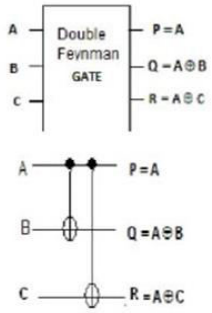
•The outputs are defined by P=A, Q=A xor B. Quantum cost of a Feynman gate is 1.

•Feynman Gate (FG) can be used as a copying gate. Since a fan-out is not allowed in reversible logic, this gate is useful for duplication of the required outputs.



A	B	P	Q
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

2 Double Feynman Gate (F2G)



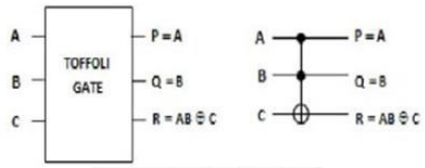
A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	1	1
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	0	0

Fig 4.1 shows a 3*3 Double Feynman gate.

•The input vector is I (A, B, C) and the output vector is O (P, Q, R). The outputs are defined by P = A, Q=A x or B, R=A x or C.

•Quantum cost of double Feynman gate is 2.

3 Toffoli Gate:



A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

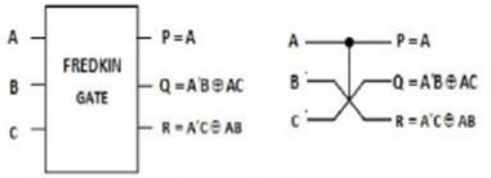
Fig 4.2 : shows a 3*3 Toffoli gate.

•The input vector is I (A, B, C) and the output vector is O (P,Q,R).

•The outputs are defined by P=A, Q=B, R=AB x or C.

•Quantum cost of a Toffoli gate is 5.

4 Fredkin Gate



A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	1	1

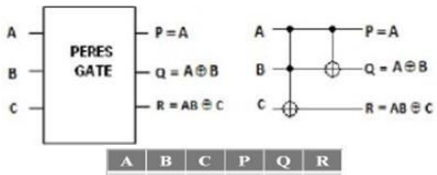
Fig 4.3: shows a 3*3 Fredkin gate.

•The input vector is I (A, B, C) and the output vector is O (P, Q, R).

•The output is defined by P=A, Q=A′B x or AC and R=A′C x or AB.

•Quantumcost of a Fredkin gate is 5.

5 Peres Gate



A	B	C	P	Q	R
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

Fig 4.4:shows a 3*3 Peres gate.

•The input vector is I(A, B, C) and the output vector is O (P, Q, R).

•The output is defined by P = A, Q = A xor B and R=AB xor C.

•Quantum cost of a Peres gate is 4.

•In the proposed design Peres gate is used because of its lowest quantum cost.

6 TSG gate

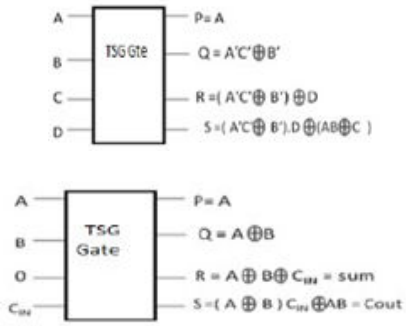


Fig 4.5 shows a 4*4 TSG gate.

•The input vector is I (A,B, C, D) and the output vector is O (P, Q, R, S).

•The output is defined by P = A, Q = A'C' xor B', R = (A'C' xor B')xor D and

•S =(A'C' xor B').D xor (A.B xor C) .

•Quantum cost of a Peres gate is 4.

•In the proposed design Peres gate is used because of its lowest quantum cost.

•It can be verified that the input pattern corresponding to a particular output pattern can be uniquely determined.

•The proposed TSG gate is capable of implementing all Boolean functions and can also work singly as a reversible Full adder.

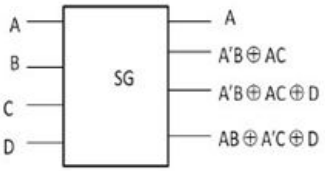
7 Sayem gate

•SG is a 1 trough 4x4 reversible gate.

•The input and output vector of this gate are, Iv = (A, B, C, D) and

•Ov = (A,A'Bxor AC, A'B xor AC xor D, AB xor A'C xor D).

•The block diagram of this gate is shown in Figure



8.HNG Gate

A reversible logic gate has the same number of inputs and outputs with one-to-one correspondence between the input and the output. Hybrid new gate (HNG) is a universal reversible gate and realizes all Boolean functions.

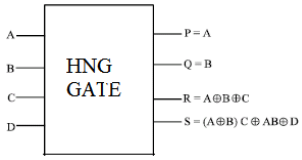


Fig 4.6: HNG Gate

9.DKG gate

The design of DKG adder is completely based on the property of reversibility computing. It is havingthe four inputs and four outputs, out of four two inputs are garbage inputs. The DKG gate is satisfied both the conditions of full-adder and full- subtractor, which is selected by using the input 'A'. If A=0, it will act as a reversible Full adder circuit and if A=1, it will act as a reversible Full subtractor. The logic diagram for the reversible DKG gate is shown in figure 5.

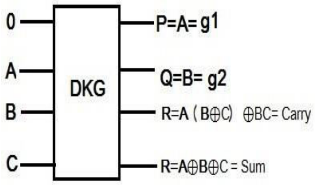


Fig 4.7 :DKG logic Gate

4 Bit RCA based DKG Adder

Multiple DKG adder circuits can be cascaded in parallel to add an N-bit number. For an N- bit parallel adder, there must be N number of DKG circuits. A ripple carry adder is a logic circuit in which the carry-out of each DKG is the carry in of the succeeding next most significant DKG. It is called a ripple carry adder because each carry bit gets rippled into the next stage. In a ripple carry adder the sum and carry out bits of any half adder stage is not valid until the carry in of that stage occurs. Propagation delays inside the logic circuitry are the reason behind this. Propagation delay is time elapsed between the application of an input and occurrence of the corresponding output. Similarly, the carry propagation delay is the time elapsed between the application of the carry in signal and the occurrence of the carry out (Cout) signal. Circuit diagram of a 4-bit ripple carry adder is shown below. Sum out S0 and carry out Cout of the DKG1 is valid only after the propagation delay of DKG1. In the same way, Sum out S3 of the DKG4 is valid only after the joint propagation delays of DKG1 to DKG4. In simple words, the final result of the ripple carry adder is valid only after the joint propagation delays of all DKG circuits inside it.

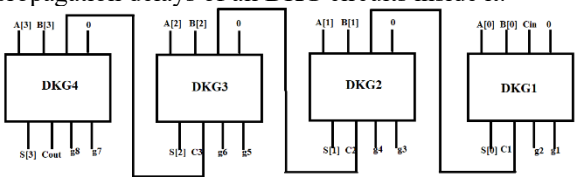


Fig 4.8:bit RCA based DKG adder

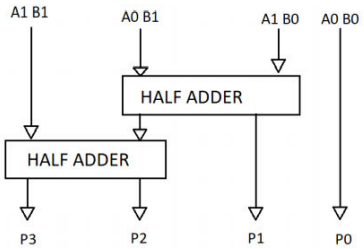


Fig 4.9: 2x2Vedic multiplier

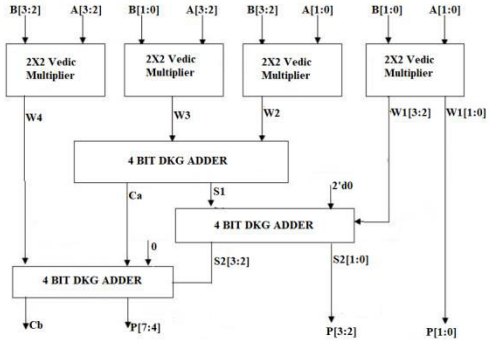


Fig4.10 : 4 Bit Vedic multiplier

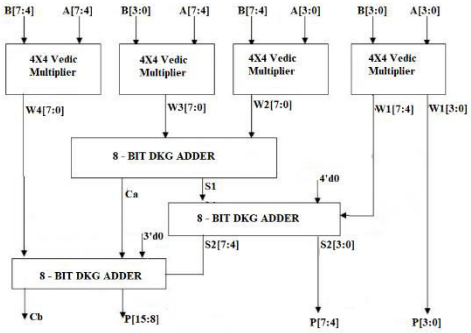


Fig 4.11: 8 Bit Vedic multiplier

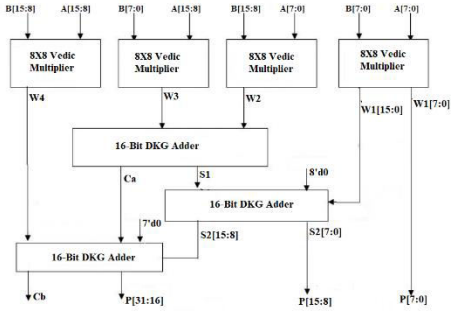


Fig 4.12: 16 Bit Vedic multiplier

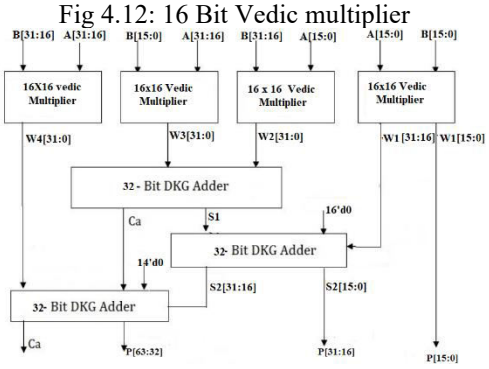


Fig 4.13: 32 Bit Vedic multiplier

Figure 6 shows the 64-bit Vedic multiplier using DKG gate. With Reference to the above architecture, the realization of Vedic multiplier is very easy to be intended with the hierarchical method. To design of a 64-bit Vedic multiplier [8], it requires the 32-bit design of a lower-level multiplier. Further, this design requires the 16, 8, 4-bit multiplier and 2- bit multiplier. Thus, it is very simple to expand the multiplier Design to a high level by maintaining its own modularity.

The primary logic for the proposed operation of a 64-bit size is simply separated into the two half of 32-bit each. The lower halves of the numbers are the inputs of the 1st stage, by simply swapping the two halves of the data in 2rd and 3rd stage and finally, the upper half is for the 4th and final stage.

The intermediate adder stage is also required for this design. Firstly, the adder stage is designed by using the new reversible DKG logic gate. The A and B are the two inputs which are applied in a transverse mode; it is a 64 bit of input size and the maximum of 128 bit of the final result. It can add the 64-bit inputs and the sum will be saved to their corresponding registers, if any carry is generated it can be added to the next consecutive stage.

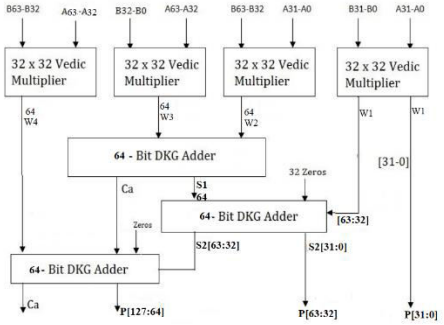


Fig 4.14: 64-bit Vedic multiplier using DKG adder

To equate the total number of bits in the addition process, we can append the zeros as one of the inputs to the adder stage, then only these addition processes able to be performed without having an error. For inspecting the design of the 64-bit multiplier we have to rectify three DKG stages. At the last stage of adder, the carry is generated, whereas the final stage, the sum output is taken from the adder. From the first stage of the multiplier, we can get the LSB sum bits directly. The design and operation of a 32-bit and 64-bit multiplier are almost the same with the exception of the 32-bit design having the 32 input bits and for 64-bit having the 64 input bits

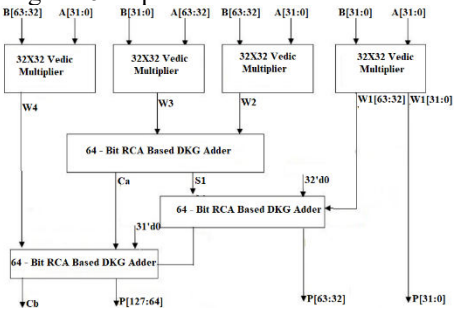


Fig 4.13 64 bit vedic multiplier

4-BITRCA based DKG ADDER

The 4bit DKG adder was designed by carry select adder.The carry-select adder generally consists of two ripple carry adders and a multiplexer. Adding two n-bit numbers with a carry-select adder is done with two

adders (therefore two ripple carry adders), in order to perform the calculation twice, one time with the assumption of the carry-in being zero and the other assuming it will be one. After the two results are calculated, the correct sum, as well as the correct carry-out, is then selected with the multiplexer once the correct carry-in is known.

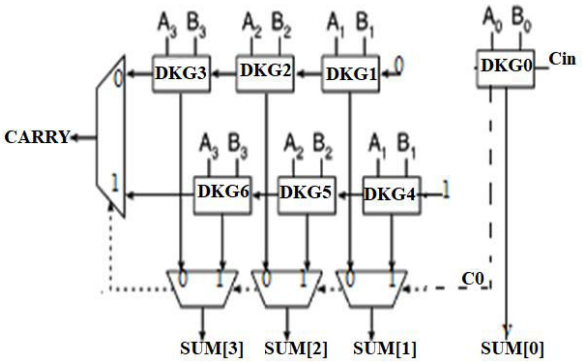


Fig 4.14 :4-BIT DKG ADDER

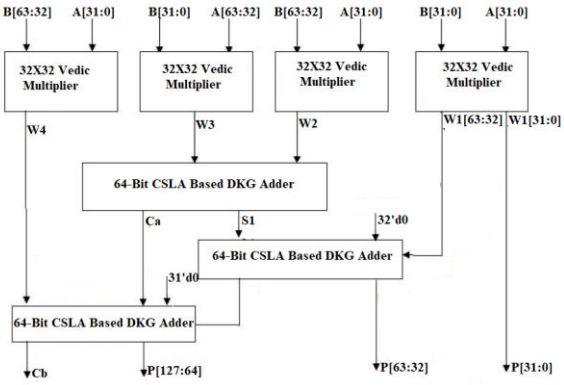


Fig 4.15 : proposed Vedic Multiplier

5. RESULTS

RTL SCHEMATIC:-

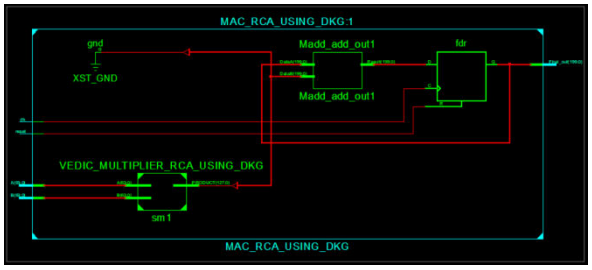


Fig 5.4: RTL Schematic of MAC using RCA based DKG adder.

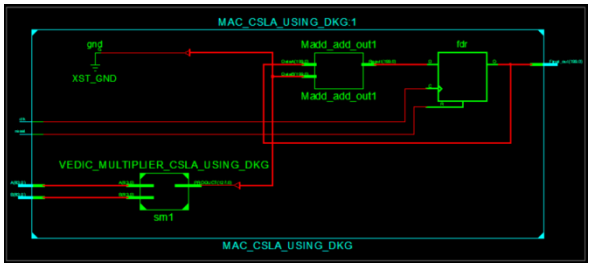


Fig 5.5: RTL Schematic of MAC using CSLA based DKG adder.

TECHNOLOGY SCHEMATIC:-

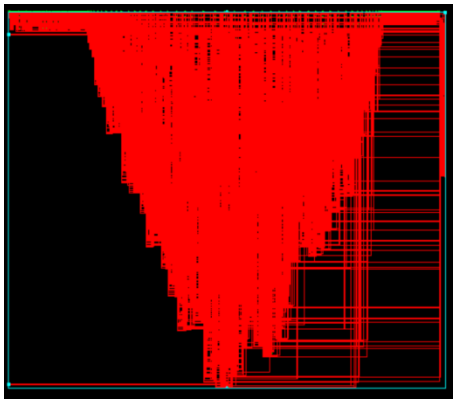


Fig :View Technology Schematic of MAC using RCA based DKG adder

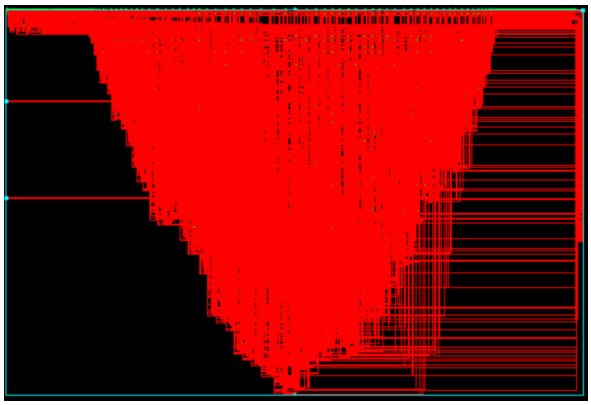


Fig :View Technology Schematic of MAC using Kogge Stone Adder

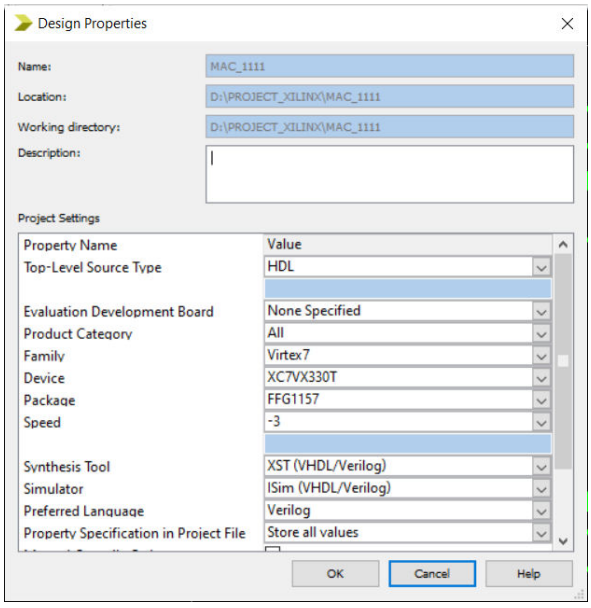


Fig :family selected for synthesis

SIMULATION:-

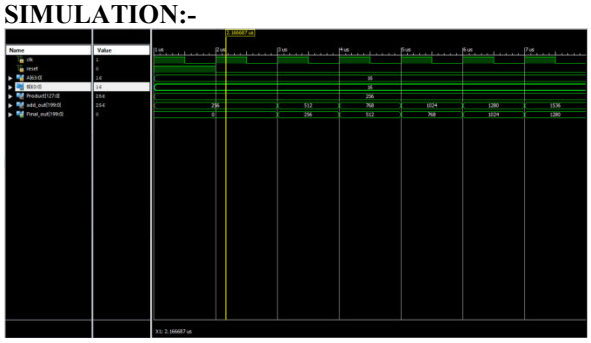


Fig :Simulated Waveforms of MAC using RCA based DKG adder

Fig :Simulated Waveforms of MAC using CSLA based DKG adder

PARAMETERS:-

Parameter	MAC using RCA based DKG Adder	MAC using CSLA based DKG Adder
No of LUTs	11507	10830
Delay (ns)	56.232	37.814

Table :parameter comparison table

GRAPH:-

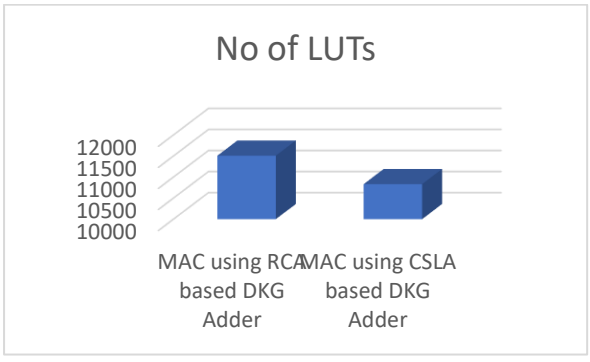


Fig :No of LUTs comparison Bar graph

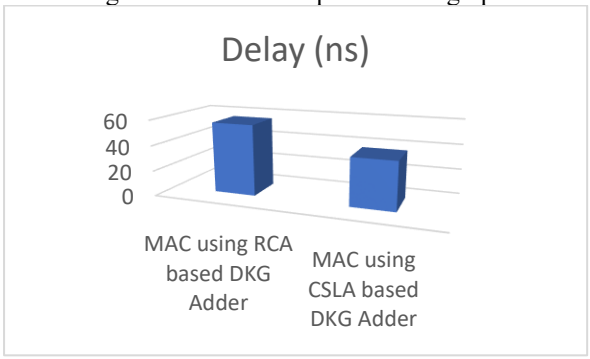


Fig :Delay comparison Bar graph

CONCLUSIONS & FUTURE SCOPE

The results are obtained from the proposed DKG adder gate design using a Vedic multiplier with reversible computing are relatively good. The proposed 64- bit MAC unit is successfully designed with Vedic multiplier using RCA and CSLA using DKG reversible logic. it has been proved that the design is optimized in terms of total delay. We are successfully designed all the 64-bit MAC architecture of fundamental analyzed for all the existing blocks. Hence, we can prove that the UrdhavaTriyagbhayam sutra with 64-bit MAC Unit and the reversible logic concept is the finest in terms of total delay aspect is shown in table 1. The overall Simulation and synthesis process is successfully carried out with Xilinx ISE 14.7 The design parameters of any architecture completely depend on the basic building blocks. For our proposed design MAC, the basic building blocks are a multiplier and the adder. In future, these basic building block designs are designed highly optimized than our proposed design obviously, it leads to reduce the total delay in the MAC architecture

REFERENCES

[1]R. Anitha1, Neha Deshmukh, Sarat Kumar Sahoo, S. Prabhakar Karthikeyan," A 32 BIT MAC Unit Design Using Vedic Multiplier and Reversible Logic Gate" International Conference on Circuit, Power and Computing Technologies [ICCPCT].

[2]Ramalatha, M.Dayalan, K D Dharani, P Priya, and S Deoborah, High Speed Energy Efficient ALU design using Vedic multiplication techniques, International

[3]Sree Nivas A and Kayalvizhi N. Article: Implementation of Power Efficient Vedic Multiplier. International Journal of Computer Applications 43(16):21-24, April 2012. Published by Foundation of Computer Science, New York, USA

[4]VaijyanathKunchigi, Linganagouda Kulkarni, Subhash Kulkarni, High Speed and Area Efficient Vedic Multiplier, International Conference on Devices, Circuitsand Systems (ICDCS), 2012.

[5]D.P.Vasudevan, P.K.Lala, J.Di and J.P.Parkerson, "Reversible logic design with online testability", IEEE Trans. on Instrumentation and Measurement, vol.55, no.2, pp.406-414, April 2006.

[6]Prabir Saha, Arindam Banerjee, Partha Bhattacharyya, Anup Dandapat, High Speed ASIC Design of Complex Multiplier

[7]Raghava Garipelly, P.Madhu Kiran, A.Santhosh Kumar A Review on Reversible Logic Gates and their Implementation International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 3, Issue 3, March 2013.

[8]www.vedicmaths.org/

[9]Asmita Haveliya, A Novel Design for High-Speed Multiplier for Digital Signal Processing Applications

(Ancient Indian Vedic mathematics approach), International Journal of Technology and Engineering System (IJTES), Vol.2, No.1, Jan -March, 2011.
[10]Aniruddha Kanhe, Shishir Kumar Das and Ankit Kumar Singh, Design and Implementation of Low Power Multiplier Using Vedic Multiplication Technique,
(IJCSC) International Journal of Computer Science and Communication Vol. 3, No. 1, January- June 2012, pp. 131-132 International Journal of Scientific and Research Publications, Volume 3, Issue 2, February 2013 ISSN 2250-315.
[11]A. Abdelgawad, Magdy Bayoumi ," High Speed and Area- Efficient Multiply Accumulate (MAC) Unit for Digital Signal Processing Applications", IEEE Int. Symp. Circuits Syst. (2007) 3199–3202.
[12]Fatemeh Kashfi, S. Mehdi Fakhraie, Saeed Safari, "Designing an ultra-high-speedmultiply-accumulate structure", Microelectronics Journal 39 (2008) 1476–1484.