IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-11, Nov 2018 E-ISSN: 2347-2693

An Effective Interlaced Separation Vedic Multiplier in FPGA Platform

M. Isaivani[1]*, V. Malathi[2], E. Sakthivel[3]

[1]Dept.of Electrical and Electronics Engineering, Anna University Regional Campus Madurai, Madurai, Tamil Nadu, India
[2]Dept. of Electrical and Electronics Engineering, Anna University Regional Campus Madurai, Madurai, Tamil Nadu, India
[3]Dept. of Electrical and Electronics Engineering, PSR Engineering College, Sivakasi, Tamil Nadu, India

Corresponding Author: isainec06@gmail.com, Tel.: +91-77084-49675

Available online at: www.ijcseonline.org

Accepted: 21/Nov/2018, Published: 30/Nov/2018

***Abstract*—**A Multiplier is one of the essential components in the embedded system and digital signal processing (DSP) applications like digital filtering, digital communications, digital image processing, and spectral analysis etc. Parallel multiplication concept was implemented earlier to achieve higher speed. But in order to design an effective multiplier, speed is not the only consideration, area and power also must be taken into account while designing a circuit. To achieve this, a new technique has been introduced here for fast multiplication of two numbers. Inputs are separated into partitions where one number is again separated by two and zeros are interlaced in each alternate partition. Then it is followed by component multipliers and adders to get the final product. Based on the application requirement, the component adders and component multipliers can be selected in order to balance area and speed. The proposed system is synthesized using Xilinx tool. Experimental results show that the proposed interlaced separation Vedic multiplier is faster and has greater power efficiency in Field Programmable Gate Array (FPGA) implementation.

***Keywords*—**Vedic Multiplier, Ripple Carry Adder, FPGA, VLSI

I. INTRODUCTION

In embedded systems, DSP and everyday computing, the efficient multiplication of integers or fixed-point numbers is the operation frequently being performed. Generally, Multiplications are so expensive and also slows down the overall operation. On account of rapid developments in computer and signal processing applications, the requirements such as lower power, lower area, and higher speed are also increasing for the purpose of reaching out the required performance successfully.

In image processing and real-time signal processing applications, arithmetic operations should have a higher throughput to make the design perfect. More specifically multiplication arithmetic operation is considered as one of the most important in such applications. It is essential to notice that the propagation of carries in the multiplier circuits adds large critical paths and long delays when computing the product of two numbers. So many researches are being processed in multiplier circuits for the last few decades. In digital systems, power consumption reduction mainly deals with optimization at various levels of the design. It is a fact in any digital circuit design, digital multipliers are the most useful components since they are more reliable, very fast and they are the efficient components which are used to implement any operation.

In this paper, Section I contains the introduction, Section II contains the related work of various multiplier circuits, Section III contains overview of Interlaced Multiplier concept, Section IV contains the architecture, methodology and essential steps in the proposed scheme, section V shows the simulation results, Section VI describes performance evaluation and Section VII concludes research work with future directions.

II. RELATED WORK

The conventional interlaced partition multiplier is implemented [1] in CMOS. Wallace suggested a method for faster multiplication based on parallel counters [2]. By Wallace's scheme, the parallel multiplication of two binary number is done in the following manner a) first all partial products are formed in parallel with AND gate arrays. b) Then (3,2) counters and (2,2) counters are strategically applied for the reduction of partial products. c) Those two numbers are added by using a fast carry propagate adder to generate the final product. Dadda [3,8] modified Wallace's scheme by implementing counter placement technique which minimizes the number counters which in turn reduces the cost of a larger carry propagate adder. In both Dadda and Wallace's methods, the total delays are directly proportional to the logarithm of the word-length of the operand and they are faster than the array multipliers. A well effective algorithm was developed for solving a particular class of

initial value recurrence problems on parallel computing systems, [4]. When the delay is focused to be the main parameter, designers implement well-known logarithmic multipliers such as Wallace, Dadda or TDM. This kind of multipliers follow reduction tree method [5], where compression of partial–product bits are being processed, that results in the worst–case delay relies logarithmically on the factor word length.

A modified design presented in [6] effectively minimizes the number of half adders in Wallace multipliers. Reference[7] presents reduced area multipliers that result in reduced components and reduced interconnect overhead than Wallace and Dadda multipliers. Braun's carry save method is the another high-speed multiplier design. Braun's methodology is compared by Habbibi and Wintz with Dadda and Wallace multipliers on the account of speed, complexity, and cost. Habibi analyzed that both Wallace's and Dadda's schemes provide the fastest multiplication yet it creates the cost and complexity issues which have to be taken into further attention [8 - 9].

Another modification of Wallace's and Dadda's methodology is reduced area (RA) multiplier [10]. Here the reduction scheme differs from the previous schemes in which more number of (3,2) counters are utilized as early as possible and also (2,2) counters are utilized to minimize the length of the word in the carry propagate adder. RA multipliers need less number of components and so reduced interconnect overhead compared to Wallace or Dadda multipliers for the same size of implementation.

A new VLSI architecture based on the implementation of efficient combinational and pipelined circuits for modular arithmetic multiplication is introduced in [11]. Booth multiplication is another powerful technique that recodes the numbers to be multiplied for obtaining smaller, faster multiplication design. Reference[12] explains the study of various multipliers such as array multiplier, constant coefficient multipliers, and Vedic multipliers. Vedic multiplication method in the VLSI environment is explained in [13 - 14]. The performance of Wallace multipliers and Dadda multipliers were compared in [15]. Reference[16] clearly showed that not only speed is sacrificed when an array multiplier is being used instead of a logarithmic multiplier, yet power too. It proposed a new design for the reduction tree which is particularly based on the partial product compression similar to the Dadda method. Mutiply and Accumulate computation is done by using Distributed Arithmetic technique which involves lookup tables [17]. A full adder circuit is implemented by using CMOS with transmission gates and the parameters such as area, power are analyzed in [18].

This paper presents a novel multiplier implementation which avoids long carry chains and lets multiplication of two numbers in the time taken to multiply two numbers of a fraction of the number of bits as the original numbers, plus several additions. It is ensured the proposed implementation outperforms standard multiplier implementation while incurring only a modest increase in area.

III. INTERLACED MULTIPLIER OVERVIEW

The Interlaced partition multiplier is implemented in CMOS [1], in which two numbers of a large number of bits can be multiplied faster by partitioning each number into numbers of fewer bits. If one of these partitioned numbers is again split into two numbers which are interlaced with zeros on every other partition, the product of the interlaced number and the other number can be calculated without carries, simply by computing all products of the smaller partitions, concatenating these products into the summands which are produced from multiplication and then performing the summation of summands directly. Moreover, these summands that are technically twice the number of bits as the inputs are seen to contain half the possible number of non-zero bits. Thus the resultant summands are added by pairs with adders which are less in wide. The sum of all the summands is the final product that is calculated in the time needed to multiply the shorter partitions plus the time needed to add the summands.

2.1. Partitioning inputs

Let A and B be the numbers of c bits such that the product AB consists of 2c bits. Let each A and B be partitioned into t numbers, A_0, A_1 ... A_{t-1} and B_0, B_1,……B_{t-1} of u bits each, where

$$u < c \tag{1}$$

$$t = c/u \tag{2}$$

A and B can be expressed in binary notation using these t numbers as

$$A = \sum_{k=0}^{t-1} A_k 2^{ku} \tag{3}$$

$$B = \sum_{k=0}^{t-1} B_k 2^{ku} \tag{4}$$

2.2. Interlacing one input

Let A be partition into two numbers, A_{EV} and A_{OD} each of c bits, such that

$$A_{EV} = \sum_{k=0}^{t/2-1} A_{2k} 2^{2ku} \tag{5}$$

$$A_{OD} = \sum_{k=0}^{t/2-1} A_{2k+1} 2^{(2k+1)u} \tag{6}$$

Thus A_{EV} consists only the even partitions of A while its odd partitions are replaced by zeros. In the same way, A_{OD}

contains the odd partitions of A with even partitions replaced by zeros.

$$AB = A_{EV}B + A_{OD}B \quad (7)$$

The products of $A_{EV}B$ and $A_{OD}B$ are being calculated and added to get the final product as AB.

2.3. Component multipliers and adders

In the exciting system, traditional array multiplier and Dadda multiplier are used as component multipliers to carry out 8-bit multiplication process. Here the segments of the multiplier and the multiplicand of both 8 bits each are taken for the operation for m x n bit array multiplier, it requires (mxn) AND gates, (m-1)*N adders in which n half adders and (m-2)*N full adders. Both Dadda multiplier and Wallace multiplier are similar hardware circuits, but Dadda method is faster and requires fewer gates than Wallace multiplier for all operand sizes.

Ripple carry adder is the adder circuit which is used to add the products $A_{EV}B$ and $A_{OD}B$ in the overall multiplication process. This circuit is using a half adder and multiple full adders to add N-bit numbers where each full adder's carry output (C_{out}) is given as the third input (C_{in}) of the adjacent full adder (ie) each carry bit ripples to the following full adder. The advantages are its lower power consumption and compact layout which gives smaller chip area. The most serious disadvantage is that the delay increases linearly with the bit length, which is shown as

$$t = (n-1)t_b + t_y \quad (8)$$

where t_b is the delay of the carry stage of the full adder and t_y is the delay to calculate the sum of the last stage, n is the number of bits. The overall performance of the existing multiplier circuit can be enhanced further in the proposed system.

IV. PROPOSED SCHEME

The following figure 1 shows the architecture of the proposed 32-bit multiplier which is implemented in FPGA platform which was done in CMOS earlier. The following step-by-step procedure clearly explains its methodology in detail.

Step I: (Separation)

Let the two inputs A and B are partitioned as A(7:0), A(15:8), A(23:16), A(31:24), B(7:0), B(15:8), B(23:16), B(31:24) which are taken as the segments A1, A2, A3, A4 for A segments and B1, B2, B3, B4 for B segments respectively.

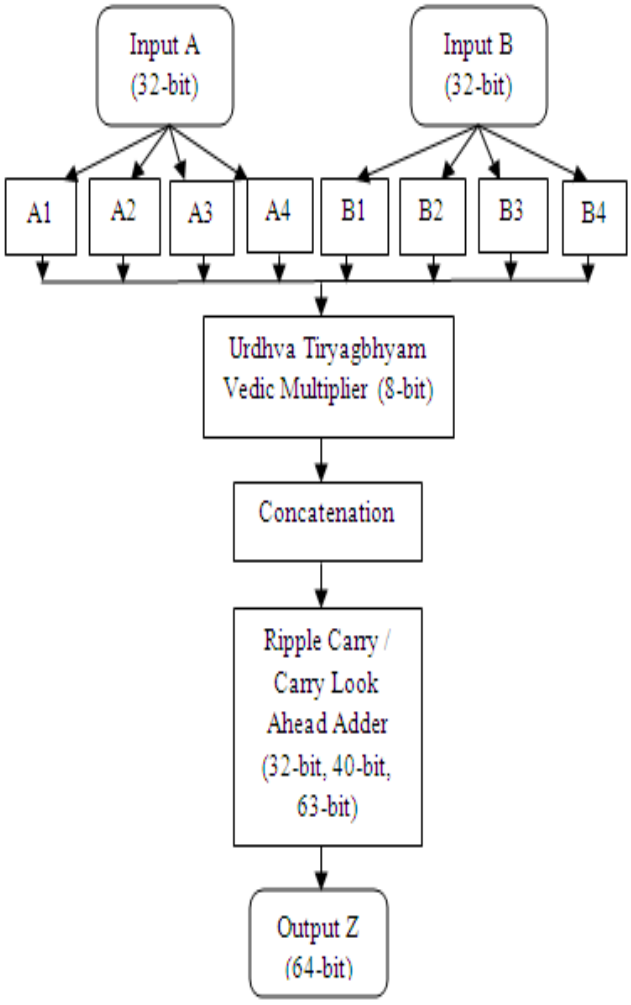

Figure 1. Architecture of 32-bit Interlaced separation Vedic Multiplier (u = 8, t = 4)

Step II: (Multiplication)

In the proposed system, Vedic multiplier performs the multiplication operation of the segments of 8 bits each efficiently. The proposed component 8 bit multiplier employs the new scheme of Vedic mathematics called as Urdhva–Tiryagbhyam sutra which is used to generate partial products. The prominent "Urdhva Tiryagbhyam" multiplication method can be applied to all types of number system such as decimal, binary, hex and octal. The Sanskrit term "Urdhva" denotes "Vertically" and "Tiryagbhyam" specifies "Crosswise". From various multiplication methodologies, this sutra is being implemented for all component multipliers because it is applicable to all cases of algorithms for NxN bit numbers and most importantly the

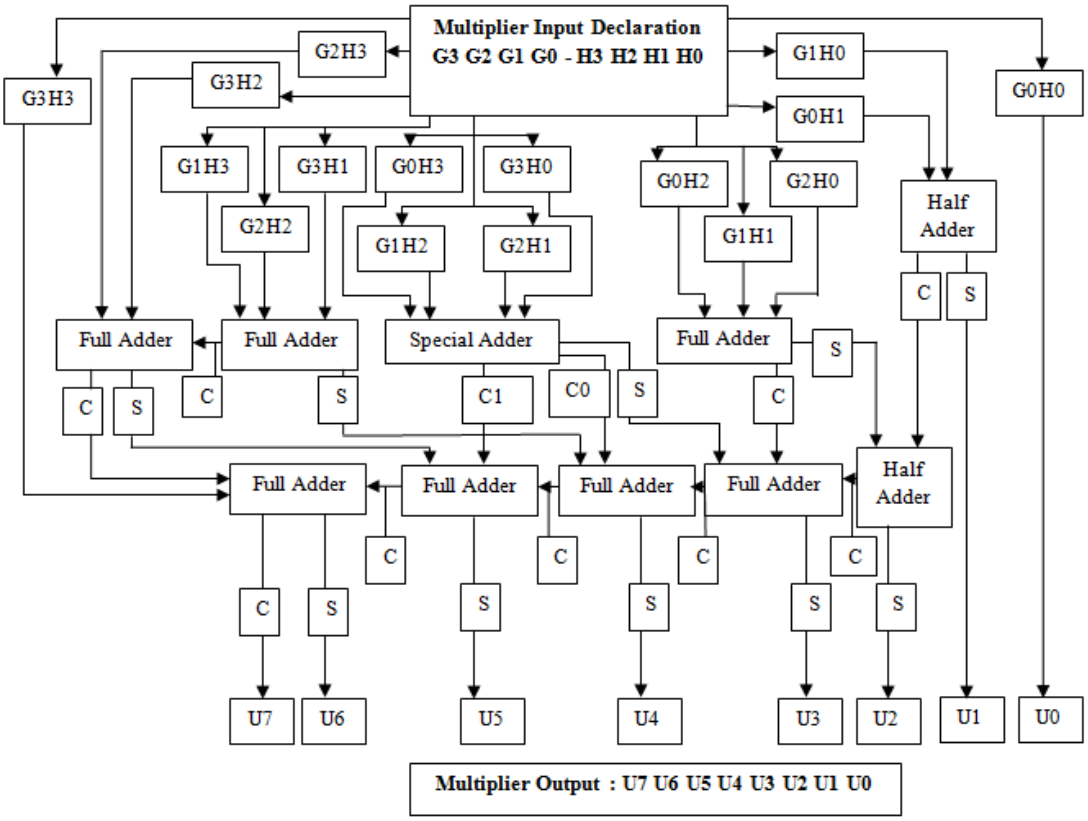

Figure 2. 4-bit Vedic Multiplier Architecture

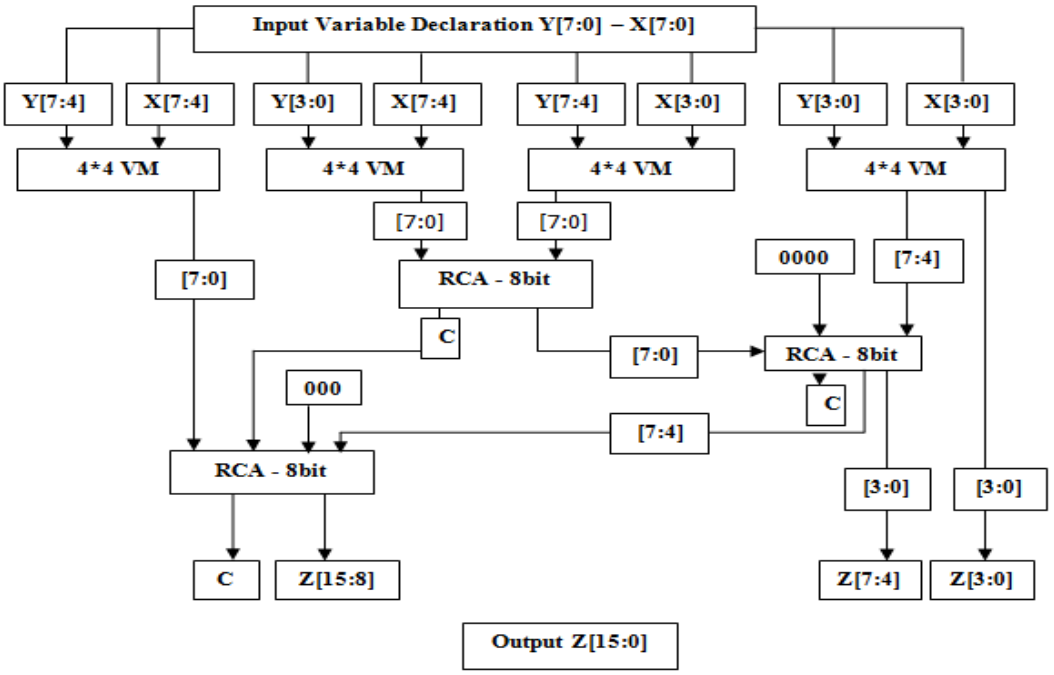

Figure 3. 8-bit Vedic Multiplier Architecture

minimum delay is obtained. Figure 2 shows the design of 4x4 Vedic multiplier that uses 7- full adders, 2-half adders and a single 4-bit special adder. The four-bit special adder utilizes two half adders and a full adder which in turn produces three output – least significant bit and most significant bit of sum output and the carry output.

Figure 3 depicts 8x8 Vedic multiplier by using ripple carry adder and 4x4 VM (Vedic Multiplier). The popular Vedic multipliers are faster than the existing conventional multipliers and most importantly, the design density is condensed and the modularity is augmented even for more number of input bits. Moreover, they minimize power, delay, and hardware requirements for multiplication of more numbers.

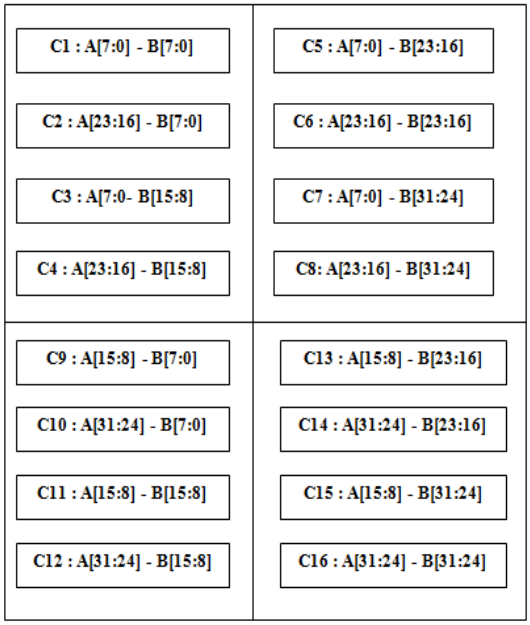

Figure 4. Partial Products Formation

In the proposed work, 16 numbers of 8-bit Vedic multipliers are used to produce the partial products of C1, C2, C3, C4, C5, C6, C7, C8, C9 C10, C11, C12, C13, C14, C15, C16 respectively which is based on splitting and interlacing technique. It is shown in figure 4.

Step III: (Concatenation)
In figure 5, eight concatenation operations are being performed for the above 16 partial product segments which are obtained from Urdhva Tiryagbhyam Vedic 8 bit multipliers

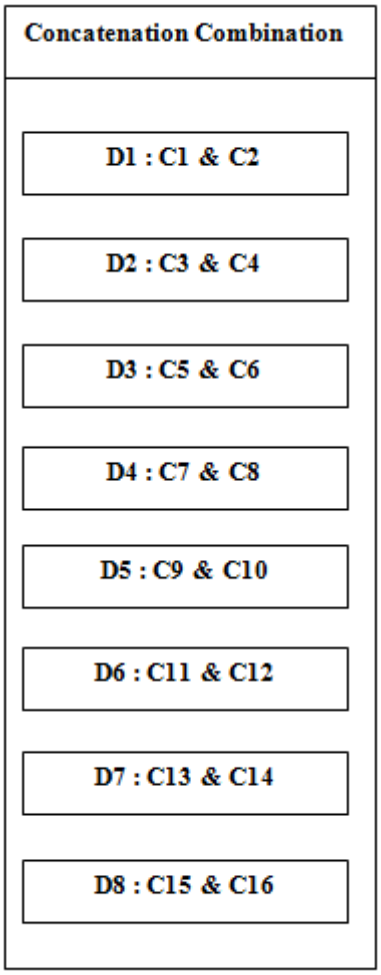

Figure 5. Concatenation Operation

Step IV: (Shifting)
The signals are being shifted left by left shifters as shown in figure 6. For binary numbers, the left shift operation shifts all the bits in its operand where each bit is simply moved as per the specified shifts and vacant bit positions are occupied by shifting n-bits left in the binary number. It equals the operation of multiplication by 2n. In the proposed work, after the concatenation operation, the bits are being left shifted according to the architecture to form the next intermediate level of result.

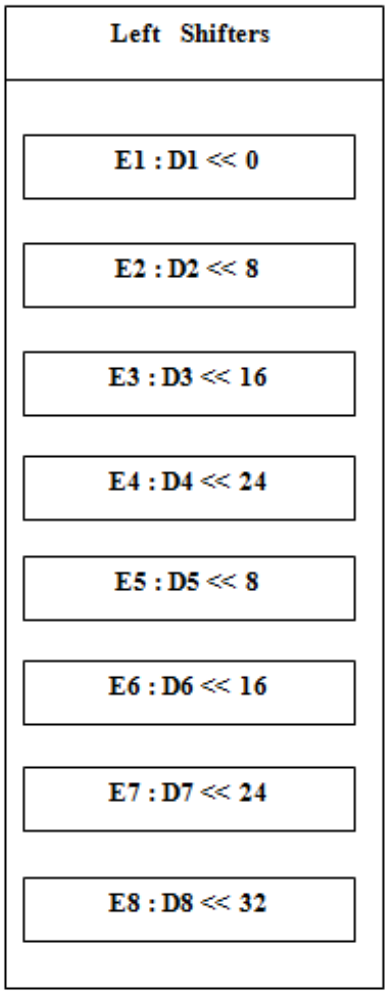

Figure 6. Shifting Operation

Step V: (Addition)
The following figure 7 depicts the addition process in the proposed method. The proposed system involves 2 types of adders to get the final product - 1. Ripple carry adder 2. Carry look Ahead Adder. When using ripple carry adder lower power, lower area are achieved but the speed is somewhat low. For that, carry look ahead adder is performed to improve the speed yet result in slight increase in power and area. The operation of ripple carry adder is already explained in detail in the previous stages.

In order to attain faster speed, the addition operation is being performed by using carry look ahead adder. Various logic design methodologies have been processed to reduce the carry propagation issue. Carry look ahead principle is the widely used technique to solve this issue by estimating the carry signals in advance, based on input signals. It is based on the note that a carry signal will be generated in any of the following two cases (a) When both input bits are 1, or (b) when one of two input bits is 1 and also the carry of the previous stage (carry –in) is 1
The second internal signals Pi and Gi are

$$P_i = L_i + M_i \quad (9)$$

$$G_i = L_i M_i \quad (10)$$

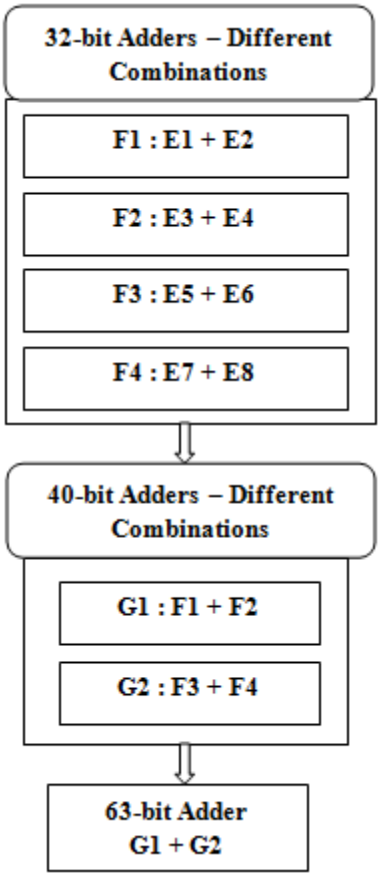

Figure 7. Addition Operation

The output sum and carry are

$$S_i = P_i XORN_i \quad (11)$$

$$N_{i+1} = G_i + P_i N_i \quad (12)$$

where Li, Mi are 2 inputs, Ni is the carry signal, Gi is the carry generate signal because a carry (Ni+1) is generated when Gi is 1, regardless of the carry signal (Ni). Pi is the carry propagate signal since when Pi is assigned as 1, the input carry is promoted to the output carry, Ni+1 = Ni (whenever Pi = 1, Gi = 0).

Thus computing the values of Pi & Gi only rely on the input operand bits Li & Mi. So the resultant signals settle in their steady state after the propagation via the corresponding

gates. Thus it is inferred that estimated values of all Pi's are valid one XOR logic gate delay after L and M are made valid and calculated values of all Gi's are valid one AND logic gate delay L & M are made valid.
The i^{th} output carry signal can be formulated as

$$N_i = F_i(P's, G's, N_0) \quad (13)$$

Here each carry signal can be expressed as direct sum of product (SOP) function of N_o rather than its prior carry signal.

The above step by step processes clearly explains the detailed operation of proposed interlaced separation Vedic multiplier which is implemented in Field Programmable Gate Array (FPGA). The recursive use of the proposed multiplier would be even more beneficial at higher number of bits. It is very important to note that the architecture of the proposed multiplier clearly provides the choice of using different multipliers as for component multipliers and adders for addition operation which must be left to the application specific cost function. Here 8-bit Vedic multipliers are used for component multipliers because of its inherent advantages of lower power, less delay, and fewer hardware requirements.

V. SIMULATION RESULTS AND PERFORMANCE EVALUATION

There are various techniques for logic implementation at circuit level that improves power dissipation, area, delay parameters, frequency constraints in VLSI (Very Large Scale Integration) design. The proposed interlaced separation Vedic multiplier is being implemented by utilizing Very High Speed Integrated Circuit Hardware Description Language (VHDL). Also, different combinations of various multipliers as component multipliers and adders are being implemented with VHDL Language for comparison purpose. The synthesis is performed using Xilinx Synthesis Tool (XST) Xilinx ISE 12.1. The following figure 8 and figure 9 show the simulation output, for example,

A="1110" = 14
B = "1111" = 15
Result = 11010010 = 210
Result = 11010010 = 210

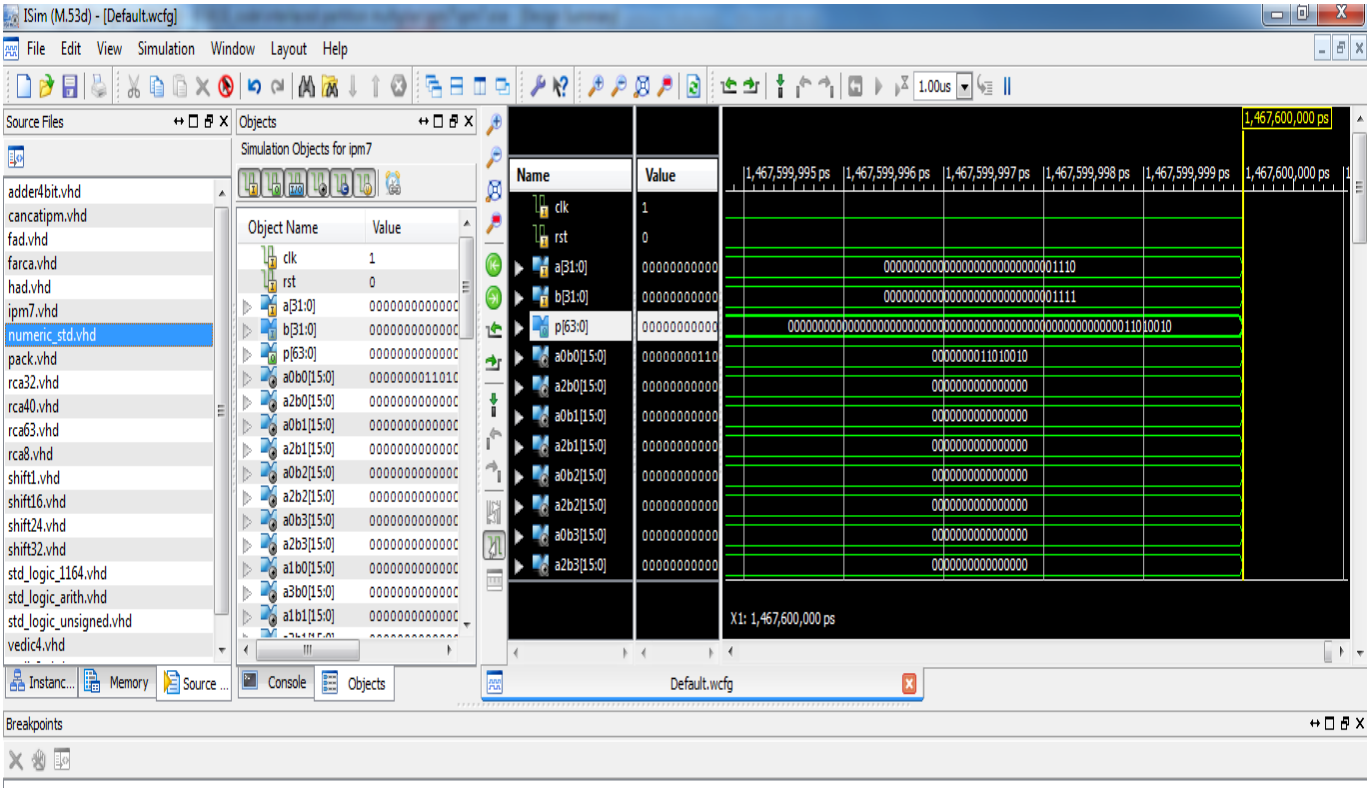

Figure 8. Interlaced Separation Vedic Multiplier involves ripple carry adder simulation output

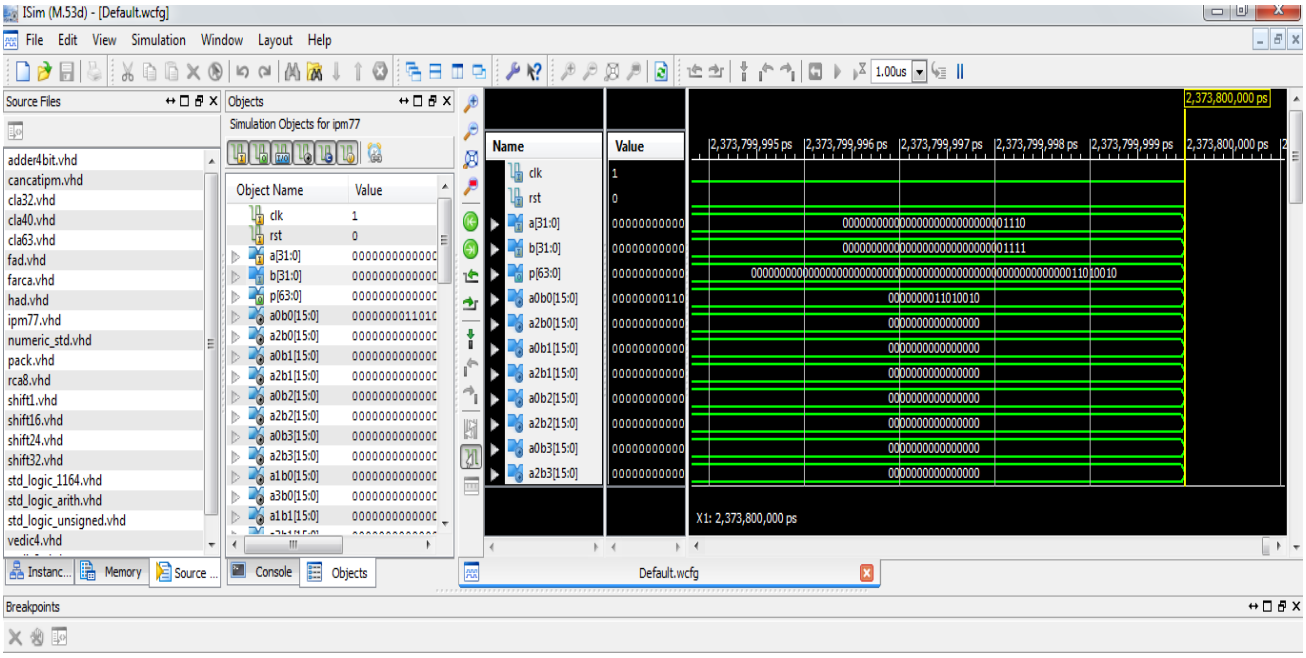

Figure 9. Interlaced Separation Vedic Multiplie

Figure 9. Interlaced Separation Vedic Multiplier involves carry look ahead adder simulation output.

Table 1 shows the performance variation in terms of FPGA level progress and comparison is made for 10 different architectures. All are implemented in Xilinx Virtex – xc6vcx75t device. All the multipliers are being processed to operate for 32 bits. For evaluation purpose, the conventional work [1] is implemented in FPGA platform here, which was implemented in CMOS earlier. The first two rows in Table I show the corresponding result. The comparison is made for different combinations of component multipliers and adders in the system.

Figure 10 (a-e) shows the performance comparison of various FPGA architectures. The parameters such as power, area in terms of look up tables and registers, delay, frequency are analyzed and shown in the graphical representation.

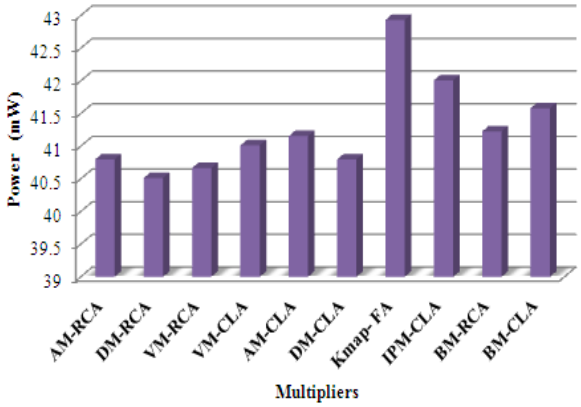

Figure 10(a). Power Consumption

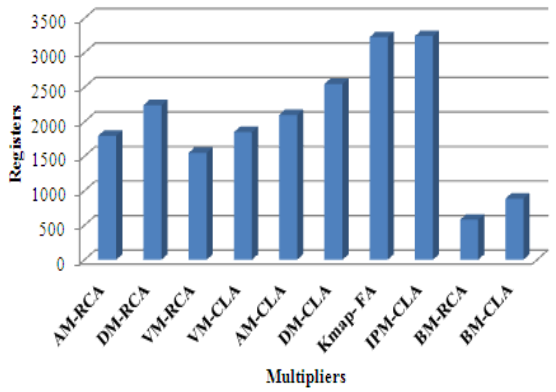

Figure 10(b). Area Requirement (Registers)

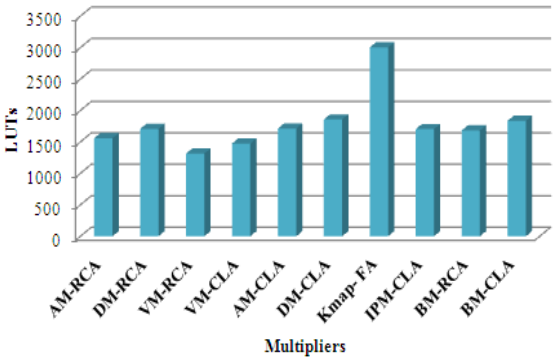

Figure 10(c). Area Requirement (Look Up Tables)

TABLE 1. PERFORMANCE ANALYSIS OF VARIOUS FPGA ARCHITECTURES

S.No	Description	Power (mW)	Area		Delay (ns)	Frequency (MHz)
1	Array Multiplier with Ripple Carry Adder (AM-RCA)	40.7976	Reg	1796/93120	1.177	849.618
			Lut	1559/46560		
			Logic	1543/46560		
			Memory	16/16720		
			Srl	16		
2	Dadda Multiplier with Ripple Carry Adder (DM-RCA)	40.5128	Reg	2239/93120	1.269	787.836
			Lut	1707/46560		
			Logic	1507/46560		
			Memory	200/16720		
			Srl	200		
3	Vedic Multiplier with Ripple Carry Adder (VM-RCA)	40.6652	Reg	1550/93120	1.177	849.618
			Lut	1314/46560		
			Logic	1300/46560		
			Memory	14/16720		
			Srl	14		
4	Vedic Multiplier with Carry Look Ahead Adder (VM-CLA)	41.0112	Reg	1849/93120	0.770	1297.859
			Lut	1473/46560		
			Logic	1473/46560		
5	Array Multiplier with Carry Look Ahead Adder (AM-CLA)	41.1536	Reg	2099/93120	0.770	1297.859
			Lut	1713/46560		
			Logic	1713/46560		
6	Dadda Multiplier with Carry Look Ahead Adder (DM-CLA)	40.7976	Reg	2547/93120	1.269	787.836
			Lut	1857/46560		
			Logic	1675/46560		
			Memory	182/16720		
			Srl	182		
7	K-map Modification in Full Adder Structure (K-map-FA)	42.9336	Reg	3225/93120	0.927	1078.167
			Lut	3002/46560		
			Logic	3002/46560		
S.No	Description	Power (mW)	Area		Delay (ns)	Frequency (MHz)
8	Interlaced Partition Multiplier as Component Multiplier with Carry Look Ahead Adder (IPM-CLA)	42.008	Reg	3241/93120	1.177	849.618
			Lut	1703/46560		
			Logic	1523/46560		
			Memory	180/16720		
			Srl	180		
9	Booth Multiplier with Ripple Carry Adder (BM-RCA)	41.2248	Reg	580/93120	1.177	849.618
			Lut	1682/46560		
			Logic	1665/46560		
			Memory	17/16720		
			Srl	17		
10	Booth Multiplier with Carry Look Ahead Adder (BM-CLA)	41.5808	Reg	885/93120	0.770	1297.859
			Lut	1835/46560		
			Logic	1835/46560		

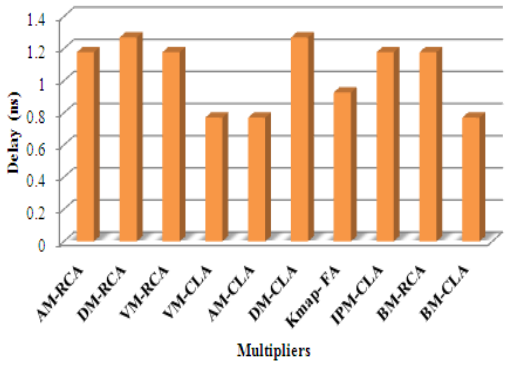

Figure 10(d). Time Delay Comparison

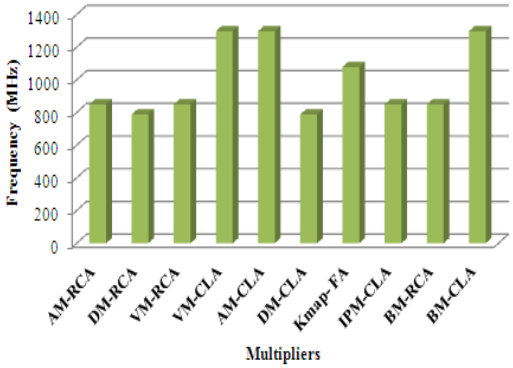

Figure 10(e). Frequency Requirement

Based on all above comparisons, it is concluded that the proposed interlaced separation multiplier with Urdhav – Tiryagbhyam Vedic multiplier outperforms all other multipliers with respect to area and power. The frequency and time delay can be made better by employing carry look ahead adder for addition operation in the multiplier with little increase in area and power.

VII. CONCLUSION AND FUTURE SCOPE

Circuits designed to perform binary multiplication are commonly used in digital signal processing applications and embedded system applications. It is noted that the interlaced separation Vedic multiplier is effectively reduced the length of the long carry chains in multiplier circuits. The domain is presented here in terms of speed and area in which each multiplier can be chosen and the choice must be left up to the application specific cost function. Various multipliers are used as component multiplier for the comparison purposes and the performances are evaluated in FPGA platform using VHDL codes. Out of these, the interlaced separation Vedic multiplier involving 8 bit Urdhav Tiryagbhyam Vedic multiplier as component multiplier and ripple carry adder shows the best and it only requires 40.6652mW power (13.24% less), 1550 registers(15.87% less), 1314 look up tables (18.65% less), and achieves 1.177ns speed, 849.618 MHZ frequency. It shows that it needs 13.24% less power, 15.87% less registers, 18.65% less look up tables than the conventional multiplier to achieve the same speed and frequency. The speed can further be increased by replacing ripple carry adder with carry look ahead adder and achieves the speed of 0.770ns (34.58% fast) and 1297.859 MHz frequency (34.54% high), but with minimum increase in power and area.

In future, the proposed multiplier circuit can further be extended to higher number of bits for very complex digital circuit applications in order to achieve its desired performance in terms of area, power and also speed can be enhanced without a compromise in other performance constraints.

REFERENCES

[1] Christopher Fritz, and Adly T. Fam, "Interlaced Partition Multiplier", IEEE Transactions on Computers, Vol. 65, No. 8, pp. 2652-2658, 2016.
[2] C. S. Wallace, "A suggestion for a fast multiplier," IEEE Transactions on Electronic Computers, Vol. EC-13, No. 1, pp. 14–17, 1964.
[3] L. Dadda, "Some schemes for parallel multipliers," Alta Frequenza, Vol. 34, pp. 349–356, 1965.
[4] P. Kogge and H. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," IEEE Transactions on . Computers., Vol. C-22, No. 8, pp. 783–791, 1973.
[5] H. Eriksson, P. Larsson-Edefors,M. Sheeran, M. Sjalander, D. Johansson, and M. Scholin, "Multiplier reduction tree with logarithmic logic depth and regular connectivity," in Proceedings of the 2006 IEEE International Symposium on Circuits and Systems, Canada, pp. 4–8.
[6] R. Waters and E. Swartzlander, "A reduced complexity Wallace multiplier reduction," IEEE Transactions on Computers, Vol. 59, No. 8, pp. 1134–1137, 2010.
[7] M. Schulte and E. Swartzlander, "Parallel reduced area multipliers," Journal of VLSI Signal Processing Systems, Vol. 9, pp. 181–191, 1995.
[8] E.L.Brawn, "Digital Computer Design", New York Academic Press, New York, pp. 494-498, 1963.
[9] A.Habibbi and P.A.Wintz, "Fast Multipliers", IEEE Transactions on Computers, Vol.C-19, pp. 153-157, 1970.
[10] K.C.Bickerstaff, M.J.Schulte, and EE.Swartzlander,Jr., "Reduced Area Multipliers", Proceedings of the 1993 International Conference on Application Specific Array Processors, USA, pp. 478-489.
[11] D. Naresh, Giri Babu Kande, "High Speed Signed multiplier for Digital Signal Processing Applications", IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE) ,Vol 8, Issue 2 , pp 57-61,2009.
[12] Mohammed Hasmat Ali, Anil Kumar Sahani, "Study, Implementation and Comparison of Different Multipliers based on Array, KCM and Vedic Mathematics Using EDA Tools", International Journal of Scientific and Research Publications, Vol. 3, Issue 6, 2013.
[13] S.K.Panda, R.Das, S.K.Saifur Raheman, Tapasa Ranjan Sahoo, "VLSI Implementation of Vedic Multiplier Using Urdhva–Tiryakbhyam Sutra in VHDL Environment: A Novelty", Vol. 5, No 1, pp.17-24, 2015.

[14] A Debasish Subudhi, Kanhu Charan Gauda, Abinash Kumar Pala, Jagamohan Das," Design and Implementation of High Speed 4x4 Vedic Multiplier", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 4, Issue 11, pp.362-366, 2014.
[15] W. J. Townsend, E. E. Swartzlander Jr., and J. A. Abraham "A comparison of Dadda and Wallace multiplier delays," in the 2003 proceedings of SPIE. on Advanced Signal Processing Algorithms, Vol. 5205, pp. 552–560.
[16] T. K. Callaway and Jr. E. E. Swartzlander, "Optimizing Multipliers for WSI," in the 1993Proceedings of the Fifth Annual IEEE International Conference on Wafer Scale Integration, USA, pp. 85-94.
[17] Chandan Singh, "Realization of FIR Filter using Distributed Arithmetic Architecture", International Journal of Scientific Research in Computer Science and Engineering Vol. 3, Issue-4, pp. 7-12, 2015.
[18] Anil Pratap, "Analysis of Full and Half Adder Using Different Logic Style", International Journal of Scientific Research in Computer Science and Engineering, Vol. 4, Issue.1, pp. 6-9, 2016.

Authors Profile

M Isaivani received B.E degree in Electronics and Instrumentation Engineering from Anna University in the year of 2006. She has completed M.E in Embedded System Technologies under Anna University in 2014. She is currently doing Ph.D. in full-time at Anna University Regional Campus Madurai. Her research interests include VLSI, Embedded System applications to Power System Protection.

V Malathi is working as a professor in the department of Electrical and Electronics Engineering and Dean in Anna University Regional Campus Madurai. She completed her Bachelor Degree in College of Engineering Guindy and her Master's Degree in Thiyagaraja College of Engg, Madurai. She Completed her Ph.D. in Anna University Chennai and her areas of interest are Intelligent Techniques and its Applications, Smart Grid, FPGA based Power System and Automation.

E Sakthivel received the Bachelor degree in Madurai Kamarajar University, Madurai and the Master's degree in Embedded system Technologies in Anna University, Thirunelveli and he completed his Ph.D. degree in Anna University, Chennai. Currently, he is working as an associative professor at PSR Engineering College. He has more than 10 years of industrial experience in VLSI Technology and Embedded System. His current research interests include Embedded Systems, Low-Power Design, Network On Chip, System-On-Chip, FPGA and ASIC based power electronics control circuits, FPGA based power system, Wireless Networks, Instrumentation and Object Recognition.

