Multiplication Algorithms for VLSI - A Review

Piyush S. Kasat[1#], Devendra S. Bilaye[1],
Harish V.Dixit[2*], Rahul Balwaik[2], Amutha Jeyakumar[2]
[1] V.C.E.T
Vasai, India
[#]impiyushkasat@gmail.com, dev_bilaye@yahoo.co.in
[2] V.J.T.I
Matunga, India
*hvdixit48@yahoo.com, rahul.balwaik@yahoo.com

***Abstract*— In today's digital world, where portable computers have become as small as the size of palm limitation on processing speed has increased. Thus there's a need for modification in the traditional approach to overcome this limitation. An implementation using parallel and pipelined approach could work at higher speed while occupying limited number of slices. Paper deals with analyzing and reviewing different multiplication algorithms viz. Vedic, Chinese, Wallace, Booth, Karatsuba and Toom-Cook by performing 11*8 bit multiplication using parallel and pipelined approach.**

Keywords- Multiplication Algorithms, Pipelining.

I. INTRODUCTION

One of the unsolved questions in computer science is "what is the fastest algorithm for multiplication of two numbers having arbitrary number of digits". Considering that this question is unsolved to date itself brings upon the importance of multipliers in computer science and it manifests even greatly in the field of VLSI design. Multipliers often find wide use in complex digital system design and are often used in processors computing FFT [1], DCT [2], general purpose processors [3] et cetera. The design of each system involves careful analysis of multiplication algorithms to find an algorithm which satisfies the system requirement at the optimum can be implemented. The most critical factors which decide this are the speed, area and the power consumption. Even the same algorithm would give different results for the above three parameters when implemented in different ways. For example, it is widely known that serial architecture involves a lot less area than a parallel one, but the speed provided by it is far less. As such the nature of the system in accordance to the specification finally decides which algorithm should be used.

Real time systems like video decoder and encoder often are required to work at high speeds. Invariably in such type of systems the speed of multiplication is of prime importance. The Discrete Cosine Transform is most commonly used in such type of encoders [4]. Generally the DCT involves matrix multiplication [5] and is thus computationally very intensive. To meet the requirements of such a system a parallel architecture is deemed suitable. Further to increase the throughput, a pipelined system can be developed. Thus for computationally intensive systems where speed rather than area are of prime constraints a parallel pipelined architecture is more suitable.

This paper reviews some well known multiplication algorithms viz. Booth[6], Chinese[7], Karatsuba[8], Toom Cook[9][10], Wallace[11], Vedic[12] and presents a comparative study of the area and speed reports of their implementation when a parallel pipelined architecture is followed.

Section II contains detail explanation on different Algorithms for multiplication. Section III contains description of Architecture. Section IV contains Results and Discussion and Section V contains the References.

II. ALGORITHMS

An algorithm for multiplication of two signed bit numbers was invented by Andrew Donald Booth in 1950[6]. Booth's algorithm works on the principle of shift and add. Fig.1 illustrates multiplication of two 4 bit nos. m and n using Booth's Algorithm. Initially a new number 'P' is formed by appending 0s, equal to one more than the number of bits of multiplier to the MSB of multiplicand. Also duplication of MSB i.e. sign bit of multiplier n and multiplicand m is done along with which 0s are appended to the LSB of multiplier and multiplicand so as to make length of multiplicand and multiplier equal to P, consider they be X and A respectively. Let S be 2's compliment of X for subtraction purpose. In Fig.1 for simplicity purpose the LSB bit of multiplier (n) and n_{lsb} are shown in bold, these are the pair of bits of multiplier which are analyzed and a respective operation of shift and add is performed on P depending on the following four conditions.

If pair of bits is:

00 or 11, circular right shift is performed on P,

01, multiplicand is added to the MSB of P and then circular right shift is performed on P,

10, subtract Multiplicand from the MSB of P and then circular right shift is performed on P.

Initially, n_{lsb} bit is considered 0, in the immediate next cycle, a copy of lsb bit of multiplier is made in n_{lsb} column and circular right shift operation is performed on multiplier. The process continues until the no. of cycles is equal to the no. of bits in the multiplier. The result so obtained in P at the end of the last cycle is the final result of multiplication.

n = 1111 X= 1 1111 0000 S = 0 0001 0000
m= 1111 A = 1 1111 0000 P = 0 0000 1111

	P	Multiplier (n)	n_{lsb}	Operation
	0 0000 1111	1111	0	
1st Cycle	0 0001 1111 1 0000 1111	1111 1111	 1	P = P + S Circular Right Shift
2nd Cycle	1 1000 0111	1111	1	Circular Right Shift
3rd Cycle	1 1100 0011	1111	1	Circular Right Shift
4th Cycle	1 1110 0001			Circular Right Shift

Figure 1. Illustration of Booth's Algorithm

Another algorithm for multiplication of two numbers was implemented by Chris Wallace in 1964 [11]. Fig.2 illustrates the working of Wallace Algorithm for multiplication of two 4 bit nos. At first, the partial products P1, P2, P3, P4 are computed. As demonstrated in Fig.2(a) first three partial products are selected where P3 is right shifted left by 2 bits and P2 by 1 bit and finally is column-wise added with P1 to give an intermediate result as shown in Fig.2(b). This intermediate result is column-wise added with the immediate next partial product term P4 which is shifted left by 1 more bit as shown in Fig.2(c). The process continues until all the partial product terms are added to give the final result.

n					1	1	1	1
m				x	1	1	1	1
P1					1	1	1	1
P2				1	1	1	1	x
P3			1	1	1	1	x	x
P4		1	1	1	1	x	x	x

P1					1	1	1	1
P2				1	1	1	1	x
P3			1	1	1	1	x	x
Sum			1	0	1	1	0	1
Carry			1	1	1	1	x	x

Sum			1	0	1	1	0	1
Carry			1	1	1	1	x	x
P4		1	1	1	1	x	x	x
Result	1	1	1	0	0	0	0	1

Figure 2. Illustration of Wallace Algorithm (a) Computation of Partial Products
(b) Addition of first three Partial Product Terms (c) Result

Another algorithm was proposed by Anatolii Alexeevitch Karatsuba in 1960 [8] the flow graph of which is shown in Fig.3. Initially both the multiplicand and the multiplier say m and n are divided into two equal parts m0,m1 and n0,n1. Algorithm proposes evaluation of three intermediate terms u_0, u_1 and u_2. They are computed as shown in Fig.3. The final result is obtained following the equation mentioned in the last layer of the flow chart. The base mentioned in the equation depends on the number system in which the two numbers are, i.e. 2 for binary and the shift mention in the equation is half the length.

Toom-Cook Algorithm was first described by Andrei Toom[9] in 1963 and was implemented by Stephen Cook in 1966 in his PhD thesis[10]. The algorithm is similar to that Karatsuba Algorithm with only one modification

which is dividing the given number into multiple equal parts. As seen in Fig. 3, where numbers are divided into two parts m0,m1 and n0,n1 similarly, numbers could be divided into multiple equal parts, say 3. Consider two numbers m and n, divided in 3 equal parts m0,m1,m2 and n0,n1 ,n2. The product of m and n could be obtained from the intermediate terms, which are obtained from the smaller terms m0,m1,m2 and n0,n1,n2.

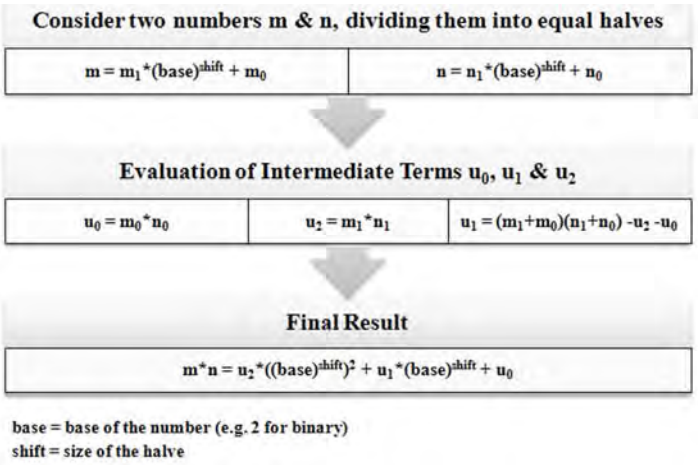

Figure 3. Flowchart illustrating working of Karatsuba Algorithm

Vedic Algorithm, specifically Urdhva Tiryagbhyam (vertical and crosswise multiplication) is one of the 20 sutras found in ancient hindu text dated back to 500B.C [12]. As shown in Fig.4 each and every bit of multiplicand and multiplier are ANDed with each other producing an intermediate result, this result is ORed with another intermediate result similar to what is pictorial representation in Fig.4 to produce partial products P1,P2,P3,…,P7. Final Result thus is obtained by adding each of the partial product terms by sequentially left shifting each term.

n1					1	1	1	1
n2				×	1	1	1	1
p1								1
p2	+					1	0	
p3	+				1	1		
p4	+		1	0	0			
p5	+		1	1				
p6	+	1	0					
p7	+	1						
Answer	1	1	1	0	0	0	0	1

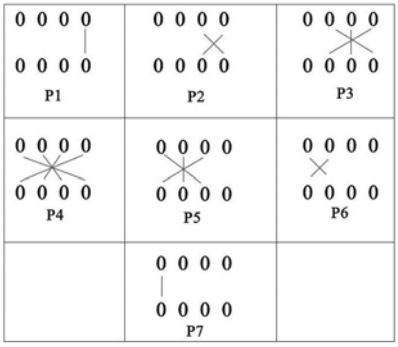

Figure 4. Illustration of Vedic Algorithm[13]

Chinese Algorithm finds its origin in the mathematical text Zhou Bi Suan Jing which goes back to 300B.C. [7]. The working of the algorithm is similar to that of Vedic. Fig.5 illustrates evaluation of partial products as per the algorithm. Initially, a table is drawn whose rows and columns are equal to the length of multiplier and multiplicand resp. and diagonals are drawn for each cell as shown in the Fig.5. Diagonal divides the cell into upper and lower triangle. The bits of multiplier and multiplicand are arranged across the table as shown in the Fig.5, these individual bits of multiplier and multiplicand are multiplied with each other to produce a result, quote the carry of result in the upper triangle and sum in the lower triangle of the corresponding cell. The partial products P1,P2,…,P8 thus will be obtained by adding elements as shown in the Fig.5. Similar to vedic algorithm these partial product terms are added to produce final result.

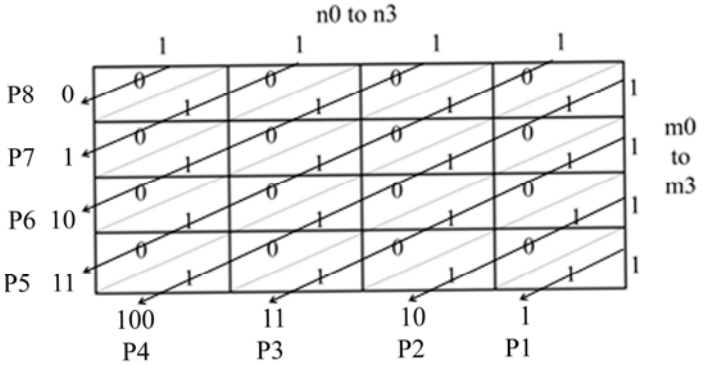

Figure 5. Evaluation of Partial Products using Chinese Algorithm

III. ARCHITECTURE

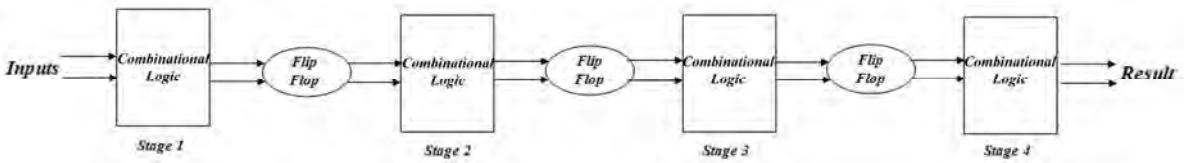

Figure 6. Illustration of Parallel and Pipeline Approach

The most prominent disadvantage of serial approach is the time delay which it takes in producing the result. This drawback can be overcome if it is possible to implement the same design in a manner where the inputs are parallel and the entire process is executed in pipelined stages. Fig.6 demonstrates the parallel and pipelined approach. The key feature of parallel and pipelined approach is that every individual unit functions separately but simultaneously. A set of input is given to the first stage which consists of combination logic which processes the inputs during a clock cycle and resulting data is passed on to the flip flops which work as a register and store the result at the arrival of the clock. At the immediate next clock pulse the stored data in the flip flop is passed on to the next stage of combination logic and a new set of input arrives at the first stage of combination logic. The resulting data of first and second stage is stored and passed on to the next stage. The process continues until the final result of the final set of input is obtained. This approach results in evenly distribution of computation process in multiple blocks. Thus it can well be stated that the processing throughput is increased by the factor of the number of pipeline stages for a given process.

In [13] an architecture has been proposed for implementation of Vedic Multiplier using parallel and pipelined approach. The same approach has been used to implement the multiplication algorithms and each of these algorithms have different number of pipeline stages ranging from 7 for Vedic Multiplier to 15 for Karatsuba Multiplier.

IV. RESULTS

The synthesis results of the above mentioned algorithms have been obtained using Xilinx ISE Design Suite (version 13.2) on Spartan 3E Starter edition development board. The primary attributes to be compared of the designed multipliers, based on different algorithms are area and speed. These factors are characterized by no. of slices utilized to implement the design and maximum frequency at which the designed multiplier can work respectively. The various algorithms can be easily compared with the following graphs, which are based on the above factors.

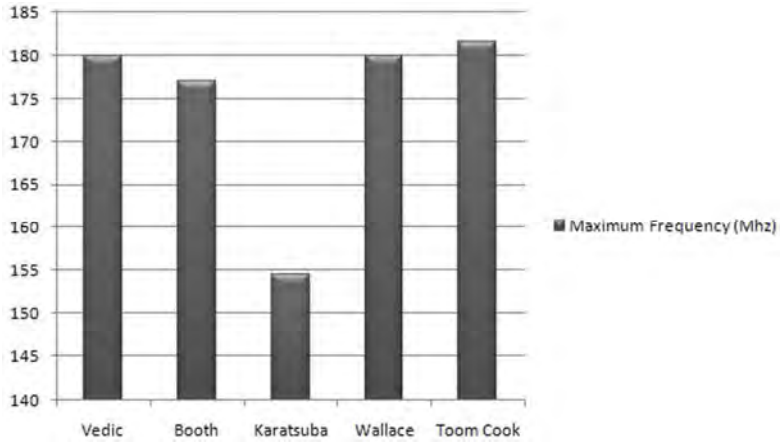

Figure 7. Maximum Clock Frequency (MHz)

From the bar graph shown in Fig. 7, it can be seen that after implementing different algorithms using Parallel and Pipelined Approach, the Toom-Cook Algorithm provides the maximum speed of 181.6MHz, next to which is Wallace and Vedic Algorithm both providing maximum speed of 179.9MHz and Booth's Algorithm providing the maximum speed of 177.1MHz. Whereas, Karatsuba provides the lowest speed of 154.6MHz.

Also form Fig. 8, which represents a bar graph showcasing the number of slices occupied, it can be seen that Wallace Algorithm occupies lowest number of slices i.e. 216, next to which is Vedic, Toom-Cook and Karatsuba Algorithm occupying 225, 245 and 277 slices respectively, whereas Booth's Algorithm occupied the maximum number of 346 slices.

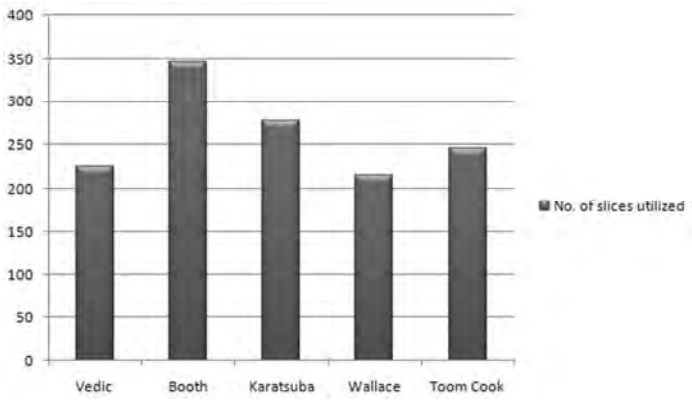

Figure 8. Number of Slices occupied

V. CONCLUSION

From the results obtained and the comparison made between different algorithms it can well be stated that there is a tradeoff between the area and speed consumed by the multipliers based on different algorithms. For example, Toom Cook multiplier gives the maximum speed of operation, but it consumes more area than the Vedic and Wallace multipliers. Thus, it can be said no multiplier efficiently satisfies all the criteria for an ideal design. Appropriate algorithm needs to be chosen as per the required application in larger system designs like DCT, DFT.

REFERENCES

[1] Nabeel Shirazi, Peter M. Athanas, and A. Lynn Abbott, Implementation of a 2-D Fast Fourier Transform on a FPGA-Based Custom Computing Machine, published in Springer Berlin Heidelberg as Lecture Notes in Computer Science at 5th International Workshop, FPL '95 Oxford, United Kingdom, August 29–September 1, 1995 Proceedings.

[2] R.Uma, FPGA Implementation of 2-D DCT for JPEG Image Compression, INTERNATIONAL JOURNAL OF ADVANCED ENGINEERING SCIENCES AND TECHNOLOGIES, Vol No. 7, Issue No. 1, 001 - 009.

[3] Petar Borisov Minev, Valentina Stoianova Kukenska, IMPLEMENTATION OF SOFT-CORE PROCESSORS IN FPGAs, published in INTERNATIONAL SCIENTIFIC CONFERENCE, November 2007, GABROVO.

[4] Olli Lehtoranta, Erno Salminen, Ari Kulmala, Marko Hännikäinen, and Timo D. Hämäläinen, A parallel MPEG-4 encoder for FPGA based multiprocessor SoC, International Conference on Field Programmable Logic and Applications, 2005.

[5] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete Cosine Transform", IEEE Trans. Computers, 90-93, Jan 1974.

[6] Andrew D. Booth. A signed binary multiplication technique. The Quarterly Journal of Mechanics and Applied Mathematics, Volume IV, Pt. 2

[7] Boyer, Carl B. (revised by Merzbach, Uta C.) (1991). History of Mathematics. John Wiley and Sons, Inc.. ISBN 0-471-54397-7

[8] A. Karatsuba and Yu.Ofman (1962)."Multiplication of Many-Digital Numbers by Automatic Computers".Proceedings of the USSR Academy of Sciences145: 293–294.

[9] Toom, Andrei L., The complexity of a scheme of functional elements realizing the multiplication of integers published in Soviet Math (translations of Dokl. Adad. Nauk. SSSR), 4, No. 3, 1963.

[10] Cook, Stephen A., On the Minimum Computation Time of Functions, 1966 PhD thesis, Harvard University Department of Mathematics.

[11] C. S. Wallace, A suggestion for a fast multiplier, IEEE Trans. on Electronic Comp. EC-13(1): 14-17 (1964)

[12] Jagadguru Swami Sri Bharath, KrsnaTirathji, “Vedic Mathematics or Sixteen Simple Sutras From The Vedas”, Motilal Banarsidas, Varanasi(India),1986.

[13] Harish Dixit, Piyush S. Kasat, Rahul Balwaik, Amutha Jeyakumar, “A Parallel Pipelined Approach to Vedic Multiplier for FPGA Implementation” ARTCom 2012, LNEE pp. 474–480, 2012. © Springer-Verlag Berlin Heidelberg 2012.