VLSI Implementation of High Speed MAC Unit Using Karatsuba Multiplication Technique

Naveen Khare
(M.Tech Scholar), Department of Electronics and Communication, SRIT, Jabalpur, (MP), India.

Divyanshu Rao
(Asst.Professor), Department of Electronics and Communication, SRIT, Jabalpur, (MP), India.

Ravi Mohan
(PG Course Incharge), Department of Electronics and Communication, SRIT, Jabalpur, (MP), India.

Abstract – This research work is devoted to design speed optimized Multiply Accumulate Unit. MAC Unit is a digital co-processor that plays a prominent role in digital domain to perform various sophisticated tasks such as FFT, DFT, resolving various complex equations being used in the digital domain etc. Furthermore that it is also used in various configurations such as IIR, FIR etc. Every digital domain based technology depends upon the operations performed by MAC Unit either partially or whole. MAC unit can performs multiply then accumulate operation. Speed is the most prominent factor of processor and controllers being used recently. To meet this major concern of "speed" it's imperative to have particular high speed MAC. That's why it is highly imperative to design speed efficient MAC unit, only because of this efficiency of those modules can be enhanced which lies upon the operations performed by MAC unit. The speed of MAC unit mainly depends upon the speed of multiplier. So many multiplication algorithms have been investigated and showcased now-a-days at algorithmic and structural level by researchers and scientists. After a thorough study and proper analysis we have seen that Karatsuba multiplication technique is the best algorithm that gives much better result in comparison to other conventionally used multipliers in terms of speed. In this dissertation work after introducing karatsuba multiplier in proper way, it has been implemented on practical grounds. Then we have proposed that multiplier for designing MAC unit. This proposed MAC Unit is able to perform different arithmetic operations at high speed. Combinatorial form has been utilized to design all sub-modules being used in the MAC unit. And integrated in the final unit, reset and clock functionality has been provided in this final unit to have better control on the circuitry. To design proposed MAC unit Verilog hardware description language (HDL) has been utilized. Data flow modeling has been used to design operational sub-modules and finally behavioral modeling style has been used to integrate these modules. For this design the target FPGA, which has been used belongs to Spartan-3 (family), XC3S50 (device), PQ208 (package) with speed grade of -5. Xilinx synthesis tool (XST) of Xilinx ISE-9.2i has been used for synthesis purpose. For the behavioral simulation purpose ISE simulator has been used.

Index Terms – Karatsuba multiplication algorithm, Arithmetic Unit, Arithmetic Logic Unit, Addition tree structure.

1. INTRODUCTION

MAC unit plays a key role in digital domain, and is able to perform "multiply then accumulate" operation. Following given block diagram can help to understand MAC unit working, where Mac unit has been showcased on the field programmable gate array (FPGA).

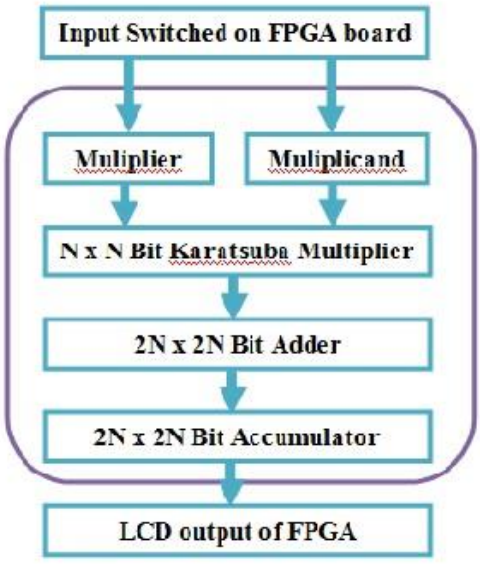


Figure 1.1 Block Diagram of MAC UNIT

There are two interfaces, through which the MAC Unit can be accessed. The input can be given using input switches of FPGA board. This input goes to multiplier and multiplicand respectively given by user. Then the processed output data can be seen at output device, which is LCD output of FPGA.

In computing, especially digital signal processing, most common operation is multiply–accumulate operation that computes the product of two numbers and adds that product

using accumulator. Hardware unit or Physical unit that performs this operation is known as a multiplier–accumulator (MAC, or MAC unit); the operation itself is also often called a MAC or a MAC operation. The MAC operation modifies an accumulator a: Multiplier–accumulator (MAC, or MAC unit); the operation itself is also often called a MAC or a MAC operation. The MAC operation modifies an accumulator a:

$$a \leftarrow a + (b \times c)$$

The MAC unit shown in the block diagram is able to perform arithmetic operations such as Addition and Multiplication.

Multiplication-and-accumulate operations are very important and dominant for digital filters. Therefore, by the use of speed optimized MAC unit high-speed filtering and other processing typical for DSP applications can be achieved. Along with this it can also process data separately than CPU, thereby reduce CPU load

The speed of this co-processor mainly depends upon the multiplication unit. As multiplication unit consumes more area, power and energy so it is a unit of high importance. So, it can be said that by optimizing the multiplication unit, MAC unit can also be optimized.

2. PROPOSED KARATSUBA MULTIPLIER ALGORITHM

In this the polynomial based multiplication is performed. For the algorithmic representation we have taken two numbers one is multiplicand say X and second one is multiplier say Y. This multiplier works on the same logic as any decimal number can be represented with its base number. For example (1432)10 can be represented as 14x102 + 32. Similarly X and Y can be rewritten as: -

X = X1 Bm + X0.

Y = Y1 Bm + Y0.

Here B is the base number, and value of m is chosen in a proper form such that Bm should be greater than the X0 and Y0.

Now multiply these two numbers X and Y: -

X*Y = (X1 Bm + X0)*(Y1 Bm + Y0).

= Z2B2m + Z1Bm + Z0

Where

Z2 = X1Y1;

Z1 = ((X1+X0)* (Y1+Y0)) – Z2– Z0;

= ((X1+X0)* (Y1+Y0)) – X0Y0– X1Y1.

Z0 = X0Y0;

Then putting these values to the equation "Z2B2m + Z1Bm + Z0" final product can be obtained. Advantage of this algorithm is its high speed performance.

Illustration with example: -

X = (1001)2 = (10*22 + 01)2;

Y = (0110)2 = (01*22 + 10)2;

Z2 = X1Y1 = (10*01)2 = (10)2;

Z0 = X0Y0 = (01*10)2 = (10)2

Z1 = ((X1+X0)* (Y1+Y0)) – Z2– Z0 = (10+01)2 * (01+10)2 – (10)2 – (10)2 = (101)2;

Final result = Z2B2m + Z1Bm + Z0 = (10)2*(24)10+ (101)2 * (22)10 + (10)2 = (110110)2.

But it also has a limitation that value of Z1 depends upon the value of Z2 and Z1. By overcoming this limitation further enhancement can be done upon the speed of this multiplier. So, to overcome this limitation there is another way to find out the value of Z1, is that we can use Vedic multiplication algorithm to get the value of Z1. As shown in the following: -

Z1 = X1Y0 + X0Y1;

Illustration how did we obtained Z1's value:-

Z1 = ((X1+X0)* (X1+X0)) – Z2– Z0

= ((X1+X0)* (X1+X0)) – X1Y1 – X0Y0

= ((X1+X0)* (X1+X0)) – X1Y1 – X0Y0

= (X1Y0 + X1Y0 + X0Y1 + X0Y0) – X1Y1 – X0Y0

= X1Y1 + X1Y0 + X0Y1 + X0Y0 – X1Y1 – X0Y0

= X1Y0 + X0Y1

And other things will be same as shown previously. If this way is used to find the value of Z1 then it is called as the "Vedic Karatsuba multiplication algorithm".

Illustration with example: -

X = (1001)2 = (10)2 * (22)10 + (01)2;

Y = (0110)2 = (01)2 * (22)10 + (10)2;

Z2 = X1Y1 = (10*01)2 = (10)2;

Z0 = X0Y0 = (01*10)2 = (10)2;

Z1 = X1Y0 + X0Y1 = (10*10 + 01*01)2 = (101)2 ;

Final result = Z2B2m + Z1Bm + Z0 = (10)2 * (24)10 + (101)2 * (22)10 + (10)2 = (110110)2.

Proposed MAC Unit design :- After designing the thirty two bit multiplier, we have placed this module in the MAC unit along with conventional modules, digital hardware for the same is given in the following, which is self-explanatory in itself: -

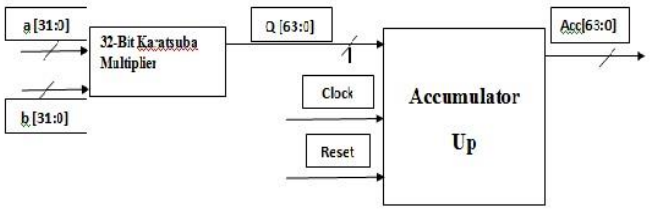


Figure 2.2 Hardware of Proposed Thirty two Bit MAC Unit

3. QUNTITATIVE RESULTS

Following table shows the area and timing constraints of proposed karatsuaba multiplier at different bit levels.

Proposed N Bit kartsua multiplier	Number of LUT used as logic	Number of occupied Slices	Total eq. gate count for design	Additional JTAG gate count for design	Maximum combinational path delay (ns)
4-Bit	25	14	226	768	14.947 ns
8-Bit	18	11	16,229	1536	14.765 ns
16-Bit	40	21	16461	3072	15.839 ns
32-Bit	80	41	16925	6144	17.403 ns

Table3.1 Device Utilization Summary of N-Bit Multiplier

4. COMPARATIVE RESULTS

To show the efficiency of proposed karatsuba multiplier at eight bit level, it has been compared with some other popular multiplier structures based on different multiplication algorithms at the eight bit level. For the comparison purpose some standard papers have been used. For true and reliable comparison, proposed multiplier has been implemented on the same platform of target FPGA, which has been used by the reference papers. Comparative tables are shown below:-

In the following given table, target FPGA belongs to Spartan 3AN (family), XC3S50A (device), -7 (speed grade). This multiplier is with clock circuitry, so we have also placed clock circuitry with our multiplier: -

Path Delay (in ns) for different multipliers at sixteen bit level	
Urdhva triyagbhyam multiplier [1]	Proposed
21.647	15.266

Table 4.1 Comparison Table No. 1 at Sixteen Bit Level

In the following given table which target FPGA has been used belongs to Spartan 3 (family), XC3S400 (device), TQ144 (Package), -5 (speed grade).

Maximum Combinational Path Delay (in ns) for different multipliers at eight bit level				
Array multiplier [3]	Booth multiplier [3]	Vedic multiplier with KSA[3]	Vedic multiplier with KSA[2]	Proposed
32.01	29.549	23.644	28.699	14.765

Table 4.2 Comparison Table No. 2 For Different Multipliers at Eight Bit Level

In the following given table which target FPGA has been used belongs to Spartan 3E (family), XC3S500E (device), FG320 (Package), -5 (speed grade).

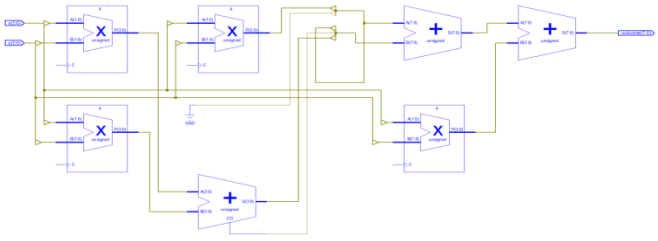
Maximum Combinational Path Delay (in ns) for different multipliers at eight,sixteen and thirty two bit level			
Bit level	Karatsuba multiplier [4]	Vedic multiplier with csa [4]	Proposed
8	31.029	15.418	13.468
16	46.811	22.604	14.339
32	82.834	31.526	23.916

Table 4.3 Comparison Table No. 3 For Different Multipliers at Eight/Sixteen/Thirty Two Bit Level

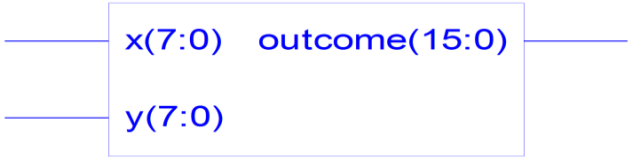
5. SYNTHESIS AND SIMULATION

1. RTL Schematic 4 Bit Karatsuba Multiplier Top View

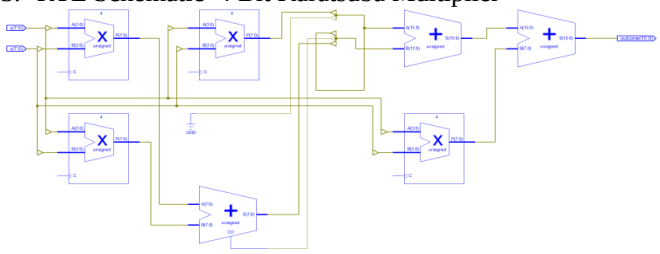
RTL Schematic 4 Bit Karatsuba Multiplier



2. RTL Schematic 4 Bit Karatsuba Multiplier Top View



3. RTL Schematic 4 Bit Karatsuba Multiplier



Device Utilization Summary of Proposed 4-Bit Karatsuba Multiplier

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of 4 input LUTs	25	1,536	1%
Logic Distribution			
Number of occupied Slices	14	768	1%
Number of Slices containing only related logic	14	14	100%
Number of Slices containing unrelated logic	0	14	0%
Total Number of 4 input LUTs	25	1,536	1%
Number of bonded IOBs	16	124	12%
Total equivalent gate count for design	226		
Additional JTAG gate count for IOBs	768		

Figure 5.1 Device Utilization Summary of Proposed 4-Bit Karatsuba Multiplier

Device Utilization Summary of Proposed 8-Bit Karatsuba Multiplier

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of 4 input LUTs	18	1,536	1%
Logic Distribution			
Number of occupied Slices	11	768	1%
Number of Slices containing only related logic	11	11	100%
Number of Slices containing unrelated logic	0	11	0%
Total Number of 4 input LUTs	20	1,536	1%
Number used as logic	18		
Number used as a route-thru	2		
Number of bonded IOBs	32	124	25%
Number of MULT18X18s	4	4	100%
Total equivalent gate count for design	16,229		
Additional JTAG gate count for IOBs	1,536		

Figure 5.2 Device Utilization Summary of Proposed 8-Bit Karatsuba Multiplier

Proposed 4-bit Karatsuba multiplier simulation waveform, where inputs and output are in decimal format: -

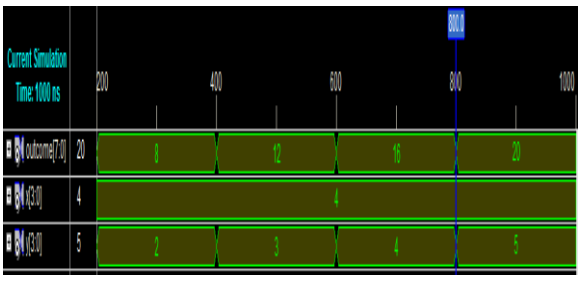


Figure 5.3 Proposed 4-Bit Karatsuba Multiplier Simulation Waveform

Proposed 8-bit Karatsuba multiplier simulation waveform, where inputs and output are in decimal format: -

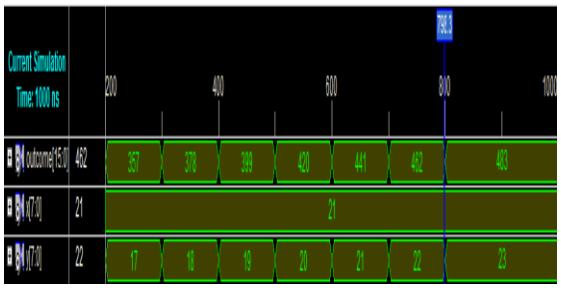


Figure 5.4 Proposed 8-Bit Karatsuba Multiplier Simulation Waveform

6. CONCLUSION

In this research work a deep analysis and study work has been performed on different types of multipliers. And found that Karatsuba multiplier is a special kind of polynomial multiplier that gives better outcomes in comparison to other conventional multipliers. So we have used this multiplier and designed 32-bit multiplier using karatsuba multiplication approach and placed it in the 32-bit MAC unit's multiplier module. After this we have synthesized and simulate this MAC unit to get reliability on our designed module.

REFERENCES

[1] Shivakumar M V, Mrs. Anitha Kumari, "A Novel approach towards performance analysis Of Vedic multiplier using FPGA's", International Journal of Electronics, Electrical and Computational System IJEECS ISSN 2348-117X Volume 3, Issue 4 June 2014.

[2] Sudeep.M.C, Sharath Bimba.M, Mahendra Vucha," Design and FPGA Implementation of High Speed Vedic Multiplier", International Journal of Computer Applications (0975 – 8887) Volume 90 – No 16, March 2014.

[3] PushpalataVerma and K.K.Mehta "Implementation of an efficient Multiplier based on Vedic Mathematics using EDA tool.", International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-1, Issue-5, June 2012.

[4] G.Ganesh Kumar, V.Charishma, "Design of High Speed Vedic Multiplier Using Vedic Mathematics Techniques", International Journal of Scientific and Research Publications, Volume 2, Issue 3, March 2012 1 ISSN 2250-3153.

[5] Gaurav Sharma, Arjun Singh Chauhan, Himanshu Joshi, Satish Kumar Alaria, "Delay Comparison of 4 by 4 Vedic Multiplier based on Different Adder Architectures using VHDL", International Journal of IT, Engineering and Applied Sciences Research (IJIEASR) ISSN: 2319-4413 Volume 2, No. 6, June 2013.

[6] 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, Volume 3, Issue 6, June 2013 ISSN 2250-3153.

[7] Krishnaveni D., Umarani T.G.,"VLSI IMPLEMENTATION OF VEDIC MULTIP-LIER WITH REDUCED DELAY", International Journal of Advanced Technology & Engineering Research (IJATER) National Conference on Emerging Trends in Technology (NCET-Tech).

[8] Devika Jaina, Kabiraj Sethi, Rutuparna Panda, "Vedic Mathematics Based Multiply Accumulate Unit", IEEE, 978-0-7695-4587-5/11, 2011.

[9] M. Ramalatha, K. Deena Dayalan, P. Dharani, S. Deborah Priya, "High Speed Energy Efficient ALU Design using Vedic Multiplication Techniques", IEEE, 978-1-4244-3834-1/09, 2009.
[10] Ramesh Pushpangadan, Vineeth Sukumaran, Rino innocent, Dinesh sasikumar, Vaisak Sundar, "High Speed Vedic Multiplier for Digital Signal Processors", IETE Journal of Research, Volume 55, Issue 6, Nov-Dec 2009.
[11] S.S. Kerur, Prakash Narchi, Jayashree C N, Harish M. Kittur, Girish V A, "Implementation of Vedic Multiplier for Digital Signal Processing",proceedings published by International Journal of Computer Applications, presented in International conference on Vlsi, Communication and Instrumentation (ICVCI) 2011.
[12] V.K.Karthik, Y.Govardhan, V.Karunakara Reddy, K.Praveena, "Design of Multiply and Accumulate Unit using Vedic Multiplication Techniques", International Journal of Scientific & Engineering Research, Volume 4, Issue 6, June-2013 756 ISSN 2229-5518.
[13] Vaijyanath Kunchigi, Linganagouda Kulkarni, Subhash Kulkarni, "32 Bit MAC unit Design Using Vedic Multiplier", International Journal of Scientific and Research Publications, Volume 3, Issue 2, February 2013 1 ISSN 2250-3153.
[14] Manoranjan Pradhan, Rutuparna Panda, Sushanta Kumar Sahu, "MAC Implementation using Vedic Multiplication Algorithm", International Journal of Computer Applications (0975 – 8887) Volume 21– No.7, May 2011.
[15] Gary C.T. Chow , Ken Eguro, Wayne Luk and Philip Leong, "",
[16] Prof. S. Srinivasan, Lecture 15 – "ARRAY MULTIPLIER", Lecture series on Digital Circuits & Systems, Department of Electrical Engineering, IIT Madras. http://www.youtube.com/watch?v=5-PI4T25OXI.
[17] kiat seng yeo, kaushik roy" low voltage low power vlsi subsystems", Book, p.p. 119-144, Mc Graw Hill Professional Engineering, 2005.
[18] Sameer Palnitkar , "Verilog HDL-A Guide to Digital Design and Synthesis", Book, Sun Soft Press,1996.
[19] Stephen D. Brown, and ZvonkoVranesic, "Fundamentals of Digital Logic with Verilog Design, 2nd Edition,", Book McGraw Hill, June, 2007.
[20] Morris Mano, "Digital circuits and systems", Book, Prentice-Hal, Last updated on-02/27/2011, 11:55.
[21] Wayne Wolf, "Computer as components", 2nd edition, Book.
[22] http://en.wikipedia.org/wiki/Verilog
[23] Xilinx press release, for Xilinix ISE 9.2i, at SAN JOSE, Calif., June 25, 2007. http://www.xilinx.com/prs_rls/2007/software/0786_ise92i.htm