Quick Review on multiplication Algorithm for enhancing efficiency of MAC Unit.

Mr. Irshad Khan [1], Mr. Sunil Shah [2], Prof. Vinod Kapse [3]
Embedded Systems and VLSI Design [1,2,3]
Department of Electronics & Communication [1,2,3]
M.tech-IVth semester [1] , Professor[2,3]
Gyan Ganga Institute of Technology and Sciences [1,2,3]
Rajeev Gandhi Technical University, Jabalpur, Madhya Pradesh, India [1,2,3]

Abstract- **This review work is devoted for seeking speed efficient Multiply Accumulate Unit. As we know that MAC Unit is a digital co-processor which can performs multiply then accumulate operation. MAC Unit is the heart of digital signal processors to perform various sophisticated tasks that includes FFT, DFT, resolving various complex equations and Convolution etc. apart from this it is also used in various other configurations such as IIR, FIR etc. Every digital domain based technology depends upon the operations performed by MAC Unit either partially or whole. Speed is the most prominent factor of processor and controllers being used recently. To meet this major concern of "speed" we need particular high speed MAC. That's why it is highly required to design high speed MAC, which can enhance the efficiency of those modules which lies upon the operations performed by MAC. The speed of MAC greatly depends upon the speed of multiplier. At algorithmic and structural level there are so many multiplication algorithms exist now-a-days. After a thorough study and proper analysis we have seen that Vedic multiplication technique is the best algorithm that gives much better result in comparison to others in terms of speed. Further we have analyzed different existing Vedic multiplication hardware, and compared those with respect to speed. And found that the Vedic multiplier with Carry Save Adder gives better outcome. Then we have proposed that multiplier for the MAC unit design. This proposed MAC Unit is able to perform different arithmetic operations at high speed. All sub-modules in the MAC unit has been designed in combinatorial form. And integrated in the final module, in which we have provided the reset and clock functionality to have better control on the circuitry.**

Keywords – **Karatsuba multiplier, MAC Unit.**

I. INTRODUCTION

Introduction about MAC unit:-

MAC is a digital coprocessor, which is able to perform "multiply then accumulate" operation. To understand about Mac unit the following block diagram can be used, where Mac unit has been implemented on the field programmable gate array (FPGA). 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.

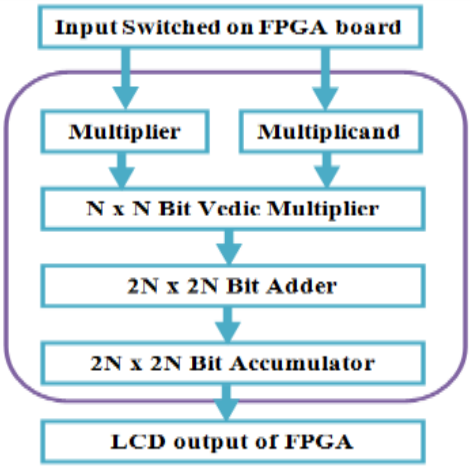

Figure 1 MAC Unit Block Diagram

In computing, especially digital signal processing, the multiply–accumulate operation is a common step that computes the product of two numbers and adds that product to an accumulator. The hardware unit that performs the 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*:

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

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.

II. ARRAY VERSUS TREE MULTIPLIER STRUCTURE

MAC Unit is the heart of all digital processors. So, by optimizing this co-processor, efficiency of those digital processors can be enhanced, which are based upon this co-processor. The speed of MAC mainly depends upon the speed of multiplication unit. So by optimizing the speed of multiplication unit of MAC, the more efficient MAC Unit can be designed. At the structural and algorithmic level many different types of the multipliers has been built up. Multipliers which work at binary level can be classified mainly by their structural design and algorithms they used to perform the operation. On the basis of structural design, multipliers can be classified in two groups first one is array multiplication structure and second one is tree multiplication structure. In every multiplier two types of operations get performed first generation of partial products and second is addition of those partially generated products to achieve final result. This addition can be either row serial or tree alike format, multipliers which uses this row serial addition structure belongs to array multiplication structure, similarly multipliers which uses tree alike addition structure belong to tree multiplication structure. After a keen observation we have found that, Out of these two structures the tree multiplication structure provides better speed in comparison to the array multiplication structure. As in tree multipliers, the partially generated products are arranged in a predefined tree alike format to reduce the critical path delay occurred in array multiplier structure. After placing these partially generated products in the pre-specified format the addition tree structure is obtained. Then addition operation is performed on this addition tree structure to achieve final result. Wallace, Modified Booth Wallace multipliers are some examples whose designs are based upon the tree multiplication structure. Most of the popular multipliers such as array, booth, Wallace, Modified Booth Wallace multipliers etc. works upon the conventional multiplication algorithms. The difference between these multipliers is only with respect to the multiplication architecture/structure and/or addition tree architecture/structure used by them. In short we can say that these multipliers are performing will at architectural level. But efficiency of these multipliers can be enhanced further by moving these multipliers towards the Vedic multiplication algorithms instead of Conventional multiplication algorithms. There is one more different kind of the multiplier named as Karatsuba multiplier. That uses a totally different algorithm to achieve result/outcome. This multiplier works upon the polynomial multiplication method. All of these multipliers are discussed in the following section.

III. DIFFERENT TYPES OF MULTIPLICATION ALGORITHMS

Multiplication is the most dominant operation in the arithmetic as well as it is the considered as the main operation in many technologies. Every multiplier works in two steps first one is the generation of partial products and second one is the addition of those partial products. But it depends upon the requirement and type of multiplier that how does it performs two these operations.

3.1 Repeated Addition Algorithm

In the early days to perform multiplication, repeated addition algorithm was used. In this to get the final resultant the multiplier get added itself multiplicand times. This was a very time consuming process but this algorithm required less number of hardware circuitry.

Advantage: - less hardware circuitry required.

Disadvantage: - Time consuming.

3.2 Shift and Add Multiplication Algorithm

This multiplier is better in terms of speed in comparison to "Repeated Addition Algorithm". Steps which are followed to gather the final resultant are as follows: -

1. Partial products get generated and after then appropriate shifting operation get performed on these partially generated products.

2. Addition of these shifted partially generated products gets performed to find the final product term.

The addition operation can be either serial or parallel. In the serial addition operation large storage registers are required to store the partially generated products. But also advantage is that it requires less hardware circuitry. In the parallel addition operation we can get higher speed compared to the serial addition operation but with the expense of high amount of hardware.

So, on the above given basis "shift and add" multiplier can be classified in two types: -

- Serial multiplier.

  Advantage: - it requires less hardware circuitry.

  Disadvantage: - Time consuming.

- Parallel multiplier.

  Advantage: - Better in speed, in comparison to serial multiplier.

  Disadvantage: - it requires high amount of hardware.

3.3 Array Multiplication Algorithm

This multiplier is based upon, parallel multiplication structure of "shift and add" multiplier and performs row serial addition. It consists of array of adders, to add partially generated products, hence it's called as Array Multiplier. To explain this multiplication algorithm, we have illustrated an example of 4x4 array multiplier, demonstrated by Prof. S. Srinivasan, IIT Madras, is as follows:-

Suppose A is the multiplicand and A = a [3:0]. B is multiplier and B = b [3:0]. And final product is P [7:0].

The diagram of 4 bit unsigned array multiplier is: -

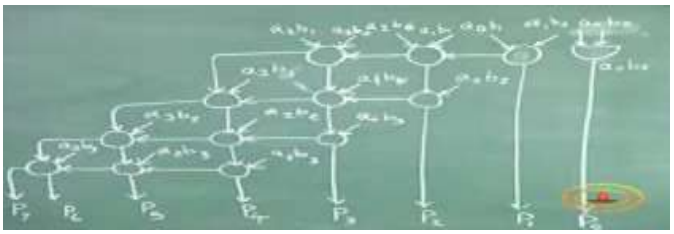

Figure 2 Four Bit Length Array Multiplication Algorithm

Hardware required for (M x N) multiplier is: -

- M x N logic and gates.
- N no. of half adders.
- (M-1) N total no. of adders.
- N (M-2) no. of full adders.

Advantage: - Regular structure.

Disadvantage: - Large no. of gates is required. Carry propagation path delay is the major issue.

3.4 Booth Multiplication Algorithm

The purpose of designing this multiplier is to allow the multiplication of two signed binary numbers in two's complement form. Booth's analysis led to conclude that an MAC Unit that could add/subtract could get the same result in more than one way.

For example 7 = 4+3

Alternatively 7 = 8-1; at this time shifting was faster than the addition. Hence, reducing the number of addition increased performance.

To perform booth multiplication we need some registers first is accumulator A. Next is multiplier Q. And Q_{-1}, which is a one bit register used to hold one bit during operation. Multiplicand is M. As many number of bits in the operands that many steps are required to get product. For example to get the product of 4x4 bit multiplier, four steps are required.

In first step compare Q [0] and Q_{-1} if both are equal then Arithmetic right shift operation occurs between the accumulator A, multiplier Q and Q_{-1}. If both are different then either addition (A+M) or subtraction (A-M) operation occurs. And the result is stored in the accumulator A. after this Arithmetic right shift operation occurs between the accumulator A, multiplier Q and register $Q_{-1.}$ This can be summarized in the tabular form: -

Q[0]	Q_{-1}	Operation performed
0	0	Right shift only
1	1	Right shift only
1	0	A<= A-M, then right shifting
0	1	A<= A+M, then right shifting

Table 1 Key Note For Booth Multiplication Algorithm

Advantages: -

- Handles both positive and negative numbers uniformly.
- Efficient when there are long number of 1's in the multiplier.
- Area efficient.

Disadvantages: -

- Average speed of algorithm is about the same as with normal multiplication algorithm.
- Strong dependency upon the last bit of multiplicand. This can sometimes generate the worst case. In worst case it operates at slower speed than normal multiplication algorithm.

3.5 Karatsuba Multiplier

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. Then as we know that any number can be represented with its base number. For example $(1432)_{10}$ can be represented as $14x10^2 + 32$. Similarly X and Y can be rewritten as: -

$X = X_1 B^m + X_{0.}$

$Y = Y_1 B^m + Y_{0.}$

Here B is the base number, and value of m is chosen in a proper form such that B^m should be greater than the X_0 and $Y_{0.}$

Now multiply these two numbers X and Y: -

$X*Y = (X_1 B^m + X_0)*(Y1 B^m + Y_0).$

$= Z_2B^{2m} + Z_1B^m + Z_0$

Where

$Z_2 = X_1Y_1$;

$Z_1 = ((X_1+X_0)* (X_1+X_0)) – Z_2– Z_0$;

$Z_0 = X_0Y_0$;

Then putting these values to the equation "$Z_2B^{2m} + Z_1B^m + Z_0$" we can get final product. Advantage of this algorithm is that its high speed. But it also has a limitation that value of Z_1 depends upon the value of Z_2 and Z_1. By overcoming this limitation we can further enhance the speed of this multiplier.

Illustration with example: -

$X = (1001)_2 = 10*2^2 + 01$;

$Y = (0110)_2 = 01*2^2 + 10$;

$Z_2 = X_1Y_1 = 10*01 = 10$;

$Z_0 = X_0Y_0 = 01*10 = 10$;

$Z_1 = ((X_1+X_0)* (X_1+X_0)) – Z_2– Z_0 = (10+01)(01+10) – 10 – 10 = 101$;

Final result = $Z_2B^{2m} + Z_1B^m + Z_0 = 10*2^4 + 101*2^2+10 = 110110$.

So, to overcome this limitation there is another way to find out the value of Z_1, is that we can use Vedic multiplication algorithm to get the value of Z_1. As shown in the following: -

$Z_1 = X_1Y_0 + X_0Y_1$;

And other things will be same as shown previously. If this way is used to find the value of Z_1 then it is called as the "Vedic Karatsuba multiplication algorithm".

3.6 Wallace Tree Multiplication Algorithm

This algorithm is an efficient hardwires implementation of a digital circuit that multiplies two numbers. This algorithm takes three steps, to achieve the outcome: -

1. Multiply each bit of one of the binary number, by each bit of the other, yielding n^2 results. Depending on position of the multiplied bits, the wires carry different weights.

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.

In more abstract way, 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.

Suppose A is the multiplicand and A = a [3:0]. B is multiplier and B = b [3:0]. Then the multiplication of A and B can be given as: -

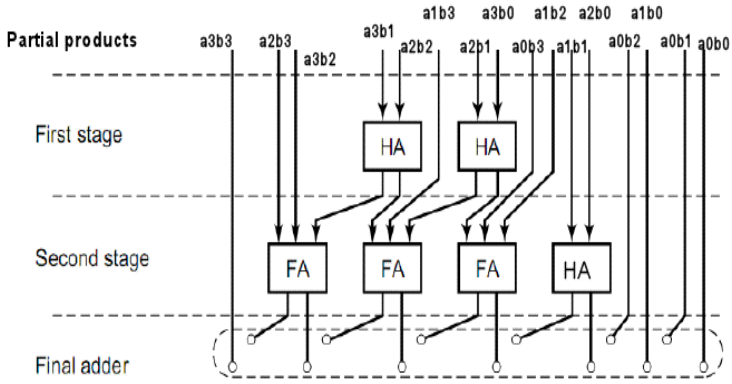

Figure 3 Wallace Tree Multiplier Block Diagram

Advantages: -

- Each layer of the tree reduces the number of vectors by a factor of 3:2.
- Minimum propagation delay.

The benefit of the Wallace tree is that there are only O (log n) reduction layers, but adding partial products with regular adders would require O(log n)2 time.

Disadvantages: -

- Wallace trees do not provide any advantage over ripple adder trees in many FPGAs.
- Due to the irregular routing, they may actually be slower and are certainly more difficult to route.
- Adder structure increases for increased bit multiplication.

3.7 Modified Booth Wallace Multiplication Algorithm

This multiplier architecture comprises of two architectures which are modified booth and Wallace tree. Modified booth increases the speed because it reduces partial products to half. Further the delay in multiplier can be reduced by using Wallace tree which has been discussed previously. But still there is possibility for enhancing the speed of this multiplier by designing the deep level pipeline architecture.

6. Proposed work: -

After a thorough study we have seen that Karatsuba multiplier provides better results among all other multipliers. So we have proposed to use this multiplier in our proposed MAC unit so that it can provide much better result in terms of speed.

4. Expected result:-

- Significant enhancement in the speed of MAC unit.

5. Application:-

- Multiplication-and-accumulate operations are very important and dominant for digital filters. Therefore, by the use of speed optimized MAC unit high-speed filtering can be achieved.

- MAC unit is also useful to optimize other processing typical for DSP applications.

This proposed MAC unit can be used in those places where speed is prominent factor in comparison to other constraints, such as Medical and Defense areas, safety critical operations. Apart from this it can be used in other places such as to design speed efficient digital signal processors, multimedia, graphics, radar equipment, and cryptology, high speed digital signal processing operations like as: Fast Fourier Transform, Convolution, etc. It can also be used to design the speed efficient Infinite Impulse Response Filter, Finite Impulse Response Filter.

IV. REFERENCES

[1] 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.

[2] KIAT SENG YEO, KAUSHIK ROY" LOW VOLTAGE LOW POWER VLSI SUBSYSTEMS", BOOK, P.P. 119-144, MC GRAW HILL PROFESSIONAL ENGINEERING, 2005.

[3] SAMEER PALNITKAR , "VERILOG HDL-A GUIDE TO DIGITAL DESIGN AND SYNTHESIS", BOOK, SUN SOFT PRESS,1996.

[4] STEPHEN D. BROWN, AND ZVONKOVRANESIC, "FUNDAMENTALS OF DIGITAL LOGIC WITH VERILOG DESIGN, 2ND EDITION,", BOOK MCGRAW HILL, JUNE, 2007.

[5] MORRIS MANO, "DIGITAL CIRCUITS AND SYSTEMS", BOOK, PRENTICE-HAL, LAST UPDATED ON-02/27/2011, 11:55.

[6] WAYNE WOLF, "COMPUTER AS COMPONENTS", 2ND EDITION, BOOK.

[7] HTTP://EN.WIKIPEDIA.ORG/WIKI/VERILOG

[8] 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

[9] VIRTEX-II PRO AND VIRTEX-II PRO X PLATFORM FPGAS: COMPLETE DATA SHEET, XILINX, DS083 (V5.0) JUNE 21, 2011.

[10] Swati Malik and Sangeeta Dhall, "Implementation of MAC unit using booth multiplier & ripple carry adder", International Journal of Applied Engineering Research, ISSN 0973-4562 Vol.7 No.11 (2012).

[11] Bannoth Anjinaik, Mr. Y.V.S. Durga Prasad, "A new method for implementation of high speed MAC Unit",ISSN No: 2348-4845 International Journal & Magazine of Engineering, Technology, Management and Research.

[12] Chinababu Vanama, M.Sumalatha, "Implementation of High Speed Modified Booth Multiplier and Accumulator (Mac) Unit", IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) e-ISSN: 2278-2834,p- ISSN: 2278-8735.Volume 8, Issue 5 (Nov. - Dec. 2013), PP 17-25.

[13]Harish Babu N, 2Rajeev Pankaj N, "Design of High Speed Mac Unit", © 2014 IJEDR | Volume 2, Issue 2 | ISSN: 2321-9939.

[14] Soniya, Suresh Kumar, "A Review of Different Type of Multipliers and Multiplier-Accumulator Unit", International Journal of Emerging Trends & Technology in Computer Science (IJETTCS) Volume 2, Issue 4, July – August 2013.

