International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 01 | Jan 2020 www.irjet.net p-ISSN: 2395-0072

# An Efficient Multiply Accumulate Unit Design using Vedic Mathematics Algorithm

**Amandeep Kaur[1], Balpreet Kaur[2], Rupinder Kaur[3]**

[1,2,3]*Assistant Professor, Dept. of CSE, BBSBEC, Fatehgarh Sahib*

---

**Abstract -** *A novel technique for digital multiplication is explained and used that is differs from conventional multiplication method using add and shift [4]. This is an implementation of modular design style where smaller blocks are used to design bigger blocks of NxN multiplier. Using this multiplier, a Multiply Accumulate Unit has been designed which shows less computation time for performing MAC operation.*

***Key Words***: Verilog, Digital Multiplier, Multiply Accumulate Unit

## 1. INTRODUCTION

Multiply-accumulate operation is one of the basic arithmetic operations extensively used in modern digital signal processing (DSP). Operations such as digital filtering, convolution and fast Fourier transform (FFT), requires high-performance multiply accumulate operations. A Digital Multiplier is the heart of the MAC unit. The performance of MAC unit greatly depends on the multiplier [2]. This paper presents a systematic design for fast and efficient MAC unit using Vedic Mathematics [1]. This paper is organized into two sections. Section 1 presents the design methodology for NxN bit Vedic Multiplier and in Section 2, the design of Multiply Accumulate Unit, using Vedic Multiplier is explained.

## 2. Vedic Multiplier

To design a NxN multiplier, we need a 2X2 multiplier as a building block.[3] Talking about 2X2 multiplier, lets take two inputs say A (a1,a0) &B(b1,b0) , each two bits wide and the result be S (s3,s2,s1,s0) which is 4 bits wide. During this multiplication, 4 partial products are generated due to vertical and crosswise multiplication which are a0b0, a0b1,a1b0 and a1b1. The partial product a0b0 directly passes on to the product S as s0, where as the LSB of sum of a0b1 and a1b0 forms s1. The carry bit is added to partial product a1b1 to determine s3 and s2 as MSB and LSB after addition respectively. 2 bit Multiplication by this method is shown in figure 1.

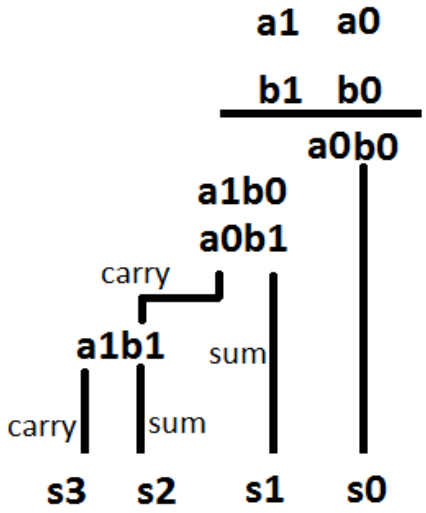


**Fig- 1:** Block Diagram of 2 bit Multiplier

Using the same technique as 2x2 bit vedic multiplier, 4,8,16 and 32 bit multipliers can be designed with slight modification. That is, the input bits need to be grouped into two halves of the input bit stream. Input bit steams ( of say N bits) can be divided into (N/2 = n) bit length, which can be further divided into n/2 bit streams and this can be continued till we reach bit streams of width 2, which can be multiplied in parallel, using 2x2 multiplier blocks[1]. Let us demonstrate this using a 4x4 multiplier which uses 4, 2x2 multipliers as its building blocks. Inputs A and B are assumed as 1101 and 1010 respectively which are grouped in chunks of 2 bits each and multiplied using 2x2 multipliers in parallel, as shown in figure.

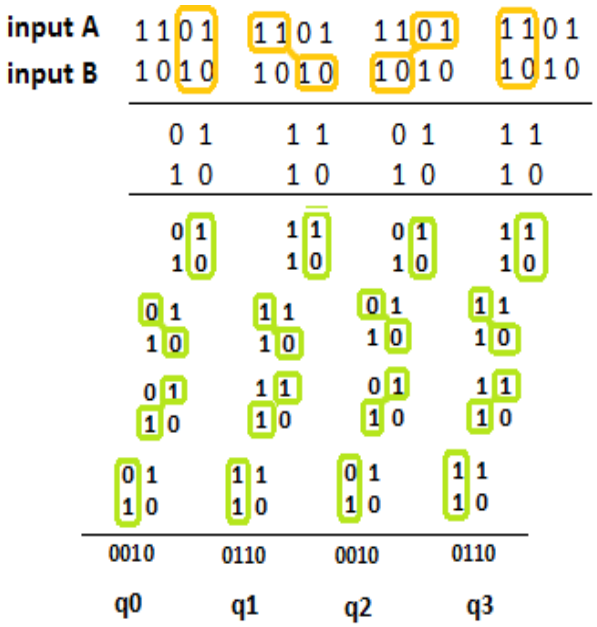


**Fig-2**: Using 2bit Multiplier for 4bit numbers

The output of 2x2 multipliers is q0,q1,q2,q3 which are 4 bits wide partial products. These are added using 3 full adders 1,2,3 and a half adder to get the final product S (s7,s6,s5,s4,s3,s2,s1,s0) as illustrated in Figure 3.

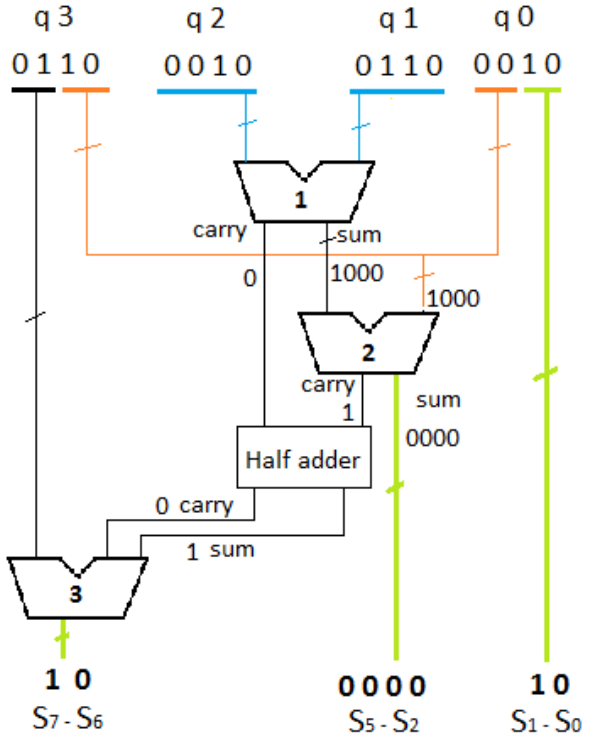


**Fig-3:** Addition tree for 4-bit Multiplier

Here, using Adder 1, q2 (0010) and q1(0110) are added to form 1000 as sum and 0 as carry bit. Pipelined carry look ahead adders should be used for addition here. The sum i.e. 1000, so formed is added to the bit stream formed by concatenation of 2 most significant bits of q0 i.e. 00 and 2 least significant bits of q3 i.e. 10 which, on concatenation forms 1000. These are added using Adder 2, which produces 0000 as sum and 1 as carry. The two least significant bits of q0 directly pass on to form two least significant bits of the final product i.e. S1,S0. The sum produced from Adder 2 i.e. 0000 forms S5,S4,S3,S2. The carry bits generated from Adder 1 and Adder 2 are added using a half adder to form a carry bit and sum bit, i.e. 0 & 1 respectively. These form MSB and LSB of one of the inputs for Adder 3, the other input being two MSBs of q3. The output from Adder 3 forms S7, S6.

Using the above method of Vedic Multiplication, a 16 bit Multiplier unit has been designed which is used in the 16 Bit Multipliy-Accumulate Unit. The 16x16 Multiplier is made by using 4 , 8x8 multiplier blocks. Here , the multiplicands are of bit size (n=16) where as the result is of 32 bit size. The input is broken into smaller chunks size of n/2 = 8, for both inputs, that is a and b. These newly formed chunks of 8 bits are given as input to 8x8 multiplier block, where again these new chunks are broken into even smaller chunks of size n/4 = 4 and fed to 4x4 multiply block, just as in case of 8x8 Multiply block. These are again divided in half, to get chunks of size 2bits, which are fed to 2x2 multiply block. The result produced, from output of 8x8 bit multiply block which is of 16 bits, are added using an addition tree using 3 full adders and a half adder. In case of 8 bit and 16 bit multiplier, extra zeroes need to be padded at MSB of one of the inputs for adder 3. For 8 bit multiplier, two zeroes will be padded and for 16 bit multiplier, 6 extra zeroes need to be padded. The addition tree of 16 bit Multiplier is shown below if figure 4.

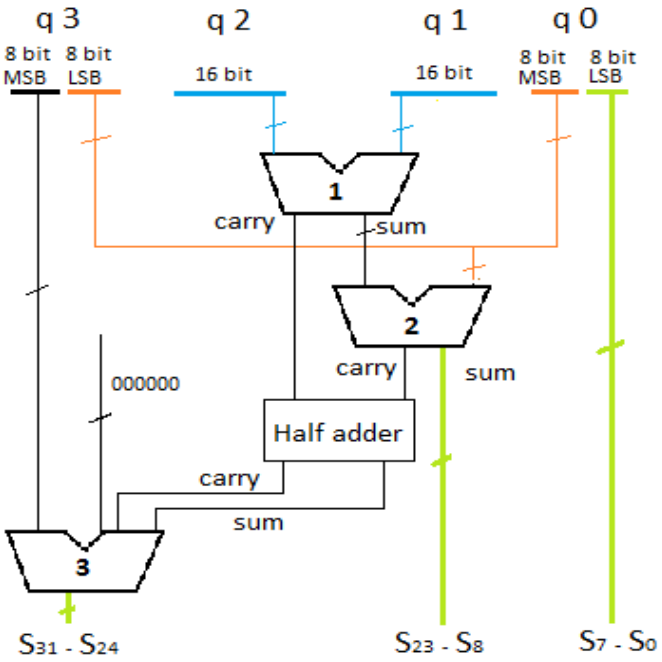


**Fig-4:** Addition tree for 16 bit Multiplier

## 3. MULTIPLY ACCUMULATE UNIT

The MAC unit employs a fast multiplier fitted in the data path and the multiplied output of multiplier is fed into a fast adder which is set to zero initially. The result of addition is stored in an accumulator register. The MAC unit produces output in one clock cycle and the new result of addition is added to the previous one and stored in the accumulator register. MAC architecture. Here the multiplier that has been used is a Vedic Multiplier built using Urdhva Tiryakbhyam Sutra. As, 16bit vedic multiplier is faster than other multipliers for example booth multiplier, even the MAC unit made using vedic multiplier is faster than the conventional one as well.

In the MAC unit, the data inputs, A and B are 16 bits wide and they are stored in two data registers, that is Data A and Data B, both being 16 bits wide, Then the inputs are fed into a Vedic multiplier, which stores the multiplied output in a 32 bit wide register named, Multiply out register. The contents of this register are continuously fed in to an adder and the result is stored in a 64 bit wide register "Data out register". For fast operation, adder should be fast and especially we have to use pipelined carry look-ahead adder [4].

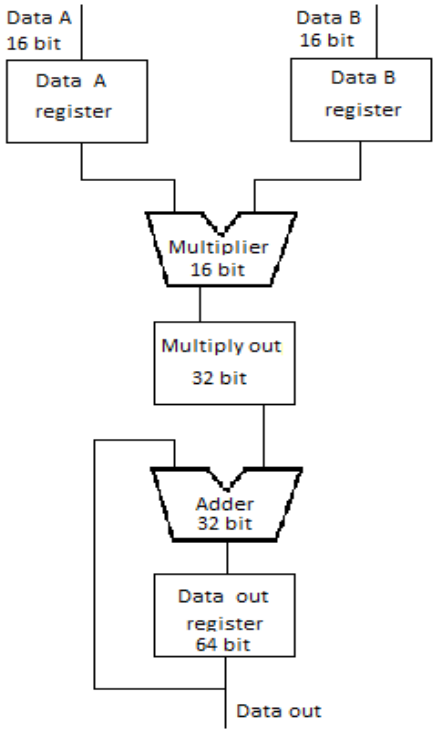


**Fig-5**: Block Diagram of 16 bit MAC unit

### 3.1. Simulation result

The 16 bit multiplier and 16 bit MAC unit have been designed in Verilog HDL and their working has been verified for all the possible input combinations by writing a test bench. Here the inputs are taken to be 1111111111111111 for both inputs.

The result obtained is as given below:

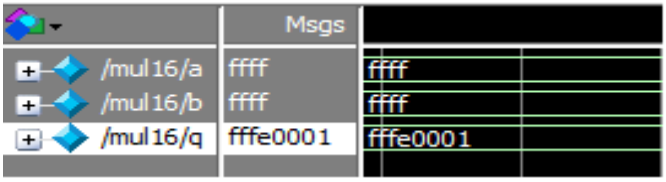


**Fig -6**: Simulation result of 16 bit multiplier

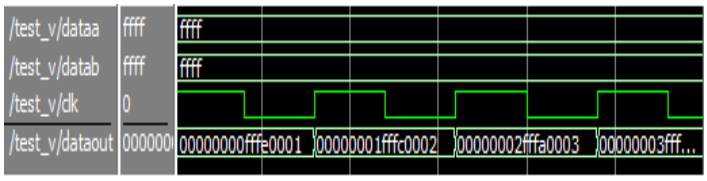


**Fig- 7**: Simulation result of 16 bit MAC unit

### 3.2 Synthesis Result

Synthesis was done using ISE 7.1. The device chosen for synthesis is XC3S500e-4fg320 with speed grade -4. Delay was measured after back annotation using Simprim library. The table given below gives circuit delay for the worst case input for the corresponding no. of bits in the input i.e. for N=16, worst case inputs were taken as 1111111111111111 at both A and B. The synthesis was done using Xilinx ISE software.

**Table -1:** Combinational Delay

| Component | Delay |
|---|---|
| 16 bit Multiplier | 9.334 ns |
| 16 bit MAC unit | 11.151 ns |

**Table -2:** Device Utilization for 16bit MAC Unit

| Device Utilization Summary | |
|---|---|
| Number of Slices | 384 |
| Number of LUTs | 712 |

**Table -3:** Power Dissipation for 16 bit MAC Unit

| Power Dissipation | |
|---|---|
| Frequency ( MHz) | Power (mW) |
| 20 | 90 |
| 50 | 188 |

## 4. CONCLUSION

In this paper, a Vedic method of multiplication is presented and used for MAC operation of 16 bit numbers. This gives us method for hierarchical multiplier design which has been used in MAC operation. So the design complexity reduces for large inputs and modularity increases which can be effectively used in various Digital Signal Processing applications.

## REFERENCES

1. Shamim Akhter, "VHDL Implementation of Fast NXN Multiplier Based on Vedic Mathematics", Jaypee Institute of Information Technology University, Noida, 201307 UP, INDIA, 2007 IEEE

2. Neil H.E Weste, David Harris, Ayan Banerjee,"CMOS VLSI Design, A Circuits and Systems Perspective", Third Edition, published by Person Education, PP-327-328]

3. Himanshu Thapliyal and Hamid R. Arabnia, "A Time Area Power Efficient Multiplier and Square Architecture Based On Ancient Indian Vedic Mathematics", Department of Computer Science, The University of Georgia, 415 Graduate Studies Research Center Athens, Georgia 30602-7404, U.S.A.

4. CRAWLEY, D., and AMARATUNGA, G:'pipelined carry look-ahead adder design', Electron. Lett., 1996,22, (12),pp. 661-662.