ISSN: 2321-7782 (Online)
Volume 2, Issue 10, October 2014
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Universal Floating Point Multiplier using Vedic Mathematics

Nithu S. Mangalath[1]
Dept. of E&TC Engineering
D.Y.Patil College of Engg.,Akurdi
Pune, India

R. Arokia Priya[2]
Prof., Dept. of E&TC Engineering
Dr.D.Y.Patil College of Engineering,Akurdi
Pune, India

Dr. P. Malathi[3]
HOD, Dept. of E&TC Engineering
D.Y.Patil College of Engineering,Akurdi
Pune, India

Abstract: ***Floating point multiplication is of key importance to many modern applications. These applications usually involve floating point calculations with single and/or double precision format. For this reason, most modern processors have hardware support for single precision and double precision floating point multiplication. Achieving this goal however, usually effects the throughput since most FPUs convert the single precision operands to double precision operands and then translate again the result to single precision format. This output and precision is inadequate for many scientific computations like modelling of climate conditions, processing of digital signals, graphic accelerators etc. All these higher level applications uses quadruple precision floating point arithmetic. The quadruple precision arithmetic specifications was included in the IEEE 754-2008 revised standard. Its precision is twice as compared to the double precision format. The design proposed in this paper performs all three precision multiplication operation. The universal floating point multiplier is implemented using Vedic Mathematics (Nikhilam Navatascaramam Dasatah Sutra).***

Keywords: Quadruple precision; Double precision; Single precision; Floating point multiplier; Floating point unit; Vedic Mathematics; Navatascaramam Dastah Sutra.

I. INTRODUCTION

In many modern applications such as 3D graphics accelerators, Digital Signal Processors (DSPs), High Performance Computing etc. is of key importance. High Performance Computing etc. is of key importance. Floating point calculations with single and double precision format are usually involved in these applications. Most Floating Point Units therefore tend to support both single and double precision floating point operations. Since most FPUs covert the single precision operands to double precision and then translate the result again to single precision format, it effects the throughput of the FPUs. Another drawback is that many of the existing designs execute floating point operations only sequentially. In the area of intense graphics applications and multimedia, parallel execution has become a necessity [6]. In many scientific fields like computational physics, computational geometry, climate modelling etc. there is a need for increased precision in floating point calculations. They require great accuracy and high precision calculation, features which double precision format is not able to provide. In such cases the quadruple precision arithmetic is used since it provides twice the precision of double precision format, improving the accuracy of calculations and leading to more reliable results. The quadruple precision arithmetic specifications has been included in the IEEE 754-2008 revised standard for floating point arithmetic. This paper presents a multi-mode floating point multiplier able to handle all three precision modes.

Several architectures which consist of the stand alone floating point multipliers can be found in the literature. In Dual precision IEEE floating point multiplier the latency of double precision was three clock cycles [1]. The architecture was based on ancient technique and did not support quadruple precision multiplication. In Quad-double precision floating point arithmetic the time complexity was more and did not support the single precision multiplication [2]. The Flexible multiplier for media

processing was limited for media processing applications [3]. The dual-mode multiplier had separate architectures for dual-mode quad and dual-mode double precision multiplication [4]. The multiplier block in a Floating Point Unit is the most complex block. The classical or long multiplication method is the simplest method to multiply two numbers. Whereas for addition or subtraction of two n-bit numbers requires n number of addition or subtraction operation. Several algorithms have been proposed to decrase the delay in multiplication operation. The simplest one is the divide and conquer algorithm. The Karatsuba algorithm is also based on divide and conquer algorithm. A two-digit multiplication using Karatsuba algorithm can be done in three multiplication operation instead of four. For large numbers this algorithm can be applied recursively. Nikhilam Sutra is a multiplication algorithm based on Vedic Mathematics. A two-digit multiplication operation using Nikhilam Sutra can be performed using only one multiplication operation instead of four.multiplication operation. In the proposed technique all the three precisions are included within a single architecture and the multiplier is designed using Vedic mathematics.

II. Floating Point Multiplication Algorithm

Floating point numbers which are normalized have the form of Z= (-1S) * 2 (E - *Bias*) * (1.M). Significant is the mantissa with an extra MSB bit. The following steps are carried out to multiply two floating point numbers.:

1. The significands are multiplied.
2. The decimal point is placed in the result.
3. The exponents are added.
4. The sign bit is obtained by XOR operation of S_1 and S_2; i.e. S_1 XOR S_2.
5. The final result is normalized i.e. obtaining '1' at the MSB of the results significand.
6. Rounding operation is performed on the result to fit in the available bits.

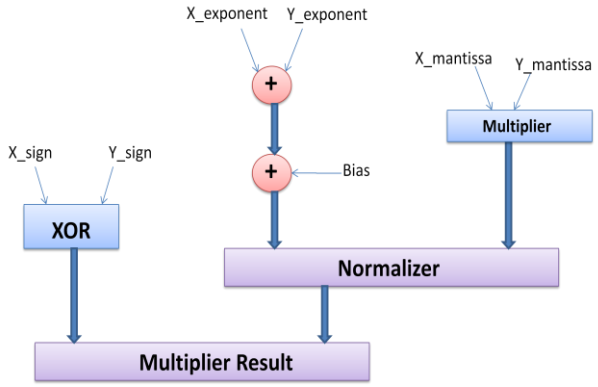

Fig. 1 Block diagram of Floating Point Multiplier

III. "Nikhilam Navatascaramam Dasatah" Sutra

The Nikhilam Sutra can be translated as: "All from 9 and the last from 10" Jagadguru Swami Sri Bharati Krsna Tirthaji Maharaja claims that this Sutra cryptically explains how to perform multiplications of numbers above 5 without previous knowledge of the higher multiplications of the multiplication tables. It is one of the sixteen sutras of Vedic mathematics. With the help of few extra add, subtract and shift operations, it can be used to convert large-digit multiplication to small-digits multiplication. For two digit multiplication, the Karatsuba algorithm requires three one digit multiplication. While using Nikhilam Sutra only one digit multiplication is required.

For example, the two numbers 106 and 107 are to be multiplied. The procedure is as follows:

1. The nearest base is subtracted from the multiplicand: P = 106 – 100 = 6
2. The nearest base is subtracted from the multiplier: Q = 107 – 100 =7

3. Compute: R = P * Q = 6 * 7 = 42
4. Compute: S = 106 + 7 = 107 + 6 = 113
5. Result: 100 * S + R = 11342

For the above three digit multiplication, the standard method will take nine multiplications and the Karatsuba method will take four multiplication operations. The principle behind this method is as follows:

Let the numbers to be multiplied be a and b and x is the nearest base to both these numbers such that:

$$a = x + m$$

$$b = x + n$$

Then according to Nikhilam Navatascaramam Dasatah Sutra:

$$a * b = (x + m) * (x + n) = x (a + n) + mn$$

Table I
Multiplication of a * b

	Integer	Base Difference
Multiplicand	A	(x + m) – x
Multiplier	B	(x + n) – x
	x + m + n = (a + n)	mn
Result	x (a + n) + mn	

A. Binary Multiplication

Binary digit multiplication using Nikhilam Stra can be preformed by converting n-bit multiplication to (n - 1) bit multiplication and some additional add/subtract and shift operation. For 3 –bit multiplication consider the following example: 110 * 101

Table II
Multiplication of 110 * 101

	Integer	Base Difference
Multiplicand	110	110 – 100 = 10
Multiplier	101	101 – 100 = 1
	110 + 1 + 111	1 * 10 = 10
Result	11110	

In case of standard multiplication, for above multiplication nine multiplications are required, Karatsuba requires four multiplication whereas the above example shows that Nikhilam method requires only two one bit multiplication. Fig. 1 shows the hardware implementation of Nikhilam Sutra.

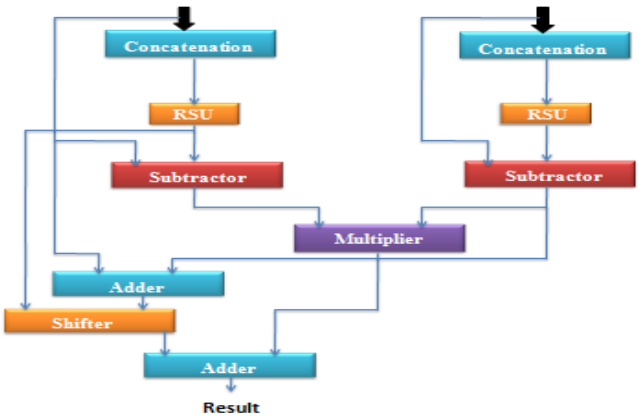

Fig. 2 Hardware Implementation of Nikhilam Sutra

IV. UNIVERSAL MULTIPLIER USING NIKHILAM SUTRA

The multi-precision floating point multiplier proposed in this paper supports the entire precision format specified by the IEEE 754-2008 standard. For storing the different precision operands two registers are used. The three precision formats standardized by IEEE are: Single Precision format, Double Precision format and Quadruple Precision format Floating Point Numbers. The single precision floating point number has one sign bit, 8 exponent bits and 23 mantissa bits. The double precision number has one sign bit, 11 exponent bits and 52 mantissa bits. The quadruple precision number has one sign bit, 15 exponent bits and 112 mantissa bits. More bits in the exponent field increases range of values. And more bits in mantissa field increases the precision of floating point numbers. A control signal is used to control the operands of the design and it also specifies the operation mode. The algorithm used for multiplication in this design is based on Vedic Mathematics. This technique is used to perform high precision multiplications. In this technique a series of smaller sized multiplications are performed initially and then these partial products are added. The architecture of the proposed system is shown in Fig 3. The binary numbers A and B are given to the floating point number extractor. This block extracts the sign bit, exponent bit and mantissa bit according to the mode selected. Then according to the algorithm the XOR operation is performed for sign bits, exponent bits are added and biased. The mantissa part of both the number are given to the multiplier block which works on the principle of Vedic mathematics. The partial products and sum are generated in one step and which reduces the carry propagation from LSB to MSB. The propagation delay of the multiplier is reduced as compared to the complex tradition multipliers.

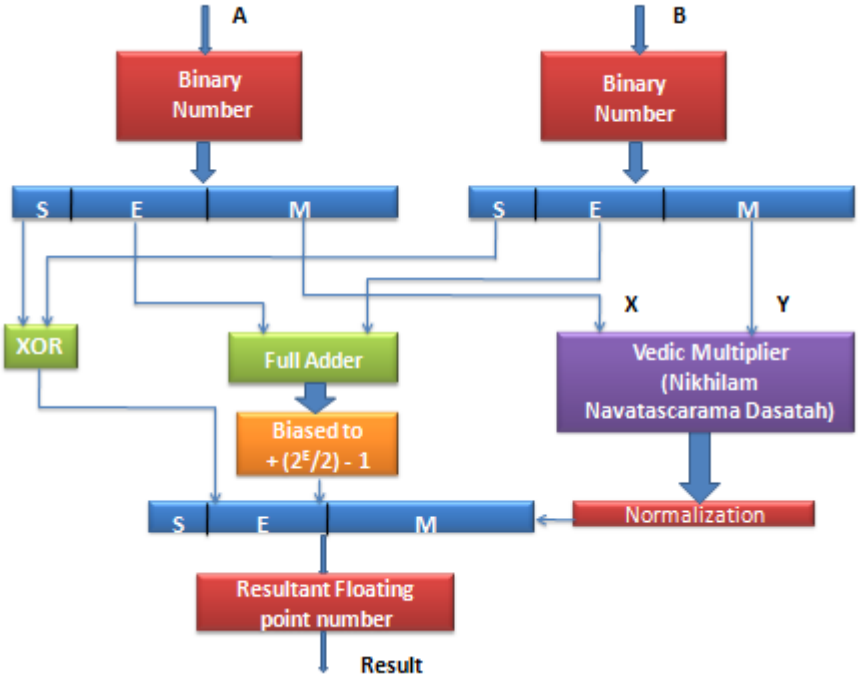

Fig. 3 Block Diagram of Universal Multiplier

V. RESULT AND SIMULATION

The multi-precision floating point multiplier presented in this paper has been designed and implemented in VHDL. The efficiency of this multiplier can be shown by considering the typical solutions of different precision floating point multipliers. A 113-bit array multiplier is used by a typical quadruple precision multiplier while our approach requires only half the area [3]. Traditionally two conventional double precision multipliers were used simultaneously for high throughput, while the proposed design executes double precision as well as quadruple precision multiplication. In our case we are implementing three different precision formats as compared to conventional multipliers which supports only one of the precisions. Therefore, though the

design utilizes slightly more area, it still provides an efficient solution to cases demanding high throughput and multiple precision operations. Fig. 4, 5 and 6 shows the test-bench waveform for single, double and quadruple precision multiplication.

For 32 bit multiplier:

A = 1 10000011 00100000000000000000000

B = 0 10000010 00110000000000000000000

Y = 1 10000110 01010110000000000000000

For 64 bit multiplier:

A = 1 10000000011 0010000000000000000000000000000000000000000000000000

B = 0 10000000010 0011000000000000000000000000000000000000000000000000

Y = 1 10000000110 0101011000000000000000000000000000000000000000000000

For 128 bit multiplier:

A = 1 100000000000011 00100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000

B = 0 100000000000010 00110000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000

Y = 1 100000000000110 01010110000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000

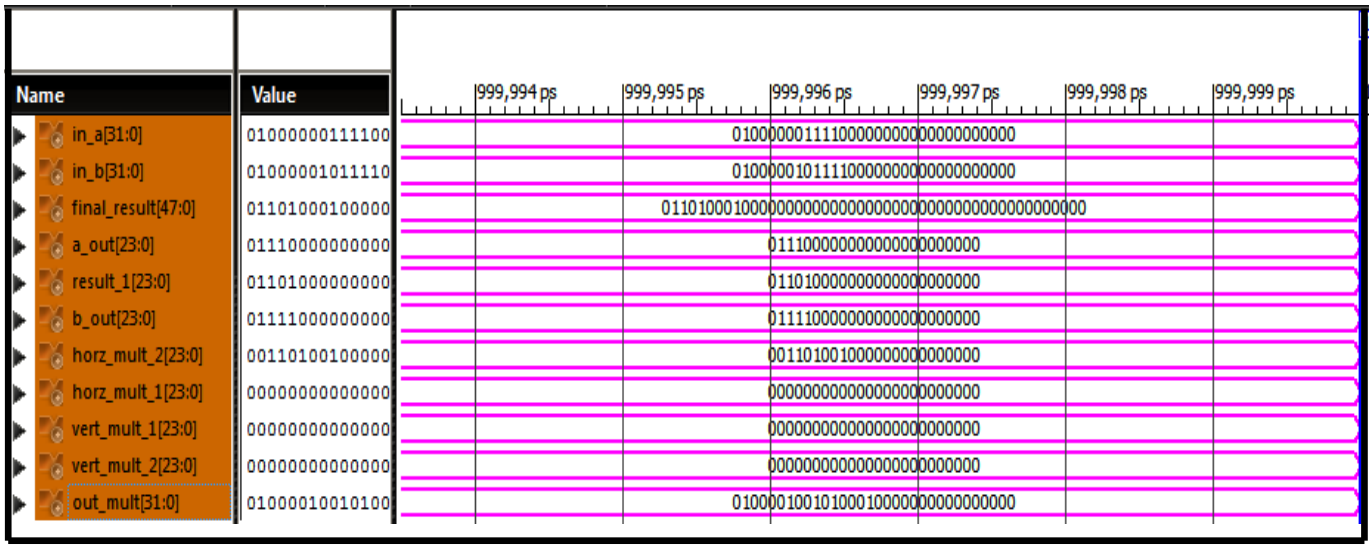

Fig. 4 : Test-bench waveform for 32-bit multiplier

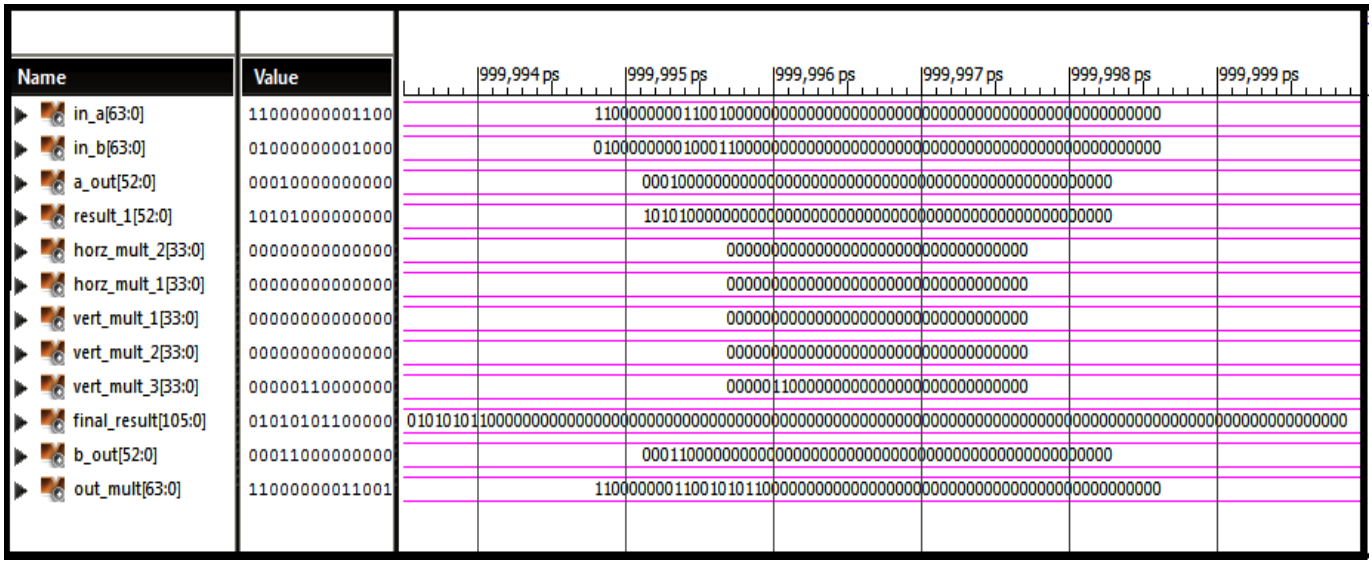

Fig. 5 : Test-bench waveform for 64-bit multiplier

Table III
Maximum combinational path delay of different multipliers

	Array Multiplier	Booth Multiplier	Ref. 9	Ref. 10	Proposed Multiplier
8 x 8	32.01ns	29.549ns	24.16ns	---	18.620ns
16 x 16	60.928ns	70.809ns	36.563ns	23.87ns	19.436ns

Table IV
Comparison of different double precision floating point multiplication

	Conventional Multiplier	Ref. 11	Proposed Multiplier
Double Precision Floating Point Multiplication	326.756ns	44.565ns	22.37ns

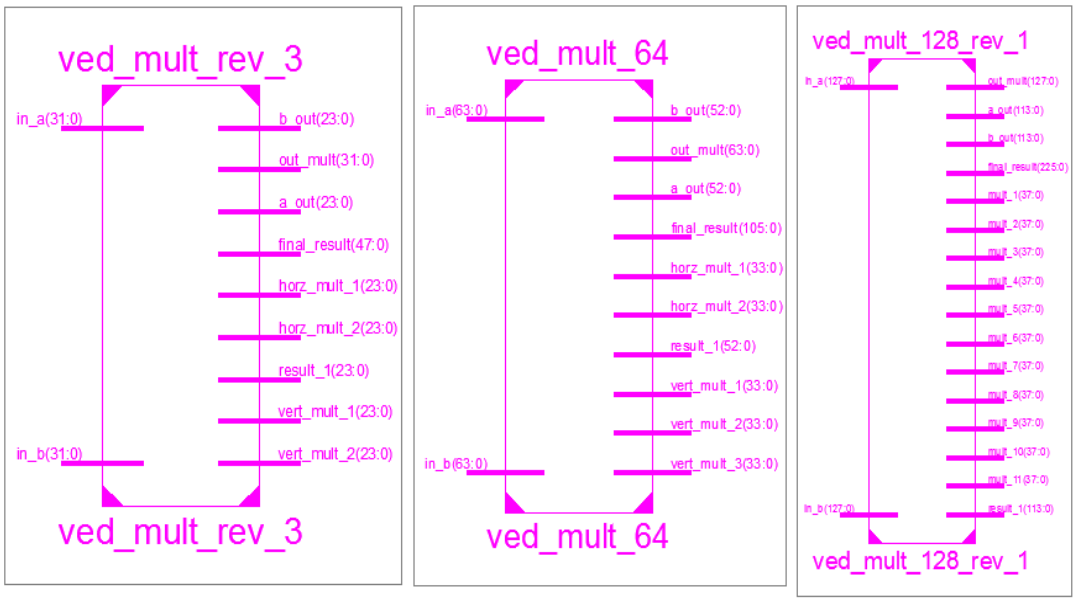

Fig. 6: RTL schematic for single, double and quadruple precision multipliers

ACKNOWLEDGMENT

I express my sincere gratitude towards the faculty members who made this project successful. I would like to express my thanks to my guide and PG coordinator Prof.Mrs. R. Arokia Priya for her whole hearted co-operation and valuable suggestions, technical guidance throughout the project work. Special thanks to our H.O.D. Dr.Mrs. P.Malathi for her kind official support given and encouragement. Finally, I would like to thank all our faculty members of E&TC Department who helped me directly or indirectly to complete this work successfully.

References

1. G.Even, S.Mueller, P.M Seidel, "*A dual precision IEEE floating point multiplier*", Integration, the VLSI Journal, Volume 29, Issue 2, pp 167-180, September 2000.
2. Y.Hida, X.S.Li, D.H.Bailey, "*Algorithms for quad-double precision floating point arithmetic*" in Proc. of 15th IEEE Symposium on Computer Arithmetic 2001, pp 155-162.
3. Claudio Brunelli, P.Salmela, J.Takala and Jari Nurmi,"*A flexible multiplier for media processing*", in Proc. of IEEE Symposium on Computer Arithmetic 2005.
4. G.Govindu, L.Zhuo, S.Choi, V.Prasanna,"*Analysis of high performance floating-point arithmetic on FPGAs*", in Proc. of 18th International parallel and distributed processing symposium, pp 149-156, April 2004.
5. A.Akkas, Michael Schulte, "*Dual-mode floating point multiplier architectures with parallel operations*", Journal of systems architecture: EUROMICRO Journal archive, Volume 52, Issue 10, pp 549-562, Oct.2006
6. Manolopoulos. K; Reisis.D.; Chouliaras.V.A, "*An efficient multiple floating point multiplier*", Electronics, Circuits and Systems (ICECS),2011 18th IEEE International Conference on, vol.,no.,pp.153,156, 11-14 Dec.2011.
7. IEEE Computer Society "IEEE Standard for Floating-Point Arithmetic", IEEE Std 754-2008.
8. Tan.D.; Lemonds, C.E; Schulte.M.J, "*Low-power multiple-precision iterative floating point multiplier with SIMD support*", Computers, IEEE Transactions, vol.58,no.2,pp.175-187, Feb.2009
9. S.S.Kerur, Prakash Narchi, Jayashree C.N., Harish M Kittur, Girish V.A, "*Implementation of Vedic Multiplier for Digital Signal Processing*", International Conference on VLSI, Communication & Instrumentation (ICVCI) 2011.
10. C.Sheshavali, K.Niranjan Kumar, "Design and Implementation of Vedic Multiplier", Int. Journal of Engineering Research and Development, Volume.8, Issue 6, pp.23-18,, Sept.2013
11. P.D Kale, M.N Thakre, R.N Mandavgane, "Design of Double PrecisionFloating Point Multiplier using Vedic Multiplication", IJARCSSE, volume 4, issue 7, July 2014.
12. http://www.xilinx.com/support/documentation/data sheets/ds312.pdf

AUTHOR(S) PROFILE

Nithu Mangalath, received the BE degree in Electronics and Telecommunication Engineering from Siddhant College of Engineering in 2012. Currently, she is pursuing ME in VLSI and Embedded Systems from D.Y Patil College of Engineering and is working on the project titled "Universal Floating Point Multiplier" for final year dissertation.

