Low-Power and Area-Efficient Design of Higher-Order Floating-Point Multipliers Using Vedic Mathematics

Haripriya Loganathan, Patnaikuni Rohit, Polamarasetty Sai Suneel and Karthi Balasubramanian

Abstract Floating-point arithmetic units form the backbone of the state-of-the-art digital signal processing algorithms. Low power and area efficient design is always a key requirement for applications that use these algorithms. This requirement is more relevant for computationally intensive jobs that use higher-order multipliers. This paper attempts to study the possibility of addressing this issue using vedic arithmetic based floating-point unit. Vedic mathematics is an ancient Indian mathematics system that has come back to prominence in the last century. In this paper, we design a IEEE 754 single precision floating-point multiplier with the integer multiplication being carried out in a vedic mathematics style using different sutras. Nikhilam and Urdhva Tiryagbhyam sutras and their combination are used to design the same. This implementation is compared with conventional implementations using Booth and array multipliers. The designs are simulated using Verilog and synthesized using gpdk 90 nm technology. The results show that vedic multiplier based design gives competing results for multipliers of larger sizes. Low power and area efficient design is achieved for higher order multipliers when the design is based on the combination of Nikhilam and Urdhva Tiryagbhyam sutras. Thus for DSP applications using large multipliers, it is envisaged this approach of vedic multiplier design would lead to more efficient system implementations.

Keywords Floating-point unit · IEEE 754 standard · Vedic multiplier · Urdhva Tiryagbhyam sutra · Nikhilam sutra

1 Introduction

Demand for high speed and low power computations has been steadily increasing for various digital signal processing (DSP) systems [1]. Floating-point multipliers (FPM) are widely used in DSP systems and the efficiency of these DSP systems rely heavily on the constituent FPM. Due to the complexity of the arithmetic involved

H. Loganathan · P. Rohit · P. S. Suneel · K. Balasubramanian (✉)
Department of Electronics and Communication Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India
e-mail: b_karthi@cb.amrita.edu

T. Hitendra Sarma et al. (eds.), *Emerging Trends in Electrical, Communications, and Information Technologies*, Lecture Notes in Electrical Engineering 569,
https://doi.org/10.1007/978-981-13-8942-9_39

in analyzing floating-point numbers, FPMs are generally area and power hungry devices. Hence, it becomes imperative to look at design solutions for FPMs that have low power and area especially for multipliers of larger sizes.

Floating-point multiplier uses adders, shifters and integer multipliers to perform floating-point multiplication [2–4]. The design of the integer multiplier block is one of the key components in the FPM design. Traditionally, array multipliers and Booth multipliers [5, 6] have been used by researchers for designing the integer multiplier in FPMs [7, 8]. Array and Booth multipliers are fast but they consume large area and power respectively.

One alternative that is less explored is the use of Vedic multipliers to do the same. Vedic multipliers are generally designed using two sutras (aphorisms) namely Urdhva Tiryagbhyam and Nikhilam. Tiwari et al. in [9], Kanhe et al. in [10] and Havaldar et al. in [11] have used Urdhva Tiryagbhyam sutra for the multiplier implementation while Patel et al. in [12] and Budhiraja in [13] have designed both Urdhva Tiryagbhyam and Nikhilam based multipliers. An example of a vedic mathematics based multiplier-accumulator block is given in [14]. Works like these have analyzed and shown the implementation results for different multiplier sizes using the above two sutras independently but they have not explored the possibility of using both the sutras in one design itself.

The hardware structure of the Urdhva Tiryagbhyam based design is very similar to an array multiplier that requires multiple adders at the final stage of calculation. Thus it becomes less efficient while dealing with large numbers. Nikhilam sutra based design is more effective for large numbers but its efficiency is fully harnessed only when the numbers being multiplied are close to a reference value. This paper is aimed at designing an integer multiplier by using the combination of both the sutras and analyzing the area and power of the resultant floating-point multiplier.

It has been also noticed that a relative analysis of the effect of multiplier size among the different implementations is missing in most of the published work. This paper analyzes the area and power overhead for different multiplier sizes for both the vedic and the conventional methods and shows how the vedic multiplier out-performs the conventional methods, especially for large multiplier sizes.

This paper is organized as follows. Section 2 introduces the IEEE 754 floating point representation that is followed by the description of the floating-point multiplication algorithm in the subsequent section. Vedic multiplication is introduced in Sect. 4 where the various sutras and the multiplier design based on those are discussed. Section 5 details the reader with the results and discussions and the paper concludes in Sect. 5.

2 IEEE 754 Standard for Floating-Point Representation

Floating-point number representation is a method to represent a wide range of real numbers using limited number of bits. The decimal point is not fixed and is made to float i.e., the decimal point can be located anywhere with respect to the significant numeral of the specified number. Floating-point numbers are represented using IEEE

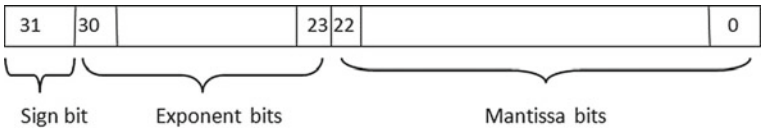

Fig. 1 IEEE 754 standard of a single precision floating-point number

754 standard and can be either based on single or double precision methodology. Floating-point numbers are represented using significant (mantissa), the base and an exponent. Single precision involves the use of 32 bits with 8 bits for the exponent and 23 bits for the mantissa with one bit as the sign bit [15]. Figure 1 shows the single precision floating- point representation.

Apart from this, the standard also describes double and quadruple precision formats that use 62 and 128 bits for representation respectively [16].

3 Floating-Point Multiplication

Figure 2 shows the flowchart used for the design of the floating-point multiplication unit for multiplying 2 numbers *M* and *N*.

The main component is the integer multiplier that is designed using multiple techniques including Booth recording, array multiplication with carry look-ahead

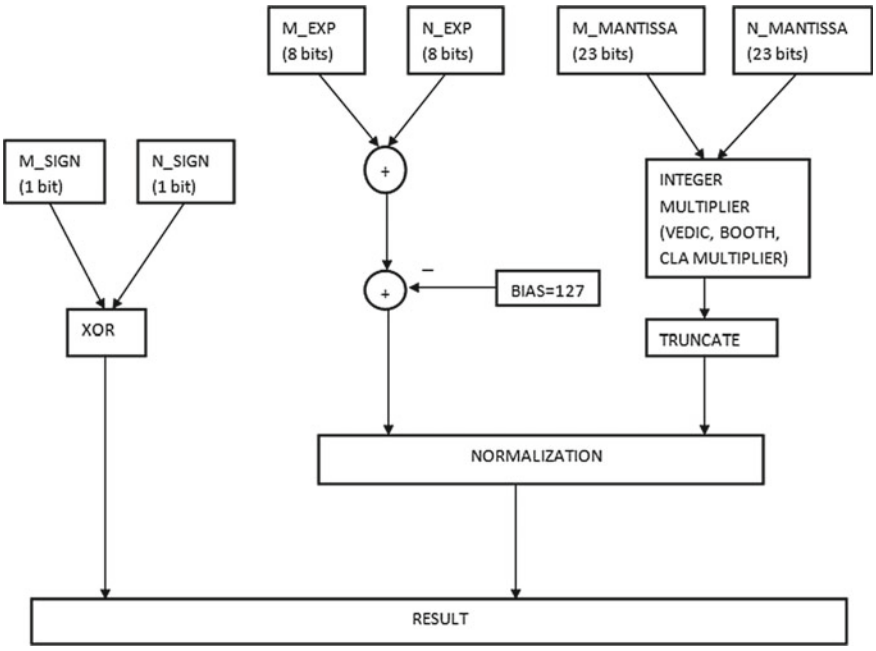

Fig. 2 Single precision floating-point multiplier structure

adder and vedic sutras. Booth multiplier and carry-look ahead adder based array multiplications are designed using conventional methods [5, 6]. The reader is now guided through the design of the vedic sutras based technique in the following section.

4 Vedic Multiplier

Vedic sutras (aphorisms)

In vedic mathematics, all the mathematical principles are presented in the form of aphorisms known as sutras. There are sixteen fundamental sutras that cover all the branches of mathematics [17]. The list of these sutras along with their meanings are elucidated in [9] and is being reproduced here. The sutras are as follows:

- (Anurupye) Shunyamanyat
- Chalana-Kalanabyham
- Ekadhikena Purvena
- Ekanyunena Purvena
- Gunakasamuchyah
- Gunitasamuchyah
- Nikhilam Navatashcaramam Dashatah
- Paravartya Yojayet
- Puranapuranabhyam
- Sankalana-vyavakalanabhyam
- Shesanyankena Charamena
- Shunyam Saamyasamuccaye
- Sopaantyadvayamantyam
- Urdhva-Tiryagbhyam
- Vyashtisamanstih
- Yaavadunam

In this work, we use the Urdhva Tiryagbhyam and the Nikhilam Sutras for the design of our integer multiplier.

4.1 Urdhva Tiryagbhyam Sutra

Urdhva Tiryagbhyam, that translates to 'Vertically and Crosswise', is a multiplication formula that can be used for numbers in any base. Figure 3 shows an example for multiplying two 3-bit numbers using this sutra. The same methodology can be extended for multiplication of larger numbers also.

Figure 4 shows the gate level implementation of a Urdhva Tiryagbhyam sutra based 2 * 2 multiplier and Fig. 5 shows the hierarchical design of a 4 * 4 multiplier

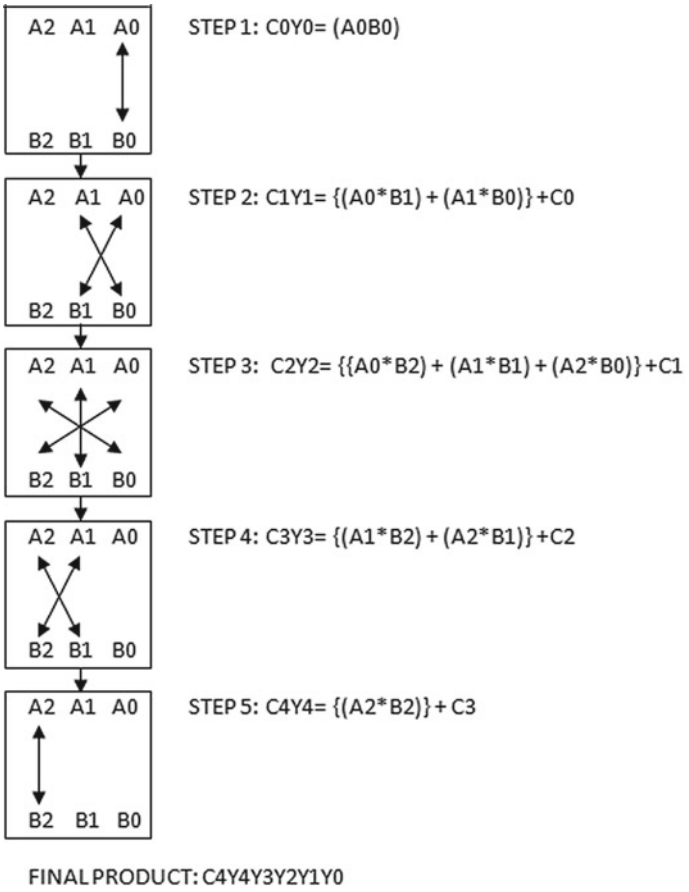

Fig. 3 An example for Urdhva Tiryagbhyam sutra

using 2 * 2 component multipliers. In the same manner, higher order multipliers can be built using the corresponding lower level component multipliers.

Similar to array multipliers, it can be seen that the partial products are generated in parallel and not in a sequential manner which makes it relatively fast since the delay associated is mainly the carry propagation delay through the adders in the array. However, this design is not very efficient while dealing with large numbers since the size of the adder array increases and carry propagation logic suffers from large delay [9]. To overcome this, carry look ahead adder may be used in the final stage but that will come with higher area penalty.

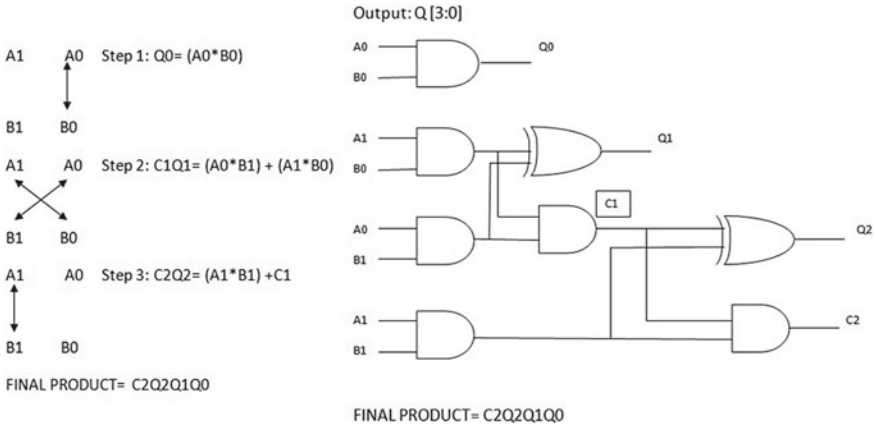

Fig. 4 Design of 2 * 2 vedic multiplier using Urdhva Tiryagbhyam sutra

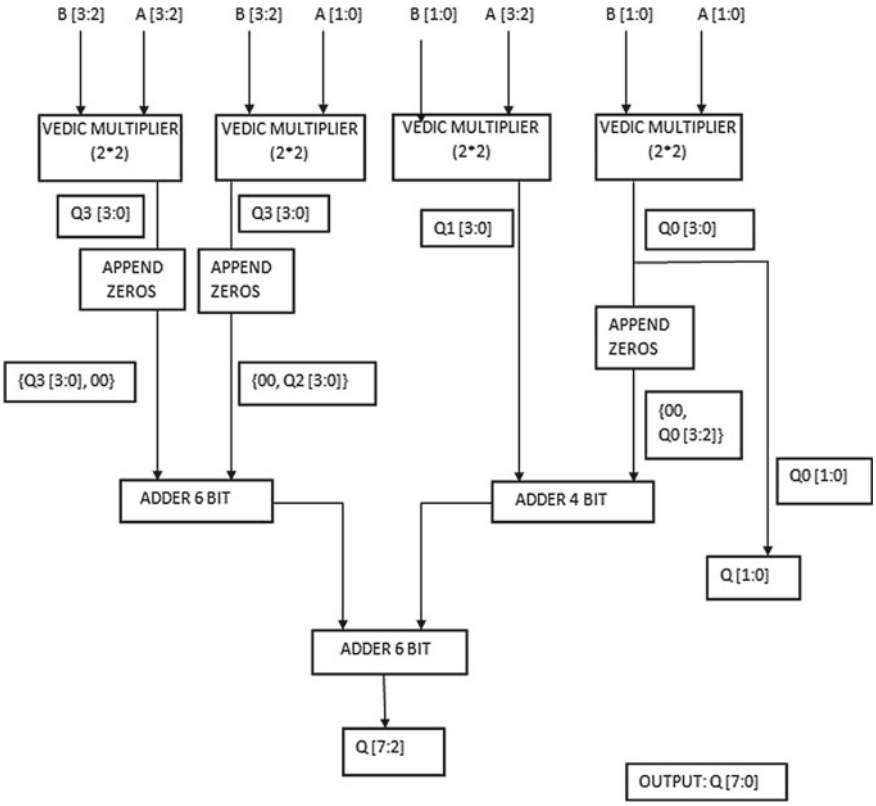

Fig. 5 Design of hierarchical 4 * 4 vedic multiplier using Urdhva Tiryagbhyam sutra based 2 * 2 component multipliers

4.2 Nikhilam Sutra

Nikhilam sutra is an alternative sutra that is commonly used for multiplying large numbers. This sutra can be used for multiplying all numbers, however its effectiveness is more while dealing with large numbers. Figure 6 runs through an example using two numbers G and H. In this example, the numbers are in base 10 (decimal) and the common reference is taken as some power of the base such that both the numbers are either lesser or greater than the common reference. Here, both numbers are less than 10^2 and so we take the reference as 100.

Figure 7 shows the generalized algorithm for a 16 * 16 multiplier. In this case, since we deal with 16 bit binary numbers we take the reference as 2^{16}.

Effectiveness of Nikhilam sutra: Nikhilam sutra analysis involves calculating the complements of numbers from the common reference. A detailed hardware implementation of a multiplier using Nikhilam sutra has been performed by Patel et al. in [12] and they show how Nikhilam sutra implementation is more effective than the

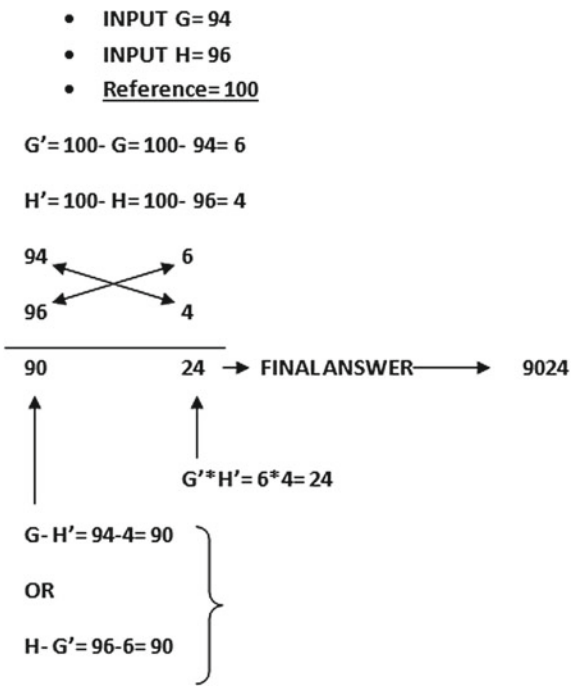

Fig. 6 An example for Nikhilam sutra

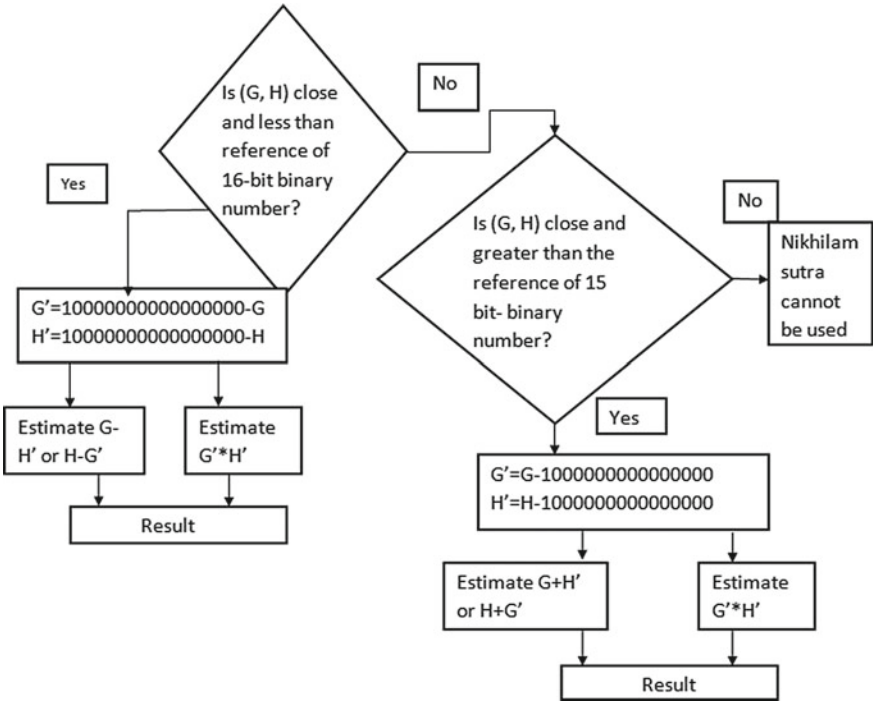

Fig. 7 Algorithm for 16 bit multiplication using Nikhilam sutra

Urdhva Tiryagbhyam sutra. However, they also point out that this is valid only when the numbers are close to the reference value.

4.3 Fusion of Nikhilam and Urdhva Tiryagbhyam Sutras

To overcome the limitations of Urdhva Tiryagbhyam and Nikhilam sutras based multiplier designs, an implementation that combines both the sutras at different levels of hierarchy is being proposed in this paper. This combined design is produced by modifying the Urdhva Tiryagbhyam based design by using Nikhilam based multipliers as the sub-modules for lower level multiplications. For e.g., a combined implementation of the 4 * 4 design is got by replacing the 2 * 2 multipliers in Fig. 5 with the Nikhilam equivalent multiplier. Similarly, for 8 * 8, 16 * 16 and 32 * 32 multipliers, the underlying sub-multipliers are designed using 4 * 4, 8 * 8 and 16 * 16 Nikhilam based designs respectively. Figure 8 shows the block level design of a 32 * 32 multiplier designed by this methodology.

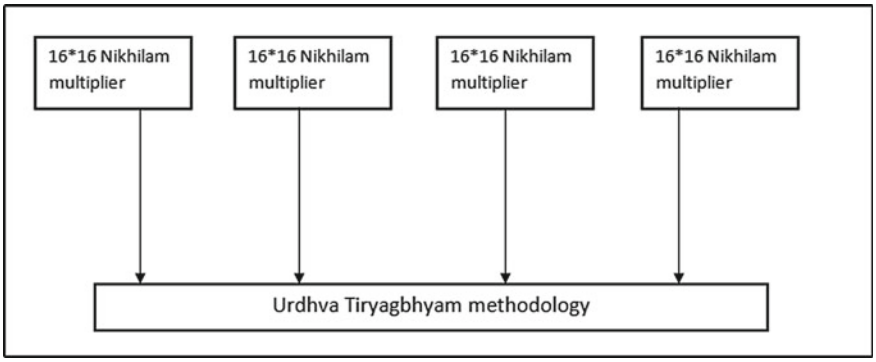

Fig. 8 Hierarchical structure of a 32 bit vedic multiplier using Nikhilam-Urdhva Tiryagbhyam combination

5 Results and Discussions

The following multipliers were designed and performance was analyzed.

- Conventional Multipliers
  - Array multiplier with carry look ahead adder
  - Booth multiplier
- Vedic Multipliers using
  - Urdhva Tiryagbhyam sutra
  - Nikhilam sutra
  - Nikhilam-Urdhva Tiryagbhyam combination

Multipliers of sizes 2 * 2, 4 * 4, 8 * 8, 16 * 16 and 32 * 32 were designed using Verilog and synthesized using gpdk 90 nm technology in Synopsys. Total power consumed and the area utilized were calculated and Tables 1 and 2 show the results for all the cases.

Table 1 Power (μW) dissipated for multipliers of different orders

Multiplier order	Multiplier type				
	Array	Booth	Urdhva Tiryagbhyam	Nikhilam	Urdhva Tiryagbhyam + Nikhilam
4 * 4	01.12	1.75	00.48	02.73	02.02
8 * 8	08.07	08.59	02.80	07.89	02.46
16 * 16	49.83	48.05	14.11	40.835	08.87
32 * 32	275.49	263.60	62.96	197.75	39.77

Table 2 Area (μm^2) utilized for multipliers of different sizes

Multiplier order	Multiplier type				
	Array	Booth	Urdhva Tiryagbhyam	Nikhilam	Urdhva Tiryagbhyam + Nikhilam
4 * 4	17.27	21.77	10.75	42.61	33.19
8 * 8	79.29	80.01	52.53	103.86	48.41
16 * 16	353.00	328.48	247.92	438.73	139.40
32 * 32	1577.41	1335.19	1167.35	1756.44	456.00

The same results are plotted graphically and given in Figs. 9 and 10 showing the power and area for the different order multipliers for each of the five different implementations.

It can be seen that as the multiplier size increases, there is a rapid increase in the power and area of the conventional multipliers while the increase is gradual for the vedic multipliers (one exception being the area of the Nikhilam based design). Among the vedic multipliers, Nikhilam and Urdhva Tiryagbhyam based designs by

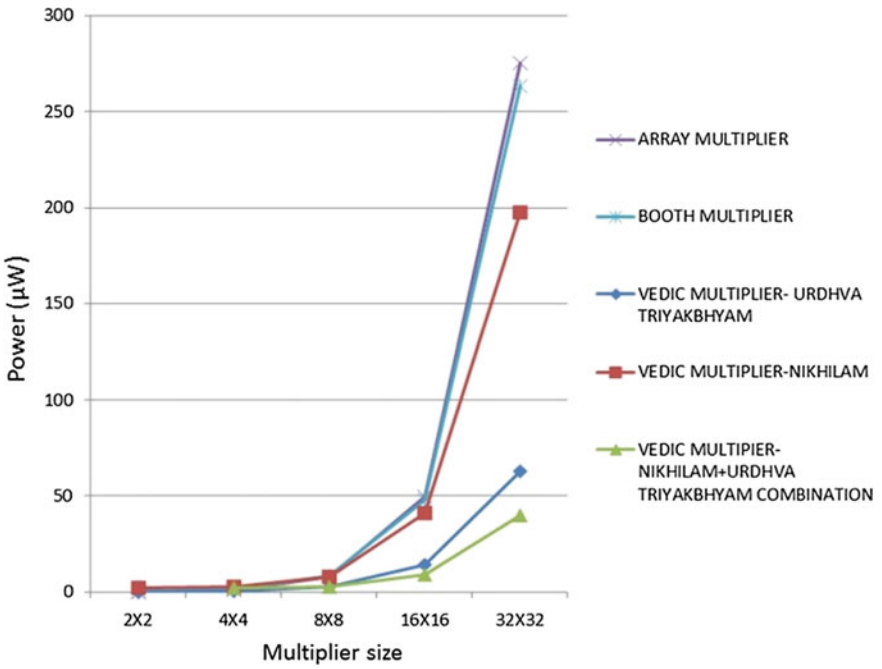

Fig. 9 Power plot of different multipliers of different orders. It can be seen that the combined use of Nikhilam and Urdhva Tiryagbhyam sutras results in the least power dissipation and the difference is prominently seen for higher order multipliers

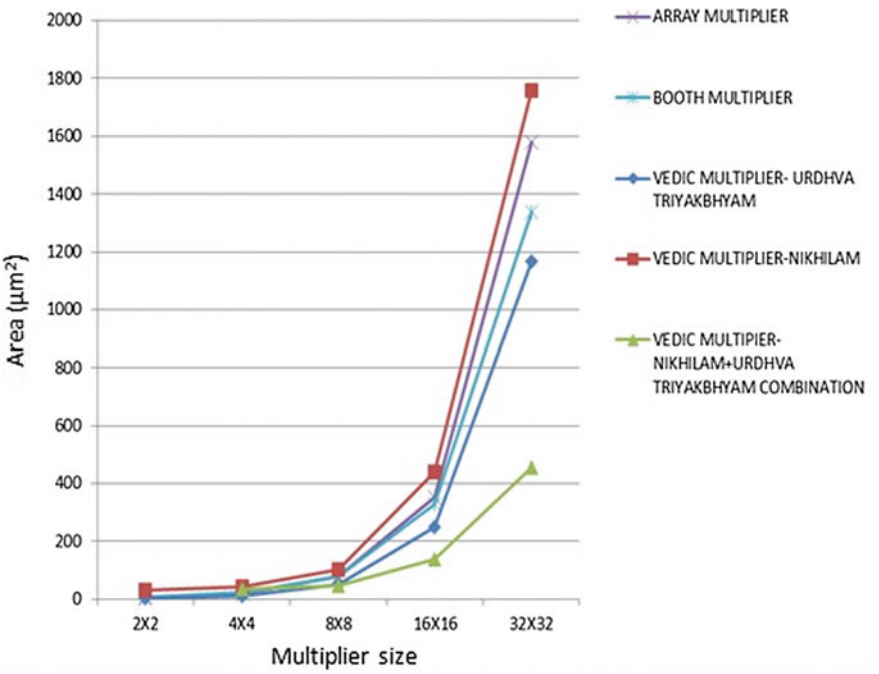

Fig. 10 Area plot of different multipliers of different orders. It can be seen that the combined use of Nikhilam and Urdhva Tiryagbhyam sutras results in the least area utilization and the difference is prominently seen for higher order multipliers

themselves are not very much area-power efficient while considering multipliers of large sizes. On the other hand, multiplier design using Nikhilam-Urdhva Tiryagbhyam combination gives the least area and power and the difference is significant for higher order multipliers.

6 Conclusions

A comparative analysis of floating-point multipliers using conventional multipliers (array and Booth) and vedic multipliers as the component integer multiplier unit has been done. Urdhva Tiryagbhyam sutra and Nikhilam sutra based designs were explored and a new design utilizing the two sutras together is proposed. Calculation of area and power of the different designs showed that the combined use of Nikhilam and Urdhva Tiryagbhyam based design is very efficient in terms of power and area for higher order multipliers and can be used for designing large multipliers for DSP applications.

References

1. Andraka R (1998) A survey of CORDIC algorithms for FPGA based computers. In: Proceedings of the 1998 ACM/SIGDA sixth international symposium on field programmable gate arrays. ACM, pp 191–200
2. Hamid LSA, Shehata K, El-Ghitani H, El- Said M (2010) Design of generic floating point multiplier and adder/subtractor units. In: 2010 12th international conference on computer modelling and simulation (UKSim). IEEE, pp 615–618
3. Marcus G, Hinojosa P, Avila A, Nolazco-FIores J (2004) A fully synthesizable single-precision, floating-point adder/substractor and multiplier in VHDL for general and educational use. In: Proceedings of the fifth ieee international caracas conference on devices, circuits and systems, 2004, vol 1. IEEE, pp 319–323
4. Prabhu E, Mangalam H, Karthick S (2016) Design of area and power efficient Radix-4 DIT FFT butterfly unit using floating point fused arithmetic. J Cent South Univ 23(7):1669–1681
5. M. M. Mano *et al.*, "Computer system architecture," 1982
6. Booth AD (1951) A signed binary multiplication technique. Q J Mech Appl Math 4(2):236–240
7. Al-Ashrafy M, Salem A, Anis W (2011) An efficient implementation of floating point multiplier. In: 2011 Saudi international electronics, communications and photonics conference (SIECPC). IEEE, pp 1–5
8. Ramteke P, Mhala N, Lakhe P (2014) An efficient implementation of double precision floating point multiplier using booth algorithm. Int J Adv Res Electr Electron Instrum Eng 3(7)
9. Tiwari HD, Gankhuyag G, Kim CM, Cho YB (2008) Multiplier design based on ancient indian vedic mathematics. In: ISOCC'08. international SoC design conference, 2008, vol 2. IEEE, pp II–65
10. Kanhe A, Das SK, Singh AK (2012) Design and implementation of floating point multiplier based on vedic multiplication technique. In: 2012 international conference on communication, information & computing technology (ICCICT), 2012, pp 19–20
11. Havaldar S, Gurumurthy K (2016) Design of vedic IEEE 754 floating point multiplier. In: IEEE international conference on recent trends in electronics, information & communication technology (RTEICT). IEEE, pp 1131–1135
12. Patel P, Shandilya A, Brahmbhatt N, Raval K, Deb D (2015) Vedic and conventional methods of n n binary multiplication with hardware implementation. In: International conference on smart sensors and systems (IC-SSS). IEEE, pp 1–6
13. Budhiraja H, Syed M, Ramya MA (2016) Verilog implementation of vedic multiplier. Int J Adv Eng Tech, Manag Appl Sci 3(5)
14. Jithin S, Prabhu E (2015) Parallel multiplier-accumulator unit based on vedic mathematics. ARPN J Eng Appl Sci 9(22):3608–3613
15. Zuras D, Cowlishaw M, Aiken A, Applegate M, Bailey D, Bass S, Bhandarkar D, Bhat M, Bindel D, Boldo S et al (2008) IEEE standard for floating-point arithmetic. IEEE Std 754-2008, pp 1–70
16. Rao YS, Kamaraju M, Ramanjaneyulu D (2015) An FPGA implementation of high speed and area efficient double-precision floating point multiplier using Urdhva Tiryagbhyam technique. In: 2015 conference on power, control, communication and computational technologies for sustainable growth (PCCCTSG). IEEE, pp 271–276
17. Bharath JSS, Tirathji K (1986) Vedic mathematics or sixteen simple sutras from the vedas. Motilal Banarsidas, Varanasi (India)