International Journal of Nanoelectronics and Materials
Volume 13, No. 2, Apr 2020 [315-322]

Square & Cube Computation using Vedic Algorithms in FPGA

A. Kamaraj[1*], Vidya. B[2], Sugapriya. M[2] and P. Marichamy[3]

[1,2]*Department of Electronics and Communication Engineering, Mepco Schlenk Engineering College, Sivakasi, Tamil Nadu 626 005, India.*
[3]*P.S.R. Engineering College, Sivakasi.*

Received 24 October 2019, Revised 5 January 2020, Accepted 6 February 2020

ABSTRACT

Modern computational devices are in the thirst for speedy computation. Adders and multipliers are the major computational units. Various types of multiplier architectures are suggested so far towards faster computation of the product. The speed of the multipliers could be improved by reducing the number of steps required for obtaining the product. One of the methods to reduce the number of steps is Vedic mathematics. There are 16 sutras in ancient Vedic mathematics. This research aims to design a square and cube computation using the Vedic algorithms. Yavadunam sutra (whatever the extent of its deficiency) is used for squaring and Anurupyena sutra (proportionately) is used to compute the cube of the binary number. In the Yavadunam sutra, bit reduction technique is employed to obtain deficiency, thereby, reducing the bit size to N-1 bits. Thus, reduces the delay. Urdhva Tiryagbhyam sutra (vertical and crosswise) is an efficient algorithm used for the multiplication operation. The design was implemented on a Xilinx-Spartan6 (XC6SLX16) FPGA.

Keywords: Vedic Mathematics, Yavadunam sutra, Anurupyena sutra, Urdhva Tiryagbhyamsutra.

I. INTRODUCTION

Vedic Mathematics (Vedic Maths) is a methodology used by ancient Indians for various computations. Vedic Maths was derived from "Vedas" which means the storehouse of knowledge [4]. Vedic Maths is not only a mathematical wonder but it is also logical. It also deals with various branches of mathematics such as algebra, geometry, and, arithmetic. The main feature of Vedic Maths is the coherence. It has sixteen different sutras for various mathematical operations. The application of Vedic sutras reduces computation time in complex operations. The following are the 16 main sutras of Vedic Maths: Ekadhikena Purvena, Nikhilam Navatascharamam Dastah, Urdhwa-triyagbhyam, Parvartya Yojet, Sunyam Samyasamuchchaye, Anurupyena, Sankalanavyavkalanabhyam, Puranpuranabhyam, Chalana Kalanabhyam, Yavadunam, Vyastisamastih, Sopantyadyamantyam, Ekanyunena Purven, Gunitasamuchachayah and Gunaksamuchchaya [11].

Many sub-sutras are also discovered from Vedas. Vedic Maths presents effective algorithms that can be applied to various branches of engineering. Squaring is one of the important operations in high-speed applications like Digital Signal Processing (DSP), animation and image processing. The speed of the squaring and cubing operation can be improved with the help of Yavadunam sutra (whatever the extent of its deficiency) and Anurupyena sutra (if one is in ratio, the other is zero) [9].

[*]Corresponding Author: *kamarajvlsi@gmail.com*

2. RELATED WORK

Sethi *et al.* [1] proposed a high-speed squaring circuit for binary numbers which is based on Vedic mathematics. It discussed the implementation of squaring architecture with reduced delay. It explains about the squaring technique based on Vedic mathematics is efficient in terms of speed. The space requirement is slightly more in this method. The presented squaring circuit is useful for the design of hardware for computer arithmetic.

Ganesh *et al.* [9] presented a design of High-Speed Vedic Multiplier (HSVM) using Vedic mathematics techniques. It allows the parallel generation of partial products and it discards the unwanted multiplication steps with zeros. It has incorporated urdhva tiryakbhyam sutra for multiplication. It proves that better performance was obtained in terms of speed. Deepak *et al.* [2] presented an efficient algorithm for fast cubing calculation. Bellam *et al.* [3] presented a low power and simple square and cube architectures in which the duplex property of the Dwandwa Yoga method was used for squaring [12].

3. PROPOSED WORK

3.1 Squaring using Yavadunam Sutra

Yavadunam sutra is one of the 16 Sutras of Vedic Mathematics. It is an efficient method to calculate the square of a number. Bit reduction technique is employed in this method by removing the MSB bit. In this design, the number of inputs to the squaring architecture gets reduced thereby reducing the delay and area. There are two modes in squaring using Yavadunam Sutra.

If the MSB bit is one or the given numbers is greater than $2^{(N-1)}$, then squaring is done by mode1. However, if the MSB bit is zero or the given number is less than $2^{(N-1)}$, then squaring is done by mode2. Let the input binary number as $A_3A_2A_1A_0$. Bit reduction is done by removing the MSB bit (A_3). Now the input to the architecture will be $A_2A_1A_0$ and these bits are known as a deficiency. Now the 4-bit number is reduced to a 3-bit number. If the input to the squaring architecture is 4-bits, then the output will be 8-bits and if the input is 3-bits the output will be of 6-bits. One of the modes is when the deficiency is positive (if the input is greater than $2^{(N-1)}$) and the other mode is when the deficiency is negative (if the input is less than $2^{(N-1)}$).

1) Algorithm for mode1

INPUT: $A_3A_2A_1A_0$

OUTPUT: $B_7....B_3B_2B_1B_0$

Step-1: Reduce the weight of the input by removing the MSB, now input become $A_2A_1A_0$ which is the deficiency

Step-2: Let the deficiency be $D_2D_1D_0=A_2A_1A_0$

Step-3: Square the deficiency, the output of squaring = $X_5...X_2X_1X_0$

Carry = $X_5X_4\ X_3$ and now

LHS =$X_2X_1X_0$ = $B_2B_1B_0$

Step-4: Add the deficiency and the input $A_3A_2A_1A_0+D_2D_1D_0 = Y_4Y_3Y_2Y_1Y_0$

Step-5: Add the above output to the carry of the LHS

$Y_4Y_3Y_2Y_1Y_0+X_5X_4X_3 = B_7B_6B_5B_4\ B_3$ = RHS

Step-6: Concatenating LHS and RHS, the output is

$B_7B_6B_5B_4B_3B_2B_1B_0$ = square of $A_3A_2A_1A_0$

2) Algorithm for mode2

INPUT: $A_3A_2A_1A_0$
OUTPUT: $B_5 \ldots B_2B_1B_0$

Step-1: Reduce the weight of the input by removing the MSB, now input becomes $A_2A_1A_0$.

Step-2: Take two's complement of $A_2A_1A_0$, the output is the deficiency. Let the deficiency be $D_2D_1D_0 = A_2A_1A_0$

Step-3: Square the deficiency, the output of squaring=$X_5 \ldots X_2X_1X_0$
Carry=$X_5X_4X_3$ and now LHS=$X_2X_1X_0$=$B_2B_1B_0$

Step-4: Subtract the deficiency from the bit reduced number $A_2A_1A_0$ - $D_2D_1D_0 = Y_2Y_1Y_0$

Step-5: If the subtractor output is positive then add the above output to the carry $X_5X_4X_3$.
$Y_2Y_1Y_0 + X_5X_4X_3 = B_5\ B_4\ B_3$ =RHS

Step-6: If the subtractor output is negative then subtract the output $Y_2Y_1Y_0$ from the carry $X_5X_4X_3$.
i.e. $X_5X_4X_3 - Y_2Y_1Y_0 = B_5B_4B_3$ = RHS

Step-7: Concatenating LHS and RHS, the output is
$B_5B_4B_3B_2B_1B_0$ = square of $A_3A_2A_1A_0$

The complete flow diagram of the squaring algorithm is presented in Figure 1.

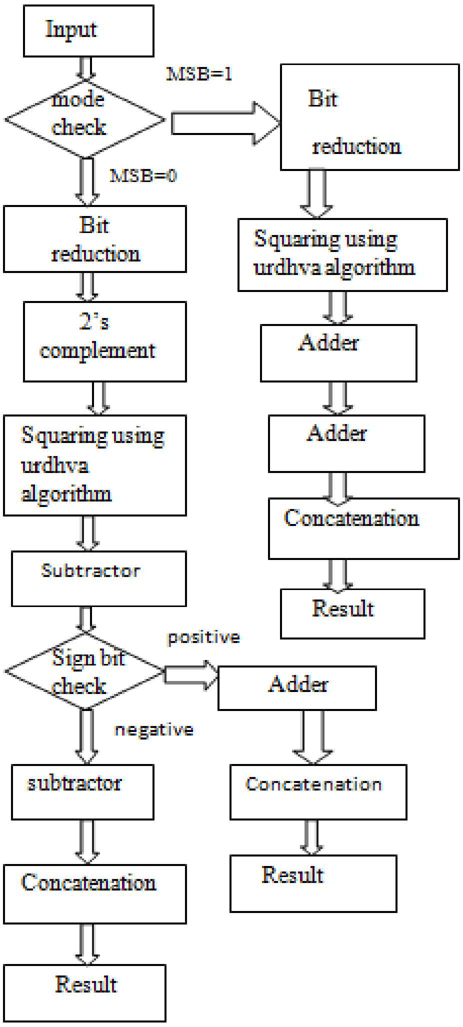


Figure 1. Flow diagram for squaring.

3.2 Cubing using Vedic Mathematics

i. Urdhva Tiryagbhyam algorithm: It does multiplication using "vertically and crosswise". The multiplication process is carried out according to Figure 2. Digits of both the sides of the cross line are multiplied and added with the carry from the previous step. If more than one line is in one step, all the results should be added to the previous carry. In every step, LSB bit is taken as a result and the remaining bits are considered as carry.

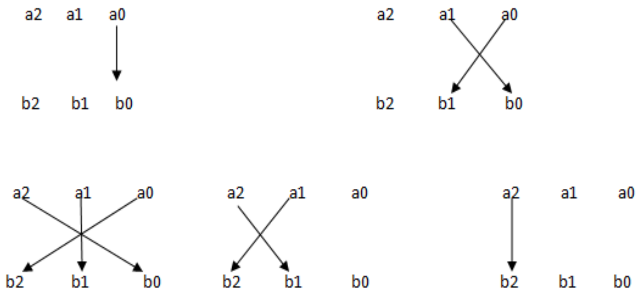


Figure 2. Line diagram for Urdhva Tiryagbham sutra.

ii. Vedic sutra to compute the Cube

a) Anurupyena sutra

The meaning of Anurupyena sutra is "proportionately formula" and can be applied to compute cube for any 2 digits number. For multiplication, the proposed method uses Urdhva triyakbhyam sutra. Let us consider the two-digit number as xy where y is one's digit number and x is ten's digit number. The following are the steps to compute the cube of a two-digit number which is represented in Figure 3.

i. Compute the cube of y.
ii. Compute xy^2 and $2xy^2$, add these two partial products.
iii. Compute x^2y and $2x^2y$, add these two partial products.
iv. Compute the cube of x. In each step add the carry from the previous step with the partial products.

$$\begin{array}{cccccc} (x\,y)^3 & = & x^3 & x^2y & xy^2 & y^3 \\ & & + & 2x^2y & 2xy^2 & \\ \hline & & x^3 & 3x^2y & 3xy^2 & y^3 \end{array}$$

Figure 3. Illustration of Anurupyena sutra.

b) Yavadunam sutra

Cube computation is performed with the help of Yavadunam sutra in Vedic mathematics. Carry look-ahead adders are used for addition. This adder is a fast-parallel adder which reduces the carry propagation delay [12]. In addition to this combinational shift registers are used to insert the partial products at suitable places and then concatenating them to obtain the final result. In Figure 4, "R" is known as the base of operation which determines the mode of operation. Elementary cubing was done by using Anurupyena Method. Urdhva triyakbhyam sutra is used for the multiplication operation. This Yavadunam sutra can be applied to any number of bits but the input number should be closest to the power of 10.

The flowchart in Figure 4 illustrates the computation of cube using Yavadunam formula in Vedic Maths.

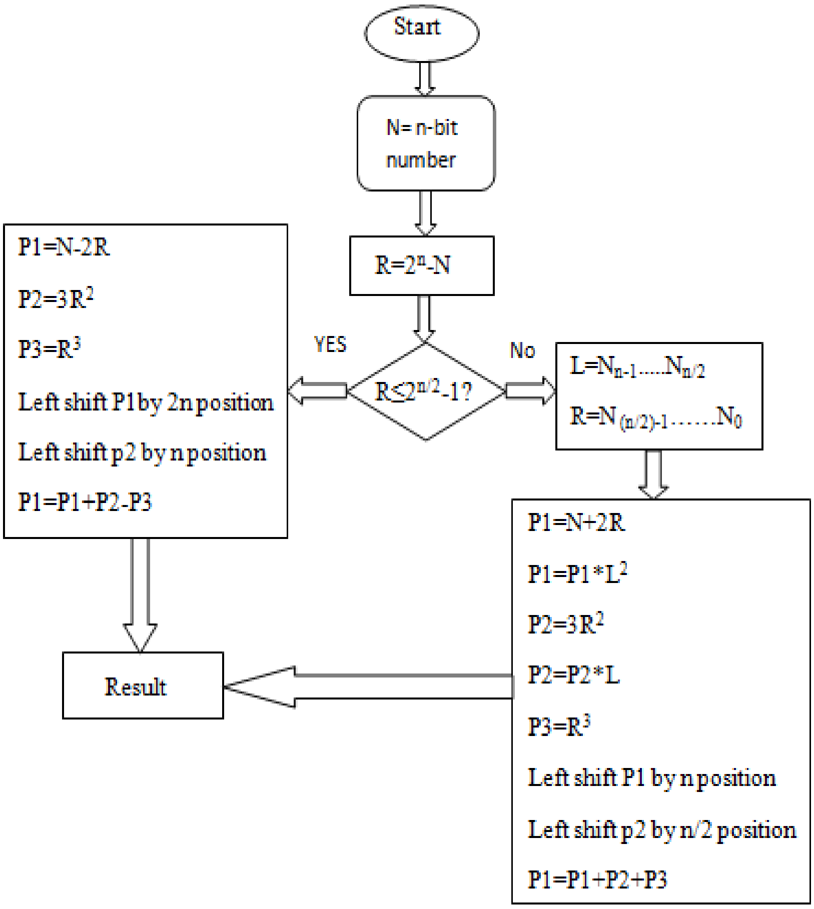
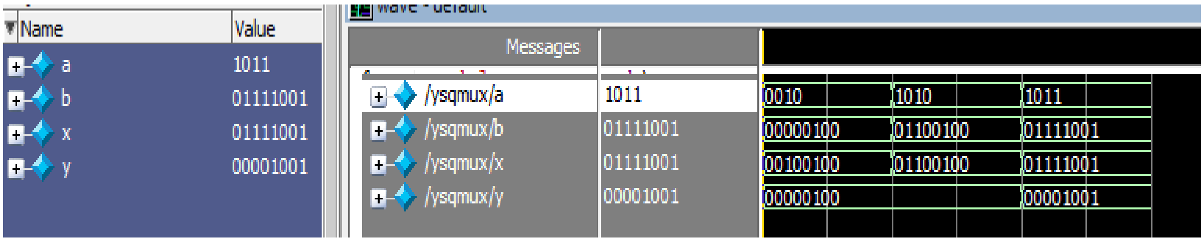


Figure 4. Flow diagram for Cube Computation.

4. RESULTS AND DISCUSSION

The above discussed squaring algorithm (Yavadunam) and cubing algorithms (Anurupyena & Yavadunam) are translated into Verilog-Hardware description language (Verilog HDL) and its functional simulations are performed. 4-bit, 8-bit and 16-bit squaring units are constructed, and their functional simulations are shown in Figure 5(a), 5(b), and 5(c), respectively.

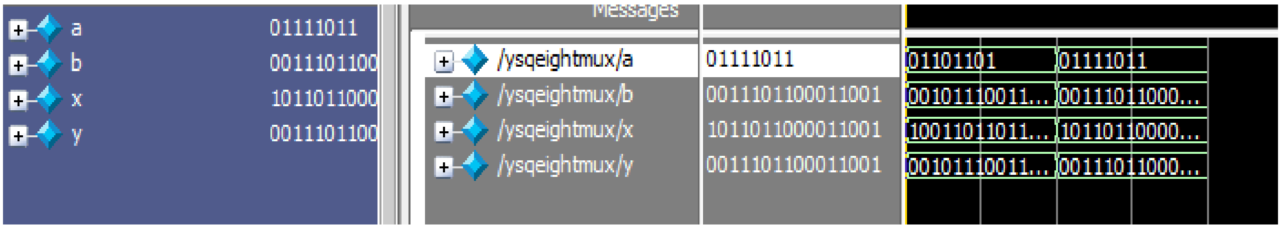
A. Squaring Algorithm



Name	Value
a	1011
b	01111001
x	01111001
y	00001001

Messages				
/ysqmux/a	1011	0010	1010	1011
/ysqmux/b	01111001	00000100	01100100	01111001
/ysqmux/x	01111001	00100100	01100100	01111001
/ysqmux/y	00001001	00000100		00001001

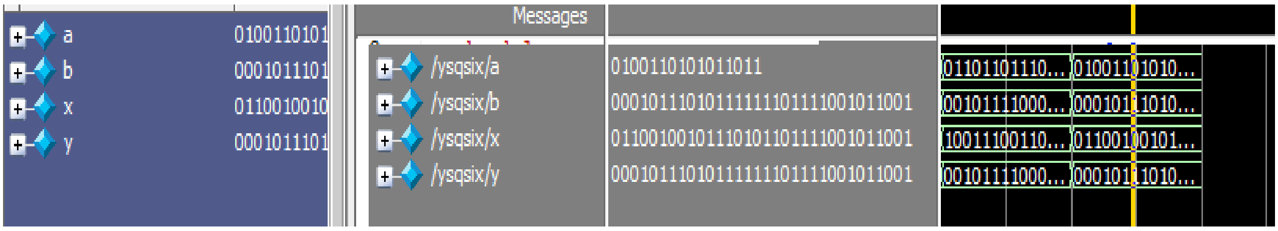
Figure 5(a). Simulation of 4-bit squaring.



```
a    01111011
b    0011101100
x    1011011000
y    0011101100

Messages
/ysqeightmux/a    01111011             01101101         01111011
/ysqeightmux/b    0011101100011001     00101110011...   00111011000...
/ysqeightmux/x    1011011000011001     10011011011...   10110110000...
/ysqeightmux/y    0011101100011001     00101110011...   00111011000...
```

Figure 5(b). Simulation of 8-bit squaring.



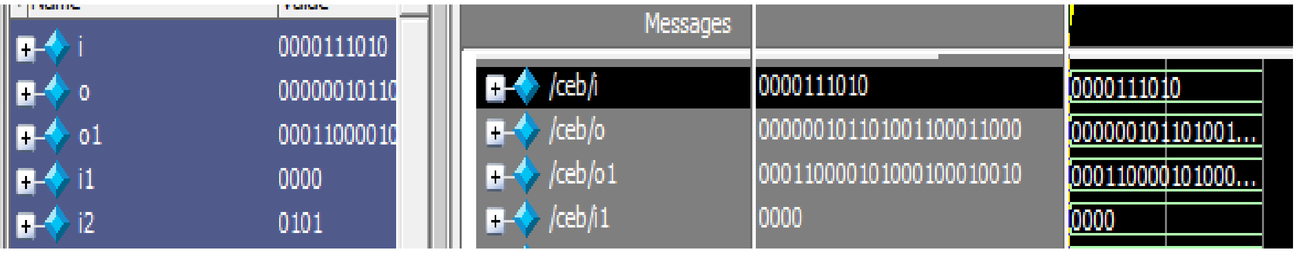
```
a    0100110101
b    0001011101
x    0110010010
y    0001011101

Messages
/ysqsix/a    0100110101011011                    01101101110...   01001101010...
/ysqsix/b    00010111010111111101111001011001    00101111000...   00010111010...
/ysqsix/x    01100100101110101101111001011001    10011100110...   01100100101...
/ysqsix/y    00010111010111111101111001011001    00101111000...   00010111010...
```

Figure 5(c). Simulation of 16-bit squaring.

B. *Cubic Algorithm*

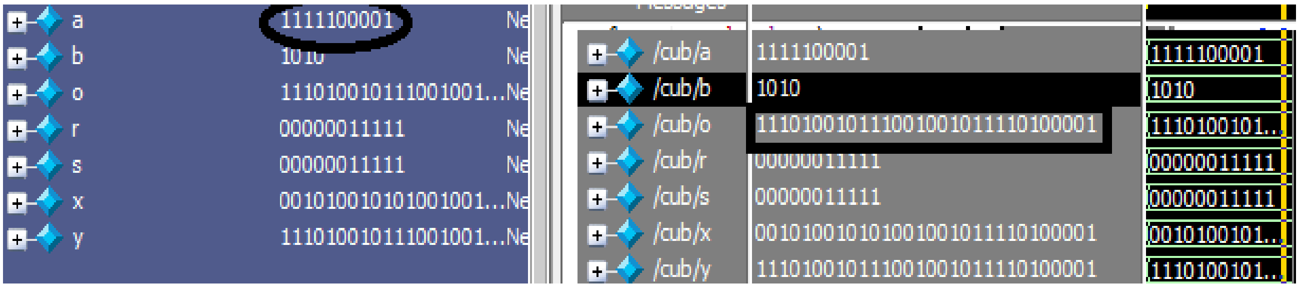
The Cubic algorithms Anurupyena and Yavadunam are designed for 2 digits and 3 digits respectively. Also, their simulation waveforms are shown in Figure 6(a) and 6(b).



```
i     0000111010
o     00000010110
o1    00011000010
i1    0000
i2    0101

Messages
/ceb/i     0000111010                    0000111010
/ceb/o     0000001011010011000110000     000000101101001...
/ceb/o1    0001100001010001000100010     000110000101000...
/ceb/i1    0000                          0000
```

Figure 6(a). Simulation of Anurupyena.



```
a    1111100001                Ne
b    1010                      Ne
o    1110100101110010 01...    Ne
r    00000011111               Ne
s    00000011111               Ne
x    0010100101010010 01...    Ne
y    1110100101110010 01...    Ne

Messages
/cub/a    1111100001                        1111100001
/cub/b    1010                              1010
/cub/o    111010010111001001011110100001    1110100101..
/cub/r    00000011111                       00000011111
/cub/s    00000011111                       00000011111
/cub/x    001010010101001001011110100001    0010100101..
/cub/y    111010010111001001011110100001    1110100101..
```

Figure 6(b). Simulation of Yavadunam.

The Vedic algorithms based squaring and cubing is implemented in Xilinx- Spartan6 FPGA. The implementation results are shown in Table 1, Table 2, Table 3 and Table 4.

Table 1 shows the device utilization of the dedicated multiplier cell available in the Spartan6 FPGA and Yavadunam based squaring method. It is observed that the Vedic multipliers are efficient in terms of area and delay. Both optimized dedicated multiplier and Vedic multipliers have the same performance, which indicates that the Vedic multipliers have a natural improvement.

Table 1 Resource utilization of Dedicated and Yavadunam Squaring

DEVICE UTILIZATION	DEDICATED MULTIPLIER	YAVADUNAM SUTRA
Number of slices	3 out of 2278	3 out of 2278
Number of 4input LUTs	3 out of 9112	3 out of 9112
Delay in ns	6.292	6.292

Table 2 shows the device utilization of the conventional array multiplier based squaring and Yavadunam based squaring method. It is identified that the Vedic multipliers have lesser number of AND logic and Adders than the conventional array multiplier. Also, they occupy the minimum area and reduced delay than array multipliers.

Table 2 Resource utilization of Conventional and Yavadunam Squaring

DEVICE UTILIZATION	CONVENTIONAL	YAVADUNAM SUTRA	% of Improvement
AND logic	16	9	43.75%
Number of Adders	10	6	40%
Number of slices	13 out of 1920	4 out of 1920	69.23%
Number of 4 input LUTs	23 out of 3840	6 out of 3840	73.91%
Delay in ns	17.293	7.862	54.54%

Table 3 shows the implementation results of the 8 and 16-bit Yavadunam algorithm. It is compared with conventional. The number of hardware logic required is 23% lesser than the conventional logic. Table 4 shows the resource utilization of Anurupyena Cubing in FPGA.

Table 3 Resource utilization of Yavadunam Squaring (8-Bit & 16-Bit)

DEVICE UTILIZATION	8-BIT	16-BIT
AND Logic	Conventional-64 Yavadunam-49	Conventional-256 Yavadunam-226
Number of slices	41 out of 2278	412 out of 2278
Number of LUTs	100 out of 9112	1076 out of 9112
Delay in ns	8.042	69.482

Table 4 Resource utilization of Anurupyena Cubing

DEVICE UTILIZATION	ANURUPYENA SUTRA
Number of slices	245 out of 2278
Number of LUTs	636 out of 9112
Delay in ns	156.117

5. CONCLUSION

The addition and multiplication are the major functional unit for the computing devices. They consume huge resources and contributing to more delay. So, the computation process can be speeded up, if the area and delay of the above process are reduced. The Vedic algorithms are the best suitable methods to reduce the number of computation steps. This paper has adapted

Yavadunam and Anurupyena algorithms for squaring and cubing of integers. The algorithmic procedures are converted into the hardware logic for that purpose. The capabilities of the algorithms are checked against the conventional array multiplier and dedicated multipliers in the Spartan 6 FPGA. The implementation results confirm that the Vedic based squaring and cubing computations are faster and the area also minimal compared to the array multiplication. It is observed that there is an improvement of up to 23% for the area and 12% for the speed.

REFERENCES

[1] Kabiraj Sethi & Ruuparna Panda, "An improved Squaring circuit for binary numbers", International Journal of Advanced Computer Science and Applications **3**, 2 (2012) 1-6.

[2] Deepak Kumar, Prabir Saha, Anup Dndapat, "Vedic Algorithm for cubic computation and VLSI Implementation", An International Journal of Engineering science and Technology **20** (2017) 1494-1499.

[3] Bellam Sreekanya, Rakesh K, "Square and Cube Architecture using Less Complex and Low Power Architecture", International Journal of Computer Engineering in Research Trends **2**, 12 (December 2015) 1232-1235.

[4] Ruchi Anchaliya, Chiranjeevi G N, Subash Kulkarni, "Efficient Computing Techniques using Vedic Mathematics Sutras", International Journal of Innovative Research in Electrical, Electronics, Instrumentation and Control Engineering **3**, 5 (May 2015) 1-4.

[5] Poornima M, Shivaraj Kumar Patil, Shivukumar, Shridhar K P, "Implementation of Multiplier using Vedic Algorithm", International Journal of Innovative Technology and Exploring Engineering **2**, 6 (May 2013) 1-5.

[6] Deepa A, Marimuthu C N, "High Speed VLSI Architecture for Squaring Binary Numbers Using Yavadunam Sutra and Bit Reduction Technique", International Journal of Applied Engineering Research **13**, 6 (2018) 4471-4474.

[7] Prashant D Pawale, Venkat N Ghodke, "High Speed Vedic Multiplier design and Implementation on FPGA", International Journal of Applied Research **1**, 7 (2015) 239-244.

[8] Shubham Mishra, Shailendra Kumar Dhakad, "A high speed and device efficient FPGA based Squaring circuit", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering **2**, 11(2013) 5647-5654.

[9] G Ganesh Kumar, V Charishma, "Design of High Speed Vedic Multiplier using Vedic Mathematics Techniques", International Journal of Scientific and Research Publications **2**, 3 (March 2012) 1-5.

[10] Jagannath Samanta, Mousam Halder, Bishnu Prasad De, "Performance Analysis of High-Speed Low Power Carry Look-Ahead Adder Using Different Logic Styles", International Journal of Soft Computing and Engineering (IJSCE) **2**, 6 (Jan 2013) 330-336.

[11] W. B. Vasantha Kandasamy, Vedic Mathematics - 'Vedic' or 'Mathematics': A Fuzzy & Neutrosophic Analysis, arXiv:math/0611347, (2006).

[12] Vaijyanath Kunchigi, Linganagouda Kulkarni, Subhash Kulkarni, Low power Square and Cube Architectures Using Vedic Sutras, Fifth International Conference on Signals and Image Processing.