# Semi-Custom Design of Fast Multiplier Based on Vedic Sutras

**Sushma P. S., Shrividya G.**

*Associate Professor, NMAM Institute of Technology, Nitte Deemed to be a university. Email: pssushma@nitte.edu.in*

A multiplier is a crucial component of general-purpose processors and digital signal processors, which are frequently employed in tasks like image processing, communication systems, and digital signal processing, among others. Multipliers are necessary for fundamental signal processing tasks like convolution, Fast Fourier Transform, digital filtering, etc. As a result, there is a growing need for optimised high-speed, low-power multipliers. Most VLSI implementations make use of multiplier architectures such as the Wallace Tree multiplier, Array Multiplier, Booth Multiplier, Bot Serial Multiplier, Carry Save Multiplier, and Modified Booth Multiplier. This essay primarily focuses on the application of Vedic mathematics to develop a high-speed multiplier with a very compact footprint. The broad Vedic mathematics sutra is Urdhva-Tiryagbhyam, one of the sixteen. The present study involves the creation of a 16-bit fixed point multiplier utilising the Urdhva-Tiryagbhyam (vertically and across) sutra. The implementation of the multiplier is carried out on the Spartran6 FPGA, utilising the Verilog HDL description. The design is functionally verified using ISim Simulator and synthesised using the Xilinx ISE 14.2 tool. The 16-bit multiplier, designed utilising Vedic sutra, demonstrates exceptional performance in terms of speed, memory utilisation, and area utilisation.

**Keywords:** Vedic Multiplier, Vedic Mathematics, Urdhva-Tiryagbhyam sutra.

## Introduction

Due to an overabundance of computing and signal processing applications, there is an increasing need for high-speed signal processing. Arithmetic functional units with high throughput are essential in achieving the required performance in numerous real-time signal and image processing-related applications. Multiplication is a crucial arithmetic process in several applications. Over the years, there has been interest in the creation of rapid multiplier circuits. For many applications, minimising time delays and power consumption is a crucial requirement. A quick, low-power, and space-efficient multiplier is based on Vedic mathematics.

Operations such as convolution, Fast Fourier Transform (FFT) and filtering are most common in Digital Signal Processing (DSP) applications. In the Arithmetic and Logic Unit

(ALU) of microprocessors, currently Computation Intensive Arithmetic Functions (CIAF) are widely employed in multiplication-based operations like Multiply and Accumulate (MAC) and inner product A high-speed multiplier is required since most DSP algorithms spend the majority of their processing time multiplying numbers. Multiplication time still has a significant impact on a DSP processor's instruction cycle time. Optimising at all design stages is necessary to reduce power consumption for digital systems.

This design optimisation encompasses the technology employed to achieve the design, the type and arrangement of the circuits, the architecture, and the system-level algorithms. Digital multipliers are often employed components in the construction of digital circuits. These components are utilised for the execution of various operations because to their rapidity, reliability, and efficacy. Various types of multipliers are available depending on the method of component assembly.

The performance algorithm is ultimately determined by the multiplier, which is situated in the critical time delay route in many processing algorithms. In both DSP and conventional processors, operating speed during multiplication is crucial. Traditionally, multiplying integers has been done using standard addition, subtraction, and shift operations. In the literature, there have been numerous algorithms and suggestions for performing multiplication, each with a unique set of benefits and trade-offs in terms of speed, footprint, complexity of the circuit and on the amount of power dissipation.

The multiplier is a significant cause of power loss in addition to being a sizable component of the computer system. The MAC units are the fundamental building blocks of all digital signal processors, and their speed has a substantial impact on the DSP's processing speed. In digital circuits, array multipliers and Booth's multiplier are the two most commonly used architectures. Because the partial products are computed individually and concurrently, the computation time of the array multiplier is greatly decreased. The delay associated to the array multiplier measures how long it takes for signals to move through the array of adders. Another crucial multiplication algorithm is the Booth algorithm, which requires enormous Booth arrays.

Vedic Mathematics:

The ancient Indian sages gave the world of Vedic mathematics as a gift. Compared to modern mathematics, this is easier and more entertaining. Vedic mathematics is straightforward, thus calculations can be made mentally. The use of an adaptable mental system has various benefits. Students are not constrained to a single method and are free to devise their own. This results in students who are smarter and more creative. The procedure of carrying out computations based on a the 16 Sutras and associated sub-sutras or corollaries is referred to as an algorithm in Vedic mathematics. These Sutras can be used to mentally answer any mathematical problem, including those involving algebra, arithmetic, geometry, and trigonometry.

A novel and incredibly effective approach to different fields of mathematics that covers a wide spectrum is provided by Vedic mathematics [1]. It begins with simple addition and

ends with somewhat more complex subjects, such as solution of non-linear partial differential equations. In any practical application, it fully utilises the features of numbers. It is a unit of the Stapatya Veda, an Upa- Veda of the Atharva Veda and a text about civil engineering and architecture.

His Divineness Jagadguru Swami Sri Shankaracharya Bharathi Krishna Theerthaji Maharaja's (1884–1960) works are collectively credited to the Govardhana Matha in Puri, Odisha, India. He provided a mathematical analysis of each work and discussed its potential uses. Swamiji composed 16 sutras (formulae) and 16 Upa-sutras (sub formulae) after conducting rigorous research in the Atharva Veda. Vedic mathematics is a mathematical miracle as well as being logical. Due to its extraordinary qualities, it has already travelled outside of India and established itself as a hotly debated subject outside. It deals with a variety of elementary and sophisticated mathematical processes. Basic math procedures in particular are exceedingly simple and efficient.

The 16 upa-sutras can be applied on trigonometry, spherical & plane geometry, differential & integral calculus, as well as other types of mathematics. As was previously noted, all of these Sutras were recreated from ancient Vedic writings in the early 20th century. Vedic mathematics is beautiful because it simplifies computations that might otherwise appear to be complex in traditional mathematics. This is true because it is said that the Vedic equations are founded on the same basic laws that govern how the human mind functions. This article discusses some efficient algorithms that can be used in many engineering fields, including computers and digital signal processing.

**Literature Survey:**

Vedic mathematics is based on a total of 16 sutras that cover geometry, algebra, and arithmetic operations. The most widely used sutra for multiplication is UrdhvaTiryagbhyam, which produces results quickly and efficiently. The power consumption of integrated circuits can be decreased using a variety of methods [1]. To reduce the circuit's power usage, an adiabatic Vedic multiplier is suggested.

The significance of multiplication in digital circuits was proposed by V. Meghana et al. [2]. The most basic operation in many DSPs is multiplication, which is used for functions including convolution, filtering, FFT, and ALU units. DSP systems' primary computational needs are for multiplication, necessitating the use of a high-speed multiplier device that is also power-efficient. The array multiplier's sequential processing of the products increases the computational delay. The booth multiplier delays computing by doing add-or-subtract and arithmetic-shift operations repeatedly. Due to the expansion of signal and computer processing applications, there has been a consistent rise in the demand for high-speed high throughput operations. Numerous applications for practical signal and image processing rely on mathematical calculations with increased throughput to deliver the requisite performance. For many applications, the decrease of delay and power is a crucial need. The multiplier designed utilized the idea of Vedic mathematics to fulfil these specifications. Vedic mathematics was used to greatly decrease the amount of partial products computed, speeding up the multiplier's operation.

The application of the Urdhva Tiryakbhyam sutra in multiplication was described by S. N. Gadakh et al. in [3]. Shri Bharati Krishna Tirthaji (1884–1960), after studying the Atharva Vedas for eight years, discovered the Vedic mathematics theory. Vedic mathematics is founded on sixteen sutras. This contains a number of effective algorithms that can be applied to different fields of engineering. Studies of Vedic mathematics have revealed that the following two sutras—Urdhvatiryagbhyam, which means "vertically and crosswise," and "Nikhilam Navatashcaramam Dashatah", which means "all from 9 and last from 10"—depict the multiplication operations. The multiplicand and multiplier in the Nikhilam sutra should be closer to the bases of 10, 100, and 1000, or powers of 10. When the multiplicand and multiplier have a wide disparity, this sutra does not apply.

The multiplier design is as per the UrdhvaTriyagbhyam approach, which is utilised in ancient Indian Vedic mathematics, according to S. P. Pohokar et al. [4]. The UrdhvaTriyagbhyam Sutra is a universal multiplication algorithm that is applicable in every circumstance. It is founded on an innovative theory that enables the simultaneous generation of all partial products and their addition. To accomplish parallelism in the construction of partial products and their summation, Urdhava Triyagbhyam is utilised. As a result, the carry propagation time from LSB to MSB is decreased by this factor.

The UrdhvaTiryagbhyam algorithm's steps for multiplying two 2-bit binary values were described by K. D. R. C. Gangadhar et al. in detail [5]. As multiplicand and multiplier, two 2-bit binary values a1a0 and b1b0a are considered. The lowest significant bit s0 of the final result is obtained by first multiplying vertically the least significant bits of the multiplicand and multiplier as indicated. Then, as illustrated in, the crosswise product of the multiplicand's bit a0 and the multiplier's bit b1 is added to the multiplier's bit a1 and the multiplicand's bit a0 to produce the second bit s1 of the final product and carry bit c1. The multiplier multiplies the most important bit, a1, of the multiplicand, b1.

Vedic multiplier is faster than array and Booth multiplier, claim R. Raju et al. [6]. The Vedic multiplier's delay duration is much less than that of other multipliers as the number of bits rises from 8 to 16. The Vedic multiplier has an advantage over other multipliers in terms of the consistency of the gates and structure delays. As a result, this multiplier operates at the fastest pace among traditional multiplies. It is preferable over other multipliers with a great advantage. When compared to other multiplier architectures, the Vedic square multiplier requires a very tiny amount of space. The Vedic square multiplier is the fastest and smallest of the reviewed architectures, according to the outcome. Thus, it was shown that, in terms of speed and area, the Vedic design was superior to Booth Array and even modular multiplication.

Four 8-bit multiplier blocks and three 16-bit adders make up the architecture of a 16-bit multiplier, according to S. V. Mogre et al. [7]. The output is 32 bits in size, but the multiplicands in this case are 16 bits. Small blocks of size n/2=8 are created from the input. The freshly created 8-bit block is given as the input to an 8-bit Vedic multiplier block, which then further divides the block into smaller blocks of size n/4=2 and provides the input to a 2-bit Vedic multiplier block. The output of the 16-bit, 8-bit Vedic multiplier block is sent to

the 16-bit adder as the output.

The benefits of Vedic multiplier over other multipliers were described by D. K. Kahar et al. in their article [8]. Compared to other multipliers like the booth multiplier and array multiplier, the Vedic multiplier has a shorter time delay. Due to its avoidance of shifting and adding partial products, the suggested design can process data at high speeds and throughputs. Comparing Vedic multiplier to other multipliers, less slices are used. Vedic multiplier is faster than other algorithms. When compared to other multiplier methods, the size needed to run this technique is extremely small.

The majority of VLSI implementations employ various multiplier architectures, including the Wallace tree multiplier, array multiplier, Booth multiplier, bit-serial multiplier, carry save multiplier, and modified Booth multiplier. Combinational logic is used in array multipliers to multiply two binary values. This multiplier is quicker because it executes the product of all bits at once, but it is not cost-effective due to the high number of gates needed. Bits are processed one at a time in a carry save multiplier to add carry in an adder. As a result, as the number of bits rises, its execution time increases since it depends on the prior carry. A one-bit complete adder in a Wallace tree is given three-bit signals as input. The next stage full adder, which is in a higher position, receives the sum of these inputs as its output. The Vedic multiplier is suggested as a way to boost speed and get over the downside of these multipliers.

**Proposed Method:**

The Urdhva Triyagbhyam sutra is useful in every case involving multiplication. It indicates "vertically and crosswise". This sutra permits both the inclusion of all partial products and their simultaneous creation. The method can be expanded to work with N × N bit numbers. The multiplier is unaffected by the clock of the processor due to the concurrent calculation of partial products and their sums. The augmentation of the input and output data buses can effectively enhance the processing capacity of the multiplier, owing to its inherent regular structure and uncomplicated construction on a silicon chip. As the number of bits rises, the gate delay and area required exhibit a gradual increase in comparison to alternative multipliers. As a consequence, it consumes less amounts of energy, time, and space.

In this work, the Urdhva Triyagbhyam sutra is used to implement a 16-bit multiplication process. First, a 2-bit multiplier is created. The implementation of higher-level multipliers, such as 4-bit, 8-bit, and 16-bit multipliers, is then done using this module. For the design, Verilog code is created. The functional verification and the logic synthesis is performed using Xilinx ISE 14.2 tool which has an integrated ISim Simulator. The design is implemented on Spartan6 FPGA, XC6SLX9 device.

Design and Descriptions:

To illustrate Urdhva Triyagbhyam sutra, consider two 2-digit decimal numbers 36 and 25 to be multiplied. Assuming that the product to be 4 digits, let p3 p2 p1 p0 be the digits of the product. The least significant digit (LSD) 6 of multiplicand is multiplied vertically by LSD 5 of the multiplier, get their product as 30 and set 0 for the LSD (p0) and the digit 3 is

considered as carry. Then 3 and 5, and 6 and 2 are multiplied in the crosswise direction and added to the carry generated in the earlier step to get 30 which is taken as sum = 0 and carry = 3, the sum bit is the middle part of the answer (p1). Then 3 and 2 are multiplied in the vertical direction, and added to the previous carry and get p3 p2 as 09 as their product and put it downs as the most significant part of the product, (p3 p2). So, 36 X 25 = 0900. Multiplication steps for 2-digit decimal numbers are depicted in Figure 1.

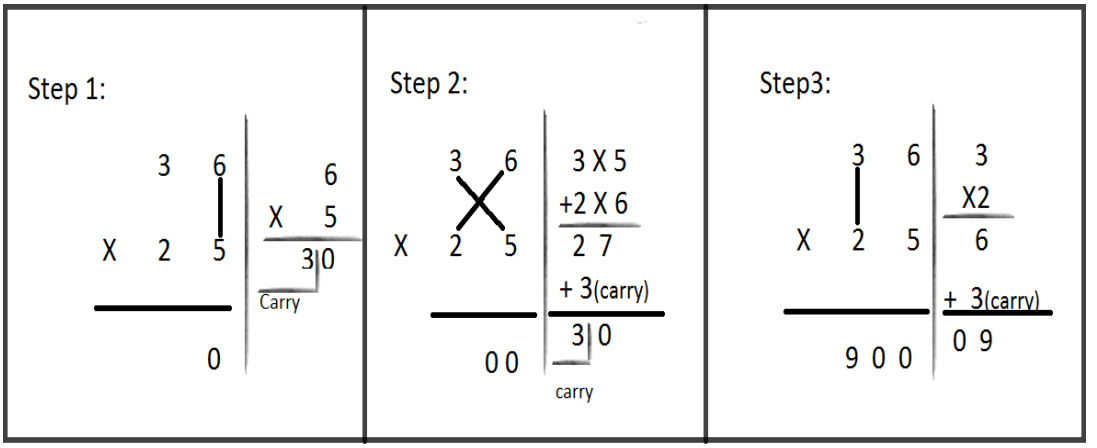


Figure 1: Multiplication steps for 2-digit Decimal numbers

Design of 2-bit Multiplier:

The least significant bit (LSB) products A0 and B0 are vertically multiplied in step 1, resulting in the LSB product S0 shown in Figure 2. The cross multiplication of A1 B0 and A0 B1 is then carried out in step 2. The cross multiplication's partial products are added together to generate a product bit of S1 and a carry bit of C0. The most significant bit (MSB) vertical multiplication comes last. After adding the carry C0 from step 2 to the partially created product, a product bit of S2 and COUT is obtained. The finished item, COUT S2 S1 S0 (4 bit), is the result of these procedures.

| Pictorial Representation | Equations |
|---|---|
| Step 1<br>•(A1) •(A0)<br>•(B1) •(B0) | S0 = A0*B0 |
| Step 2<br>•(A1) •(A0)<br>•(B1) •(B0) | C0 S1 = (A1*B0) + (A0*B1) |
| Step 3<br>•(A1) •(A0)<br>•(B1) •(B0) | $C_{OUT}$ S2 = (A1*B1) + C0 |

Figure 2: Design Steps for 2-bit Multiplier using Urdhva Tiryagbhyam Algorithm

Circuit Design:

The logic diagram for the 2-bit Vedic multiplier is depicted in Figure 3. The 2-bit Vedic multiplier is composed of four AND gates that will function as 2-bit multipliers and two half adders that will add the products to obtain the final product.

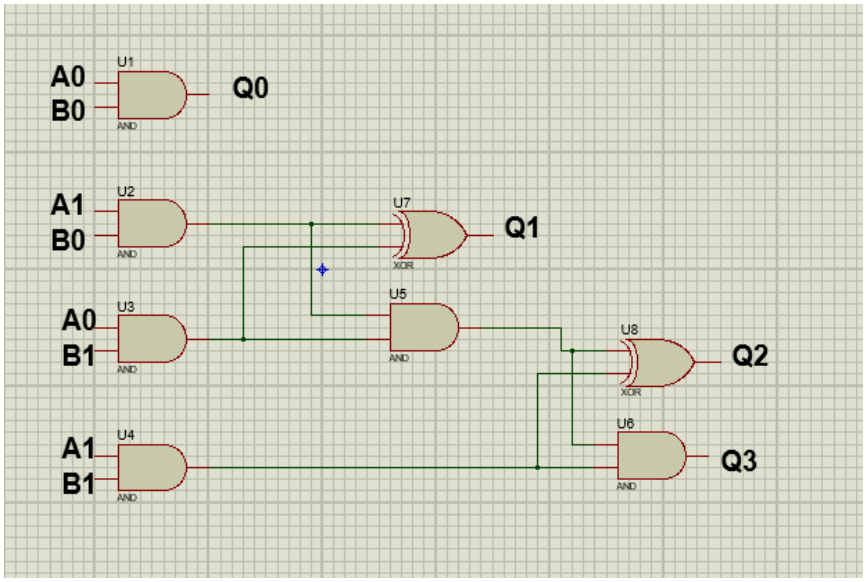


Figure 3: Circuit for 2-bit Multiplier

Design of 4 bit multiplier:

Consider two 4-bit binary values, a3 a2 a1 a0 and b3 b2 b1 b0 as input to the Vedic multiplier. There are seven steps in the process, and each one results in a partial output. Initially, the LSB of the multiplier and the multiplicand (vertical multiplication) are multiplied as seen in step 1 of Figure 4. The resultant value is the product's LSB. In step 2, the next higher bit of the multiplier is multiplied by the LSB of the multiplicand, and the following higher bit of the multiplicand is multiplied by the LSB of the multiplier. The remaining bits of these two partial products are added, and the remaining bits are sent on to the next phase. The following higher bit in the result is determined by the LSB of the sum. As an illustration, if the result of the intermediate step is 1101, 1 will act as the result bit (abbreviated as rn) and 110 as the carry bit (abbreviated as cn). Therefore, cn might be a multi-bit number. Similar actions are taken in other steps, as shown by the line diagram. Here, all partial products and their sums for every step can be computed in parallel. Thus, for step 1 to 7 in Figure 3 are evaluated as per expressions given by equation 1 to 7 respectively.

$r0 = a0b0$ (1)

$c1r1 = a1b0 + a0b1$ (2)

$c2r2 = c1 + a2b0 + a1b1 + a0b2$ (3)

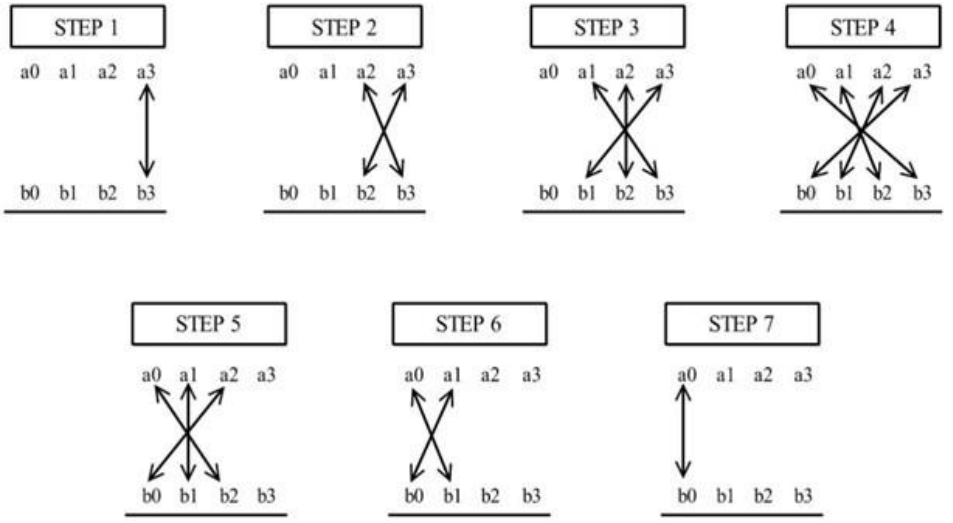


Figure 4: Design Steps for 4-bit Multiplier using Urdhva Tiryagbhyam Algorithm

$c3r3 = c2 + a3b0 + a2b1 + a1b2 + a0b3$ (4)

$c4r4 = c3 + a3b1 + a2b2 + a1b3$ (5)

$c5r5 = c4 + a3b2 + a2b3$ (6)

$c6r6 = c5 + a3b3$ (7)

c6 r6 r5 r4 r3 r2 r1 r0 is the final product. The remaining digits are therefore solely considered to be carry digits and the LSB of the resulting number is used as the product

digit. The initial carry is assumed to be 0.

Figure 5 illustrates how to construct a 4-bit Vedic multiplier using three adders and four 2-bit Vedic multipliers.

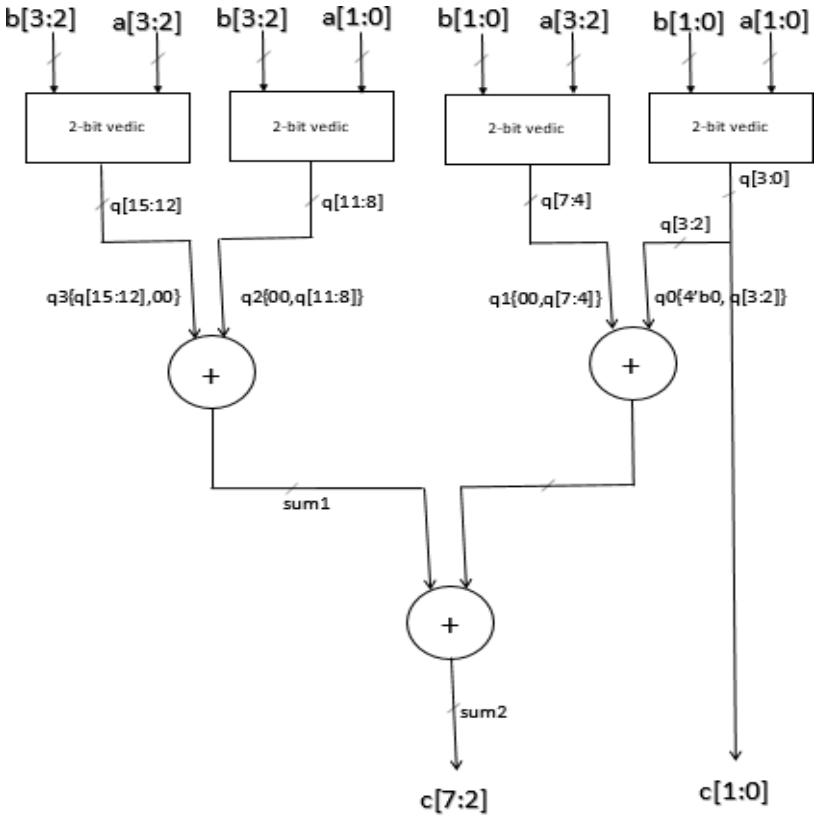


Figure 5: Block Diagram for 4-bit Multiplier [6]

Design of 8-bit Multiplier:

The 4 bit (nibble) of each 8 bit input will be grouped as the initial step in the building of an 8-bit Vedic multiplier block. Vertical and crosswise product terms will be created by these quadruple words. To create eight partial product rows, a separate 4-bit Vedic multiplier handles each input bit-quadruple. The final product bits are produced by adding these partial products rows in an 8-bit adder. The design process is shown in Figure 6. The terms for the Urdhva cross and vertical products are represented by the partial products. Then using half adder assembly the final product is obtained. Addition of generated partial product is done as shown in Figure 7.

| | | | |
|---|---|---|---|
| A | = | A7A6A5A4 | A3A2A1A0 |
| | | (X1) | (X0) |
| B | = | B7B6B5B4 | B3B2B1B0 |
| | | (Y1 ) | (Y0) |

X1 X0
X Y1 Y0
---
F E D C

CP = X0 * Y0 = C
CP = X1 * Y0 + X0 * Y1 = D
CP = X1 * Y1 = F E
Where CP = Cross Product.

Figure 6: Design Steps for 8-bit multiplier

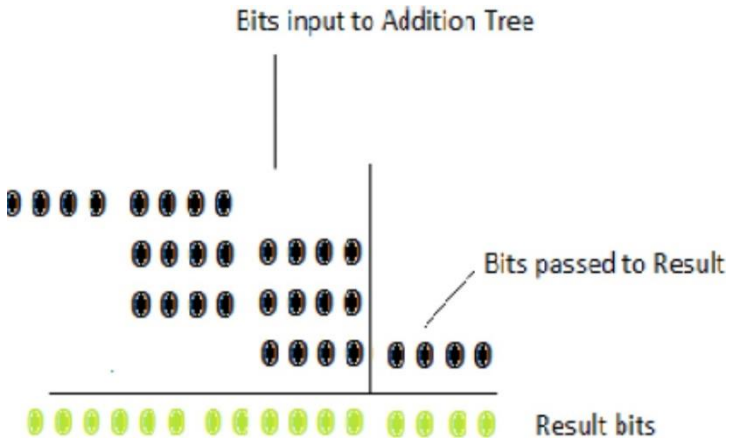


Figure 7: Addition of partial products in 8-bit Vedic multiplier

The 8-bit Vedic multiplier is implemented using four 4-bit Vedic multiplier blocks and three adders. Block diagram of this design is shown in Figure 8.

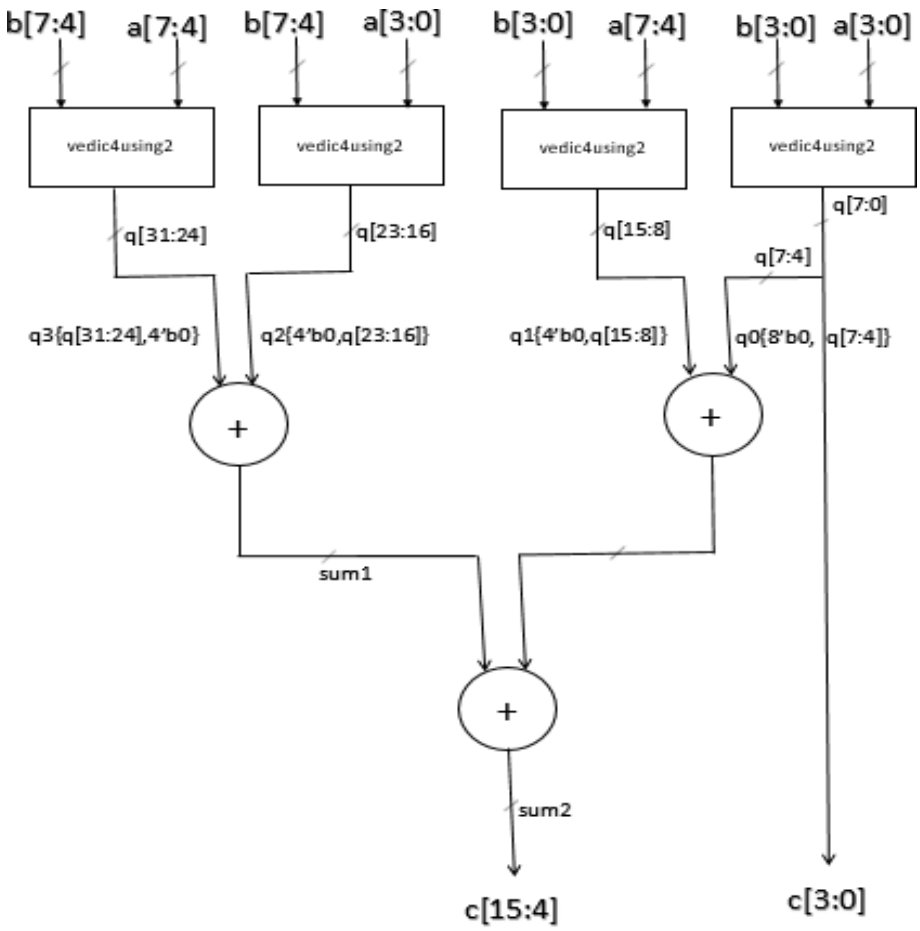


Figure.8: Block Diagram for 8-bit Multiplier[6]

Block Diagram and design steps for the proposed method:

For reducing the delay of 16-bit Vedic multiplier, it is implemented using four 8-bit Vedic multiplier blocks and three adders. Design steps are as shown in figure 9.

| | | | |
|---|---|---|---|
| A | = | A15A14A13A12A11A10A9A8 (X1) | A7A6A5A4A3A2A1A0 (X0) |
| B | = | B15B14B13B12B11B10B9B8B (Y1 ) | B7B6B5B4B3B2B1B0 (Y0) |

X1 X0
X Y1 Y0
---
F E D C

CP = X0 * Y0 = C
CP = X1 * Y0 + X0 * Y1 = D
CP = X1 * Y1 = F E
Where CP = Cross Product.

Figure.9: Design Steps for 16-bit Multiplier

The addition process in a 16 bit Vedic multiplier is as shown in figure 10.


Figure 10: Addition of partial products in 16-bit Vedic multiplier

Block diagram for the 16 bit multiplier is shown in figure 11.

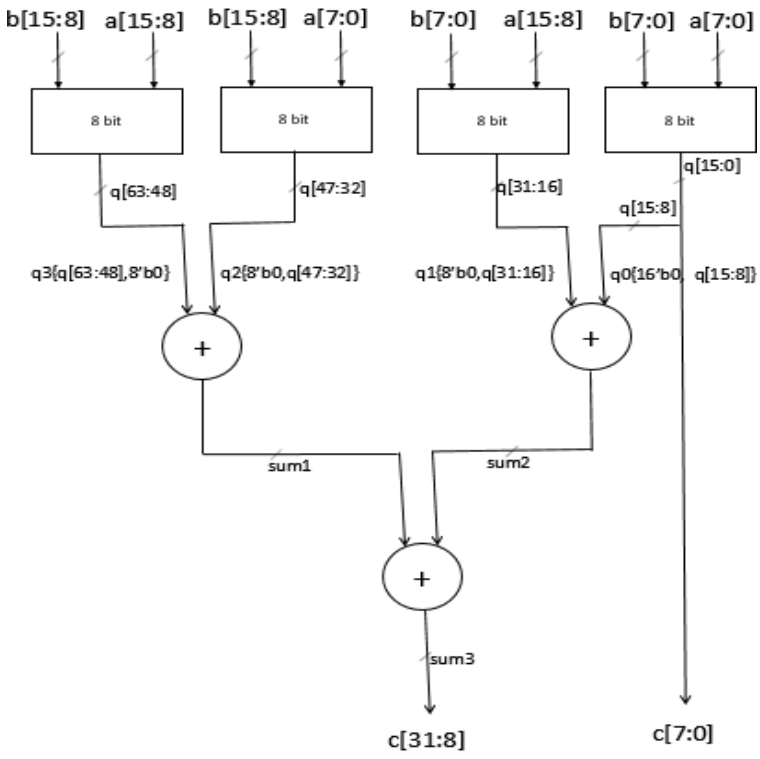


Figure 11: Block Diagram for 16-bit Vedic multiplier

**Results and Implementation:**

A modular design approach with regular design units which are simple and small is used to design 16-bit multiplier. The functional verification, logic synthesis and implementation are done using Xilinx Synthesis tool. Hardware used for the implementation of Vedic multiplier is Spartan-6 FPGA with TQG144 package and device number XC6SLX9, Seven Segment Display and Dip switches. The Urdhva Tiryagbhyam Sutra-based 2-bit multiplier serves as the design's fundamental building element.

The block diagram, Register Transfer Level (RTL) and simulation results obtained using iSim simulator are shown in Figure 12.

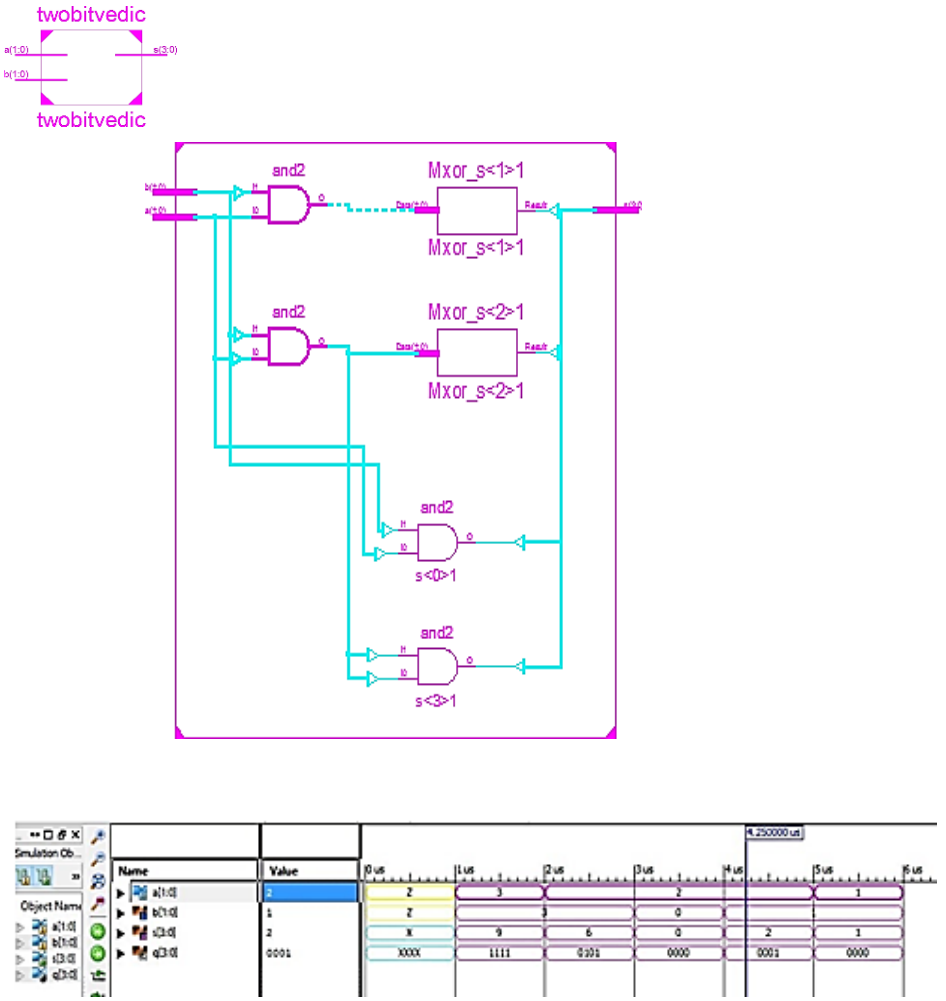


Figure 12. Block diagram, RTL schematic and simulation results of 2-bit multiplier

Using four 2-bit Vedic multiplier and three adders we can build a 4-bit Vedic multiplier. The block diagram, RTL schematic and ISim simulator outputs are shown in Figure 13.

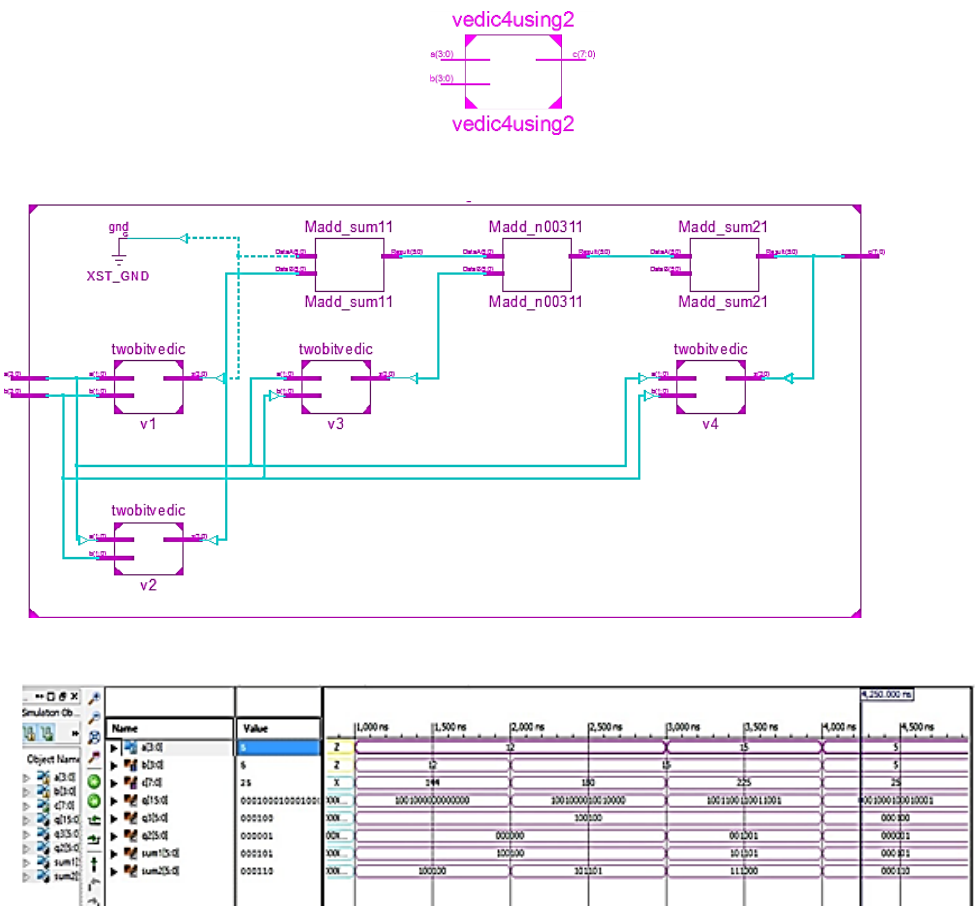


Figure 13: Block diagram, RTL schematic and simulation results of 4-bit multiplier

The 8-bit Vedic multiplier is implemented using four 4-bit Vedic multiplier blocks and three adders. The block diagram, RTL schematic and ISim simulator outputs are shown in Figure 14.

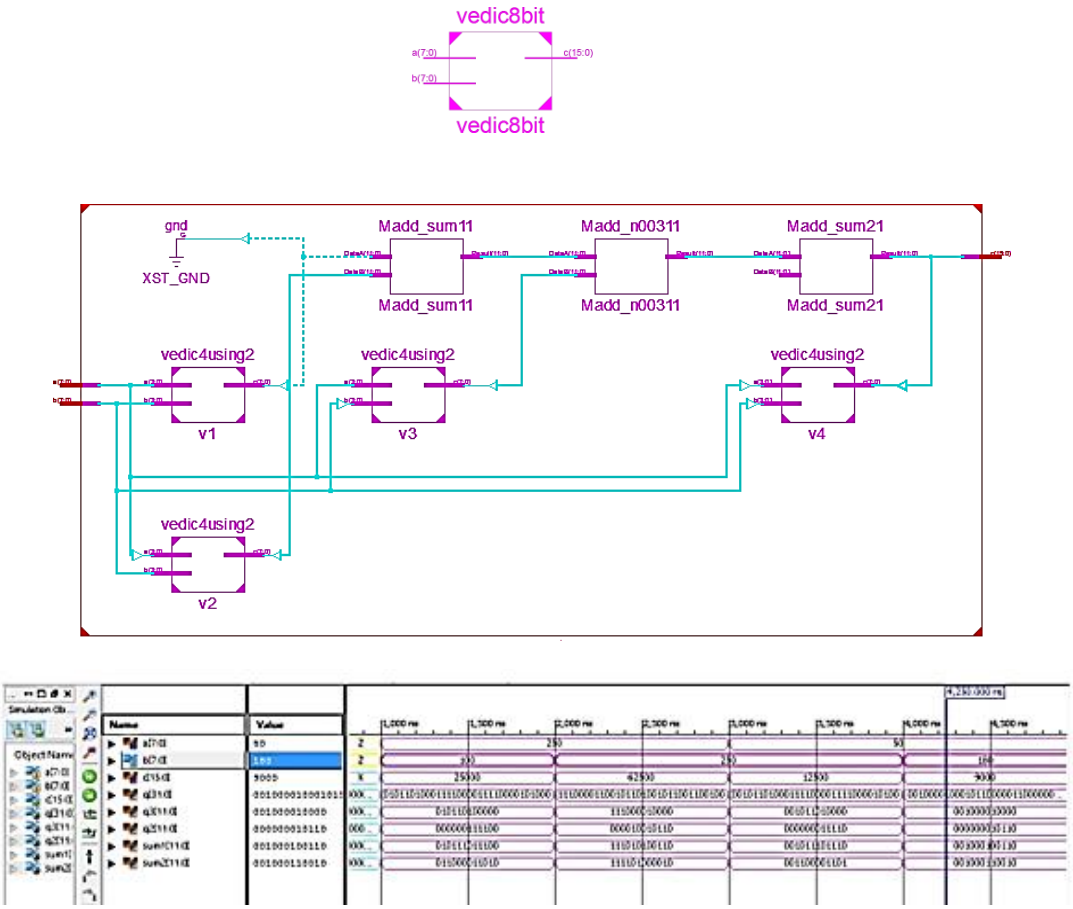


Figure 14: Block diagram, RTL schematic and simulation results of 8-bit Vedic Multiplier

The 16-bit Vedic multiplier is implemented using four 8-bit Vedic multiplier blocks and three adders. The block diagram, RTL schematic and ISim simulator outputs are shown in Figure 15.

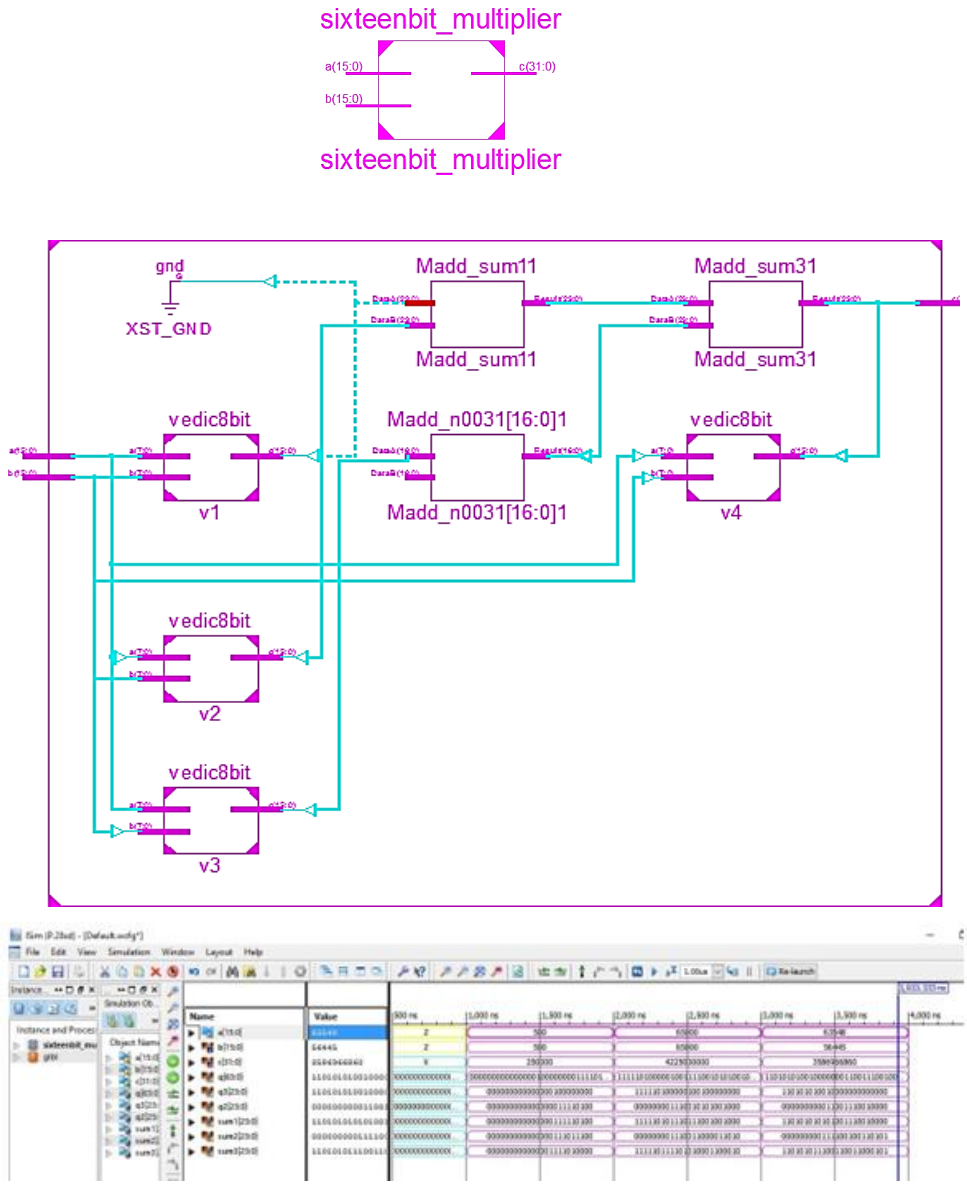


Figure 15. Block diagram, RTL schematic and simulation results of 16-bit Vedic Multiplier

A 16-bit Vedic multiplier using Urdhva Tiryagbhyam Sutra in binary is implemented using Verilog HDL. The synthesis is done using XST available with Xilinx ISE 14.2.The hardware implementation of a 8-bit Vedic multiplier shown in Figure 16 is done in Spartan6 FPGA. The bit programming file generated by the synthesis tool configures Spartan6.

Figure 16. FPGA implementation of a 8-bit Vedic multiplier

The device utilization and delay summary of 2, 4, 8 and 16-bit Vedic multipliers using Urdhva Tiryagbhyam are summarized in Table 1. The device utilization of implemented Vedic multiplier is compared with the Booth multiplier implemented before as shown in Table 2, Table 3, Table 4 for 4, 8 and 16 bit multipliers respectively.

Table 1: Device utilization and delay summary of Vedic multiplier using Urdhva Tiryagbhyam.

| | 2-bit Multiplier | 4-bit Multiplier | 8-bit Multiplier | 16-bit Multiplier |
|---|---|---|---|---|
| Delay | 6.25ns | 13.42ns | 16.57ns | 21.104ns |
| I/O Ports | 8 | 16 | 32 | 64 |
| No. of Slices | 4/5720 | 26/5720 | 128/5720 | 583/5720 |
| Memory Usage | 287164kB | 287612kB | 287740kB | 288444kB |

Table 2: Comparison of device utilization of 4-bit multiplier

| Parameter | Booth multiplier [9] | Vedic multiplier (This work) |
|---|---|---|
| Combination path delay | 13.224ns | 13.42ns |
| No. of Slices | 23 out of 3584 (0%) | 26 out of 5720 (0%) |
| No.of bonded IOBs | 16 out of 141 | 16 out of 102 |

Table 3: Comparison of device utilization of 8-bit multiplier

| Parameter | Booth multiplier [9] | Vedic multiplier (This work) |
|---|---|---|
| Combination path delay | 50.866ns | 16.570ns |
| No. of Slices | 185 out of 3584 (5%) | 128 out of 5720 (2%) |
| No.of bonded IOBs | 32 out of 141 | 32 out of 102 |

Table 4: Comparison of device utilization of 16-bit multiplier

| parameter | Booth multiplier [9] | Vedic multiplier (This work) |
|---|---|---|
| Combination path delay | 93.86ns | 21.104ns |
| No. of Slices | 718 out of 3584 (20%) | 583 out of 5720 (10%) |
| No.of bonded IOBs | 64 out of 141 | 64 out of 102 |

From Table 2 to Table 4, as the size of the multiplier increases, the increase indelay of the Vedic multiplier is much less compared to the Booth multiplier.

**Conclusion:**

The 16-bit multiplier design is implemented on Spartan6: device number XC6SLX9, package TQG144 with speed grade -2. It is designed using Urdhva Tiryagbhyam sutra from

"Vedic mathematics" concept. From the synthesis report, it can be seen that use of Vedic mathematics for the design of large size multipliers results in area efficient high-speed multipliers. The proposed work can be further improved for speed and power. It is possible to minimise the latency, power consumption, and hardware requirements by designing the multiplier using other sutras, such as the Nikhilam and Anurupye sutras. Faster multipliers that use less space and power must be put into practice. Using Urdhva Tiryagbhyam sutra, it is also possible to design multiplier that can-do floating-point operations using single precision and double precision formats. Vedic multipliers can be employed in the design of ALUs because of their high-speed multiplication. Faster Real Time Hardware Image Processing is effectively implemented with a low area and high-performance Vedic multiplier.

## References

[1] W. B. V. Kandasamy, Vedic Mathematics - 'Vedic' or 'Mathematics': A Fuzzy and Neutrosophic Analysis, 2006.

[2] V. Meghana, S. S., A. R., and C. Gururaj, "High speed multiplier implementation based on vedic mathematics," International Conference on Smart Sensors and Systems (IC-SSS), Bangalore, pp. 1–5, 2015.

[3] S. N. Gadakh and A. Khade, "Design and optimization of 16×16 bit multiplier using vedic mathematics," International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT), Pune, pp. 460–464, 2016.

[4] S. P. Pohokar, R. S. Sisal, K. M. Gaikwad, M. M. Patil, and R. Borse, "Design and implementation of 16×16 multiplier using Vedic mathematics," International Conference on Industrial Instrumentation and Control (ICIC), Pune, pp. 1174– 1177, 2015.

[5] K. D. R. C. Gangadhar and P. K. Korrai, "Fpga implementation of complex multiplier using minimum delay vedic real multiplier architecture," IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Elec- tronics Engineering (UPCON), Varanasi, pp. 580–584, 2016.

[6] R. Raju and S. Veerakumar, "Design and implementation of low power and high performance vedic multiplier," International Conference on Communication and Signal Processing (ICCSP), Melmaruvathur, pp. 0601–0605, 2016.

[7] S. V. Mogre and D. G. Bhalke, "Implementation of high speed matrix multi- plier using vedic mathematics on fpga," International Conference on Computing Communication Control and Automation, Pune, pp. 959–963, 2015.

[8] D. K. Kahar and H. Mehta, "High speed vedic multiplier used vedic mathemat- ics," International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, pp. 356–359, 2017.

[9] P. Soni, S. Kadam, H. Dhurape, and N. Gulavani, "Implementation of 16×16 bit multiplication algorithm by using vedic mathematics over booth algorithm," International Journal of Research in Engineering and Technology, vol. 04, pp. 371–376, May 2015.