8-Bit Vedic Multiplier

Archis Khuspe, Chirag Agrawal, Sanskar Agrawal

Abstract—The majority of scientific activities use floating-point calculations. It is vital to implement quicker multipliers that take up less space and use less power. Multipliers are essential components of any digital design. Despite the usage of many multiplication methods, the performance of Vedic multipliers has yet to get much attention. The application of 16 sutras or algorithms is involved in Vedic mathematics. One of these, the Urdhva Tiryagbhyam (UT) sutra for multiplication, has been studied in this paper. The UT sutra is employed because it reduces calculation time compared to traditional multipliers. Digital Signal Processing applications rely on the multiplication of binary floating-point integers. This sutra is utilised in the IEEE754 floating point multiplier implementation. The UT approach simplifies the multiplication of the Mantissa portion. Verilog HDL is used to architecturally represent the multipliers. Xilinx® ISE 14.2 is used for simulation and synthesis. The construction of an 8-bit high-speed floating point based on the UT sutra is described in the research.

Index Terms—Urdhva Tiryagbhyam (UT), Vedic Mathematics, Floating Point Multiplier, Verilog, Xilinx

- *Archis Khuspe is with the Vishwakarma Institute of Technology, Pune, India. E-mail: archis.khuspe20@vit.edu*
- *Chirag Agrawal is with the Vishwakarma Institute of Technology, Pune, India. E-mail: chirag.agrawal20@vit.edu.*
- *Sanskar Agrawal is with the Vishwakarma Institute of Technology, Pune, India. E-mail: sanskar.agrawal20@vit.edu*

1 INTRODUCTION

Multipliers play an important role in many computing applications. Floating point multiplication units are essential for modern multimedia and high-performance computing, such as graphics acceleration, signal processing, image processing, etc.

With advances in technology, many researchers have tried designing multipliers that offer the design targets such as high speed, low power consumption, regularity of layout, and hence less area or even a combination of them in one multiplier, thus making them suitable for various high speed, low power, and compact VLSI implementation and improve performance of floating-point computations. Floating point units are not only complex but also require more area and hence more power consumption as compared to fixed point multipliers. The complexity of the floating-point team increases, affecting the output's accuracy. Floating-point helps mitigate several representation problems because the conventional fixed point is restricted to a specific limit which prohibits it from representing very large or small numbers. Also, when two large numbers are separated, a fixed point is exposed to loss of precision. But as the precision increases, multiplier area, delay, and power increases drastically.

The most widely used standard for floating-point computation is the Binary Floating-Point IEEE-754 Standard. It specifies the format for the floating-point representation of numbers. In addition, this format also specifies floating-point arithmetic, interconversion between floating-point and integer formats, conversions among various floating-point forms, and handling of exceptions.

This paper proposes an 8-bit digital multiplier, which relies on the UT Sutra of Vedic Mathematics. Its implementation is done in Verilog HDL on the Xilinx® ISE 14.2 platform for the simulation and synthesis. The term UT (Urdhva Tiryagbhyam) means vertical and crosswise. It is identified as an Indian multiplier circuit that performs multiplication by dividing the multiplier circuit into smaller parts and synthesising their results. Two binary numbers are multiplied with this Sutra.

1.1 Vedic Mathematics

Vedic maths is an antiquated Indian arrangement of mathematics composed of Vedic sutras. It gives a streamlined and enhanced arrangement solution with regular numerical calculations. This Vedic calculation is applied to a few numerical tasks, such as arithmetic, geometric and trigonometric operations. One such operation is the Vedic multiplier which is utilised for the computerised multiplier in digital signal processing applications. The ancient system of Vedic Mathematics was rediscovered between 1911 to 1918 by Sri Bharati Krisna Tirthaji (1884-1960) from the Indian Sanskrit texts brought up from the Atharva Vedas. As per his research, all mathematics relies on sixteen Sutras or word formulas. These formulae describe how the mind naturally works and are excellent in directing the scholar to the appropriate solution method. The computerised multipliers dependent on the Vedic multiplier are the quickest, most reliable, proficient, and low-power. The system becomes faster when a delay is decreased by diminishing the partial products. Of the 16 sutras, the fourteenth sutra is the Urdhva Tiryakbhyam sutra, which is connected to the multiplication operation and is best appropriate for lesser magnitude values.

2 LITERATURE REVIEW

The floating-point multiplier and adder/subtractor units were studied by taking the number of pipeline stages of the units as a parameter and utilising throughput/area as the measure. The single precision FPM is made up of four distinct modules: Sign bit calculator, Unsigned Adder, Unsigned Multiplier and Normalizer. [1] This paper proposes the development of high-speed floating-point multipliers using Vedic Mathematics. Because the Vedic Multiplier (VM) has a regular structure, it can be laid out in a Silicon chip. Ripple Carry Adder (RCA) and Carry Look-ahead Adder (CLA) are two different types of adders used in two separate VMs (CLA). These two VM structures create a single precision floating point multiplier. Verilog HDL is used to architecturally represent the multipliers. Xilinx® ISE 14.2 is used for simulation and synthesis. A 24u24 unsigned multiplier, an unsigned adder, a sign bit calculator, and a normalizer are all included in a single precision FPM. [2] The methodical rounding tactic implemented comprises scheming a forecast algorithm and picking rounding digits, as well as creating a rounding database with all feasible precomputed significant values. The number of critical values for operative hardware implementation is reduced through the forecast. Rounding digits, lessen the complexity of the rounding logic. Hardware application influences prognosis and rounding digit selections. This rounding technique enables methodical hardware refinement and simple verification. [3] This research compares floating point implementation (24-bit) to standard multipliers centred around maths techniques of Vedic origin that use the multiplication method of IEEE 754. To build a NuN Multiplier (Vedic), this paper recommends the N/2uN/2 Multipliers and three adder architectures. It also reveals the way the adder project of a multiplier influences its latency. The multipliers run-on Xilinx ISE 10.1 and are written in Verilog. [4] This project used an IEEE 754-2008 compliant design for a fast-floating point multiplier. The design's major purpose is to make the multiplier quicker by lowering latency. Carry propagation in adders with the shortest power delay constant may reduce the delay. [5] This study demonstrates how to use a very efficient mixture of the vedic algorithms to limit the percentage surge in latency and extent of a floating-point multiplier. Pipelining methods may enhance the model's latency, and efficient truncation and rounding procedures can improve the result's exactness. [6] A high-speed FP FW RLNS multiplier with reduced truncation errors is built. Because of their uniform structure and steadily diminishing area, power, and delay, RLNS multipliers were chosen. In such FWMs, RLNS also has the benefit of less latency; delay is caused by the time required for the execution process or by high system performance. It is also demonstrated that the suggested Taylor series error approximation technique may minimise quantization or truncation errors by up to 89 percent compared to the current method error. The Xilinx ISE 14.7 simulator was used to generate the simulation results. Based on power, area, and delay with varied FWM, the suggested RLNS multiplier designs were developed. Compared to existing multipliers, the proposed RLNS multiplier has a high speed. The inclusion of RLNS increased the FWM speed as experimental findings validated the many selections made to improve the performance and space requirements of FP units and may be used as a reference by hardware designers for implementing this type of DSP application in FW property in future work. [7] This study provides a revolutionary complex number multiplier outline based on traditional Indian Vedic Mathematics recipes, which are suitable for fast complex number-crunching circuits with widespread applicability in VLSI and signal processing. The code was written in HDL and Xilinx. This new engineering combines the advantages of Vedic science for growth, which experiences phases and midway item reduction. The suggested Vedic multiplier has a delay of 6.216ns and a forced usage of 0.027mW. According to the overview, the proposed design and simulation of 16 Bit Vedic Multiplier employing Urdhva Tiryakbhyam Algorithm 551 Architecture requires 203 exclusive reasoning components. The advantages of this proposed engineering include rate and area proficiency (1ess assets used, for example, a reduced number of multipliers and adders) and flexibility in de layout. [8] The pipeline-based Vedic multiplier achieves high data speeds. Compared to the Traditional Vedic Multiplier, the suggested approach has higher throughput. While computing the products for present values, data for the subsequent multiplication is fetched. Although the complexity of the circuit design has grown here, it is more favourable when compared to current technologies and trend needs. By incorporating sophisticated pipeline techniques into the circuit, data speed may be increased even more. Power optimization techniques can also be used to minimise power usage. Modifying the vehicle course transistor level may reduce the cell area, reducing power consumption. [9] This work tests the multiplier of decimal and Vedic origin (16 bits). The Vedic multiplier has a capacity of 0082w. The 0081w, on the other hand, represents the capacity of the multiplier (decimal). The multiplier (decimal) outperforms the multiplier (Vedic). The delays are 52249ns (decimal) and 11198ns (Vedic). In terms of uncertainty, the multiplier (decimal) is slower than the multiplier (Vedic), although using less energy. The paper tackles energy multiplier improvement, which would benefit DSP applications, image processing, all processors, and so on. [10] This paper focuses on the implementation of optimised IEEE 754 single precision floating point multipliers on the hardware. Modelsim is used for simulation and Xilinx for synthesization of designing parts. The main motive of the paper is to increase the speed and reduce the area of the multiplier. Speed of multiplication is increased with the use of MCLA in CSD instead of CLA. MCLA is 31.25% faster than CLA with the use of 12 adders for mantissa multiplication. Also by choosing proper modules for latency, area and delay the speed performance is improved [11]

3 METHODOLOGY

3.1 8-bit Vedic Multiplier

The outcome of these Vedic multipliers is added by altering the logic levels of the carry adder of the ripple type. The number-multiple generators of binary numbers (8-bit) for adding partial products utilise tetrad (4) number multiple generators (4x4) of Vedic methodology, which uses the carry skip technique as well as UT sutra.

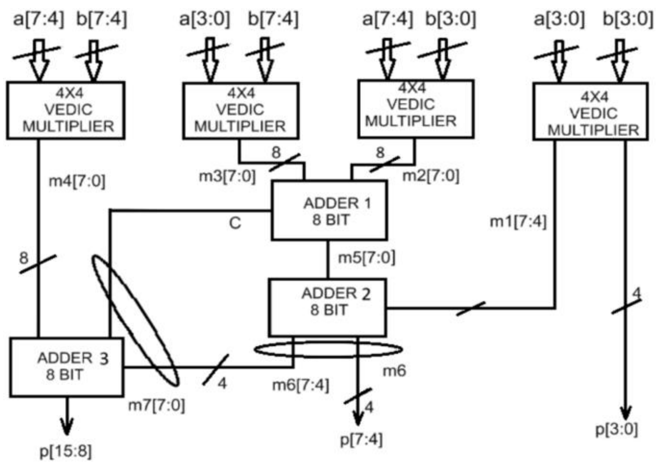


Fig. 1. Block diagram of the 8x8 multiplier [Number-multiple generator (8-bit)]

The 8-bit input sequence is divided into two 4-bit integers and transmitted to the 4-bit multiplier blocks (a[7:4] & b[7:4], a[3:0] & b[7:4], a[7:4] & b[3:0], a[3:0] & b[3:0]). The four multipliers used (in figure 1) are equivalent and yield 8-bit intermediate results when paired with overlapping circuitry and three modified parallel adders (ADDER-1, ADDER-2, and ADDER-3). As illustrated in Figure 2, the partial products of the four multipliers are separated into four zones. The four LSB product bits P[3:0] are directly taken from one of the multipliers. Because the output of the second and third multiplier blocks overlaps, the output of the second and third multiplier blocks is added using ADDER-1. Using ADDER-2, the higher-order bit of the first multiplier block is then added to the overlapping total. The higher-order bit of the first multiplier block is then added to the overlapping total using ADDER-2, obtaining the result P[7:4]. Finally, P[15:8] MSB bits are created by adding the fourth multiplier output to the carry from ADDER-1 (added at the fifth bit position) and higher order bits from ADDER-3 (acts as a lower nibble of addend).

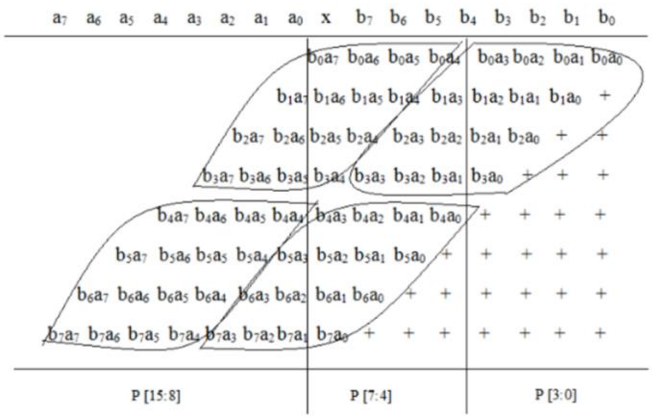


Fig. 2. . Separation of the partial products

3.2 Design of 4-bit Multiplier

The UT sutra was used to create the number-multiple generator (4-bit of Vedic methodology, which uses the skip approach for adding partial multiples. In number-multiple generators of the Vedic process, the carry from adding each partial multiple is transferred to the subsequent bit. The cross-wise system and the vertical approach of determining the o/p value of the Vedic multiplier are depicted.

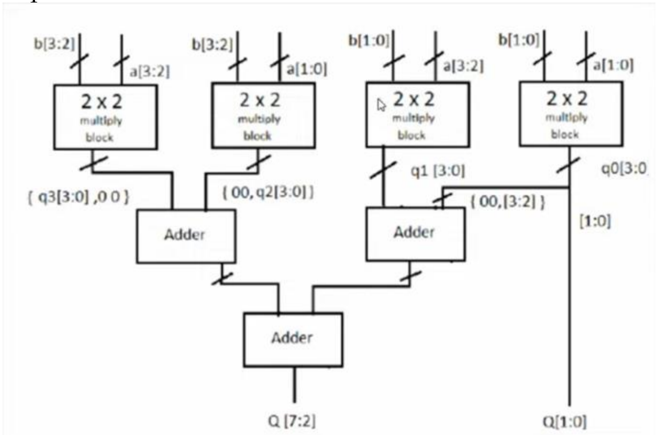


Fig. 2. . 4-bit Vedic Multiplier [Number-multiple generator (4-bit) of Vedic methodology]

3.3 Design of 2-bit Multiplier

Based on the UT sutra, it is a Number-multiple generator (2-bit). It comprises four AND gates, each with two I/Ps. It also features a Half Adder. The starting point for the 8-bit Vedic multiplier is also the primary bit of the Number-multiple generator (4-bit).

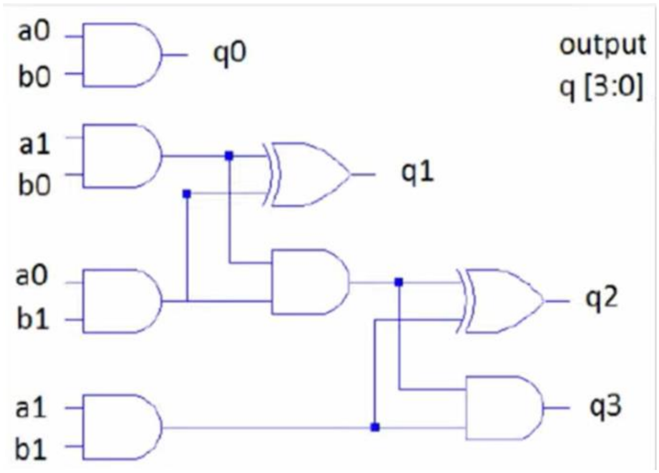


Fig. 3. 2-bit Vedic Multiplier [Number-multiple generator (2-bit) of Vedic methodology]

3.4 Elements Used in the 8-bit multiplier

3.4.1. AND Gate

The AND gate is a rudimentary gate that is digital. It has dual I/Ps and single O/P, defined and their workings visualised using the truth table. The circuit integration of the carrying choose adder {element of the Number-multiple generator (8-bit)]} and the integration of the Number-multiple generator (2-bit) in this project employs the AND gate.

Input	**Input**	**Output**
X	*Y*	*X and Y*
0	0	0
0	1	0
1	0	0
1	1	1

Table 1. Truth table of the AND gate

3.4.2. Half Adder

An adder is crucial for any circuit. The situation is no different, even with a Vedic number-multiple generator. In the Half adder, dual I/P bits are added, but what gets ignored is the carry. The inclusion of dual bits is what the output is established upon. The fact that the incorporation of the circuit is more straightforward and extra efficient separates the typical half-adder and this circuit.

3.4.3. Full Adder

The full adder adds binary integers by applying a similar structuring procedure, a central constituent of Vedic Multiplier, whether a 4x4 or an 8x8. It aims to account for carry-out and carry-in.

3.4.4 Carry Save Adder

The Number-multiplier's (8-bit) integration has a critical component: the carry save adder. AND gates and XORs are the carry-keeps adder demonstration pieces. It simplifies hardware setups significantly. Three NORs are used to create the AND gate.

4 Novelty

As the number of bits required for Vedic multiplication increases, there is a significantly less increase in gate delay along with the space required when compared with other multiplier architectures. Using such a design, many resources on FPGA can be saved for DSP applications.

5 Results

The proposed 8-bit multiplier is written in Verilog HDL, simulated with the Xilinx ISim simulator, synthesised with the Xilinx XST for Spartan 6, and validated for the inputs provided. The Verilog HDL test bench is used to create inputs.

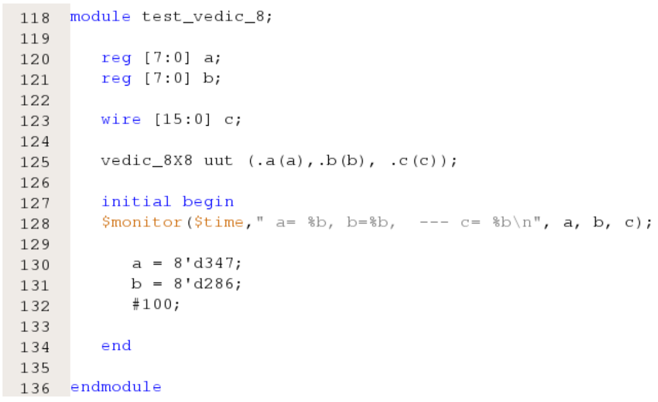
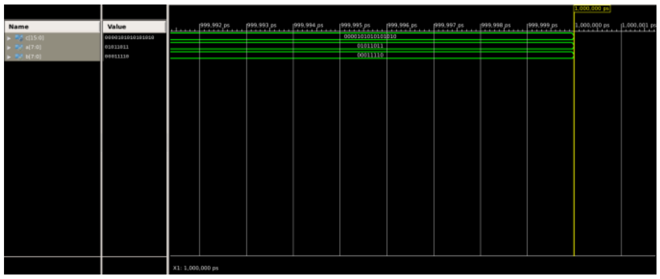

```
module test_vedic_8;

   reg [7:0] a;
   reg [7:0] b;

   wire [15:0] c;

   vedic_8X8 uut (.a(a),.b(b), .c(c));

   initial begin
   $monitor($time," a= %b, b=%b,  --- c= %b\n", a, b, c);

      a = 8'd347;
      b = 8'd286;
      #100;

   end

endmodule
```

Fig. 4. Testbench of the 8-bit multiplier specifying the 2 input numbers

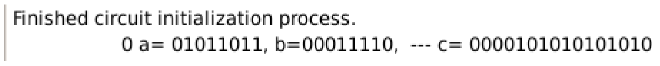
Inputs:
a = "101011011",
b = "100011110"

Product:
p = "110000011110101010"



Name	Value
c[15:0]	0000101010101010
a[7:0]	01011011
b[7:0]	00011110

Fig. 5 Testbench simulation on Xilinx



```
Finished circuit initialization process.
                    0 a= 01011011, b=00011110,  --- c= 0000101010101010
```

Fig. 6. Output in the Verilog console

7 CONCLUSION

The floating point multiplier which is required in various electronic devices is proposed in this paper using an algorithm belonging to vedic mathematics. Generally, floating point multipliers consume much space and power from the device as calculating the product of float point is a complex thing. This is challenging for production of small size and low energy consuming devices. To solve this issue and to save space and avoid energy wastage, an Urdhva Tribhagyam algorithm from vedic maths is applied which makes it possible to calculate multiplication of floating point numbers consuming less space and power of the electronic device. This algorithm not only helps in the issue of space and power but also increases the speed of calculation. So, the implementation of multipliers based on this algorithm in devices can solve various problems.

ACKNOWLEDGMENT

The authors wish to express gratitude and sincerest regards to Prof. (Dr.) Abhay Chopde, who made this work possible. His direction and guidance helped them through all the stages of drafting this paper.

The authors would also like to acknowledge their family and friends for their constant support and understanding throughout the research.

REFERENCES

[1] Gokul Govindu, L. Zhuo, S. Choi and V. Prasanna, "Analysis of high-performance floating-point arithmetic on FPGAs," 18th International Parallel and Distributed Processing Symposium, 2004. Proceedings., 2004, pp. 149-, doi: 10.1109/IPDPS.2004.1303135.

[2] Anjana, Sa, Ca Pradeep, and Philip Samuel. "Synthesize of high speed floating-point multipliers based on Vedic mathematics." Procedia Computer Science 46 (2015): 1294-1302.

[3] N. T. Quach, N. Takagi and M. J. Flynn, "Systematic IEEE rounding method for high-speed floating-point multipliers," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 12, no. 5, pp. 511-521, May 2004, doi: 10.1109/TVLSI.2004.825860.

[4] M. S. Athira Menon and R. J. Renjith, "Implementation of 24 bit high speed floating point vedic multiplier," 2017 International Conference on Networks & Advances in Computational Technologies (NetACT), 2017, pp. 453-457, doi: 10.1109/NETACT.2017.8076814.

[5] Sunesh N.V and Sathishkumar P, "Design and implementation of fast floating point multiplier unit," 2015 International Conference on VLSI Systems, Architecture, Technology and Applications (VLSI-SATA), 2015, pp. 1-5, doi: 10.1109/VLSI-SATA.2015.7050478.

[6] S. Arish and R. K. Sharma, "An efficient floating point multiplier design for high speed applications using Karatsuba algorithm and Urdhva-Tiryagbhyam (UT) algorithm," 2015 International Conference on Signal Processing and Communication (ICSC), 2015, pp. 303-308, doi: 10.1109/ICSPCom.2015.7150666.

[7] Rubia, J. Jency, and G. A. Sathish Kumar. "A high-speed fixed width floating-point multiplier using residue logarithmic number system algorithm." The International Journal of Electrical Engineering & Education 57, no. 4 (2020): 361-375.

[8] Agarwal, Aditya, Jay Prakash Narayan Verma, and Nishant Srivastava. "Design and Simulation of 16 Bit Vedic Multiplier using Urdhva Tiryakbhyam Algorithm."

[9] RAO, Y. NARASIMHA, DR GSVP RAJU, and PENMETSA V. KRISHNA RAJA. "DESIGN OF HIGH SPEED VEDIC MULTIPLIER WITH PIPELINE TECHNOLOGY." Journal of Theoretical and Applied Information Technology 67, no. 3 (2014).

[10] Bairwa, Vijendra, and Poonam Jindal. "Analysis of 16x16 Vedic and 16 Bit Floating Point Multiplier-A Comparative Study." Available at SSRN 4043575 (2022).

[11] S. V. Siddamal, R. M. Banakar and B. C. Jinaga, "Design of High-Speed Floating Point Multiplier," 4th IEEE International Symposium on Electronic Design, Test and Applications (delta 2008), 2008, pp. 285-289, doi: 10.1109/DELTA.2008.19.