ISSN:2230-9
993

32-bit RISC Processor Using Vedic Mathematics

Navaneetha A Nair
B.Tech Student
Department of Electronics Engineering
Saintgits College of Engineering
Kottayam,Kerala
navaneethakathu@gmail.com

P S Sreeya
B.Tech Student
Department of Electronics Engineering
Saintgits College of Engineering
Kottayam,Kerala
pssreeya243@gmail.com

Ribin K Thomas
B.Tech Student
Department of Electronics Engineering
Saintgits College of Engineering
Kottayam,Kerala
ribinkthomas37@gmail.com

Vijay R
B.Tech Student
Department of Electronics Engineering
Saintgits College of Engineering
Kottayam,Kerala
vijayr2906@gmail.com

Anish M George
Assistant Professor
Department of Electronics Engineering
Saintgits College of Engineering
Kottayam,Kerala
anish.mg@saintgits.org

***Abstract*—RISC (Reduced Instruction Set Computer) architecture provides high performance by executing an instruction by a single action and thereby operation time is optimised. A 16-bit RISC processor can handle a large instruction set by a simple design using Xilinx Vivado 2018.3. But it's not possible to handle graphics applications. Only a 32-bit RISC processor can be a solution to overcome the limitations of a 16-bit RISC processor. But the speed of a 32-bit processor is low compared to a 16-bit processor due to complexity. By taking the concept of Vedic Mathematics to the 32-bit RISC processor which will have a great advantage over normal 32-bit RISC processor in critical path delay which will make the processing faster. The main achievement of this work is that the ALU (Arithmetic Logic Unit) and MAC (Multiplier Accumulator) unit are implemented using vedic sutras.**

Keywords—RISC, Vivado 18.3, ALU, MAC, Vedic Mathematics

I INTRODUCTION

There are mainly two architectural designs for the Central Processing Unit. They are RISC and CISC. RISC is Reduced Instruction Set Computer, which is a design methodology that gives better performance, improves the speed of operation and is capable of executing more instructions with simple designs. Even though RISC has large code size and requires memory, it has a greater number of registers, fixed size of instructions and has optimised use of clocks which is also three to four times that of CISC architecture. Here in this paper, we have designed a 32-bit RISC processor using vedic mathematics using Verilog HDL and further simulated and synthesised the same using Xilinx Vivado 2018.3.

RISC Processor architecture is a system which makes utilisation of synchronous execution of instruction and thereby improves the working efficiency of the processor. The main aim of the processor is to execute every instruction as per the machine language. It consists of an ALU unit, which performs both arithmetic and logical operations in the processor. This Unit is designed for performing various operations in various numbers using some instruction sets. It consists of opcodes and operands. The opcode tells ALU which instruction is to be performed. Register Bank is the data holding place of the processor. The results of ALU operations are stored first in an accumulator which is later on placed in a storage register and it checks the bits and indicates whether the operation was performed successfully. If not successfully executed then some type of status will be shown i.e. even known as Z-Flag or status register. Its function is to execute programs and operate efficiently for the data stored in memory. Instruction Register (IR), stores instructions, which are basically the commands for operation. And RISC architecture is illustrated in fig:1

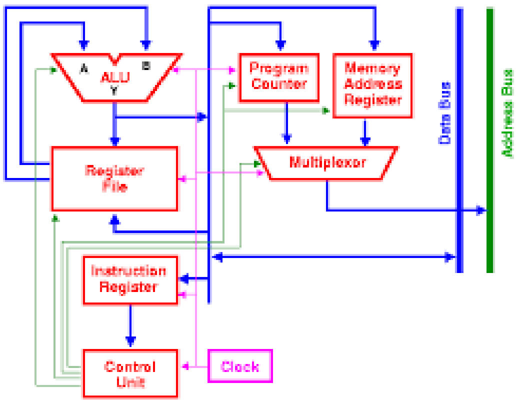


Fig. 1. RISC architecture

The control unit holds the instruction to be executed. In CPU, the registers such as address register, data register and an instruction register are present. The performance of the CPU is to fetch, decode and execute the operations on memory according to the registers. The task of IR includes decoding the op-code, determining the instruction, determining which operands are in memory, retrieving the operands in memory then assigning a command to a processor to execute the instruction. Program Counter, which contains the address of next instruction to be fetched. The instruction is loaded into the register after the processor fetches it from the memory location the PC points to. The Control Unit is the essential part of any system because it generates the timing and control signals for operations performed by CPU. The multiplexer block works as input selector. It can control wires which act as select lines. It is a circuit which takes multiple inputs and gives the single output. The MAR is also called an address buffer; the address in the program counter is applied to memory so after the increment in PC to the next address the current instruction is stored in the Memory location.

Vedic Mathematics is derived from the ancient Indian scriptures. It gives mathematical outputs and basic understandable structures. The word “Vedic” is derived from the word “Veda” which means the storage facility of all the information. It mainly contains 16 sutras. Here in this paper, we are discussing on the 32-bit RISC Processor with Vedic Multiplier designed using one among the 16 sutras called “Urdhva Triyakbhyam”. S Lad and V.S Bendre [1] has explained the comparison of various types of multipliers and proved how vedic multiplication is efficient than tree and array multiplication. Abhyarthana Bisoyil, MituBaratl, Manoja Kumar[7] had done a comparison of vedic multiplier with normal multiplier in their paper “Comparison of 32-bit Vedic Multiplier with a conventional multiplier”.

The main aim of this paper is to advance the characteristics of 32-bit RISC processors by using Vedic Mathematics when compared to 16-bit Processors and to increase the efficiency and accuracy of 32-bit processors.

II LITERATURE REVIEW

1. Design and Comparison of Multiplier using Vedic Sutras by Lad, Shraddha, and Varsha S. Bendre[8] :In this paper they implemented highly efficient Vedic Multiplier using various Vedic Sutras and analysed different performance parameters like area,speed and delay. It is observed that 70.26% of reduction in area is achieved using Urdhwa Tiryakbhyam sutra with 90.41% increase in speed as compared to Ekadhikena Purvena Sutra.

2. High speed vedic multiplier designs-A review by Bansal, Yogita, Charu Madhu, and Pardeep Kaur[9] : Vedic Multiplier is seen to be efficient in speed, power and area in digital designs with respect to other multipliers. Compressor based Vedic multiplier with Urdhva Tiryakbhyam sutra is seen as a promising technique in terms of speed and area and came to the conclusion that vedic multiplier can be used in ALU and accumulator designs.

3. Design of Efficient 16-bit Vedic Multiplier by Chowdary, K. Keshav Sai, et al[10]:The concept behind vedic multiplier is that as the number of multiplication bits increases the timing delay also increases in normal multiplier.But when we use Vedic sutras delay can be minimised.

4. Comparison of a 32-bit Vedic multiplier with a conventional binary multiplier by A. Bisoyi, M. Baral and M.K.Senapati[7] : Inorder to develop a digital multiplier,Urdhvatriyakbhyam sutra is used to implement vertical and crosswise operations.After implementation, it has been observed that the number of I/Os needed for 32-bit

Vedic multiplier and conventional binary multiplier are 128 out of 232 because of which the utilisation becomes 55% for both of the multipliers.

From these reviews, we come to the conclusion that the performance of 32-bit RISC Processor can be improved by incorporating Vedic sutras.Out of the 16 sutras,we found Urdhvatriyakbyam Sutra can be used for designing highly efficient RISC Processorwhich will meet all the performance parameters.

III PROPOSED METHODOLOGY

All the signal processing applications require multiplication as a basic operation. As study says multiplication needs more silicon area and it consumes more and more power due to the switching activity so they will create the longest path via adders and carry. So in order to overcome this situation Vedic multipliers can be implemented in processors.So when we compare the conventional processor and the processor with vedic multiplier, it is noticed that the performance, reduction in power consumption and speed of operation has been improved. In any case, the Vedic structure isn't just an accumulation of quick techniques; it is a framework, a brought together methodology. Vedic multipliers can be implemented using vedic sutras, which reduces both critical path and dynamic power. Therefore, the detailed study of designing the multiplier and comparison using various sutra is performed using Vedic mathematics algorithms.

The word “Vedic” is derived from the word “Veda” which means the store-house of knowledge. Vedic mathematics is mainly based on 16 Sutras (or aphorisms) dealing with various branches of mathematics like arithmetic, algebra, geometry etc. These Sutras along with their brief meanings are enlisted below alphabetically.

1. (Anurupye) Shunyamanyat – If one is in ratio, the other is zero.

2. Chalana-Kalanabyham – Differences and Similarities.

3. Ekadhikina Purvena – By one more than the previous One.

4. Ekanyunena Purvena – By one less than the previous one.

5. Gunakasamuchyah – The factors of the sum are equal to the sum of the factors.

6. Gunitasamuchyah – The product of the sum is equal to the sum of the product.

7. Nikhilam Navatashcaramam Dashatah – All from 9 and last from 10.

8. Paraavartya Yojayet – Transpose and adjust.

9.Puranapuranabyham – By the completion or noncompletion.

10. Sankalana- vyavakalanabhyam – By addition and by subtraction.

11. Shesanyankena Charamena – The remainder by the last digit.

12. Shunyam Saamyasamuccaye – When the sum is the same, that sum is zero.

13. Sopaantyadvayamantyam – The ultimate and twice the penultimate.

14. Urdhva-tiryagbhyam – Vertically and crosswise. 15. Vyashtisamanstih – Part and Whole.

16. Yaavadunam – Whatever the extent of its deficiency.

The sutra used in design of multiplier for the proposed system is “UrdhvaTriyakbhyam” sutra. The UrdhvaTriyakbhyam sutra is applied to multiplication which follows the vertically and crosswise pattern. The multipliers, with low power requirement and maximum speed, give information of “Urdhva-Tiryagbhyam” algorithm of ancient Indian Vedic mathematics, which has utilised for multiplication to improve speed and area.

In the proposed design an array multiplier in the Arithmetic Logic Unit (ALU) is replaced by a vedic multiplier inorder to improve the speed of operation, performance and to reduce power consumption. In the comparison of simulation results of Vedic ALU and MAC design done by Yadav A. and Bendre. V [1] with the existing ALU and MAC results, it is observed that there is 44% savings in power in case of MAC and that of 12% in case of ALU is achieved compared to conventional ALU and MAC respectively. Also the delay is reduced by 45% in case of MAC and that of 35% in case of ALU in comparison with conventional ALU and MAC correspondingly. This has led to the design of a 32-bit processor using vedic mathematics. Here, initially we have designed a conventional 32-bit RISC Processor. Then we have designed a 32-bit vedic multiplier and compared the performance, speed and power consumption of the designed vedic multiplier with a normal 32-bit array multiplier which is commonly being used in the conventional processor. By comparing the two we came to the conclusion that vedic multiplier shows higher performance than array multiplier .So Vedic Multiplier can be implemented in a 32-bit RISC Processor to improve its performance.

As mentioned above “Urdhva Triyakbhyam” Sutra is used in our design as well. As Kamaraj, A., Parimalah A.D.

and Priyadharshini, V [2] mentioned in their journal out of the 16 sutras, UrdhvaTriyakbhyam Sutra results in faster outcomes by increasing the accuracy and speed of multipliers. We have implemented vedic multiplier using the sutra - UrdhvaTriyakbhyam Sutra.

IV SIMULATION AND SYNTHESIS RESULTS

The simulation and synthesis results of different units in the CPU of the normal 32 bit RISC processor and area, power and delay comparison of normal 32-bit multiplier and vedic 32-bit multiplier are shown below.

A. *ALU Unit*

In the design of ALU, we have used XilinxVivado 18.3 for the synthesis and simulation. We have given 32-bit inputs a, b and 6-bit opcode i.e alufn[6:0].We obtained the output otp[] and an overflow depending on the arithmetic and logical operations performed. The Synthesis and Simulation Results of the ALU are given in Figure 2 and 3 respectively.

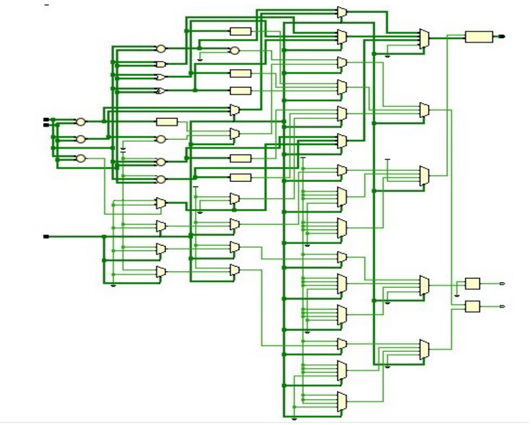
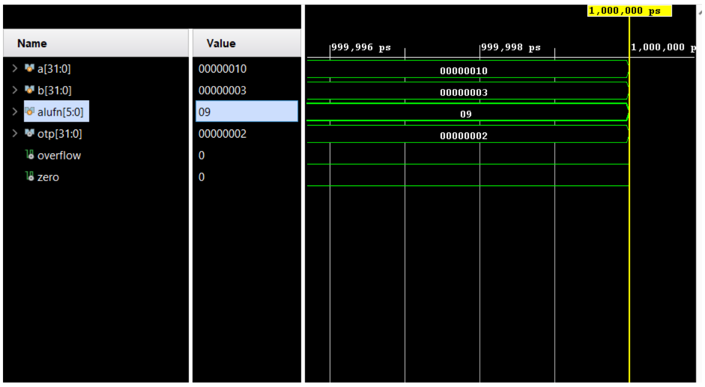


Fig. 2. ALU Synthesis Result



Name	Value
a[31:0]	00000010
b[31:0]	00000003
alufn[5:0]	09
otp[31:0]	00000002
overflow	0
zero	0

Fig. 3. ALU Simulation Result

B. *CPU*

In the design of CPU, we have called all the functions such as ALU functions, Control unit, register file, program counter and multiplexer through the verilog code. All the units are instantiated within the CPU module. In this way we have synthesised and simulated our CPU. The Synthesis and Simulation Results of the CPU are given in Figure 4 and 5 respectively.

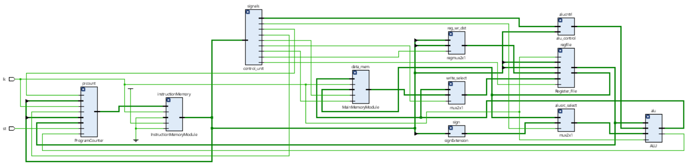
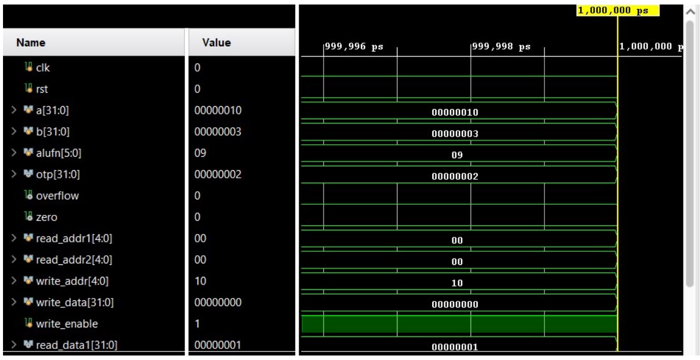


Fig. 4. CPU Synthesis

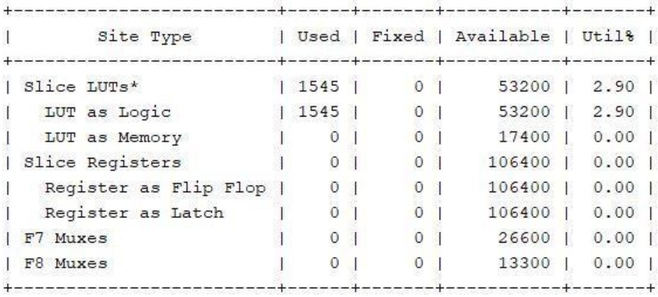


Name	Value
clk	0
rst	0
a[31:0]	00000010
b[31:0]	00000003
alufn[5:0]	09
otp[31:0]	00000002
overflow	0
zero	0
read_addr1[4:0]	00
read_addr2[4:0]	00
write_addr[4:0]	10
write_data[31:0]	00000000
write_enable	1
read_data1[31:0]	00000001

Fig. 5. CPU Simulation

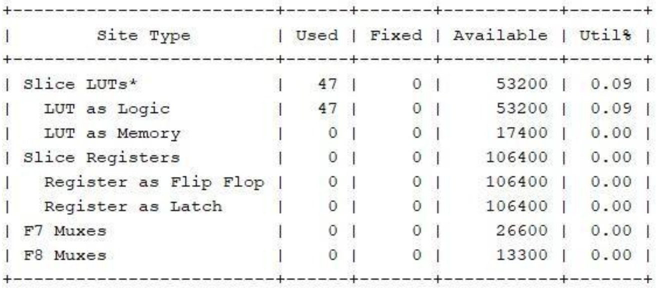
C. *32-bit Multiplier v/s 32-bit Vedic Multiplier*

The comparison of 32-bit RISC multiplier and 32-bit vedic multiplier is necessary to know the LUT numbers and the critical path delay of the processors, as the multiplier is a part of ALU, so we get an idea about normal 32-bit RISC processor and Vedic 32-bit RISC processor.



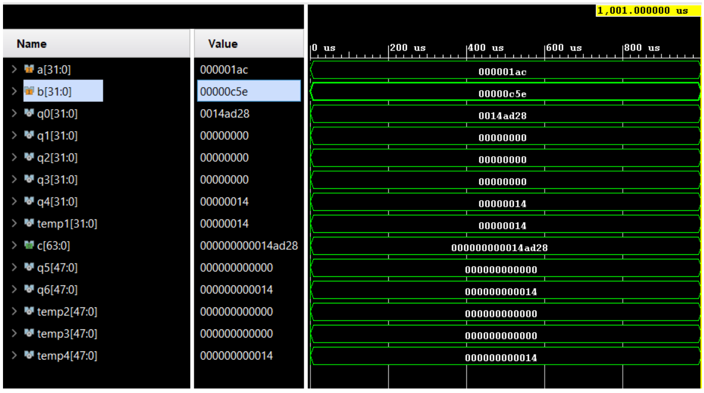
```
+-------------------------+------+-------+-----------+-------+
|        Site Type        | Used | Fixed | Available | Util% |
+-------------------------+------+-------+-----------+-------+
| Slice LUTs*             | 1545 |     0 |     53200 |  2.90 |
|   LUT as Logic          | 1545 |     0 |     53200 |  2.90 |
|   LUT as Memory         |    0 |     0 |     17400 |  0.00 |
| Slice Registers         |    0 |     0 |    106400 |  0.00 |
|   Register as Flip Flop |    0 |     0 |    106400 |  0.00 |
|   Register as Latch     |    0 |     0 |    106400 |  0.00 |
| F7 Muxes                |    0 |     0 |     26600 |  0.00 |
| F8 Muxes                |    0 |     0 |     13300 |  0.00 |
+-------------------------+------+-------+-----------+-------+
```

Fig. 6. Resource utilization of 32-bit multiplier



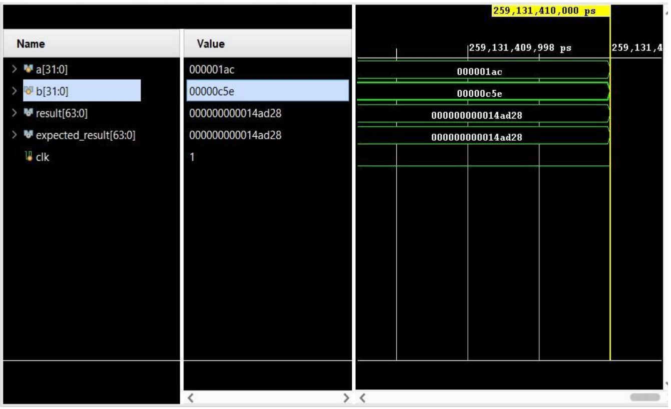
```
+-------------------------+------+-------+-----------+-------+
|        Site Type        | Used | Fixed | Available | Util% |
+-------------------------+------+-------+-----------+-------+
| Slice LUTs*             |   47 |     0 |     53200 |  0.09 |
|   LUT as Logic          |   47 |     0 |     53200 |  0.09 |
|   LUT as Memory         |    0 |     0 |     17400 |  0.00 |
| Slice Registers         |    0 |     0 |    106400 |  0.00 |
|   Register as Flip Flop |    0 |     0 |    106400 |  0.00 |
|   Register as Latch     |    0 |     0 |    106400 |  0.00 |
| F7 Muxes                |    0 |     0 |     26600 |  0.00 |
| F8 Muxes                |    0 |     0 |     13300 |  0.00 |
+-------------------------+------+-------+-----------+-------+
```

Fig. 7. Resource utilization of 32- bit Vedic multiplier



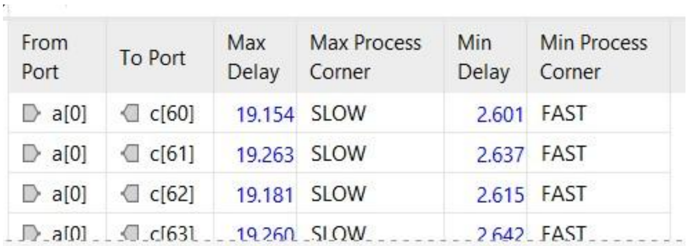
Name	Value
a[31:0]	000001ac
b[31:0]	00000c5e
q0[31:0]	0014ad28
q1[31:0]	00000000
q2[31:0]	00000000
q3[31:0]	00000000
q4[31:0]	00000014
temp1[31:0]	00000014
c[63:0]	000000000014ad28
q5[47:0]	000000000000
q6[47:0]	000000000014
temp2[47:0]	000000000000
temp3[47:0]	000000000000
temp4[47:0]	000000000014

Fig. 8. Simulation of 32 -bit Array multiplier



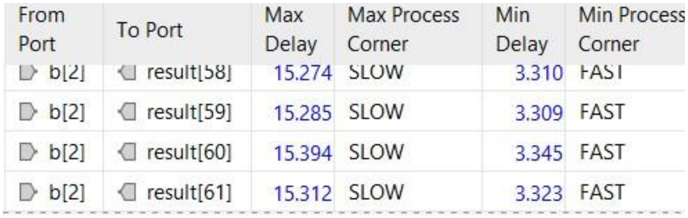
Name	Value
a[31:0]	000001ac
b[31:0]	00000c5e
result[63:0]	000000000014ad28
expected_result[63:0]	000000000014ad28
clk	1

Fig. 9. Simulation o 32-bit Vedic multiplier



From Port	To Port	Max Delay	Max Process Corner	Min Delay	Min Process Corner
a[0]	c[60]	19.154	SLOW	2.601	FAST
a[0]	c[61]	19.263	SLOW	2.637	FAST
a[0]	c[62]	19.181	SLOW	2.615	FAST
a[0]	c[63]	19.260	SLOW	2.642	FAST

Fig 10 : 32-bit Array Multiplier Timing Report



From Port	To Port	Max Delay	Max Process Corner	Min Delay	Min Process Corner
b[2]	result[58]	15.274	SLOW	3.310	FAST
b[2]	result[59]	15.285	SLOW	3.309	FAST
b[2]	result[60]	15.394	SLOW	3.345	FAST
b[2]	result[61]	15.312	SLOW	3.323	FAST

Fig 11 : 32-bit Vedic Multiplier Timing Report

By comparing the timing report and LUT unit reports we can see a major decrease in the critical path delay of 32-Bit Vedic multiplier with respect to ordinary 32-Bit multiplier (Fig 10 and 11) and by taking the reference of the lookup table(LUT) (fig 6 and 7) we found that the area has been reduced.

A lookup table (LUT) is a hardware device that stores a set of predefined outputs based on a set of input values. Slice LUTs are a type of LUT that are commonly used in field-programmable gate arrays (FPGAs) to implement digital logic functions.

32-bit multiplier LUT and slice LUT :

- The 32-bit multiplier LUT is a dedicated LUT that performs multiplication of two 32-bit operands using a look-up table. It requires a single LUT to perform multiplication and is therefore more area-efficient compared to slice LUT 1545 .
- Slice LUT 1545 , on the other hand, is a more general-purpose LUT that can be used to implement a wide range of logic functions. It requires a combination of four slice LUTs to perform multiplication, making it less area-efficient compared to the dedicated 32-bit multiplier LUT

32-bit Vedic multiplier LUT and slice LUT 47:

- The 32-bit Vedic multiplier LUT is a type of multiplier that is based on ancient Indian Vedic mathematics principles. It uses a combination of addition, subtraction, and bitwise operations to perform multiplication. It requires a single LUT to perform multiplication and is therefore more area-efficient compared to slice LUT 47.

- Slice LUT 47 is a small, low-capacity LUT that can be used to implement simple logic functions. It requires a combination of eight slice LUTs to perform multiplication, making it less area-efficient compared to the 32-bit Vedic multiplier LUT.

In summary, dedicated multipliers such as the 32-bit multiplier LUT and 32-bit Vedic multiplier LUT are generally more area-efficient for multiplication operations compared to slice LUTs such as slice LUT 1545 and slice LUT 47, which are more general-purpose LUTs that can be used for a wide range of logic functions. However, the choice of which multiplier to use depends on the specific requirements of the application, including area, speed, power consumption, and accuracy.

There is a major reduction in the LUT cells. From this we can conclude that bringing in vedic mathematics can increase the speed and reduce the size of a processor.

TABLE. I PARAMETERS COMPARISON

Parameter	Array Multiplier	Vedic Multiplier
No of LUTs	1545	47
On Chip Power	81.01 W	67.11 W
Critical Path Delay	19s	15s

V CONCLUSIONS

From the Table 1 shown above we can conclude that vedic mathematics has brought a major improvement in the performance characteristics of a 32-bit multiplier. From the comparisons we can see 97% reduction in LUT units, 17.15% reduction of on chip power 21.05% improvement in critical path delay.Hence when a 32-bit vedic multiplier is integrated to the ALU unit of 32-bit RISC processor there will be a major positive impact on its size, power and speed.

REFERENCES

[1] Yadav, A. and Bendre, V., 2021, March. Design and Verification of 16-bit RISC Processor Using Vedic Mathematics. In 2021 International Conference on Emerging Smart Computing and Informatics (ESCI) (pp. 759-764). IEEE.

[2] Kamaraj, A., Parimalah, A.D. and Priyadharshini, V., Realisation of Vedic Sutras for Multiplication in Verilog. SSRG International Journal of VLSI and Signal Processing (SSRG-IJVSP), 4(1), pp.25-29.

[3] Lai, J.Y., Chen, C.A., Chen, S.L. and Su, C.Y., 2021, November. Implement 32-bit RISC-V Architecture Processor using Verilog HDL. In 2021 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS) (pp. 1-2). IEEE.

[4] Kumar, G.G. and Charishma, V., 2012. Design of high speed vedic multiplier using vedic mathematics techniques. International Journal of Scientific and Research Publications, 2(3), p.1.

[5] Gadda, Nilam, and U. Eranna., "64-bit ALU Design using Vedic Mathematics." 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE). IEEE, 2020.

[6] Galani Tina, G., Riya Saini, and R. D. Daruwala., "Design and Implementation of 32-bit RISC Processor using Xilinx." International Journal of Emerging Trends in Electrical and Electronics 5.1 (2013).

[7] A. Bisoyi, M. Baral and M. K. Senapati, "Comparison of a 32-bit Vedic multiplier with a conventional binary multiplier," International Conference on Advanced Communications, Control and Computing Technologies, Ramanathapuram, 2014 .

[8] Lad, Shraddha, and Varsha S. Bendre. "Design and comparison of multiplier using vedic sutras." 2019 5th International Conference On Computing, Communication, Control And Automation (ICCUBEA). IEEE, 2019.

[9] Bansal, Yogita, Charu Madhu, and Pardeep Kaur. "High speed vedic multiplier designs-A review." *2014 Recent Advances in Engineering and Computational Sciences (RAECS)*. IEEE, 2014.

[10] Chowdary, K. Keshav Sai, et al. "Design of Efficient 16-bit Vedic Multiplier." *2021 3rd International Conference on Signal Processing and Communication (ICPSC)*. IEEE, 2021.