iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** II **Month of publication:** February 2024

DOI: https://doi.org/10.22214/ijraset.2024.58332

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue II Feb 2024- Available at www.ijraset.com

Implementation of 64-Bit RISC Processor Using Vedic Multiplier

Thulasi Venkata Sai[1], Sunku Vinay[2], Reddyvari Tharun[3], Vidyalavaru Subhash Reddy[4], Sadhu Vinay Kumar[5], G.Padma Priya[6]
Department of ECE, Sri Venkateswara College of Engineering, Tirupati, Andhra Pradesh, India

Abstract: ***In this project, 64 bit RISC processor designed with Vedic multiplier design. Reduced Instruction Set Computer (RISC) is a design which presents better performances, higher speed of operation and favors the smaller and simpler set of instructions. In addition to multiplier which is implemented using vedic mathematics we are also proposing an adder which is mux- based full adders for building higher bit adders in an area and speed efficient which is implemented in addition as well as for compression in vedic mathematic to obtain the output. A 64 bit RISC processor designed in this paper is capable of executing more number of instructions with simple design, using the Verilog Hardware Description Language (HDL) and the design is simulated in the Xilinx Vivado 2018.3. The main achievement in this work is that the multiplier unit in Arithmetic and Logic Unit (ALU) and Multiplier and Accumulator (MAC) is implemented using Vedic Sutras. The main principle used in Vedic mathematics is to reduce the typical calculation of conventional mathematics to very simple one and hence reduce the overall computational complexity. Vedic Multiplier design is based on "Urdhva Triyakbhyam" which is among the 16 Vedic Sutras and MUX-Based Full Adders The proposed RISC processor is very simple and capable of executing 14 instructions. The achievement in this work is that savings in power in case of MAC and ALU is achieved compared to conventional ALU and MAC respectively. Also the delay is reduced in MAC and ALU in comparison with conventional ALU and MAC correspondingly. These Vedic MAC and ALU are then integrated with other blocks in processor and 64-bit Vedic processor is developed. This reduces the delay and saves area compared to conventional processor. Hence the improvement in speed of operation, and less area utilization are the key features of designed RISC processor.***

Keywords: Reduced Instruction Set Computer; VonNeumann architecture; Verilog HDl, Vedic Mathematics, Urdhva-Tiryagbhyam Sutra

I. INRODUCTION

Vedic mathematics was more briefly researched by Bharati Krishna, who was a research scholar, specified that overall 16 sutras and 13 sub-sutras describes Vedic mathematics which in turn increases the simplicity of solving mathematical equation for computation.
. For these task multiplications is a fundamental hardware part. Hence, the presentation of the multiplier is a key component in deciding the exhibition of the whole framework. This is on the grounds that, the multiplier is the slowest and most tedious component in the framework. Along these lines, the enhancement of the multiplier speed and area is a noteworthy test for the framework architects. This test can be effectively overwhelmed by the utilization of old Vedic mathematics techniques.

Binary multipliers are used in digital circuit plan which makes them quick, dependable and effective to execute any task. Contingent upon the course of action of the segments, there are various kinds of multipliers accessible. Vedic multiplier is based on Vedic mathematics containing 16 different sutras which are divided based upon its execution of different arithmetic operations. The intensity of Vedic mathematics isn't just kept to its effortlessness, consistency, yet in addition it is sensible. Its high level of importance is ascribed to the previously mentioned certainties. The logic behind Urdhva Tiryakbhyam sutra is especially like the conventional array multiplier.

Nikhilam Sutra is all from nine and last from ten. To perform the multiplication operation, it finds out the compliment of the large number from its nearest base. Ekanyunena Purvena implies one not exactly past one or one short of what one preceding. This sutra is implicable for multiplication of numbers where multiplier is 9 or cluster of 9. Squaring task is helpful in digital signal processing (DSP) applications in which multiplication activity requires repetitive addition operations. Ekadhikena Purvena Sutra is summed up for finding square of a number.

A multiplier usually has three steps. The first step is the partial product generation (PPG) process. For example, AND gates can be used to generate a partial product matrix (PPM) for an unsigned multiplication.

The second step is the partial product reduction (PPR) process. By using the Dadda tree approach or the Wallace tree approach, the PPM can be reduced to become two rows. The third step is the final addition. An adder (called the final adder) is used to perform the summation of the final two rows. For an *N*-bit multiplier, a *(2N-1)*-bit adder is required for the final addition.

Various adder architectures have been proposed for the trade-offs among delay, area, and power. Furthermore, various MAC unit models can be developed by replacing the multiplier as well as the accumulator (adder) with various architectures. Comparisons on delay, area and power among different MAC unit models are reported.

The most vital task in mathematical operations is the multiplication which can be executed through many ways such as MAC, CIAF (Computations Intensive Arithmetic Functions) and are presently applied in DSP applications. As the multiplication is the most important task, speed of any system is decided upon the type of multiplier used for the operation. This work presents different multiplier architectures which are fast and low power based on Vedic Mathematics. Depending upon the arrangement of the components, there are different types of multipliers available.

Jain and A. Jain compared two different architectures such as Array and Vedic to know the best architecture for multiplication in turns of power and delay characteristics. Studying and analyzing power and delay characteristics, author came to know that “Urdhva Tiryakbhyam” multiplier is the best multiplier compared to array and Nikhilam multiplier when compared to delay and power calculations.

II. EXISTING SYSTEM

The existing system presents 32-bit RISC Processor using Vedic Multiplier.The function of the processor is to execute each and every instruction set efficiently as per the machine language. In Processor, ALU inputs consist of instruction (machine word) which is operation code (opcode) and some operands. So the opcode tells the ALU which and what operation is to be performed then these operands are used in the operation.

There is a small set of data holding place that is known as Register bank. The ALU stores the result of operation in accumulator which later on is 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. A processor has a set of instructions which is nothing but a command to perform a task in a computer. The control unit holds the instruction to be executed. In CPU, the registers such as address register, data register and an instruction register is 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 the 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. This is done with the help of control unit which generates the timing signals that controls the various processing elements which involves in execution of the instruction.

The MAR is also called as 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. The MAR is completely loaded with Binary words which point the location of the word in RAM. This location stores the instruction in it.

Ankita Yadav et al. [1] in this paper presents a 16-bit RISC processor designed using Verilog Hardware Description Language (HDL) and simulated in the Xilinx ISE 14.7 design suite. The main achievement is the implementation of the multiplier unit in Arithmetic and Logic Unit (ALU) and Multiplier and Accumulator (MAC) using Vedic Sutras. The proposed RISC processor is simple and capable of executing 14 instructions. The design achieves 44% savings in power and 12% reduction in delay compared to conventional ALU and MAC. When integrated with other blocks, the 16-bit Vedic processor reduces delay by 34% and saves around 88% power compared to a conventional processor. The key features of the designed RISC processor include improved speed, reduced power utilization, and less area utilization. The Vedic processor is implemented using Xilinx ISE 14.7 Design Suite in Verilog and verified using QuestaSim of Mentor Graphics.

Aditi Chhabra et al. [2] in this paper presents an alternative approach to designing the MAC unit of a neural network using an 8x8 Vedic Multiplier. The network consists of four 4x4 Vedic Multipliers and a 16-bit Rippler Carry Adder instead of a conventional array multiplier. The paper compares different configurations of Vedic multipliers and traditional array multipliers, aiming to obtain an alternative approach for the realization of the MAC unit of the neural network. The proposed network achieves an accuracy of 88%, which can be increased by increasing the precision of the multiplier. The paper also highlights the emerging Reversible Logic Technology in Vedic Multipliers. The proposed network prioritizes power efficiency and maintains the precision of the multiplier, resulting in an accuracy of 88%.

Aslesa Singh, et al. [3] presented in this paper about a 32-bit Datapath with RISC-V instruction set architecture based on RV32I CPU instruction set. The processor is optimized using a six staged folded pipeline core and tested on two Virtex family FPGA boards. The Ultrascale board offers better power-performance tradeoff at higher costs, with a higher total chip power but lower dynamic power. The design is parameterized and user-defined, with over a dozen options for power efficiency or performance. The RISC-V environment allows for extensive tweaking, making it suitable for error-free placement on devices. The design can be widely used in upcoming processor designs with a RISC-V ISA due to its unique design pipelining component and highly parametrized core.

Tariquzzaman, et al. [4] presented in this paper about a 64-bit RISC processor with a Vedic multiplier feature (using the Nikhlam Sutra), implemented using VHDL. The processor is designed to reduce execution time and address large memory up to 16 Exabytes. The Vedic multiplier architecture is 135.05% faster than conventional multipliers, reducing time in multiplication operations. The design is verified on the Xilinx ISE 10.1i simulator and programmed using VHDL. Future work will focus on increasing instructions and creating a pipelined design with less clock cycles per instruction.

Shradda Lad et al. [5] presented in this paper about a highly efficient Vedic multiplier unit using various Sutras of Vedic Mathematics. The design, synthesis, and simulation of a 16-bit Vedic multiplier unit were performed using Vivado 17.1 and Verilog. Performance parameters like delay, power, and area were analyzed. The Ekadhikena Purvena sutra achieved a 70.26% reduction in area and a 90.41% increase in speed compared to Urdhva Tiryakbhyam. The Nikhilam sutra provided the optimum area and delay, with the most effective in terms of power. These performance values can be used for selecting multipliers in image and signal processing applications and digital filters.

Rashmi Samanth, et al. [6] presented in this paper about the design and implementation of a 32-bit Functional unit for RISC-based processors, including Arithmetic and Logic Unit (ALU) modules. The design uses multiplexers for operations and is compared to conventional Microprocessor without interlocked Pipeline Stages (MIPS). The design reduces power dissipation by 30.449%, area by 6%, and delay by 34.49%. The design uses behavioral and structural modeling, and simulation is supported using Xilinx ISE and Cadence RTL Compiler.

III. PROPOSED SYSTEM

The function of the processor is to execute each and every instruction set efficiently as per the machine language. ALU is the combinational circuit which means Arithmetic and Logical Unit. This unit is designed to perform various numbers using various instruction sets. In Processor, ALU inputs consist of instruction (machine word) which is operation code (opcode) and some operands. So the opcode tells the ALU which and what operation is to be performed then these operands are used in the operation.

There is a small set of data holding place that is known as Register bank. The ALU stores the result of operation in accumulator which later on is 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. A processor has a set of instructions which is nothing but a command to perform a task in a computer. The control unit holds the instruction to be executed. In CPU, the registers such as address register, data register and an instruction register is 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 the 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. This is done with the help of control unit which generates the timing signals that controls the various processing elements which involves in execution of the instruction.

The MAR is also called as 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. The MAR is completely loaded with Binary words which point the location of the word in RAM. This location stores the instruction in it.

“Urdhva Tiryakbhyam” (Vertical and Crosswise) sutra (Algorithm). It is a general multiplication formula equally applicable to all cases of multiplication. The algorithm generates all partial product and sum in one step. The algorithm is generalized for nxn bit number .This Multiplier has advantage that when we increase the number of bits ,gate delay and area increases very slowly compared to other multipliers. Due to its regular structure, Multiplier processing power increase by increasing the input and output data bus widths. The digits on the two ends of the line are multiplied and the result is added with the previous carry. When there are more lines in one step, all the results are added to the previous carry. The least significant digit of the number thus obtained acts as one of the result digits and the rest act as the carry for the next step. Initially the carry is taken to be as zero.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue II Feb 2024- Available at www.ijraset.com

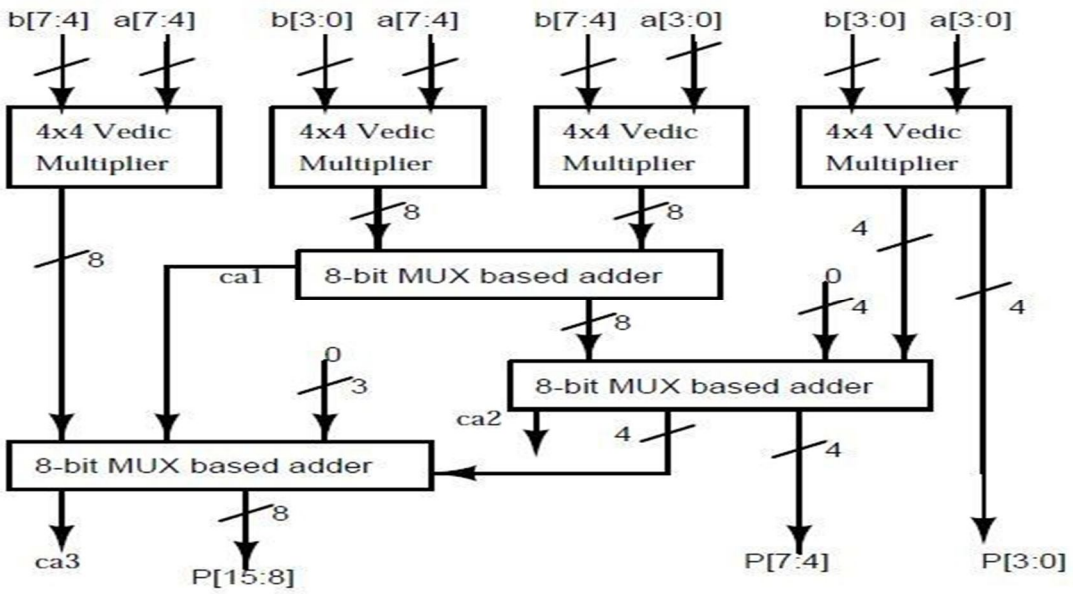


Fig. 1: Block diagram of proposed method

A. Modified full Adder

Where this modified full adder consists of two 4:1 multiplexer. Using this modified full adder gets decreases with low power consumption.

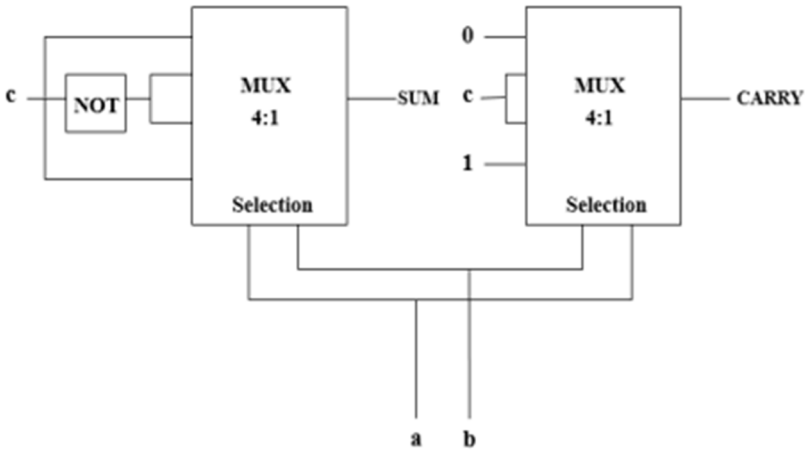


Fig: Proposed full adder multiplexers

Multiplexer is a combinational circuit that has maximum of 2^n data inputs, 'n' selection lines and single output line. One of these data inputs will be connected to the output based on the values of selection lines. Since there are 'n' selection lines, there will be 2^n possible combinations of zeros and ones. So, each combination will select only one data input. Multiplexer is also called as Mux. 4x1 Multiplexer has four data inputs I_3, I_2, I_1 & I_0, two selection lines s_1 & s_0 and one output Y. The block diagram of 4x1 Multiplexer is shown in the following figure. In this modified full adder consists of two 4:1 multiplexers here a, b are two selection lines for both the multiplexers C is the only input to the multiplexer that gives sum as output, 0,c,1 are the inputs for the multiplexer that gives carry as the output. After studying the conventional full adder design, a multiplexer based design is modeled by identifying its operation using truth table and the input and output values when logic 0 and logic 1 as input at the consequent input line. Using this multiplexer based adder designs it can be observed that there is a maximum reduction in the gate count of the logic design, thus making it an efficient implementation.

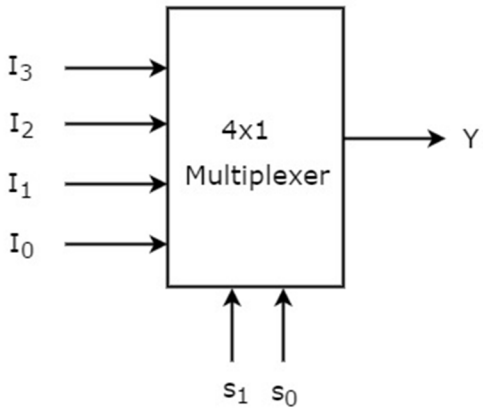
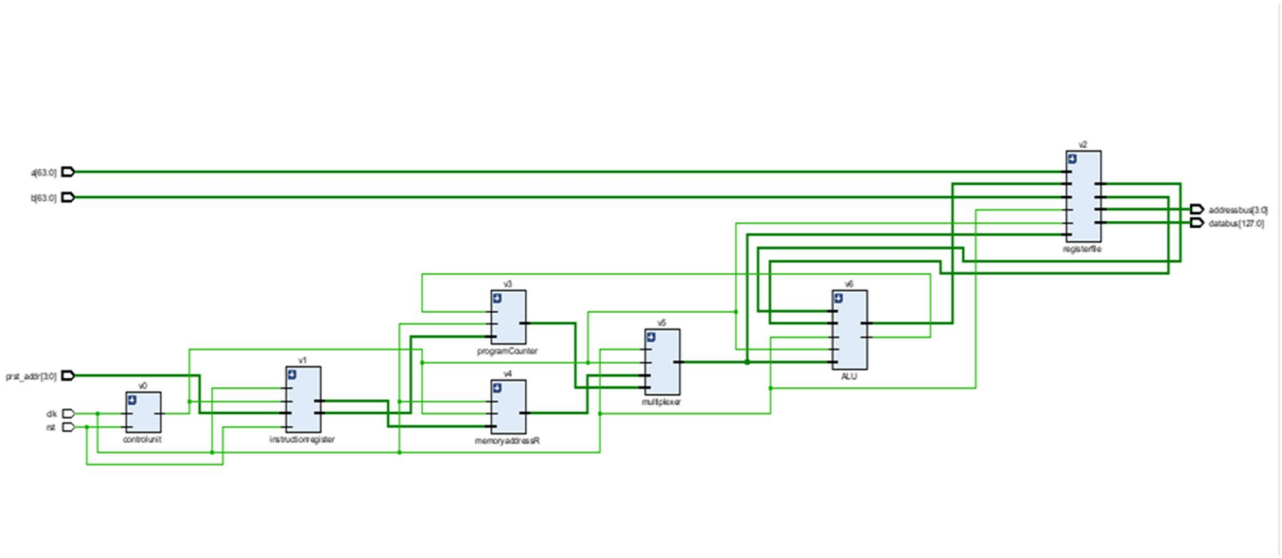


Fig: 4:1 multiplexer

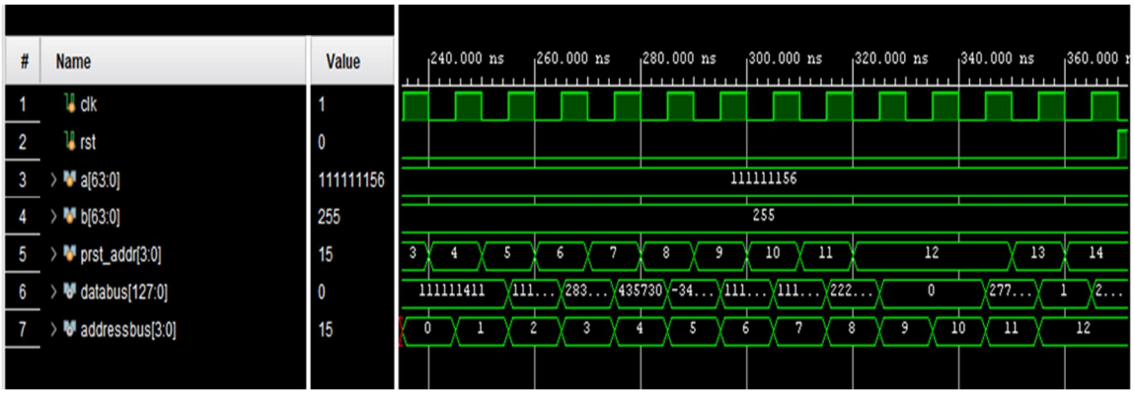
IV. RESULTS

A. *RTL Schematic*



B. *Technology Schematic*

C. Simulation



D. Area

Name	Slice LUTs (134600)	Slice Registers (269200)	Slice (33650)	LUT as Logic (134600)	Bonded IOB (400)	BUFGCTRL (32)
processor_64bit_extension	12815	747	3639	12815	266	1
v0 (controlunit)	1	1	1	1	0	0
v1 (instructionregister)	2	8	1	2	0	0
v2 (registerfile)	11908	403	3525	11908	0	0
v3 (programCounter)	0	4	1	0	0	0
v4 (memoryaddressR)	0	4	1	0	0	0
v5 (multiplexer)	339	4	122	339	0	0

E. Delay

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay
Path 1	∞	1111	66	180	v2/R1_reg[34]/C	v6/aluout_reg[0]/D	328.989	144.069	184.922
Path 2	∞	1094	66	180	v2/R1_reg[34]/C	v6/aluout_reg[1]/D	325.928	141.726	184.203
Path 3	∞	1076	64	180	v2/R1_reg[34]/C	v6/aluout_reg[2]/D	319.947	139.385	180.564
Path 4	∞	1058	63	180	v2/R1_reg[34]/C	v6/aluout_reg[3]/D	315.320	137.051	178.270
Path 5	∞	1040	62	180	v2/R1_reg[34]/C	v6/aluout_reg[4]/D	310.425	134.717	175.710
Path 6	∞	1023	61	180	v2/R1_reg[34]/C	v6/aluout_reg[5]/D	306.118	132.465	173.655
Path 7	∞	1005	60	180	v2/R1_reg[34]/C	v6/aluout_reg[6]/D	301.441	130.256	171.186

F. Evaluation Table

	Area (LUT's)	**Delay (ns)**
Processor64	13901	331.101
Extension64	12815	328.989

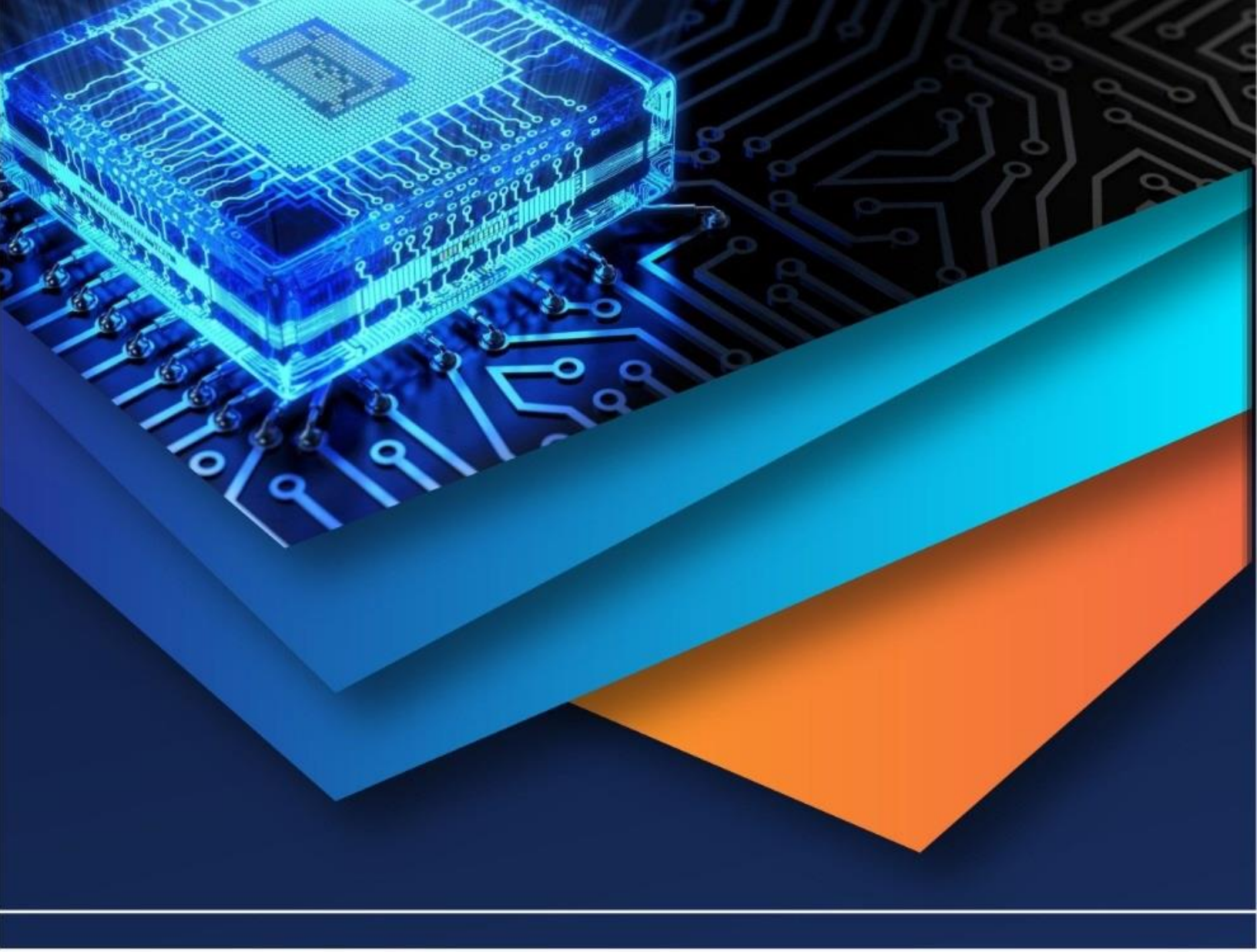
V. CONCLUSION

In this paper, 64- bit RISC processor based on Vedic sutra –Urdhva Triyakbhyam in ALU has been implemented. Vedic Multipliers are used for multiplication to improve the speed and reduce the area and power budget of the multipliers and in addition to it adders are getting replaced with mux-based adders which will improve the area efficiency slightly and hardware complexity will be optimized compared conventional adders. The utilization of Vedic Multiplier allows for a high speed, low power and reduction in the overall gate count, contributing to a more streamlined and resource-efficient architecture. For the 64 bit we are unable to optimize with divide and conquer based adder implementation so we can proceed with MFA implementation along with programmable clock for extending it to applications.

In this work, Vedic processor is designed using Vedic MAC and Vedic ALU along with other conventional blocks in processor. The comparison of simulation result of Vedic ALU and MAC design is done with the existing ALU and MAC results by using mux based adders not only for multipliers addition operation in ALU can also be improvised. The 64-bit Vedic processor reduces the Area, delay compared to conventional processor. Hence the improvement in speed of operation, and less area utilization are the key features of designed RISC processor.

REFERENCES

[1] Ankita Yadav and Varsha Bendre (2021). Design and Verification of 16-bit RISC Processor Using Vedic Mathematics. 2021 International Conference on Emerging Smart Computing and Informatics (ESCI) AISSMS Institute of Information Technology, Pune, India. Mar 5-7, 2021.
[2] Aditi Chhabra and Jasdeep Dhanoa (2020). A Design Approach for Mac Unit Using Vedic Multiplier. 5th IEEE International Conference on Recent Advances and Innovations in Engineering- ICRAIE 2020 (IEEE Record#51050).
[3] Aslesa Singh, Neil Franklin, Nidhi Gaur, Paurush Bhulania. Design and Implementation of a 32-bit ISA RISC-V Processor Core using Virtex-7 and Virtex-UltraScale.2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA). Galgotias University, Greater Noida, UP, India. Oct 30-31, 2020.
[4] Tariquzzaman, Syed Rizwan Ali, Nahid Kausar. FPGA implementation of 64 bit RISC processor with Vedic multiplier using VHDL. IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE) e-ISSN: 2278-1676, p-ISSN: 2320-3331 PP 12-16.
[5] S. Lad and V. S. Bendre, "Design and Comparison of Multiplier using VedicSutras," 2019 5th International Conference On Computing, Communication, Control And Automation (ICCUBEA), Pune, India, 2019, pp.1-5
[6] Rashmi Samanth, Ashwini Amin, Subramanya G. Nayak. Design and Implementation of 32-bit Functional Unit for RISC architecture applications 2020 5th International Conference on Devices, Circuits and Systems (ICDCS) 978-1-7281-6368-0/20/$31.00 ©2020 IEEE 10.1109/ICDCS48716.2020.243545.
[7] Jagjeet Sharma, Candy Goyal. High Speed 16- Bit Vedic Multiplier Using Modified Carry Select Adder. Research Cell: An International Journal of Engineering Sciences, January 2016, Vol. 17 ISSN: 2229-6913 (Print), ISSN: 2320-0332 (Online) -, Web Presence: http://www.ijoes.vidyapublications.com.
[8] Suggula Naga Sai Vishnu, Abhishek Gandluru, Ramesh S R. 32-Bit RISC Processor Using Vedic Multiplier 2022 3rd International Conference for Emerging Technology (INCET) Belgaum, India. May 27-29, 2022.
[9] Amrutha Vani S , Dr T Somassoundaram. DESIGN AND VERIFICATION OF 32-BIT RISC PROCESSORUSING VEDIC MULTIPLIER1.PG-Scholar, Department of ECE, Sri Venkateswara College of Engineering & Technology (SVCET) R.V.S. Nagar, Chittoor, India.
[10] Balpande Vishwas V, Abhishek B. Pande, Meeta J. Walke, Bhavna D. Choudhari and Kiran R. Bagade. “Design and Implementation of 16 Bit Processor on FPGA.” (2015).
[11] Seung Pyo Jung, Jingzhe Xu, Donghoon Lee, Ju Sung Park, Kang-joo Kim and Koon-shik Cho, "Design & verification of 16 bit RISC processor," 2008 International SoC Design Conference, Busan, 2008, pp. III-13-III-14, doi: 10.1109/SOCDC.2008.4815726.
[12] S. P. Ritpurkar, M. N. Thakare and G. D. Korde,” Design and simulation of 32-Bit RISC architecture based on MIPS using VHDL,” 2015 International Conference on Advanced Computing and Communication Systems, 2015, pp. 1-6, doi: 10.1109/ICACCS.2015.7324067.
[13] Singh, A. & Gupta, R. (2015). Design Challenges in Integrating Vedic Multipliers into Modern RISC Processors. IEEE Journal of Solid-State Circuits, 50(6), 1500-1510.
[14] Jain, A. & Khanna, R. (2009). Efficient Adder Design for RISC Processors using Vedic Mathematics. IEEE Transactions on VLSI Systems, 17(5), 645-655.
[15] Mehta, K. & Roy, D. (2018). A Comparative Study: Traditional vs. Vedic Multiplier in RISC Architecture. ACM Transactions on Embedded Systems, 7(4), 25-35.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY