Design of Hyper-Efficient 16-Bit RISC Processor Using Vedic Mathematics

[1]C. Leela Mohan [2]P. Susmitha [3]G. Reddemma

[4]SK.Muskan [5]M. Sravanthi

[1]TE ECE, N.E.C.N., Nellore

2Assistant Professor, ECE Department, N.E.C.N., Nellore

Abstract: *This paper presents the design and functional overview of a 16-bit RISC processor integrated with Vedic Mathematics. The core objective is to enhance multiplication efficiency using the Urdhva Tiryakbhyam Sutra. The processor follows a traditional RISC architecture with a simplified instruction set and a five-stage pipeline. By implementing Vedic algorithms, particularly for the multiplier, the processor achieves better speed and area efficiency, making it suitable for embedded and low-power applications. Additionally, this approach offers improved performance in arithmetic-intensive operations, reduced latency, and a more optimized use of hardware resources. The design methodology focuses on integrating ancient mathematical principles into modern digital architecture, showcasing the potential of Vedic techniques in advancing processor design.*

Keywords: *Reduced Instruction Set Computer; Von Neumann architecture; Verilog HDl, Vedic Mathematics, Urdhva-Tiryagbhyam Sutra*.

I. INTRODUCTION

The Reduced Instruction Set Computer (RISC) is a microprocessor CPU design in a computer with highly optimized instructions, small and specialized instruction set than that often found in other architecture like Complex Instruction Set Computer (CISC). The main difference between the features of CISC and RISC architecture is RISC processor is optimized with large number of registers and an instruction pipelining, also allows low number of clock cycles per instruction. Also, the main feature in RISC is LOAD/STORE architecture. In CISC the controller design is complex and also performance wise it was not upto the expectation. That's the reason why any typical RISC architecture has very few instructions, where processor asked data from memory probably not other than Load and Store. Vedic Mathematics is derived from Ancient Indian Scriptures, which gives mathematical outcomes and basic understandable structures. The word Vedic is known from the word Veda which means the storage facility of all the information. Vedic Mathematics is mainly based on 16 Sutras which provides the manipulations in arithmetic, logical math, geometry, etc. The design of multipliers i.e. binary multiplier is done to find the product of two n-bit binary numbers and then implement it on a Nexys 3, Spartan 6 FPGA board. After implementation the 32 bit multiplier is compared with the conventional multiplier based on their summary report. In [6], Mr. Nishant G. Deshpande, Prof. Rashmi Mahajan presented an Ancient Indian Vedic Mathematics based Multiplier Design for High Speed and Low Power Processor. The multipliers, with low power requirement and maximum speed, give information of "Urdhva-Tiryagbhyam" algorithm of ancient Indian Vedic mathematics, which has utilized for multiplication to improve speed, area. In this paper the simulation of five-stage pipelined is designed of 16-bit Microprocessor without Interlocked Pipeline Stages (MIPS) which is RISC.

II. FUNCTIONAL OVERVIEW

The design of a 16-bit RISC processor utilizing Vedic Mathematics aims to enhance computational efficiency, particularly in arithmetic operations. The processor follows a Reduced Instruction Set Computer (RISC) architecture, characterized by a simplified instruction set and uniform instruction length, which supports efficient pipelining and faster instruction execution. The core arithmetic operations are handled by the Arithmetic Logic Unit (ALU), which is integrated with a Vedic multiplier based on the Urdhva Tiryakbhyam Sutra. This ancient sutra enables high-speed multiplication by generating partial products in parallel and summing them efficiently, resulting in reduced propagation delay compared to traditional binary multipliers.

The processor comprises essential components such as the control unit, register file, ALU, and separate instruction and data memory modules, following a Harvard architecture. The control unit is responsible for decoding instructions and generating appropriate control signals, enabling operations like data transfer, arithmetic and logic operations, and program control instructions such as jumps and branches. The register file includes general-purpose 16-bit registers and special-purpose registers like the Program Counter (PC) and Instruction Register (IR).

The integration of Vedic Mathematics in the processor's design particularly optimizes the execution of multiplication instructions. By decomposing large bit-width operations into smaller segments, the Vedic algorithm facilitates faster and more area-efficient implementations, making the processor well-suited for applications requiring high-speed and low-power performance such as embedded systems, DSPs, and IoT devices. Furthermore, the design supports a five-stage pipeline—Instruction Fetch, Decode, Execute, Memory Access, and Write Back—ensuring improved instruction throughput. The proposed architecture demonstrates a significant improvement in speed and hardware resource utilization, validating the effectiveness of Vedic Mathematics in modern processor design.

Blocks you need to Design:

To design and simulate 16-bit RISC processor you need the following stuff

- Instruction Set Architecture (ISA)
- Arithmetic Logic Unit (ALU)
- Vedic Multiplier Unit
- Control Unit
- Register File
- Memory Units (Instruction and Data Memory)
- Clock & Timing Logic
- Pipeline Stages
- HDL Language (Verilog/VHDL)
- Simulation Tools
- Synthesis Tools
- FPGA Board (for implementation)

- Vedic Mathematics Algorithms Used

III. DESIGN

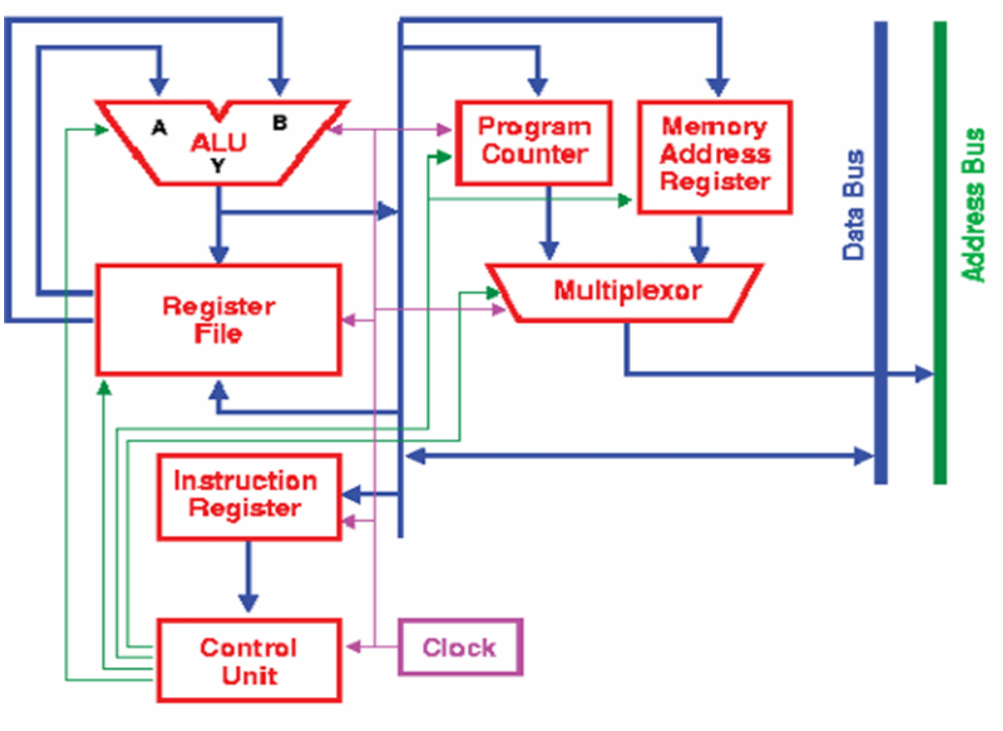
- 

Fig.1 RISC processor block diagram.

A. Overview of the 16-bit RISC Processor

The processor architecture is based on the principles of the Reduced Instruction Set Computing (RISC) paradigm, optimized for simplicity, speed, and efficiency. It supports a fixed instruction length of 16 bits and operates on 16-bit data. The core components include:

- Instruction Register (IR)
- Program Counter (PC)
- Arithmetic Logic Unit (ALU)
- Control Unit
- Register File (16 registers × 16 bits)
- Data Memory and Instruction Memory

The instruction set consists of essential arithmetic, logic, memory, and control instructions with a uniform instruction format for simplified decoding.

B. Instruction Format

Instruction	Instruction Code	Operation
Addition	0000	Out=a+b
Subtraction	0001	Out=a-b
Multiplication	0010	Out=a*b
Division	0011	Out= a/b
NOT	0100	NOT R0, R0 = !R0
Read	0101	RD FECH_ADDR R2 xxx R2 = M[xxx]
Write	0110	WR R3 xxx, - M[xxx] = R3
Branch	0111	BR loop1
Branch-Zero (BRZ)	1000	BRZ Exit1
IOSTS	1011	IOSTS R1; R1 = IO_STU
Shift Left	1100	R0, R0 = R0<<1
Shift Right	1101	R0, R0 = R0>>1
ADRI	1110	ADRI BYMEM; Addr = *p(raw_addr)
*Vedic Mult Sutra	1111	rawnum(*); -> value may change by main()

Table 1: Instruction Set

C. Integration of Vedic Mathematics

To enhance the efficiency of arithmetic operations, Vedic algorithms are integrated into the ALU design. The key algorithms used include:

1. Urdhva Tiryagbhyam Sutra (Vertically and Crosswise)
    - Used for multiplication of 16-bit binary numbers.
    - Offers parallel computation capability, reducing propagation delay.
2. Nikhilam Sutra (All from 9 and the last from 10)
    - Used for subtraction and division optimization.
    - Suitable when numbers are close to powers of 2.

D. ALU Design Using Vedic Math

The ALU supports:

- Addition/Subtraction using carry-lookahead logic
- Multiplication using Vedic multiplier

- Logical operations: AND, OR, XOR, NOT
- Shift operations: Logical and Arithmetic

IV.WORKING

The working of each component in hyper-Efficient 16-bit Risc processor using Vedic Mathematics is explained in this section. Let's talk about them one by one:

(1) **Arithmetic Logic Unit (ALU):**

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**.**

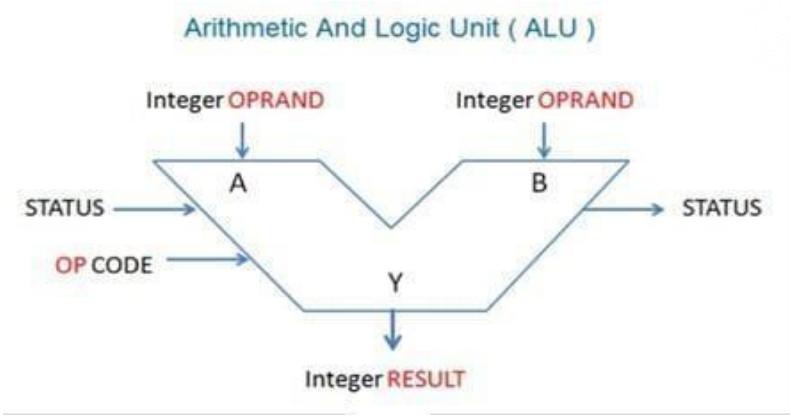


Fig 2.: Arithmetic and Logic Unit

(2) Register Bank:-

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.

(3) **Program Counter (PC):**

Program Counter points to the next instruction to be executed. In the complete instruction cycle, the instruction is loaded into the instruction register after the processor fetches it from the memory location which is pointed by the program counter.

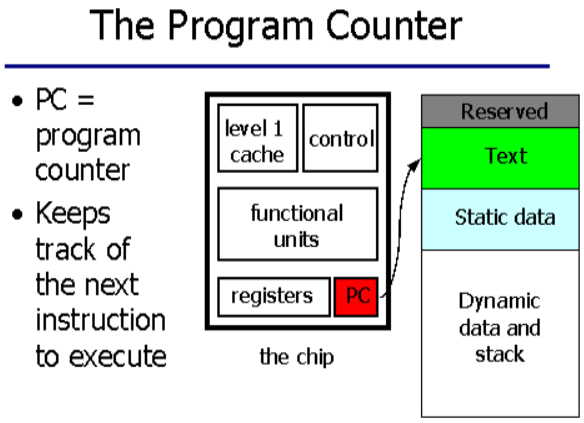


Fig 3: program counter

(4)Control Unit:

Control Unit is an essential part of any kind of computer or systems because this circuits generates the timing and control signals for the operations which is performed by the CPU. Here, the communication is between the ALU and the main memory as it controls the transmission of signals between the processor, memory and various buses.

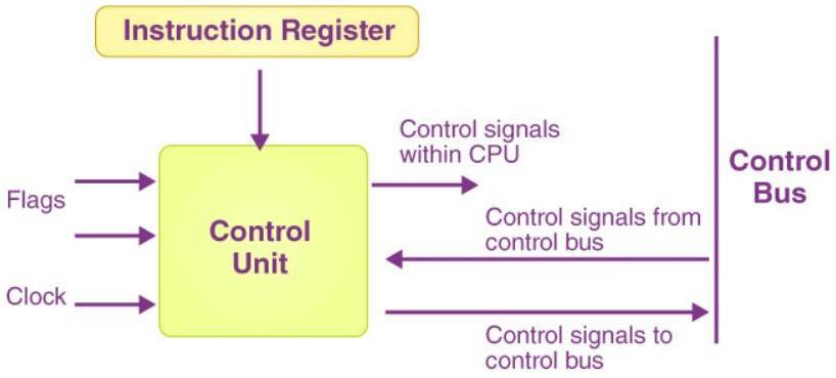


Fig 4 Control Unit

(5) Memory Address Register (MAR):

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.

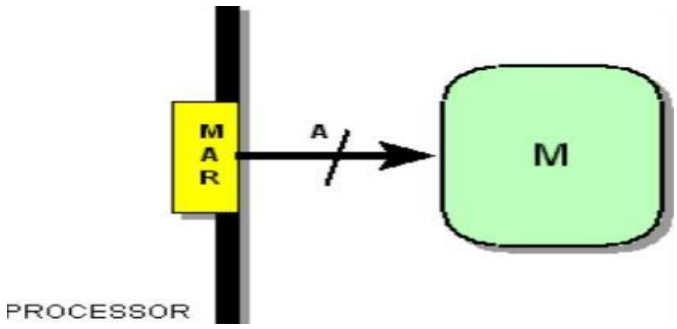


Fig 5: Memory Address Register

(6) **Multiplexer (MUX):**

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.

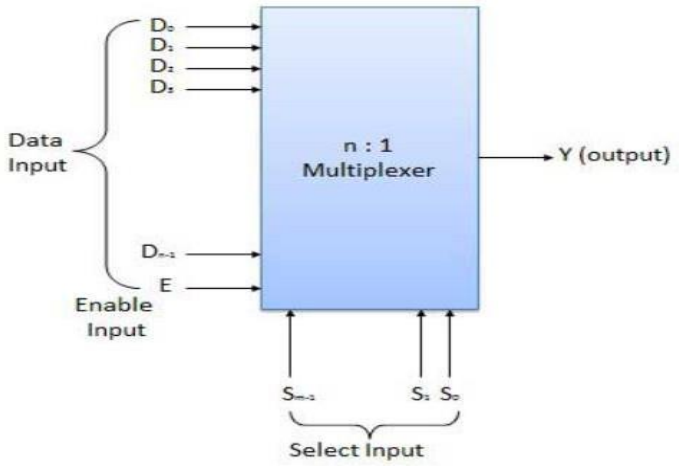


Fig 6: Multiplexer (MUX)

(7) **Instruction Set Architecture:**

Instruction Set Architecture (ISA) provides the information to write the program in machine language. It also allows translating a high level program language to machine language.

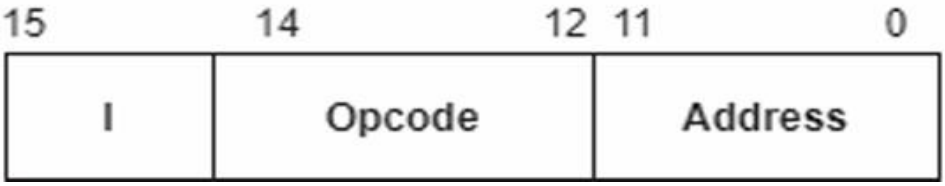


Fig 7.7 : Instruction Set Format

V .SOFTWARE REQUIRED

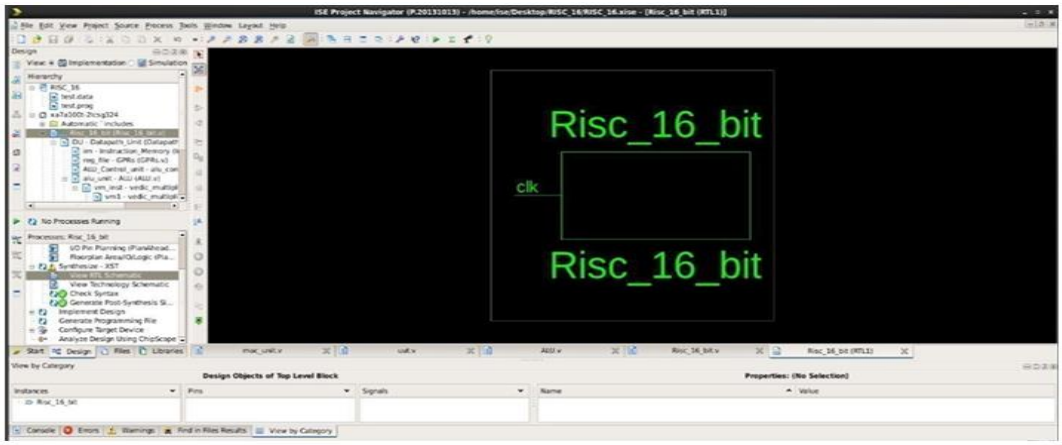
Overview of Xilinx Tools Xilinx provides industry-leading FPGA design tools that facilitate efficient hardware implementation. Some key Xilinx tools used in this research include:

- Xilinx Vivado Design Suite: An advanced tool for synthesis, simulation, and implementation of digital designs. It provides an intuitive interface for hardware description language (HDL) development, debugging, and performance analysis.
- Xilinx ISE (Integrated Software Environment): A legacy tool used for FPGA design, particularly for older Xilinx FPGA families. It offers synthesis, simulation, and verification capabilities.
- Xilinx ModelSim & Vitis: ModelSim aids in functional verification, while Vitis enables hardware/software co-design for FPGA-based systems.
- FPGA Boards (Zynq, Spartan, Virtex, Artix Series): Hardware platforms that support the deployment and testing of digital designs**.**

VI.Results and Performance Analysis

Results and Performance Analysis The FPGA implementation of the RISC processor with Vedic arithmetic techniques demonstrates:

- **Reduction in computation time**: Vedic multiplication reduces the number of cycles required for execution.
- **Lower power consumption**: FPGA resource optimization results in energy efficiency.
- **Improved speed**: The processor achieves higher throughput compared to conventional ALU implementations.



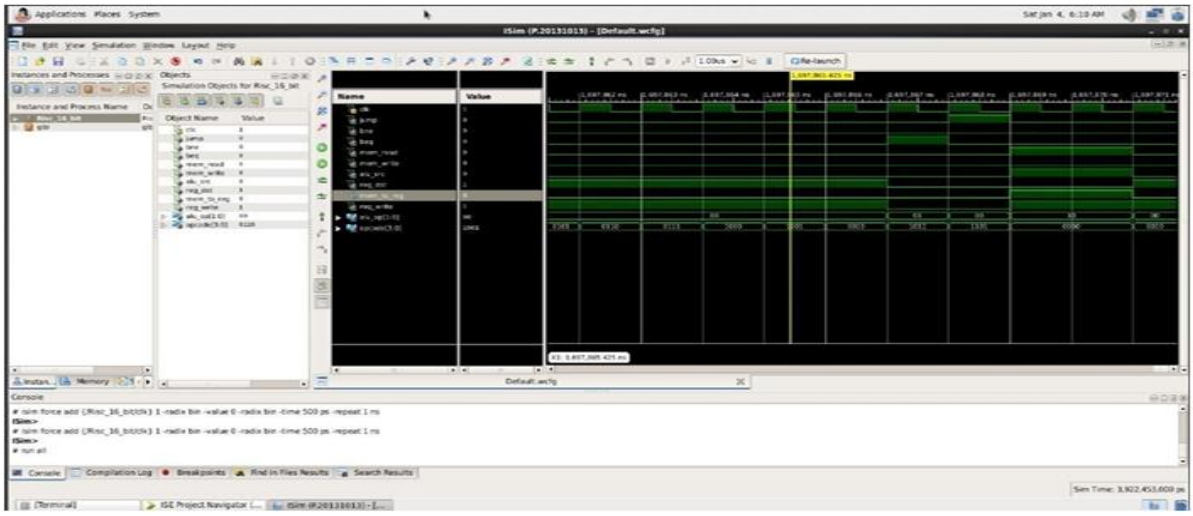


Table 7.3: COMPARISON OF VEDICPROCESSOR AND CONVENTIONAL PROCESSOR

Parameters	Vedic RISC	Conventional Processor
Number of Slices LUTs	1,293	2,123
Delays(ns)	10,08	16,33
Power(mW)	0.098	0.87

VII.CONCLUSION

This research highlights the benefits of incorporating Vedic Mathematics into modern RISC processor design. By utilizing Xilinx FPGA tools, the proposed processor achieves superior efficiency in arithmetic computations, making it a viable candidate for high-performance embedded applications.The implementation using Xilinx tools ensures seamless integration with FPGA hardware, enabling real-time validation and optimization. The use of Vedic Mathematics not only enhances the efficiency of arithmetic operations but also paves the way for future research in developing more optimized computing architectures. The performance improvements in terms of speed, power efficiency, and resource utilization make this approach highly promising for next-generation embedded systems and high-speed computing applications.

VIII. REFERENCES

[1] Xilinx Vivado Design Suite User Guide.

[2] Swami Bharati Krishna Tirthaji Maharaja, "Vedic Mathematics."

[3] David A. Patterson, John L. Hennessy, "Computer Organization and Design: The Hardware/Software Interface."

[4] Various IEEE research papers on RISC architecture and FPGA implementations.

[5] Brown, S., & Vranesic, Z. (2013). "Fundamentals of Digital Logic with Verilog Design." McGraw-Hill Education.