Advanced International Journal for Research (AIJFR)
E-ISSN: 3048-7641 ● Website: www.aijfr.com ● Email: editor@aijfr.com

Development of a Low Cost Lightweight Pipelined RV32IM Processor for Edge AI Processing

Mekala N.[1] , Sunil M. P.[2]

[1]Department of Elctronics and Communication Engineering,
Jain (Deemed-to-be) University
Bangalore, Karnataka, India
[2]Assistant Professor
[2]Department of Electronics and Communication Engineering,
Faculty of Engineering and Technology,
Jain (Deemed-to-be) University
Bangalore, Karnataka, India

Abstract
This paper discusses the development of a lightweight 32-bit pipelined processor implementing the RISC-V instruction set architecture. The processor uses a 5-stage pipeline scheme that provides hazard detection and data forwarding. To address the needs of the computational-intensive processing, the design uses a more advanced version of the standard RV32I instruction set with built-in multiplication–accumulation operation in the ALU complemented by lightweight SIMD computations and ReLU operation. The processor is modeled in Verilog HDL code and verified on the Xilinx Artix-7 FPGA board. Results of synthesis show that the design requires only 1.5% of LUTs and less than 1% of registers. The design is clocked by 100 MHz clock generator and all timing requirements are satisfied with 3.661 ns of slack. Energy consumption simulation shows on-chip power consumption of about 0.075 W. Simulation confirms proper operation of arithmetic and memory instructions, as well as MAC operation. The proposed architecture achieves a balance between performance, efficiency, and computational capability, making it suitable for embedded and edge-AI applications.

Keywords— RISC-V, RV32IM, Pipeline Processor, FPGA, Low Power Design.

1. INTRODUCTION

There is a significant need for effective, affordable, and adaptable CPU designs due to the quick development of embedded systems, Internet of Things (IoT) devices, and edge computing applications. Conventional proprietary instruction set architectures (ISAs) like x86 and ARM [1] [2] [3] sometimes impose restrictions on extension, flexibility, and licensing. Because of its modular architecture, open-source nature, and support for user-defined extensions, the RISC-V ISA has become a viable substitute in this regard [2] [3]. Because of these characteristics, it is especially well suited for domain-specific processor development, education, and research. [4]

In addition to general-purpose computation, modern embedded and edge applications, such digital signal processing and lightweight artificial intelligence (AI) inference, increasingly need fast execution of arithmetic-intensive workloads [10] [12]. Although they offer a straightforward and effective basis, conventional RISC-V processors built on the RV32I instruction set lack the specialized computing power required for such applications. However, improving speed through sophisticated methods like superscalar architectures or out-of-order execution results in substantial hardware overhead [7], higher power consumption, and design complexity [8], all of which are undesirable for situations with limited resources [11].

In order to overcome performance bottlenecks without compromising complexity, pipelining of processors is extensively used. The basic 5-stage pipelined architecture includes fetching instructions, decoding them, executing them, accessing memory, and writing back results, thereby increasing the efficiency of instruction execution through concurrent stage execution. But with the introduction of pipelines, issues like data hazards, control hazards, and pipeline stalls arise, which negatively impact performance, if not mitigated effectively [10].

Various approaches have been considered in recent years to enhance the performance of RISC-V processors using stall minimization, instruction set extensions, and low-power processor design. [17] Although these approaches have significantly contributed to achieving improvements in their respective domains, there is no existing solution that addresses both the pipelined and application-specific requirements together.

A 5-stage pipelined RV32I processor with hazard detection and forwarding capabilities is designed and synthesized by Verilog HDL and implemented on a Xilinx Artix-7 FPGA board in this study. To improve the functionality of the processor, domain-specific instructions have been added to the existing RV32I instructions. In particular, a multiply-accumulate (MAC) computation instruction was introduced, which is executed directly in the ALU block. Moreover, simple SIMD-type instructions have been implemented to provide an effective execution of arithmetic-heavy kernels.

The proposed processor achieves very low resource consumption, robust timing behavior, and low power dissipation on FPGAs. It shows promising potential for the design of application-specific processing cores with high-performance computation engines.'

2. Literature Review

The open-source RISC-V instruction set architecture (ISA) has become more popular due to the rising need for effective and adaptable processor architectures in embedded and edge computing. [15] Researchers may investigate different optimization tactics spanning performance, power efficiency, reliability, and

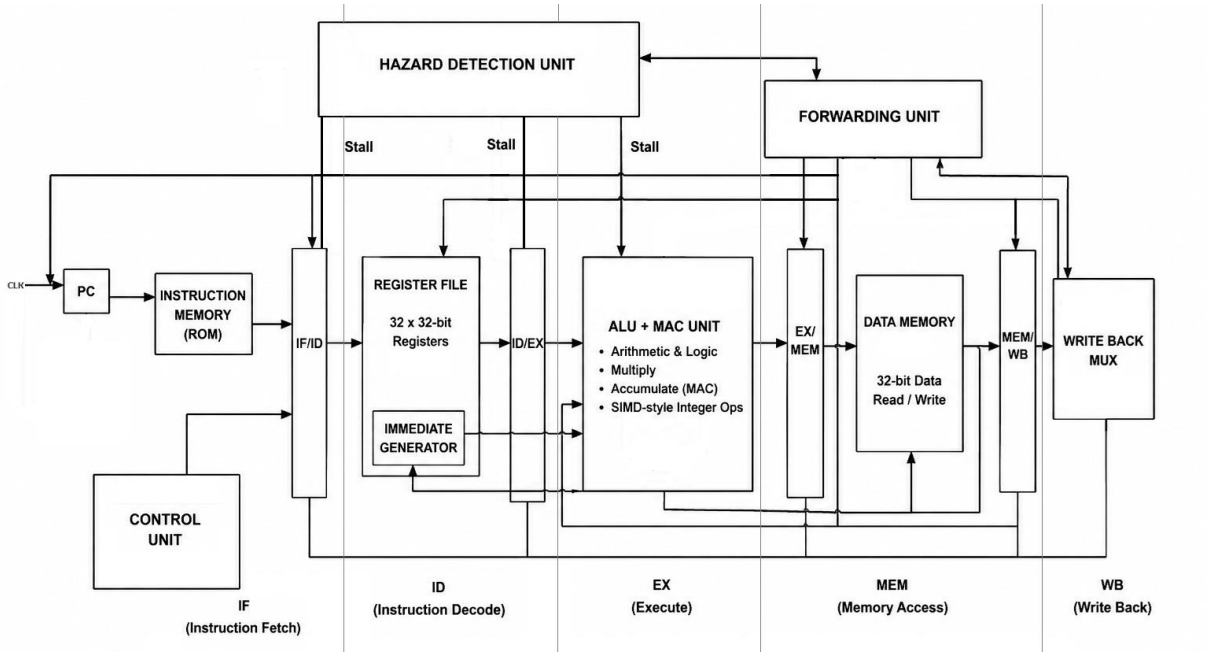

Fig. 1: High level block diagram of the proposed 5 stage pipelined architecture

application-specific acceleration because to its modular and extensible design. [16] Several works in recent literature have addressed these aspects from different perspectives.

Several pieces of research concern themselves with increasing pipeline efficiency and solving performance problems. For example, the recent study on bubble-free pipelining RISC-V architecture (Ray, and K.C) [4] describes ways of removing load-use hazard and thereby greatly boosting CPI and benchmark performance. [14] Pipelining without using out-of-order execution results in increased throughput [13] and is relatively simple architecturally, but the problem remains limited to general-performance improvement only, without any special application-oriented computation enhancement.

However, there is another kind of research which places greater emphasis on simplicity and energy consumption. Specifically, the paper about the bitRISC architecture (Jain, Sharma, et al) [5] introduces BMI to increase efficiency in executing certain instructions in embedded systems. In addition, some processors use a serial data path combined with pseudo-pipelining methods to decrease energy consumption and make hardware simpler. These processors have high energy efficiency and can be effectively used in IoT nodes; however, they have a much higher CPI and lower throughput because of their simple data paths. Other research endeavors involve designing highly optimized processors based on aggressive microarchitecture enhancements and benchmarking. High levels of performance can be seen in the metrics DMIPS/MHz and CoreMark/MHz [6], where their performance is similar to or even better than that of commercial ARM Cortex processors. Such processors optimize instruction execution and high frequency operations; they are often implemented on both FPGAs and ASICs. Yet, they do not possess specialized computational units for novel tasks such as edge AI.

In addition to performance and power consumption, reliability has been another key topic in RISC-V research. Fault tolerant RISC-V processors with TMR (triple modular redundancy) and DMR (double modular redundancy) schemes (Ray, Shukla, and K.C.) [7] have been developed for mission-critical applications in high-radiation areas. Although they improve system reliability, there is substantial overhead

in terms of area and power consumption; therefore, they are unsuitable for resource-limited embedded systems.

Moreover, there are some works that deal with advancements made at the system level rather than those related to the processor core itself. To give an example, RISC-V-based computing platforms have been utilized to study novel NVM technologies like STT-RAM and racetrack memory [8]. Such researches offer insights into issues of power consumption and scaling, yet do not advance the field of processor microarchitecture per se.

3. Proposed System

From the above literature analysis, it can be clearly seen that current studies concentrate on particular aspects of the design and implementation process such as pipelining optimization, low power implementation, performance evaluation, reliability, and memory subsystems. There are, however, few studies concentrating on combining pipelining efficiency with application-level acceleration, especially those related to recent edge-based AI applications.

In this regard, the current study will be focused on developing a five-stage pipelined RV32I processor with additional instruction extensions dedicated to specific application computations, namely MAC and SIMD-like instructions. Different from the previous processors designed with respect to general-purpose optimization and extremely low-power consumption, the proposed processor will strive to find the best combination of performance and computational abilities within limited resources.

I. Implementation

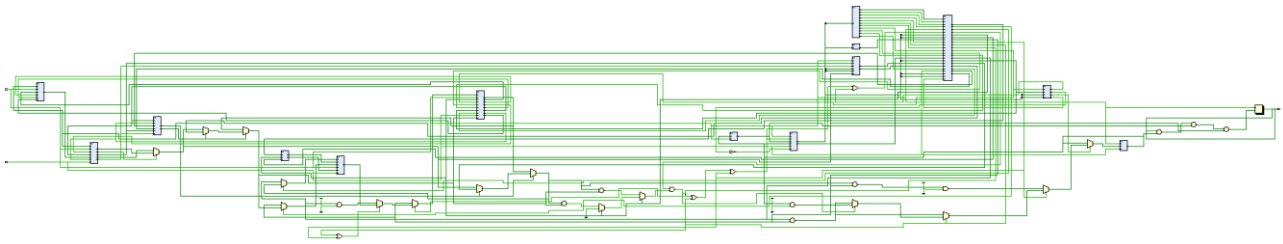

Fig. 2: RTL elaborated schematic of the processor

To facilitate effective execution within the embedded and FPGA systems, the proposed processor design employs a 32-bit pipelined RISC-V (RV32I) CPU implemented using Verilog HDL. The implementation process leverages a Xilinx Artix-7 xc7a35tcsg324-1 device for synthesis and prototyping, which utilizes a conventional 5-stage pipeline configuration. For applications requiring real-time processing capabilities, the selected clock frequency of 100 MHz achieves a balance between reliability and efficiency.

The multiply (MUL) instruction constitutes one of the arithmetic instructions from the M-extension supported by the processor besides the basic RV32I instruction set. Increased processing capability is achieved through this strategic inclusion without increasing the level of complexity in hardware resources. Input programs written in hexadecimal machine codes can be processed by the developed instruction decoder, while instruction execution follows conventional RISC-V instruction coding.

The instruction memory will be structured such that its size is 32 KB and therefore able to hold up to 8192 instructions of 32-bit width. This structure is made possible by mapping the instruction memory onto BRAM on the FPGA such that it enables efficient fetching of instructions. Data memory and register file structures will be formed from distributed memories within the FPGA chip.

The processor structure is defined by a modular architecture, where separate Verilog files are provided for important modules including the program counter, instruction memory, control unit, register file, ALU, and pipeline registers. Hazard detection and data forwarding modules are included to maintain proper execution flow and avoid stalls in the pipeline. Simulations and syntheses will be done with standard FPGA-based tools.

Instead of implementing the entire set of multiplication and division instructions, this version simply uses

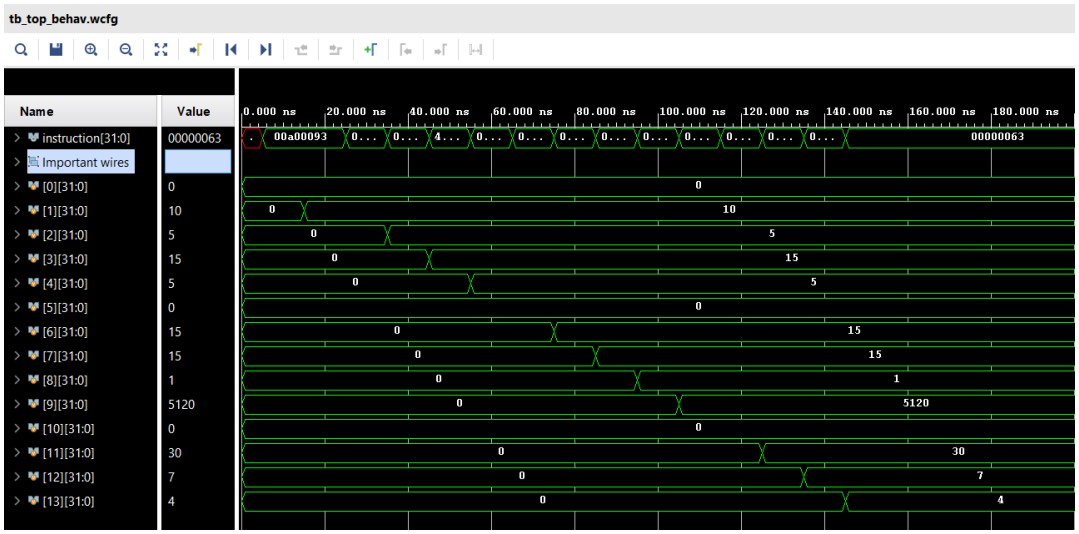

Fig. 3: Waveform obtained in Vivado validating the operation of the processor

the MUL instruction from the M extension and adds a special Multiply-Accumulate (MAC) instruction. The intended application domain of edge-AI acceleration, hardware efficiency, and architectural simplicity all play a role in this design choice. The memory subsystem is closely linked with the control and hazard detection units to guarantee efficient pipeline execution. Additionally, by avoiding data from subsequent pipeline stages wherever feasible, forwarding methods aid in lowering performance penalties.

4. VERIFICATION

The functional correctness of the 32-bit pipelined RV32I processor was checked by extensive simulation and testing done using the Xilinx Vivado Simulator. Verification was done at the RTL level to ensure that all the pipeline stages, control unit, and data path operate correctly in different instructions.

A special testing environment was created to provide clock and reset signals to the circuit, load instructions into the instruction memory, and capture outputs from the processor while executing them. In this respect, the system clock was set at 100 MHz and was generating periodic toggle signals for use by the circuit. Further, an asynchronous reset was used to initialize all the registers and pipeline stages to specific values

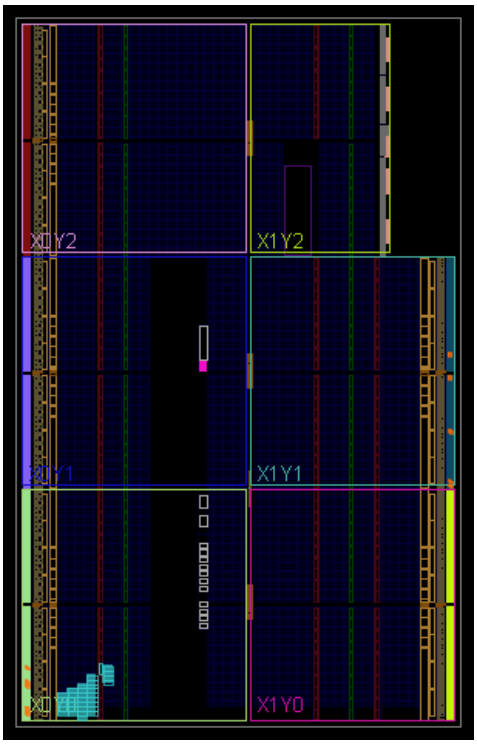

Fig. 4: Total utilization represented in the overall target board area, shaded in cyan

at the beginning of simulation. Machine instructions were loaded into the instruction memory in hexadecimal notation.

A series of testing programs was developed to verify the various functional blocks of the processor. Arithmetic operations such as addition, subtraction, bit-wise AND, OR, XOR, and shifting instructions were implemented using a sequence of instructions to make sure ALU worked correctly. The functionality of loading and storing instructions was also examined by executing memory operations with different bit widths to make sure the data was properly aligned.

Particular attention was paid to the verification of the arithmetic extensions added to the processor in the current version. Multiply and multiply–accumulate operations were performed iteratively and checked for proper data accumulation and interaction with the ALU pipeline. The correctness of these operations was made sure using comparisons of simulation output registers with the expected register values.

Pipeline analysis was conducted by looking at waveforms. All instructions should pass all five pipeline stages correctly: Instruction Fetch (IF), Instruction Decode/Read Instruction Registers (ID), Execution/Read Data Registers (EX), Memory Access (MEM), and Write Back (WB). To make sure the data hazards were handled correctly, we executed dependent instructions sequences and inspected simulation waveforms. Debug outputs, such as program counter updates and register write-back operations,

were also included to track internal processor states. These outputs made it easier to validate pipeline transitions and instruction execution step-by-step.

II. Results and Analysis

A. Functional Verification

The pipeline RV32I processor with MAC support was tested using simulation in Xilinx Vivado, and the test outcomes show that instructions have been executed appropriately, data paths operate well, and all hardware components have been integrated properly. During the stage of testing in a single cycle implementation, all instructions of RV32I have been tested and shown to work fine. Outputs were calculated appropriately by the ALU, registers worked well, and the program counter has incremented well. Instruction decoder has been able to produce control signals for all types of instructions, namely R-type, I-type, S-type, B-type, U-type, and J-type instructions.

Within the pipelined design, waveform analysis verifies the correctness of the instruction flow at the five different pipeline stages, namely Instruction Fetch (IF), Instruction Decode (ID), Execute (EX), Memory (MEM), and Write Back (WB). Overlapping of instructions occurs during the execution process, whereby the different stages are synchronized appropriately. Pipeline write-back ensures that the correct output is achieved. The hazard detection techniques have been verified to function properly within the system. Data hazards are solved via forwarding and stalls to ensure that no incorrect instruction flow occurs because of data dependency issues.

The MAC unit was also proven to be successful. The simulation showed that multiplication and accumulation processes were successful for different inputs. The MAC unit increases the efficiency of the calculations by decreasing the number of instructions used in mathematical tasks.

B. Performance and Area Analysis

The designed architecture has been successfully synthesized and tested for hardware utilization. The hardware analysis shows that the designed architecture has highly efficient resource utilization.

The designed architecture utilizes only 313 Slice LUTs (1.50%) and 201 Slice Registers (0.48%) from the total FPGA resources, indicating its compactness. In addition to this, one Block RAM (BRAM) is used (1%) for instruction memory, which helps in fetching instructions efficiently. Furthermore, no DSP blocks have been used as arithmetic operations are handled by logic units. This proves that the designed architecture is highly efficient but still can be enhanced through use of DSPs for MAC operations. Also, 10 bonded I/Os pins (4.76%) and one BUFG (3.13%) clock buffer are utilized. Hence, total FPGA resource utilization is 10.87%.

TABLE I. FPGA Resource Utilization Summary

Resource Type	Used	Available	Utilization
Slice LUTs	313	20,800	1.50%
Slice Registers	201	41,600	0.48%
Block RAM (BRAM)	1	100	1.00%
DSP Blocks	0	90	0.00%

Bonded I/O	10	210	4.76%
BUFG (Clock Buffer)	1	32	3.13%
Total Utilization			10.87%

C. Timing Analysis

The static timing analysis indicates that the design satisfies all timing requirements for operation at the targeted clock speed of 100 MHz. The value of the Worst Negative Slack (WNS) is +3.661 ns, meaning there is enough timing slack to ensure that there are no setup timing violations. There is also an absence of timing violations in the hold time analysis because the Worst Hold Slack (WHS) is +0.087 ns, which indicates that the data is captured reliably. There are also no pulse width violations because the Worst Pulse Width Slack (WPWS) is +3.750 ns.

TABLE II. TIMING SUMMARY

Parameter	Value
Clock Requirement	10 ns
Worst Negative Slack (WNS)	+3.661 ns
Total Negative Slack (TNS)	0.000 ns
Worst Hold Slack (WHS)	+0.087 ns
Total Hold Slack (THS)	0.000 ns
Worst Pulse Width Slack	+3.750 ns
Maximum Frequency Achieved	193 MHz

D. Power Analysis

The power consumption estimated via Vivado tool amounts to a total on-chip power consumption value of 0.075 W, signifying that the project can be considered to be a very efficient project in terms of energy consumption. Dynamic power consumption has been measured at only 0.004 W, whereas static power consumption is at its peak, measuring up to 0.072 W. The small dynamic power consumption rate is owed to the decreased switching of components. In terms of temperature, a junction temperature of 25.4°C is achieved, which is very near the ambient temperature value.

TABLE III. POWER REPORT

Parameter	Value
Total On-Chip Power	0.075 W
Dynamic Power	0.004 W
Static Power	0.072 W
Junction Temperature	25.4 °C
Max Ambient Temperature	84.6 °C
Confidence Level	Medium

COMPARATIVE ANALYSIS

The pipelined RV32I architecture is compared to state-of-the-art RISC-V processors concerning performance, energy efficiency, and computing ability. Recent literature such as the design of bubble-free pipelined RV32I processors [4] presents modest CPI enhancement (19%) without any domain-specific

accelerations. Lightweight processors like bitRISC [5] and SDPF pseudo-pipelined structures [6], which aim at energy savings and simplicity, have high CPIs due to the lack of deep pipelining.

RISC-V processors for high performance [7] are characterized by higher efficiency on benchmarks (4.13 CoreMark/MHz), but no accelerator structures specific to any applications have been employed. Likewise, the fault-tolerant multi-core structure [8] uses redundant cores to provide higher reliability but at the expense of higher area consumption and energy efficiency, and system-oriented memory-based approaches [9].

The suggested architecture, on the other hand, integrates a Multiply-Accumulate (MAC) unit and SIMD-style improvements for increased computational efficiency while achieving a balanced design with CPI = 1 and low FPGA power consumption (~75 mW). The suggested solution is more appropriate for edge-AI workloads than previous efforts since it incorporates lightweight AI-oriented operations like MAC and ReLU-style computing capabilities within a pipelined RV32I framework. It provides a good trade-off between performance, power, and computational capabilities on FPGA even though it does not yet have fault-tolerant redundancy or conventional benchmark validation.

TABLE IV. COMPARISON OF EXISTING WORKS AND PROPOSED PROCESSOR

Work / Reference	Architecture Type	Key Contribution	Performance (CPI / Benchmark)	Power / Efficiency	AI / Compute Capability	Limitations
LW Bubble-Free RV32I (2025) [4]	5-stage pipelined	Eliminates load-use stalls	~19% CPI improvement, CoreMark ↑	Moderate	None	No AI acceleration, limited scope
bitRISC (BMI-based) [5]	Single-cycle	Custom bit manipulation instructions	High CPI (single-cycle)	Low power, simple	None	No pipelining, limited performance
SDPF Pseudo-Pipeline [6]	Serial datapath, 2-stage	Ultra-low power design	CPI ≈ 36 (very high)	Very low power	None	Extremely low throughput
High-Performance RISC-V Core [7]	Optimized pipeline	Benchmark-driven optimization	1.71 DMIPS/MHz, 4.13 CoreMark/MHz	Very high efficiency (µW/MHz)	None	No application-specific acceleration
Fault-Tolerant Quad-Core (DMR/TMR) [8]	Multi-core (4 cores)	Reliability via redundancy	Performance penalty (~10%)	Higher power/area	None	High complexity, area overhead

NVM Evaluation Platform [9]	RISC-V + memory emulator	Evaluates emerging memory systems	Benchmark-driven	Energy-efficient memory focus	None	Not focused on processor design
Proposed Work	5-stage pipelined RV32I	MAC + SIMD + efficient datapath	CPI ≈ 1, high throughput	Low FPGA power (~75 mW)	MAC + SIMD + ReLU	No benchmark standardization, no fault tolerance

The comparison shows that current research mostly concentrates on certain areas including fault tolerance, instruction set expansions, low-power design, and pipeline optimization. By combining a pipelined architecture with application-specific computational improvements like MAC and SIMD operations, the suggested processor, on the other hand, offers a balanced design. Because of this, the suggested approach is more suited for developing edge-AI and embedded computing applications.

CONCLUSION

The processor design is a 32-bit RV32IM pipelined processor based on Verilog HDL designed on a Xilinx Artix-7 FPGA. The design guarantees efficient implementation of functionality with a robust 5-stage pipeline, hazards resolution, and MAC operation support. From the findings, there is reduced resource utilization (~10.87%) on the FPGA, stable operating speed of 100 MHz, and low power dissipation (~75 mW). Generally, the processor design shows a balance of efficiency, power optimization, and enhanced functionality that qualifies it for embedding applications.

FUTURE WORK

In future studies, inclusion of cache memory hierarchy and branch prediction as well as proper benchmarking against other popular RISC-V processors using CoreMark/Dhrystone will be necessary. Furthermore, more enhancements such as extended SIMD and AI accelerator modules can be added to it alongside SoC integration.

REFERENCES

1. S. K. Narla, V. Yamsani, R. K. Gurrala, K. Jamal and C. U. Kumari, "RISC-V Processor Design: RTL to GDSII Implementation Using EDA Tools," 2025 International Conference on Smart & Sustainable Technology (INCSST), Chikodi, India, 2025, pp. 1-6, doi: 10.1109/INCSST64791.2025.11210272.
2. T. -D. Ta, Q. -T. Tran, C. -K. Pham and D. -H. Le, "Implementation of a Compact and Resource-Efficient 32-bit Microcontroller Based on RISC-V Steel on FPGA and ASIC," 2025 IEEE Region 10 Symposium (TENSYMP), Christchurch, New Zealand, 2025, pp. 1-5, doi: 10.1109/TENSYMP63728.2025.11144958.
3. S. Ahmed and A. B. M. Harun-Ur-Rashid, "Design, Implementation and Verification of Five Stage Pipeline RISC-V Core (RV32I ISA)," 2025 International Conference on Electrical, Computer and Communication Engineering (ECCE), Chittagong, Bangladesh, 2025, pp. 1-6, doi: 10.1109/ECCE64574.2025.11013913.

4. S. Kumar and K. C. Ray, "Micro-Architecture of LW Driven Bubble-Free Five-Stage Pipelined RISC-V Processor Core for Energy Constraint Low-End Application," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 73, no. 1, pp. 207-215, Jan. 2026, doi: 10.1109/TCSI.2025.3629838.
5. V. Jain, A. Sharma and E. A. Bezerra, "Implementation and Extension of Bit Manipulation Instruction on RISC-V Architecture using FPGA," 2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT), Gwalior, India, 2020, pp. 167-172, doi: 10.1109/CSNT48778.2020.9115759.
6. S. Shukla and K. C. Ray, "A Low-Overhead Reconfigurable RISC-V Quad-Core Processor Architecture for Fault-Tolerant Applications," in IEEE Access, vol. 10, pp. 44136-44146, 2022, doi: 10.1109/ACCESS.2022.3169495.
7. S. Bora and R. Paily, "A High-Performance Core Micro-Architecture Based on RISC-V ISA for Low Power Applications," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 68, no. 6, pp. 2132-2136, June 2021, doi: 10.1109/TCSII.2020.3043204.
8. Y. Zhao, S. Ullah, S. S. Sahoo and A. Kumar, "NvMISC: Toward an FPGA-Based Emulation Platform for RISC-V and Nonvolatile Memories," in IEEE Embedded Systems Letters, vol. 15, no. 4, pp. 170-173, Dec. 2023, doi: 10.1109/LES.2023.3299202.

E. *Taheri, S. Bayat-Sarmadi and S. Hadayeghparast, "RISC-HD: Lightweight RISC-V Processor for Efficient Hyperdimensional Computing Inference," in IEEE Internet of Things Journal, vol. 9, no. 23, pp. 24030-24037, 1 Dec.1, 2022, doi: 10.1109/JIOT.2022.3191717.*

9. Y. Zhang, Z. Guo, J. Li, F. Cai and J. Zhou, "AnnikaCore: RISC-V Architecture Processor Design and Implementation for IoT," 2021 IEEE 15th International Conference on Anti-counterfeiting, Security, and Identification (ASID), Xiamen, China, 2021, pp. 200-203, doi: 10.1109/ASID52932.2021.9651690.
10. Sustainable RISC-V Processor with Intrinsic PUF for Edge AI," 2025 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Kalamata, Greece, 2025, pp. 1-6, doi: 10.1109/ISVLSI65124.2025.11130297.
11. P. Haribabu, G. Sasirekha, M. Rao, J. Bapat and D. Das, "RISC-V Core for Ethical Intelligent IoT Edge: Analysis & Design Choice," 2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing Workshops (CCGridW), Bangalore, India, 2023, pp. 110-117, doi: 10.1109/CCGridW59191.2023.00031.
12. P. R. Nikiema et al., "Towards Dependable RISC-V Cores for Edge Computing Devices," 2023 IEEE 29th International Symposium on On-Line Testing and Robust System Design (IOLTS), Crete, Greece, 2023, pp. 1-7, doi: 10.1109/IOLTS59296.2023.10224862.
13. U. Sainz-Estebanez, U. Martinez-Corral and K. Basterretxea, "Hardware coprocessor integration with NEORV32: characterization for efficient implementation of RISC-V-based AI SoCs," 2024 39th Conference on Design of Circuits and Integrated Systems (DCIS), Catania, Italy, 2024, pp. 1-6, doi: 10.1109/DCIS62603.2024.10769177
14. T. Zhao and Z. Ye, "ZeroVex: A Scalable and High-performance RISC-V Vector Processor Core for Embedded Systems," 2024 IEEE 35th International Conference on Application-specific Systems, Architectures and Processors (ASAP), Hong Kong, Hong Kong, 2024, pp. 32-33, doi: 10.1109/ASAP61560.2024.00018.

15. Caon, Michele & Choné, Clément & Schiavone, Pasquale & Levisse, Alexandre & Masera, Guido & Martina, Maurizio & Atienza, David. (2024). Scalable and RISC-V Programmable Near-Memory Computing Architectures for Edge Nodes. 10.48550/arXiv.2406.14263.
16. S. K. Narla, V. Yamsani, R. K. Gurrala, K. Jamal and C. U. Kumari, "RISC-V Processor Design: RTL to GDSII Implementation Using EDA Tools," 2025 International Conference on Smart & Sustainable Technology (INCSST), Chikodi, India, 2025, pp. 1-6, doi: 10.1109/INCSST64791.2025.11210272.