International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(19s) 2026 pp. 1042-1052
https://doi.org/10.70917/ijcisim-2026-4546
Article

Optimized Pipelined RISC-V Processor with Data Hazard Detection for Verification of Ethernet Controller

Supreetha Rao[1], Satish Bhairannawar[2]

[1]Research Scholar,Department of ECE, SDMCET, Dharwad, VTU, India
e-mail :supree.rao@gmail.com
[2]Professor, Department of ECE, SDMCET, Dharwad, VTU, India.
e-mail :satishbhairannawar@gmail.com

Abstract: High-speed networking systems require more and more embedded SoC (System-on-Chip) architectures for real-time packet processing and protocol translation. However, intense data streaming often causes instruction pipeline pauses that impose tight performance constraints when integrating open-source RISC-V processors with media access control (Ethernet MAC) cores. In this study, we propose the design and FPGA implementation of an Optimized Pipelined RISC-V Processor, designed as a high throughput controller for Ethernet networking environment. It has an optimized 32-bit five stage in-order execution pipeline core with RV32I base instruction set support. The Dedicated Hardware Data Hazard Detection and Forwarding Unit is firmly connected into the execution stages to avoid the CPU bottleneck during high frequency Direct Memory Access (DMA) and Ethernet operations. To improve instruction throughput while preserving low hardware complexity, the suggested solution uses a pipelined data path based on the RV32I instruction set. Additionally, FPGA-specific optimizations are used to improve timing performance while lowering area and power consumption, such as the use of block RAM resources and efficient logic mapping. Further a robust, coverage-driven verification environment developed for an FPGA-based design using SystemVerilog. The effectiveness of the proposed framework is verified against recent state-of-the-art literature. Comparative analysis demonstrates that the proposed work achieved 100% of functional coverage while existing implementations utilizing SystemVerilog structures achieve limited coverage ranging from 90.12% to a maximum of 94.44%. The processor was implemented and evaluated on FPGA platforms of Artix-7 XC7A200T and ZYNQ XC7Z020. Experimental results show that the proposed design can run with a maximum operational frequency of 119.5 MHz on Artix-7 XC7A200T FPGA and 156.37 MHz on ZYNQ XC7Z020 FPGA. The suggested processor delivers frequency gains of 19.5% and 24.1% compared to the existing implementations. The results demonstrate the usefulness of the proposed architectural changes to improve the processor throughput, timing closure and overall FPGA implementation performance.

Keywords: RISC-V; FPGA; Pipelined Processor; Verilog HDL; Embedded Systems

1. Introduction

RISC-V is a design that emphasizes a short, straightforward instruction set that may be efficiently implemented in hardware. Its modular architecture enables designers to incorporate only the instruction sets and expansions they require. This makes it appropriate for a wide range of use cases, from low power embedded systems to high performance computing platforms.The standard RISC-V ISA has integer instruction sets, such as RV32I, RV64I, and RV128I. It can be extended by optional modules for multiplication and division (M), atomic operations (A), single-precision floating-point (F), double-precision floating-point (D), and compressed instructions (C). The extensibility allows flexibility and scalability, while ensuring compatibility of the software across different implementations.In response to these requirements, the Reduced Instruction Set Computer (RISC) design was created. RISC focuses on

simpler instruction sets faster execution, and lower power consumption compared to Complex Instruction Set Computer (CISC) architectures [1].

Instruction pipelining is a major strategy for enhancing instruction throughput and speed in the architecture of modern processors [2]. A classic pipelined processor breaks instruction execution into multiple stages, typically instruction fetch, instruction decode, execution, memory access, and write-back, so that it can execute multiple instructions concurrently. The method significantly enhances the performance, without increasing the clock frequency. Narrow-width CPUs are increasingly being used for embedded applications with limited resources and low power requirements. Simplicity, low area consumption and energy economy are of the utmost importance. Authors in [3] provided a detailed review of RISC-V CPU verification approaches. Their study underlined the increasing need of effective verification frameworks for more and more sophisticated RISC-V architectures.

The design and implementation of a five-stage pipelined RISC-V processor for better speed and efficiency than a single-cycle architecture. The main advantages are a significant increase of the operating frequency, reduced power consumption and greater throughput due to pipelining and hazard-handling techniques [4]. The drawbacks are higher consumption of FPGA resources due to the added control and forwarding logic and the complexity of the hardware.Authors in [5] designed a five-stage pipelined RV32I RISC-V processor. Implementation via Verilog and verification by FPGA-based simulation. Mechanisms for handling structural, data and control hazards were included in the processor. The experimental results show that the throughput is improved over non-pipelined architectures and pipeline hazards are also resolved successfully. A 5-stage pipelined RISC-V processor design was implementation on Basys-3 FPGA platform [6]. The architecture used a hazard control unit capable of generating stalls and obtained a substantially greater operating frequency than a single-cycle implementation. Their work showed how pipelining may be used to improve the performance of FPGA-based systems.

The work attempts to fill this gap by examining optimization strategies for a five-stage pipeline processor design [7]. It include RISC V jump instruction branch prediction (speed optimization), memory structure (memory access and resource optimization), and division operations based on data correlation (fetch optimization). The performance of the CPU core was tested on CoreMark benchmark using a Field Programmable Gate Array (FPGA) and the effect of enhancements such as branch prediction and caching on the processor performance was analyzed. This work describes a dual-core RISC-V architecture built for energy-efficient AI acceleration at the edge, comprising dynamic sharing of MAC units, frequency scaling (DFS), and FIFO-based resource arbitration [8]. The system consists of two RISC-V cores that share computational resources: a single Multiply-Accumulate (MAC) unit and a shared external memory subsystem. This work offers an optimized RV32I five-stage pipeline processor, NRP, and two optimization strategies to improve the performance of NRP [9]. These are optimization of instruction decoding unit and optimization of branch prediction

Designs such as PicoRV32 and VexRiscv provide configurable and resource-efficient architectures. However, these designs often involve trade-offs between performance and hardware utilization and do not adequately address verification challenges for complex pipeline behaviour [10]. Similarly, hardware aware optimizations in FPGA-based systems improve performance using DSP slices and BRAM but primarily focus on hardware efficiency rather than systematic verification [11].Literature demonstrates that RISC-V provides a flexible foundation for processor implementation, which allows for a large range of design choices using FPGA [12-14]. Most of the previous research focus on performance, power optimization or application specific enhancement, while complete functional verification and systematic validation are not well addressed, especially for pipelined systems. Formal verification approach use simulation-based verification, formal verification, assertion-based verification, and transaction-level verificationoffers mathematical guarantees of correctness, but are computationally demanding and difficult to scale for entire processor systems [15-17].

From the preceding literature, pipeline-based designs have issues like as hazard handling, control complexity and critical route optimization, which are not consistently addressed in the earlier research. Thus a balanced RISC-V

processor design is required considering performance, resource efficiency, architectural completeness and reliable functional verification on diverse FPGA platforms.

In our work, we offer a 32-bit pipelined RISC-V processor architecture that strikes a balance between performance and hardware efficiency for tiny embedded systems and educational platforms.

The main contributions are as follows:

1. Design of an Optimized 5-Stage Pipelined RISC-V Processor with Embedded Ethernet Controller.
2. Designed and integrated specialized hazard detection unit capable of detecting RAW dependency and avoiding wrong execution by stopping of the pipeline.
3. Coverage-driven approaches for functional verification and performance analysis.
4. FPGA implementation and optimization with comparison study showing improvement in speed.

The rest of this paper is organized as follows. A complete literature assessment is performed; problem characterization is also provided, highlighting the limitations in the existing RISC-V CPU architectures. The architecture of the proposed 32-bit pipelined RISC-V processor is then outlined along with its datapath design and pipeline stages. Next, implementation details, simulation results and performance evaluation are discussed, along with a comparative study with the current works. Finally, the report ends with the conclusion which summarizes the important contributions and future scope of the suggested work.

2. Proposed RISC V ARCHITECTURE

The block-level design of the proposed 32-bit pipelined RISC-V processor is shown in Fig 1. The architecture is based on a five stage pipeline. The stages are Instruction Fetch, Instruction Decode, Execute, Memory Access and Write-Back. Pipeline registers are used to separate each step and allow several instructions to be executed at the same time. The instruction fetch stage uses the program counter and instruction memory.

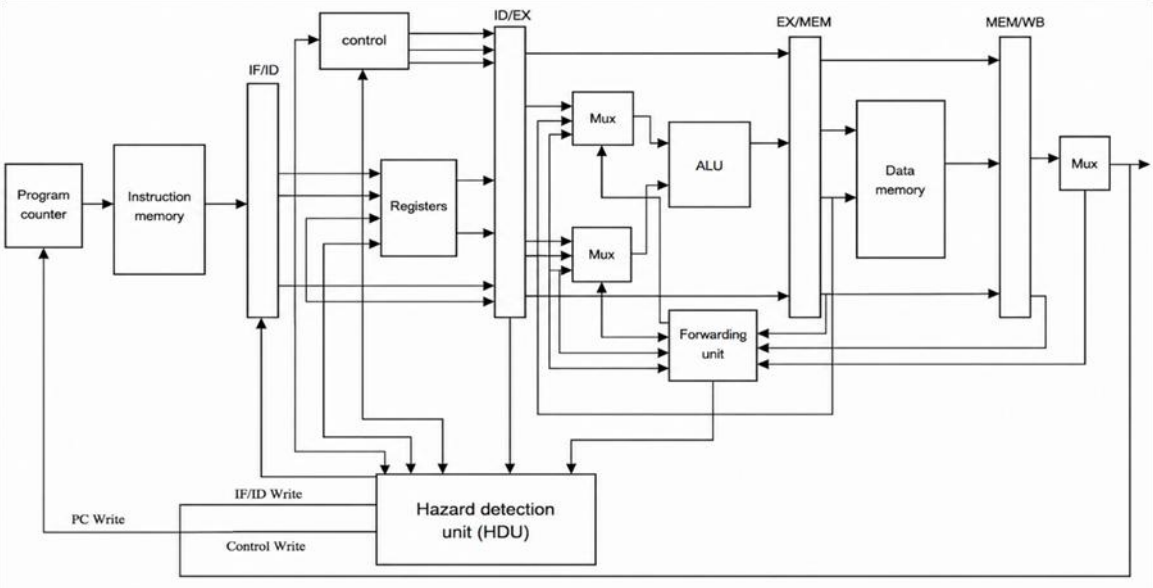

Fig.1. Block Diagram of RISC-V

The decode stage uses the register file and control unit to extract operands and generate control signals. The execute stage has the arithmetic logic unit (ALU) and multiplexers to choose operands. A forwarding unit is used to transfer intermediate results straight to the ALU inputs. This makes pipeline more efficient. The memory access stage interfaces to the data memory for load and store operations and the write-back stage writes into the register file by

using a result picked by a multiplexer. This architecture is intended for low hardware complexity and efficient pipelined operation for low power embedded applications. While processing ethernet packets, it is common to execute instructions such as load/store, arithmetic, logical, branch and jump operations to access controller registers, manage packet buffers, analyze packet headers and handle transmission/reception events. We propose a Hazard Detection Unit that keeps track of relationships between these instructions and eliminates pipeline delays, thus enhancing packet-processing throughput and reducing communication latency. This reduces pipeline stalls by effectively solving hazards, ensures correct access to Ethernet controller registers and packet buffers, reduces latency in packet processing and improves the overall throughput and reliability of real-time network communication systems on FPGA.

Architecturally, the processor can be viewed as a discrete time state machine. At clock cycle t, the state consists of the program counter, the register file, the memory, and the control/status registers. The next state is obtained by executing the current instruction and control logic.

$$S(t+1) = F(S(t), I(t)) \quad (1)$$

$$S(t) = \{PC(t), RF(t), MEM(t), CSR(t)\} \quad (2)$$

Here, S(t) is the processor state at cycle t, I(t) is the instruction being processed, PC is the program counter, RF is the integer register file, MEM represents the memory state and CSR represents control and status registers. This state-transition representation is useful for RTL modelling and formal verification because every clock cycle produces a well-defined next state

2.1 Five stage pipeline

Our solution provides exceptionally modular and adaptable pipeline organizations breaking away from this rigidity exhibiting the 5 steps of pipelining in Fig. 2. This flexibility allows the designer to adjust the pipeline depth and functionality according to the power, area, and performance constraints. One of the major areas of advancement is in the scalability of pipelines. Lightweight coresare usually not pipelined or have very little pipelining to reduce hardware complexity and power consumption. Thus they are suitable for embedded systems.

	T	**2T**	**3T**	**4T**	**5T**	**6T**	**7T**	**8T**	**9T**
Instruction 1	IF	ID	EX	MA	WB				
Instruction 2		IF	ID	EX	MA	WB			
Instruction 3			IF	ID	EX	MA	WB		
Instruction 4				IF	ID	EX	MA	WB	

Fig 2. Five Stage Execution of Instructions

A conventional educational RISC-V implementation can use five stages: Instruction Fetch (IF), Instruction Decode/Register Read (ID), Execute (EX), Memory Access (MEM), and Write Back (WB). For an individual instruction, a simplified cycle model is:

$$C_i = C_IF + C_ID + C_EX + C_MEM + C_WB \quad (3)$$

In an ideal balanced pipeline, the stages overlap. Consequently, after the pipeline is filled, one instruction can complete approximately every clock cycle for a single-issue processor.

$$CPI_ideal \approx 1 \quad (4)$$

2.2 Timing diagram

Fig. 3 illustrates the time sequence for instruction execution in a five stage pipeline processor. Each instruction passes through these steps consecutively. Each of these steps takes one clock cycle. The first stage (T = 0 ns) introduces the first instruction into the pipeline, taking the IF stage. In the next few clock cycles, this instruction moves through the pipeline and fresh instructions are fetched.

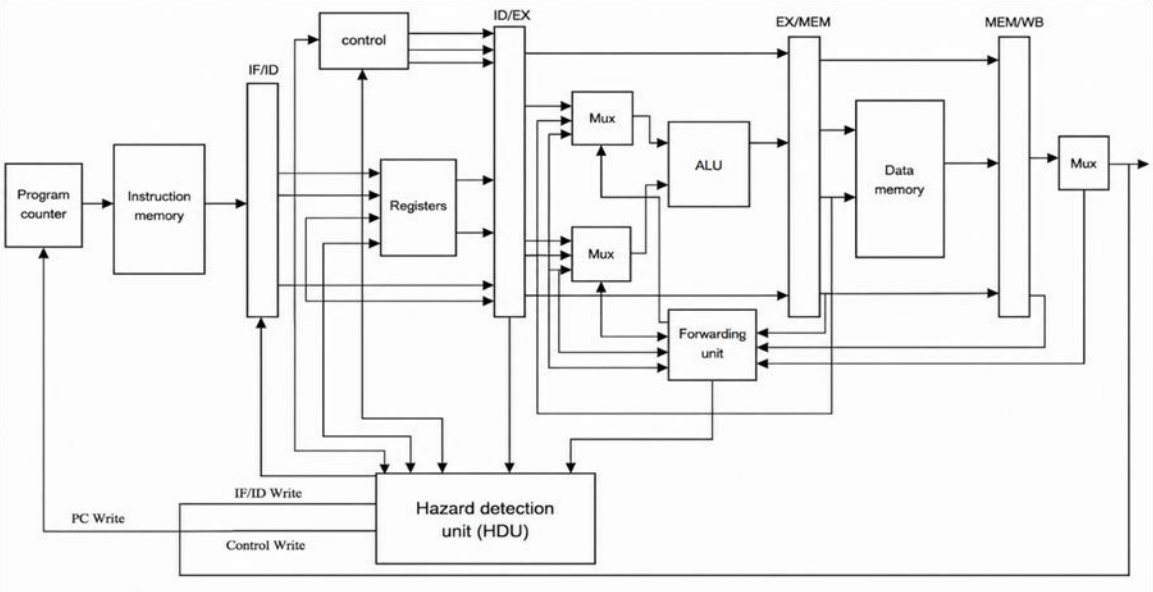

Fig 3. Pipeline Timing diagram

3. RISC-V BASED ETHERNET CONTROLLER VERIFICATION

The RISC-V Based Ethernet Verification Architecture is proposed to verify the functional correctness of an Ethernet Controller by a structured and modular test bench technique is shown in Fig 4. Here, a RISC-V CPU is used as an instruction generator to generate realistic traffic patterns and transactions. These transactions are transferred using a conventional APB bus interface to excite the Ethernet Controller (DUT).

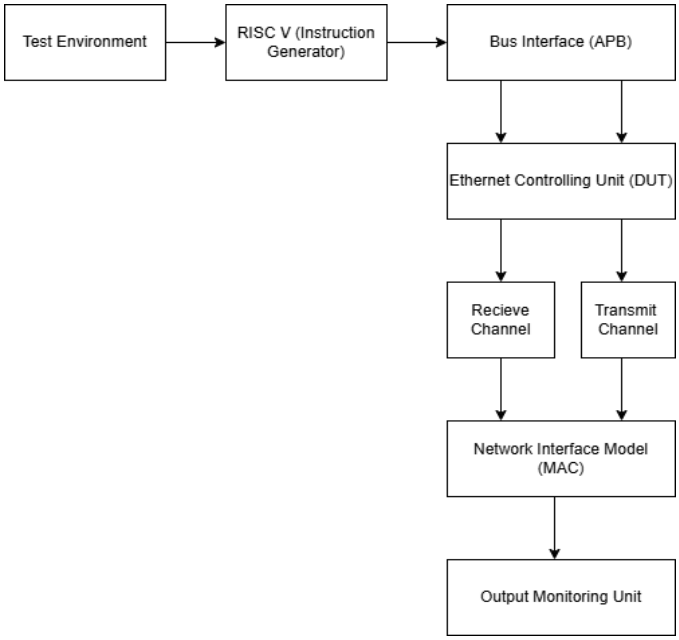

Fig .4. Verification Flow of Ethernet controller

3.1 Ethernet Controller (DUT)

The design under test module is shown in the Fig. 5, which is a Synchronous Dual port RAM. In this case first the data packet with the cyclic code is inserted into the FIFO block which is then stored temporarily into the RAM block.

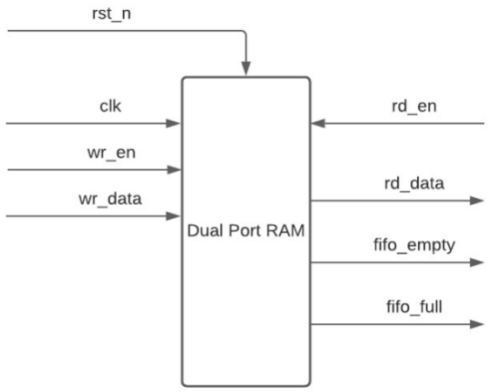

Fig 5. Dual port RAM

The test patterns are generated using the FSM model with random source and destination addresses [18]. This operation allows to transmit and receive Ethernet data and is a basic initialization sequence to verify the interaction between a RISC-V processor [19] and an Ethernet controller.

3.2 Without Data Hazard Test

Cycle	**LW**	**ADD**	**SUB**
1	IF		
2	ID	IF	
3	EX	ID	IF
4	MEM	EX	ID
5	WB	MEM	EX

3.3 With data Hazard Unit

Cycle	**LW**	**ADD**	**SUB**
1	IF		
2	ID	IF	
3	EX	ID	IF
4	MEM	Stall	ID
5	WB	EX	Stall
6		MEM	EX
7		WB	MEM
8			WB

The Hazard Detection Unit inserts a stall (bubble). The stall gives enough time for LW to produce valid data.the result of ADD becomes available before Write Back.Instead of stalling the Forwarding Unit sends the ADD result

directly to the SUB instruction. No extra cycle is needed. This improves performance. Thus, the proposed architecture, with clearly defined functional blocks and a structured verification flow, ensures thorough validation of the Ethernet controller. The integration of RISC-V based stimulus generation enhances realism in testing, while the layered verification components provide high confidence in design correctness and reliability.

Data hazards occur when an instruction depends on the result of an earlier instruction whose result has not yet reached the required pipeline stage. In an in-order RISC-V pipeline, RAW (Read After Write) hazards are the principal data dependency that normally requires forwarding or stalls.

$$C\_hazard = N\_RAW \times P\_RAW \quad (5)$$

$$CPI\_hazard = (N\_hazard \times P\_stall) / IC \quad (6)$$

$$CPI = CPI\_ideal + CPI\_hazard \quad (7)$$

N_hazard is the number of instructions or dependency events that cause a stall, while P_stall is the number of cycles lost per event. If forwarding resolves a dependency without a stall, its effective stall penalty is zero.

4. Results and Disscussions

4.1 Simulation Results

HDL Simulation verifies the functional correctness of the design before synthesis and FPGA implementation.The simulation results were obtained by implementing the Verilog code in Xilinx ISE and this work [20] set us the standard for simulation. The proposed design was implemented on the FPGA platform. The simulation results show that the CPU executes instructions correctly in many situations, including branch operations, sequential execution and data dependencies.

4.2 Top Module

The entire processor was validated by executing a sequence of instructions that covers all supported operations including arithmetic, logical, memory and control instructions as shown in Fig. 6. It is efficient and has five stages which allows instructions to be executed in an overlapping manner. After an initial latency for filling the pipeline, one instruction is executed per clock cycle, indicating better throughput. Data propagation through the pipeline registers and the coordination of control signals are correct as shown by the examination of waveforms. The final outputs obtained from the register file and memory is consistent with the expected findings proving the correctness and robustness of the proposed 32-bit RISC-V processor design.

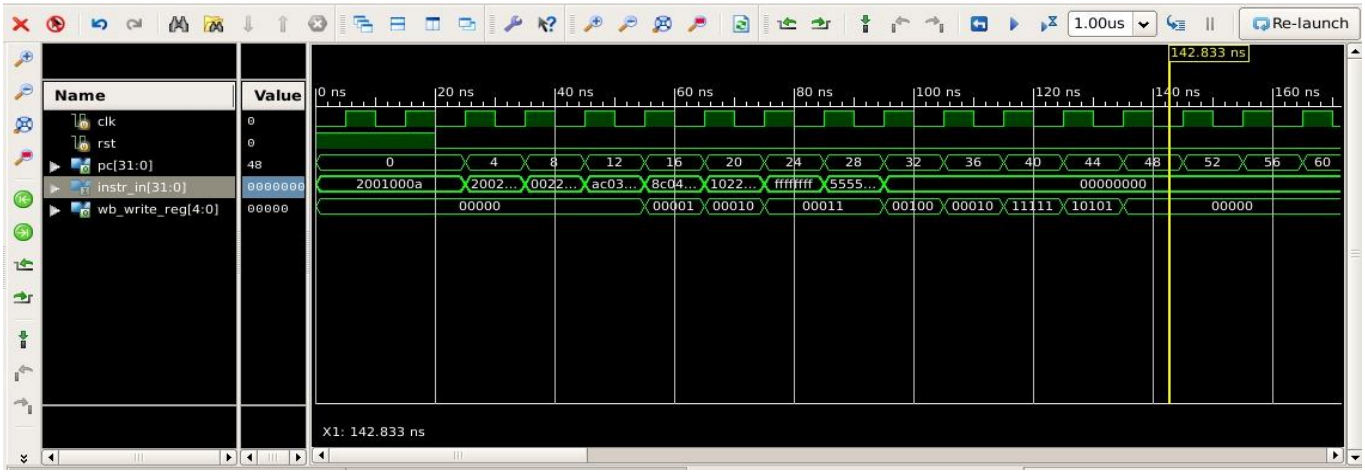

Fig.6. Output of RISC-V on Xilinx

4.3 RTL Schematic Synthesis

The RTL (Register Transfer Level) schematic is a graphical depiction of the hardware structure derived from the HDL verilog code before technology mapping is shown in paper [21]. During the synthesis process it is vital for the designer to verify that the synthesis tool has interpreted the desired hardware accurately w.r.t Slice registers, Look-Up Tables (LUTs), and timing performance with lower hardware consumption and latency. Similar hardware performance measures are taken into account in the current study to assess the suggested design, and this work is a useful reference for FPGA-based digital filter implementation. Fig. 7 shows the RTL schematic of the proposed 32-bit RISC-V processor, which gives a structural representation of the entire datapath and control logic and also shows the connections between all the key modules.

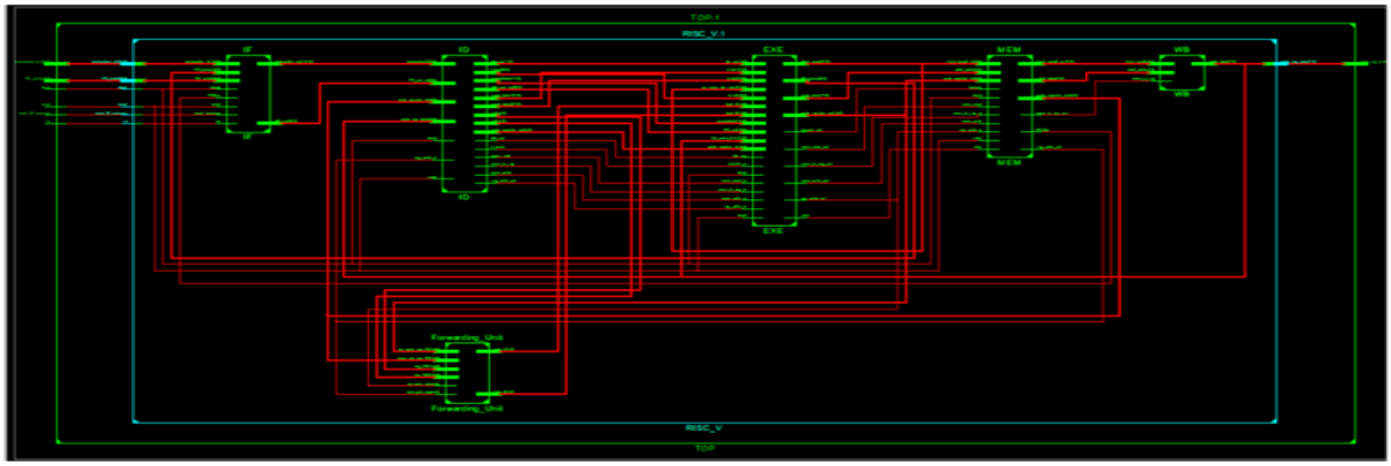

Fig.7. RTL Schematic diagram of RISC-V

4.4 Comparision with Existing works

A quantitative comparison of the proposed 32-bit RISC-V processor with existing FPGA-based implementations is presented in Table I. The comparison includes works Hyun and Ji [22] and Kirali and Fidan [23], representing different design priorities such as area efficiency and balanced architectural implementation.

Hyun and Jiachieved a maximum operating frequency of 100 MHz on the Artix-7 XC7A200T FPGA. In comparison, the RISC-V processor offered reaches 119.5 MHz on the same FPGA hardware, a 19.5 % gain. The improvement is mainly due to the streamlined architecture of five-stage pipeline and the implementation of an efficient hazard identification mechanism. The proposed architecture reduces the data dependency delays and unnecessary pipeline pauses to minimize the critical path and enhance the timing performance. Moreover, the control and datapath units are developed in a modular way, that allows a better synthesis optimization and hence a much higher working frequency.

TABLE I: COMPARISION OF FPGA BASED RISC-V

IMPLEMENTATION

Sl	*Author*	*FPGA Board*	*Speed(Maximum frequency)*
1	Hyun and Ji [22]	Artix-7 XC7A200T	100 Mhz
	Proposed work	Artix-7 XC7A200T	119.5 Mhz
2	Kirali and Fidan [23]	ZYNQ XC7Z020-ICLG400C	126 Mhz
	Proposed Work	ZYNQ XC7Z020-ICLG400C	156.37 Mhz

Further when compared with Kirali and Fidan achieved 126 MHz maximum operating frequency on ZYNQ XC7Z020 FPGA. The proposed CPU achieves 156.37 MHz on the identical hardware, which is 24.1% improvement. This improvement is mainly due to a more simplified datapath organization and enhanced pipeline stage balance, reducing the critical path time between consecutive items. The proposed hazard detection architecture preserves the data dependencies with less hardware overhead successfully. The proposed architecture results in greater clock performance while maintaining the functional correctness. These upgrades make the CPU especially suitable for evaluation of Ethernet controllers and high throughput embedded networking applications.

4.5 Verification Results and Analysis of RISC-V Based Ethernet Controller

The proposed verification methodology is implemented using SystemVerilog, where both the Ethernet controller (design under test) and the verification environment are developed within the same framework. The verification architecture integrates a RISC-V based stimulus generator, which produces realistic instruction-driven transactions for validating the Ethernet controller functionality.The high coverage is achieved due to the use of RISC-V based stimulus generation combined with FSM-driven test case design, which ensures systematic exploration of all possible functional states and transitions.The success of the suggested methodology is further supported by the comparison of the acquired coverage with previous methodologies, as shown in Table II. Previous works employing SystemVerilog and UVM based verification methodologies achieved coverage results of 94.44% and 90.12 % respectively. In comparison, the suggested method achieves 100 % coverage, thereby proving better verification efficiency and completeness.

TABLE II. COVERAGE COMPARISON OF ETHERNET CONTROLLER VERIFICATION APPROACHES.

Authors	**Language**	**Tool**	**% Coverage**
Naidu and Srikanth [24]	UVM	QuestaSim	94.44%
Nelam et al., [25]	System Verilog+ UVM	ModelSim	90.12%
Proposed RISC based Ethernet	System Verilog	ModelSim	100 %

This upgrade is mainly motivated by the inclusion of a programmable RISC-V stimulus generator for flexible and realistic test scenario development and FSM-based test case modelling for a complete study of all operational circumstances. Hence the proposed verification technique provides more confidence on the accuracy and reliability of the Ethernet controller design.

5. Conclusion

In this study, we discuss the design, implementation and verification of a pipelined 32-bit RISC-V processor aimed for resource limited embedded applications. The solution described employs simplified five-stage pipeline architecture to provide higher instruction throughput, while limiting hardware complexity. Theoretically, the performance of the proposed processor is a function of the trade-off between pipeline depth against critical path time.The proposed design has achieved the highest frequency of 119.5 MHz, with 19.5% improvement in comparison with the implementation of Artix-7 XC7A200T. In comparison to the ZYNQ XC7Z020 implementation the designed CPU obtains 156.37 MHz, which is 24.1% better. These improvements are obtained by using an optimized pipelined design, effective hazard detection and forwarding schemes, balanced pipeline stages, and reduced critical-path time. The higher operating frequency boosts the processor throughput and makes the suggested design appropriate for Ethernet controller verification and other high performance embedded applications.

Declaration of competing interest

The authors declare that there are no conflicts of interest regarding the publication of this manuscript. The authors have no financial, professional, or personal relationships that could inappropriately influence the work presented in this study.

Acknowledgment

The authors would like to express their sincere gratitude to the department of electronics and communication engineering and the research supervisors for their valuable guidance, continuous support, and encouragement throughout this research work.

Data availability

The data supporting the findings of this study are available from the corresponding author upon reasonable request.

AUTHORS PROFILE

Dr. Satish S Bhairannawaris the ***Dean*** of Industry Institute Interface (III), and ***Professor*** *of Electronics & Communication Engineering,* SDMCET, Dharwadandalso he served as a visiting faculty for the subject **VLSI Design**, IIIT, Dharwad. He was associated with Prof. L. M Patnaik, SERC, Indian Institute of Science (IISc, Bangalore) for research activities in 2005. He is currently pursuing his **Post-Doctoral Fellowship research at University of Manitoba.** He has served in Department of Electronics and Communication, DSCE, Bangalore before joining to SDMCET. He has been involved in research activities funded by reputed state and central Govt. Organizations and has completed sponsored research projects from DST and K-BITS. Currently, he is in to "AI based precision agriculture for Hardware processing" project funded with Rs.30 Lacs by VGST KFIST L2, Government of Karnataka. He has also contributed to Book Chapter titled: "*Efficient medical Image Enhancement using HSV & CLAHETechnique"*,Soft Computing Medical Image Enhancement–Elsevier Publication. He has over 50 research publications in *SCIE, ECSI & SCOPUS* indexed International Journals and Conference Proceedings and also served as reviewer including IEEE, Springer and Elsevier. He also received **Best paper award** in IEEE international conference organized by VIT, Vellore for the paper titled "Efficient image enhancement using modified adaptive histogram equalization". His papers have over 350 citations with a*h-index* of 12 and *i10-index* of 12. He has conducted many Faculty Development and Student Training program in area of Image Processing using MATLAB, RTL Design using HDL, Advanced VLSI Design and many more in reputed engineering colleges like PES University, KLE university, Dr. AIT, Bangalore etc.

REFERENCES

1. A. Singh, A. Kumar, A. Singh, R. A. Reddy, and K. N. Pushpalatha, "Design and Implementation of RISC-V ISA (RV32IM) on FPGA," *SSRG International Journal of VLSI & Signal Processing*, vol. 10, no. 2, pp. 17–21, 2023.
2. Y. Chang, Y. Liu, C. Peng, J. Guo, and Y. Zhao, "Design of a Configurable Five-Stage Pipeline Processor Core Based on RV32IM," *Electronics*, vol. 13, pp. 120, 2024.
3. C. Tain, S. Patil, and H. Al-Asaad, "Survey of Verification of RISC-V Processors," Journal of Electronic Testing, vol. 41, pp. 111–138, 2025.
4. S. Kalapothas et al., "A Survey on RISC-V-Based Machine Learning Ecosystem," *Information*, vol. 14, pp. 64, 2023.
5. Z. Qin, "Design and Hazard Solving of Five-Stage Pipeline RISC-V Processor Structure," *Applied and Computational Engineering*, vol. 36, pp. 198–203, 2024..
6. Hussain, Faeq and Santanu Sarkar. "Design and FPGA Implementation of Five Stage Pipelined RISC -V Processor." *IEEE 9th International Conference for Convergence in Technology (I2CT)* pp. 1-6, 2024.
7. Jin, Zhiwei, Tingpeng Hu, Zhiyi Jie, and Peng Wang., "Research and Implementation of Performance Optimization Methods for RISC-V Level-5 Processors," *Applied Sciences* , vol.. 21: 11634., 2025. https://doi.org/10.3390/app152111634
8. N. Sulaiman et al., "Design and Implementation of FPGA-Based Systems A Review," *Australian Journal of Basic and Applied Sciences*, vol. 3(4), pp. 3575-3596, 2009.
9. Hongkui Li, Chaoxia Jing and Jie Liu, "Performance-Optimised Design of the RISC-V Five-Stage Pipelined Processor NRP",*International Journal of Advanced Computer Science and Applications (IJACSA)*vol. 15, no. 2*,* pp. 276-281*,* 2024. http://dx.doi.org/10.14569/IJACSA.2024.0150229
10. P. H. W. Leong, "Recent Trends in FPGA Architectures and Applications," in *Proceedings of DELTA*, Hong Kong, China, pp. 137–141, 2008,
11. D. Efnusheva, "FPGA Implementation of RISC-based Memory-centric Processor Architecture," *IJACSA*, vol. 10, no. 9, pp. 6-17, 2019.

12. Ludovico Poli, Sangeet Saha, Xiaojun Zhai, Klaus D. McDonald-Maie, “Design and Implementation of a RISC-V Processor on FPGA,” in *Proceedings of MSN*, Exeter, U.K., pp. 161–166, 2021.
13. P. Manjusha,Prabha Niranjan and Dileep Kumar M J, “Design and Implementation of 32-bit RISC-V Processor Using Verilog,” in *Proceedings of DISCOVER*, pp. 181–186, 2024
14. J. Jeemon, “Pipelined 8-bit RISC processor design using Verilog HDL on FPGA,” in *Proceedings of RTEICT*, pp. 2023–2027, 2016.
15. H. Chippagi, “Evaluation of Coverage Metrics for Assessing Test Suite Effectiveness in RISC-V Core Verification,” *IJRITCC*, vol. 11, no. 9, pp. 5680–5683, Oct. 2023.
16. R. Qiu and Y. Liu, “An Integrated UVM-TLM Co-Simulation Framework for RISC-V Functional Verification and Performance Evaluation,” *arXiv preprint arXiv:2505.10145*, 2025.
17. P. D. Schiavone et al., “An Open-Source Verification Framework for Open-Source Cores: A RISC-V Case Study,” *in Proceedings of VLSI-SoC*, Verona, Italy, 2018, pp. 43–48.
18. S. parameshwara, G.V sagar, H R S Dasappa, and S nagaraj, “Effective ethernet controller protocol architecture verification strategy using system Verilog”, *IJECE*, vol. 14, no.6, 2024.
19. D. A. Patterson and J. L. Hennessy, Computer Organization and Design RISC-V Edition: The Hardware/Software Interface,*1st ed. Cambridge, MA*, USA: Morgan Kaufmann, 2017.
20. A. Eddla and V. Y. J. Pappu, “Optimal IIR Filter Design and FPGA Realization”, *Eng. Technol. Appl. Sci. Res.*, vol. 16, no. 1, pp. 31009–31014, Feb. 2026.
21. T. Saidani, R. Ghodhbani, A. Alhomoud, A. Alshammari, H. Zayani, and M. Ben Ammar, “Hardware Acceleration for Object Detection using YOLOv5 Deep Learning Algorithm on Xilinx Zynq FPGA Platform”, *Eng. Technol. Appl. Sci. Res.*, vol. 14, no. 1, pp. 13066–13071, Feb. 2024.
22. Hyun Woo Kang, Ji Woong Choi, “basic_RV32s: An Open-Source Microarchitectural Roadmap for RISC-V RV32I”,*eprint{2510.15887}*, 2025.
23. K. Kırali and C. B. Fidan, “Implementation of FPGA based 32-bit RISC-V processor,” *Engineering Science and Technology, an International Journal*, vol. 70, pp. 102139, Oct. 2025.
24. K. Jagannadha Naidu and M. Srikanth,”Design and Verification of Slave Block in Ethernet Management Interface using UVM”, *Indian Journal of Science and Technology*, vol. 9, no. 5, 2016.
25. B. G. Nelam *et al*., "Coverage-Driven Verification of Ethernet MAC Using System Verilog and UVM," *IEEE 6th India Council International Subsections Conference (INDISCON)*, Rourkela, India, pp. 1-6, 2025. doi: 10.1109/INDISCON66021.2025.11253760.