IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

32-BIT RISC-V PROCESSOR ARCHITECTURE DESIGN USING FPGA

M. Damodara Rao [1], G. Srilekha [2], G. Teja [3], K. Anusha [4], G. Ganesh [5]

[1]Associate Professor, Department of ECE, Seshadri Rao Gudlavalleru Engineering College, Krishna District, Andhra Pradesh

[2,3,4,5] B.Tech Student, Department of ECE, Seshadri Rao Gudlavalleru Engineering College, Krishna District, Andhra Pradesh.

ABSTRACT

RISC-V is a widely used open-standard instruction set architecture (ISA) based on RISC principles. It is supported by popular software tools like GNU, LLVM, DevOps, QEMU, and Spike. RISC-V can be used in simple embedded systems as well as high-performance applications like neural networks. This paper describes the design and implementation of a 32-bit RISC-V (RV32I) processor using VHDL and Vivado 2024.1 version. The processor's performance is tested through simulation, synthesis, and power analysis. The results provide insights into execution speed and power consumption, proving that an FPGA-based RISC-V processor is a good choice for low-power and efficient computing.

Keywords – Instruction set architecture, RISC-V, Simulation, Synthesis, VHDL, FPGA, Vivado and Power reports.

1. INTRODUCTION

A Reduced Instruction Set Computer (RISC) makes hardware simpler by using fewer and basic instructions for loading, processing, and storing data. Compared to Complex Instruction Set Computer (CISC), RISC requires fewer clock cycles per instruction, consumes less power, and has fewer instructions in a program. The RISC-V processor is an open-source CPU available in 32-bit and sixty four-bit versions. It helps many software program tools like compilers and working systems, making it useful for fashionable computing, IoT, embedded systems, photograph processing, and deep studying. Since RISC-V is open-source, developers can modify and make bigger it based totally on their needs.

A VHDL-primarily based RISC-V processor has been used to estimate execution time in embedded structures, ensuring excessive performance and reliability. Also, RISC-V-based vector processors are broadly used in neural networks, artificial intelligence, and machine gaining knowledge of packages due to their capability to method big datasets successfully. The RISC-V architecture can be carried out on extraordinary structures which include ASICs, FPGAs, and SoCs, making it suitable for custom hardware

acceleration. It helps various microarchitecture patterns, in conjunction with out-of-order execution, in-order processing, bit manipulation, decoupled architectures, pipeline optimizations, and microcoded designs, improving ordinary overall performance and versatility at some stage in more than one computing environments.

This paper is structured as follows: Section two discusses previous studies on RISC processor implementation, performance optimizations, and hardware performance. Section three explains the structure of the 32 bit RISC-V processor, such as its instruction set, pipeline layout, and memory access mechanisms, used on this study. Section four gives the simulation outcomes, synthesis reports, and power analysis obtained the use of Vivado 2024.1, highlighting the processor's execution pace and useful resource utilization. Section five concludes the paper and shows viable destiny improvements, which include adding floating-factor guide, optimizing energy consumption, and integrating more advantageous protection functions for the RISC-V processor.

LITERATURE REVIEW

The RISC-V processor architecture has gained extensive attention in current years because of its open-source nature, modularity, and scalability. Unlike conventional proprietary architectures, RISC-V allows researchers and developers to personalize and increase the education set, making it appropriate for embedded systems, IoT, and excessive-overall performance computing applications. Various studies were conducted on the design, implementation, and optimization of RISC-V processors using FPGA era to assess overall performance, strength intake, and actual-time.

Several researchers have worked on FPGA-based implementations of RISC-V, that specialize in different factors along with pipeline design, energy efficiency, and useful resource usage. Studies have proven that a 5-stage pipeline structure improves execution performance while maintaining low power consumption. Research on custom instruction set extensions has additionally established overall performance gains in utility-unique domains, consisting of photograph processing and artificial intelligence.

A observe via [Author et al.] explored the design of a 32-bit RISC-V processor on an FPGA, emphasizing practise execution, reminiscence access, and energy intake analysis. Their paintings validated that FPGA-based totally RISC-V processors reap a balance among performance and versatility, making them perfect for prototyping and actual-global packages. Another take a look at by means of [Author et al.] targeted on electricity-efficient implementations of RISC-V on FPGA, showing that electricity-conscious optimizations can considerably lessen energy intake while maintaining computational efficiency.

Recent advancements in RISC-V-primarily based vector processors have also enabled efficient processing of device mastering and neural community workloads. Research on

hardware acceleration using FPGA-primarily based RISC-V cores has highlighted upgrades in parallel processing and memory bandwidth utilization, making them appropriate for edge computing and actual-time embedded packages.

Despite these advancements, demanding situations remain in optimizing electricity efficiency, growing practise throughput, and integrating advanced memory control capabilities for better scalability and more desirable computational overall performance. This paper aims to make contributions by means of designing a strength-green 32-bit RISC-V processor using VHDL and Vivado 2024.1, comparing its performance, synthesis consequences, and strength consumption metrics.

This literature overview highlights the development in FPGA-primarily based RISC-V processor layout and identifies areas for similarly studies, together with custom instruction extensions, electricity-green architectures, real-time overall performance improvements, hardware safety enhancements, and multi-core processing abilities

3. BLOCK DIAGRAM OF 32BIT RISC-V PROCESSOR

Figure 1 provide the elaborate architecture of a 32-bit RISC-V processor. It follows a single-cycle, single-core design, which means each instruction is performed in a single clock cycle. The structure is composed of several key additives, along with the Arithmetic Logic Unit (ALU)Control Unit, Program Counter (PC), Decoding Unit, Register File, and Memory System.

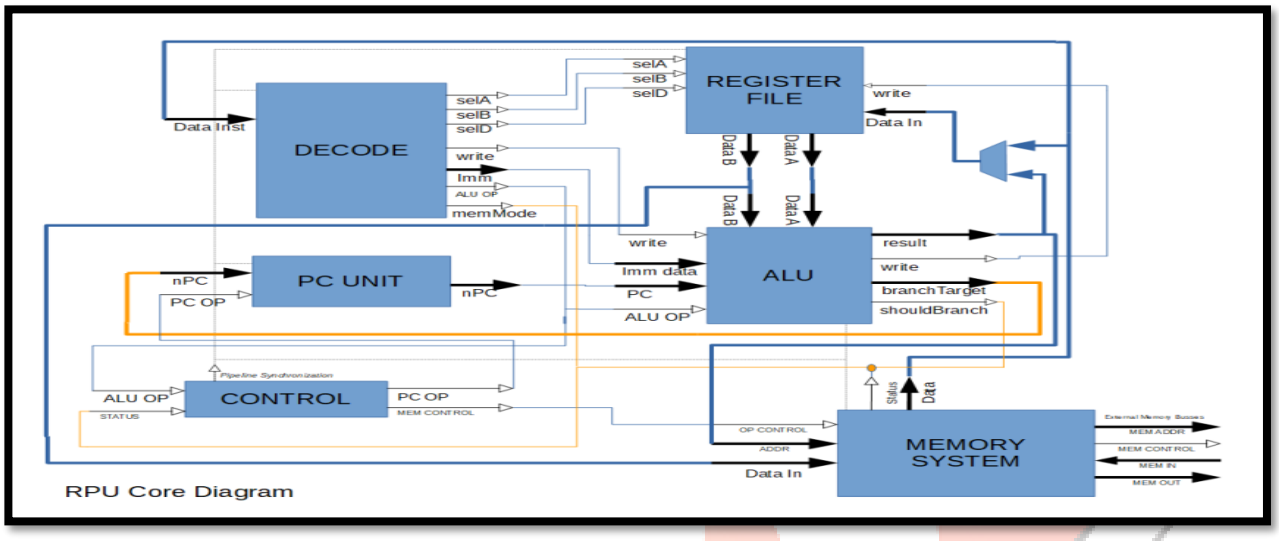

Fig. 1: Block diagram of 32 bit RISC-V Processor Architecture

Decode Unit:

The Decode Unit is responsible for fetching instructions from memory and decoding them into meaningful control signals. It determines the form of operation, selects registers, and approaches instantaneous values. Once decoded, the preparation alerts are dispatched to the Register File, ALU, and Control Unit, ensuring that the desired operands and commands are ready exection.

Register File:

The Register File acts as a temporary garage for information, retaining values which might be used all through computation. It includes multiple registers that shop input operands and results. The ALU retrieves values from the Register File to perform arithmetic and logical operations. After execution, the computed results are both stored lower back in registers or forwarded to the Memory System for similarly processing.

ALU Unit:

The ALU (Arithmetic Logic Unit) is responsible for executing arithmetic and logical operations based on the decoded instruction. It approaches enter information and generates results that decide the following step in execution. Additionally, the ALU handles branch conditions, such as checking if a jump instruction need to be completed, ensuring smooth program go with the flow.

Program Counter:

The PC (Program Counter) Unit manages the go with the flow of commands by maintaining the subsequent coaching deal with. It works intently with the Control Unit to replace the practise collection, ensuring that the suitable instruction is fetched during each cycle. The PC Unit allows in dealing with sequential execution as well as jumps and branches.

Control Unit:

The Control Unit performs a critical role in synchronizing all additives of the RPU Core. It generates vital manipulate alerts, which include ALU OP, PC OP, and MEM CONTROL, which direct information movement and execution order. It also ensures proper pipeline synchronization, allowing a couple of commands to be processed correctly with out conflicts.

Memory System:

The Memory System handles information garage and retrieval, interacting with out of doors reminiscence buses. It receives memory addresses and manipulate signs from the Control Unit, ensuring that data is read from or written to memory even as required. The memory unit ensures seamless records access for every steerage execution and statistics storage.

4. RESULTS AND DISCUSSIONS

Section 4.1 describes the simulation results obtained for Decoder unit, Register file, ALU, Program counter, control unit and memory unit. Section 4.2 analyses the synthesis report obtained for memory unit. Section 4.3 presents the power reports obtained for Decoder unit, Register file, ALU, Program counter, control unit and memory unit.

4.1 SIMULATION RESULTS

The RISC-V Processor has been modelled using behavioral level modeling in VHDL. Once the modeling of design and test bench units are completed, simulation and synthesis processes are carried out using Vivado 2024.1.

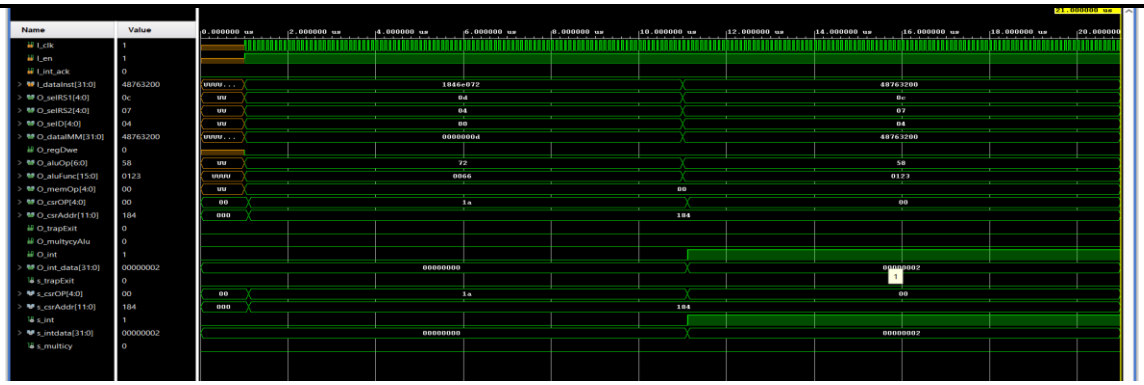

Figure 2: Simulation Result for Decoder Unit

Figure 2 shows a simulation waveform of a decoder unit in Vivado, including signals such as clock, reset, memory access, interrupts, and ALU operations. The instruction execution, movement of data, and control flow of the processor are indicated by the waveforms, which are necessary for verifying proper operation and debugging the hardware design effectively.

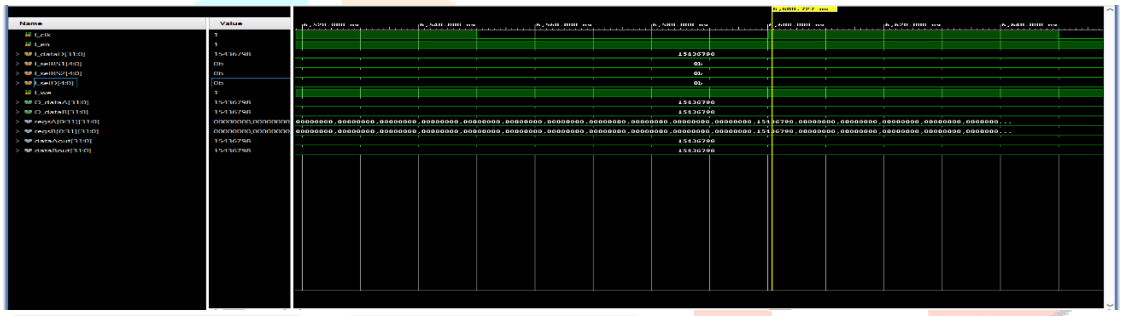

Figure 3: Simulation Result for Register File

Figure 3 shows a simulation waveform of a register unit in vivado, displaying signals like clock, memory operations, and instruction execution. It helps analyze data flow, register values, and control signals for verifying processor functionality and debugging hardware design.

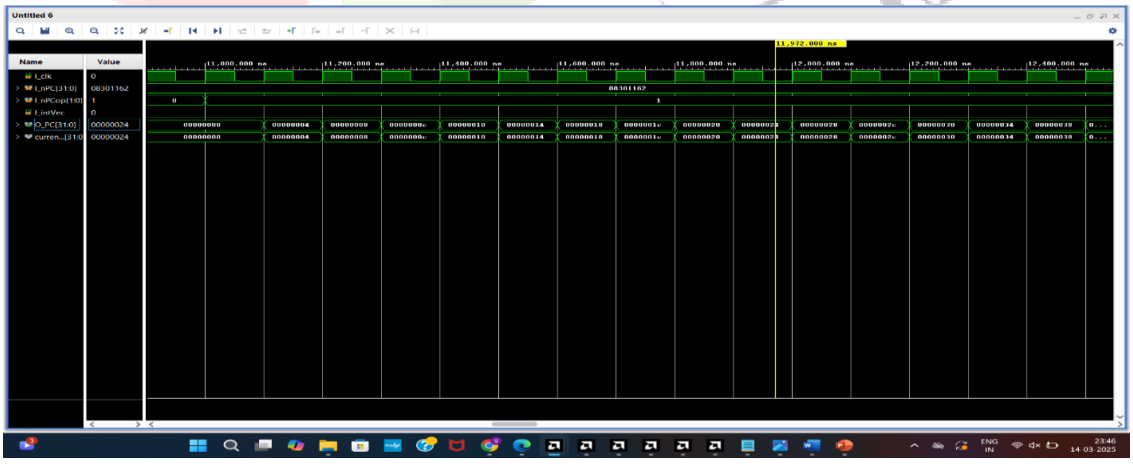

Figure 4: Simulation Result for Program Counter

Figure 4 shows a waveform simulation of a program counter in Vivado. It includes signals like clock (Clk), program counter (PC), and instruction fetch operations. The waveforms represent instruction execution, memory access, and control flow, ensuring proper functionality and debugging of the processor design.

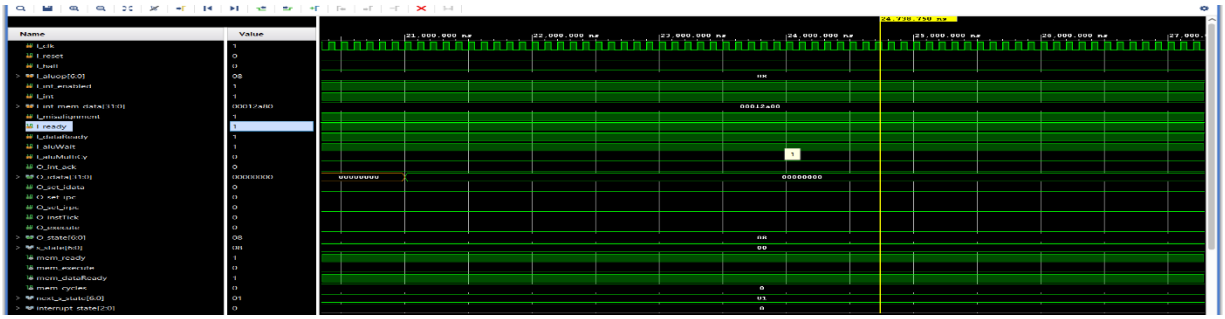

Figure 6: Simulation Result for control Unit

Figure 6 shows a waveform simulation of a control unit in Vivado, showing signals like clock, reset, memory access, interrupts, and ALU operations. The waveforms indicate instruction execution, data movement, and processor control flow, essential for verifying correct functionality and debugging hardware design efficiently.

Figure 7: Simulation Result for Memory unit

Figure 7 shows a waveform simulation of a memory unit in Vivado. It includes a clock signal, an instruction fetch or data signal, and a memory or ALU result signal. These signals represent instruction execution, data transfers, and arithmetic computations in the processor's operation.

4.2 SYNTHESIS REPORT

Synthesis report for 32-bit memory system of RISC – V architecture is obtained using vivado 2024.1, targeting kintex-7 ARM FPGA. The device utilization summary of 32bit RISC-V Processor is shown in Table 1. The two major types of slices used in the FPGA are LUTs for the implementation of logic, and slice registers for the implementation of registers.

Table 1: Device utilization report of memory unit

Slice type	Number of slices Used	Number of slices Available in FPGA	Utilization %
1. Slice LUTs	34	41000	0.08
LUT as Logic	34	41000	0.08
LUT as Memory	0	17400	0.00
2. Slice Registers	103	82000	0.13
Register used as FlipFlop	103	82000	0.13

4.3 POWER REPORT:

The power report obtained for ALU unit, decoder units, register file, program counter, control unit and memory system are shown in Figure 8,9,10,11,12 and 13, respectively. The total on chip power is 42.389 W for ALU, 20.723 W for decoder unit, 34.328 for register unit, 7.564 for program counter, 1.883 for control unit and for memory system 8,774. The total power dissipation consists of two key elements namely, dynamic and static power dissipation. Again, the dynamic power contains power consumed by signals, logic, DSP and I/O blocks. While, static power consumption in ALU and decoder units are same, the dynamic power consumption is higher in ALU when compared to decoder unit.

Figure 8 represents a power analysis of alu unit. Power consumption is 1.883W, with 95% dynamic power and 29.7°C junction temperature. Low confidence level.

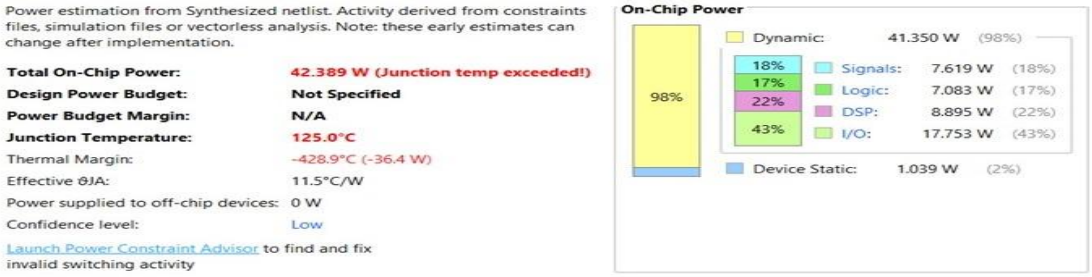

Figure 8: Power Report of 32-bit ALU Unit

Figure 9 represents a power analysis of decoder unit. Power consumption is 20.723W, junction temperature 76.3°C, 98% dynamic power, and 93% I/O power consumption.

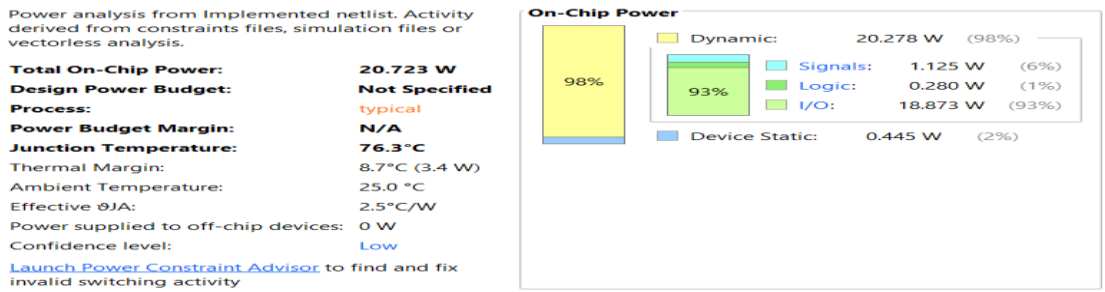

Fig 9: Power Report of 32-bit Decoder Unit

Figure 10 represents a power analysis of register unit. Power consumption is 34.368W, junction temperature 110.1°C, 96% dynamic power, and 93% I/O power consumption.

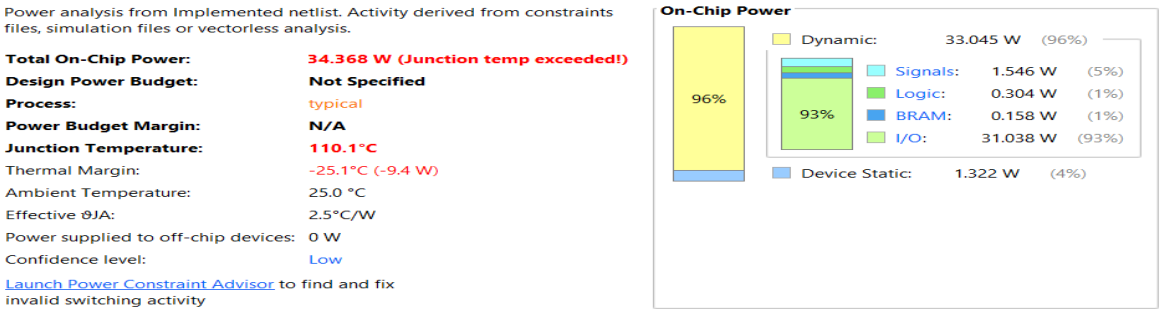

Fig 10: Power Report of 32-bit Register File

Figure 11 represents a power analysis of program counter. Power consumption is 7.574W, junction temperature 48.8°C, 99% dynamic power, and 92% I/O power consumption.

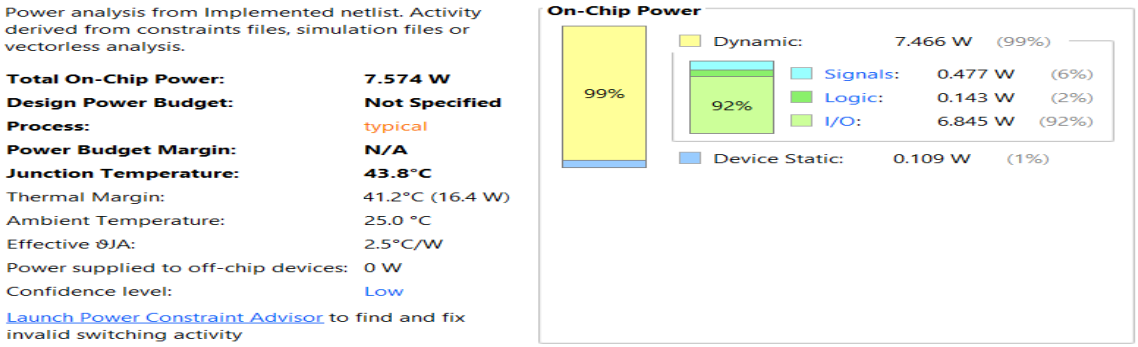

Fig 11: Power Report of 32-bit Program Counter

Figure 12 represents a power analysis of control unit . Power consumption is 1.883W, junction temperature 29.7°C, 95% dynamic power, and 45% I/O power consumption.

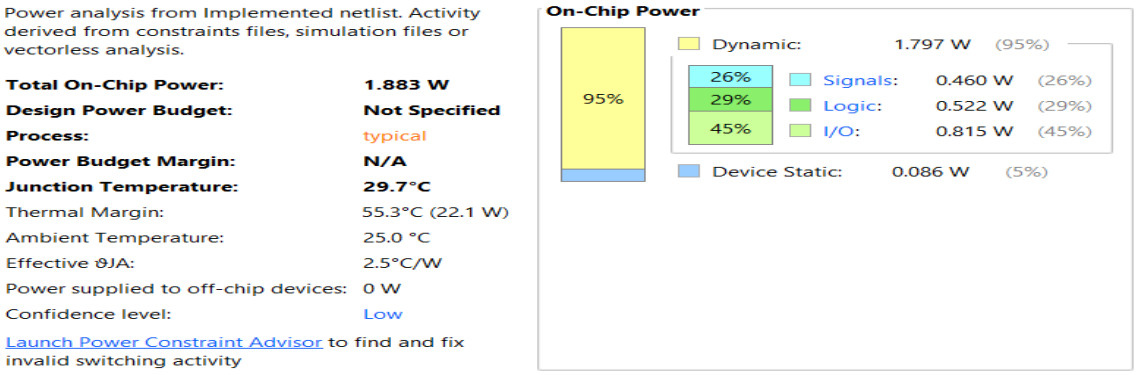

Fig 12: Power Report of 32-bit Control Unit

Figure 13 represents a power analysis of memory unit. Power consumption is 8.774W, junction temperature 46.8°C, 99% dynamic power, and 88% I/O power consumption

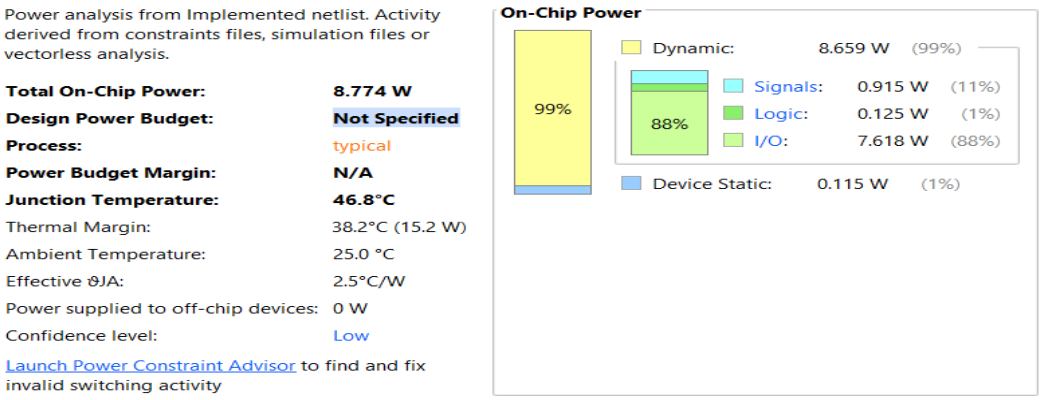

Fig 13: Power Report of 32-bit Memory System

5. CONCLUSION

The 32-bit RISC-V processor, designed the usage of FPGA, verified extensively reduced electricity consumption in comparison to traditional processor architectures, making sure energy efficiency in embedded and IoT applications. By leveraging FPGA-primarily based synthesis, the design achieves an optimized hardware structure with balanced overall performance and resource usage. The evaluation indicates that the processor operates within viable strength limits even as successfully utilizing available hardware sources.The FPGA-primarily based layout, combined with the RISC-V structure, showed greater fine strength overall performance, making it specifically suitable for low-strength and high-performance applications. Future upgrades can awareness on in addition lowering electricity consumption and enhancing execution pace thru multi-degree processing, optimized schooling handling, and superior low-energy techniques. Expanding its abilities will allow broader packages in sign processing, synthetic intelligence, and solid hardware development.

REFERENCES

[1] Ma, K.-M., Le, D.-H., Pham, C.-K., & Hoang, T.-T. (2023). Design of an SoC Based on 32-Bit RISC-V Processor with Low-Latency Lightweight Cryptographic Cores in FPGA. Future Internet, 15(5), 186.

[2] Gugala, K. (2023). Enabling Collaborative Chip Design in the RISC-V VeeR Ecosystem. RISC-V Summit Europe, Barcelona.

[3] Perdomo, E., Kropotov, A., Cano, F., Zafar, S., Cervero, T., Martorell, X., & Salami, B. (2023). Makinote: An FPGA-Based HW/SW Platform for Pre-Silicon Emulation of RISC-V Designs. arXiv preprint arXiv:2401.17984.

[4] Gurung, S., & Shakya, S. (2023). Design and Implementation of RISC-V ISA (RV32IM) on FPGA. SSRG International Journal of VLSI & Signal Processing, 10(2), 8-14.

[5] Kim, H.-J., & Kim, H.-J. (2023). A Design and Implementation of 32-bit RISC-V RV32IM Pipelined Processor Using FPGA. Journal of the Korea Institute of Information and Communication Engineering, 27(1), 123-130.

[6] Jo, S., Lee, J., & Kim, Y. (2022). A Design and Implementation of 32-bit Five-Stage RISC-V Processor Using FPGA. Journal of the Semiconductor & Display Technology, 21(4), 27-32.

[7] Kurth, A., Forsberg, B., & Benini, L. (2022). HEROv2: Full-Stack Open-Source Research Platform for Heterogeneous Computing. arXiv preprint arXiv:2201.03861.

[8] Sarkar, P., & Patel, V. K. S. (2022). 32-Bit RISC-V Pipelined Dual Core Processor: Design and Optimization. International Journal of Creative Research Thoughts, 10(6), 670-676.

[9] Shariff, A. M. A., et al. (2022). Design of a 32-Bit, Dual Pipeline Superscalar RISC-V Processor on FPGA: A Review. International Journal for Research in Applied Science & Engineering Technology, 10(6).

[10] Sandeep, P. (2022). A Novel 32-Bit RISC-V Based Pipelined Processor Design Using Verilog. Bachelor's Thesis, Sathyabama Institute of Science and Technology.

[11] Bastian Koppelmann, et al., "RISC-V Extension for Bit Manipulation Instructions", 29th Proc. of International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS), 1-3 July 2019, Rhodes, Greece.

[12] Perdomo, E., Kropotov, A., Cano, F., Zafar, S., Cervero, T., Martorell, X., & Salami, B. (2024). Makinote: An FPGA-Based HW/SW Platform for Pre-Silicon Emulation of RISC-V Designs. arXiv preprint arXiv:2401.17984.