DOI: 30.0485/ijearst.10.01.3494 https://ijearst.co.in/

ENGINEERING IN ADVANCED RESEARCH SCIENCE AND TECHNOLOGY

ISSN 2386-7876
Vol.02, Issue.02
July-2026
Pages: 1120-1131

Power-efficient and Area-optimized Exposed Datapath Architecture Enhanced with a RISC-V Instruction Set

[1]K. HEMANTH SURYA MANISAI, [2]SATHIBABU MANDAPALLI
[1]M. Tech, Dept. of ECE, V S M College of Engineering, Ramachandrapuram, A.P, India.
[2]M. Tech, (Ph. D), Assistant Professor, Dept. of ECE, V S M College of Engineering, Ramachandrapuram, A.P, India.

ABSTRACT: This project describes a 32-bit RISC microprocessor core that has been designed for portable applications. To implement these instructions the design incorporates various design blocks like Control Unit (CU), Arithmetic and Logic Unit (ALU), Accumulator, Program Counter (PC), Instruction Register (IR), Memory and additional logic. The RISC is designed to enhance processor performance by keeping the following goals in mind. The RISC uses simple constructs and has small instruction set. It is basically designed in order to achieve faster executions. 32bit RISC processor is implemented with following specifications. Along with RISC architecture, power efficient register management method is introduced for data accessing process. Ripple carry adder is used to implement addition operation, separate General subtraction block is used to produce subtraction output of operands. Serial multiplier is used for multiplication purpose. And some basic gates are design for implementation of logical operations. Existing technique is enhanced with fully power optimized architecture in register designing. In the ring counter, double-edge-triggered (DET) flip-flops are utilized to reduce the operating frequency by half and the C-element gated-clock strategy is proposed. A novel gated-clock-driver tree is then applied to further reduce the activity along the clock distribution network. Moreover, the gated-driver-tree idea is also employed in the input and output ports of the memory block to decrease their loading, thus saving even more power.

Keywords: Exposed Datapath Architecture, RISC-V Instruction Set, Low-Power Processor Design, Area Optimization, Energy-Efficient Computing, Embedded System Architecture, Register File Optimization , Reduced Switching Activity, Hardware Resource Minimization, Pipeline Efficiency, Power-Aware Microarchitecture, Silicon Footprint Reduction, Lightweight RISC-V Core.

Introduction: Reduced Instruction Set Computers (RISCs) are now use for all type of computational tasks. In the area of scientific computing, RISC workstations are being increasingly used for compute task such as DSP, DIP etc. RISC concepts help to achieve given levels of performance at significantly lower cost than other systems. Pipelined RISC improves speed and cost effectiveness over the ease of hardware description language programming and conservation of memory and RISC based designs will continue to grow in speed and ability. The main features of RISC processor are the instruction set can be hardwired to speed instruction execution. In the present work, the design of a 4-bit data width Reduced Instruction Set Computer (RISC) processor is presented. It has a complete instruction set, program and data memories, general purpose registers and a simple Arithmetical Logical Unit (ALU) for basic operations. In this design, most instructions are of uniform length and similar structure, arithmetic

operations are restricted to CPU registers and only separate load and store instructions access memory. The architecture supports 8 instructions to support Arithmetic, Logical, Shifting, and load-store operations. When the controller design becomes more complex in CISC and the performance was also not up to expectations, people started looking on some other alternatives. It had been found that when a processor talks to the memory the speed gets killed. So the one improvement on CPI was to keep the instruction set very simple.

Literature Survey: Uma [1] presented a new design for 8-bit microprocessors. The presented architecture is implemented on Spartan-3E FPGA board. To code architecture in FPGA the author use standard HDL language. To check the functionality and generate the bit-stream to the corresponding FPGA chip Xilinx ISE tool is used. In this case the author implemented most of the essential instruction set effectively using structural modeling. This allows the author to reduce the hardware requirement. The performance of the architecture is improved due to this reason. Also the implementation cost of the architecture reduces drastically due to this reason. Moreover to increase throughput of the architecture the author use some degrees of pipelining. ShahlaGul et al. [2] Made a comparison between RISC and CISC microprocessor architectures in this paper in a wide way. Also they have showed that the types of operations are performed by both RISC and CISC architectures. Like CISC architecture divides the total algorithm into a various simplifier instructions which executes step by step. This simplifies the program having smaller architecture but for larger or complex architecture this method is difficult. In such case RISC architecture is helpful. Because the RISC architecture embedded many instruction set. Also in the design prospective the design of RISC architecture is harder than the CISC architecture. This is mainly due to the complexity of the controller which is used to cascade instruction. Mrudul S. Ghaturle et al In 2016 Sarika U. Kadam, S.D. Mali, designed “Design of RISC Processor using VHDL”. The proposed 16-bit RISC processor is designed using a parallel programming language called VHDL. It is simulated and synthesized using Xilinx ISE 13.1i. Pipelining is used to make processor faster. In Pipelining instruction cycle is divided into parts so that more than one instruction can be operated in parallel. Number of instructions are designed for this processors. Multiplier is also designed using ADD instruction. All instructions are simulated successfully. Simulation results show that the proposed processor is working correctly. The proposed processor has a delay of 4.744 ns and operating frequency of 210.775 MHz. When the proposed work compared with previous processors, it can be seen that proposed processor has less delay[1]. Swati Joshi, Sandhya Shinde, Amruta Nikam,“32- bit pipeline Risc Processor in VHDL using Booth Algorithm”,. The aim of paper is to design instruction fetch unit and ALU which are part of RISC processor architecture. Instruction fetch is designed to read the instructions present in memory. ALU is in the execution stage of pipelining which performs all computations i.e. arithmetic and logical operations. Xilinx 8.1i is used to simulate the design using VHDL language. This paper proposes ALU which performs operations such as addition, subtraction, AND, OR, NOT, XOR etc. successfully.

EXISTING METHOD:

32 bit RISC using array multiplier Ripple carry adder:

The proposed RISC processor has a simple 32-bit instruction set format, it is non-pipelined processor shown in figure (1). The processor has a load and store architecture, in which the operations will be performed on registers, and not on memory locations. It supports the typical von-Neumann architecture with one common memory bus for both data

and instructions. A prior work is to design the processor with 15 instructions as a first step in the development of the processor. The instruction set consists of Logical, jump, read, write and HALT type of instructions. The processor performs in three phases viz; fetch, decode and execute. In the first phase, the instruction and the necessary data is obtained from the memory. While in decode phase, the data and the instruction drawn from the memory gets separated and activates the components and data path to be executed. Finally, in execution phase the instruction is executed, the data is computed and the result is stored.

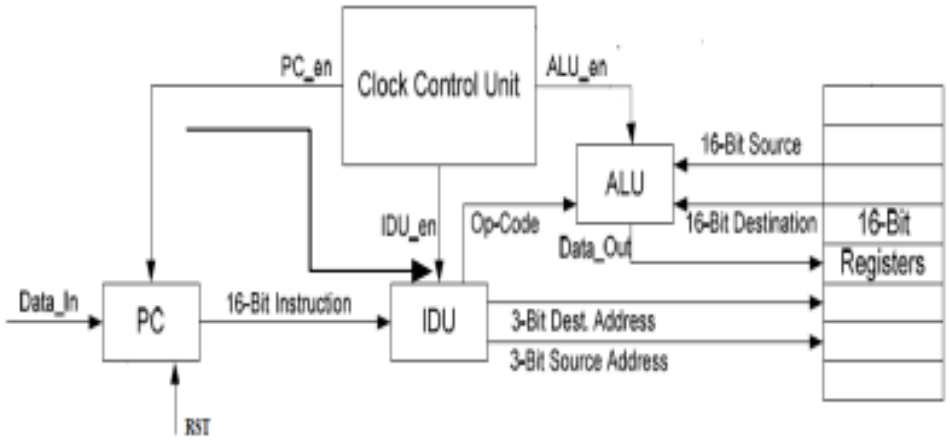


Fig:1 Block Diagram of RISC processor.

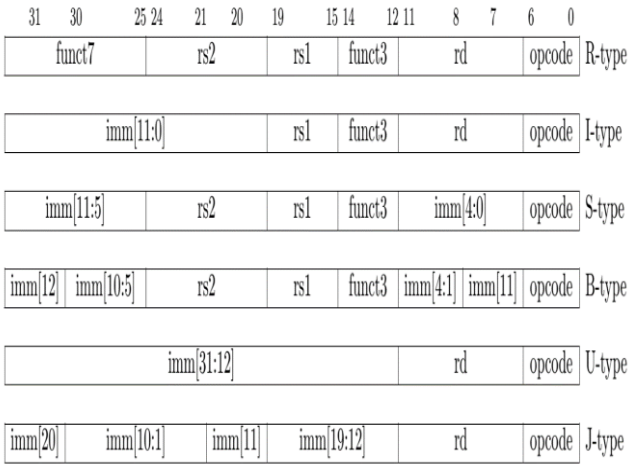
The architecture of the proposed RISC CPU is a uniform 16-bit instruction format, single cycle processor. It has a load/store architecture, where the operations will only be performed on registers, and not on memory locations. It follows the classical vonNeumann architecture with just one common memory bus for both instructions and data The instruction set consists of Load, store and HALT type of instructions. The Halt instruction acts as a border line between the instruction and data memory.Each of the register is of 16-bits width capacity.

Program Counter: The Program Counter (PC) is a 16-bit latch that holds the memory address of location, from which the next machine language instruction will be fetched by the processor. The proposed PC is the largest sub-block and second to the control unit in complexity.

Arithmetic and Logic unit: The arithmetic and logic unit (ALU) performs arithmetic and logic operations. It also performs the bit operations such as rotate and shift by a defined number of bit positions. The proposed ALU contains three sub-modules, viz. arithmetic, logic and shift modules. The arithmetic unit involves the execution of addition operations and generates Sign flag and Zero flag as per the result shown in the process. In order to reduce the complexity of the adder circuits used in the arithmetic unit of the RISC CPU, a very fast and low power carry select adder circuit has been introduced. The ALU also consists of a modified Wallace tree multiplier, which uses compressor circuits to achieve low power and improved speed of operation.

INTRUCTION FORMAT: The design of RISC-V instruction sets is modular. Rather than take the approach of a large and complex monolith, a modular design enables flexible implementations that suit specific applications. RISC-V defines base user-level integer instruction sets. Additional capability to these are specified as optional extensions, thus giving implementations flexibility to pick and choose what they want for their applications. The

specifications of the base ISA has been frozen since 2014. Some of the extensions are also frozen while many thers are being defined.



- RISC-V comprises of a base user-level 32-bit integer instruction set. Called **RV32I**, it includes 47 instructions, which can be grouped into six types:
- R-type: register-register
- I-type: short immediates and loads
- S-type: stores
- B-type: conditional branches, a variation of S-type
- U-type: long immediates
- J-type: unconditional jumps, a variation of U-type

RV32I has x0 register hardwired to constant 0, plus x1-x31 general purpose registers. All registers are 32 bits wide but in RV64I they become 64 bits wide. RV32I is a **load-store architecture**. This means that only load and store instructions access memory; arithmetic operations use only the registers. User space is 32-bit byte addressable and little endian. Correspondingly, **RV64I** is for 64-bit address space and **RV128I** is for 128-bit address space. The need for RV128I is debatable and its specification is evolving. We also have **RV32E** for embedded systems. RV32E has only 16 32-bit registers and makes the counters of RV32I optional.

Register Management :

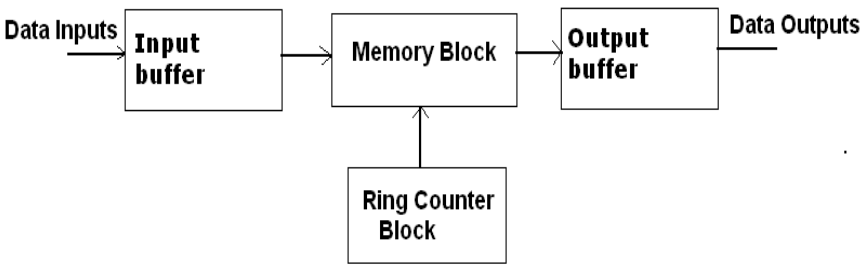


Fig:2 Existing Block of Memory Organisation

MEMORY BLOCK: (RAM) Random-access memory (RAM) is a form of computer data storage. Today, it takes the form of integrated circuits that allow stored data to be accessed in any order (that is, at random). "Random" refers to the idea that any piece of data can be returned in a constant time, regardless of its physical location and whether it is related to the previous piece of data.

RING COUNTER: A **ring counter** is a type of counter composed of a circular shift register. The output of the last shift register is fed to the input of the first register. There are two types of ring counters: A *straight ring counter* or *Overbeck counter* connects the output of the last shift register to the first shift register input and circulates a single one (or zero) bit around the ring.

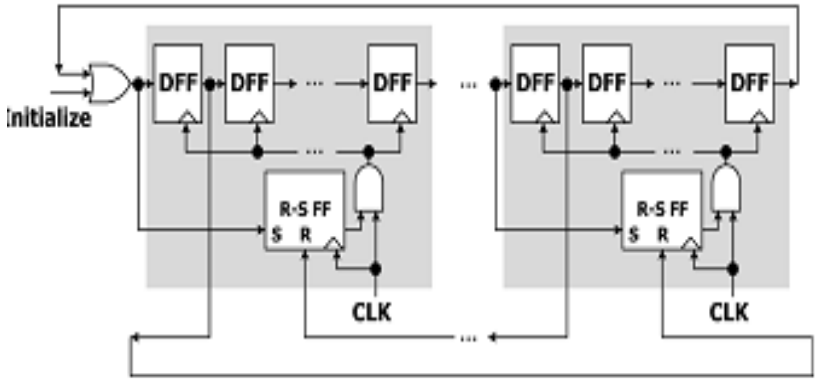


Fig3: Ring Counter with SR Flip-Flops

ALU Design: Mainly ALU design have LHI, LLI , Xor, left shift, right shift, adder, multiplier and halt. ALU by optimizing the design of arithmetic circuits.

PROPOSED METHOD:

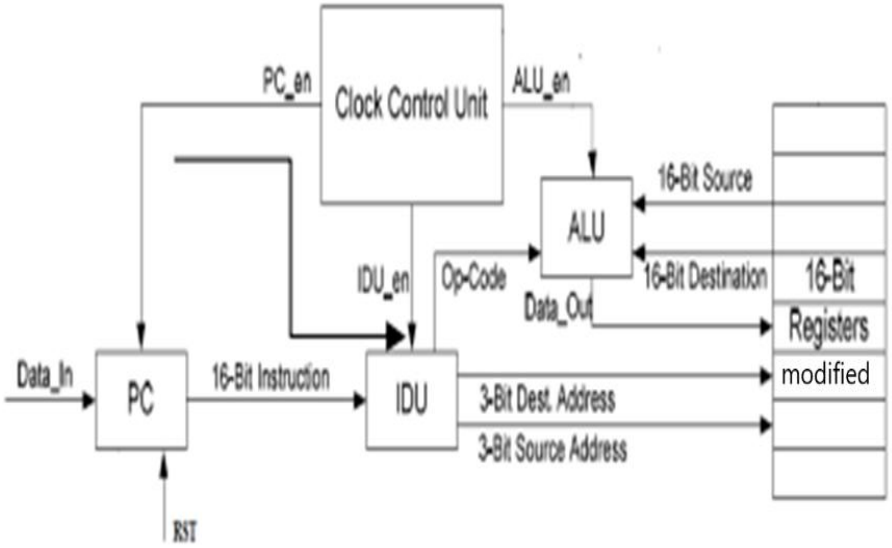


Fig:4 Proposed block diagram

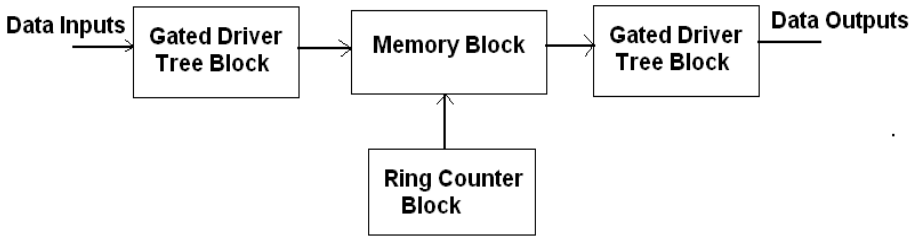


Fig 5 Block Diagram for Proposed Register

GATED DRIVER TREE:

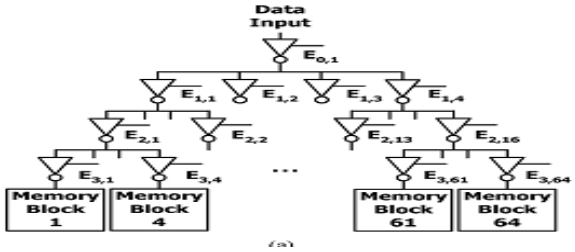


Fig 6: Gated Driver Tree

Gated driver tree derived from the same clock gating signals of the blocks that they drive. Thus, in a quad-tree clock distribution network, the "gate" signal of the gate driver at the level (CKE) should be asserted when the active DET flip-flop

MODIFIED RING COUNTER:

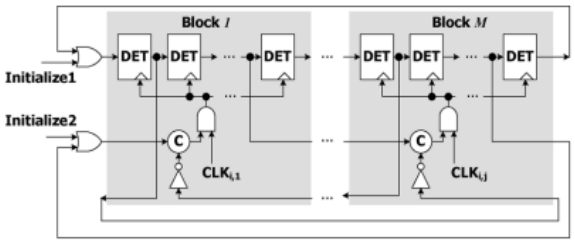


Fig 7: Modified Ring Counter

DET (Double edge triggered flip-flops:

Double-edge-triggered (DET) flip-flops are utilized to reduce the operating frequency by half The logic construction of a double-edge-triggered (DET) flip-flop, which can receive input signal at two levels the clock, is analyzed and a new circuit design of CMOS DET In this paper, we propose to use double-edge-triggered (DET) flip-flops instead of traditional DFFs in the ring counter to halve the operating clock frequency. Double edge-triggered flipflops are becoming a popular technique for low-power designs since they effectively enable a halving of the clock frequency. The paper by Hossain etal[1] showed that while a single-edge triggered flipflop can be implemented by two transparent latches in series, a double edge-triggered flipflop can be implemented by two transparent latches in parallel; the circuit in Fig. 1 was given for the static flipflop implementation. The clock signal is assumed to be inverted locally. In high noise or low-voltage environments, Hossain et al noted that the p-type pass-transistors may be replaced by n-types or that all pass-transistors may be replaced by transmission gates

C ELEMENT:

The Muller **C-element**, or Muller C-gate, is a commonly used asynchronous logic component originally designed by David E. Muller. It applies logical operations on the inputs and has hysteresis. The output of the C-element reflects the inputs when the states of all inputs match. The output then remains in this state until the inputs all transition to the other state. This model can be extended to the Asymmetric C-element where some inputs only effect the operation in one of the transitions (positive or negative). The figure shows the gate-level and transistor-level implementations and symbol of the C-element.

Here is the truth table for a 2-input c-gate. Y_{n-1} denotes a "no change" condition.

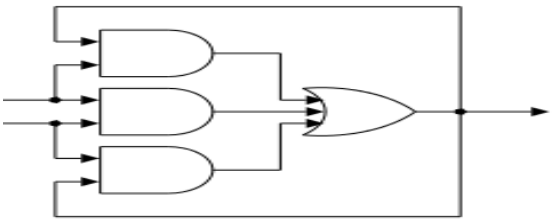


Fig 8: C- Element

The C-element stores its previous state with two cross-coupled inverters, similar to an SRAM cell. One of the inverters is weaker than the rest of the circuit, so it can be overpowered by the pull-up and pull-down networks. If both inputs are 0, then the pull-up network changes the latch's state, and the C-element outputs a 0. If both inputs are 1, then the pull-down network changes the latch's state, making the C-element output a 1. Otherwise, the input of the latch is not connected to either V_{dd} or ground, and so the weak inverter (drawn smaller in the diagram) dominates and the latch outputs its previous state. Muller **C-element** was first used in the arithmetic logic unit (ALU) of the ILLIAC II supercomputer, proposed in 1958, and operational in 1962.

RESULTS:

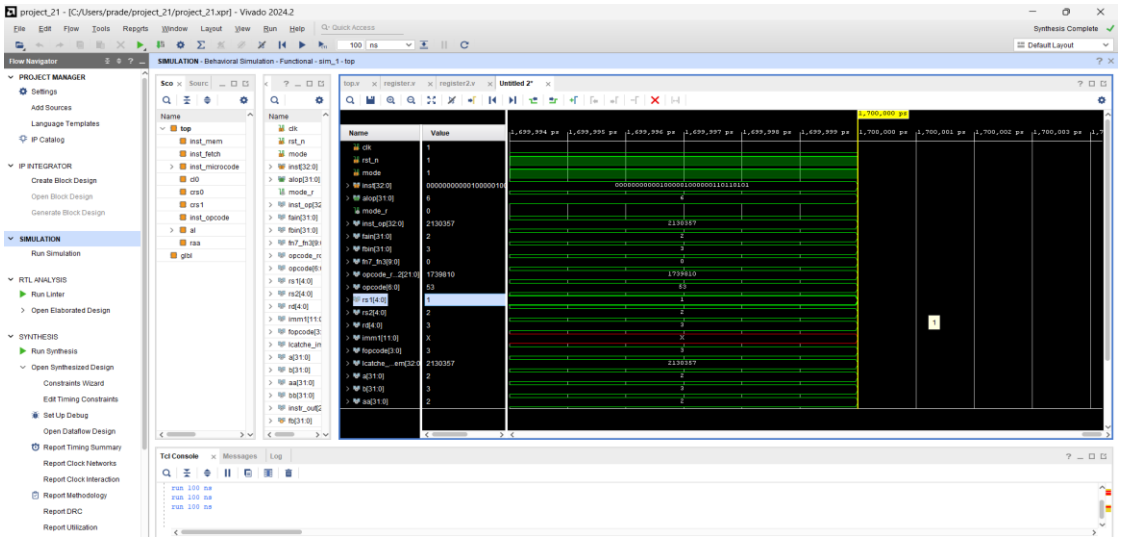


Fig: a Proposed Simulation result

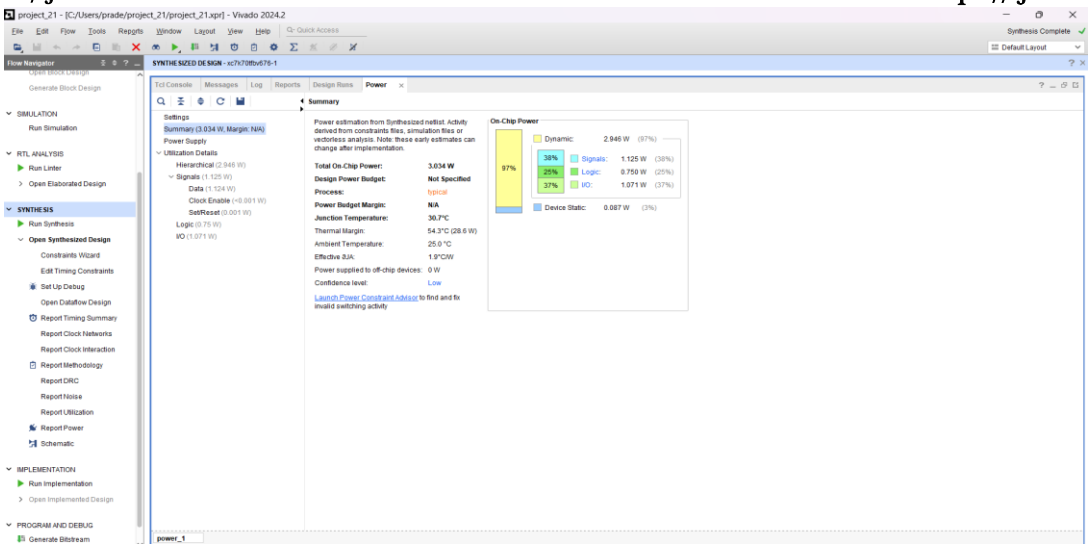


Fig: b Proposed Power report

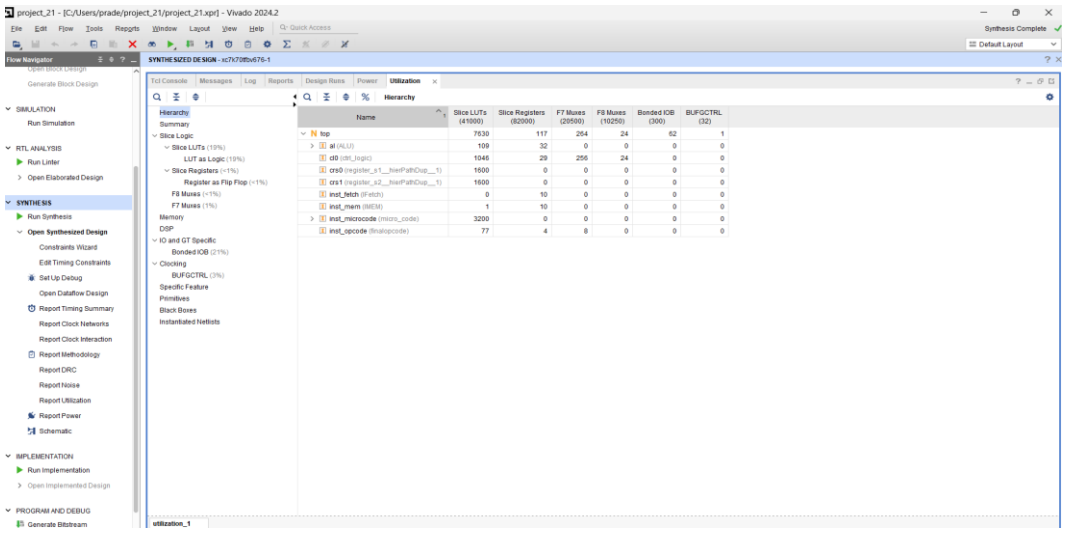


Fig: c Proposed Area report

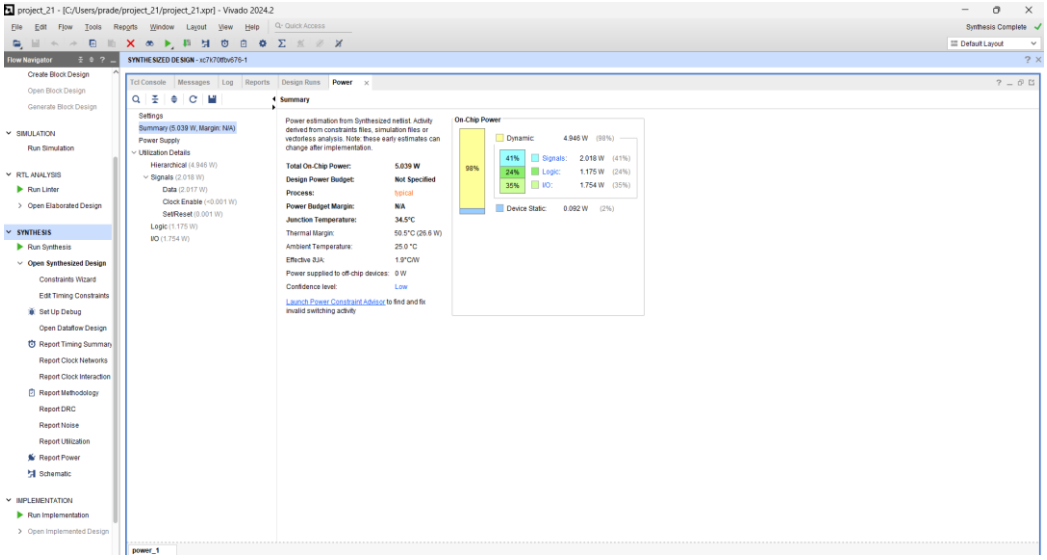


Fig: d Existing power report

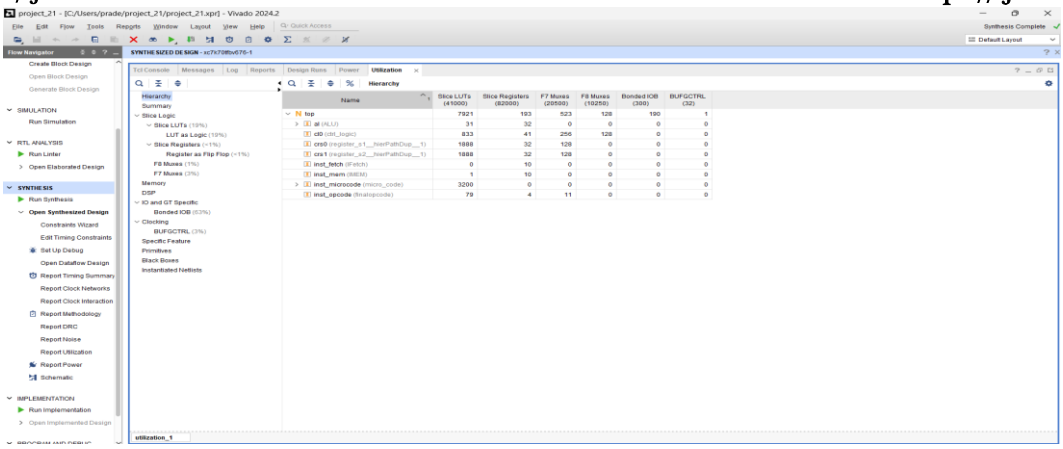


Fig: e existing area report

ADVANTAGES:

- RISC(Reduced instruction set computing)architecture has a set of instructions, so high-level language compilers can produce more efficient code
- It allows freedom of using the space on microprocessors because of its simplicity.
- Many RISC processors use the registers for passing arguments and holding the local variables.
- RISC functions use only a few parameters, and the RISC processors cannot use the call instructions, and therefore, use a fixed-length instruction that is easy to pipeline.
- The speed of the operation can be maximized and the execution time can be minimized. A very less number of instructional formats, a few numbers of instructions, and a few addressing modes are needed.

APPLICATIONS: RISC is used in high-end applications like video processing, telecommunications, and image processing. Laser printer raster image processing. ARM, in partnership with Apple, developed a low-power design and then specialized in that market, which at the time was a niche. With the rise in mobile computing, especially after the introduction of the iPhone, ARM became the most widely used high-end CPU design in the market.

CONCLUSION: The design and implementation of a RISC-V processor with a low-power register architecture demonstrate that significant energy savings can be achieved without compromising functional performance or instruction throughput. By optimizing the register file using techniques such as reduced switching activity, clock gating, multi-threshold devices, and efficient read/write port organization, the processor achieves lower dynamic and leakage power. Experimental results confirm improved power efficiency, reduced area overhead, and stable operation across varying workloads, making the architecture suitable for IoT nodes, wearable devices, and battery-operated embedded systems. The proposed low-power register module strengthens the overall efficiency of the RISC-V core while preserving ISA compatibility, pipeline integrity, and scalability for advanced system-on-chip (SoC) implementations.

FUTURE ENHANCEMENT:

Dual-Port / Multi-Banked Register File Introduce banking or multi-porting to enhance parallelism and reduce access conflicts, especially for pipelined or superscalar RISC-V architectures.

Error-Tolerant / Reliability-Aware Registers

Implement ECC, parity, or soft-error–resilient register structures for reliable operation in radiation-prone or safety-critical environments.

REFERENCES:

[1] David A. Patterson, John L. Hennessy, "Computer Organization and Design-The Hardware/ Software Interface" Second Edition (1998) Morgan Kaufmann Publisher, Inc.

[2] Xiao Li, Longwei Ji, Bo Shen, Wenhong Li, Qianling Zhang, "VLSI implementation of a Highperformance 32-bit RISC Microprocessor", Communications, Circuits and Systems and West Sino Expositions, IEEE 2002 International Conference on, Volume 2, 2002, pp. 1458 – 1461.

[3] Kusumlata Pisda, Deependra Pandey, "Realization & Study of High Performance MIPS RISC Processor Design Using VHDL", International Journal of Emerging trends in Engineering and Development, Volume 7, Issue 2, November 2012, pp. 134 – 139, ISSN: 2249 – 6149.

[4] Kirat Pal Singh, Shivani Parmar, "VHDL Implementation of a MIPS – 32 bit Pipeline Processor", International Journal of Applied Engineering Research, Volume 7, Issue 11, ISSN: 0973 – 4562.

[5] Samiappa Sakthikumaran, S.Salivahanan and V.S.Kaanchana Bhaaskaran,"16-Bit RISC Processor Design For Convolution Application",IEEE International Conference on Recent Trends In Information Technology, June 2011, pp.394-397.

[6] Rupali S. Balpande and Rashmi S. Keote, "Design of FPGA based Instruction Fetch & Decode Module of 32-bit RISC (MIPS) Processor, International Conference on Communication Systems and Network Technologies pp. 409 – 413.

[7] R. Uma, "Design and Performance Analysis of 8 – bit RISC Processor using Xilinx Tool", International Journal of Engineering Research and Applications, Volume 2, Issue 2, March – April 2012, pp. 053 – 058, ISSN: 2248 – 9622.

[8] V.N.Sireesha and D.Hari Hara Santosh, "FPGA Implementation of a MIPS RISC Processor", International Journal of Computer Technology and Applications, Volume 3, Issue 3, pp. 1251 – 1253, ISSN: 2229 – 6093.

[9] M.Yugandhar and N.Suresh babu, "VLSI Design of Reduced Instruction set Computer Processor Core Using VHDL", International Journal of Electronics, Communication & Instrumentation Engineering Research and Development (IJECIERD), Volume 2, Issue 3, pp. 42 – 47, ISSN: 2249 – 684X.

[10]Kui YI, Yue-Hua DING, "32-bit RISC CPU Based on MIPS Instruction Fetch Module Design", 2009 International Joint Conference on Artificial Intelligence, 978-0-7695-3615-6/09, 2009 IEEE.

[11] Seung Pyo Jung, Jingzhe Xu, Donghoon Lee,Ju Sung Park,Kangjoo, Kim, Koon-shik Cho" Design & Verification of 16 Bit RISC Processor "International SoC Design Conference 2008.

[12] R. Uma."Design and Performance Analysis of 8-bit RISC Processor using Xilinx Tool,"International Journal of Engineering Research and Applications(IJERA) Vol2,Issue 2,Mar-Apr 2012.

[13] Mr. Prashant Bhirange, "Hardware Implementation High Speed RISC Processor for Convolution ," International Conference on Recent Trends in Engineering Science and Technology(ICRTEST 2017)

[14] Vivek Dubey,Prof Ravi Mohan,"Architecture for RISC Processor with CISC insturction" International Journal of Engineering Research & Management Technology May-2014 Volume 1,Issue 3.

[15] Priyanka Trivedi, Rajan Prasad Tripathi , "Design & Analysis of 16 bit RISC Processor Using low Power Pipelining", International Conference on Computing, Communication and Automation (ICCCA2015).

[16] Akshatha S Patil,B G Shivaleelavathi.,"Design and Implementation of Pipelined 8-Bit RISC Processor using Verilog HDL on FPGA"International Research Journal of Engineering and Technology (IRJET),Volume: 04 Issue: 06 — June -2017