International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *10 September 2023* ***Revised:*** *20 October 2023* ***Accepted:*** *30 October 2023*

RISC-V Processor for IOT Applications

Rajveer Singh[1*], Manu Bansal[2], Anil Singh[3]

[1*,2,3]Department of Electronics and Communication Engineering, Thapar Institute of Engineering and Technology College, Patiala, Punjab, India

***Corresponding Author:** Rajveer Singh
*Department of Electronics and Communication Engineering, Thapar Institute of Engineering and Technology College, Patiala, Punjab, India

Abstract
RISC-V is a recently introduced instruction-set architecture (ISA) that offers innovative advantages, including low power consumption, affordability, and scalability. Utilizing an open, non-proprietary Instruction Set Architecture (ISA) enables the creation of on-the-fly design of soft error countermeasures at the microarchitecture level. This may significantly enhance the resilience of Application Specific Standard Products (ASSP) and FPGA implementations. This paper offers a quick overview of the RISC-V architecture. This paper presents a plan to create and execute a 32-bit single-cycle RISC-V processor using Verilog HDL in the Vivado software.

1. INTRODUCTION

To comprehend any computer architecture, one must initially acquire knowledge of its programming language.The linguistic units employed in a computer's programming are referred to as instructions. The computer's language is referred to as the instruction set. Computer instructions provide the desired action and the data on which the action will be performed. The operands might be sourced from memory, registers, or instruction. Computer hardware exclusively processes binary code consisting of 1s and 0s. Therefore, instructions are written as binary integers using a syntax known as machine language. Similar to how letters are utilized for encoding human language, computers employ binary digits to encode machine language. RISC-V is not the first nor the only Reduced Instruction Set Computer that is widely used. The RISC-V architecture encodes each instruction as a 32-bit word. RISC-V is described as an open Instruction Set Architecture (ISA). The RISC-V developers aim to offer many CPU architectures that may be used freely. In order to ensure the success and widespread use of RISC-V, it has been specifically engineered to accommodate address spaces of 32-bit, 64-bit, and 128-bit. There are several different versions of the RISC-V architecture.

1. Single-Cycle: Each instruction is executed in a single cycle.
2. Multicycle: Every command is divided into a sequence of lesser stages.
3. Pipelined - Each command is divided into a sequence of stages, and many instructions are executed simultaneously.

The paper utilizes a single-core, in-order, non-busbased, one-cycle architecture fully supporting the RV32I basic integer instruction set.

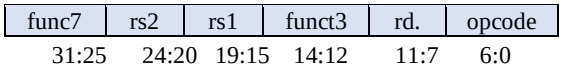
Now, addressing the Instruction Set Architecture (ISA) of the RISC, the RV32I instruction set, which is limited to 32-bit integer operations, consists of just 40 instructions. The RISC-V RV32I architecture consists of 32 registers, each of which is 32 bits in size. The ISA (Instruction Set Architecture) consists of two source operands and one destination operand.

The instructions are categorized into only six distinct formats.

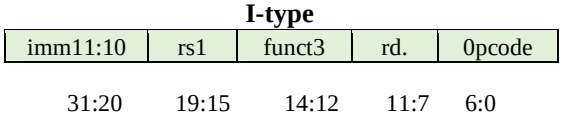
1. R-type (Register/register) instructions only employ registers as both source and destination operands. This instruction category is mainly utilized for performing arithmetic and logic operations that involve the Arithmetic Logic Unit (ALU).
2. I-type (Immediate) instructions have a 12bit constant (or immediate) as one of the two source operands defined within the 32bit instruction word. This constant is considered a 12-bit signed 2's complement integer, and it is always sign extended to provide a 32-bit operand.
3. S-type (Store) instructions are specifically employed for transferring the contents of a register to the data memory for storage purposes.
4. B-type instructions, also known as Branch instructions, are utilized to govern the flow of a program. This operation involves the comparison of two operands that are stored in registers. If the comparison is true, the program will branch to a destination address relative to the Program Counter's current value.
5. J-type (Jump) instructions are employed to initiate subroutine calls.
6. U-type (Upper instantaneous) instructions indicate a register's upper 20 bits immediate value.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *10 September 2023* ***Revised:*** *20 October 2023* ***Accepted:*** *30 October 2023*

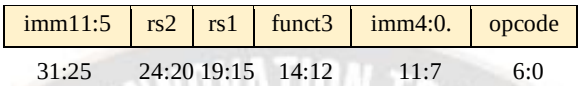
R-type



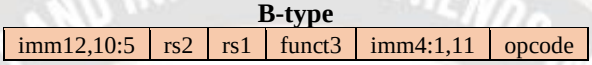
I-type



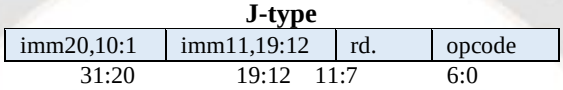
S-type



B-type



J-type



U-type

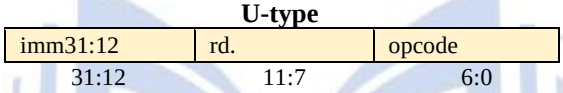


TABLE-1 RISC-V has 32 registers numbered X0-X31.

NAME	USAGE
X0	Constant Value 0
X1	Return address
X2	Stack pointer
X3	Global Pointer
X4	Thread Pointer
X5-X7	Temporaries
X8-X9	Saved
X10-X17	Arguments
X18-X27	Saved
X28-X31	Temporaries

2. LITERATURE SURVEY

The RISC-V instruction set architecture is highly appealing, as it enables the construction of IoT devices with many options and increased performance. The RISC-V instruction set architecture (ISA), initially created to facilitate research and teaching in Computer Architecture, is now poised to become a widely adopted, freely available, and open standard for academic and industry purposes [1]. The RV32I instruction set architecture consists of 6 distinct instruction types. Each format employs a unique coding, with RV32I encompassing a total of 40 discrete instructions. In their study, D.K. Dennis et al. [2] used FPGA technology to construct an RV32I implementation based on the RISC architecture.

A revolutionary single-cycle microprocessor was built with a processing speed of 32MHZ. Currently, devices such as the PlayStation Portable, Nintendo 64, and Linksys WRT54G series of home gateways use the MIPS [3] range of processors. Hitachi also created SuperH (SH), a 32-bit RISC ISA. In addition to being extendable, the design is modular. Using an RV32I as the foundation, Garcia-Ramirez et al. [4] developed a low-power system on chip (SoC) for wearables appliances. Architectures that are based on RISC [2] are the ones that dominate the market for low-power and low-cost embedded system components. The majority of hand-held devices, as well as the majority of Android-based devices, Apple's iPhone and iPad, and other devices employ the ARM architecture. An open-source instruction set architecture (ISA) that is based on RISC principles is the goal of OpenRISC [5]. An architecture that is implemented by OpenRISC has a fixed 32-bit instruction length and either 16 or 32 general purpose registers that are 32 or 64 bits in size. Both the OR1200 and the mor1kx are examples of mainline processor core implementations for OpenRISC technology. Despite the fact that it is not currently being actively developed, the OR1200 is the first original version of the

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *10 September 2023* ***Revised:*** *20 October 2023* ***Accepted:*** *30 October 2023*

processor that is extensively utilized in Verilog HDL. Mor1kx is an innovative implementation that is more refined and has a variety of variations with regard to the presence of a delay slot, the number of pipeline stages, and the closely connected memory. There are a variety of system-on-chips (SoC) that are available for OpenRISC. These SoCs may be utilised to carry out RTL simulations, SystemC simulations, or an FPGA synthesis of an OpenRISC-powered whole system.

3. METHODOLOGY

An adder is utilised to increment the value of the programme counter (PC) by 4 during a single clock cycle. Since every control signal is generated in a single cycle, the control unit is strictly combinatorial.

The 31 general purpose registers, register addressing, read and write logic, and arithmetic logic unit (ALU) make up the register bank and ALU logic module. The ALU in this module specifically handles arithmetic operations between registers and immediate values. The sign extension, shift, and shuffle logic block is responsible for performing register-immediate operations. It ensures that the 32bit instant operand is correctly sign extended and reordered for the ALU. The second operand is obtained from the register bank. The ALU does not participate in the computation of branch and jump target addresses. Instead, separate adders have been included for this purpose. Branch instructions include the Arithmetic Logic Unit (ALU) evaluating the condition and providing the results to the control unit. The control unit then determines if a branch should be taken. The ALU is positioned close to the register bank to minimize the delay along the crucial path. There are five distinct instant formats, specifically referred to as I, S, B, U, and J-type immediate formats.

The decoding process entails correctly rearranging the immediate values by left shifting or sign extension. The encoding is selected to minimize the complexity of implementation across ISA extensions. The sign extension, shift, and shuffle logic module is responsible for instant decoding of this nature. The memory control logic module is designed to handle both word-aligned and misaligned addresses of the data memory, allowing for varied store/load operations.

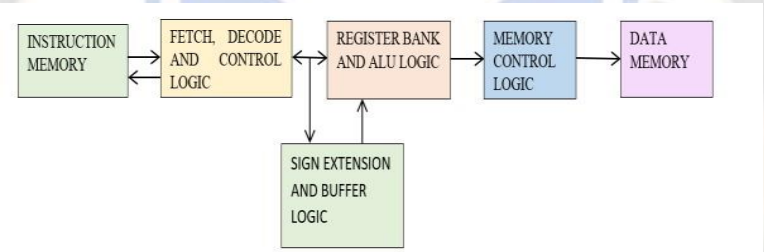


FIG1. BLOCK DIAGRAM OF 32-BIT RISC-V PROCESSOR

3.1 ARCHITECTURAL DESIGN

During the processor design process, we incorporate two units:

1. Datapath Unit - This unit provides instructions on how to execute a given command.
2. Control Unit- The Control Unit receives the current command from the Datapath and generates signals such as multiplexer logic select, register enable, and memory write to control the functioning of the Datapath.

Figure 2 illustrates the execution of the data and control pathways in the design. The opcode (0 to 6 bits), funct3 (12 to 14 bits), and the instruction's 30th bit are the three inputs that the control unit (CU) receives. The CU sends control signals to modules based on inputs, ensuring appropriate instruction execution. The CU provides a total of 12 control lines. ALU Ctrl (4-bit) instructs the ALU on which operation to do on the operands. When Reg WR (1bit) is "high," it enables the register file to write data. When Reg WR is "low," it sets the register file to read-only mode. Likewise, other modules are regulated. Instruction and data memory have a maximum capacity of 4GB each. The register file consists of 32 registers, each with a width of 32 bits. The program counter, particularly the program counter plus four, may be accessed by using the JAL instruction and setting the offset to zero. The suggested design requires some redundancy due to its single cycle to achieve a balance between component reuse and critical path length. Therefore, the computation of the branch target is achieved by employing an extra adder, and each of the five I-type instructions is equipped with specific modules for sign extension and reordering. The program counter is equipped with a specific adder that is used to increment its value.

The program counter determines the memory address of the instruction that is to be executed. The necessary inputs from the instruction are sent to the control unit, register file, and sign extension, shift, and shuffle logic modules. The control unit (CU) manages each module by supplying the necessary signals to each module. The instruction specifies the location of the registers (A1, A2, and A3) in the register file.

The Arithmetic Logic Unit (ALU) executes the operation that is unique to the instruction it receives. This operation is performed on the inputs of the ALU, which are determined by the multiplexers. The control of these multiplexers is managed by the Control Unit (CU).

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *10 September 2023* ***Revised:*** *20 October 2023* ***Accepted:*** *30 October 2023*

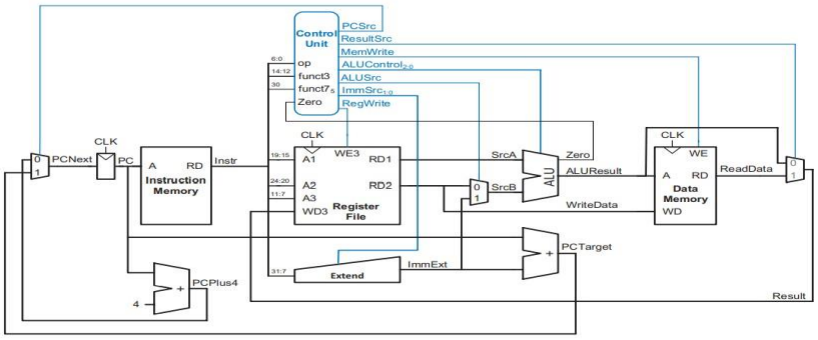


FIG2. ELABORATED DATAPATH DESIGN OF 32-BIT RISC-V PROCESSOR [6]

The ALU output is determined by the control bits, and it is then utilized to either write in the register file or specify the location in the data memory (in load or store instructions) or program counter (in jump instructions). The multiplexer controls the address specification of the program counter.

4. RESULT

The simulation of Verilog code is executed on Vivado Software. The waveform of register file can be shown in FIG3.

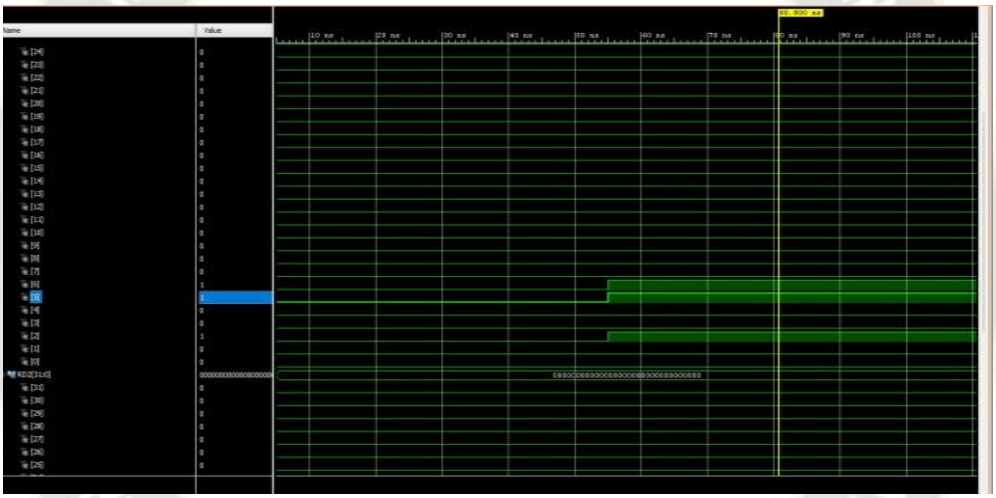


FIG3. REGISTER FILE WAVEFORM

According to the values that are displayed in the waveform, it is clear that the RD1 and RD2 destination registers are storing the values in accordance with the ALU operation that is being utilised.

Now the FIG4 represents the waveform of instruction fetch.

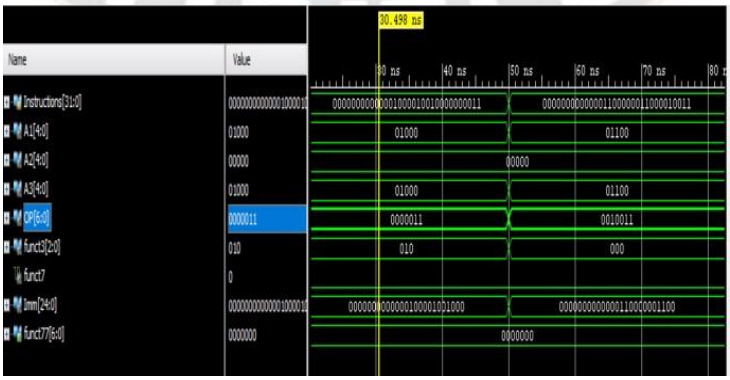


FIG4. INSTRUCTION FETCH WAVEFORM

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 11
Article Received: *10 September 2023* ***Revised:*** *20 October 2023* ***Accepted:*** *30 October 2023*

In this way all the other modules and processor module has been implemented

Now the 32-bit RISC-V processor output is shown in FIG5.

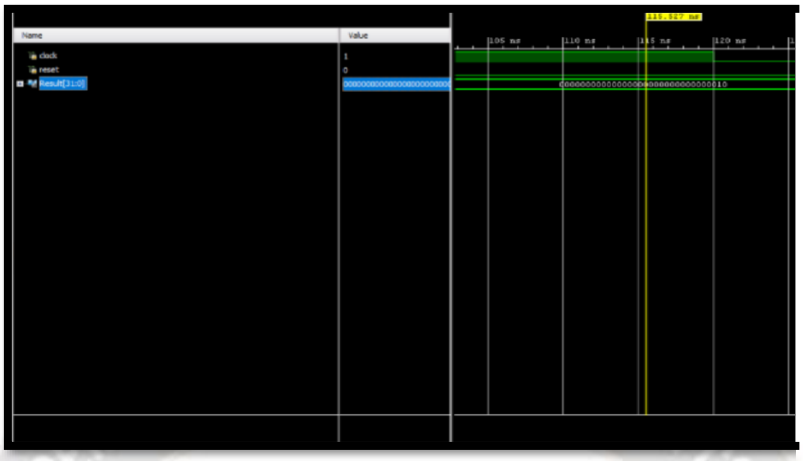


FIG5. 32 BIT PROCESSOR OUTPUT

5. CONCLUSION

This paper briefly introduces the RISC-V architecture and demonstrates the implementation and design of a 32-bit single-cycle RISC-V architecture using Verilog HDL. Given the benefits of the expanding RISC-V community and the availability of tools and software for this new instruction set, the current processor design enables future implementations for specific and general applications in IoT and real-life embedded systems. Future objectives encompass the integration of an I/O bus, the addition of a multi-level caching mechanism, the incorporation of interrupts, and the amalgamation with a floating-point co-processor. To validate the ultimate outcome, we shall employ FPGA technology for the purpose of verification.

REFERENCES

1. Waterman, A., Lee, Y., Patterson, D., Asanovic, K., level Isa, V. I. U., Waterman, A., ... & Patterson, D. (2014). The RISC-V instruction set manual. *Volume I: User-Level ISA', version*, *2*.
2. Dennis, D. K., Priyam, A., Virk, S. S., Agrawal, S., Sharma, T., Mondal, A., & Ray, K. C. (2017, December). Single cycle RISC-V micro architecture processor and its FPGA prototype. In *2017 7th International Symposium on Embedded Computing and System Design (ISED)* (pp. 1-5). IEEE.
3. Eljhani, M. M., & Kepuska, V. Z. (2021, May). Reduced Instruction Set Computer Design on FPGA. In *2021 IEEE 1st International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering MI-STA* (pp. 316-321). IEEE.
4. Garcia-Ramirez, R., Chacón-Rodríguez, A., Castro-Gonzalez, R., Arnaud, A., Miguez, M., Gak, J., ... & Rimolo-Donadio, R. (2020, February). Siwa: a risc-v rv32i based micro-controller for implantable medical applications. In *2020 IEEE 11th Latin American Symposium on Circuits & Systems (LASCAS)* (pp. 1-4). IEEE.
5. Tandon, J. (2011). The openrisc processor: open hardware and linux. *Linux Journal*, *2011*(212), 6.
6. Harris, S., & Harris, D. (2021). *Digital Design and Computer Architecture, RISC-V Edition*. Morgan Kaufmann.