Design and Implementation of RISC-V Processor ALU using Multiplexers and LUT.

Prof. Srinivasan K

Assistant Professor
Electronics and Communication Engineering
Sri Sairam Engineering College line 4: Chennai, India
srinivasan.ece@sairam.edu.in

Niranjana A
Electronics and Communication Engineering
Sri Sairam Engineering College line 4: Chennai, India
sec19ec075@sairamtap.edu.in

Nandhitha S T
Electronics and Communication Engineering
Sri Sairam Engineering College line 4: Chennai, India
sec19ec140@sairamtap.edu.in

Agnes Abina S
Electronics and Communication Engineering
Sri Sairam Engineering College line 4: Chennai, India
sec19ec034@sairamtap.edu.in

***Abstract*— RISC-V (Reduced Instruction Set Computing) is a high-performance Instruction Set Architecture (ISA) capable of performing CISC level functions. ALUs form the fundamental execution unit of any processor. The conventional ALU's are designed using the combinational circuits like the adders, subtractors, multipliers, decoders etc Also, they are designed in such a way that it calculates the result each time for the selected operation and then assigns it to the output signal. This leads to more delay (because the operands are calculated only at that instant) and the increased computational complexity. The proposed multiplexer based ALU focuses on power optimization and reduces the delay in the execution of the processor which results in improved performance of the processor. Multiplexers are used instead of other combinational circuits where it uses the registers to store the intermediate results and for selecting the corresponding output signal based on the selected operation on the same data. Hence the ALU designed using these multiplexers will have less delay when compared to the conventional ALU's.**

Keywords—RISC (Reduced Instruction Set Computing) V, ALU, Multiplexer, LUT, Xilinx Vivado, Verilog.

I. INTRODUCTION

In today's world , processors are the brain of any digital systems right from phones , laptops to PCs[2]. It has become an integral part of any simple to complex working digital circuits. Hence the demand for efficient , fast and more sophisticated processors is indubitably increasing. Building such a chip to support more complex systems involves careful and detailed analysis right from the specification of the design, building the architecture, efficient coding of the specified design and good verification and validation etc. Also, several trade-off s are also taken into consideration while designing such a chip. Trade-off between area, power and delay plays a crucial role.

A. RISC V

Modern systems designed nowadays run on the concept of parallel processing for faster data transmissions. On the other hand, parallel processing tends to consume more power. This problem is often faced by the Silicon Engineers.

With the advancement of nanometre technology , RISC V is an extension to RISC , which is an open source ISA(Instruction Set Architecture) .RISC V has a standard 49 instructions unlike other RISC architectures like ARM (Advanced RISC Machine) which has more than 200 possible machine instructions. This architecture mainly includes Branch Prediction, Data Cache, Debug Unit, Instruction Cache, & optional Multiplier or Divider Units.

Several standardised extensions comes with the implementation of core ISA of RISC V based on the application.

Name	Description	Version	Status	Integer Count
RV32I	Base Integer Instruction Set - 32-bit	2.1	Frozen	49
RV32E	Base Integer Instruction Set (embedded) - 32-bit, 16 registers	1.9	Open	Same as RV32I
RV64I	Base Integer Instruction Set - 64-bit	2.0	Frozen	14
RV128	Base Integer Instruction Set - 128-bit	1.7	Open	14

Fig 1 RISC V ISA Extensions

B. RISC V Execution Unit

The parallel processing is achieved through a five stage pipelining in the existing design of RISC V architecture. The pipelining stages are as follows : Instruction fetch , Instruction Decode , Execute , Memory , Writeback.[5]

The execution unit of the pipelining stage in a processor executes the specified instruction on the operands. The instruction to be executed is decoded using an opcode in the Instruction decode stage. In this proposed design, the ALU (Arithmetic Logic Unit) of the execution unit is modified and designed using Multiplexers and LUTs (Look up tables) instead of other combinational circuits like adders, subtractors, multipliers, decoders etc...[1]. which often tend to consume more power than the multiplexer.

II. .PROPOSED NOVEL MUX

A novel MUX (Multiplexer) - ALU has better performance because it utilizes the registers to store the intermediate results and for selecting the corresponding output signal based on the selected operation on the same data[3]. Hence the ALU designed using these multiplexers will have less delay when compared to the conventional ALU's.

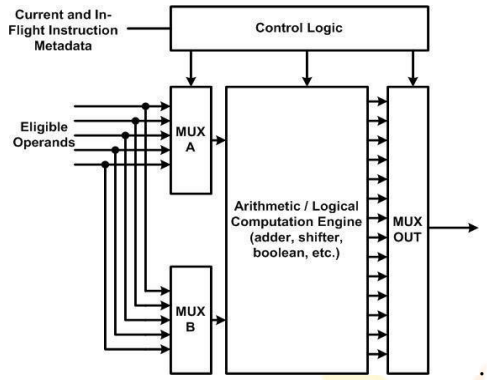


Fig 2 General Mux incorporated with ALU unit.

A. *Operation of novel Mux with LUT:*

LUTs: The look up table is used the store the array of data. LUTs are tables that map a set of input values to a corresponding set of output values. An addition operation can be implemented using a 2:1 multiplexer to select the carry-in value and a LUT to generate the sum.

A+B	00	01	10	11
00	00	01	01	11
01	01	10	11	00
10	01	11	00	01
11	11	00	01	10

A-B	00	01	10	11
00	0	1	10	11
01	-1	0	1	10
10	-10	-1	0	1
11	-11	-10	-1	0

A*B	00	01	10	11
00	0	0	0	0
01	0	1	01	011
10	0	10	100	0110
11	0	011	0110	01001

Fig 3 LUT internal arrangement and operation.

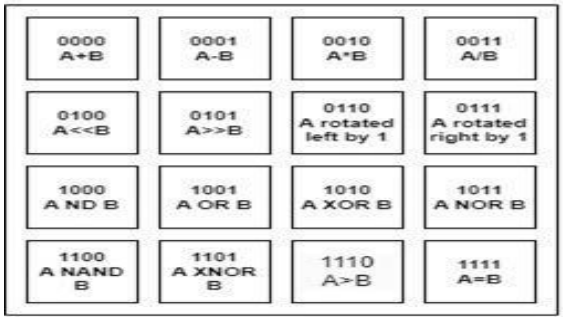


Fig 4 Operation of registers

B. *Step by Step implementation of the proposedmodifications*

The design and implementation of a RISC-V processor ALU using multiplexers and LUT's involves the following steps:

- a.Define the input and output signals for the ALU. The input signals may include two operands, an operation code, and other control signals. The output signals may include the result of the operation, the condition flags, and other control signals.
- b. Implement the ALU functions using multiplexers and LUTs. For example, an addition operation can be implemented using a 2:1 multiplexer to select the carry-in value and a LUT to generate the sum.
- c. Combine the individual ALU functions into a single ALU unit. This can be done by connecting the output of one function to the input of another function using multiplexers and logic gates.
- d. Verify the functionality of the ALU by simulating its operation using a digital logic simulator. This will help to identify any design errors or timing issues.
- e. Integrate the ALU into the RISC-V processor design. This involves connecting the ALU to the instruction decoder, register file, and other components of the processor.
- f. Verify the overall functionality of the RISC-V processor by testing it with a set of instructions and comparing the output to the expected result.

Overall, the design and implementation of a RISC-V processor ALU using multiplexers and LUTs involves a systematic and iterative approach to digital logic design.

C. *Proposed Logic*

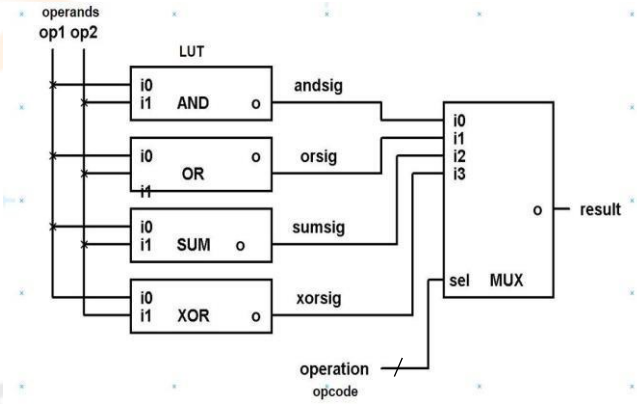
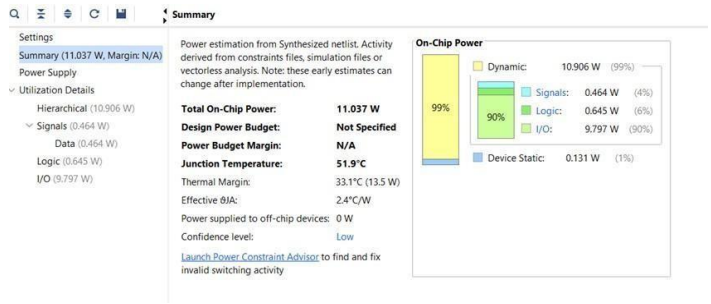


Fig 5. 4:1 Multiplexer with LUT connection

III. POWER ANALYSIS:

The power consumed from the existing model is more than the proposed one.

Power Analysis and Resource Utilization of Existing and

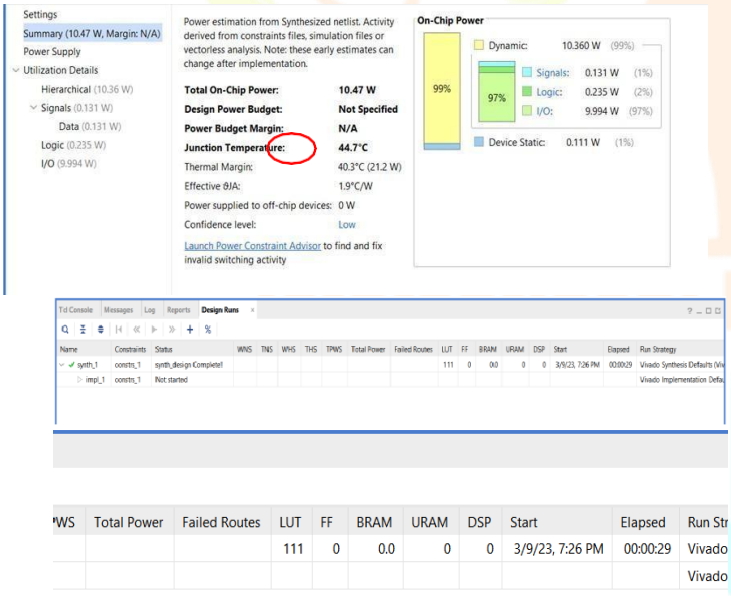


Proposed ALU:

Fig 6. Power analysis summary of existing ALU (execution unit).

Fig 7. Power analysis summary of existing ALU (execution unit).

A. Count of LUT- Analysis



WS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Str
			111	0	0.0	0	0	3/9/23, 7:26 PM	00:00:29	Vivado
										Vivado

Fig 8. Number of LUTs in the existing ALU unit

TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Stra
			49	0	0.0	0	0	3/14/23, 11:05 PM	00:00:24	Vivado S
NA	10.766	0	49	0	0.0	0	0	3/14/23, 11:06 PM	00:00:51	Vivado I

Fig 9. Number of LUTs in the existing ALU unit

IV. SIMULATION RESULTS

The design of the ALU of the execution unit using lookup tables is designed using Verilog HDL and has been simulated in Xilinx Vivado hardware simulation tool (*v2020.2*).

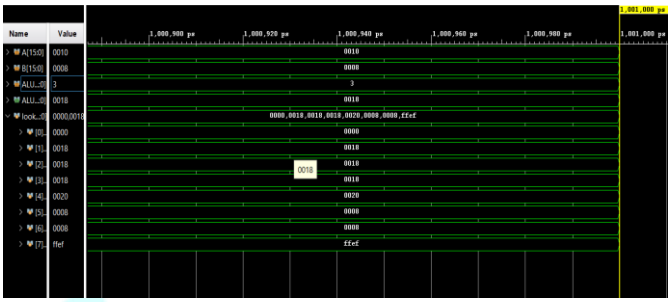


Fig 10. Behavioural simulation result

V. CONCLUSION.

Based on the design and implementation of the RISC-V Processor ALU with Multiplexers and LUT, the project has achieved the objective of developing a high-performance ALU for RISC-V processors. The project utilized a combinational logic design approach that includes multiplexers and LUTs to reduce area and power consumption. Thus, the proposed ALU can perform arithmetic and logical operations efficiently with low latency and high throughput.

The project's ALU design also demonstrates a considerable improvement with respect to area (Number of Lookup tables realized) and power efficiency compared to other existing solutions. Although the project was successful in achieving its objectives, there are still opportunities for future improvements. One of these opportunities is to optimize the design further by doing gate level modifications and dynamic logic. In conclusion, the project has developed a highly efficient RISC-V Processor ALU with Multiplexers and LUT that can be implemented in RISC-V processors.

This project's contribution can potentially enhance the performance and reduce the power consumption of future RISC-V processors.

TABLE 1

Parameter	**ALU unit**		
	Existing	***Proposed***	***Power_ diff***
Power_Consumed	11.04W	10.45W	8% ⬇

.

REFERENCES

[1] Ajay Joshi, Siew Lam, Yee Chan (2011) "Design of an Improved Multiplier Unit for an Experimental RISC CPU." - International Conference on Electric and Electronics (EEIC 2011) in Nanchang, China on June 20-22, 2011, Volume 3.

[2] Shafqat Khan, Muhammad Rashid (2011) "A high-performance processor architecture for multimedia applications".

[3] Sakshi Rajput (2013), "Implementation of Power Efficient Novel Multiplexer Based Arithmetic Logic Unit." - International Journal of Engineering Research & Technology (IJERT).

[4] M.Johns, T.Kazmierski (2020), "A Minimal RISC-V Vector Processor for Embedded Systems" - IEEE Forum for Specification and Design Languages (FDL).

[5] C Venkatesan (2019), "Design of a 16-Bit Harvard Structure RISC Processor in Cadence 45nm Technology." - IEEE International Conference on Advanced Computing and Communications Systems(ICACCS)

[6] Manit Kantawala(2018), "Design and implementation of 8 bit and 16-bit ALU using verilog language." - International Journal of Engineering Applied Sciences and Technology, 2018 Vol. 3, Issue 2, ISSN No. 2455-2143.