International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-11 Issue-6, March 2023

Implementation of LVCMOS based 4 Bit FPGA Based ALU on SP 701 Board for New Digital Age Technologies

Chandrashekhar Patel, Abhay Saxena, Anita Rawat, Omprakash Nautiyal

***Abstract: Objectives:** The 4-bit ALU of a RISC processor is designed as shown by the researcher in this paper. The 4-bit ALU used in this work can perform 2^4 = 16 various arithmetic and logical operations, including addition, subtraction, multiplication, and division as well as logical AND, OR, NAND, NOR, NOT, XOR, XNOR, INCREMENT, DECREMENT, ROTATE LEFT, and ROTATE RIGHT. **Methods:** The author used the Vivado simulation tools with the Verilog HDL language to build the FPGA-based ALU, and the SP701 Spartan FPGA board was used to implement the entire design. It has been implemented to use energy-efficient IO standard approaches. **Findings:** By calculating the overall power usage at the pre- and post-levels, this research has developed a new method for building energy-efficient FPGA-based ALUs. Author utilized Vivado simulation tool for this investigation. The SP701 FPGA board has also been used to implement this idea. **Novelty:** The Internet of Things and other emerging digital era technologies will undoubtedly benefit from this research work, and its energy efficient design will support environmental initiatives.*

***Keywords:** SP701, FPGA, VIVADO, RISC, LVCMOS, Verilog*

I. INTRODUCTION

In today era the usage of microprocessors is on demand so, it would be always challenge to design high speed and high-performance arithmetic logic unit (ALU). Basic arithmetic operations are accomplished by standalone hardware in all recent processors. With response to rapid arithmetic hardware, processors have now on memory (cache) that allows them to achieve major speed improvements by minimizing data access delay from primary memory. The primary goal of this project is to inscribe the blueprint of functional blocks. In the design of such a digital system, an Arithmetic Logical Unit is a fundamental subsystem. It is a combinational logic unit and a fundamental component of a microprocessor that conducts arithmetic and logic operations. ALUs are now becoming smaller and more advanced in order to allow the developer for the development of even much more powerful but compact systems and processors. All types of microprocessors are used in various applications for arithmetic computation like Add, Multiply, Divide and subtract as well as logical operations such as XOR, NAND, AND, OR, NOT.

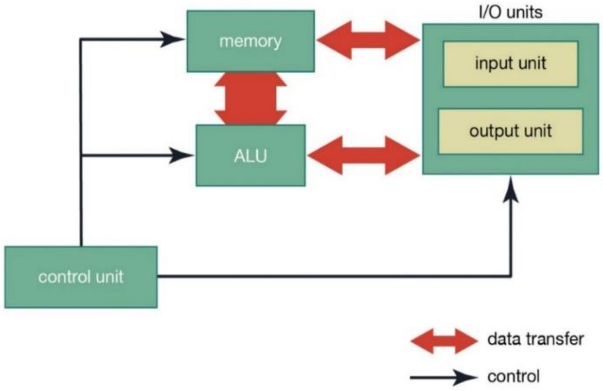


Figure 1: Working of ALU

The ALU is a central part of CPU (Central Processing Unit). ALUs in today's CPUs and graphics processing units (GPUs) are extremely powerful and complicated, with a single unit containing multiple ALUs shown in Figure 1. VHDL is used to design the ALU in this research work. VHDL is also called hardware description language which is used to design several different digital components that is used in electrical circuit design. The ALU is designed using the Mixed Modeling Style. For HDL design, synthesis and analysis of an ALU here Xilinx ISE (Integrated Software Environment) has been used. It allows developers to integrate their prototypes, execute timing evaluation, and evaluate RTL illustrations.

The ALU is implemented in parallel through using different functional unit that executes specific functions like addition, subtraction, and so on. Data operands are given to the respective unit that corresponds to particular operation to generate a desire result and transmits it to the ALU's output lines. Select lines are used to choose among different operations. A modular design is utilized to fabricate the ALU, which comprises of smaller, more accessible blocks, which could be recycled.

Manuscript received on 24 February 2023 | Revised Manuscript received on 14 March 2023 | Manuscript Accepted on 15 March 2023 | Manuscript published on 30 March 2023.

*Correspondence Author(s)

Dr. Chandrashekhar Patel*, Department of Computer Science, Dev Sanskriti Vishwavidyalaya, Haridwar (Uttarakhand), India. Email: shekharrockin1988@gmail.com, ORCID ID: https://orcid.org/0000-0003-0379-673X

Prof. Abhay Saxena, Dean, School of Technology, Management & Communication, Dev Sanskriti Vishwaviidyalya, Hardwar (Uttarakhand), India. E-mail: abhaysaxena2009@gmail.com, ORCID ID: https://orcid.org/0000-0001-7685-7607

Prof. (Dr.) Anita Rawat, Director, Uttarakhand Science Education and Research Centre (USERC), Dehradun (Uttarakhand), India. E-mail: u.serc@rediffmail.com

Prof. Omprakash Nautiyal, Uttarakhand Science Education & Research Centre (USERC) Dehradun (Uttarakhand), India. E-mail: nautiyal_omprakash@yahoo.co.in

II. LITERATURE REVIEW

[1] High Performance FIFO Design for Processor through Voltage Scaling Technique [2]HSTL IO Standards Based Processor Specific Green Counter [3]Capacitance Scaling Based Low Power Comparator Design on 28nm FPGA [4] SSTL Based Energy Efficient FIFO Design for High-performance Processor of Portable Devices [5] Energy Efficient CRC Design for Processor of Workstation, and Server using LVCMOS [6] Cyclic redundancy check is a basic requirement for speed-optimized computation [7] Memory controller logic includes a CRC component configured to enable the CRC processes on the individual ranks [8] Speed-optimized computation of cyclic redundancy check codes [9] Cyclic Redundancy Check (CRC) False Detection Reduction in Communication Systems [10] Researcher developed an instruction set architecture for programmable cyclic redundancy check (CRC) computation [11] An Approach for Testing Programmable/Configurable Field Programmable Gate Arrays [12] SSTL Based Energy Efficient FIFO Design for High Performance Processor of Portable Devices.

III. TOOLS REQUIRED

3.1. SP701 Evaluation Board

Most designs are becoming increasingly concerned about power. Lowering power increases cost and reliability, as well as facilitating higher performance, in addition to fulfilling supply and thermal restrictions. The Xilinx® 7 Series FPGAs meet the design goals for an increasingly wide range of applications with breakthrough reductions in power consumption. The XC7S100FGGA676 device, a member of the Xilinx® 7 series FPGA family, used in the SP701 evaluation board shown in Figure 2.

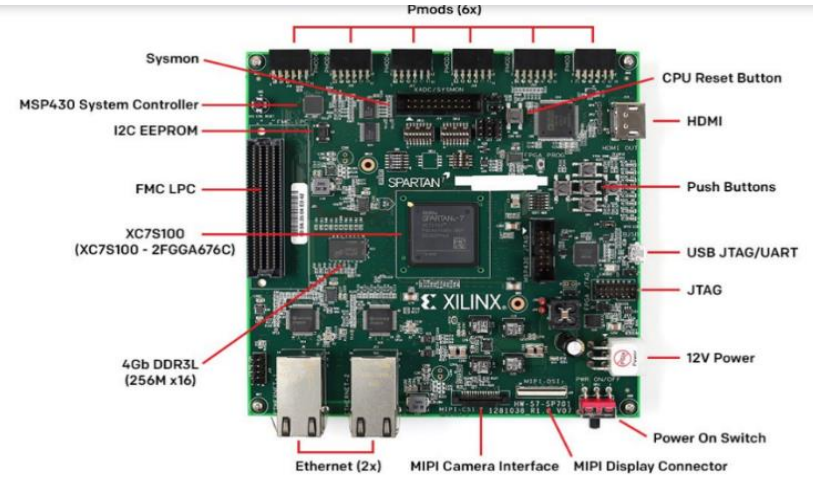


Figure 2: SP701 Evaluation Board.

3.2. Vivado

Vivado is software that is produced by Xilinx for the analysis and synthesis of hardware description language designs. It is popularly used to design, program, and debug Xilinx's line of FPGA. It provides a complex integrated development environment (IDE) tool for the process of FPGA design and its implementation. If we talk about its origin, it was introduced to the world in April 2012 with software to hardware level tools that can be used for various purposed. It is based on the TCL scripting language. The eye-catching feature of Vivado is a high-level synthesis that includes a tool chain that helps to convert C code into programmable logic. It consists of many features that prove rich and excellent in High-Level Synthesis. It can be used for different purposes like designing some prototypes, hardware-software co-simulation, attaching peripherals to the board. Figure 3. shows the Vivado home screen.

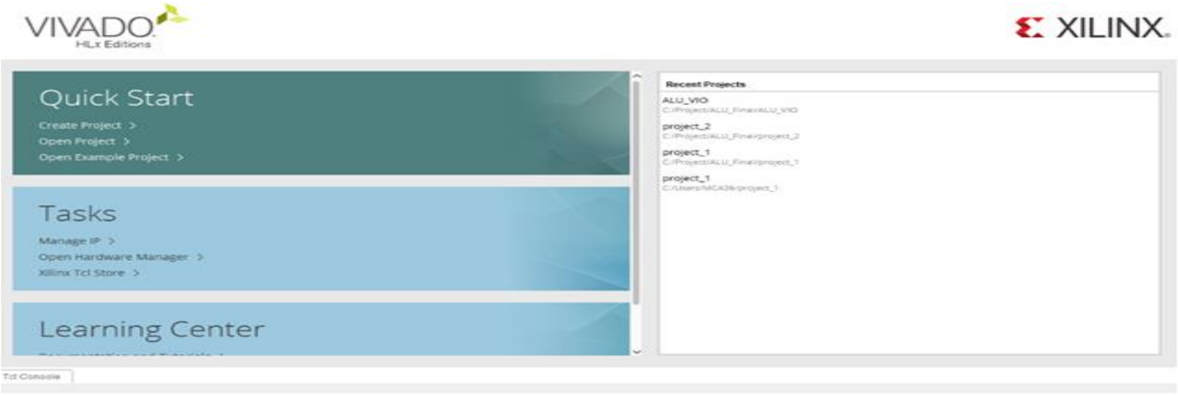


Figure 3: Vivado IDE home Screen

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

IV. DESIGN METHODOLOGY

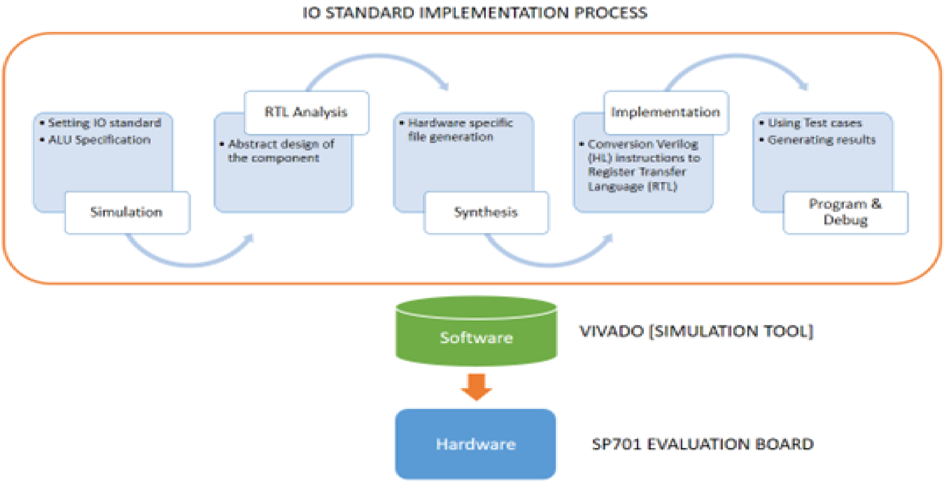
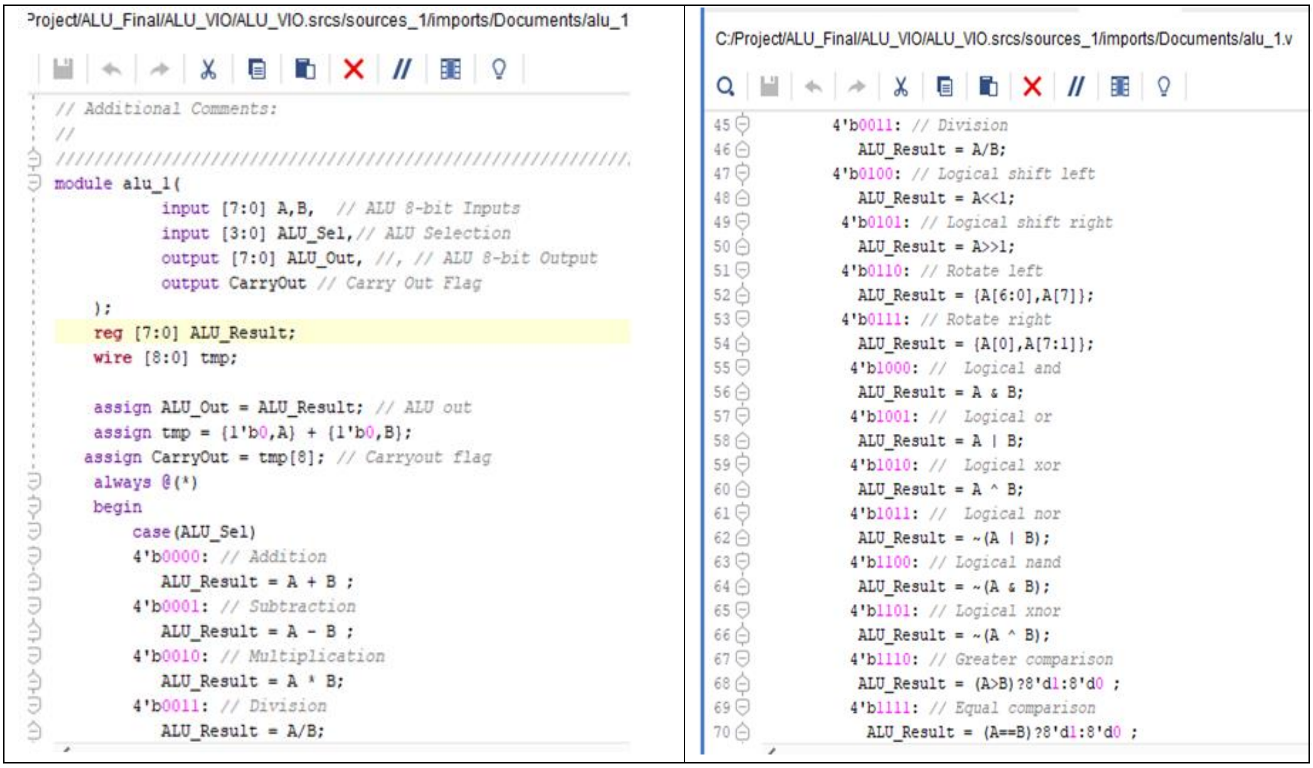


Figure 4: Design Methodology of FPGA based ALU

In Figure 4. the author has shown the research methodology of the FPGA based ALU and below he is described every process into details.

4.1 Simulation

One of the most powerful analysis tools accessible to those in charge of the design and/or operation of complex processes or systems is simulation. The Xilinx High Level Synthesis Vivado Tool is used to develop and simulate the ALU in Verilog language. Figure 5. shown the Verilog code of ALU.


Project/ALU_Final/ALU_VIO/ALU_VIO.srcs/sources_1/imports/Documents/alu_1
C:/Project/ALU_Final/ALU_VIO/ALU_VIO.srcs/sources_1/imports/Documents/alu_1.v

```
// Additional Comments:
//
////////////////////////////////////////////////////////////////////.
module alu_1(
            input [7:0] A,B,  // ALU 8-bit Inputs
            input [3:0] ALU_Sel,// ALU Selection
            output [7:0] ALU_Out, //, // ALU 8-bit Output
            output CarryOut // Carry Out Flag
    );
    reg [7:0] ALU_Result;
    wire [8:0] tmp;

    assign ALU_Out = ALU_Result; // ALU out
    assign tmp = {1'b0,A} + {1'b0,B};
   assign CarryOut = tmp[8]; // Carryout flag
    always @(*)
    begin
        case(ALU_Sel)
        4'b0000: // Addition
           ALU_Result = A + B ;
        4'b0001: // Subtraction
           ALU_Result = A - B ;
        4'b0010: // Multiplication
           ALU_Result = A * B;
        4'b0011: // Division
           ALU_Result = A/B;
```

```
        4'b0011: // Division
           ALU_Result = A/B;
        4'b0100: // Logical shift left
           ALU_Result = A<<1;
         4'b0101: // Logical shift right
           ALU_Result = A>>1;
         4'b0110: // Rotate left
           ALU_Result = {A[6:0],A[7]};
         4'b0111: // Rotate right
           ALU_Result = {A[0],A[7:1]};
          4'b1000: //  Logical and
           ALU_Result = A & B;
          4'b1001: //  Logical or
           ALU_Result = A | B;
          4'b1010: //  Logical xor
           ALU_Result = A ^ B;
          4'b1011: //  Logical nor
           ALU_Result = ~(A | B);
          4'b1100: // Logical nand
           ALU_Result = ~(A & B);
          4'b1101: // Logical xnor
           ALU_Result = ~(A ^ B);
          4'b1110: // Greater comparison
           ALU_Result = (A>B)?8'd1:8'd0 ;
          4'b1111: // Equal comparison
            ALU_Result = (A==B)?8'd1:8'd0 ;
```

Figure 5: Verilog code of ALU

4.2 RTL Analysis

The register-transfer level (RTL) is a design abstraction that represents the flow of digital signals (data) between hardware registers and the logical operations performed on those signals in a synchronous digital circuit. For each step of the design, the Vivado IDE supports "one click" execution. Following synthesis, the file is co-simulated once again to acquire Verilog files and so examine the RTL Schematic shown in Figure 6.

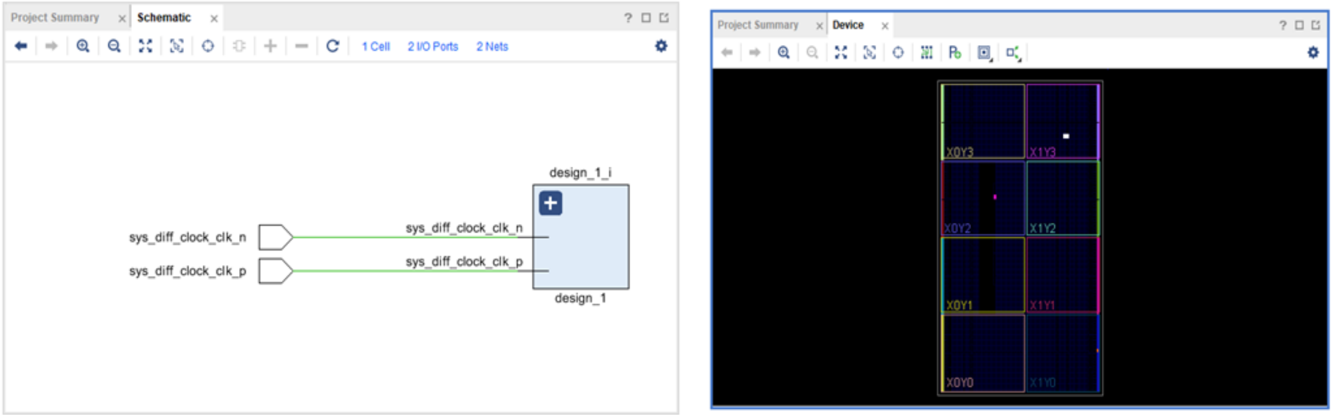


Figure 6: RTL Analysis and Synthesized Design of ALU Design of ALU

4.3 Synthesis

The act of combining parts to form something substantial is known as synthesis shown in Figure 5.8.

4.4 Implementation

ALU is a digital electrical circuit that can do arithmetic and binary values calculations. The Vivado HLS is built on converting a high-level language implementation into a register transfer level implementation described in Figure 7.

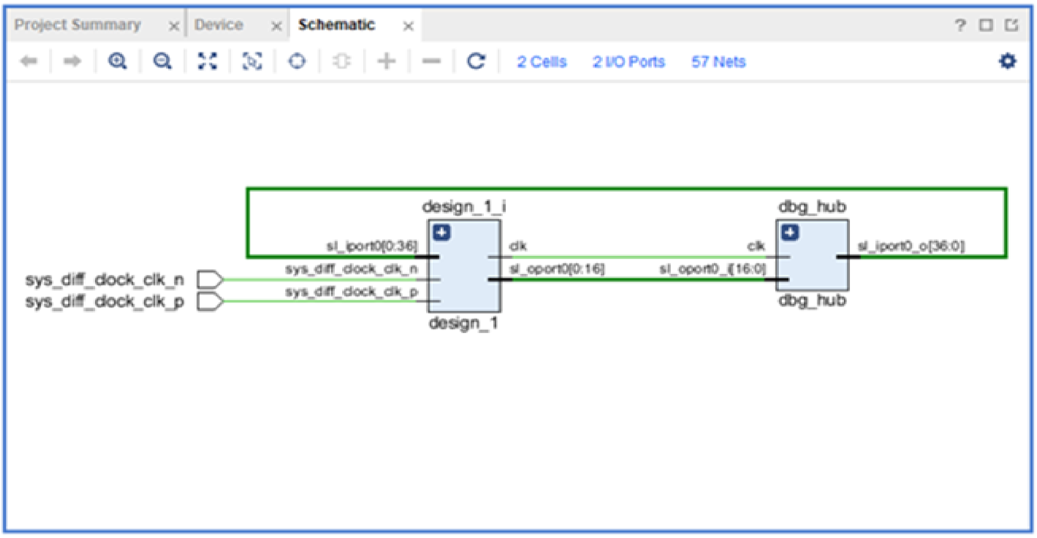


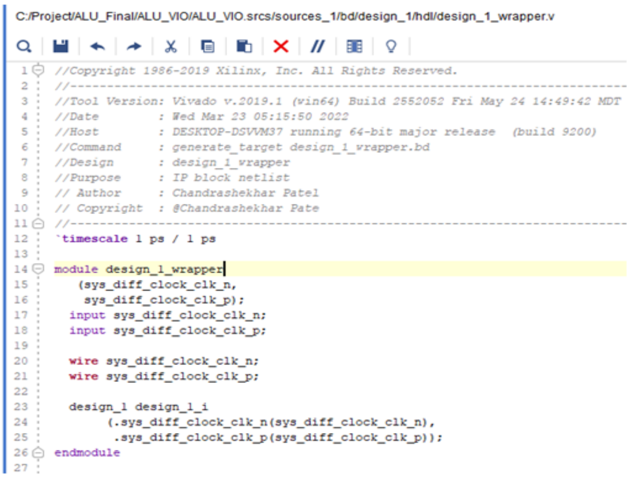
Figure 7: Implementation phase of ALU

4.5 Program and Debug

Internal FPGA signals may be monitored and driven in real time using the Virtual Input/output (VIO) debug function.

4.6 Setting Clock

Clock signals control the outputs of the sequential circuit. That is it determines when and how the memory elements change their outputs. If a sequential circuit is not having any clock signal as input, the output of the circuit will change randomly. Figure 8. shows the code for setting clock for the ALU (Arithmetic Logic Unit) circuit.



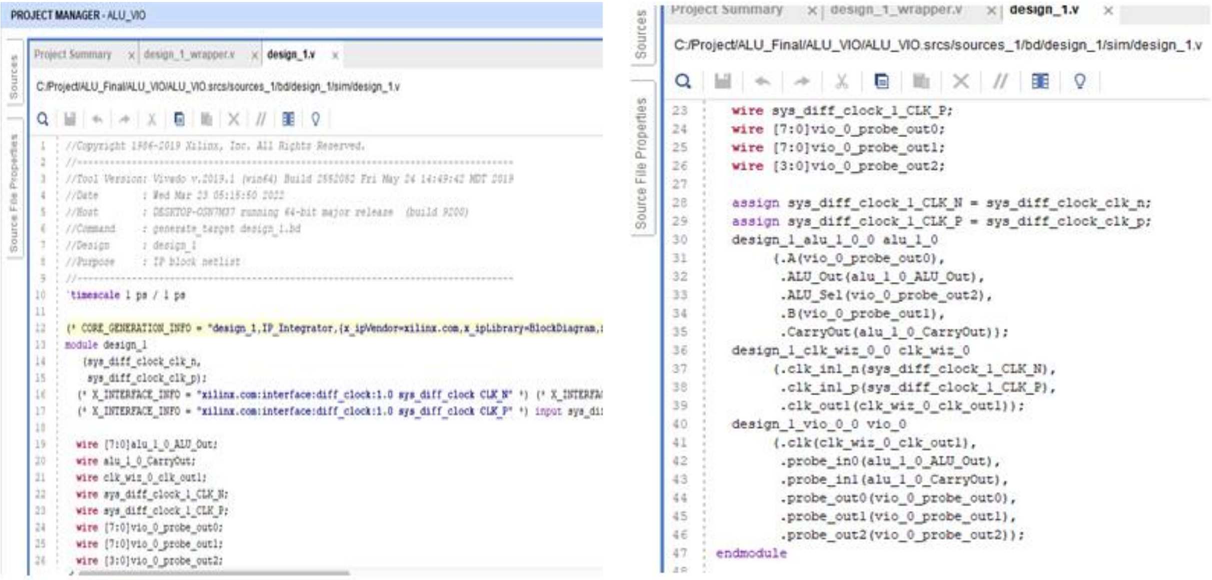
C:/Project/ALU_Final/ALU_VIO/ALU_VIO.srcs/sources_1/bd/design_1/hdl/design_1_wrapper.v

```
//Copyright 1986-2019 Xilinx, Inc. All Rights Reserved.
//--------------------------------------------------------------------------------
//Tool Version: Vivado v.2019.1 (win64) Build 2552052 Fri May 24 14:49:42 MDT
//Date        : Wed Mar 23 05:15:50 2022
//Host        : DESKTOP-DSVVM37 running 64-bit major release  (build 9200)
//Command     : generate_target design_1_wrapper.bd
//Design      : design_1_wrapper
//Purpose     : IP block netlist
// Author     : Chandrashekhar Patel
// Copyright  : @Chandrashekhar Pate
//--------------------------------------------------------------------------------
`timescale 1 ps / 1 ps

module design_1_wrapper
   (sys_diff_clock_clk_n,
    sys_diff_clock_clk_p);
  input sys_diff_clock_clk_n;
  input sys_diff_clock_clk_p;

  wire sys_diff_clock_clk_n;
  wire sys_diff_clock_clk_p;

  design_1 design_1_i
       (.sys_diff_clock_clk_n(sys_diff_clock_clk_n),
        .sys_diff_clock_clk_p(sys_diff_clock_clk_p));
endmodule
```

Figure 8: Setting clock for the ALU (Arithmetic Logic Unit) circuit.

4.7 I/O Pin Planning

The I/O planning features include: an integrated design environment (IDE) to create, configure, assign and manage the I/O Ports and clock logic objects in the design. In Figure 9. shown the code for I/O planning for FPGA based ALU in Vivado IDE.



C:/Project/ALU_Final/ALU_VIO/ALU_VIO.srcs/sources_1/bd/design_1/sim/design_1.v

```
//Copyright 1986-2019 Xilinx, Inc. All Rights Reserved.
//--------------------------------------------------------------------------------
//Tool Version: Vivado v.2019.1 (win64) Build 2552052 Fri May 24 14:49:42 MDT 2019
//Date        : Wed Mar 23 05:15:50 2022
//Host        : DESKTOP-OGN7M37 running 64-bit major release  (build 9200)
//Command     : generate_target design_1.bd
//Design      : design_1
//Purpose     : IP block netlist
//--------------------------------------------------------------------------------
`timescale 1 ps / 1 ps

(* CORE_GENERATION_INFO = "design_1,IP_Integrator,{x_ipVendor=xilinx.com,x_ipLibrary=BlockDiagram,
module design_1
   (sys_diff_clock_clk_n,
    sys_diff_clock_clk_p);
  (* X_INTERFACE_INFO = "xilinx.com:interface:diff_clock:1.0 sys_diff_clock CLK_N" *) (* X_INTERFA
  (* X_INTERFACE_INFO = "xilinx.com:interface:diff_clock:1.0 sys_diff_clock CLK_P" *) input sys_di

  wire [7:0]alu_1_0_ALU_Out;
  wire alu_1_0_CarryOut;
  wire clk_wiz_0_clk_out1;
  wire sys_diff_clock_1_CLK_N;
  wire sys_diff_clock_1_CLK_P;
  wire [7:0]vio_0_probe_out0;
  wire [7:0]vio_0_probe_out1;
  wire [3:0]vio_0_probe_out2;

  assign sys_diff_clock_1_CLK_N = sys_diff_clock_clk_n;
  assign sys_diff_clock_1_CLK_P = sys_diff_clock_clk_p;
  design_1_alu_1_0_0 alu_1_0
       (.A(vio_0_probe_out0),
        .ALU_Out(alu_1_0_ALU_Out),
        .ALU_Sel(vio_0_probe_out2),
        .B(vio_0_probe_out1),
        .CarryOut(alu_1_0_CarryOut));
  design_1_clk_wiz_0_0 clk_wiz_0
       (.clk_in1_n(sys_diff_clock_1_CLK_N),
        .clk_in1_p(sys_diff_clock_1_CLK_P),
        .clk_out1(clk_wiz_0_clk_out1));
  design_1_vio_0_0 vio_0
       (.clk(clk_wiz_0_clk_out1),
        .probe_in0(alu_1_0_ALU_Out),
        .probe_in1(alu_1_0_CarryOut),
        .probe_out0(vio_0_probe_out0),
        .probe_out1(vio_0_probe_out1),
        .probe_out2(vio_0_probe_out2));
endmodule
```

Figure 9: Setting I/O Pin planning for the ALU (Arithmetic Logic Unit) circuit.

4.8 Generating Block Design

In Vivado, a *Hierarchical Block* is a block design within a block design. These blocks allow engineers to partition their designs into separate functional groups. This guide steps through the process of adding a pre-existing hierarchical block to a block design, recreating its example software application, and running the design in hardware illustrates in Figure 10.

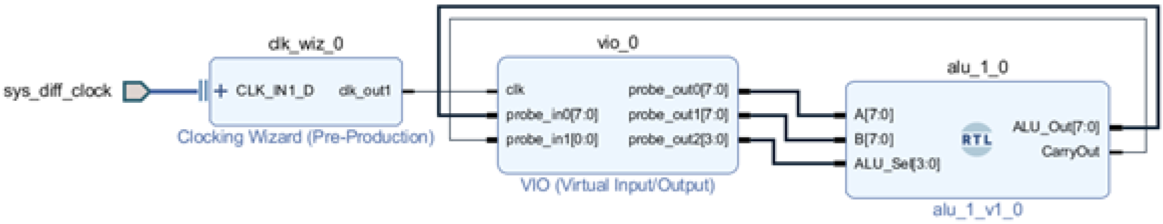


Figure 10: Generating Block design of FPGA based ALU

A bit stream file contains not only the bits needed to configure an FPGA, but also human-readable fields that describe those bits. In fact, the FPGA configuration process is described using an assembly-like instruction set. This note is an attempt to lead you through it's. bit stream file is analogous to an executable programme on a high level. A bit stream, like the ELF format, has its own format for describing its contents. It's worth noting that the file format is openly described. As a result, you may examine the contents of a bitstream file and comprehend the stages involved in configuring the FPGA.

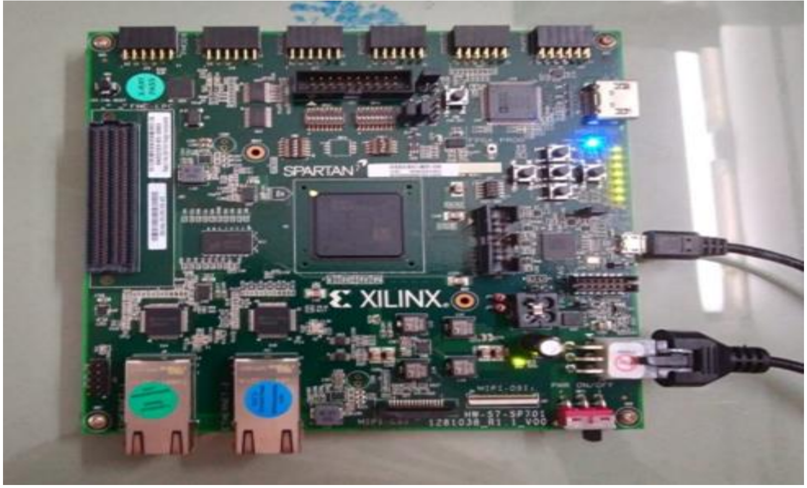


Figure 11: Implementation of ALU with SP701 FPGA Board

Figure 11. and Figure 12. shows the implementation of FPGA based ALU on SP701 FPGA Board in the premises of IoT lab Dev Sanskriti Vishwavidyalaya.

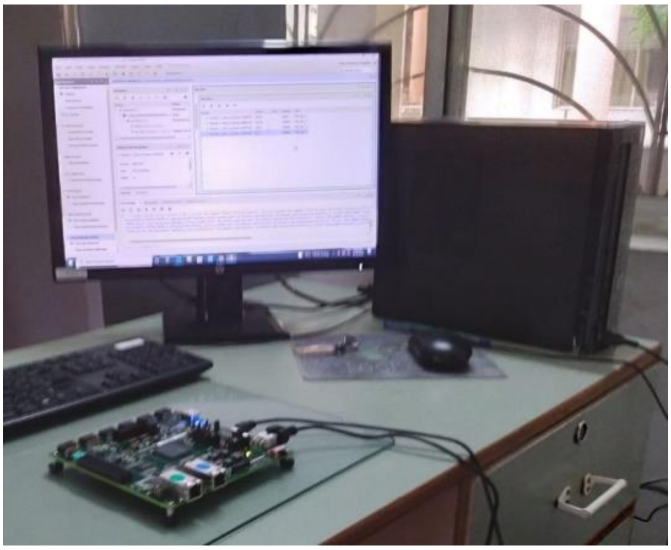
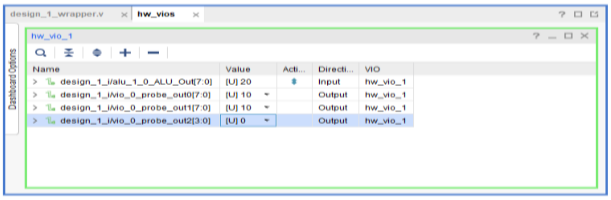
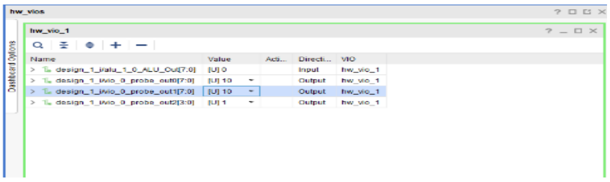
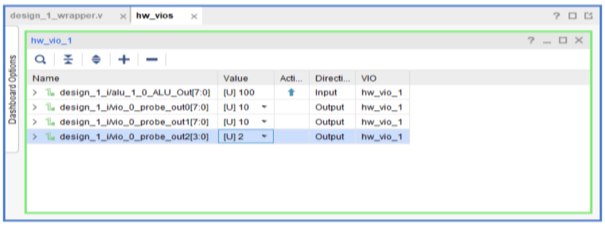
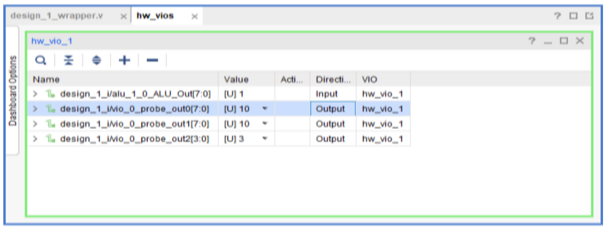


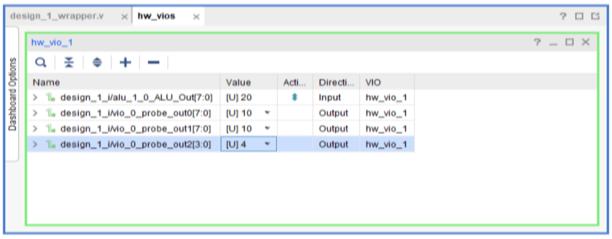
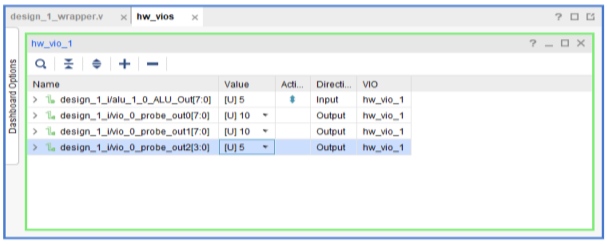
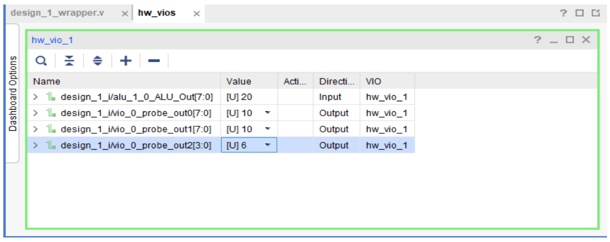
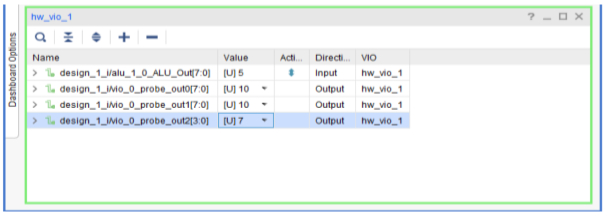
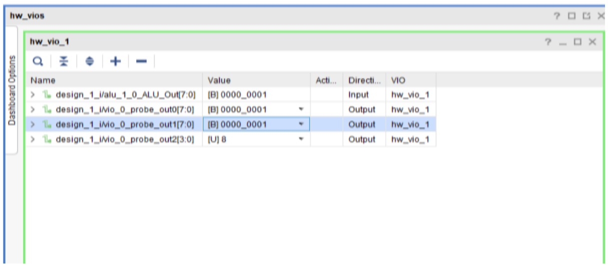
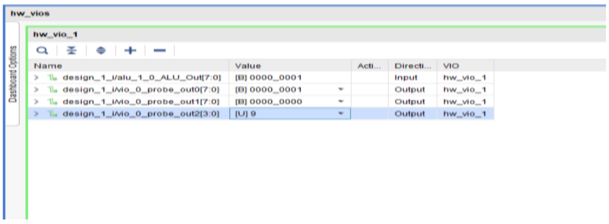
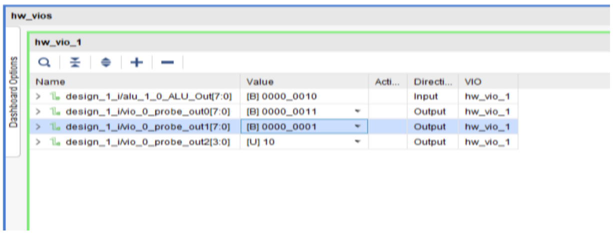
Figure 12: Complete Environment of FPGA based ALU

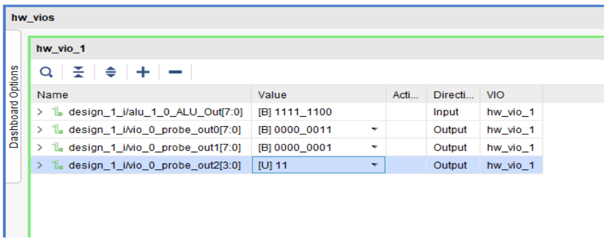
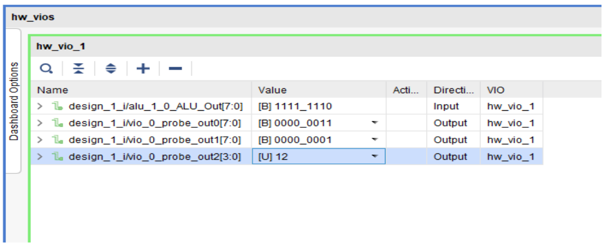
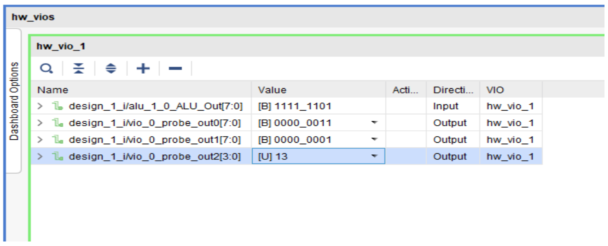
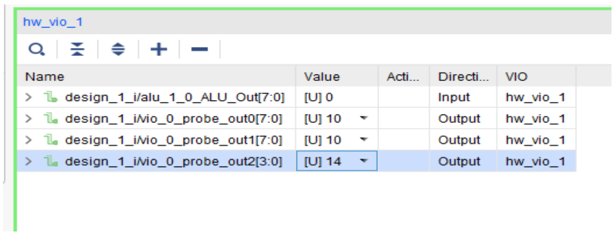
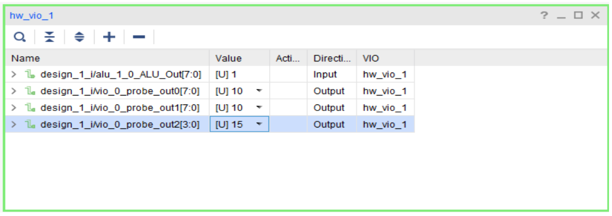
V. SIMULATION RESULTS

During the ALU design researcher work with 4 bit ALU so the total number operation is 2^4=16. Table 1 shown all the operation performed by the ALU with I/O (Input & Output) with its simulation result.

Table 1: Simulation result by FPGA based Arithmetic Logic Unit

S.N.	Operation	Input (1)	Input (2)	Output	Simulation Result
1	ADD	10	10	20	 Name / Value / Acti... / Directi... / VIO design_1_i/alu_1_0_ALU_Out[7:0] / [U] 20 / / Input / hw_vio_1 design_1_i/vio_0_probe_out0[7:0] / [U] 10 / / Output / hw_vio_1 design_1_i/vio_0_probe_out1[7:0] / [U] 10 / / Output / hw_vio_1 design_1_i/vio_0_probe_out2[3:0] / [U] 0 / / Output / hw_vio_1
2	SUBTRACTION	10	10	0	 Name / Value / Acti... / Directi... / VIO design_1_i/alu_1_0_ALU_Out[7:0] / [U] 0 / / Input / hw_vio_1 design_1_i/vio_0_probe_out0[7:0] / [U] 10 / / Output / hw_vio_1 design_1_i/vio_0_probe_out1[7:0] / [U] 10 / / Output / hw_vio_1 design_1_i/vio_0_probe_out2[3:0] / [U] 1 / / Output / hw_vio_1
3	MULTIPLICATION	10	10	100	 Name / Value / Acti... / Directi... / VIO design_1_i/alu_1_0_ALU_Out[7:0] / [U] 100 / / Input / hw_vio_1 design_1_i/vio_0_probe_out0[7:0] / [U] 10 / / Output / hw_vio_1 design_1_i/vio_0_probe_out1[7:0] / [U] 10 / / Output / hw_vio_1 design_1_i/vio_0_probe_out2[3:0] / [U] 2 / / Output / hw_vio_1
4	DIVISION	10	10	1	 Name / Value / Acti... / Directi... / VIO design_1_i/alu_1_0_ALU_Out[7:0] / [U] 1 / / Input / hw_vio_1 design_1_i/vio_0_probe_out0[7:0] / [U] 10 / / Output / hw_vio_1 design_1_i/vio_0_probe_out1[7:0] / [U] 10 / / Output / hw_vio_1 design_1_i/vio_0_probe_out2[3:0] / [U] 3 / / Output / hw_vio_1

5	BINARY SHIFT LEFT	10	10	20	 design_1_wrapper.v hw_vios hw_vio_1 Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [U] 20 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 4 Output hw_vio_1
6	BINARY SHIFT RIGHT	10	10	5	 design_1_wrapper.v hw_vios hw_vio_1 Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [U] 5 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 5 Output hw_vio_1
7	ROTATAE LEFT	10	10	20	 design_1_wrapper.v hw_vios hw_vio_1 Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [U] 20 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 6 Output hw_vio_1
8	ROTATE RIGHT	10	10	5	 hw_vio_1 Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [U] 5 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 7 Output hw_vio_1
9	LOGICAL AND	10	10	0000_0001	 hw_vios hw_vio_1 Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [B] 0000_0001 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [B] 0000_0001 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [B] 0000_0001 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 8 Output hw_vio_1
10	LOGICAL OR	10	10	0000_0001	 hw_vios hw_vio_1 Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [B] 0000_0001 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [B] 0000_0001 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [B] 0000_0000 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 9 Output hw_vio_1
11	LOGICAL XOR	10	10	0000_0010	 hw_vios hw_vio_1 Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [B] 0000_0010 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [B] 0000_0011 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [B] 0000_0001 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 10 Output hw_vio_1

12	LOGICAL NOR	10	10	1111_1100	 hw_vios hw_vio_1 Dashboard Options Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [B] 1111_1100 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [B] 0000_0011 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [B] 0000_0001 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 11 Output hw_vio_1
13	LOGICAL NAND	10	10	1111_1110	 hw_vios hw_vio_1 Dashboard Options Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [B] 1111_1110 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [B] 0000_0011 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [B] 0000_0001 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 12 Output hw_vio_1
14	Logical XNOR	10	10	1111_1101	 hw_vios hw_vio_1 Dashboard Options Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [B] 1111_1101 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [B] 0000_0011 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [B] 0000_0001 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 13 Output hw_vio_1
15	Greater Comparison	10	10	0	 hw_vio_1 Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [U] 0 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 14 Output hw_vio_1
16	Equal Comparison	10	10	1	 hw_vio_1 Name Value Acti... Directi... VIO design_1_i/alu_1_0_ALU_Out[7:0] [U] 1 Input hw_vio_1 design_1_i/vio_0_probe_out0[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out1[7:0] [U] 10 Output hw_vio_1 design_1_i/vio_0_probe_out2[3:0] [U] 15 Output hw_vio_1

VI. CONCLUSION

This paper presents a novel concept for a 4-bit ALU for a processor. An SP701 FPGA board has been used to implement this concept. The primary goal of this effort is to use IO industry standards to decrease the ALU's power consumption. To do this, the researcher determined the best IO standard (LVCMOS) for the ALU by calculating the total power usage using several IO standards. The researcher is additionally attempting to improve the design architecture so that new features may be introduced without modifying the hardware.

FUTURE SCOPE

In this research work researcher has designed ALU on Spartan board but in future this work can be carry forward with some advanced board like Artix-7 Boards. And instead of Io Standard techniques we can do change at architectural level to improve this research work.

DECLARATION

Funding/ Grants/ Financial Support	No, we did not receive.
Conflicts of Interest/ Competing Interests	No conflicts of interest to the best of our knowledge.
Ethical Approval and Consent to Participate	No, the article does not require ethical approval and consent to participate with evidence.
Availability of Data and Material/ Data Access Statement	Data Collection: Vivado Software Implementation tool: SP701 Board

Authors Contributions	Dr. Chandrashekhar Patel: Implementation with SP701 Board, Prof. Abhay Saxena: Idea Generation, Dr. Anita Rawat: Framing the whole idea in conceptual manner Dr. Om Prakash Nautiyal: Analysis of the whole collected data

REFERENCES

1. S.Pandey G.Verma B. Das T.Kumar M.Dhankar ”Energy Efficient Solar Charge Sensor Design Using Spartan-6 FPGA “Gyancity Journal of Electronics and Computer Science, Vol.1, No.1, pp.18-24, September 2016 ISSN: 2446–2918 DOI: 10.21058/gjecs.2016.11004. [CrossRef]
2. A Saxena, A Bhatt, P Gautam, P Verma, C Patel,”High Performance FIFO Design for Processor through Voltage Scaling Technique” In Indian Journal of Science and Technology Vol 9(45), DOI: 10.17485/ijst/2016/v9i45/106916, December 2016. [CrossRef]
3. Swiegers, W., Johan H.R. Enslin, 1998. An Integrated Maximum Power Point Tracker for Photovoltaic Panels. [Online], Available: IEEE Explore database. [20th July 2006].
4. Hussein, K.H, I. Muta, T. Hoshino and M. Osakada, 2006. Maximum Photovoltaic Power Tracking: an algorithm for rapidly changing atmospheric conditions. [Online], IEEE Proceeding of Generation, Transmission and Distribution, pp: 142. [CrossRef]
5. A Saxena, S Gaidhani, A Pant, C Patel “Capacitance Scaling Based Low Power Comparator Design on 28nm FPGA” in International Journal of Computer Trends and Technology (IJCTT) – Volume X Issue Y- Month 2015 [CrossRef]
6. A Saxena, C Patel, M.Khan “Energy Efficient CRC Design for Processor ofWorkstation, and Server using LVCMOS ” in Indian Journal of Science and Technology, Vol 10(4), DOI: 10.17485/ijst/2017/v10i4/110890, January 2017. [CrossRef]
7. A.Singla,A.Kaur, B.Pandey “LVCMOS based energy efficient solar charge sensor design on FPGA” in Power Electronics (IICPE), 2014 IEEE 6th India International Conference DOI: 10.1109/IICPE.2014.7115800. [CrossRef]
8. M. Renovell, J. Figueras, Y. Zorian, “Test of RAM-Based FPGA: Methodology and Application to the Interconnect”, 15th IEEE VLSITest Symposium, pp. 230-237, 1997, Monterey, CA. [CrossRef]
9. R Roux, G. Schoor,P. Vuuren" Block RAM-based architecture for realtime reconfiguration using Xilinx R FPGAs" Research Article – SACJ 56, July 2015. [CrossRef]
10. C.Patel, P.Verma, P. Agarwal, A.Omer,B. Gururani, S.Verma “Designing Green ECG Machine Basedon Artix-7 28nm FPGA ” Gyancity Journal of Engineering and Technology,Vol.3, No.1, pp. 36-41, January 2017ISSN: 2456-0065 DOI:10.21058/gjet.2017.31006 [CrossRef]
11. W.K. Huang and F. Lombardi, “An Approach for Testing Programmable/Configurable Field Programmable Gate Arrays,14th IEEE VLSI Test Symposium, pp. 450-455, Princeton, NJ,USA, May 1996.
12. A Saxena, S Sharma, P Agarwal, C Patel “SSTL Based Energy Efficient FIFO Design for High Performance Processor of Portable Devices” in International Journal of Engineering and Technology (IJET)Vol 9 No 2 Apr-May 2017DOI: 10.21817/ijet/2017/v9i2/170902113. [CrossRef]

AUTHOR PROFILE

Dr. Chandrashekhar Patel is currently associated with the Dev Sanskriti Viswavidyalaya Haridwar. He works in the area of Green Computing. He holds a Post Graduate Degree in the field of Computer Science and currently he is research scholar in Dev Sanskriti University. He has attended many national and international level conferences, workshops, and seminars.

Prof. Abhay Saxena, Dean (School of Technology, Management & Communication) at Dev Sanskriti Vishwaviidyalya, Hardwar, India. Doctoral Degree in Computer Science, Artificial Neural Networks (ANN). Visiting Professor at 3 International University, Academic and industry experience of 26 years. Recently book is An Internet of things - futuristic computing. Authored 8 Books Completed 3 Govt. funded projects completed. Likely to take Joint Research publication, Project along with Visiting Professorship with International Peers.

Prof (Dr.) Anita Rawat working as Director, Uttarakhand Science Education and Research Centre (USERC), Department of Information and Science Technology, Govt. of Uttarakhand is associated with Department of Higher Education, Govt. of Uttarakhand. She obtained her higher education at Panjab University, Chandigarh and later at HNB, Garhwal University, Srinagar, Uttarakhand. She has been meritorious student throughout and medalist in the University. Prof. Anita has been teaching zoology and its allied subjects at Undergraduate and Postgraduate level for 27 years and has been actively engaged in Research from the last almost 25 years in the field of Aquatic Biodiversity, Environmental Management and Environmental Biotechnology.

Prof. Omprakash Nautiyal is acting Director at USERC, Govt. of Uttrakhand. He is a Senior scientist holding Doctoral degree. in Physics. He had completed various DST and national level projects with USERC team.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of the Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)/ journal and/or the editor(s). The Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP) and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.