

A REPORT
ON
ALU (ARITHMETIC AND LOGIC UNIT) USING VERILOG

By

Names of the students

Registration No.

NIKHIL DANGETI

AP21110020072

ARAVIND GONGADA

AP21110020106

Prepared in the partial fulfillment of the
Summer Internship Course

UNDER

(DR PRADYUT KUMAR SANKI SIR)



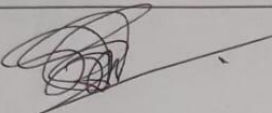
SRM UNIVERSITY, AP

(July, 2023)

Internship Completion Certificate

CERTIFICATE

This is to certify that Summer Internship Project of Nikhil Dangeti titled ALU(Arithmetic And Logic Unit) using Verilog to the best of my knowledge is a record of bonafide work carried out by him under my guidance and/or supervision. The contents embodied in this report, to the best of my knowledge, have not been submitted anywhere else in any form for the award of any other degree or diploma. Indebtedness to other works/publications has been duly acknowledged at relevant places. The project work was carried during 15/06/2023 to 31/08/2023 under Dr Pradyut Kumar Sanki

	
Signature of Faculty Mentor	Signature of industry Mentor/Supervisor (Not required for research internship)
Name: Dr Pradyut Kumar Sanki	Name:
Designation: ASSOCIATE PROFESSOR, ELECTRONICS AND COMMUNICATION ENGINEERING	Designation:
Place: SRM University, AP Date: 31/10/2023	(Seal of the organization with Date)

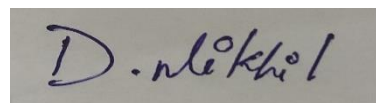
SUMMER INTERNSHIP COURSE, 2023-24

JOINING REPORT

Date: 15/06/2023

Name of the Student	NIKHIL DANGETI
Roll No	AP21110020072
Programme (BTech/ BSc/ BA/MBA)	BTech
Branch	ECE-B
Name and Address of the Internship Company [For research internship, it would be SRMAP]	Telephone No: Email:
Period of Internship	From 15/06/2023 to 06/08/2023

I hereby inform that I have joined the summer internship on 15/06/2023 for the In-plant Training/ Research internship in the industry.



Date : 15/06/2023

Signature of the Student

CERTIFICATE FROM INDUSTRY MENTOR/HR (FACULTY MENTOR FOR RESEARCH INTERNSHIP)

Certified that the above-mentioned student has joined our organization for the INTERNSHIP / INDUSTRIAL TRAINING / ACADEMIC ATTACHMENT in the industry / Organization.

Name of the Industry Mentor	Dr PRADYUT KUMAR SANKI SIR
Designation	ASSOCIATE PROFESSOR, ELECTRONICS AND COMMUNICATION ENGINEERING
Phone No (If any)	
Email	Pradyut.s@srmap.edu.in
Signature & Date	

Acknowledgement:

Apart from the effort of us. The success of any Research Internship depends on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

We would like to show my greatest appreciation to Dr. Pradyut Kumar Sanki Sir. We cannot say thank you enough for his tremendous support and help. We feel motivated and encouraged every time we attend the class. Without his encouragement and guidance this Research Internship would not have materialized.

We are very grateful for his constant support and help.

Abstract:-

This summer internship project focuses on the essential aspects of an Arithmetic Logic Unit (ALU) through the implementation of Verilog modules and testbenches. The ALU serves as a critical component in digital processors, performing arithmetic and logic operations essential for data manipulation. The Verilog modules encapsulate diverse arithmetic and logic operations, providing a versatile and flexible ALU design.

The project primarily involves the creation of Verilog code for the ALU module and its corresponding testbench for functional validation. By concentrating on these fundamental components, the internship aims to provide a hands-on experience in Verilog programming and verification techniques. The outcome of this project will be a well-defined ALU module and a comprehensive testbench, demonstrating a practical understanding of digital circuit design using Verilog.

Table of Contents :-

- 1) INTRODUCTION
- 2) ALU DESIGN OVERVIEW
- 3) Verilog for ALU Design
- 4) CIRCUIT DIAGRAM OF ALU
- 5) TRUTH TABLE OF ALU FOR 3BIT CONTROL INPUT
- 6) MODULE CODE FOR N-BIT ALU WITH 3BIT CONTROL PIN
- 7) TESTBENCH CODE FOR N-BIT ALU WITH 3BIT CONTROL PIN
- 8) OUTPUT WAVEFORMS FOR ALU WITH 3-BIT CONTROL PIN
- 9) TRUTH TABLE OF ALU FOR 4BIT CONTROL INPUT
- 10)MODULE CODE FOR N-BIT ALU WITH 4BIT CONTROL PIN
- 11)TESTBENCH CODE FOR N-BIT ALU WITH 4BIT CONTROL PIN
- 12) OUTPUT WAVEFORMS FOR ALU WITH 4-BIT CONTROL PIN

1) INTRODUCTION :-

ALU stands for Arithmetic Logic Unit. It is a fundamental component of a computer's central processing unit (CPU) responsible for performing arithmetic and logic operations on binary data. The ALU is the part of the CPU where actual computations take place, such as addition, subtraction, multiplication, and logical operations like AND, OR, and NOT.

The ALU operates on binary numbers, which are represented using a series of 0s and 1s. It takes input data from registers within the CPU, processes the data based on the operation specified by the CPU's control unit, and then produces the result as output.

The ALU is a critical component in the execution of computer programs, as it performs the basic arithmetic and logical operations necessary for processing data and making decisions. In modern CPUs, ALUs are highly optimized and can execute multiple operations in parallel to improve overall performance.

2) ALU Design Overview

The ALU typically consists of various functional units responsible for executing specific operations. The key components of an ALU include:

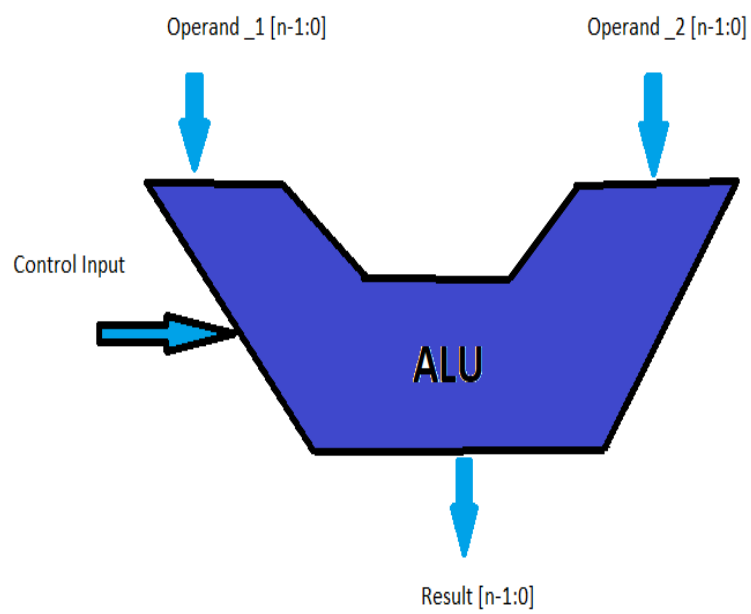
- a. Arithmetic Unit: Handles arithmetic operations such as addition and subtraction.
- b. Logic Unit: Performs logical operations like AND, OR, and NOT.
- c. Control Unit: Determines the operation to be performed based on the control signals received.

3) Verilog for ALU Design

Verilog is a hardware description language commonly used for designing digital circuits. It allows us to describe the behaviour and structure of the ALU in a modular and concise manner. The Verilog code for an ALU typically includes the following sections:

- a. **Module Definition:** The ALU is encapsulated within a module, which serves as the top-level entity. It defines the inputs, outputs, and internal components of the ALU.
- b. **Input/Output Ports:** The module includes input and output ports for data and control signals. These ports facilitate communication between the ALU and other components of the CPU.
- c. **Internal Signals and Registers:** Internal signals and registers are declared within the module to hold intermediate and final results during ALU operations.
- d. **Arithmetic and Logical Operations:** The Verilog code includes the necessary logic to perform arithmetic and logical operations based on the control signals and input data.
- e. **Control Logic:** The control signals determine the type of operation to be executed by the ALU. The control logic interprets these signals and selects the appropriate operation.

4) CIRCUIT DIAGRAM OF ALU



5) TRUTH TABLE OF ALU FOR 3BIT CONTROL INPUT

C3	C2	C1	Operations
0	0	0	A+B Addition
0	0	1	A-B Subtraction
0	1	0	A&B Bitwise AND
0	1	1	A B Bitwise OR
1	0	0	A^B Bitwise XOR
1	0	1	A<<2 Left Shift by 2
1	1	0	A>>2 Right shift by 2
1	1	1	A+1 Adding 1 to A

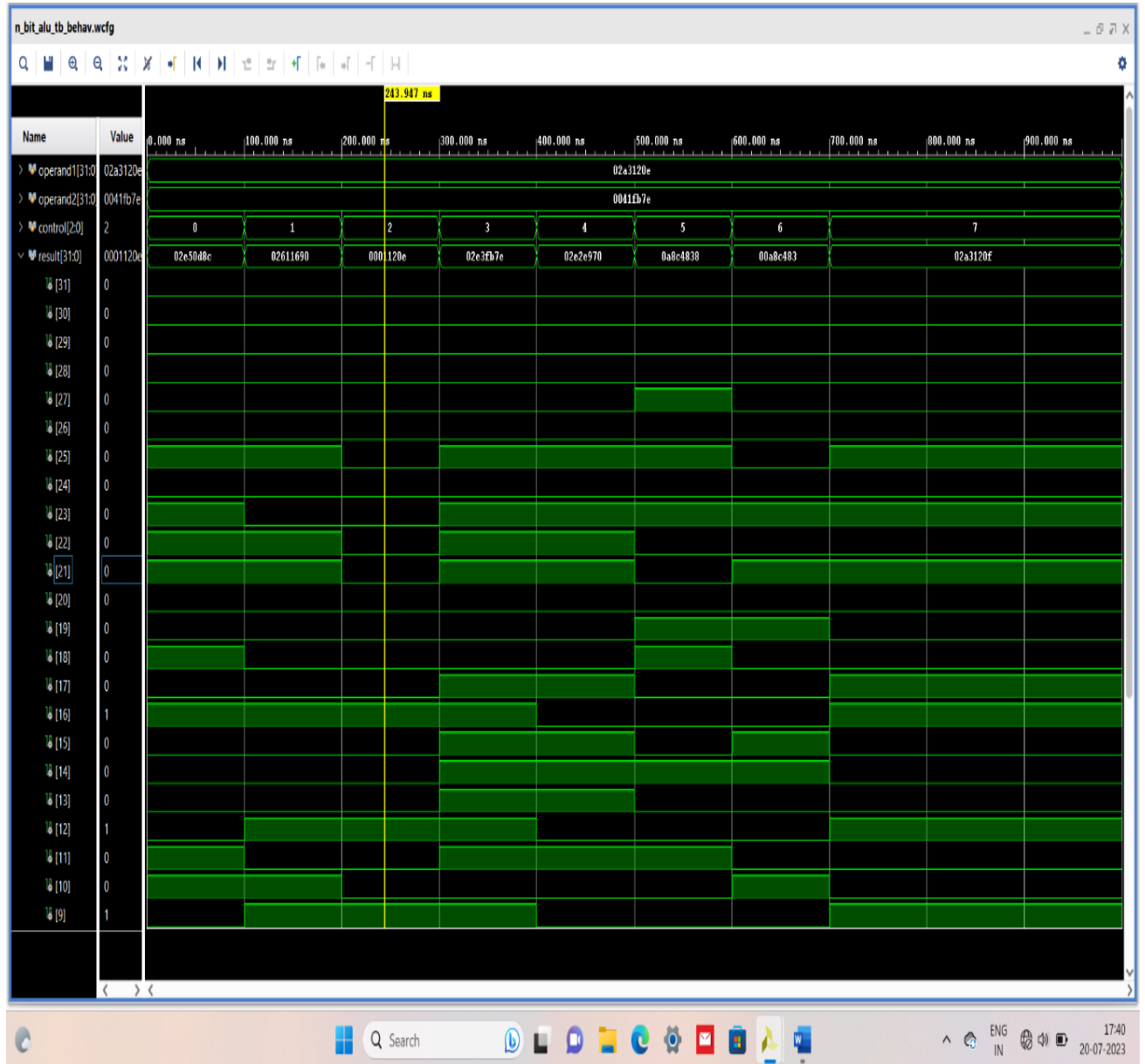
6) MODULE CODE FOR N-BIT ALU FOR 3-BIT CONTROL PINS

```
module n_bit_alu #(parameter n = 32) (  
    input wire [n-1:0] operand1,  
    input wire [n-1:0] operand2,  
    input wire [2:0] control,  
    output reg [n-1:0] result,  
    output reg zero  
);  
always @(*)  
begin  
    case (control)  
        3'b000: result = operand1 + operand2; // Addition  
        3'b001: result = operand1 - operand2; // Subtraction  
        3'b010: result = operand1 & operand2; // Bitwise AND  
        3'b011: result = operand1 | operand2; // Bitwise OR  
        3'b100: result = operand1 ^ operand2; // Bitwise XOR  
        3'b101: result = operand1 << 2; // Left shift  
        3'b110: result = operand1 >> 2; // Right shift  
        3'b111: result = operand1+1; //Increment by 1  
    endcase  
    zero = (result == 0);  
end  
endmodule
```

7) TESTBENCH CODE FOR N-BIT ALU FOR 3-BIT CONTROL PINS

```
module n_bit_alu_tb();  
    // Parameters  
    parameter n = 32;  
  
    // Signals  
    reg [n-1:0] operand1;  
    reg [n-1:0] operand2;  
    reg [2:0] control;  
    wire [n-1:0] result;  
    wire zero;  
  
    initial begin  
        operand1=32'd44241422;  
        operand2=32'd4324222;  
        control=3'b000;  
        #100 control=3'b001;  
        #100 control=3'b010;  
        #100 control=3'b011;  
        #100 control=3'b100;  
        #100 control=3'b101;  
        #100 control=3'b110;  
        #100 control=3'b111;  
    end  
  
    n_bit_alu A(operand1, operand2,control,result,zero);  
  
endmodule
```

8) OUTPUT WAVEFORMS



9) TRUTH TABLE OF ALU FOR 4BIT CONTROL INPUT

C3	C2	C1	C0	Operations
0	0	0	0	A+B Addition
0	0	0	1	A-B Subtraction
0	0	1	0	A*B Multiplication
0	0	1	1	A/B Division
0	1	0	0	Rotate left by 1 position
0	1	0	1	Rotate right by 1 position
0	1	1	0	A&B Bitwise AND
0	1	1	1	A B Bitwise OR
1	0	0	0	A^B Bitwise XOR
1	0	0	1	A<<2 Left Shift by 2
1	0	1	0	A>>2 Right shift by 2
1	0	1	1	~(A B) Bitwise NOR
1	1	0	0	~(A&B) Bitwise NAND
1	1	0	1	~(A^B) Bitwise XNOR
1	1	1	0	(A>B)?4'd1:4'd0 Greater Comparison
1	1	1	1	(A==B)?4'd1:4'd0 Equal Comparison

10) MODULE CODE FOR N-BIT ALU FOR 4-BIT CONTROL PINS

```
module n_bit_alu_four #(parameter n = 32) (  
    input wire [n-1:0] A,  
    input wire [n-1:0] B,  
    input wire [3:0] control,  
    output reg[n-1:0] ALU_Result,  
    output reg zero  
);  
always @(*)  
begin  
    case (control)  
        4'b0000: ALU_Result = A + B ;// Addition  
        4'b0001: ALU_Result = A - B ; // Subtraction  
        4'b0010: ALU_Result = A * B; // Multiplication  
        4'b0011: ALU_Result = A/B; // Division  
        4'b0100: ALU_Result = {A[6:0],A[7]}; // Rotate left by 1 position  
        4'b0101: ALU_Result = {A[0],A[7:1]}; // Rotate right by 1 position  
        4'b0110: ALU_Result = A & B; // Logical and  
        4'b0111: ALU_Result = A | B; // Logical or  
        4'b1000: ALU_Result = A ^ B; // Logical xor  
        4'b1001: ALU_Result = A<<2; // Logical shift left  
        4'b1010: ALU_Result = A>>1; // Logical shift right  
        4'b1011: ALU_Result = ~(A | B); // Logical nor  
        4'b1100: ALU_Result = ~(A & B); // Logical nand  
        4'b1101: ALU_Result = ~(A ^ B); // Logical xnor  
    endcase  
end
```

```
4'b1110: ALU_Result = (A>B)?4'd1:4'd0 ;// Greater comparison
4'b1111: ALU_Result = (A==B)?4'd1:3'd0 ;// Equal comparison
default: ALU_Result = A + B ;
endcase

zero = (ALU_Result == 0);

end

endmodule
```


11) TESTBENCH CODE FOR N-BIT ALU FOR 4-BIT CONTROL PINS

```
module n_bit_alu_four_tb #(parameter n = 32)();  
    reg [n-1:0] A;  
    reg [n-1:0] B;  
    reg [3:0] control;  
    wire [n-1:0] ALU_Result;  
    wire zero;  
    initial begin  
        A=32'd44241422;  
        B=32'd4324222;  
        control=4'b0000;  
        #100 control=4'b0001;  
        #100 A=32'd230005; B=32'd5; control=4'b0010;  
        #100 control=4'b0011;  
        #100 A=32'd44241422; B=32'd4324222; control=4'b0100;  
        #100 control=4'b0101;  
        #100 control=4'b0110;  
        #100 control=4'b0111;  
        #100 control=4'b1000;  
        #100 control=4'b1001;  
        #100 control=4'b1010;  
        /* #100 control=4'b1011;  
        #100 control=4'b1100;  
        #100 control=4'b1101;
```

```

#100 control=4'b1110;

#100 control=4'b1111;*/

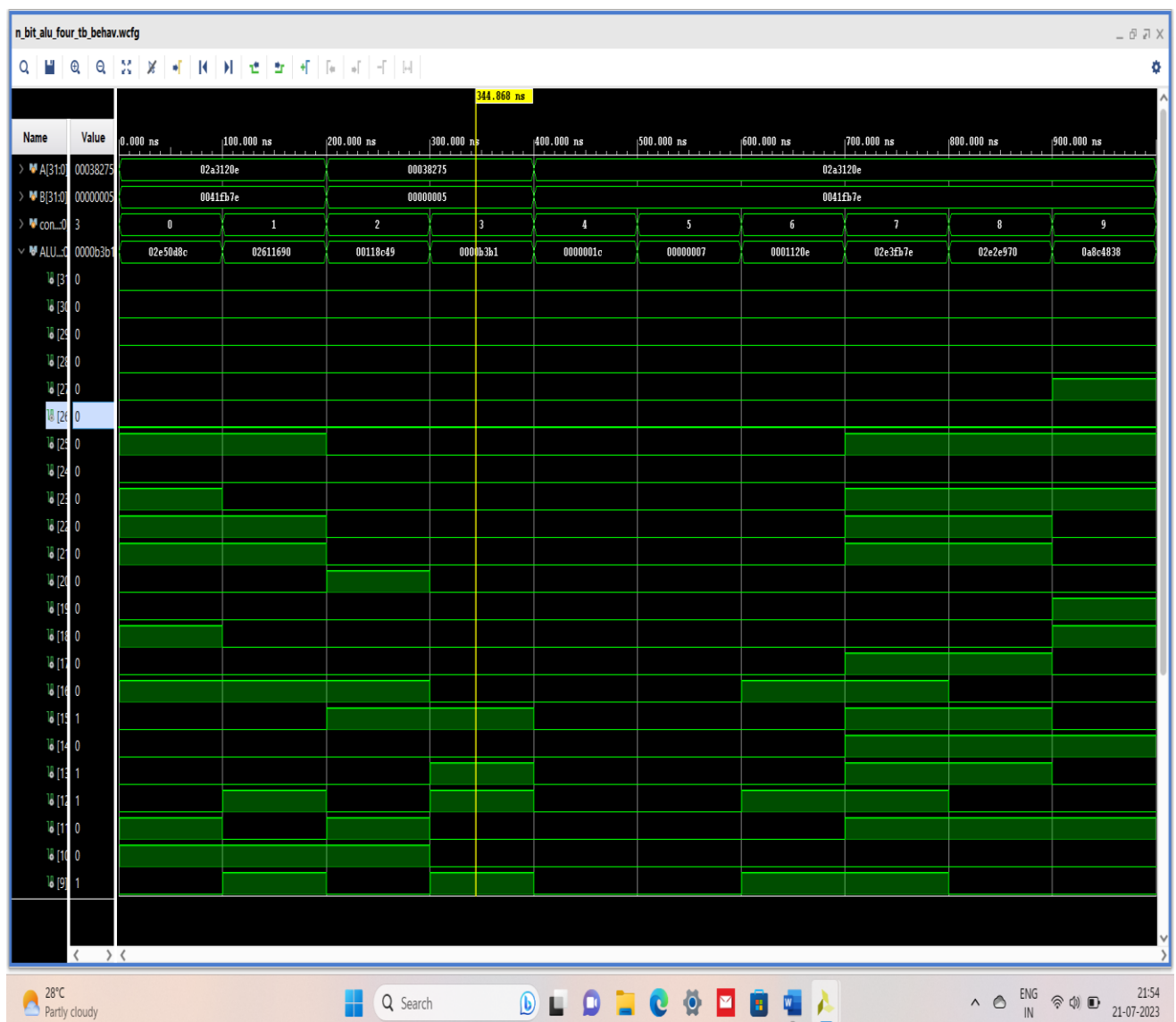
end

n_bit_alu_four AL(A,B,control,ALU_Result,zero);

endmodule

```

12)OUTPUT WAVEFORMS



Conclusion:-

In conclusion, the ALU is a fundamental component of a CPU responsible for executing arithmetic and logical operations. Verilog provides a convenient and efficient way to design and implement an ALU. By utilizing modular design principles and appropriate control logic, we can create a versatile and efficient ALU that meets the computational requirements of modern computing systems.

In this Project we wrote verilog code for n-bit ALU with 3-bit and 4-bit control pin. In place of n we can give 4 to get 4 bit ALU, we can give 8 to get 8 bit ALU, 16 to give 16bit ALU and 32 to get 32bit ALU.

REFERENCES:-

[1] Sakshi Samaiya and Anupreksha jain, "A review article of FPGA ALU unit design based on GA", International journal of scientific reasearch and engineering trends, Volume 4, Issue 4, July-Aug 2018, ISS(online).

[2] N. RAVINDRAN, R.MARY LOURDE, "An optimum VLSI design of 16-bit ALU".

