ALU Project Report

1. Introduction

This project demonstrates a basic Arithmetic Logic Unit (ALU) designed in Verilog.

The ALU supports five basic operations: Addition, Subtraction, AND, OR, and NOT.

It is implemented using a simple case-based structure and is verified using a testbench.

2. ALU Verilog Code

```
module ALU (
    input  [3:0] A,
    input  [3:0] B,
    input  [2:0] opcode,
    output reg [3:0] result
);

always @(*) begin
    case (opcode)
        3'b000: result = A + B;      // Addition
        3'b001: result = A - B;      // Subtraction
        3'b010: result = A & B;      // AND
        3'b011: result = A | B;      // OR
        3'b100: result = ~A;         // NOT (on A only)
        default: result = 4'b0000;
    endcase
end

endmodule
```

3. ALU Testbench Code

ALU Project Report

```
module ALU_tb;

reg [3:0] A, B;
reg [2:0] opcode;
wire [3:0] result;

ALU uut (
    .A(A),
    .B(B),
    .opcode(opcode),
    .result(result)
);

initial begin
    $display("Time | A  B  Opcode | Result");
    $monitor("%4t | %d  %d   %b    | %d", $time, A, B, opcode, result);

    A = 4'd5; B = 4'd3; opcode = 3'b000; #10;
    A = 4'd9; B = 4'd4; opcode = 3'b001; #10;
    A = 4'b1100; B = 4'b1010; opcode = 3'b010; #10;
    A = 4'b1100; B = 4'b1010; opcode = 3'b011; #10;
    A = 4'b1010; B = 4'b0000; opcode = 3'b100; #10;
    opcode = 3'b111; #10;

    $finish;
end

endmodule
```

4. Simulation Report

ALU Project Report

Sample Output from Simulation:

Time | A B Opcode | Result

 0 | 5 3   000 | 8

10 | 9 4   001 | 5

20 | 12 10 010 | 8

30 | 12 10 011 | 14

40 | 10 0 100 | 5 (bitwise NOT of 1010 = 0101)

50 | 10 0 111 | 0