**ALU Design in Verilog**

**// ALU - Basic ALU Module**

module ALU (

input [3:0] A, // 4-bit input A

input [3:0] B, // 4-bit input B

input [2:0] ALU\_Sel, // 3-bit select signal

output reg [3:0] ALU\_Out,

output reg Zero // Zero flag

);

always @(\*) begin

case (ALU\_Sel)

3'b000: ALU\_Out = A + B; // Addition

3'b001: ALU\_Out = A - B; // Subtraction

3'b010: ALU\_Out = A & B; // AND

3'b011: ALU\_Out = A | B; // OR

3'b100: ALU\_Out = ~A; // NOT A

default: ALU\_Out = 4'b0000; // Default case

endcase

**// Set Zero flag**

if (ALU\_Out == 4'b0000)

Zero = 1;

else

Zero = 0;

end

endmodule

**Testbench for the ALU**

**// ALU - Testbench for ALU**

`timescale 1ns / 1ps

module ALU\_tb;

reg [3:0] A, B;

reg [2:0] ALU\_Sel;

wire [3:0] ALU\_Out;

wire Zero;

**// Instantiate the ALU**

ALU uut (

.A(A),

.B(B),

.ALU\_Sel(ALU\_Sel),

.ALU\_Out(ALU\_Out),

.Zero(Zero)

);

initial begin

$display("Time\tA\tB\tSel\tResult\tZero");

A = 4'b0101; B = 4'b0011; ALU\_Sel = 3'b000; #10; // ADD

$display("%0t\t%b\t%b\t%0d\t%b\t%b", $time, A, B, ALU\_Sel, ALU\_Out, Zero);

A = 4'b0101; B = 4'b0011; ALU\_Sel = 3'b001; #10; // SUB

$display("%0t\t%b\t%b\t%0d\t%b\t%b", $time, A, B, ALU\_Sel, ALU\_Out, Zero);

A = 4'b1100; B = 4'b1010; ALU\_Sel = 3'b010; #10; // AND

$display("%0t\t%b\t%b\t%0d\t%b\t%b", $time, A, B, ALU\_Sel, ALU\_Out, Zero);

A = 4'b1100; B = 4'b1010; ALU\_Sel = 3'b011; #10; // OR

$display("%0t\t%b\t%b\t%0d\t%b\t%b", $time, A, B, ALU\_Sel, ALU\_Out, Zero);

A = 4'b1111; B = 4'b0000; ALU\_Sel = 3'b100; #10; // NOT

$display("%0t\t%b\t%b\t%0d\t%b\t%b", $time, A, B, ALU\_Sel, ALU\_Out, Zero);

$finish;

endendmodule

**Simulation Report (Sample Output)**

**Time A B Sel Result Zero**

10 0101 0011 0 1000 0 **// ADD: 5 + 3 = 8**

20 0101 0011 1 0010 0 **// SUB: 5 - 3 = 2**

30 1100 1010 2 1000 0 **// AND: 1100 & 1010 = 1000**

40 1100 1010 3 1110 0 **// OR: 1100 | 1010 = 1110**

50 1111 0000 4 0000 1 **// NOT: ~1111 = 0000 (4-bit result)**