Task 1 - Basic ALU Design (Verilog)

Objective: Design an ALU that supports ADD, SUB, AND, OR, and NOT operations.

Verilog Code (alu.v)

```
module ALU (
    input [3:0] A,
    input [3:0] B,
    input [2:0] ALU_Sel,
    output reg [3:0] ALU_Out
);
always @(*) begin
    case (ALU_Sel)
        3'b000: ALU_Out = A + B;
        3'b001: ALU_Out = A - B;
        3'b010: ALU_Out = A & B;
        3'b011: ALU_Out = A | B;
        3'b100: ALU_Out = ~A;
        default: ALU_Out = 4'b0000;
    endcase
end
endmodule
```

Testbench Code (alu_tb.v)

```
module ALU_tb;
reg [3:0] A, B;
reg [2:0] ALU_Sel;
wire [3:0] ALU_Out;

ALU uut (.A(A), .B(B), .ALU_Sel(ALU_Sel), .ALU_Out(ALU_Out));

initial begin
    $monitor("A=%b B=%b Sel=%b => Out=%b", A, B, ALU_Sel, ALU_Out);
    A = 4'b0101; B = 4'b0011;
    ALU_Sel = 3'b000; #10;
    ALU_Sel = 3'b001; #10;
    ALU_Sel = 3'b010; #10;
    ALU_Sel = 3'b011; #10;
    ALU_Sel = 3'b100; #10;
end
endmodule
```