

Exp -3 Post lab

Design an ALU for any eight operations

Code:

```
//RA2111004010177
```

```
module ALU(result, A,B, X);  
    output reg[7:0] result;  
    input [7:0] A,B;  
    input [2:0] X;  
    always @ (*) begin  
        case({X})  
            3'b000: result = A + B;  
            3'b001: result = A - B;  
            3'b010: result = ~A;  
            3'b011: result = A * B;  
            3'b100: result = A & B;  
            3'b101: result = A | B;  
            3'b110: result = ~(A | B);  
            3'b111: result = A ^ A;  
            default: result = 0;  
        endcase  
    end  
end  
endmodule
```

Testbench:

```
//RA2111004010177
```

```
module ALU_tb_v;  
    reg [7:0] A;  
    reg [7:0] B;  
    reg [2:0] X;  
    wire [7:0] result;  
    ALU uut (  
        .result(result),
```

```
.A(A),  
.B(B),  
.X(X)  
);  
initial begin  
    A = 8'b00001000;  
    B = 8'b00000010;  
    X = 3'b000;  
    #100;  
    A = 8'b00001000;  
    B = 8'b00000010;  
    X = 3'b001;  
    #100;  
    A = 8'b00001000;  
    B = 8'b00000010;  
    X = 3'b010;  
    #100;  
    A = 8'b00001000;  
    B = 8'b00000010;  
    X = 3'b011;  
    #100;  
    A = 8'b00001000;  
    B = 8'b00000010;  
    X = 3'b100;  
    #100;  
    A = 8'b00001000;  
    B = 8'b00000010;  
    X = 3'b101;  
    #100;  
    A = 8'b00001000;  
    B = 8'b00000010;  
    X = 3'b110;
```

```

        #100;

        A = 8'b00001000;

        B = 8'b00000010;

        X = 3'b111;









        #100;

    end

endmodule

```

Output:

  /ALU_tb_v/A	00001000	00001000							
  /ALU_tb_v/B	00000010	00000010							
  /ALU_tb_v/X	000	000	001	010	011	100	101	110	111
  /ALU_tb_v/result	00001010	00001010	00000110	11110111	00010000	00000000	00001010	11110101	00000000