## Day 08 8-bit ALU in Verilog

#### 8-bit ALU in Verilog

#### **Verilog Code**

```
📳 | 🔲 🗗 🏥 賃 | 🖪 🗗 🖺 🕦 😈 🖫 | 🛂 | 🚟 💆
1 module ALU(A,B,sel,result,carry);
module ALU(A,B,Se

2 input [7:0]A,B;

3 input [3:0]sel;

4 output reg [15:0]

5 output reg carry;

6 ⊟always @(*)begin

7

8 result = 0;

9 carry = 0;
     output reg [15:0]result;
     output reg carry;
Ō.
1 =case(sel)
2 |
3 =4'b0000: begin // Addition
4 |{carry, result} = A + B;
end
.6
7 =4'b0001: begin // Subtraction
8.
    |\{carry, result\}| = A - B;
9
     end
0
1 \(\begin\) // Multiplication
2 3 4
     |result = A * B;
     end
5 ⊟4'b0011: begin // Division
6 ⊟if (B != 0) begin
7 |result = A / B;
8
   end
9 ⊟else begin
0
    |result = 16'hxxxx;
1 end
2 end
3 4 =4'b0100: begin // AND
   |result = A \& B;
     end
```

```
8 □4'b0101: begin // OR
9
   |result = A \mid B;
ō
    end
1 2 3 4
  □4'b0110: begin // XOR
    result = A \wedge B;
    end
5
6
7
8
  □4'b0111: begin // NOT A
   |result = \sim A;
    end
.9
0
 □4'b1000: begin // Left Shift
1
   |result = A << 1;
2
    end
4
  □4'b1001: begin // Right Shift
   |result = A >> 1;
6
   end
  ⊟default: begin
8
   |result = 0;
9
    carry = 0;
   rend
12345
   endcase
   end
    endmodule
```

#### Testbench Code

```
💾 | 50 {} | 😉 😉 | M M M M | U 🐚 🐚 🙋 | [28] 🖃
1 module ALU_tb;
2
3 reg [7:0] A,
4 reg [3:0] sel
5 wire [15:0] r
6 wire carry;
7 ALU dut(A,B,s
8 initial begin
9
10
11
A = 8'd10;
         reg [7:0] A, B;
reg [3:0] sel;
         wire [15:0] result;
         ALU dut(A,B,sel,result,carry);
         initial begin
            A = 8'd10; B = 8'd5; sel = 4'b0000; \#10; // Test Addition
11
12
13
            sel = 4'b0001; #10; // Test Subtraction
14
15
            sel = 4'b0100; #10; // Test AND
16
17
            sel = 4'b0101; #10; // Test OR
18
            sel = 4'b1000; #10;// Test Left Shift
19
            $finish;
20
         end
21
22 endmodule
```

## **Waveform**

<b>≥1</b> +	Msgs							
+// /ALU_tb/A	00001010	00001010						
<b>III</b> → /ALU_tb/B	00000101	00000101						
<b>±</b> -∜ /ALU_tb/sel	0101	0000		0001		0100		0101
→ /ALU_tb/result	0000000000001111	000000000000	1111	000000000000000000000000000000000000000	1	000000000000000000000000000000000000000	0	0000000000000
/ /ALU_tb/carry	St0							
<u> </u>								

# **Schematic**

# Refer github

### **EDA Tools Used**

- •IntelQuartusPrime
- ModelSim

© 2025 Satyam Nishad | 100 Days RTL Coding Challenge