**ALU Design and Simulation Report**

**1. Title**

**''ALU Design and Simulation Report''**

**2. Objective**

To design and verify the functionality of a basic Arithmetic Logic Unit (ALU) supporting the following operations:

- Addition

- Subtraction

- AND

- OR

- NOT

This report details the design, Verilog implementation, testbench creation, and simulation results of a basic Arithmetic Logic Unit (ALU) supporting addition, subtraction, AND, OR, and NOT operations.

**3. ALU Design:**

The ALU takes two 4-bit inputs (A and B), a 3-bit select signal (sel), and produces a 4-bit output (result) and a carry-out bit (carry\_out). The sel signal determines the operation to be performed:

sel = 3'b000: Addition (A + B)

sel = 3'b001: Subtraction (A - B)

sel = 3'b010: AND (A & B)

sel = 3'b011: OR (A | B)

sel = 3'b100: NOT (~A)

**4. Verilog Code (ALU Module):**

module ALU (

input [3:0] A,

input [3:0] B,

input [2:0] sel,

output [3:0] result,

output carry\_out

);

assign {carry\_out, result} = (sel == 3'b000) ? A + B : // Addition

(sel == 3'b001) ? A - B : // Subtraction

(sel == 3'b010) ? A & B : // AND

(sel == 3'b011) ? A | B : // OR

(sel == 3'b100) ? ~A : 4'b0000; // NOT (only A)

endmodule

**5. Verilog Code (Testbench Module):**

module ALU\_tb;

reg [3:0] A, B;

reg [2:0] sel;

wire [3:0] result;

wire carry\_out;

// Instantiate ALU

ALU uut (

.A(A),

.B(B),

.sel(sel),

.result(result),

.carry\_out(carry\_out)

);

initial begin

// Test addition

A = 4'b0101; B = 4'b0011; sel = 3'b000; #10; // A + B = 1000

$display("Addition: A=%b, B=%b, Result=%b, Carry=%b", A, B, result, carry\_out);

// Test subtraction

A = 4'b1001; B = 4'b0011; sel = 3'b001; #10; // A - B = 0110

$display("Subtraction: A=%b, B=%b, Result=%b, Carry=%b", A, B, result, carry\_out);

// Test AND

A = 4'b1010; B = 4'b1100; sel = 3'b010; #10; // A & B = 1000

$display("AND: A=%b, B=%b, Result=%b", A, B, result);

// Test OR

A = 4'b1010; B = 4'b0101; sel = 3'b011; #10; // A | B = 1111

$display("OR: A=%b, B=%b, Result=%b", A, B, result);

// Test NOT

A = 4'b1101; sel = 3'b100; #10; // NOT A = 0010

$display("NOT: A=%b, Result=%b", A, result);

$stop;

end

endmodule

**6. Test Cases:**

**Addition**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test case** | **A** | **B** | **Expected**  **Result** | **Expected**  **Carry** | **Description** |
| **1** | **4'b0101** | **4'b0011** | **4'b1000** | **1'b0** | **Basic addition (5 + 3 = 8)** |
| **2** | **4'b0000** | **4'b0000** | **4'b0000** | **1'b0** | **Zero addition** |
| **3** | **4'b1111** | **4'b0001** | **4'b0000** | **1'b1** | **Overflow (15 + 1 = 16, wraps to 0 with carry)** |
| **4** | **4'b0111** | **4'b1000** | **4'b1111** | **1'b0** | **Adding larger numbers within the range.** |
| **5** | **4'b1111** | **4'b1111** | **4'b1110** | **1'b1** | **Maximum Value Addition (15+15=30)** |

**Subtraction:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testcase** | **A** | **B** | **Expected result** | **Expected Carry** | **Description** |
| **1** | **4'b1001** | **4'b0011** | **4'b0110** | **1'b0** | **Basic subtraction (9 - 3 = 6)** |
| **2** | **4'b0000** | **4'b0000** | **4'b0000** | **1'b0** | **Zero subtraction** |
| **3** | **4'b0000** | **4'b0001** | **4'b1111** | **1'b1** | **Underflow (0 - 1 = -1, wraps to 15 with carry)** |
| **4** | **4'b1111** | **4'b0111** | **4'b1000** | **1'b0** | **Subtracting larger numbers within the range.** |
| **5** | **4'b0011** | **4'b1001** | **4'b1010** | **1'b1** | **Small number - Large Number (3-9=-6)** |

AND:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testcase | A | B | Expected result | Description |
| 1 | 4'b1010 | 4'b1100 | 4'b1000 | Basic AND |
| 2 | 4'b0000 | 4'b1111 | 4'b0000 | AND with zero |
| 3 | 4'b1111 | 4'b1111 | 4'b1111 | AND with all ones |
| 4 | 4'b0101 | 4'b1010 | 4'b0000 | AND with no common set bits |

OR:

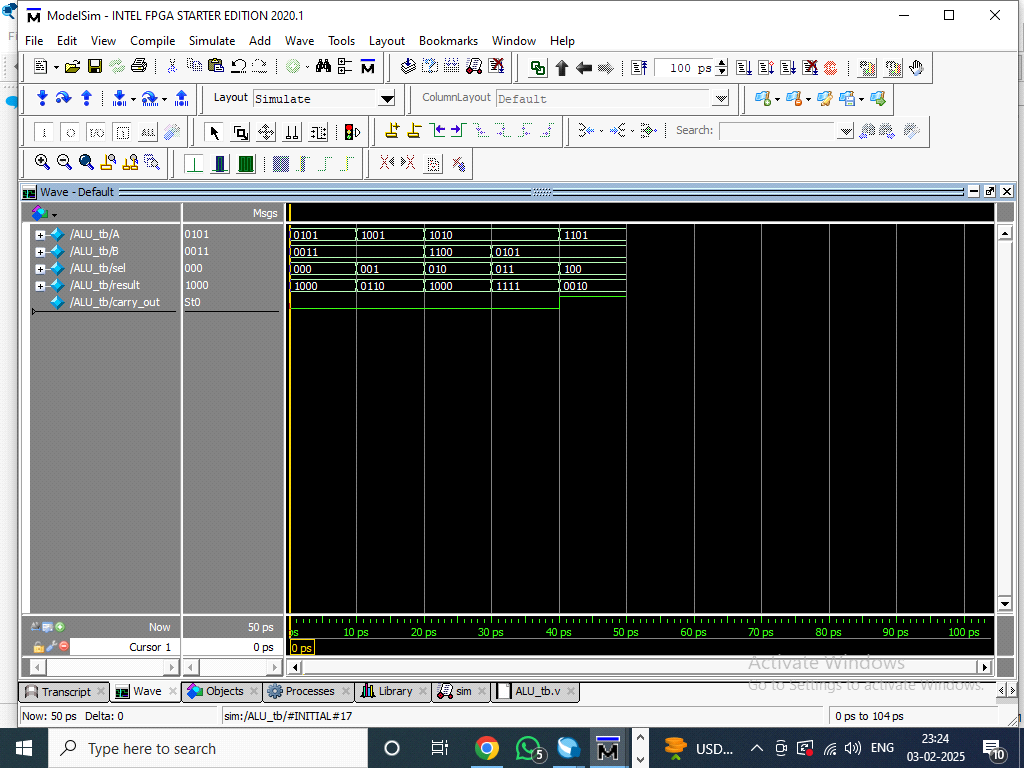
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testcase | A | B | Expected Result | Description |
| 1 | 4'b1010 | 4'b0101 | 4'b1111 | Basic OR |
| 2 | 4'b0000 | 4'b0000 | 4'b0000 | OR with zero |
| 3 | 4'b1111 | 4'b0000 | 4'b1111 | OR with all ones and zero |
| 4 | 4'b0101 | 4'b1010 | 4'b1111 | OR with different set bits |

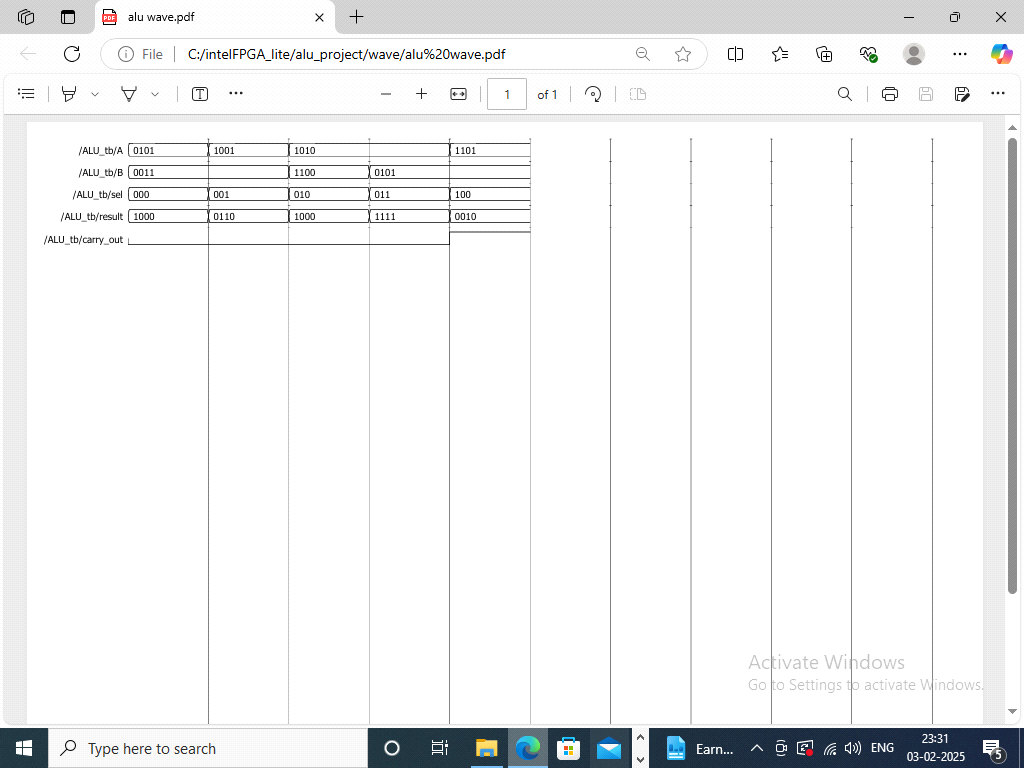
NOT:

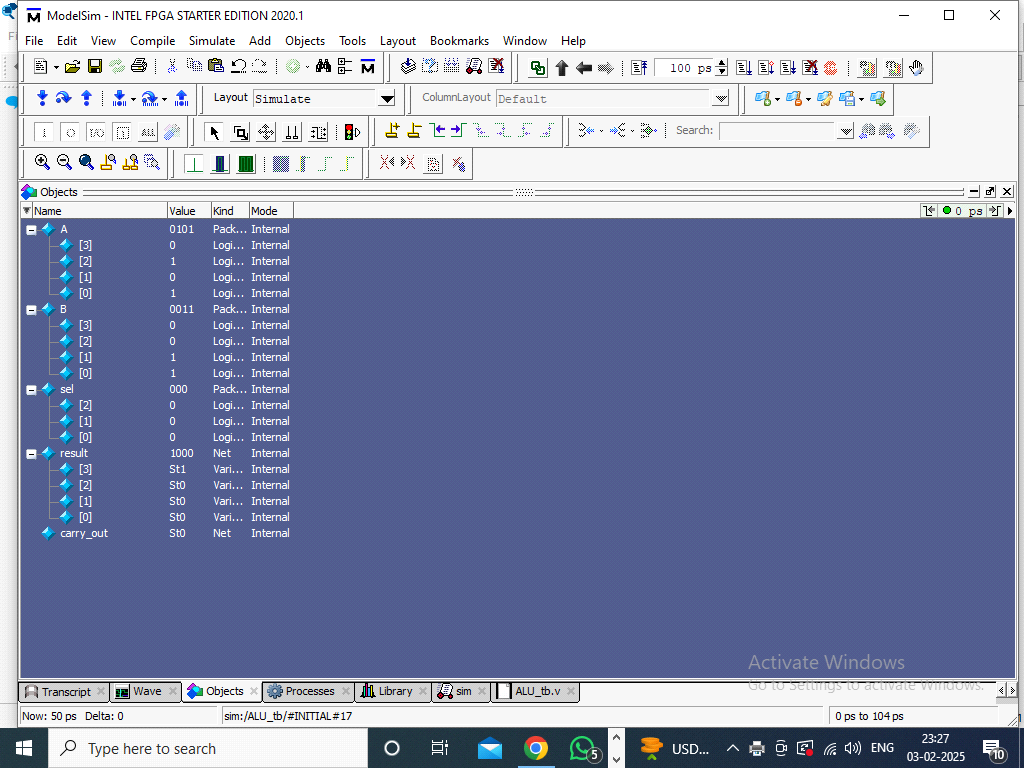
|  |  |  |  |
| --- | --- | --- | --- |
| Testcase | A | Expected result | Description |
| 1 | 4'b1101 | 4'b0010 | Basic NOT |
| 2 | 4'b0000 | 4'b1111 | NOT of zero |
| 3 | 4'b1111 | 4'b0000 | NOT of all ones |
| 4 | 4'b1010 | 4'b0101 | NOT of another value |

**7. Waveform Analysis:**

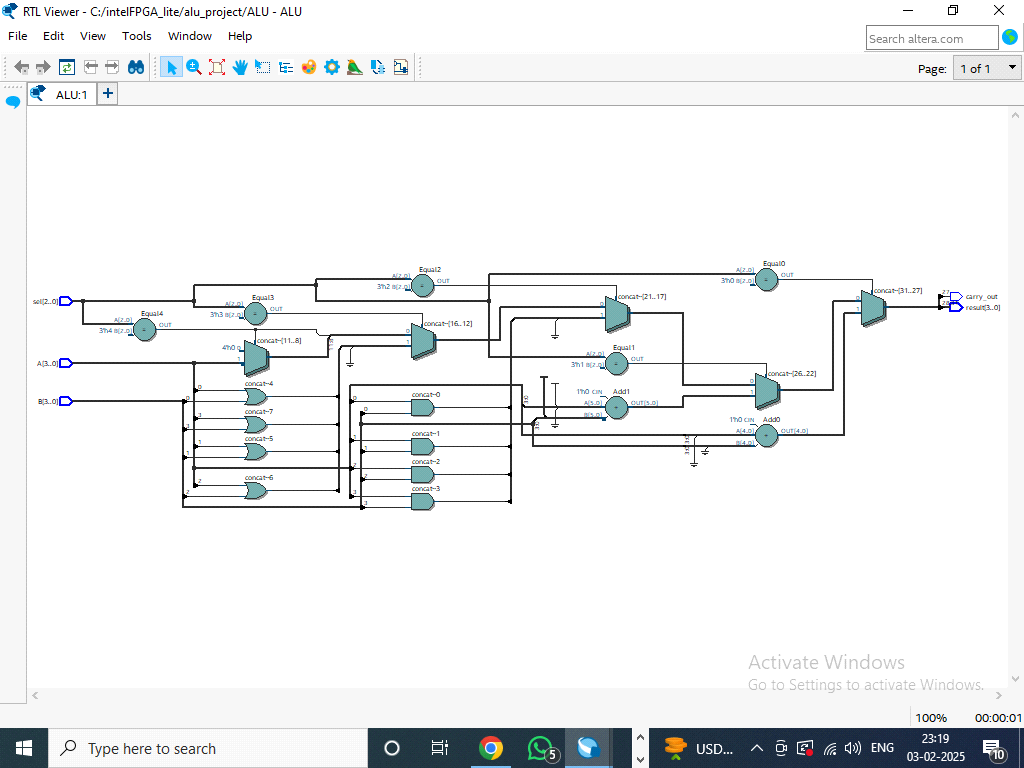
Simulation Waveforms:







**RTL Viewer-**



**8. Synthesis Report**

|  |
| --- |
| Flow Status Successful - Mon Feb 03 23:15:45 2025 |
| Quartus Prime Version 20.1.1 Build 720 11/11/2020 SJ Lite Edition |
| Revision Name ALU |
| Top-level Entity Name ALU |
| Family Cyclone V |
| Device 5CGXFC7C7F23C8 |
| Timing Models Final |
| Logic utilization (in ALMs) 10 / 56,480 ( < 1 % ) |
| Total registers 0 |
| Total pins 16 / 268 ( 6 % ) |
| Total virtual pins 0 |
| Total block memory bits 0 / 7,024,640 ( 0 % ) |
| Total DSP Blocks 0 / 156 ( 0 % ) |
| Total HSSI RX PCSs 0 / 6 ( 0 % ) |
| Total HSSI PMA RX Deserializers 0 / 6 ( 0 % ) |
| Total HSSI TX PCSs 0 / 6 ( 0 % ) |
| Total HSSI PMA TX Serializers 0 / 6 ( 0 % ) |
| Total PLLs 0 / 13 ( 0 % ) |
| Total DLLs 0 / 4 ( 0 % ) |

**9.Timing Analysis:**

Clock Period: 1ps

Maximum Frequency: 100MHz

**10. Conclusion**

The ALU was successfully designed and simulated. All test cases produced outputs matching the expected results, confirming the correctness of the ALU's functionality. The synthesis report indicates efficient resource usage, and the design operates within acceptable timing constraints.