



## Laboratory Report # 4

Name: Christian Jay Gallardo

Date Completed: 9/19/2025

Laboratory Exercise Title: Dataflow Modeling of Combinational Circuits

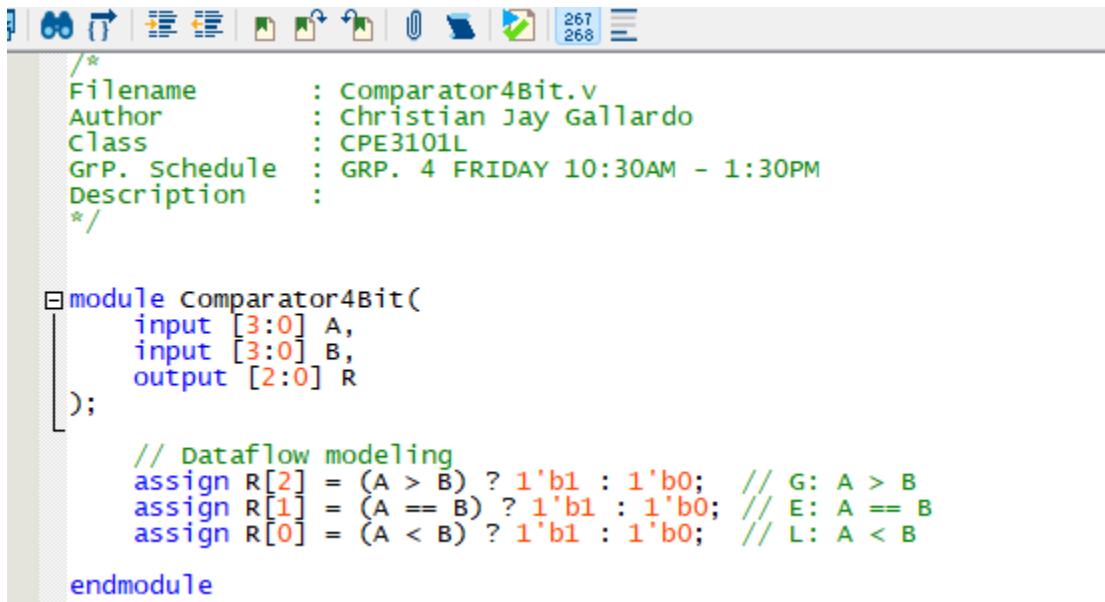
### Target Course Outcomes:

**CO1:** Create descriptions of digital hardware components such as in combinational and sequential circuits using synthesizable Verilog HDL constructs.

**CO2:** Verify the functionality of HDL-based components through design verification tools.

---

### Exercise 4A:



The screenshot shows a Verilog HDL code editor window. The title bar indicates the file is named "Comparator4Bit.v". The code itself defines a module named "comparator4Bit" with three inputs (A, B) and one output (R). The output R is a 3-bit vector where each bit represents a comparison result: R[2] for A > B, R[1] for A == B, and R[0] for A < B. The code uses a case statement to assign the appropriate value to each bit based on the comparison of A and B.

```
/*
Filename      : Comparator4Bit.v
Author        : Christian Jay Gallardo
Class         : CPE3101L
GrP. Schedule : GRP. 4 FRIDAY 10:30AM - 1:30PM
Description   :
*/
module comparator4Bit(
    input [3:0] A,
    input [3:0] B,
    output [2:0] R
);
    // Dataflow modeling
    assign R[2] = (A > B) ? 1'b1 : 1'b0; // G: A > B
    assign R[1] = (A == B) ? 1'b1 : 1'b0; // E: A == B
    assign R[0] = (A < B) ? 1'b1 : 1'b0; // L: A < B
endmodule
```

Figure 1 Verilog HDL Code for the 4-bit Comparator



The screenshot shows the Quartus Prime Lite software interface. The main window displays a Verilog HDL testbench code for a 4-bit comparator. The code defines a testbench module `tb\_Comparator4Bit` that instantiates a `Comparator4Bit` unit. It contains 10 test cases for various input combinations (A, B) and output conditions (G, E, L). The test cases are labeled from 1 to 10. The code uses the \$display system task to print the test cases and their results. The software interface includes a Project Navigator, Entity Instance list, and various toolbars and panels for project management and simulation.

```
Quartus Prime Lite Edition - C:/Users/CJ/Documents/HDL/Lab4-20250919T142727Z-1-001/Lab4/Comparator4Bit - Comparator4Bit
File Edit View Project Assignments Processing Tools Window Help
Project Navigator Hierarchy ▾ EntityInstance MAX 10:10M02DCU324AG6
Comparatore4Bit
EntityInstance
MAX 10:10M02DCU324AG6
Comparatore4Bit
Comparatore4Bitv tb_Comparator4Bitv Compilation Report - Comparator4Bit IP Catalog
Search altera.com
8
9 module tb_Comparator4Bit;
10    reg [3:0] A, B;
11    wire [2:0] R;
12
13    // Instantiate the comparator
14    Comparator4Bit uut(
15        .A(A),
16        .B(B),
17        .R(R)
18    );
19
20 initial begin
21    // Test Case 1: A > B
22    A = 4'b1010; B = 4'b101; #10;
23    $display("Test 1: A=%b, B=%b, R=%b (G=%b, E=%b, L=%b)", A, B, R, R[2], R[1], R[0]);
24
25    // Test Case 2: A < B
26    A = 4'b0011; B = 4'b101; #10;
27    $display("Test 2: A=%b, B=%b, R=%b (G=%b, E=%b, L=%b)", A, B, R, R[2], R[1], R[0]);
28
29    // Test Case 3: A = B
30    A = 4'b1111; B = 4'b1111; #10;
31    $display("Test 3: A=%b, B=%b, R=%b (G=%b, E=%b, L=%b)", A, B, R, R[2], R[1], R[0]);
32
33    // Test Case 4: A > B (edge case)
34    A = 4'b0001; B = 4'b0000; #10;
35    $display("Test 4: A=%b, B=%b, R=%b (G=%b, E=%b, L=%b)", A, B, R, R[2], R[1], R[0]);
36
37    // Test Case 5: A < B (edge case)
38    A = 4'b1100; B = 4'b110; #10;
39    $display("Test 5: A=%b, B=%b, R=%b (G=%b, E=%b, L=%b)", A, B, R, R[2], R[1], R[0]);
40
41    // Test Case 6: A > B
42    A = 4'b1100; B = 4'b1011; #10;
43    $display("Test 6: A=%b, B=%b, R=%b (G=%b, E=%b, L=%b)", A, B, R, R[2], R[1], R[0]);
44
45    // Test Case 7: A < B
46    A = 4'b0010; B = 4'b1010; #10;
47    $display("Test 7: A=%b, B=%b, R=%b (G=%b, E=%b, L=%b)", A, B, R, R[2], R[1], R[0]);
48
49    // Test Case 8: A = B
50    A = 4'b1010; B = 4'b1010; #10;
51    $display("Test 8: A=%b, B=%b, R=%b (G=%b, E=%b, L=%b)", A, B, R, R[2], R[1], R[0]);
52
53    // Test Case 9: A > B
54    A = 4'b1110; B = 4'b1101; #10;
55    $display("Test 9: A=%b, B=%b, R=%b (G=%b, E=%b, L=%b)", A, B, R, R[2], R[1], R[0]);
56
57    // Test Case 10: A < B
58    A = 4'b0010; B = 4'b0011; #10;
59    $display("Test 10: A=%b, B=%b, R=%b (G=%b, E=%b, L=%b)", A, B, R, R[2], R[1], R[0]);
60
Tasks Compilation Task
Compile Design
Analysis & Synthesis
Edit Settings
View Report
Analysis & Elaboration
Partition Merge
Netlist Viewers
RTL Viewer
State Machine Viewer
TechHindu Man View
File Edit View Project Assignments Processing Tools Window Help
All Find Next
Type ID Message
22036 successfully launched NativeLink simulation (quartus_sh -t "c:/intelfpga_lite/20.1/quartus/common/tcl/internal/nativelink/qnativesim.tcl" --rtl_sim "Comparator4Bit" "Comparator4Bit")
22036 For messages from NativeLink execution see the NativeLink Log file c:/users/CJ/documents/HDL/Lab4-20250919T142727Z-1-001/Lab4/Comparator4Bit_nativeLink_simulation.rpt
System (4) Processing (128)
100% 00:00:18
26°C Mostly cloudy
Search ENG INTL 11:01 pm 19/09/2023

```

Figure 2 Verilog HDL testbench code for the 4-bit Comparator

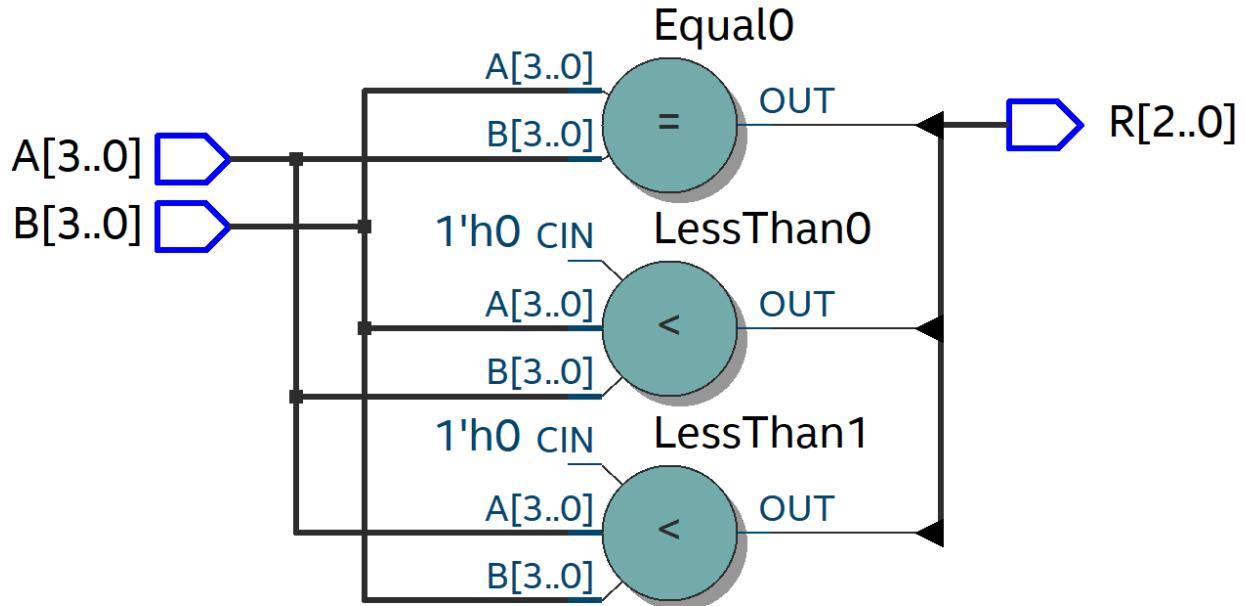


Figure 3 RTL Viewer of the 4-bit Comparator

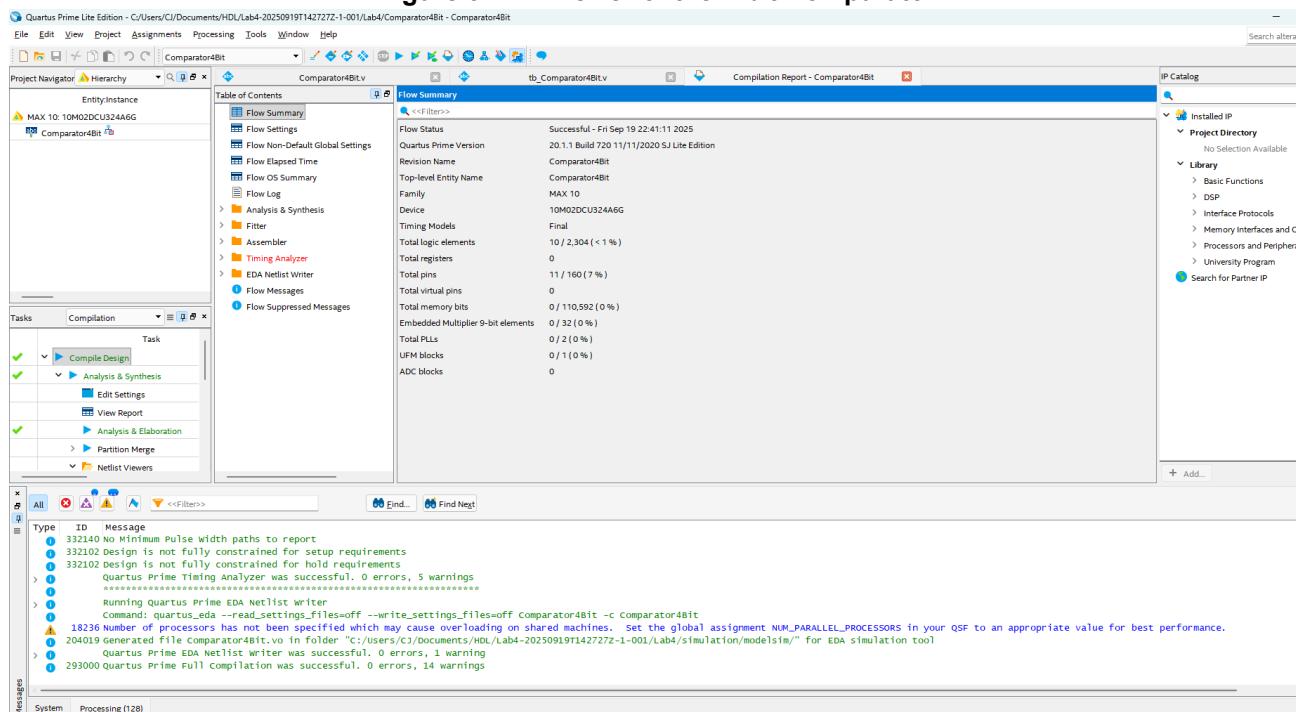


Figure 4 Summary of Design Synthesis for the 4-bit comparator



Figure 5 Successful simulation results with 10 test cases with annotations



### Exercise 4B: Part 1

Table 1 Truth Table

S (Selector)	A (Input)	B (Input)	C (Input)	D (Input)	Y (Output)
00	0001	0010	0100	1000	0001
01	0001	0010	0100	1000	0010
10	0001	0010	0100	1000	0100
11	0001	0010	0100	1000	1000

```
/*
Filename      : The_Mux_4x1_nbit.v
Author        : Christian Jay Gallardo
Class         : CPE3101L
GrP. Schedule : GRP. 4 FRIDAY 10:30AM - 1:30PM
Description   : The Mux_4x1_nbit is a parameterized 4-to-1 multiplexer that selects one of four
                n-bit inputs based on a 2-bit selector and outputs the selected value.
*/



module Mux_4x1_nbit #(parameter n = 4) (
    input [n-1:0] A, B, C, D, // 4 input buses
    input [1:0] S,           // selector input (2 bits)
    output [n-1:0] Y         // output bus
);
    // Simple case statement to select the output based on S
    assign Y = (S == 2'b00) ? A :
               (S == 2'b01) ? B :
               (S == 2'b10) ? C :
               (S == 2'b11) ? D : A; // Default case (for safety)
endmodule
```

Figure 6 Verilog HDL code for the n-bit 4-to-1 multiplexer



```
/*
  Filename      : tb_Mux_4x1_nbit.v
  Author       : Christian Jay Gallardo
  Class        : CPE3101L
  GRP. Schedule : GRP. 4 FRIDAY 10:30AM - 1:30PM
  Description   : The tb_Mux_4x1_nbit testbench tests the 4-bit multiplexer by applying all
                  selector values (S) and displaying the corresponding output (Y) for each set of input
*/
module tb_Mux_4x1_nbit;
    // Parameters and signals for the testbench
    reg [3:0] A, B, C, D; // 4-bit input buses
    reg [1:0] S;          // selector input (2 bits)
    wire [3:0] Y;         // output wire

    // Instantiate the Mux_4x1_nbit module
    Mux_4x1_nbit #(4) uut(
        .A(A), .B(B), .C(C), .D(D),
        .S(S), .Y(Y)
    );

    // Test procedure
    initial begin
        // Apply test vectors
        A = 4'b0001; B = 4'b0010; C = 4'b0100; D = 4'b1000;

        // Test case 1: S = 00
        S = 2'b00; #10;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // Test case 2: S = 01
        S = 2'b01; #10;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // Test case 3: S = 10
        S = 2'b10; #10;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // Test case 4: S = 11
        S = 2'b11; #10;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // End of simulation
        $finish;
    end
endmodule
```

Figure 7 Verilog HDL testbench code for the 4-bit 4-to-1 multiplexer

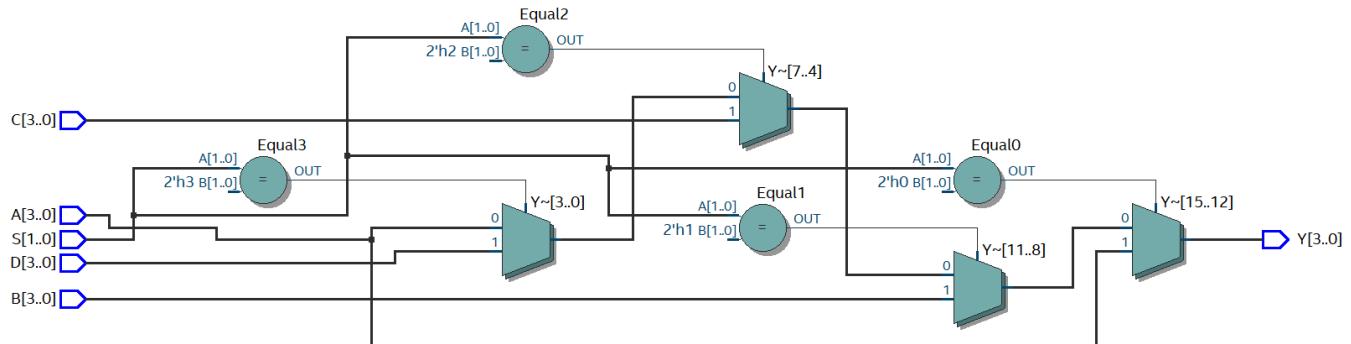


Figure 8 RTL Viewer of the n-bit 4-to-1 Multiplexer

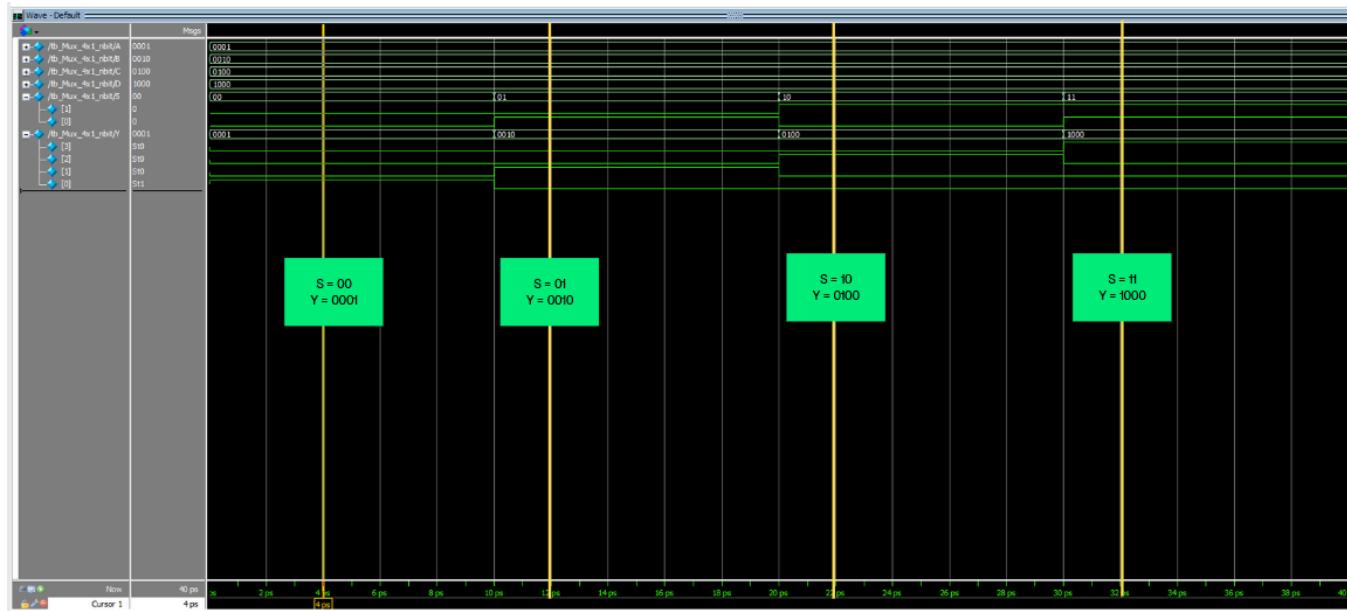


Figure 9 Successful simulation results of the 4-bit 4-to-1 Multiplexer with annotations

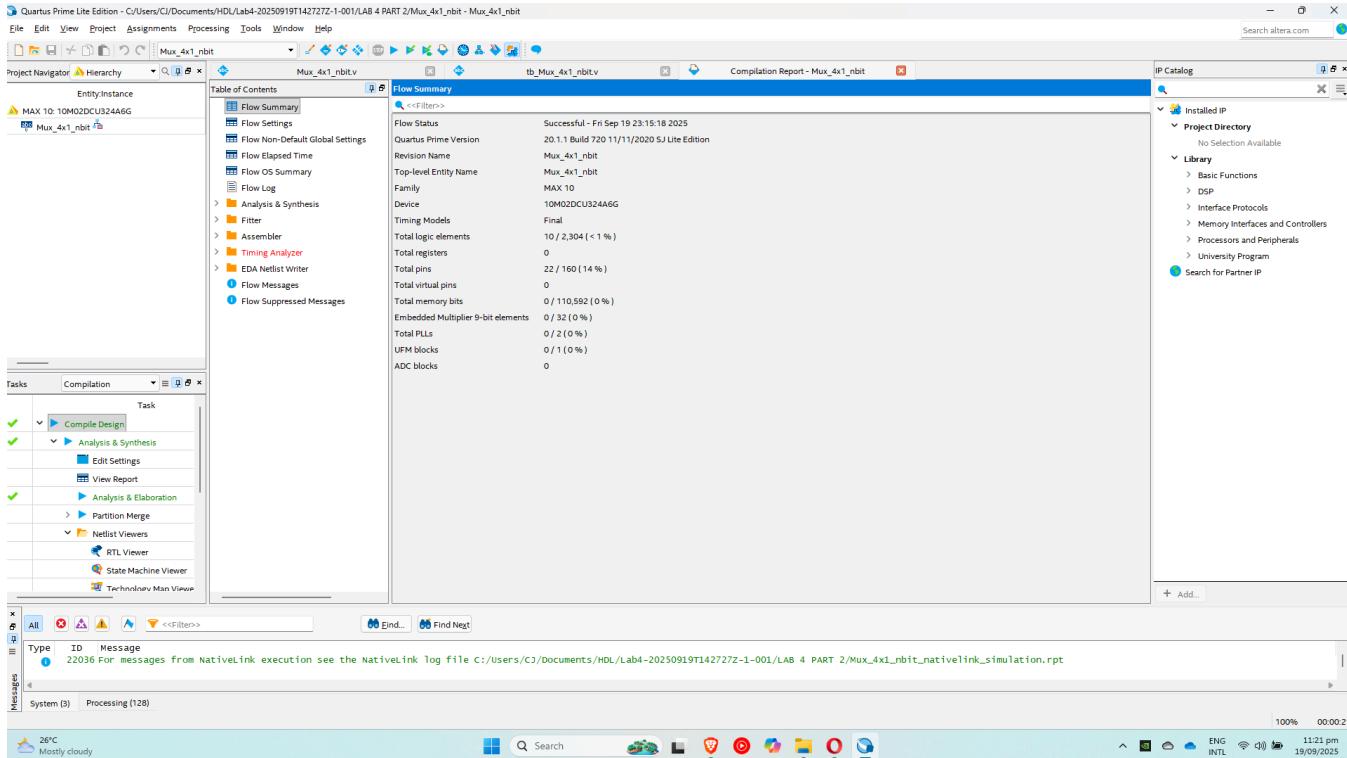


Figure 10 Summary of Design Synthesis of the 4-bit 4-to-1 Multiplexer with annotations



## Exercise 4B: Part 2

Table 2 Truth Table

S (Selector)	A (Input)	B (Input)	C (Input)	D (Input)	Y (Output)
00	1111	0000	1010	0101	1111
01	1111	0000	1010	0101	0000
10	1111	0000	1010	0101	1010
11	1111	0000	1010	0101	0101

The screenshot shows the Quartus Prime Lite Edition interface with the following details:

- Project Navigator:** MAX 10 10M02DCDU324AGG, EntityInstance Mux\_4x1\_nbit.
- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Toolbar:** Standard icons for Open, Save, Print, etc.
- Central Area:** Displays the Verilog HDL code for the testbench. The code instantiates a MUX\_4x1\_nbit module and defines a test procedure to apply various input combinations and check the output Y.
- IP Catalog:** Shows installed IP components like Project Directory, Library, and University Program.
- Tasks:** Compilation, Analysis & Elaboration, Netlist Viewers, RTL Viewer, State Machine Viewer, Technology Map View.
- Messages:** Shows a single message: "293000 Quartus Prime Full compilation was successful. 0 errors, 14 warnings".
- System Bar:** Shows system status (26°C, Mostly cloudy), search bar, and system icons.

```
module tb_Mux_4x1_nbit2;
    // Parameters and signals for the testbench
    reg [3:0] A, B, C, D; // 4-bit input buses
    reg [1:0] S; // selector inputs (2 bits)
    wire [3:0] Y; // output wire

    // Instantiate the mux_4x1_nbit module
    MUX_4x1_nbit #() uut (
        .A(A), .B(B), .C(C), .D(D),
        .S(S), .Y(Y)
    );

    initial begin
        // Apply test vectors
        A = 4'b0001; B = 4'b0100; C = 4'b1000;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);
        // Test case 1: S = 00, output should be A
        S = 2'b00;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // Test case 2: S = 01, output should be B
        S = 2'b01;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // Test case 3: S = 10, output should be C
        S = 2'b10;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // Test case 4: S = 11, output should be D
        S = 2'b11;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // Test case 5: Change inputs to check for different combinations
        A = 4'b1111; B = 4'b0000; C = 4'b0101; D = 4'b1010;

        // Test case 6: S = 00 with new inputs, output should be A
        S = 2'b00;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // Test case 7: S = 01 with new inputs, output should be B
        S = 2'b01;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // Test case 8: S = 10 with new inputs, output should be C
        S = 2'b10;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // Test case 9: S = 11 with new inputs, output should be D
        S = 2'b11;
        $display("S = %b, A = %b, B = %b, C = %b, D = %b, Y = %b", S, A, B, C, D, Y);

        // End of simulation
    end
endmodule
```

Figure 11 Verilog HDL testbench code for the 4-bit 4-to-1 multiplexer

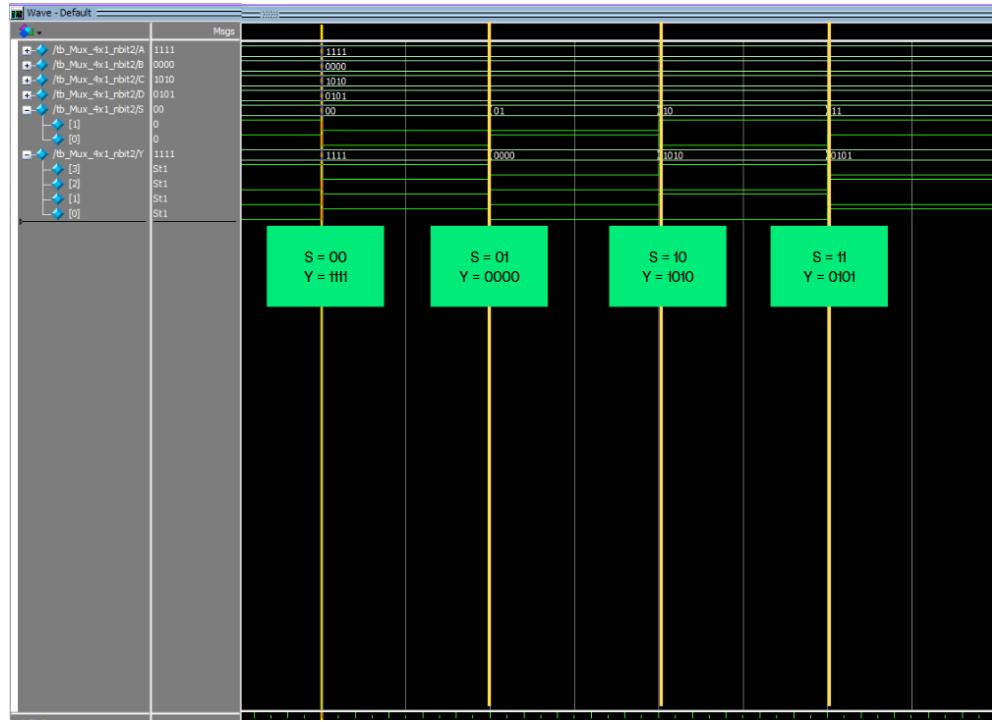


Figure 12 Successful simulation results of the 8-bit 4-to-1 Multiplexer with annotations

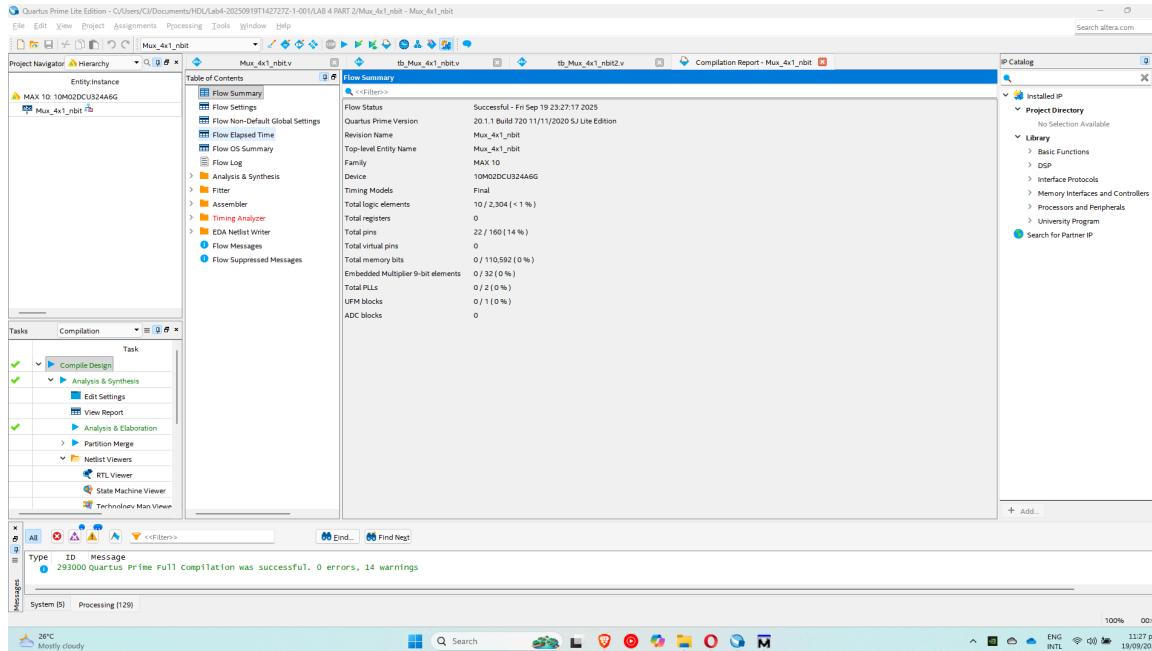


Figure 13 Summary of Design Synthesis of the 8-bit 4-to-1 Multiplexer with annotations