LAB NO 12

Fall 2024

CSE-304L Computer Organization and Architecture Lab

Submitted by:

Name : **Hassan Zaib Jadoon**

Reg no**. : 22PWCSE2144**

Class Section **: A**

Signature: _____

Submitted to:

Dr. Amaad Khalil

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

ADDER AND ADDER SUBTRACTOR

TASK:1

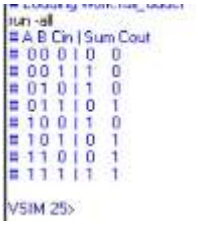
Write a Verilog code for Full Adder using Dataflow Level modeling.

CODE:

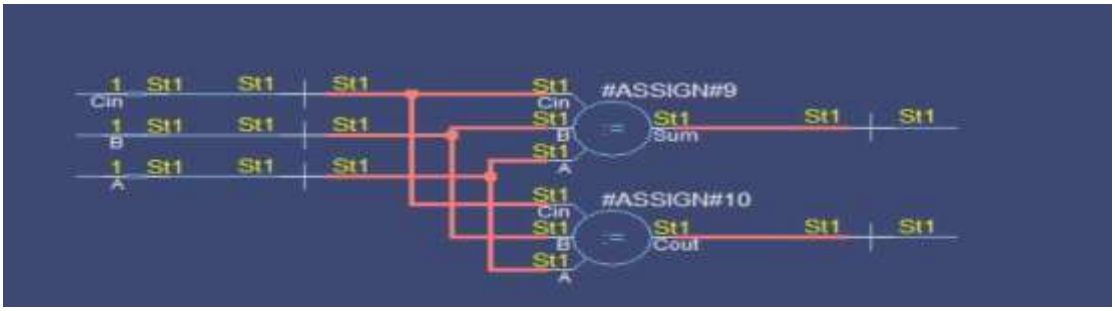
```
module stimFA();
    reg A, B, Cin;      // Inputs: A, B, and Carry-in
    wire Sum, Cout;     // Outputs: Sum and Carry-out

    // Instantiate the full adder module
    full_adder FA(A, B, Cin, Sum, Cout);

    initial
    begin
        $display("A B Cin | Sum Cout");

        // Test all input combinations
        A = 0; B = 0; Cin = 0; #10
        $display(" %b %b  %b  |  %b    %b", A, B, Cin, Sum, Cout);

        A = 0; B = 0; Cin = 1; #10
        $display(" %b %b  %b  |  %b    %b", A, B, Cin, Sum, Cout);

        A = 0; B = 1; Cin = 0; #10
        $display(" %b %b  %b  |  %b    %b", A, B, Cin, Sum, Cout);

        A = 0; B = 1; Cin = 1; #10
        $display(" %b %b  %b  |  %b    %b", A, B, Cin, Sum, Cout);

        A = 1; B = 0; Cin = 0; #10
        $display(" %b %b  %b  |  %b    %b", A, B, Cin, Sum, Cout);

        A = 1; B = 0; Cin = 1; #10
        $display(" %b %b  %b  |  %b    %b", A, B, Cin, Sum, Cout);

        A = 1; B = 1; Cin = 0; #10
        $display(" %b %b  %b  |  %b    %b", A, B, Cin, Sum, Cout);

        A = 1; B = 1; Cin = 1; #10
```

OUTPUT:

Table output:



```
run -all
# A B Cin | Sum Cout
#  0 0  0  |  0    0
#  0 0  1  |  1    0
#  0 1  0  |  1    0
#  0 1  1  |  0    1
#  1 0  0  |  1    0
#  1 0  1  |  0    1
#  1 1  0  |  0    1
#  1 1  1  |  1    1

VSIM 25>
```



REMARKS:

The Verilog code for the Full Adder using Dataflow Level modeling is concise and clear, utilizing assign statements to directly express the relationships between inputs and outputs. It accurately implements the Full Adder logic by calculating the Sum as the XOR of inputs and the Cout using the OR of multiple AND operations. The code is compact, easy to understand, and correctly models the Full Adder's behavior. However, it can be improved in terms of adding comments for further clarification, especially for beginners, and ensuring the code adheres to specific naming conventions if required by a larger project.

TASK:2

Write a Verilog code for Half Adder using Dataflow Level modeling.

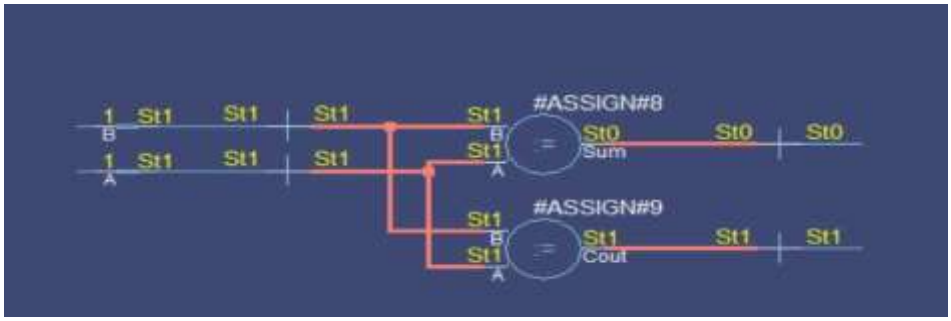
CODE:

```
module HalfAdder (
    input A,     // First input
    input B,     // Second input
    output Sum,  // Sum output
    output Cout // Carry output
);

    // Dataflow modeling
    assign Sum = A ^ B;          // Sum is XOR of inputs
    assign Cout = A & B;         // Carry is AND of inputs

endmodule
```

```
module stimHA();
    reg A, B;          // Inputs: A and B
    wire Sum, Cout;    // Outputs: Sum and Carry-out

    // Instantiate the Half Adder module
    half_adder HA(A, B, Sum, Cout);

    initial
    begin
        $display("A B | Sum Cout");

        // Test all possible input combinations
        A = 0; B = 0; #10
        $display("%b %b |  %b    %b", A, B, Sum, Cout);

        A = 0; B = 1; #10
        $display("%b %b |  %b    %b", A, B, Sum, Cout);

        A = 1; B = 0; #10
        $display("%b %b |  %b    %b", A, B, Sum, Cout);

        A = 1; B = 1; #10
        $display("%b %b |  %b    %b", A, B, Sum, Cout);
    end
endmodule
```

OUTPUT:

```
run -all
# A B | Sum Cout
# 0 0 | 0   0
# 0 1 | 1   0
# 1 0 | 1   0
# 1 1 | 0   1

VSIM 30>
```



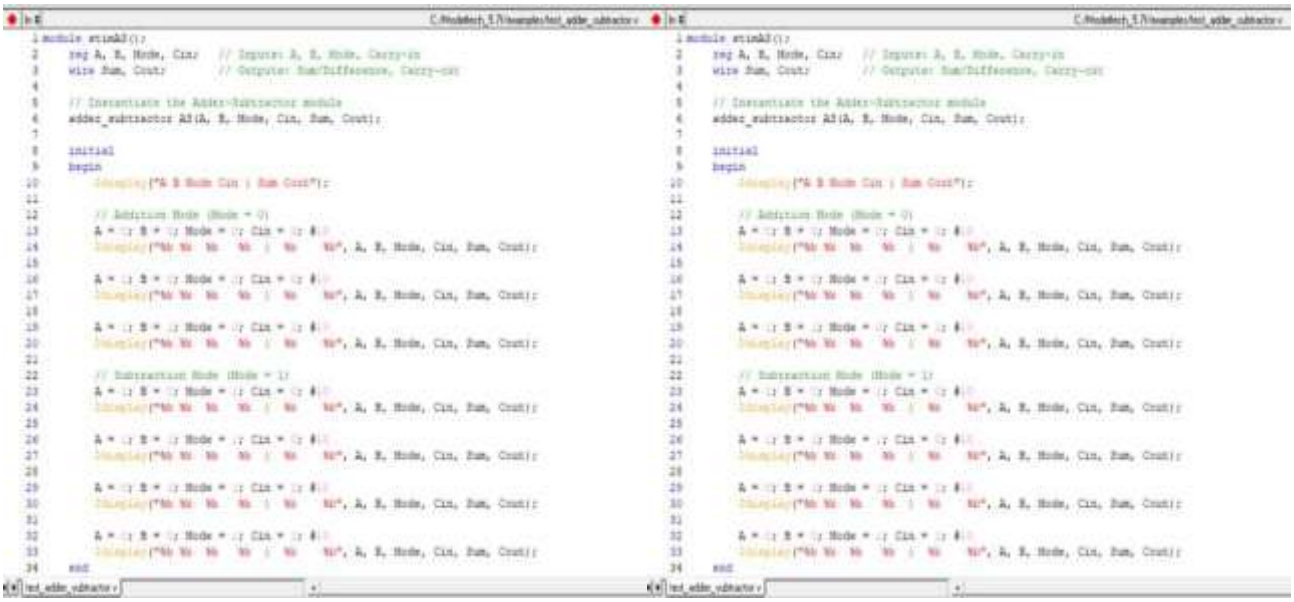
REMARKS:

The Verilog code for the Half Adder using Dataflow Level modeling is simple, clear, and correct. It utilizes assign statements to express the relationships between inputs and outputs, with the Sum calculated using the XOR operation (A ^ B) and the Cout using the AND operation (A & B). The code accurately models the behavior of a Half Adder, making it easy to understand and verify. However, for readability and completeness, it could include additional comments, especially for beginners, and adhere to consistent naming conventions if used in a larger design project.

TASK 3:

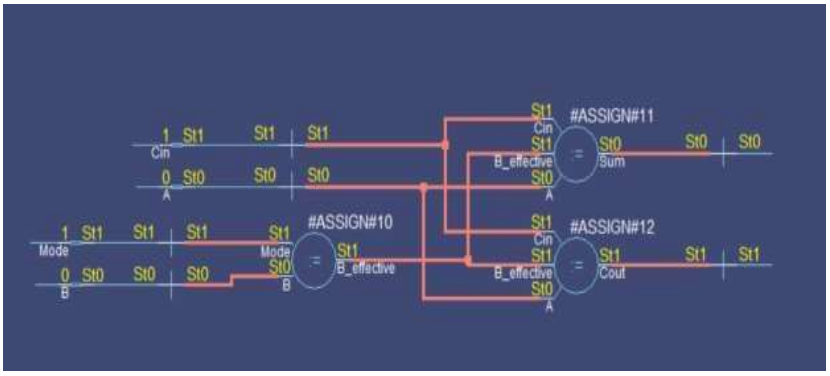
Write a Verilog code for Adder and Subtractor using Dataflow Level modeling.

Code:



Output:

Table:




```
run -all
#A B Mode Cin | Sum Cout
#0 0  0   0  | 0   0
#1 0  0   0  | 1   0
#1 1  0   1  | 1   1
#1 0  1   0  | 0   1
#1 1  1   0  | 1   0
#0 1  1   1  | 1   0
#0 0  1   1  | 0   1

VSIM 20>
```