LAB NO 13

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

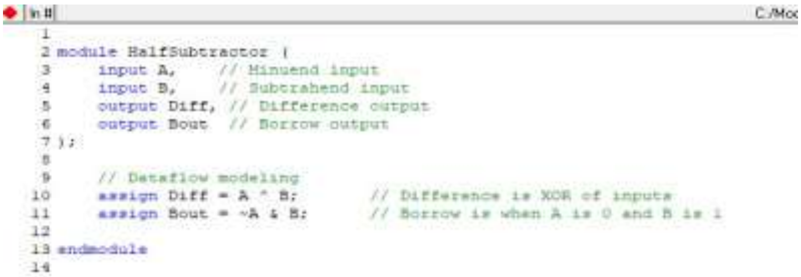
University of Engineering and Technology, Peshawar

SUBTRACTOR:

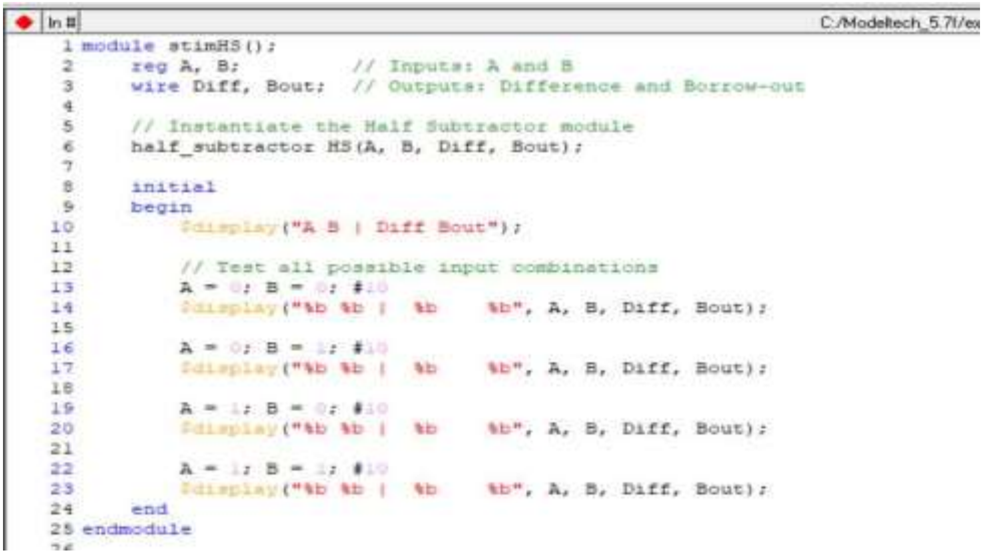
TASK:1

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

CODE:

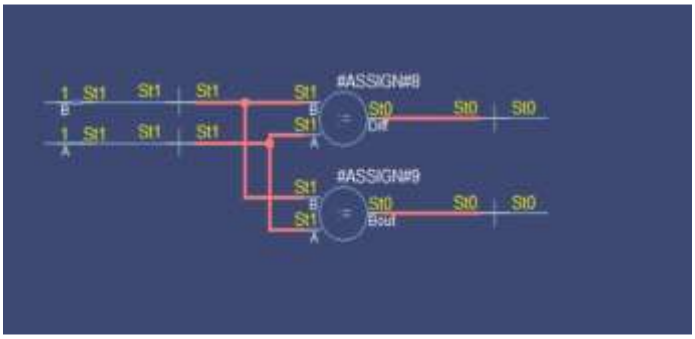


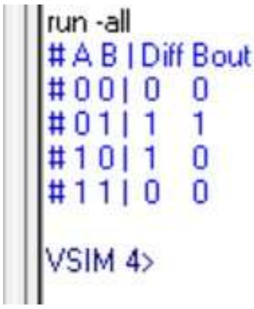
```verilog

module HalfSubtractor (
    input A,     // Minuend input
    input B,     // Subtrahend input
    output Diff, // Difference output
    output Bout  // Borrow output
);

    // Dataflow modeling
    assign Diff = A ^ B;        // Difference is XOR of inputs
    assign Bout = ~A & B;       // Borrow is when A is 0 and B is 1

endmodule

```



```verilog
module stimHS();
    reg A, B;          // Inputs: A and B
    wire Diff, Bout;  // Outputs: Difference and Borrow-out

    // Instantiate the Half Subtractor module
    half_subtractor HS(A, B, Diff, Bout);

    initial
    begin
        $display("A B | Diff Bout");

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

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

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

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

OUTPUT:





```
run -all
# A B | Diff Bout
# 0 0 | 0   0
# 0 1 | 1   1
# 1 0 | 1   0
# 1 1 | 0   0

VSIM 4>
```

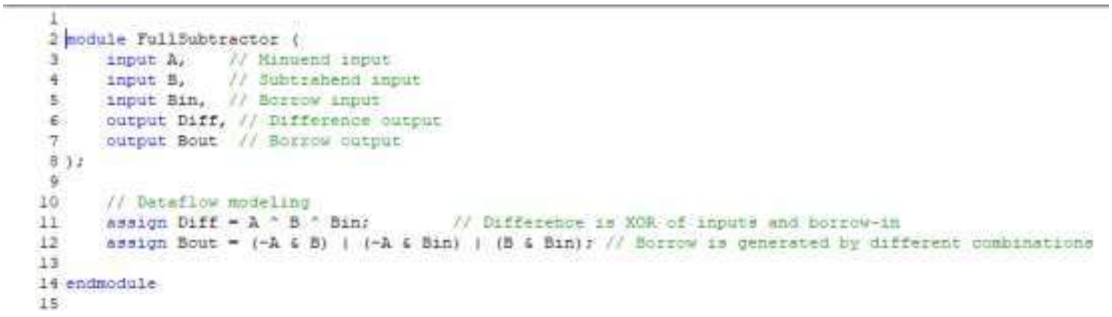
REMARKS:

The Verilog code for the Half Subtractor uses XOR for the Difference (A ^ B) and AND for the Borrow (~A & B), accurately modeling the subtraction logic. It's simple, clear, and efficient.

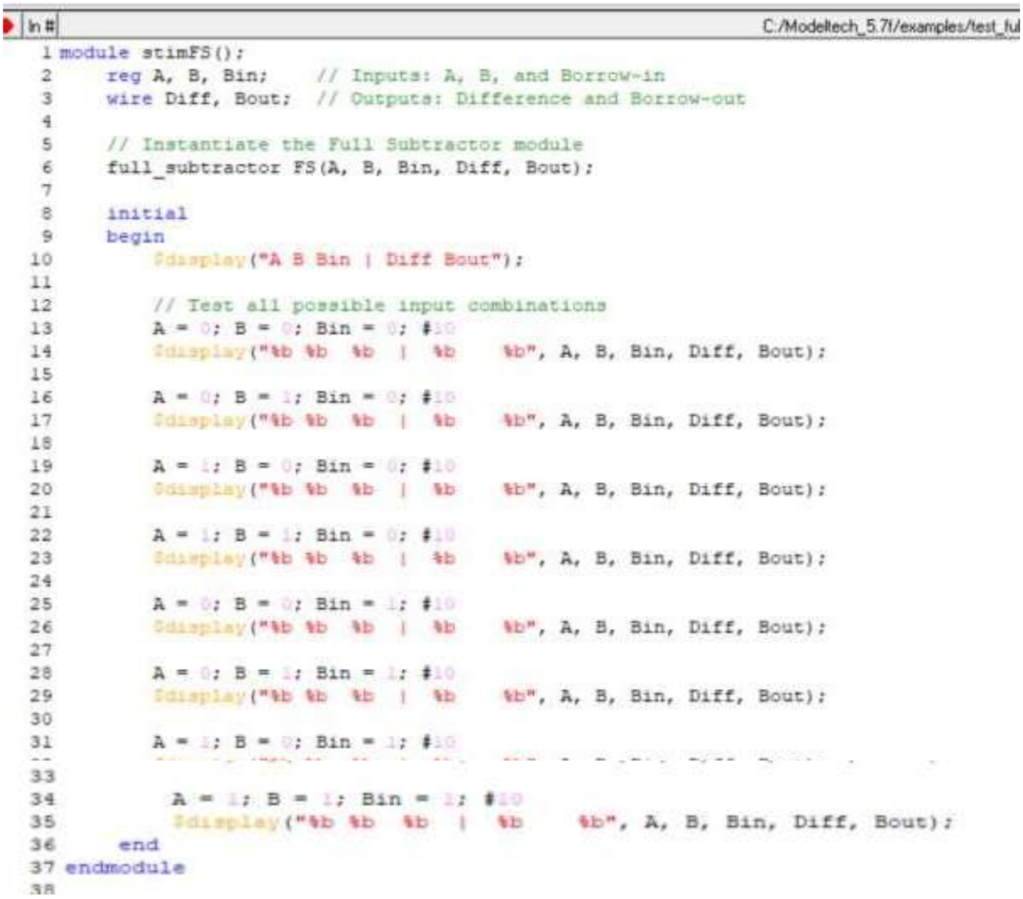
TASK:2

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

CODE:

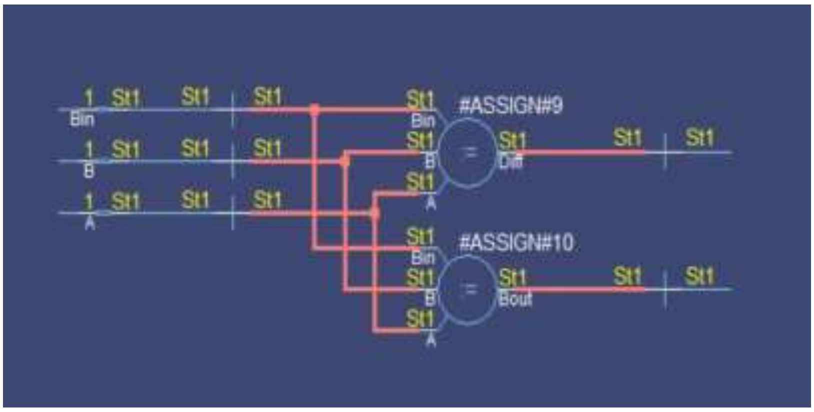


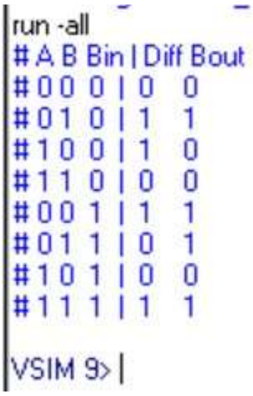
```

module FullSubtractor (
    input A,     // Minuend input
    input B,     // Subtrahend input
    input Bin,   // Borrow input
    output Diff, // Difference output
    output Bout  // Borrow output
);

    // Dataflow modeling
    assign Diff = A ^ B ^ Bin;          // Difference is XOR of inputs and borrow-in
    assign Bout = (~A & B) | (~A & Bin) | (B & Bin); // Borrow is generated by different combinations

endmodule

```



```
module stimFS();
    reg A, B, Bin;    // Inputs: A, B, and Borrow-in
    wire Diff, Bout;  // Outputs: Difference and Borrow-out

    // Instantiate the Full Subtractor module
    full_subtractor FS(A, B, Bin, Diff, Bout);

    initial
    begin
        $display("A B Bin | Diff Bout");

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

        A = 0; B = 1; Bin = 0; #10
        $display("%b %b  %b  |  %b    %b", A, B, Bin, Diff, Bout);

        A = 1; B = 0; Bin = 0; #10
        $display("%b %b  %b  |  %b    %b", A, B, Bin, Diff, Bout);

        A = 1; B = 1; Bin = 0; #10
        $display("%b %b  %b  |  %b    %b", A, B, Bin, Diff, Bout);

        A = 0; B = 0; Bin = 1; #10
        $display("%b %b  %b  |  %b    %b", A, B, Bin, Diff, Bout);

        A = 0; B = 1; Bin = 1; #10
        $display("%b %b  %b  |  %b    %b", A, B, Bin, Diff, Bout);

        A = 1; B = 0; Bin = 1; #10

         A = 1; B = 1; Bin = 1; #10
         $display("%b %b  %b  |  %b    %b", A, B, Bin, Diff, Bout);
     end
endmodule

```

OUTPUT:





```
run -all
# A B Bin | Diff Bout
# 0 0  0 | 0   0
# 0 1  0 | 1   1
# 1 0  0 | 1   0
# 1 1  0 | 0   0
# 0 0  1 | 1   1
# 0 1  1 | 0   1
# 1 0  1 | 0   0
# 1 1  1 | 1   1

VSIM 9> |
```

REMARKS:

The Verilog code for the Full Subtractor uses XOR to compute the Difference (A ^ B ^ Bin) and AND/OR operations for the Borrow (Bout). It's a compact and efficient implementation of the subtraction logic.