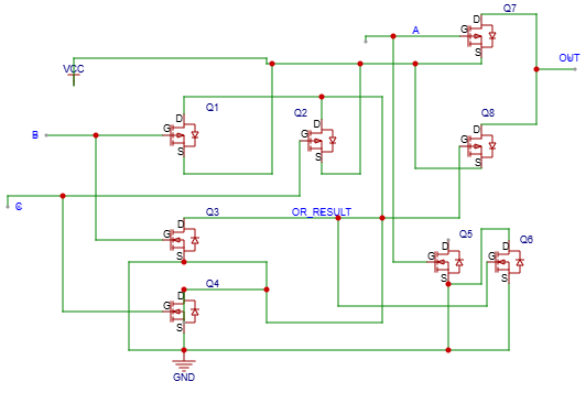
San Francisco Bay University
EE461L - Digital Design and HDL Lab
Quiz #1

Student Name: Anika Haque **Student ID: 20283**

1. **Design the module for a combinational gate by using *MOS* primitives based on the following logic expression, and then write the testbench to verify it.**

$$out = a(b + c)$$

a. Draw the circuit schematic in *cmos* and *pmos* first.



b. Write the Verilog module according to the above circuit.

```
module full_circuit(output out, input a, b, c);

    // OR Gate: Implements b + c
    wire or_result;         // Intermediate signal for the OR gate
output
    wire pmos_pullup_temp;  // Intermediate wire for the PMOS
network in the OR gate

    // NMOS Network: Pulls or_result low (0) when b = 1 or c = 1
    nmos n1(or_result, 1'b1, b);    // Pull down if b = 1
    nmos n2(or_result, 1'b1, c);    // Pull down if c = 1

    // PMOS Network: Pulls or_result high (1) when b = 0 or c = 0
    pmos p1(pmos_pullup_temp, 1'b0, b);  // Pull up temp if b = 0
    pmos p2(or_result, pmos_pullup_temp, c); // Pull up output if
temp = 1 and c = 0

    // AND Gate: Implements a · (b + c)
```

```
    wire temp; // Intermediate signal for the NMOS network in the
AND gate

    // PMOS Network: Pulls out high (1) when a = 0 or or_result = 0
    pmos n3(out, 1'b0, a);          // Pull up if a = 0
    pmos n4(out, 1'b0, or_result);  // Pull up if or_result = 0

    // NMOS Network: Pulls out low (0) when both a = 1 and or_result
= 1
    nmos p3(temp, 1'b1, a);         // Pull down temp if a = 1
    nmos p4(out, temp, or_result);  // Pull down output if temp = 0
and or_result = 1

endmodule
```

c. Create the testbench to verify it and show the waveforms.

Testbench

```
module full_circuit_tb;

    reg a, b, c;         // Input signals
    wire or_result;      // Intermediate result from the OR gate
    wire out;            // Final output from the AND gate

    // Instantiate the full circuit
    full_circuit uut (
        .out(out),
        .a(a),
        .b(b),
        .c(c)
    );

    // Initial block for stimuli and waveform
    initial begin
        $dumpfile("waveform.vcd");
        $dumpvars(0, full_circuit_tb);

        // Monitor the intermediate OR gate result and final AND
gate result
        $monitor("Time = %0t | a = %b, b = %b, c = %b | or_result
= %b | out = %b",
                 $time, a, b, c, uut.or_result, out);

        // Test cases
        a = 0; b = 0; c = 0; #10; // Expect or_result = 0, out = 0
        a = 0; b = 0; c = 1; #10; // Expect or_result = 1, out = 0
        a = 0; b = 1; c = 0; #10; // Expect or_result = 1, out = 0
        a = 0; b = 1; c = 1; #10; // Expect or_result = 1, out = 0
        a = 1; b = 0; c = 0; #10; // Expect or_result = 0, out = 0
        a = 1; b = 0; c = 1; #10; // Expect or_result = 1, out = 1
        a = 1; b = 1; c = 0; #10; // Expect or_result = 1, out = 1
        a = 1; b = 1; c = 1; #10; // Expect or_result = 1, out = 1
```

```
        $finish;
    end

endmodule
```

Simulation Output

testbench.sv

```
module full_circuit_tb;

    reg a, b, c;        // Input signals
    wire or_result;     // Intermediate result from the OR gate
    wire out;           // Final output from the AND gate

    // Instantiate the full circuit
    full_circuit uut (
        .out(out),
        .a(a),
        .b(b),
        .c(c)
    );

    // Initial block for stimuli and waveform
    initial begin
        $dumpfile("waveform.vcd");
        $dumpvars(0, full_circuit_tb);

        // Monitor the intermediate OR gate result and final AND gate result
        $monitor("Time = %0t | a = %b, b = %b, c = %b | or_result = %b | out = %b",
                 $time, a, b, c, uut.or_result, out);

        // Test cases
        a = 0; b = 0; c = 0; #10; // Expect or_result = 0, out = 0
        a = 0; b = 0; c = 1; #10; // Expect or_result = 1, out = 0
        a = 0; b = 1; c = 0; #10; // Expect or_result = 1, out = 0
        a = 0; b = 1; c = 1; #10; // Expect or_result = 1, out = 0
        a = 1; b = 0; c = 0; #10; // Expect or_result = 0, out = 0
        a = 1; b = 0; c = 1; #10; // Expect or_result = 1, out = 1
        a = 1; b = 1; c = 0; #10; // Expect or_result = 1, out = 1
        a = 1; b = 1; c = 1; #10; // Expect or_result = 1, out = 1

        $finish;
    end

endmodule
```

design.sv

```
module full_circuit(output out, input a, b, c);

    // OR Gate: Implements b + c
    wire or_result;         // Intermediate signal for the OR gate output
    wire pmos_pullup_temp;  // Intermediate wire for the PMOS network in the OR gate

    // NMOS Network: Pulls or_result low (0) when b = 1 or c = 1
    nmos n1(or_result, 1'b1, b);    // Pull down if b = 1
    nmos n2(or_result, 1'b1, c);    // Pull down if c = 1

    // PMOS Network: Pulls or_result high (1) when b = 0 or c = 0
    pmos p1(pmos_pullup_temp, 1'b0, b);  // Pull up temp if b = 0
    pmos p2(or_result, pmos_pullup_temp, c); // Pull up output if temp = 1 and c = 0

    // AND Gate: Implements a · (b + c)
    wire temp; // Intermediate signal for the NMOS network in the AND gate

    // PMOS Network: Pulls out high (1) when a = 0 or or_result = 0
    pmos n3(out, 1'b0, a);          // Pull up if a = 0
    pmos n4(out, 1'b0, or_result);  // Pull up if or_result = 0

    // NMOS Network: Pulls out low (0) when both a = 1 and or_result = 1
    nmos p3(temp, 1'b1, a);         // Pull down temp if a = 1
    nmos p4(out, temp, or_result);  // Pull down output if temp = 0 and or_result = 1

endmodule
```

Log | Share

```
[2024-12-12 15:07:47 UTC] iverilog '-Wall' '-g2012' design.sv testbench.sv  && unbuffer vvp a.out
VCD info: dumpfile waveform.vcd opened for output.
Time = 0 | a = 0, b = 0, c = 0 | or_result = 0 | out = 0
Time = 10 | a = 0, b = 0, c = 1 | or_result = 1 | out = 0
Time = 20 | a = 0, b = 1, c = 0 | or_result = 1 | out = 0
Time = 30 | a = 0, b = 1, c = 1 | or_result = 1 | out = 0
Time = 40 | a = 1, b = 0, c = 0 | or_result = 0 | out = 0
Time = 50 | a = 1, b = 0, c = 1 | or_result = 1 | out = 1
Time = 60 | a = 1, b = 1, c = 0 | or_result = 1 | out = 1
Time = 70 | a = 1, b = 1, c = 1 | or_result = 1 | out = 1
testbench.sv:34: $finish called at 80 (1s)
Done
```

Waveform