Switching Theory Lab – CS226

Name: M. Maheeth Reddy **Date**: 09-May-2020

Roll No.: 1801CS31

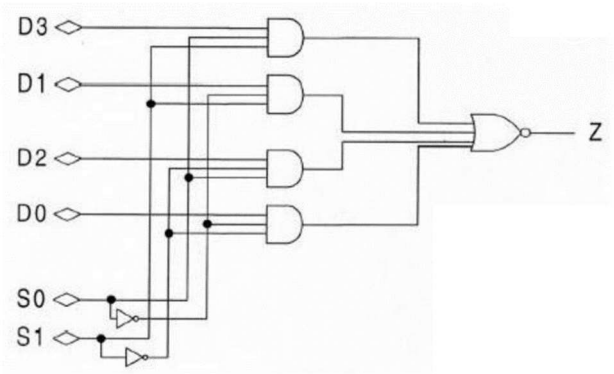
Lab No.: 12

Ans 1:

Source Code

```
// Model for given Logic Function
module p1(z,d3,d2,d1,d0,s1,s0);

    // Variable Declaration
    output z;
    input d3,d2,d1,d0;
    input s1,s0;

    wire _s0, _s1, z0, z1, z2, z3;

    // Inverted Select Lines
    not n0 (_s0,s0);
    not n1 (_s1,s1);

    // Generating Inputs for NOR gate
    and a0 (z0, _s1, _s0, d0);
    and a1 (z1, _s1,  s0, d1);
    and a2 (z2,  s1, _s0, d2);
    and a3 (z3,  s1,  s0, d3);

    // Final Output z
    nor nor_0(z,z0,z1,z2,z3);

endmodule
```

Circuit Diagram



Test Bench

```
module tb_p1();
    reg d3,d2,d1,d0,s1,s0;
    wire z;
    p1 UUT (z,d3,d2,d1,d0,s1,s0);
    initial begin
        d0 = 1'b0;
        d1 = 1'b1;
        d2 = 1'b0;
        d3 = 1'b1;
        s1 = 1'b0;
        s0 = 1'b0;
        #10;
        s1 = 1'b0;
        s0 = 1'b0;
        d0 = 1;
        #10;
        s1 = 1'b0;
        s0 = 1'b1;
        #10;
        s1 = 1'b0;
        s0 = 1'b1;
        d1 = 0;
        #10;
        s1 = 1'b1;
        s0 = 1'b0;
        #10;
        s1 = 1'b1;
        s0 = 1'b0;
        d2 = 1;
        #10;
        s1 = 1'b1;
        s0 = 1'b1;
        #10;
        s1 = 1'b1;
        s0 = 1'b1;
        d3 = 0;
    end
    initial begin
        $monitor("d3=%b d2=%b d1=%b d0=%b s1=%b s0=%b z=%b",d3,d2,d1,d0,s1,s0,z);
    end
endmodule
```

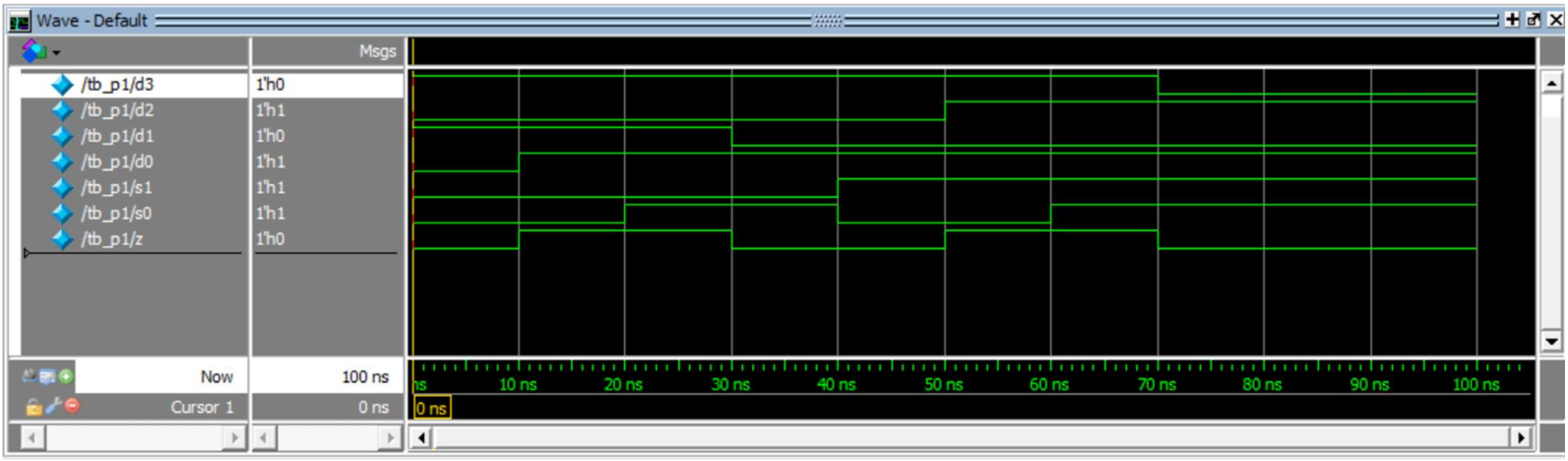
Text Output

```
VSIM 36> run
# d3=1 d2=0 d1=1 d0=0 s1=0 s0=0 z=0
# d3=1 d2=0 d1=1 d0=1 s1=0 s0=0 z=1
# d3=1 d2=0 d1=1 d0=1 s1=0 s0=1 z=1
# d3=1 d2=0 d1=0 d0=1 s1=0 s0=1 z=0
# d3=1 d2=0 d1=0 d0=1 s1=1 s0=0 z=0
# d3=1 d2=1 d1=0 d0=1 s1=1 s0=0 z=1
# d3=1 d2=1 d1=0 d0=1 s1=1 s0=1 z=1
# d3=0 d2=1 d1=0 d0=1 s1=1 s0=1 z=0
```

Observations

1. For S1 = 0, S0 = 0, Output is affected only by changing D0.
2. For S1 = 0, S0 = 1, Output is affected only by changing D1.
3. For S1 = 1, S0 = 0, Output is affected only by changing D2.
4. For S1 = 1, S0 = 1, Output is affected only by changing D3.

Simulation Wavefront



Ans 2:

Source Code:

```
// Design for Fuel Level Detector Model
module p2(L,b);
    output L;
    /* Indicator Light --> Glows when there is low fuel
     *                     i.e., when level is less than 3 */
    input [2:0] b;  // Fuel Level (3-bit binary)
    wire w;
    /* Final Expression = ~b2.~b1 + ~b2.~b0
                        = ~b2.(~b1 + ~b0)
                        = ~b2.~(b1.b0)
                        = ~(b2 + b1.b0)
                        = nor(b2, b1.b0)
    */
    and and_0(w, b[1], b[0]);   // w = b1.b0
    nor nor_0(L, b[2], w);  // L = nor(b2, b1.b0)
endmodule
```

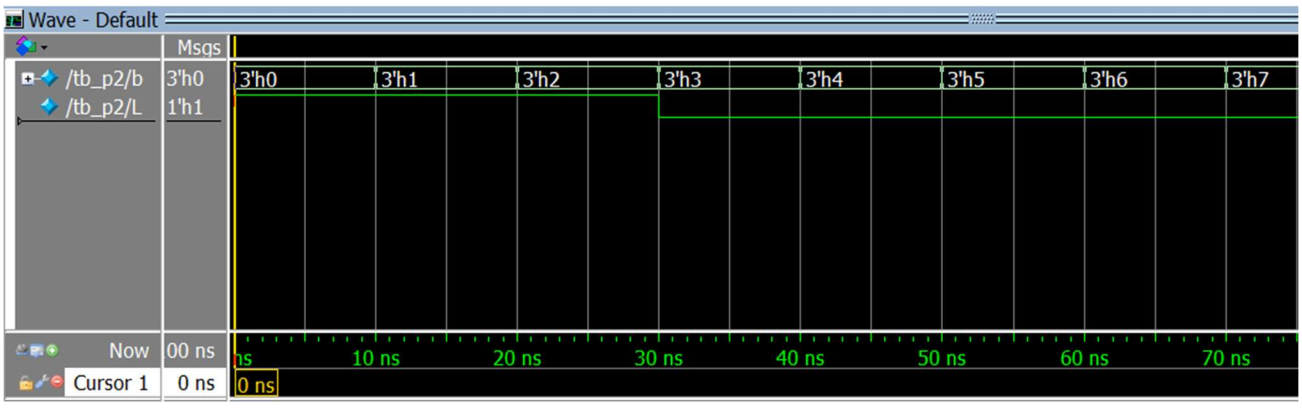
Test Bench:

```
module tb_p2();
    reg [2:0] b;
    wire L;
    p2 UUT (L,b);
    initial begin
        b = 3'b000;
        #10;
        b = 3'b001;
        #10;
        b = 3'b010;
        #10;
        b = 3'b011;
        #10;
        b = 3'b100;
        #10;
        b = 3'b101;
        #10;
        b = 3'b110;
        #10;
        b = 3'b111;
    end
    initial begin
     $monitor("b=%b %b %b, L=%b, time=%t\n", b[2], b[1], b[0], L, $time);
    end
endmodule
```

Text Output

```
VSIM 4> run
# b=0 0 0, L=1, time=                    0
#
# b=0 0 1, L=1, time=                   10
#
# b=0 1 0, L=1, time=                   20
#
# b=0 1 1, L=0, time=                   30
#
# b=1 0 0, L=0, time=                   40
#
# b=1 0 1, L=0, time=                   50
#
# b=1 1 0, L=0, time=                   60
#
# b=1 1 1, L=0, time=                   70
#
```

Simulation Wavefront



Observations:

1. Whenever the fuel level is 0 or 1 or 2 i.e., less than 3, the indicator should glow.
2. Therefore, if $b_2b_1b_0$ represents fuel level, $L = b_2' \& (b_1' + b_0')$

Ans 3:

Source Code:

```
module p3(f,g,h,a,b,c);

    // Variable Declaration
    input a,b,c;
    output f,g,h;

    // Boolean Functions
    assign f = (a & ~b) | (~b & ~c) | (a & c);
    assign g = (~b | c) & (a | b | ~c);
    assign h = (~b & ~c) | (b & c) | (a & c);
endmodule
```

Test Bench:

```
module tb_p3();
    reg a,b,c;
    wire f,g,h;
    p3 UUT (f,g,h,a,b,c);
    initial begin
        a = 0;
        b = 0;
        c = 0;
        while ((a & b & c) != 1) begin
            #10;
            a = a^(b&c);
            b = b^c;
            c = c^1;
        end
    end
    initial begin

        $monitor("a=%b b=%b c=%b f=%b g=%b h=%b",a,b,c,f,g,h);
    end
endmodule
```

Given Information

f = ab'+b'c'+ac
g = (b'+c)(a+b+c')
h = b'c'+bc+ac

Text Output

```
VSIM 50> run
# a=0 b=0 c=0 f=1 g=1 h=1
# a=0 b=0 c=1 f=0 g=0 h=0
# a=0 b=1 c=0 f=0 g=0 h=0
# a=0 b=1 c=1 f=0 g=1 h=1
# a=1 b=0 c=0 f=1 g=1 h=1
# a=1 b=0 c=1 f=1 g=1 h=1
# a=1 b=1 c=0 f=0 g=0 h=0
# a=1 b=1 c=1 f=1 g=1 h=1
```

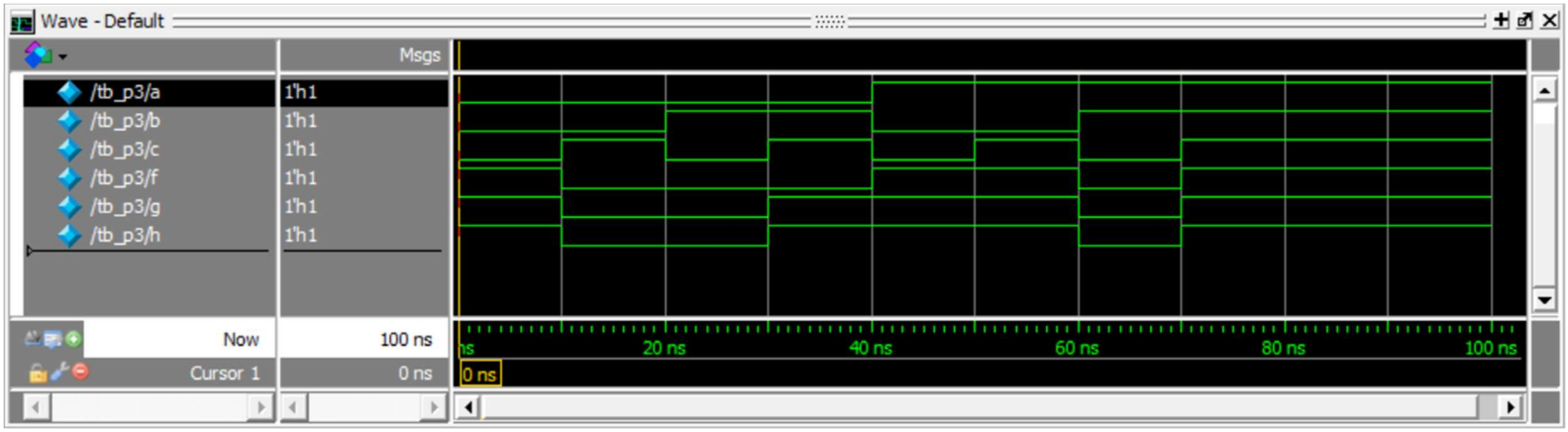
Truth Table

a	b	c	f	g	h
0	0	0	1	1	1
0	0	1	0	0	0
0	1	0	0	0	0
0	**1**	**1**	**0**	**1**	**1**
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	1	1	1

Observations:

1. For **a = 0, b = 1, c = 1**, the value of **f is 0**, whereas the value of **g is 1 and h is 1**.
2. Hence, all the given functions are **not equal**.

Simulation Wavefront



Ans 4:

Source Code:

```
module comparators(eq,neq,lt,lte,gt,gte,a,b);
parameter N = 8;
input [N-1:0] a, b;
output eq, neq;
output lt, lte;
output gt, gte;
assign eq = (a == b);
assign neq = (a != b);
assign lt = (a < b);
assign lte = (a <= b);
assign gt = (a > b);
assign gte = (a >= b);
endmodule
```

Test Bench:

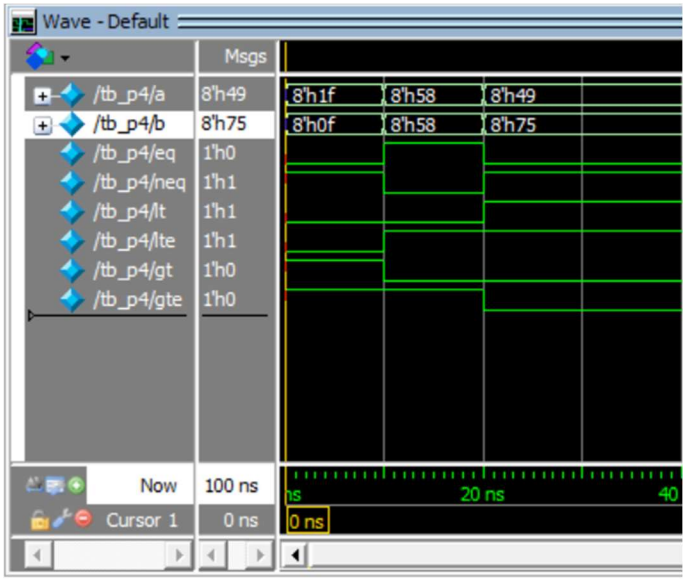
```
module tb_p4();
    parameter N = 8;
    reg [N-1:0] a, b;
    wire eq, neq, lt, lte, gt, gte;
    comparators uut(eq, neq, lt, lte, gt, gte, a, b);
    initial begin
        a = 8'b0001_1111; // DEC 31
        b = 8'b0000_1111; // DEC 15
        #10;
```

Text Output

```
VSIM 66> run
# For a = 00011111, and b = 00001111
#       eq = 0
#       neq = 1
#       lt = 0
#       lte = 0
#       gt = 1
#       gte = 1
#
# For a = 01011000, and b = 01011000
#       eq = 1
#       neq = 0
#       lt = 0
#       lte = 1
#       gt = 0
#       gte = 1
#
# For a = 01001001, and b = 01110101
#       eq = 0
#       neq = 1
#       lt = 1
#       lte = 1
#       gt = 0
#       gte = 0
#
```

```
        a = 8'b0101_1000; // DEC 88
        b = 8'b0101_1000; // DEC 88
        #10;
        a = 8'b0100_1001; // DEC 73
        b = 8'b0111_0101; // DEC 117
    end
    initial begin
        $monitor("For a = %b, and b = %b\n\te
q = %b\n\tneq = %b\n\tlt = %b\n\tlte = %b\n\t
gt = %b\n\tgte = %b\n",a,b,eq,neq,lt,lte,gt,g
te);
    end
endmodule
```

Simulated Wavefront



Ans 5:

Source Code:

```
/* Verilog Model for
   Alarm in Mueseum */
module p5(A,m0,m1,m2);

    input m0,m1,m2; // motion sensor in each room
    output A;       // alarm

    /* alarm sounds when there is
     motion in more than one room at a time */
    assign A = (m0 & m1) | (m1 & m2) | (m2 & m0);
endmodule
```

Test Bench:

```
module tb_p5();
    reg m2,m1,m0;   // motion sensors
    wire A;         // Alarm
    p5 UUT (A,m2,m1,m0);
    initial begin
        m2 = 0;
        m1 = 0;
        m0 = 0;

        while ((m0 & m1 & m2) != 1) begin
```

Text Output

```
VSIM 78> run
# m2=0 m1=0 m0=0 A=0
# m2=0 m1=0 m0=1 A=0
# m2=0 m1=1 m0=0 A=0
# m2=0 m1=1 m0=1 A=1
# m2=1 m1=0 m0=0 A=0
# m2=1 m1=0 m0=1 A=1
# m2=1 m1=1 m0=0 A=1
# m2=1 m1=1 m0=1 A=1
```

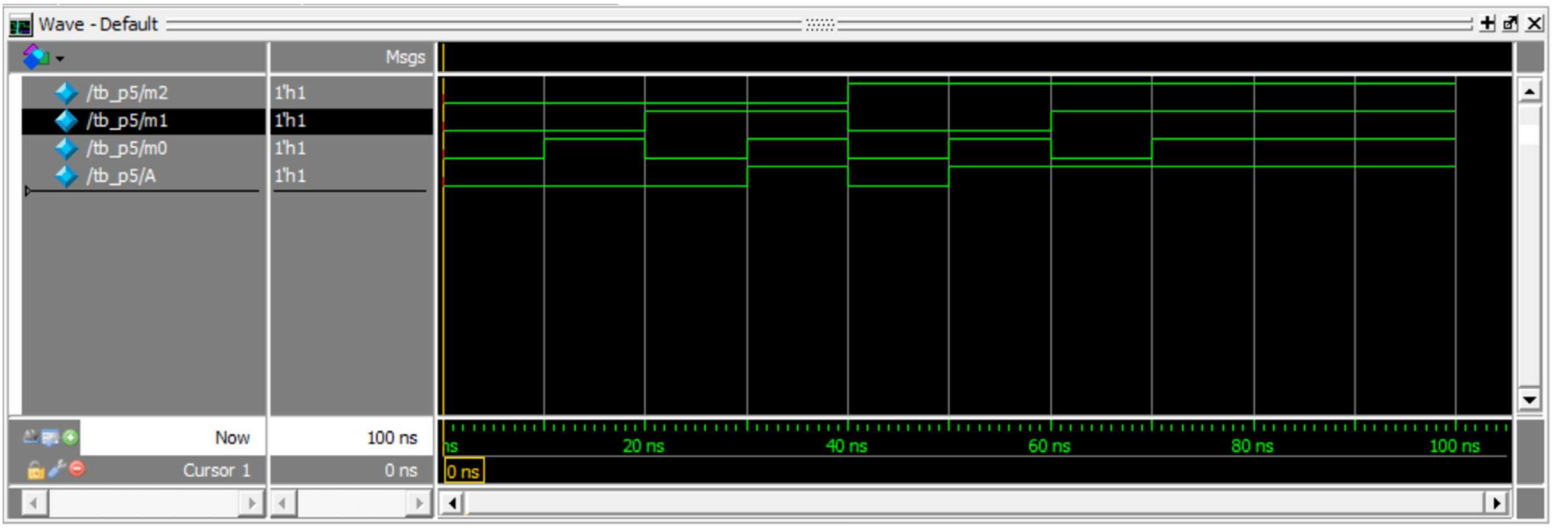
Observations

Alarm(A) sounds when there is motion in more than one room. Hence,

A = m0.m1 + m1.m2 + m2.m0

```
            #10;
            m2 = m2^(m1 & m0);
            m1 = m1^m0;
            m0 = m0^1;
        end
    end
    initial begin
        $monitor("m2=%b m1=%b m0=%b A=%b",m2,m1,m0,A);
    end
endmodule
```

Simulated Wavefront



Ans 6:

Source Code:

```
// Structural Description for given schematic
module p6(op1, op2, A, B, C);
    input A,B,C;    // Inputs
    output op1,op2; // Outputs
    /*  x1 */ nor U2a(x1,A,B);
    /*  x2 */ nand U1b(x2,A,B);
    /*  x3 */ not U6(x3,x2);
    /*  x4 */ not U3(x4,C);
```

```
    /*  x5 */ nor U2b(x5,x1,x3);
    /*  x6 */ not U8(x6,x4);
    /*  x7 */ not U9(x7,x6);
    /*  x8 */ or U5(x8,x1,x7);
    /*  x9 */ xnor U4(x9,x5,x6);
    /* op1 */ nand U1a(op1,x2,x8);
    /* op2 */ not U7(op2,x9);
endmodule
```

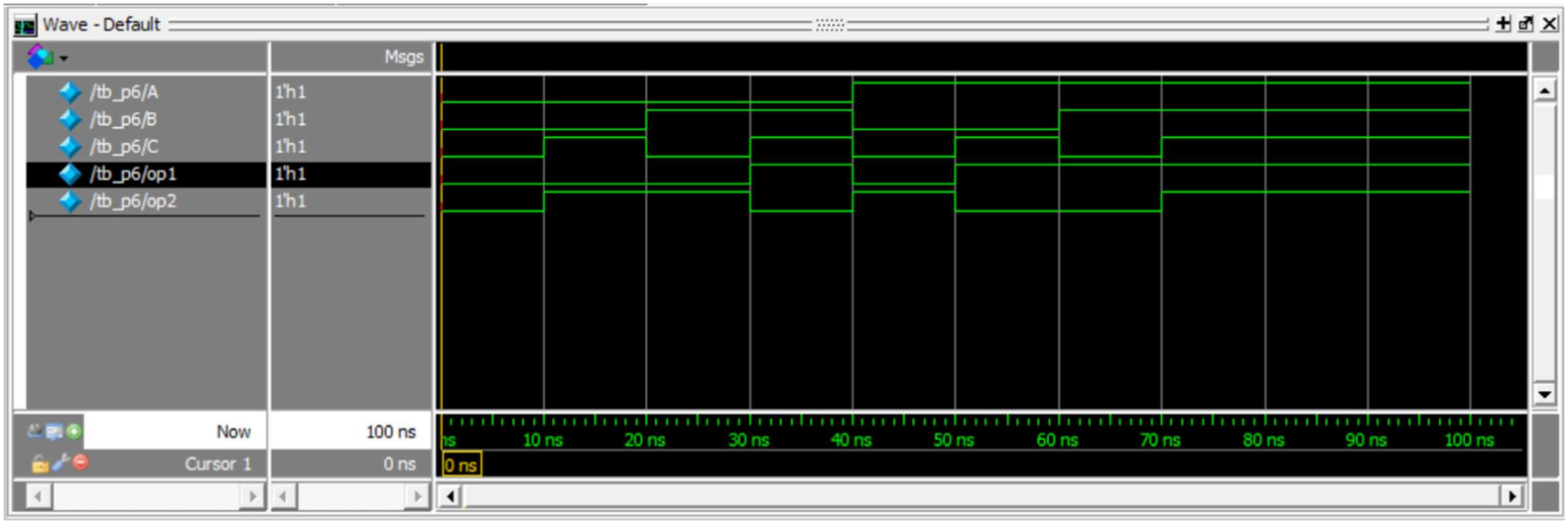
Test Bench:

```
module tb_p6();
    reg A,B,C;      // Inputs
    wire op1,op2;   // Outputs
    p6 UUT (op1,op2,A,B,C);
    initial begin
        A = 0;
        B = 0;
        C = 0;
        while ((A & B & C) != 1) begin
            #10;
            A = A^(B & C);
            B = B^C;
            C = C^1  end
        $monitor("A=%b B=%b C=%b op1=%b op2=%b",A,B,C,op1,op2);
    end
    endmodule
```

Text Output

```
VSIM 81> run
# A=0 B=0 C=0 op1=0 op2=0
# A=0 B=0 C=1 op1=0 op2=1
# A=0 B=1 C=0 op1=0 op2=1
# A=0 B=1 C=1 op1=1 op2=0
# A=1 B=0 C=0 op1=0 op2=1
# A=1 B=0 C=1 op1=1 op2=0
# A=1 B=1 C=0 op1=1 op2=0
# A=1 B=1 C=1 op1=1 op2=1
```

Simulated Wavefront



Ans 7:

Source Code:

```
// Style 1
module mux1(op,s1,s0,d3,d2,d1,d0);
    // input s1,s0;        // Select Lines
    input s1,s0,d3,d2,d1,d0;  // Inputs
    output op;          // Output

    and (op0,~s1,~s0,d0);     // Line 0
    and (op1,~s1,s0,d1);      // Line 1
    and (op2,s1,~s0,d2);      // Line 2
    and (op3,s1,s0,d3);       // Line 3
    or (op,op0,op1,op2,op3);  // Output
endmodule

// Style 2
module mux2(op,s1,s0,d3,d2,d1,d0);
    // input s1,s0;        // Select Lines
    input s1,s0,d3,d2,d1,d0;  // Inputs
    output op;          // Output

    and (op0,~s1,~s0,d0);               // Line 0
    and (op1,~s1,s0,d1);                // Line 1
    and (op2,s1,~s0,d2);                // Line 2
    and (op3,s1,s0,d3);                 // Line 3
    assign op = (op0|op1|op2|op3);      // Output
endmodule

// Style 3
module mux3(op,s1,s0,d3,d2,d1,d0);
    // input s1,s0;        // Select Lines
    input s1,s0,d3,d2,d1,d0;  // Inputs
    output op;          // Output
    assign op = (~s1 & ~s0 & d0) | (~s1 & s0 & d1) | (s1 & ~s0 & d2) | (s1 & s0 &
 d3);
endmodule

// Style 4
module mux4(op,s1,s0,d3,d2,d1,d0);
            // Select Lines
    input s1,s0,d3,d2,d1,d0;  // Inputs
    output op;          // Output
    assign op = s1 ? (s0 ? d3 : d2) : (s0 ? d1 : d0);
```

```
endmodule

// Style 5
module mux5(op,s1,s0,d3,d2,d1,d0);
    // input s1,s0;        // Select Lines
    input s1,s0,d3,d2,d1,d0;  // Inputs
    output op;          // Output
    reg op;

    always @(s1 or s0 or d3 or d2 or d1 or d0)
    begin
        if(s1==0 & s0==0)
            assign op = d0;
        else if(s1==0 & s0==1)
            assign op = d1;
        else if(s1==1 & s0==0)
            assign op = d2;
        else
            assign op = d3;
    end
endmodule
```

Test Bench:

```
module tb_p7();
  reg s1, s0, d3, d2, d1, d0;
  wire op1,op2,op3,op4,op5;
  mux1 uut1(op1, s1, s0, d3, d2, d1, d0);
  mux2 uut2(op2, s1, s0, d3, d2, d1, d0);
  mux3 uut3(op3, s1, s0, d3, d2, d1, d0);
  mux4 uut4(op4, s1, s0, d3, d2, d1, d0);
  mux5 uut5(op5, s1, s0, d3, d2, d1, d0);
  initial
    begin
        d0 = 0;
        d1 = 0;
        d2 = 0;
        d3 = 0;

        s1 = 0;
        s0 = 0;
        #10;
        d0 = 1;
        #10;
```

Text Output

```
VSIM 75> run
# d3=0 d2=0 dl=0 d0=0 sl=0 s0=0
# opl=0 op2=0 op3=0 op4=0 op5=0
# --------
# d3=0 d2=0 dl=0 d0=1 sl=0 s0=0
# opl=1 op2=1 op3=1 op4=1 op5=1
# --------
# d3=0 d2=0 dl=0 d0=1 sl=0 s0=1
# opl=0 op2=0 op3=0 op4=0 op5=0
# --------
# d3=0 d2=0 dl=1 d0=1 sl=0 s0=1
# opl=1 op2=1 op3=1 op4=1 op5=1
# --------
# d3=0 d2=0 dl=1 d0=1 sl=1 s0=0
# opl=0 op2=0 op3=0 op4=0 op5=0
# --------
# d3=0 d2=1 dl=1 d0=1 sl=1 s0=0
# opl=1 op2=1 op3=1 op4=1 op5=1
# --------
# d3=0 d2=1 dl=1 d0=1 sl=1 s0=1
# opl=0 op2=0 op3=0 op4=0 op5=0
# --------
# d3=1 d2=1 dl=1 d0=1 sl=1 s0=1
# opl=1 op2=1 op3=1 op4=1 op5=1
# --------
```

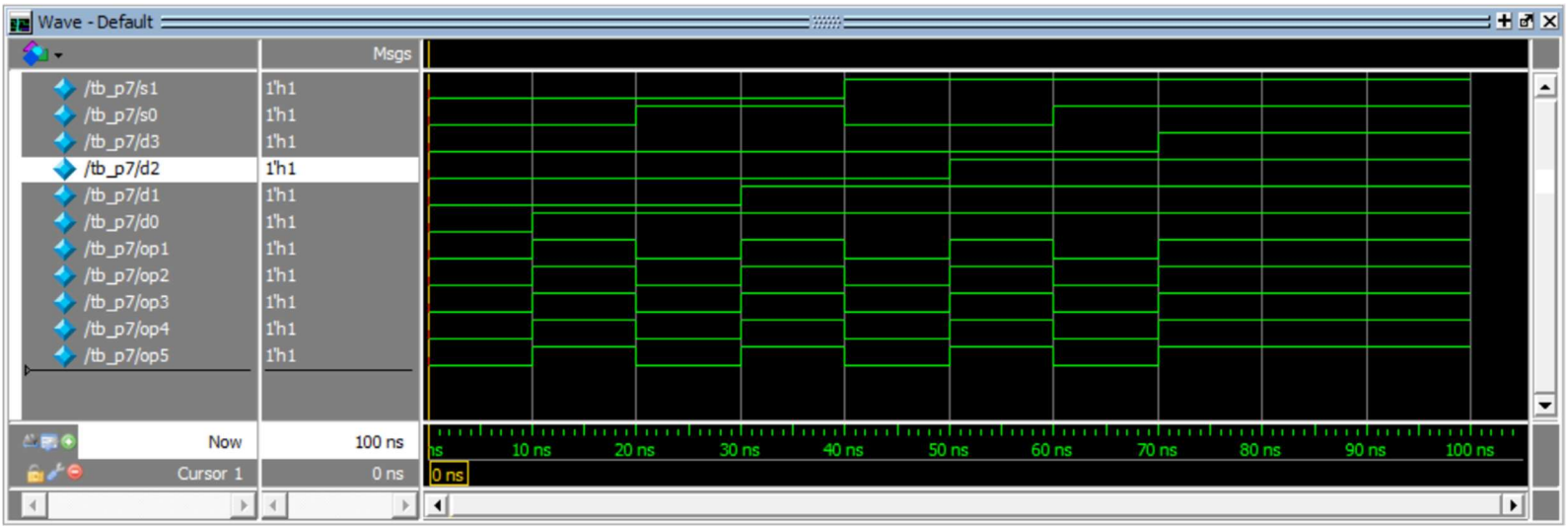
Observations

For all 5 different MUXes,

1. For S1 = 0, S0 = 0, Output is equal to D0.
2. For S1 = 0, S0 = 1, Output is equal to D1.
3. For S1 = 1, S0 = 0, Output is equal to D2.
4. For S1 = 1, S0 = 1, Output is equal to D3.

```
        s1 = 0;
        s0 = 1;
        #10
        d1 = 1;
        #10;
        s1 = 1;
        s0 = 0;
        #10;
        d2 = 1;
        #10;
        s1 = 1;
        s0 = 1;
        #10;
        d3 = 1;
    end
    initial begin
        $monitor("d3=%b d2=%b d1=%b d0=%b s1=%b s0=%b\nop1=%b op2=%b op3=%b op4=%
b op5=%b\n--------",d3,d2,d1,d0,s1,s0,op1,op2,op3,op4,op5);
    end
endmodule
```

Simulated Wavefront



Ans 8:

Source Code:

```
// 4 : 1 Multiplexer
module MUX_4to1(op,s1,s0,d3,d2,d1,d0);
    input s1,s0;        // select lines
    input d3,d2,d1,d0;  // inputs
    output op;          // output

    and (line0,~s1,~s0,d0);
    and (line1,~s1,s0,d1);
    and (line2,s1,~s0,d2);
    and (line3,s1,s0,d3);
    or (op,line0,line1,line2,line3);
endmodule

// 16 : 1 Multiplexer
module p8(op,s3,s2,s1,s0,d15,d14,d13,d12,d11,d10,d9,d8,d7,d6,d5,d4,d3,d2,d1,d0);
    input s3,s2,s1,s0;                      // select lines
    input d7,d6,d5,d4,d3,d2,d1,d0;          // inputs 0-7
    input d15,d14,d13,d12,d11,d10,d9,d8;    // inputs 8-15
    output op;                              // output

    // We use 4 MUX's to provide 16 input lines
    // Connect each of their output lines to 5th MUX
    // s1, s0 are select lines for 4 MUX's
    // s3, s2 are select lines for 5th MUX
    MUX_4to1 mux0 (line0,s1,s0,d3,d2,d1,d0);
    MUX_4to1 mux1 (line1,s1,s0,d7,d6,d5,d4);
    MUX_4to1 mux2 (line2,s1,s0,d11,d10,d9,d8);
    MUX_4to1 mux3 (line3,s1,s0,d15,d14,d13,d12);
    MUX_4to1 muxF (op,s3,s2,line3,line2,line1,line0);
endmodule
```

Test Bench:

```
module tb_p8();
    reg s3,s2,s1,s0;                        //Select Lines
    reg d7,d6,d5,d4,d3,d2,d1,d0;            // Inputs 0-7
    reg d15,d14,d13,d12,d11,d10,d9,d8;      // Inputs 8-15
    wire op;                                // Output

    // Instantiate 16:1 MUX
    p8 UUT(op,s3,s2,s1,s0,d15,d14,d13,d12,d11,d10,d9,d8,d7,d6,d5,d4,d3,d2,d1,d0);
```

```
initial begin
    // Initialize Inputs
    d0 = 0;
    d1 = 0;
    d2 = 0;
    d3 = 0;
    d4 = 0;
    d5 = 0;
    d6 = 0;
    d7 = 0;
    d8 = 0;
    d9 = 0;
    d10 = 0;
    d11 = 0;
    d12 = 0;
    d13 = 0;
    d14 = 0;
    d15 = 0;

    s0 = 0;
    s1 = 0;
    s2 = 0;
    s3 = 0;
    #10;
    d0 = 1;

    #10;
    s0 = 1;
    s1 = 0;
    s2 = 0;
    s3 = 0;
    #10;
    d1 = 1;

    #10;
    s0 = 0;
    s1 = 1;
    s2 = 0;
    s3 = 0;
    #10;
    d2 = 1;

    #10;
    s0 = 1;
```

Text Output (Part 1)

```
# s1=0 s0=0
#        d0=0 d1=0 d2=0 d3=0
#        d4=0 d5=0 d6=0 d7=0
#        d8=0 d9=0 d10=0 d11=0
#        d12=0 d13=0 d14=0 d15=0
# op=0
# s1=0 s0=0
#        d0=1 d1=0 d2=0 d3=0
#        d4=0 d5=0 d6=0 d7=0
#        d8=0 d9=0 d10=0 d11=0
#        d12=0 d13=0 d14=0 d15=0
# op=1
# s1=0 s0=1
#        d0=1 d1=0 d2=0 d3=0
#        d4=0 d5=0 d6=0 d7=0
#        d8=0 d9=0 d10=0 d11=0
#        d12=0 d13=0 d14=0 d15=0
# op=0
# s1=0 s0=1
#        d0=1 d1=1 d2=0 d3=0
#        d4=0 d5=0 d6=0 d7=0
#        d8=0 d9=0 d10=0 d11=0
#        d12=0 d13=0 d14=0 d15=0
# op=1
# s1=1 s0=0
#        d0=1 d1=1 d2=0 d3=0
#        d4=0 d5=0 d6=0 d7=0
#        d8=0 d9=0 d10=0 d11=0
#        d12=0 d13=0 d14=0 d15=0
# op=0
# s1=1 s0=0
#        d0=1 d1=1 d2=1 d3=0
#        d4=0 d5=0 d6=0 d7=0
#        d8=0 d9=0 d10=0 d11=0
#        d12=0 d13=0 d14=0 d15=0
# op=1
```

```
s1 = 1;
s2 = 0;
s3 = 0;
#10;
d3 = 1;

#10;
s0 = 0;
s1 = 0;
s2 = 1;
s3 = 0;
#10;
d4 = 1;

#10;
s0 = 1;
s1 = 0;
s2 = 1;
s3 = 0;
#10;
d5 = 1;

#10;
s0 = 0;
s1 = 1;
s2 = 1;
s3 = 0;
#10;
d6 = 1;

#10;
s0 = 1;
s1 = 1;
s2 = 1;
s3 = 0;
#10;
d7 = 1;

#10
s0 = 0;
s1 = 0;
s2 = 0;
s3 = 1;
#10;
d8 = 1;
```

Text Output (Part 2)

```
# s1=1 s0=0
#       d0=1 d1=1 d2=1 d3=0
#       d4=0 d5=0 d6=0 d7=0
#       d8=0 d9=0 d10=0 d11=0
#       d12=0 d13=0 d14=0 d15=0
# op=1
# s1=1 s0=1
#       d0=1 d1=1 d2=1 d3=0
#       d4=0 d5=0 d6=0 d7=0
#       d8=0 d9=0 d10=0 d11=0
#       d12=0 d13=0 d14=0 d15=0
# op=0
# s1=1 s0=1
#       d0=1 d1=1 d2=1 d3=1
#       d4=0 d5=0 d6=0 d7=0
#       d8=0 d9=0 d10=0 d11=0
#       d12=0 d13=0 d14=0 d15=0
# op=1
# s1=0 s0=0
#       d0=1 d1=1 d2=1 d3=1
#       d4=0 d5=0 d6=0 d7=0
#       d8=0 d9=0 d10=0 d11=0
#       d12=0 d13=0 d14=0 d15=0
# op=0
# s1=0 s0=0
#       d0=1 d1=1 d2=1 d3=1
#       d4=1 d5=0 d6=0 d7=0
#       d8=0 d9=0 d10=0 d11=0
#       d12=0 d13=0 d14=0 d15=0
# op=1
```

```
#10;
s0 = 1;
s1 = 0;
s2 = 0;
s3 = 1;
#10;
d9 = 1;

#10;
s0 = 0;
s1 = 1;
s2 = 0;
s3 = 1;
#10;
d10 = 1;

#10;
s0 = 1;
s1 = 1;
s2 = 0;
s3 = 1;
#10;
d11 = 1;

#10;
s0 = 0;
s1 = 0;
s2 = 1;
s3 = 1;
#10;
d12 = 1;

#10;
s0 = 1;
s1 = 0;
s2 = 1;
s3 = 1;
#10;
d13 = 1;

#10;
s0 = 0;
s1 = 1;
s2 = 1;
```

Observations

1. For S3 = 0, S2 = 0, S1 = 0, S0 = 0, Output is equal to D0.
2. For S3 = 0, S2 = 0, S1 = 0, S0 = 1, Output is equal to D1.
3. For S3 = 0, S2 = 0, S1 = 1, S0 = 0, Output is equal to D2.
4. For S3 = 0, S2 = 0, S1 = 1, S0 = 1, Output is equal to D3.
5. For S3 = 0, S2 = 1, S1 = 0, S0 = 0, Output is equal to D4.
6. For S3 = 0, S2 = 1, S1 = 0, S0 = 1, Output is equal to D5.
7. For S3 = 0, S2 = 1, S1 = 1, S0 = 0, Output is equal to D6.
8. For S3 = 0, S2 = 1, S1 = 1, S0 = 1, Output is equal to D7.
9. For S3 = 1, S2 = 0, S1 = 0, S0 = 0, Output is equal to D8.
10. For S3 = 1, S2 = 0, S1 = 0, S0 = 1, Output is equal to D9.
11. For S3 = 1, S2 = 0, S1 = 1, S0 = 0, Output is equal to D10.
12. For S3 = 1, S2 = 0, S1 = 1, S0 = 1, Output is equal to D11.
13. For S3 = 1, S2 = 1, S1 = 0, S0 = 0, Output is equal to D12.
14. For S3 = 1, S2 = 1, S1 = 0, S0 = 1, Output is equal to D13.
15. For S3 = 1, S2 = 1, S1 = 1, S0 = 0, Output is equal to D14.
16. For S3 = 1, S2 = 1, S1 = 1, S0 = 1, Output is equal to D15.

Hence, we confirm that **this is a 16:1 MUX**

```
        s3 = 1;
        #10;
        d14 = 1;

        #10;
        s0 = 1;
        s1 = 1;
        s2 = 1;
        s3 = 1;
        #10;
        d15 = 1;
    end
    initial begin
        $monitor("s1=%b s0=%b\n\td0=%b d1=%b d2=%b d3=%b\n\td4=%b d5=%b d6=%b d7=
%b\n\td8=%b d9=%b d10=%b d11=%b\n\td12=%b d13=%b d14=%b d15=%b\nop=%b",s1,s0,d0,d
1,d2,d3,d4,d5,d6,d7,d8,d9,d10,d11,d12,d13,d14,d15,op);
    end
endmodule
```

Simulated Wavefront