UNIVERSITY OF TEHRAN

Report Computer Assignment 1
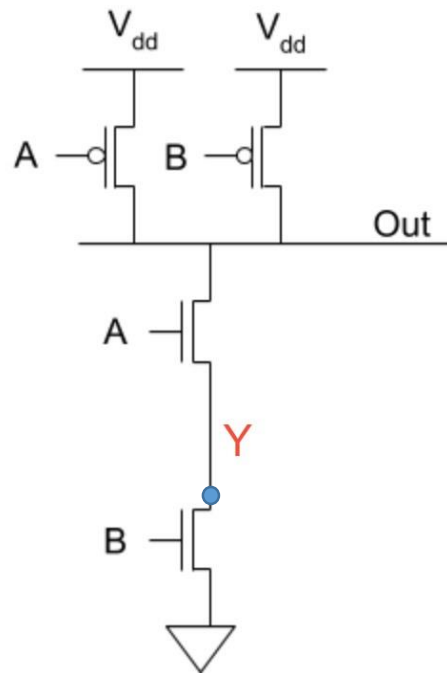
Digital Logic Design, ECE 367 / Digital Systems I, ECE 894

Danial Saeedi

Problem 1

Circuit Diagram

Here's the circuit that I'm analyzing:

NAND CMOS Circuit

Verilog Code

Here's my Verilog Code for Switch Level NAND Gate. (I used sublime for coloring this code).

The y wire is shown in the Circuit Diagram. a and b are the input of NAND Gate and the output is w :

```
1  `timescale 1ns/1ns
2  module MyNAND (input a,b,output w);
3      wire y;
4      supply1 Vdd;
5      supply0 Gnd;
6      pmos #(5,6,7) T1(w,Vdd,a);
7      pmos #(5,6,7) T2(w,Vdd,b);
8      nmos #(3,4,5) T3(y,Gnd,b);
9      nmos #(3,4,5) T4(w,y,a);
10 endmodule
```

MyNand.v

Worst-case Delay

To 1 : It takes 5ns to change the output of the PMOS transistors to 1. It takes 5 + 5 = 10ns to get Hi-Z from pull-down structure. Thus it takes **10ns** in the worst case to change the output of the NAND Gate to 1.

<p align="center"><mark>To 1 = 10ns</mark></p>

To 0 : It takes 7ns to change the output of pull-up structure to Hi-Z. It takes 4 + 4 = 8ns to change the output of pull-down structure to 0

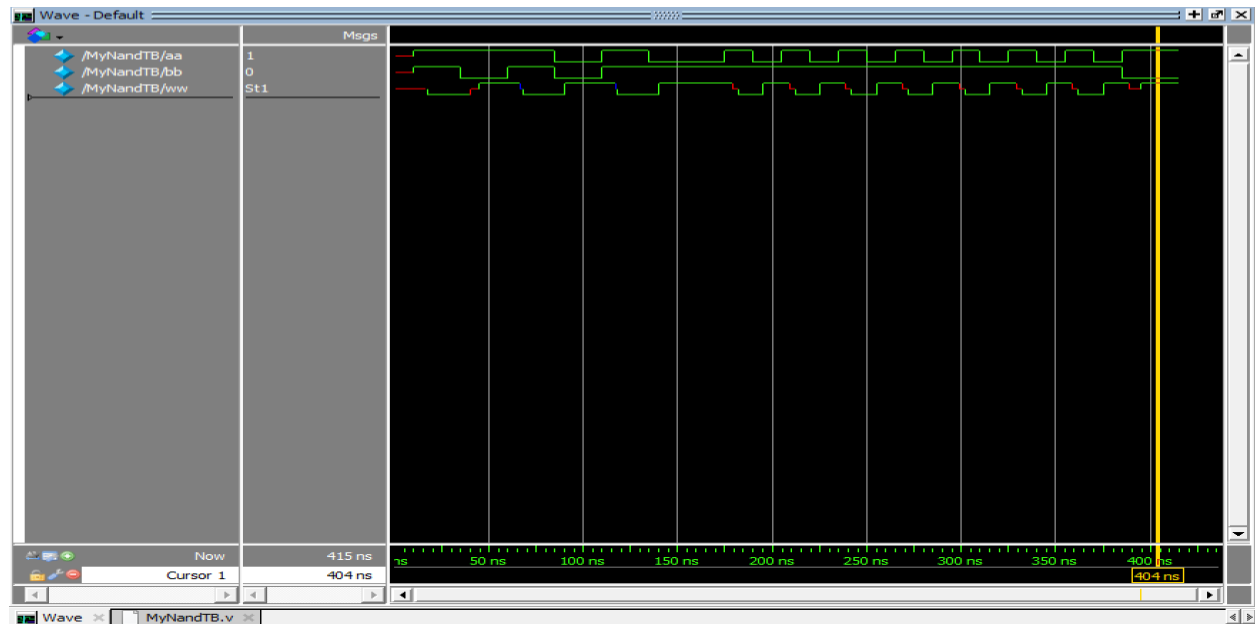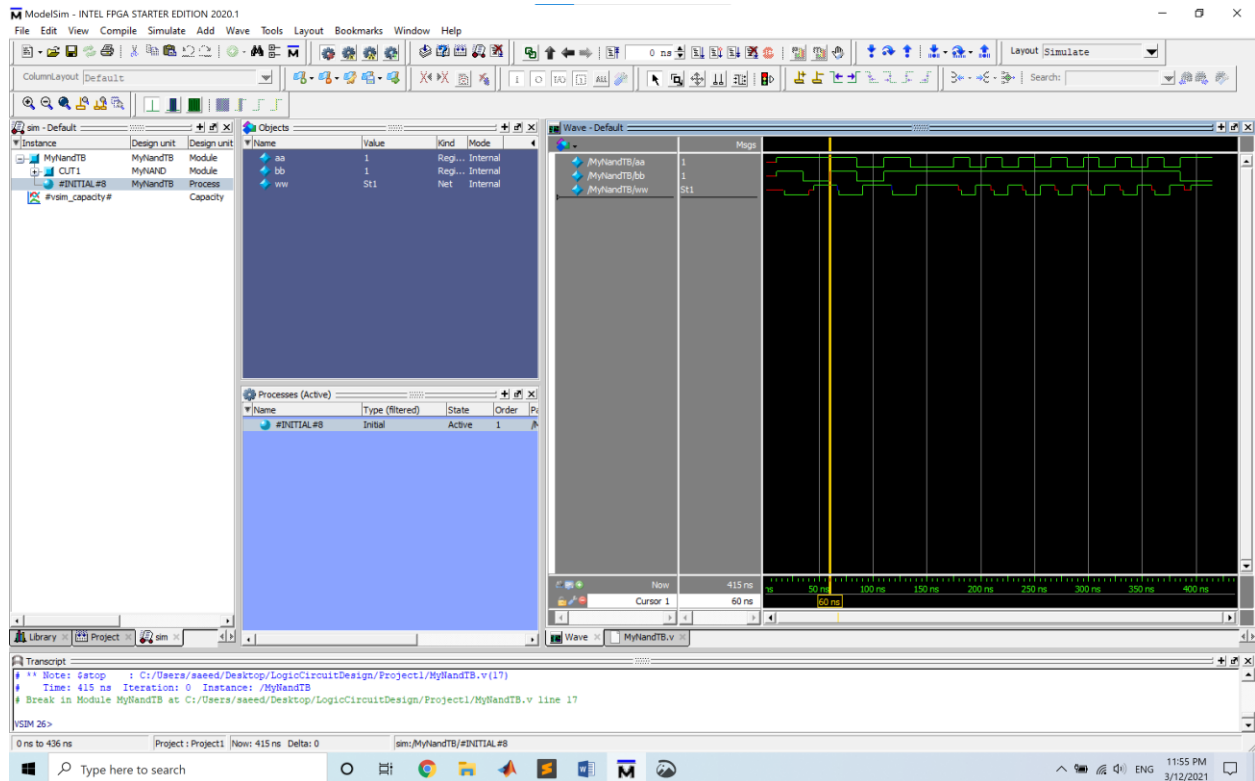<p align="center"><mark>To 0 = 8ns</mark></p>

Testbench
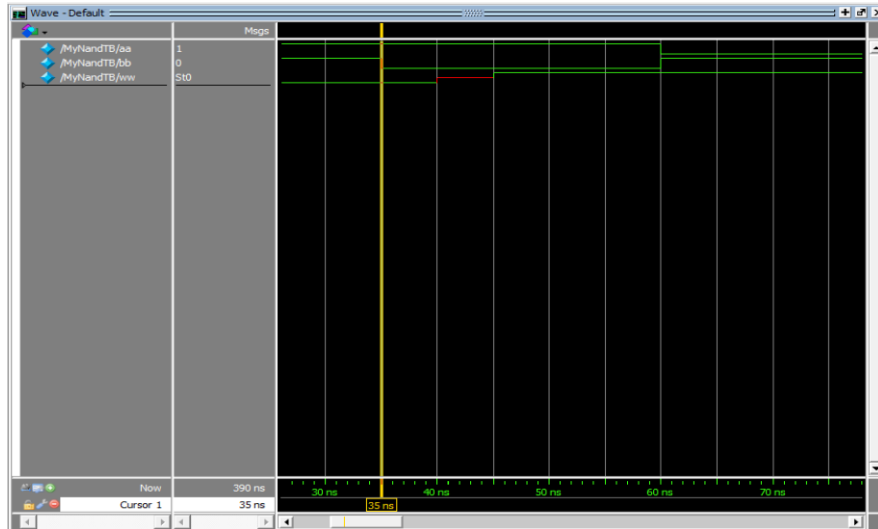
```verilog
1   `timescale 1ns/1ns
2   module MyNandTB();
3       //Inputs
4       reg aa,bb;
5       //The NAND output
6       wire ww;
7       MyNAND CUT1(aa,bb,ww);
8       initial begin
9       #10 aa = 1;bb = 1;
10      #25 aa = 1;bb = 0;
11      #25 aa = 1;bb = 1;
12      #25 aa = 0;bb = 0;
13      #25 aa = 1;bb = 1;
14      #25 aa = 0;bb = 1;
15      #25
16      repeat(15) #15 aa = ~aa; bb = ~bb;
17      #30 $stop;
18      end
19  endmodule
```
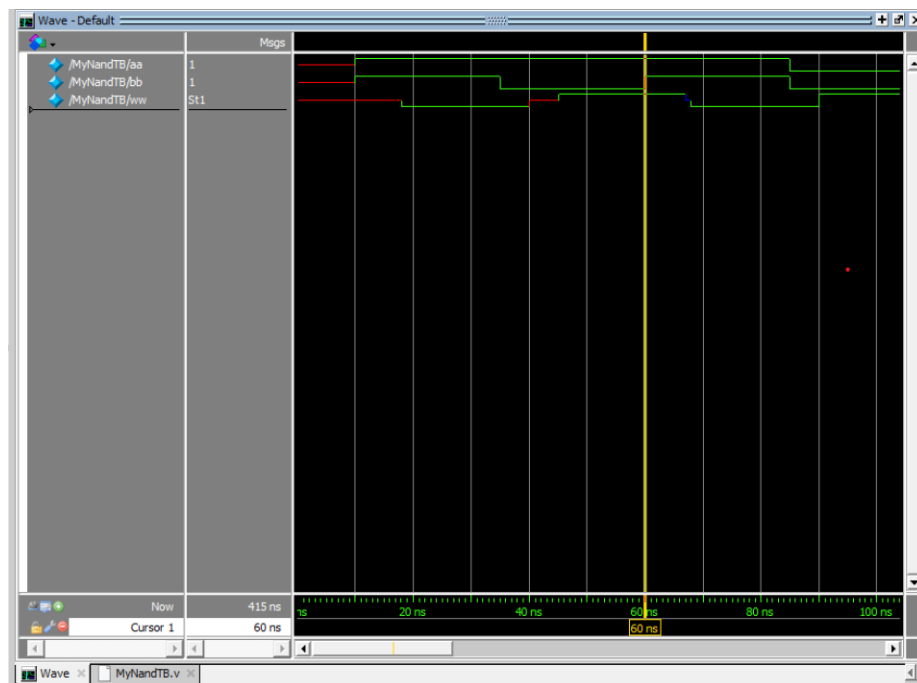
<p align="center">MyNandTB.v</p>

Simulation Result

The To1 worst case happens when the input changes from a = 1 and b = 1 to a = 1 and b = 0.We expect to see 10ns delay. We can see from the zoomed waveform, To1 delay is correct: (The output changes to 1 at 45ns)

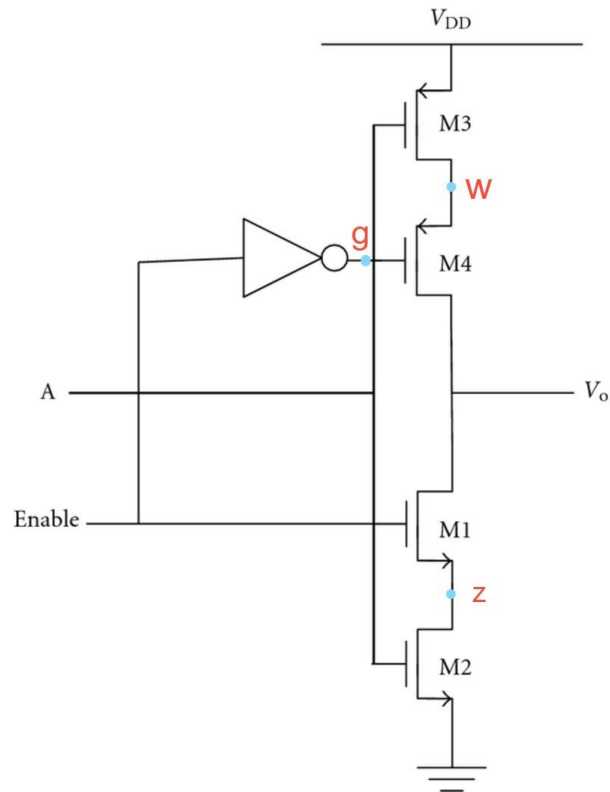The To0 worst-case delay happens when the inputs changes from a = 1 and b = 0 to a = 1 and b = 1.As we expect, the To0 delay is 8ns:

Problem 2

Circuit Diagram

Here's the circuit that I'm analyzing:

Verilog Code

The y, w and z wires are shown in the Circuit Diagram. I use the inverter module that I created in part 1.

```verilog
`timescale 1ns/1ns
module MyTriStateBuffer (input a,en,output y);
    wire g,w,z;
    supply1 Vdd;
    supply0 Gnd;
    MyInverter InverterGate(en,g);
    pmos #(5,6,7) T1(w,Vdd,a);
    pmos #(5,6,7) T2(y,w,g);
    nmos #(3,4,5) T3(y,z,en);
    nmos #(3,4,5) T4(z,Gnd,a);
endmodule
```

MyTriStateBuffer.v

Testbench

```verilog
1  `timescale 1ns/1ns
2  module MyTriStateTB();
3      wire yy;
4      reg aa,enable;
5      MyTriStateBuffer CUT1(aa,enable,yy);
6      initial begin
7      #10 aa = 1;enable = 0;
8      #21 aa = 0;enable = 1;
9      #21 aa = 1;enable = 1;
10     #21 aa = 0;enable = 0;
11     #21 aa = 1;enable = 1;
12     #20
13     repeat(15) #15 aa = ~aa; enable = ~enable;
14     #30 $stop;
15     end
16 endmodule
```

MyTriStateTB.v

Worst-case Delay

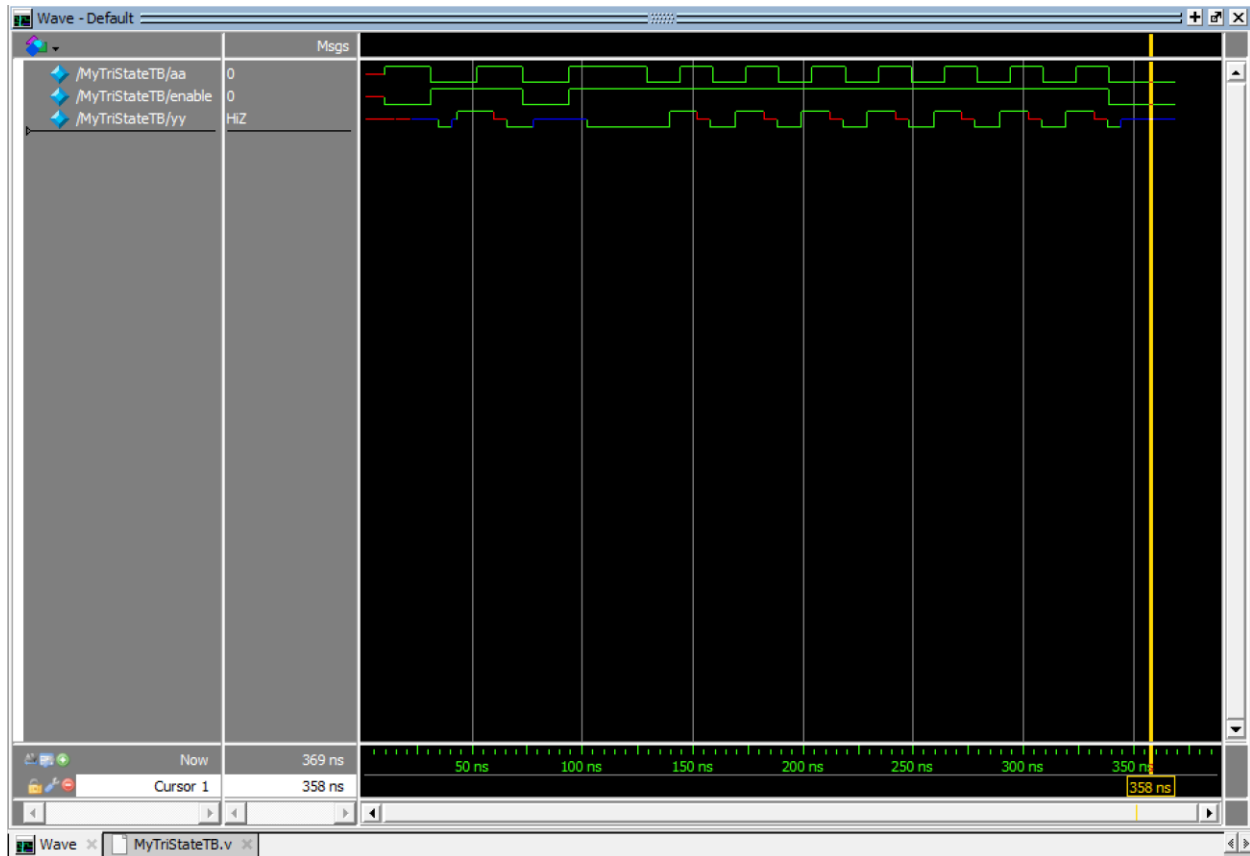**To 1 :** It takes 6 + 5 = 11ns to changes the output of the pull-up structure to 1.(NOT gate delay is included)

<mark>To 1 = 11ns</mark>

To 0 : It takes 7 + 7 = 14ns to changes the output of the pull-up structure to Hi-Z. And it takes 4 + 4 = 8ns to to changes the output of the pull-down structure to 0.
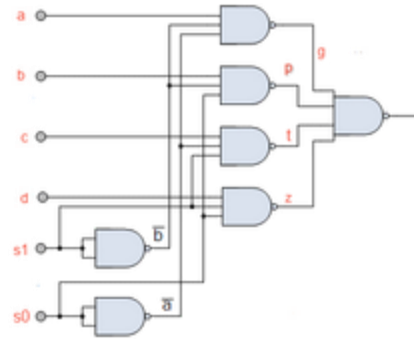
<mark>To 0 = 14ns</mark>

Simulation Result

As expected, the To 1 and To 0 are 11ns and 14ns respectively.

Problem 3

Circuit Diagram

Here's the circuit that I'm analyzing:

Approach 1

From part 1, we know that NAND gate delays are #(10,8). We use NAND from VerilogSystem for this approach.

Verilog Code For Approach 1

The s0_not,s1_not, g,p,t and z wires are shown in the circuit diagram.

```verilog
1  `timescale 1ns/1ns
2  module MyMUX(
3      input s0,s1,a,b,c,d,
4      output y
5  );
6      wire s0_not,s1_not,g,p,t,z;
7      nand #(10,8) nand_1(s0_not,s0,s0);
8      nand #(10,8) nand_2(s1_not,s1,s1);
9      nand #(10,8) nand_3(g,a,s1_not,s0_not);
10     nand #(10,8) nand_4(p,b,s1_not,s0);
11     nand #(10,8) nand_5(t,c,s1,s0_not);
12     nand #(10,8) nand_6(z,d,s1,s0);
13     nand #(10,8) nand_7(y,g,p,t,z);
14 endmodule
```

MyMUX.v

Testbench For Approach 1

```verilog
1   `timescale 1ns/1ns
2   module MyMuxTB();
3       wire yy;
4       reg ss0,ss1,aa,bb,cc,dd;
5       MyMUX CUT1(ss0,ss1,aa,bb,cc,dd,yy);
6       initial begin
7       #10 ss1 = 0; ss0 = 0; aa = 1;bb = 0; cc = 0;dd = 0;
8       #50 ss1 = 1; ss0 = 1; aa = 1;bb = 0; cc = 1;dd = 0;
9       #50 ss1 = 0; ss0 = 1; aa = 0;bb = 1; cc = 1;dd = 0;
10      #50 ss1 = 1; ss0 = 0; aa = 0;bb = 1; cc = 0;dd = 0;
11      #50 ss1 = 0; ss0 = 0; aa = 0;bb = 1; cc = 1;dd = 0;
12      #50 ss1 = 1; ss0 = 0; aa = 0;bb = 1; cc = 0;dd = 0;
13      #50 ss1 = 0; ss0 = 1; aa = 0;bb = 0; cc = 1;dd = 0;
14      #50 ss1 = 1; ss0 = 1; aa = 0;bb = 1; cc = 1;dd = 1;
15      #50 $stop;
16      end
17  endmodule
18
```

MyMUX-TB.v

Simulation Result For Approach 1
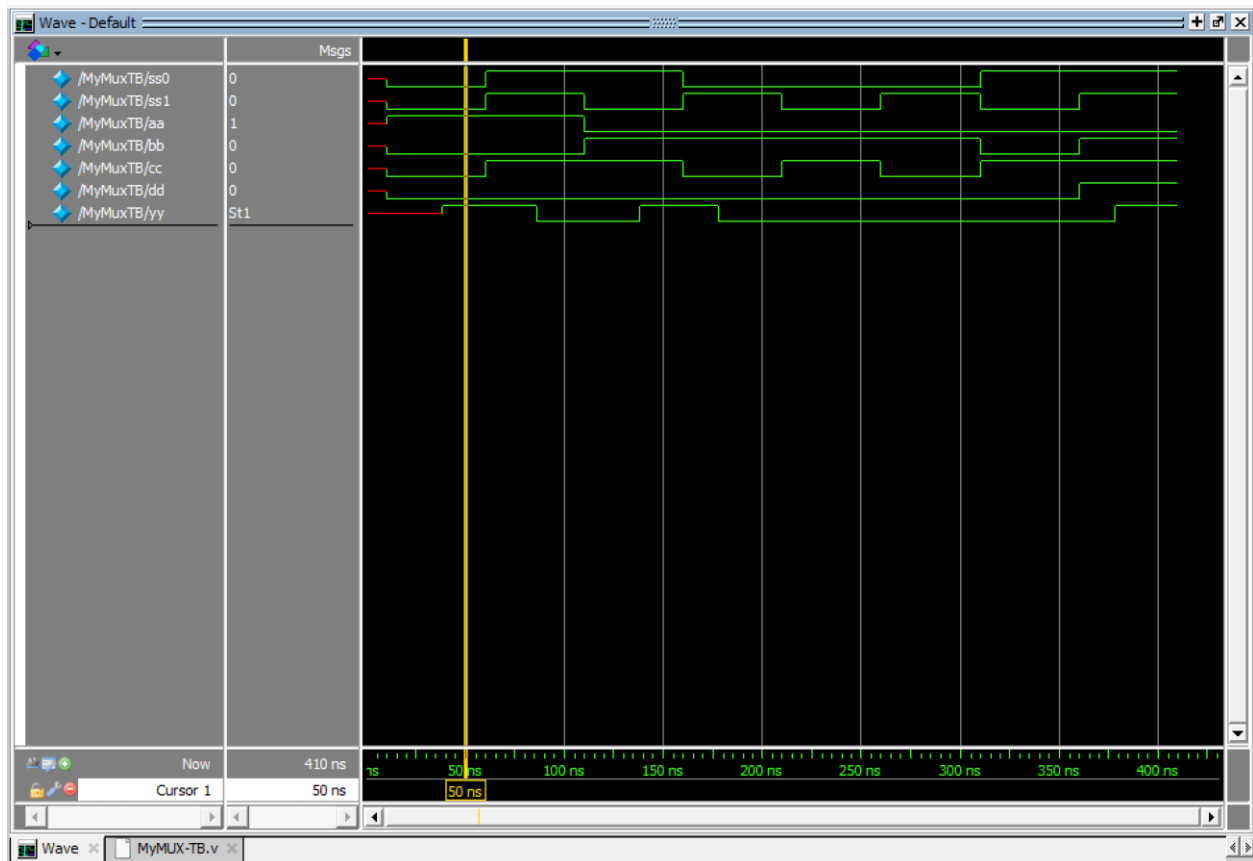
Analyzing Simulation Result

The output of MUX 4 to 1 is the result we expect.

For instance when the inputs are :

ss1 = 0; ss0 = 0; aa = 1;bb = 0; cc = 0;dd = 0;

We expect the output be 1. After 28ns delay,the output changes to 1.

Approach 2

We use the NAND gate module from part 1. I've created three NAND gates with 2, 3 and 4 inputs.

Note that this approach is much more accurate than approach 1. Because we're simulating in transistor level.

Verilog Code for 3 and 4 inputs NAND gate :

Three Inputs NAND Gate

```
1   `timescale 1ns/1ns
2   module ThreeInputsNand(input a,b,c,output w);
3       wire g,z;
4       supply1 Vdd;
5       supply0 Gnd;
6       pmos #(5,6,7) T1(w,Vdd,a);
7       pmos #(5,6,7) T2(w,Vdd,b);
8       pmos #(5,6,7) T3(w,Vdd,c);
9       nmos #(3,4,5) T4(w,z,a);
10      nmos #(3,4,5) T5(z,g,b);
11      nmos #(3,4,5) T6(g,Gnd,c);
12  endmodule
```

ThreeInputsNand.v

Four Inputs NAND Gate

```
1   `timescale 1ns/1ns
2   module FourInputsNand(input a,b,c,d,output w);
3       wire g,p,z;
4       supply1 Vdd;
5       supply0 Gnd;
6       pmos #(5,6,7) T1(w,Vdd,a);
7       pmos #(5,6,7) T2(w,Vdd,b);
8       pmos #(5,6,7) T3(w,Vdd,c);
9       pmos #(5,6,7) T4(w,Vdd,d);
10      nmos #(3,4,5) T5(w,z,a);
11      nmos #(3,4,5) T6(z,g,b);
12      nmos #(3,4,5) T7(g,p,c);
13      nmos #(3,4,5) T8(p,Gnd,d);
14  endmodule
```

FourInputsNand.v

Verilog Code For Approach 2

```verilog
 1    `timescale 1ns/1ns
 2    module MyMuxV2(
 3        input s0,s1,a,b,c,d,
 4        output y
 5    );
 6        wire s0_not,s1_not,g,p,t,z;
 7        MyNAND nand_1(s0,s0,s0_not);
 8        MyNAND nand_2(s1,s1,s1_not);
 9        ThreeInputsNand nand_3(a,s1_not,s0_not,g);
10        ThreeInputsNand nand_4(b,s1_not,s0,p);
11        ThreeInputsNand nand_5(c,s1,s0_not,t);
12        ThreeInputsNand nand_6(d,s1,s0,z);
13        FourInputsNand nand_7(g,p,t,z,y);
14    endmodule
```

MyMUX-v2.v

Testbench For Approach 2

```verilog
 1    `timescale 1ns/1ns
 2    module MyMuxV2TB();
 3        wire yy;
 4        reg ss0,ss1,aa,bb,cc,dd;
 5        MyMuxV2 CUT1(ss0,ss1,aa,bb,cc,dd,yy);
 6        initial begin
 7        #10 ss1 = 0; ss0 = 0; aa = 1;bb = 0; cc = 0;dd = 0;
 8        #50 ss1 = 1; ss0 = 1; aa = 1;bb = 0; cc = 1;dd = 0;
 9        #50 ss1 = 0; ss0 = 1; aa = 0;bb = 1; cc = 1;dd = 0;
10        #50 ss1 = 1; ss0 = 0; aa = 0;bb = 1; cc = 0;dd = 0;
11        #50 ss1 = 0; ss0 = 0; aa = 0;bb = 1; cc = 1;dd = 0;
12        #50 ss1 = 1; ss0 = 0; aa = 0;bb = 1; cc = 0;dd = 0;
13        #50 ss1 = 0; ss0 = 1; aa = 0;bb = 0; cc = 1;dd = 0;
14        #50 ss1 = 1; ss0 = 1; aa = 0;bb = 1; cc = 1;dd = 1;
15        #50 $stop;
16        end
17    endmodule
```

MyMUX-v2-TB.v

Simulation Result For Approach 2
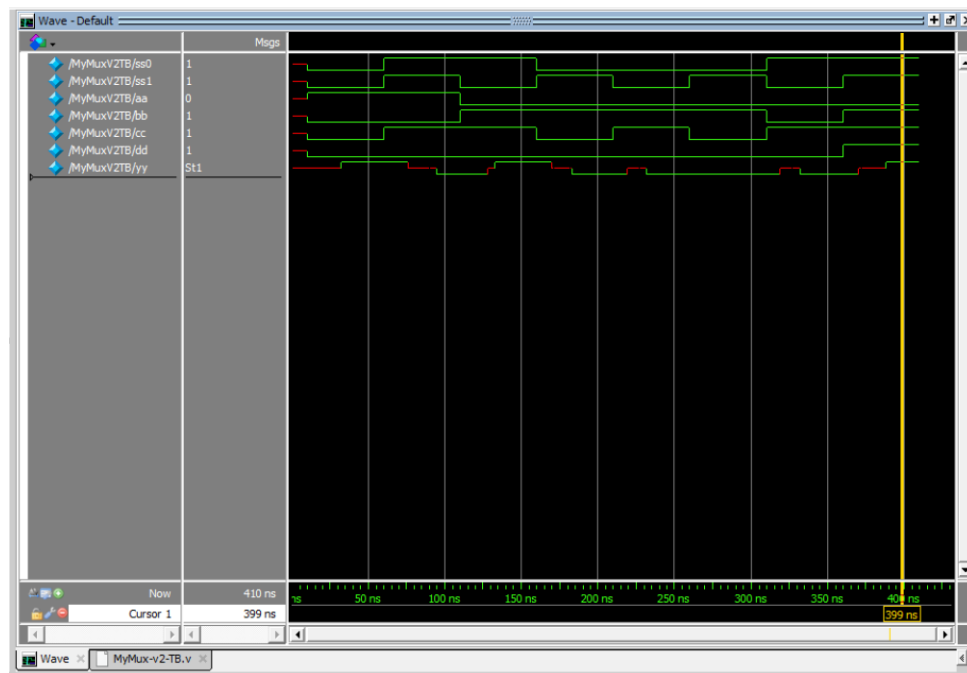
Analyzing Simulation Result

The output of this module is correct.

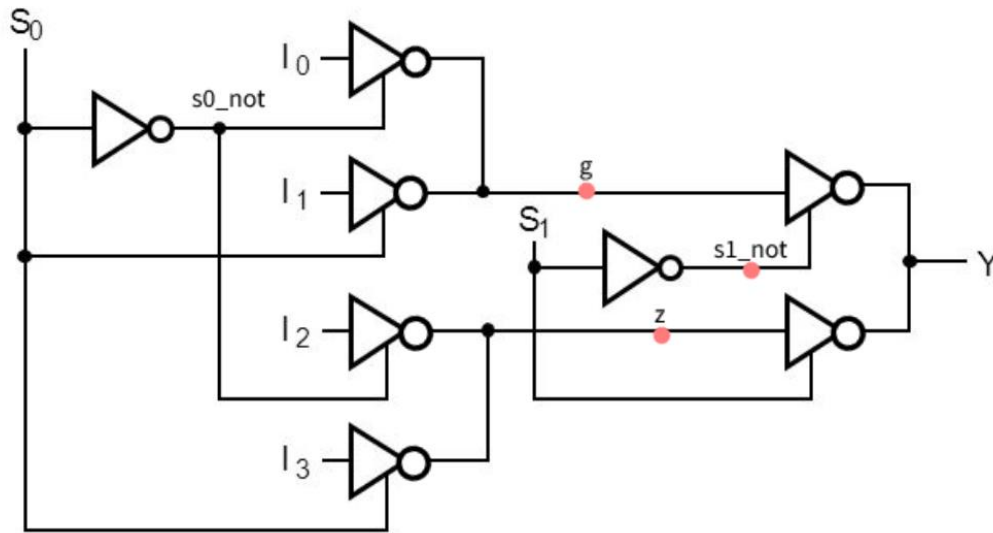The 2,3 and 4 inputs NAND gate delays are #(10,8) , #(15,12) and #(20,16) respectively.

The simulation result shows that the To0 delay is 35ns in the worst-case that I found.

The simulation result shows that the To1 delay is 38ns in the worst-case that I found.

Problem 4

Circuit Diagram

Here's the circuit that I'm analyzing:

Verilog Code

The s0_not, s1_not, g and z wires are shown in the circuit diagram.I used MyInverter module for the NOT gate and MyTriStateBuffer for the tri-state buffer gate.

```verilog
1   `timescale 1ns/1ns
2   module MyMUX2(
3       input s0,s1,a,b,c,d,
4       output y
5   );
6       wire s0_not,s1_not,g,z;
7       MyInverter inverter_gate1(s0,s0_not);
8       MyInverter inverter_gate2(s1,s1_not);
9
10      MyTriStateBuffer not_buffer1(a,s0_not,g);
11      MyTriStateBuffer not_buffer2(b,s0,g);
12      MyTriStateBuffer not_buffer3(c,s0_not,z);
13      MyTriStateBuffer not_buffer4(d,s0,z);
14
15      MyTriStateBuffer not_buffer5(g,s1_not,y);
16      MyTriStateBuffer not_buffer6(z,s1,y);
17  endmodule
```

MyMUX2.v

Testbench

```verilog
`timescale 1ns/1ns
module MyMux2TB();
    wire yy;
    reg ss0,ss1,aa,bb,cc,dd;
    MyMUX2 CUT1(ss0,ss1,aa,bb,cc,dd,yy);
    initial begin
    #10 ss1 = 0; ss0 = 0; aa = 1;bb = 0; cc = 0;dd = 0;
    #50 ss1 = 1; ss0 = 1; aa = 1;bb = 0; cc = 1;dd = 0;
    #50 ss1 = 0; ss0 = 1; aa = 0;bb = 1; cc = 1;dd = 0;
    #50 ss1 = 1; ss0 = 0; aa = 0;bb = 1; cc = 0;dd = 0;
    #50 ss1 = 0; ss0 = 0; aa = 0;bb = 1; cc = 1;dd = 0;
    #50 ss1 = 1; ss0 = 0; aa = 0;bb = 1; cc = 0;dd = 0;
    #50 ss1 = 0; ss0 = 1; aa = 0;bb = 0; cc = 1;dd = 0;
    #50 ss1 = 1; ss0 = 1; aa = 0;bb = 1; cc = 1;dd = 1;
    #50 $stop;
    end
endmodule
```

MyMUX2-TB.v

Simulation Result
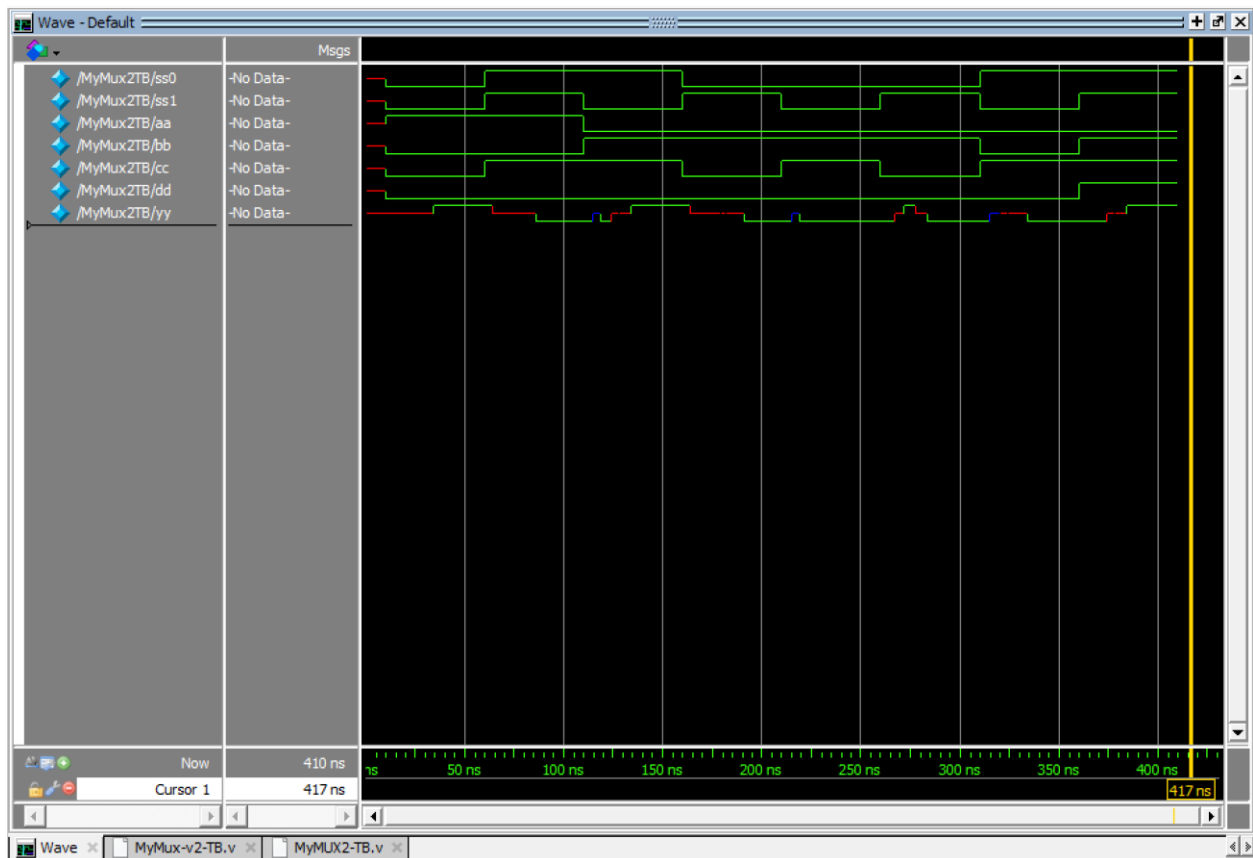
Analyzing Simulation Result

The output of this module is correct.

In problem 2, we calculated the tri-state buffer delays. This gate delays are #(11,14) and the NOT gate delays are #(5,6).

From the longest path as shown in the circuit diagram:

- I expect 5 + 14 + 11 = 30ns delay for changing the output to 1.
- I expect 5 + 11 + 14 = 30ns delay for changing the output to 0.

The simulation result shows that the To0 delay is 31ns in the worst-case that I found. (1ns difference between what I've calculated)

The simulation result shows that the To1 delay is 28ns in the worst-case that I found. (2ns difference between what I've calculated)
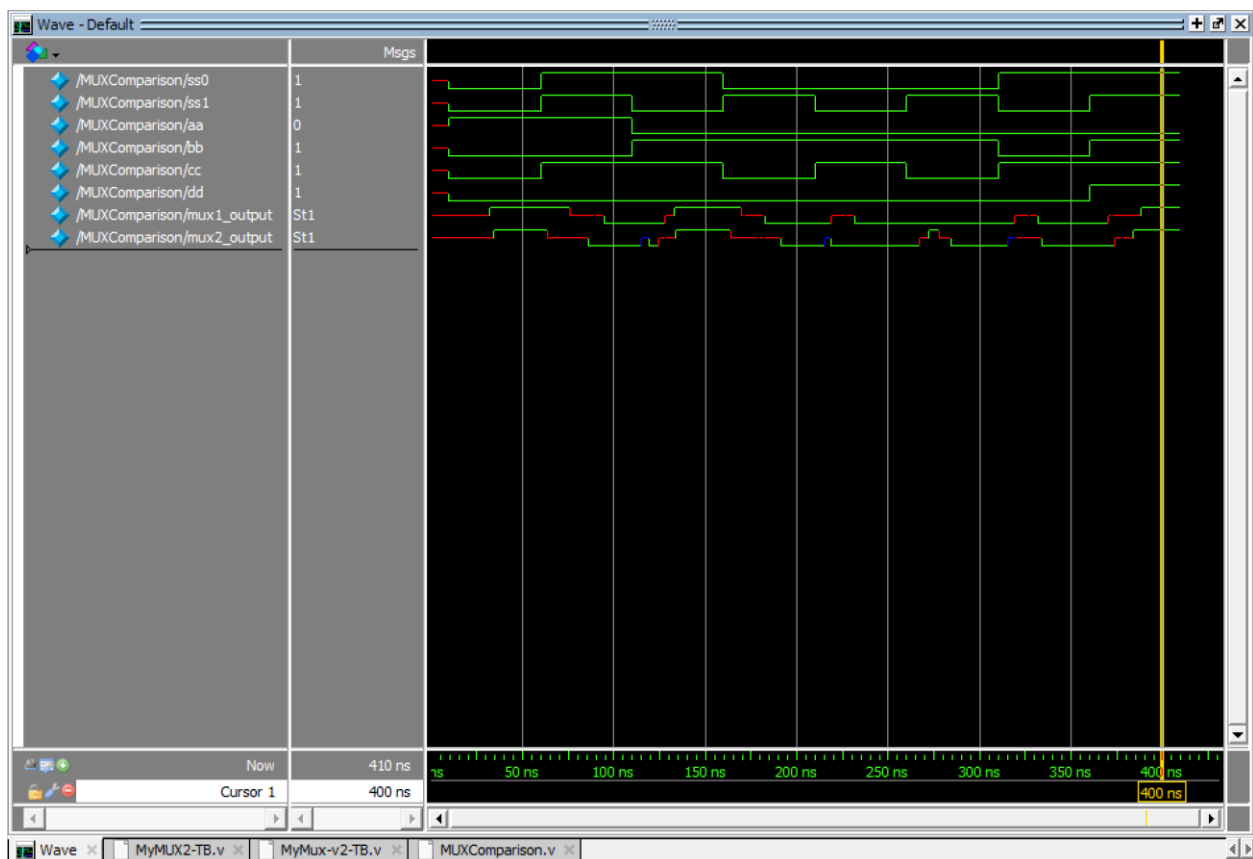
Problem 5

Testbench

```
1    `timescale 1ns/1ns
2    module MUXComparison();
3        wire mux1_output,mux2_output;
4        reg ss0,ss1,aa,bb,cc,dd;
5        //Mux Problem 3
6        MyMuxV2 mux1(ss0,ss1,aa,bb,cc,dd,mux1_output);
7        //Mux Problem 4
8        MyMUX2 mux2(ss0,ss1,aa,bb,cc,dd,mux2_output);
9        initial begin
10       #10 ss1 = 0; ss0 = 0; aa = 1;bb = 0; cc = 0;dd = 0;
11       #50 ss1 = 1; ss0 = 1; aa = 1;bb = 0; cc = 1;dd = 0;
12       #50 ss1 = 0; ss0 = 1; aa = 0;bb = 1; cc = 1;dd = 0;
13       #50 ss1 = 1; ss0 = 0; aa = 0;bb = 1; cc = 0;dd = 0;
14       #50 ss1 = 0; ss0 = 0; aa = 0;bb = 1; cc = 1;dd = 0;
15       #50 ss1 = 1; ss0 = 0; aa = 0;bb = 1; cc = 0;dd = 0;
16       #50 ss1 = 0; ss0 = 1; aa = 0;bb = 0; cc = 1;dd = 0;
17       #50 ss1 = 1; ss0 = 1; aa = 0;bb = 1; cc = 1;dd = 1;
18       #50 $stop;
19       end
20   endmodule
```

MUXComparison.v

Simulation Result

Calculating the number of transistors from problem 3

Every two-input NAND gate needs 2 PMOS and 2 NMOS transistors.(Total : 4)

Every three-input NAND gate needs 3 PMOS and 3 NMOS transistors.(Total : 6)

Every four-input NAND gate needs 4 PMOS and 4 NMOS transistors.(Total : 8)

There are 2 two-inputs NAND gate and 4 three-inputs NAND gate and 1 four-inputs NAND gate. Thus the total transistors we need for this problem is :

<div align="center">2*4+4*6+1*8 = 40 transistors</div>

Calculating the number of transistors from problem 4

Every NOT gate needs 1 PMOS and 1 NMOS transistors.(Total : 2)

Every tri-state buffer gate needs 6 transistors.

There are 2 NOT gate and 6 tri-state buffer gate. Thus the total transistors we need for this problem is :

<div align="center">6*6+2*2 = 40 transistors</div>

Cost And Power Consumption

Both problems need 40 transistors. So the cost is the same.But problem 3 and 4 need 7 and 8 gates respectively. Each gate is connected to supply 1 and supply 0 in its CMOS structure.

So problem 4 consumes more power than problem 3.