University of Tehran

School of Electrical and Computer Engineering

Digital systems 1

Dr.Navvabi

Computer Assignment #1

Amirhosein Yavarikhoo

810199514

Spring 1401

1. OAI1 gate and switch level description are shown below.

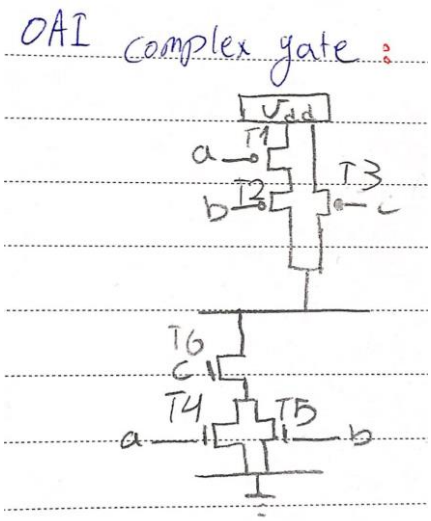
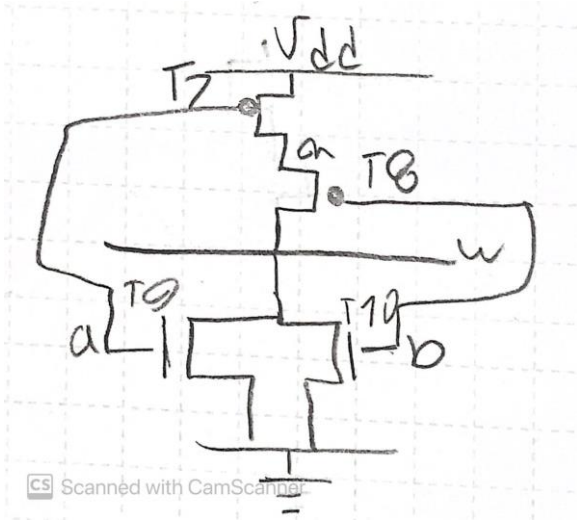


Figure2 OAI1 complex gate design

```
Ln#
1   `timescale 1ns/1ns
2   module OAI1 (input a, input b, input c, output w);
3           supply1 Vdd;
4           supply0 Gnd;
5           wire x,y,z;
6           pmos #(5,6,7) T1(x,Vdd,a);
7           pmos #(5,6,7) T2(w,x,b);
8           pmos #(5,6,7) T3(w,Vdd,c);
9           nmos #(3,4,5) T4(z,Gnd,a);
10          nmos #(3,4,5) T5(z,Gnd,b);
11          nmos #(3,4,5) T6(w,z,c);
12  endmodule
```

OAI1 switch level description1 Figure

2-input NOR gate and switch level description are shown below.




```
`timescale 1ns/1ns
module MyNor(input a, input b, output w);
        supply1 Vdd;
        supply0 Gnd;
        wire x;
        pmos #(5,6,7)T7(x,Vdd,a);
        pmos #(5,6,7)T8(w,x,b);
        nmos #(3,4,5)T9(w,Gnd,a);
        nmos #(3,4,5)T10(w,Gnd,b);
endmodule
```

Figure4 2-input NOR gate switch level description

Inverter gate and switch level description is shown below.

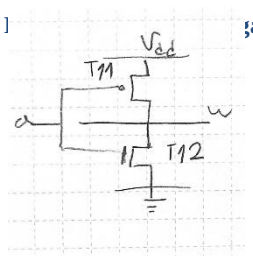


Figure6 Inverter gate

```
Ln#
1   `timescale 1ns/1ns
2   module Myinverter(input a,output w);
3           supply1 Vdd;
4           supply0 Gnd;
5           pmos #(5,6,7) T11(w,Vdd,a);
6           nmos #(3,4,5) T12(w,Gnd,a);
7   endmodule
```

Figure5 Inverter switch level description

2. OAI2 gate level design and description are shown below.

```
Ln#
1   `timescale 1ns/1ns
2   module OAI2 (input a, input b, input c ,output out);
3           wire x,y,z;
4           MyNor(.a(a),.b(b),.w(x));
5           Myinverter(.a(c),.w(y));
6           MyNor(.a(x),.b(y),.w(z));
7           Myinverter(.a(z),.w(out));
8   endmodule
```

Figure7 OAI2 gate level description

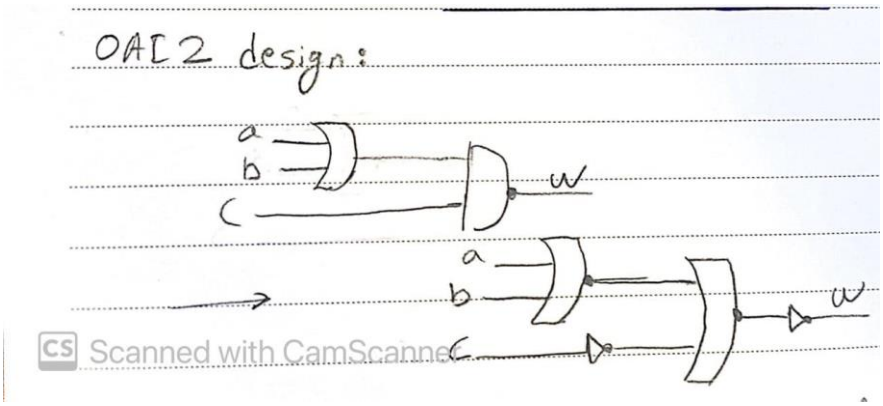


Figure8 OAI2 gate level design

3. OAI1 worst case delay calculation:

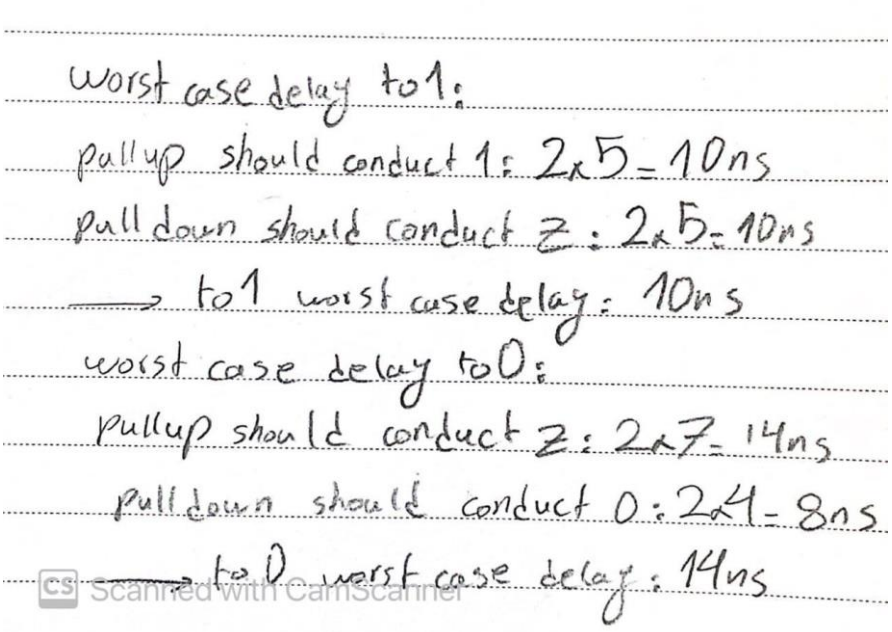


Figure9 OAI1 worst case delay calculation

OAI2 worst case delay calculation:

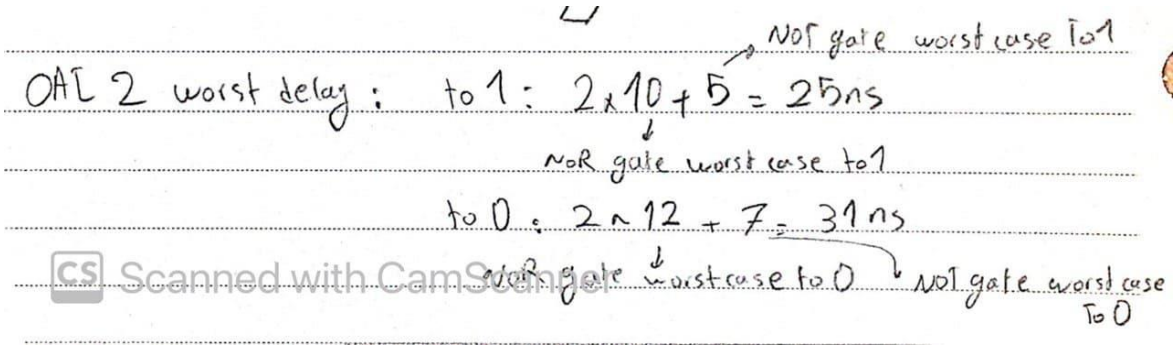


Figure 10 OAI2 worst case delay calculation

Testbench System Verilog codes:

```
Ln#
 1  `timescale 1ns/1ns
 2  module OAI1TB();
 3          logic aa=0,bb=0,cc=0;
 4          wire ww;
 5          OAI1 UUT(.a(aa),.b(bb),.c(cc),.w(ww));
 6          initial begin
 7          #12 bb=1;
 8          #37 cc=1;
 9          #39 cc=0;
10          #41 aa=1;
11          #25 $stop;
12          end
13  endmodule
14
```

Figure 11 OAI1 testbench code

```
Ln#
 1  `timescale 1ns/1ns
 2  module OAI2TB();
 3          logic aa=0,bb=0,cc=0;
 4          wire ww;
 5          OAI2 UUT(.a(aa),.b(bb),.c(cc),.out(ww));
 6          initial begin
 7          #12 bb=1;
 8          #37 cc=1;
 9          #39 cc=0;
10          #41 aa=1;
11          #25 $stop;
12          end
13  endmodule
```

Figure 12 OAI2 Testbench code

OAI1 simulation result vs handwriting result:

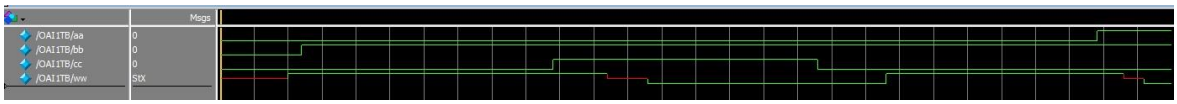


Figure 13 OAI1 testbench

OAI1 handwritten analysis:

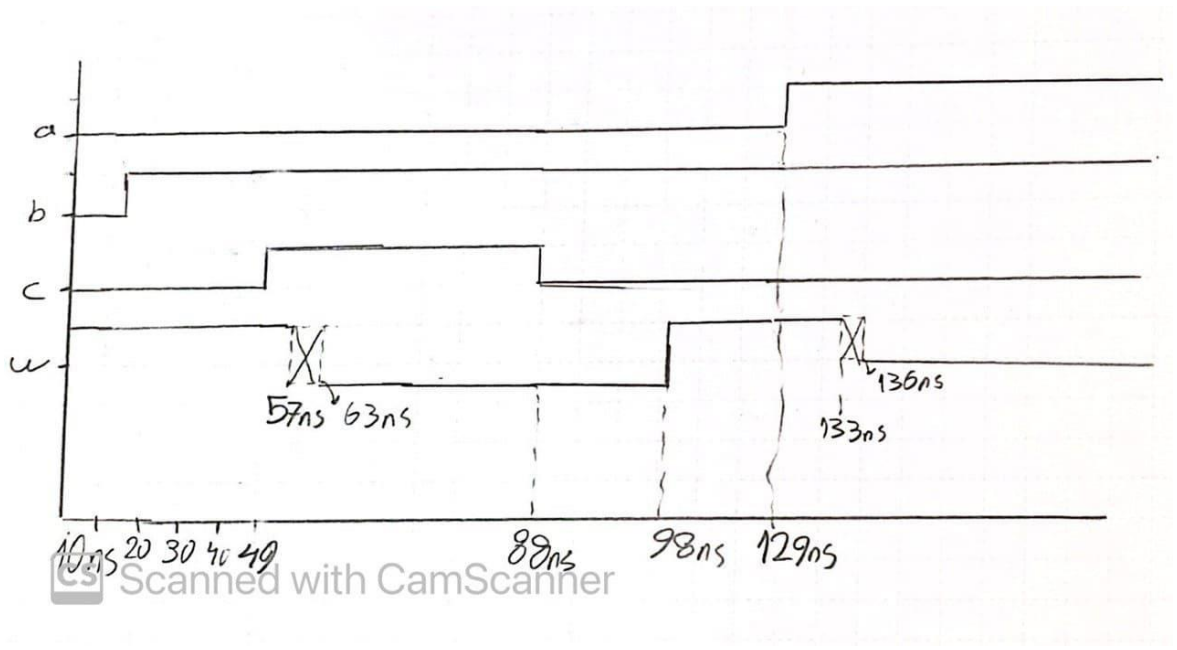


Figure 14 OAI1 analysis

The figures are almost the same. The only difference is that at the start I didn't account the delay and assumed that in the beginning w is 1.

OAI2 simulation result vs handwriting result:

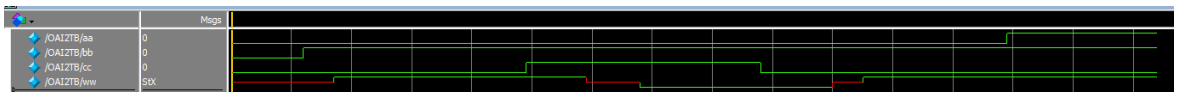
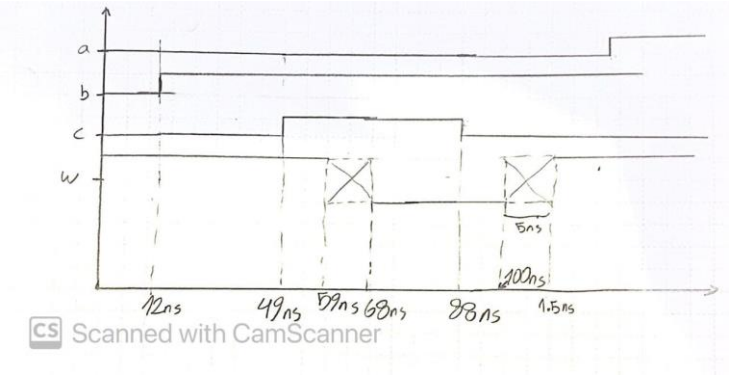


Figure 15 OAI2 testbench result

OAI2 handwritten analysis:



The handwritten analysis and testbench result are similar. As we can see from the waveforms, OAI2 has less accuracy and more delay periods (periods that are X) than OAI1 which is the result of using gates to describe OAI2 rather than writing down each transistor.

4. OAI4 code is shown below:

```
1  `timescale 1ns/1ns
2  module OAI4(input a,input b,input c,output w);
3          wire x,y;
4          or #(10,12) G1(x,a,b);
5          and #(6,8) G2(y,x,c);
6          not #(4,5) G3(w,y);
7  endmodule
```

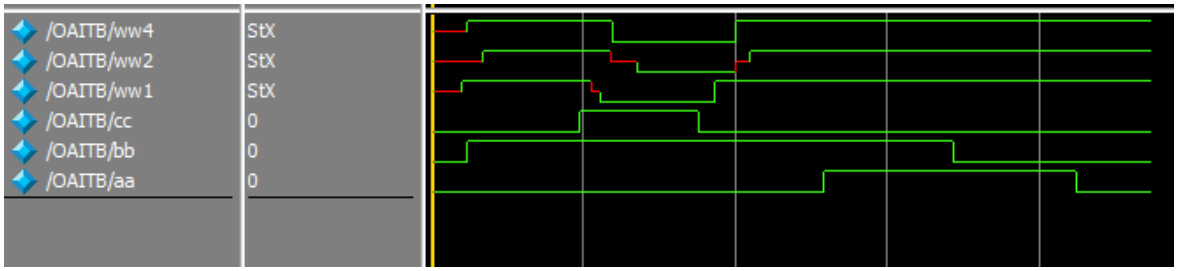
Figure 16 OAI4 code

5. The testbench code is shown below:

```
`timescale 1ns/1ns
module OAITB();
        logic aa=0,bb=0,cc=0;
        wire ww1,ww2,ww4;
        OAI1 oai1(.a(aa),.b(bb),.c(cc),.w(ww1));
        OAI2 oai2(.a(aa),.b(bb),.c(cc),.w(ww2));
        OAI4 oai4(.a(aa),.b(bb),.c(cc),.w(ww4));
        initial begin
        #12 bb=1;
        #37 cc=1;
        #39 cc=0;
        #41 aa=1;
        #43 bb=0;
        #40 aa=0;
        #25 $stop;
        end
endmodule
```

Figure 17 multiple OAIs testbench

This is the first testbench of OAI1,OAI2 and OAI4:



We change delay values of OAI4 in a new file and call it OAI5.

With almost reducing 2ns for each delay the wave looks pretty similar to OAI1.

```
1  `timescale 1ns/1ns
2  module OAI5(input a,input b,input c,output w);
3          wire x,y;
4          or #(8,10) G1(x,a,b);
5          and #(4,6) G2(y,x,c);
6          not #(2,3) G3(w,y);
7  endmodule
```

Figure18 OAI5 code

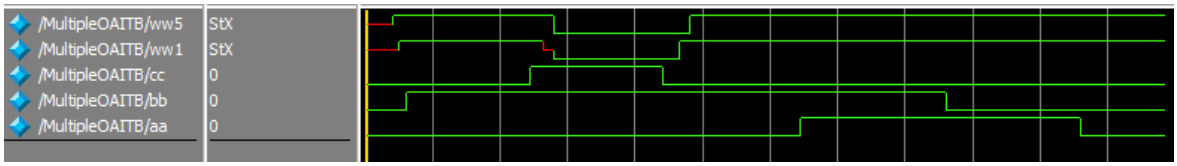


Figure 19 OAI5 wave comparison with OAI1

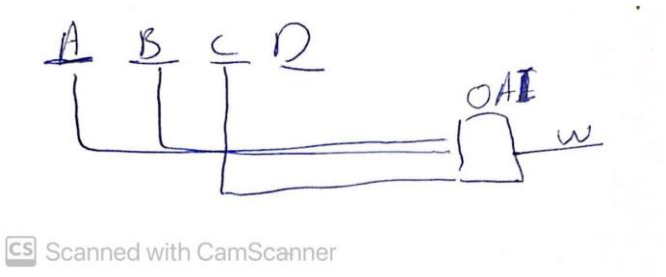
6.A valid BCD number is a 4-bit number between 0 and 9. To implement this logically, we name the bits ABCD (from left to right).If A is 0, then we have no problem because the maximum number would be 7.Now if we assume A is 1, B and C should both be 0.

Therefore we can implement the logic like this:

$$w = \bar{A} + \bar{B}\bar{C}$$

As we can see, this is the logic of OAI so we can create a BCD detector using only 1 OAI with inputs of A,B and C and the output will be w.

If w is 1,then the number is valid.



Scanned with CamScanner

Figure 20 BCD detector

7.Three different versions are shown below:

```
`timescale lns/lns
module BCD1(input a,input b,input c,output w);
        OAI1 G1(.a(a),.b(b),.c(c),.w(w));
endmodule
module BCD2(input a,input b,input c,output w);
        OAI2 G2(.a(a),.b(b),.c(c),.w(w));
endmodule
module BCD5(input a,input b,input c,output w);
        OAI5 G5(.a(a),.b(b),.c(c),.w(w));
endmodule

```

Figure 21 BCDs

8.The testbench code is shown below:

```
`timescale lns/lns
module BCDsTb();
        logic aa=0,bb=0,cc=0;
        wire ww1,ww2,ww5;
        BCD1 bcd1(.a(aa),.b(bb),.c(cc),.w(ww1));
        BCD2 bcd2(.a(aa),.b(bb),.c(cc),.w(ww2));
        BCD5 bcd5(.a(aa),.b(bb),.c(cc),.w(ww5));
        initial begin
        #12 bb=1;
        #37 cc=1;
        #39 cc=0;
        #41 aa=1;
        #43 bb=0;
        #40 aa=0;
        #40 cc=1;
        #40 aa=1;
        #25 $stop;
        end
endmodule
```

Figure 22 BCDs testbench

Here is the result of the testbench:

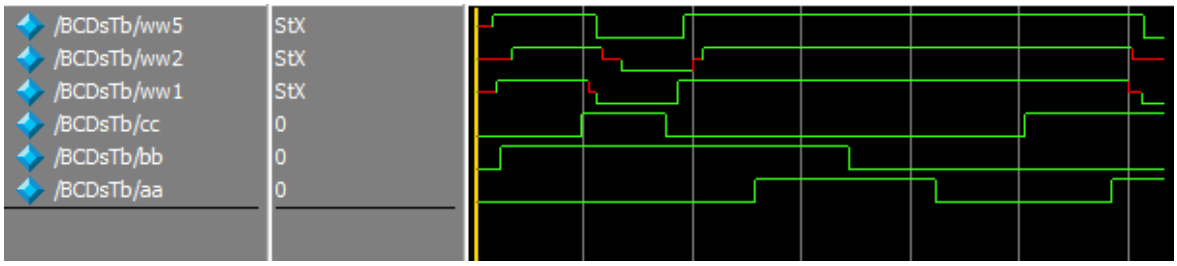


Figure 23 BCDs testbench result

Analysis: BCD1 is created in switch level design (we described OAI1 as a complex gate). This causes our detector to have X value and the simulation takes longer but the number of transistors used in this design is 6 and it is more efficient regarding power consumption. Simulating BCD2 is faster because we built it with gates. However, by using multiple gates we have sacrificed accuracy and power consumption. With a simple calculation we can

understand that BCD2 has 12 transistors but the simulation will run more smoothly. Using Verilog primitive gates in BCD5 eliminates the X value shown in BCD1 and BCD2 but it has the most power consumption due to having 14 transistors.

Codes:

```
Ln#
 1    `timescale 1ns/1ns
 2    module MyNor(input a, input b, output w);
 3            supply1 Vdd;
 4            supply0 Gnd;
 5            wire x;
 6            pmos #(5,6,7)T7(x,Vdd,a);
 7            pmos #(5,6,7)T8(w,x,b);
 8            nmos #(3,4,5)T9(w,Gnd,a);
 9            nmos #(3,4,5)T10(w,Gnd,b);
10    endmodule
```

```
`timescale 1ns/1ns
module Myinverter(input a,output w);
        supply1 Vdd;
        supply0 Gnd;
        pmos #(5,6,7) T11(w,Vdd,a);
        nmos #(3,4,5) T12(w,Gnd,a);
endmodule
```

```
Ln#
 1    `timescale 1ns/1ns
 2    module OAI1 (input a, input b, input c, output w);
 3            supply1 Vdd;
 4            supply0 Gnd;
 5            wire x,y,z;
 6            pmos #(5,6,7) T1(x,Vdd,a);
 7            pmos #(5,6,7) T2(w,x,b);
 8            pmos #(5,6,7) T3(w,Vdd,c);
 9            nmos #(3,4,5) T4(z,Gnd,a);
10            nmos #(3,4,5) T5(z,Gnd,b);
11            nmos #(3,4,5) T6(w,z,c);
12    endmodule
```

```
 1    `timescale 1ns/1ns
 2    module OAI2 (input a, input b, input c ,output w);
 3            supply1 Vdd;
 4            supply0 Gnd;
 5            wire x,y,z;
 6            MyNor G1(.a(a),.b(b),.w(x));
 7            Myinverter G2(.a(c),.w(y));
 8            MyNor G3(.a(x),.b(y),.w(z));
 9            Myinverter G4 (.a(z),.w(w));
10    endmodule
11
```

```
1   `timescale 1ns/1ns
2   module OAI1TB();
3           logic aa=0,bb=0,cc=0;
4           wire ww;
5           OAI1 UUT(.a(aa),.b(bb),.c(cc),.w(ww));
6           initial begin
7           #12 bb=1;
8           #37 cc=1;
9           #39 cc=0;
10          #41 aa=1;
11          #25 $stop;
12          end
13  endmodule
14
```

Ln#

```
1   `timescale 1ns/1ns
2   module OAI2TB();
3           logic aa=0,bb=0,cc=0;
4           wire ww;
5           OAI2 UUT(.a(aa),.b(bb),.c(cc),.w(ww));
6           initial begin
7           #12 bb=1;
8           #37 cc=1;
9           #39 cc=0;
10          #41 aa=1;
11          #25 $stop;
12          end
13  endmodule
14
```

```
1   `timescale 1ns/1ns
2   module OAI4(input a,input b,input c,output w);
3           wire x,y;
4           or #(10,12) G1(x,a,b);
5           and #(10,12) G2(y,x,c);
6           not #(10,12) G3(w,y);
7   endmodule
8
```

```
1   `timescale 1ns/1ns
2   module MultipleOAITB();
3           logic aa=0,bb=0,cc=0;
4           wire ww1,ww2,ww5;
5           OAI1 oai1(.a(aa),.b(bb),.c(cc),.w(ww1));
6           OAI2 oai2(.a(aa),.b(bb),.c(cc),.w(ww2));
7           OAI5 oai5(.a(aa),.b(bb),.c(cc),.w(ww5));
8           initial begin
9           #12 bb=1;
10          #37 cc=1;
11          #39 cc=0;
12          #41 aa=1;
13          #43 bb=0;
14          #40 aa=0;
15          #25 $stop;
16          end
17  endmodule
```

```
`timescale 1ns/1ns
module OAI5(input a,input b,input c,output w);
        wire x,y;
        or #(8,10) G1(x,a,b);
        and #(4,6) G2(y,x,c);
        not #(2,3) G3(w,y);
endmodule
```

```
`timescale 1ns/1ns
module BCD1(input a,input b,input c,output w);
        OAI1 G1(.a(a),.b(b),.c(c),.w(w));
endmodule
module BCD2(input a,input b,input c,output w);
        OAI2 G2(.a(a),.b(b),.c(c),.w(w));
endmodule
module BCD5(input a,input b,input c,output w);
        OAI5 G5(.a(a),.b(b),.c(c),.w(w));
endmodule

```

```
`timescale 1ns/1ns
module BCDsTb();
         logic aa=0,bb=0,cc=0;
         wire ww1,ww2,ww5;
         BCD1 bcd1(.a(aa),.b(bb),.c(cc),.w(ww1));
         BCD2 bcd2(.a(aa),.b(bb),.c(cc),.w(ww2));
         BCD5 bcd5(.a(aa),.b(bb),.c(cc),.w(ww5));
         initial begin
         #12 bb=1;
         #37 cc=1;
         #39 cc=0;
         #41 aa=1;
         #43 bb=0;
         #40 aa=0;
         #40 cc=1;
         #40 aa=1;
         #25 $stop;
         end
endmodule


```