# 22bec121

# Experiment – 1 Date:-11/1/2024

# Lab Work

**Q1 . Prepare all the basic gates using Data Flow Modelling. Observe the RTL and TTL design. Are they same? Change the chip and observe again and comment.**

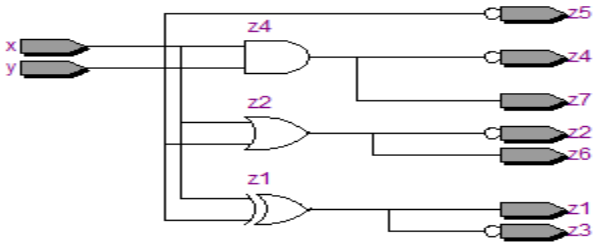
### Code

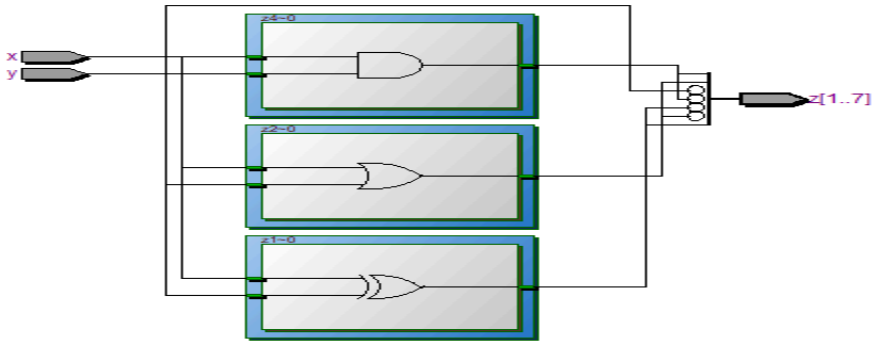
```
// Basic Gates using DataFlow Modelling
/*
module sneh1(
input x,
input y,
output z1,      // XOR Gate
output z2,      // NOR Gate
output z3,      // XNOR Gate
output z4,      // NAND Gate
output z5,      // NOT Gate
output z6,      // OR Gate
output z7       // AND Gate
);

assign z1 = x^y ;
assign z2 = ~(x|y) ;
assign z3 = ~(x^y) ;
assign z4 = ~(x&y) ;
assign z5 = ~y ;
assign z6 = x|y ;
assign z7 = x&y ;

endmodule
```

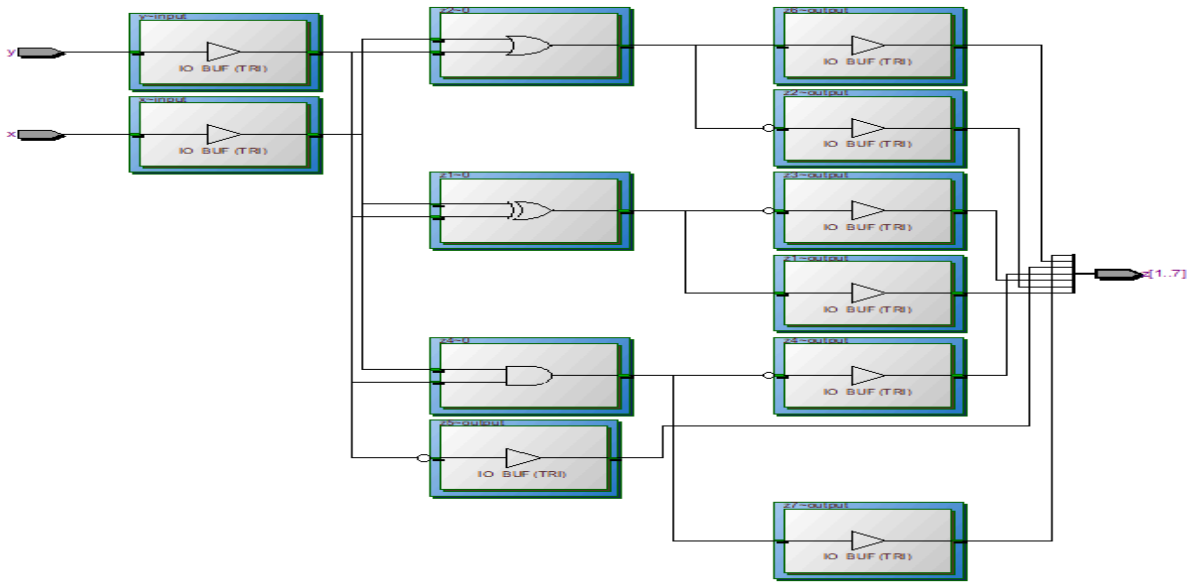
```
1   // Basic Gates using DataFlow Modelling
2
3   module snehl(
4       input x,
5       input y,
6       output z1,        // XOR Gate
7       output z2,        // NOR Gate
8       output z3,        // XNOR Gate
9       output z4,        // NAND Gate
10      output z5,        // NOT Gate
11      output z6,        // OR Gate
12      output z7         // AND Gate
13      );
14
15      assign z1 = x^y ;
16      assign z2 = ~(x|y) ;
17      assign z3 = ~(x^y) ;
18      assign z4 = ~(x&y) ;
19      assign z5 = ~y ;
20      assign z6 = x|y ;
21      assign z7 = x&y ;
22
23  endmodule
24
25
26
27
```

## Output

### 1)Cyclone II (RTL View), 2) Cyclone II (TTL View), 3) Cyclone III (TTL View)






**Q2 . Prepare all the basic gates using Behavioural Modelling. Observe the RTL and TTL design. Are they same? Compare the RTL and TTL obtained with the RTL and TTL obtained in Data Flow style of Modelling. Do you see any difference?**

## Code

```
//Basic Gates using Behavioral Modelling

module sneh1(
input x,
input y,
output reg z1,      // XOR Gate
output reg z2,      // NOR Gate
output reg z3,      // XNOR Gate
output reg z4,      // NAND Gate
output reg z5,      // NOT Gate
output reg z6,      // OR Gate
output reg z7       // AND Gate
);

always @*
begin
```

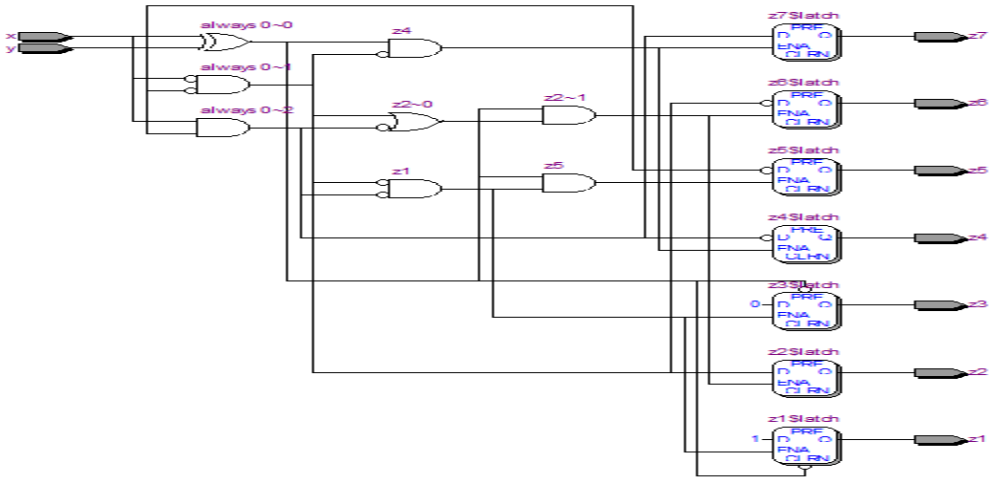
```
if (x==y)
begin
z1 = 0;
z3 = 1;
end
else if (x==0 && y==0)
begin
z2 = 1;
z6 = 0;
end
else if (x==1 && y==1)
begin
z4 = 0;
z7 = 1;
end
else
begin
z1 = 1;
z2 = 0;
z3 = 0;
z4 = 1;
z5 = ~y;
z6 = 1;
z7 = 0;
end
end
endmodule
```

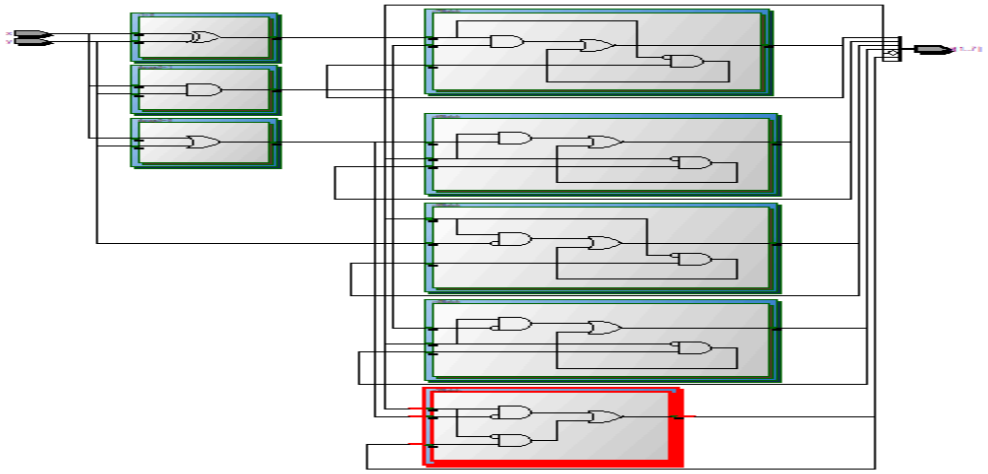
```
25    //Basic Gates using Behavioral Modelling
26
27    module snehl(
28        input x,
29        input y,
30        output reg z1,       // XOR Gate
31        output reg z2,       // NOR Gate
32        output reg z3,       // XNOR Gate
33        output reg z4,       // NAND Gate
34        output reg z5,       // NOT Gate
35        output reg z6,       // OR Gate
36        output reg z7        // AND Gate
37        );
38
39        always @*
40        begin
41            if (x==y)
42            begin
43                z1 = 0;
44                z3 = 1;
45            end
46            else if (x==0 && y==0)
47            begin
48                z2 = 1;
49                z6 = 0;
50            end
```

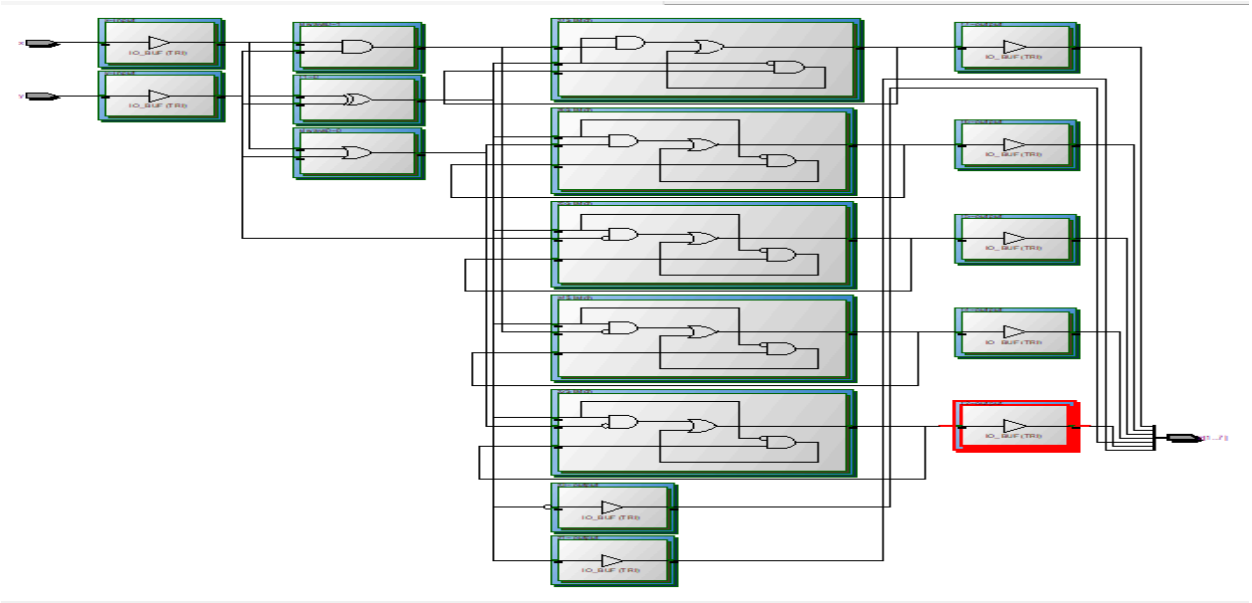
```
51          else if (x==1 && y==1)
52          begin
53              z4 = 0;
54              z7 = 1;
55          end
56          else
57          begin
58              z1 = 1;
59              z2 = 0;
60              z3 = 0;
61              z4 = 1;
62              z5 = ~y;
63              z6 = 1;
64              z7 = 0;
65          end
66      end
67  endmodule
68
69
70
```

## Output

### 1)Cyclone II (RTL View), 2) Cyclone II (TTL View), 3) Cyclone III (TTL View)





**Q3 . Prepare XOR and XNOR gates using Structural Style of Modelling. Observe the RTL and TTL design. Are they same? Compare the RTL and TTL obtained with the RTL and TTL obtained in Data Flow style of Modelling and behaviour modelling. Do you see any difference?**

```
//XOR and XNOR Gates using Structural Modelling
module not1(
input x,
output z
);

assign z = ~x;
endmodule
module and1(
input x,
input y,
output z
);

assign z = x&y;
endmodule
module or1(
input x,
input y,
```

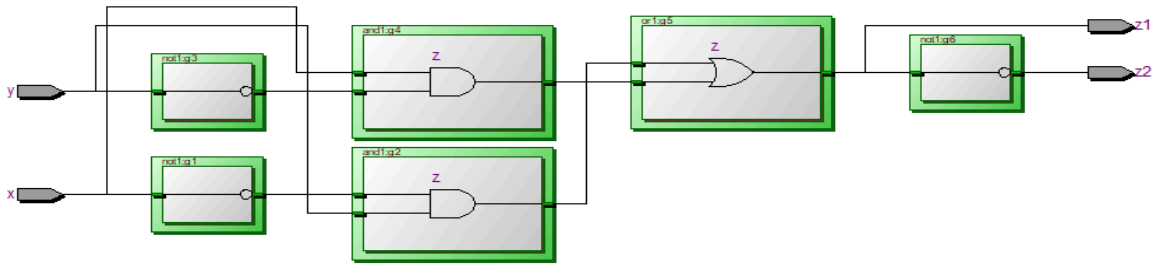
```
output z
);

assign z = x|y;
endmodule
module sneh1(
input x,
input y,
output z1,      // XOR Gate
output z2      // XNOR Gate
);

wire p,q,r,s;
not1 g1(x,p);
and1 g2(p,y,q);
not1 g3(y,r);
and1 g4(x,r,s);
or1  g5(q,s,z1);
not1 g6(z1,z2);
endmodule
```

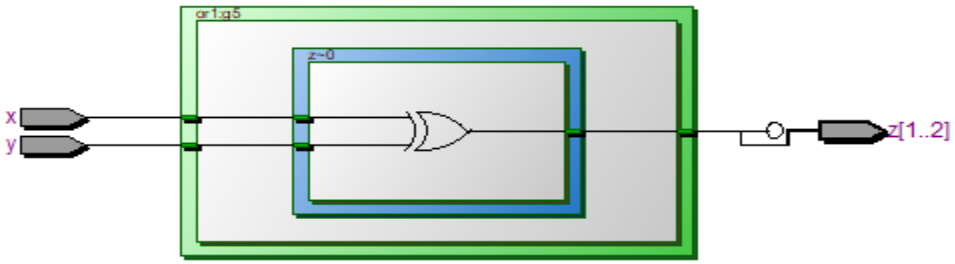
```
69    //XOR and XNOR Gates using Structural Modelling
70    module not1(
71        input x,
72        output z
73        );
74
75        assign z = ~x;
76    endmodule
77    module and1(
78        input x,
79        input y,
80        output z
81        );
82
83        assign z = x&y;
84    endmodule
85    module or1(
86        input x,
87        input y,
88        output z
89        );
90
```

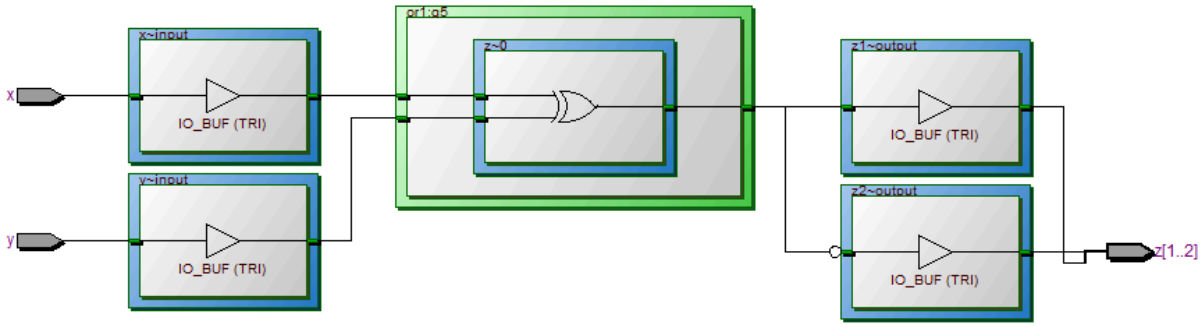
```
90
91        assign z = x|y;
92    endmodule
93    module sneh1(
94        input x,
95        input y,
96        output z1,       // XOR Gate
97        output z2       // XNOR Gate
98        );
99
100       wire p,q,r,s;
101       not1 g1(x,p);
102       and1 g2(p,y,q);
103       not1 g3(y,r);
104       and1 g4(x,r,s);
105       or1  g5(q,s,z1);
106       not1 g6(z1,z2);
107   endmodule
108
```

**Output**

**1)Cyclone II (RTL View), 2) Cyclone II (TTL View), 3) Cyclone III (TTL View)**




**Q4 . Repeat the above steps for Gate Level modelling for all the basic gates and comment.**

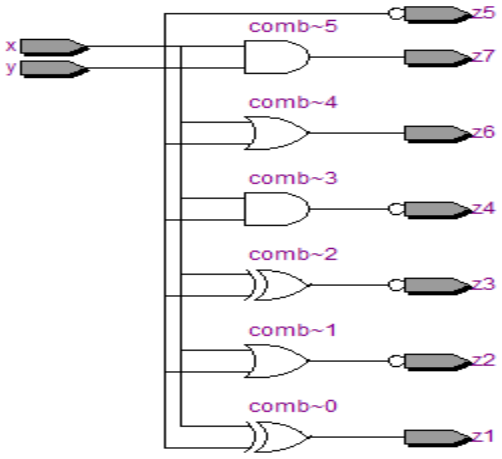
## Code

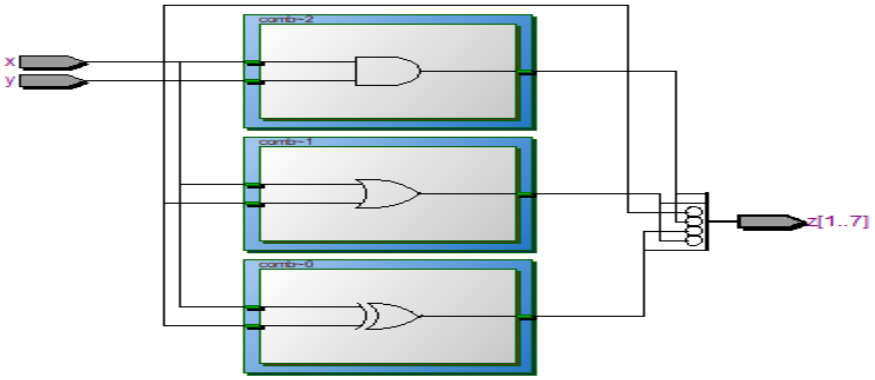
```
module sneh1(
input x,
input y,
output z1,     // XOR Gate
output z2,     // NOR Gate
output z3,     // XNOR Gate
output z4,     // NAND Gate
output z5,     // NOT Gate
output z6,     // OR Gate
output z7      // AND Gate
);

xor(z1,x,y);
nor(z2,x,y);
xnor(z3,x,y);
nand(z4,x,y);
not(z5,y);
or(z6,x,y);
and(z7,x,y);

endmodule
```

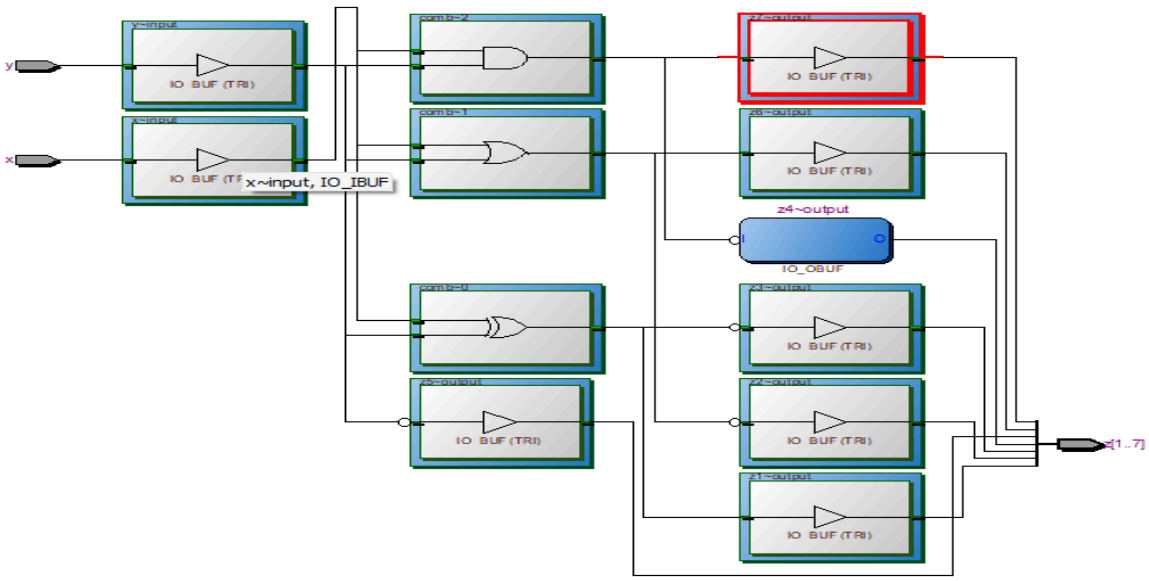
```
110   module snehl(
111       input x,
112       input y,
113       output zl,       // XOR Gate
114       output z2,       // NOR Gate
115       output z3,       // XNOR Gate
116       output z4,       // NAND Gate
117       output z5,       // NOT Gate
118       output z6,       // OR Gate
119       output z7        // AND Gate
120       );
121
122       xor(zl,x,y);
123       nor(z2,x,y);
124       xnor(z3,x,y);
125       nand(z4,x,y);
126       not(z5,y);
127       or(z6,x,y);
128       and(z7,x,y);
129
130   endmodule
131
```

## Output

## 1)Cyclone II (RTL View), 2) Cyclone II (TTL View), 3) Cyclone III (TTL View)



**Conclusion :- 1) Comments for Q1 .**

**Here , the TTL view for both the Cyclone II and Cyclone III are different due to there internal hardware and the TTL View shows the available hardware to implement the given code .**

**2) Comments for Q2 . , Q3 . and Q4 .**

**Here , the TTL View for both the Cyclone II and Cyclone III are different . In Cyclone III , the hardware includes the use of tri state buffers which are not available in the TTL View of Cyclone II Family .**

**3) In this experiment , we wrote the same logic using different Modelling Styles in Verilog and learnt about the difference that occurs in hardware for different Family of FPGA .**