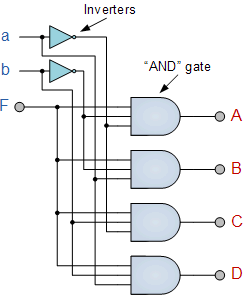
**LAB#05**

**OBJECT:** Designing of DE-MULTIPLEXER, DECODER, & ENCODER.

**TASK1:** To design 1x4 DEMUX in verilog by using Data Follow Modeling

**Circuit diagram:**

**CODING:**

//Design module

module demux4(d0,d1,d2,d3,s0,s1,din);

input s0,s1,din;

output d0,d1,d2,d3;

wire a,b;

assign a=!s0;

assign b=!s1;

assign d0=a&b&din;

assign d1=a&s1&din;

assign d2=s0&b&din;

assign d3=s0&s1&din;

endmodule

//stimulus module

module aazib;

reg s0,s1,din;

wire d0,d1,d2,d3;

demux4 d(d0,d1,d2,d3,s0,s1,din);

initial

begin

s0=1'b0; s1=1'b0; din=1'b1;

#30

s0=1'b0; s1=1'b1; din=1'b1;

#30

s0=1'b1; s1=1'b0; din=1'b1;

#30

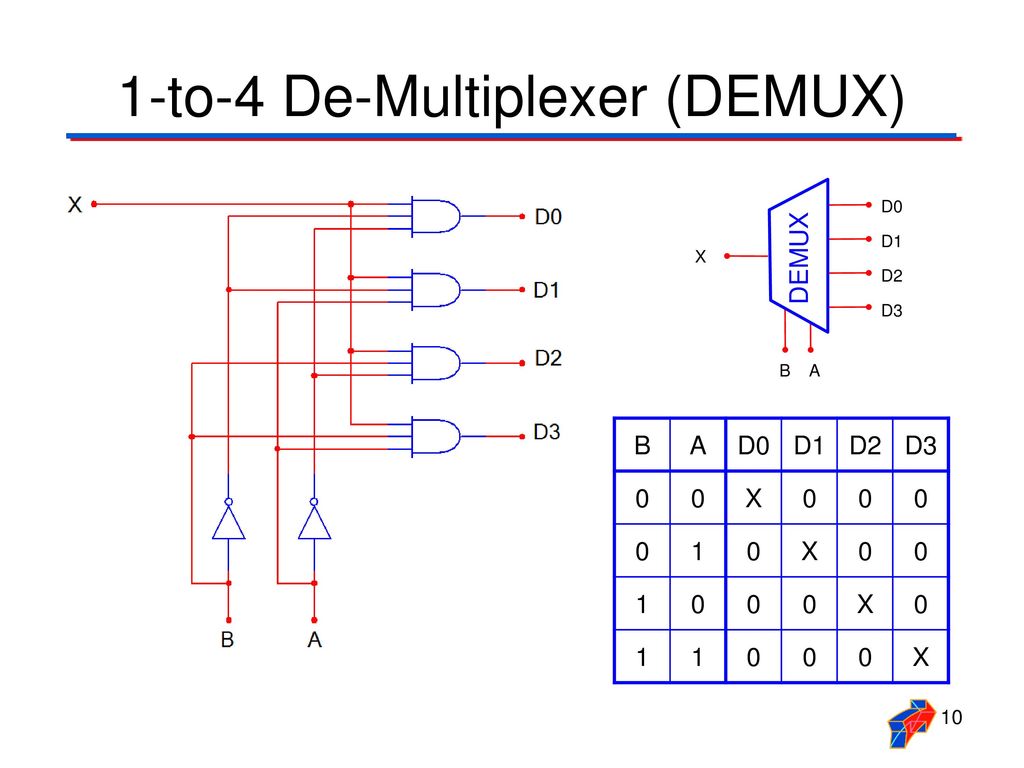
s0=1'b1; s1=1'b1; din=1'b1;

#30

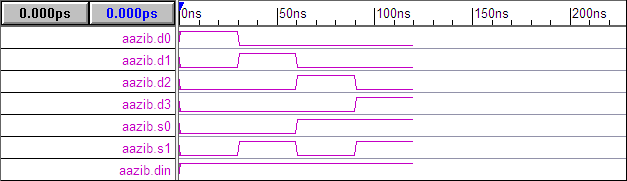
$finish;

end

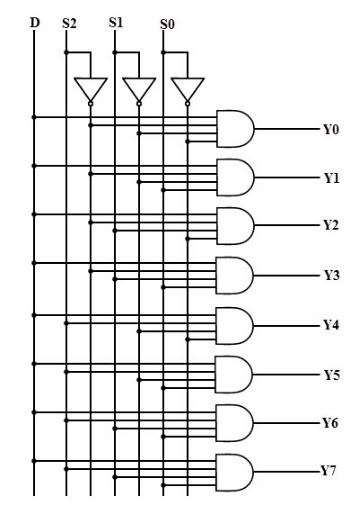
endmodule

**TRUTH TABLE:**

**RESULT:**



**TASK2:** To design 1x8 DEMUX using data flow modelling.

**CIRCUIT DIAGRAM:**

**CODING:**

//Design module

module demux4(d0,d1,d2,d3,d4,d5,d6,d7,s0,s1,s2,din);

input s0,s1,s2,din;

output d0,d1,d2,d3,d4,d5,d6,d7;

wire a,b,c;

assign a=!s0;

assign b=!s1;

assign c=!s2;

assign d0=a&b&c&din;

assign d1=a&s1&c&din;

assign d2=s0&b&c&din;

assign d3=s0&s1&c&din;

assign d4=a&b&s2&din;

assign d5=s0&b&s2&din;

assign d6=a&s1&s2&din;

assign d7=s0&s1&s2&din;

endmodule

//stimulus module

module aazib;

reg s0,s1,s2,din;

wire d0,d1,d2,d3,d4,d5,d6,d7;

demux4 d(d0,d1,d2,d3,d4,d5,d6,d7,s0,s1,s2,din);

initial

begin

s0=1'b0; s1=1'b0; din=1'b1; s2=1'b0;

#30

s0=1'b0; s1=1'b1; din=1'b1; s2=1'b0;

#30

s0=1'b1; s1=1'b0; din=1'b1; s2=1'b0;

#30

s0=1'b1; s1=1'b1; din=1'b1; s2=1'b0;

#30

s0=1'b0; s1=1'b0; din=1'b1; s2=1'b1;

#30

s0=1'b0; s1=1'b1; din=1'b1; s2=1'b1;

#30

s0=1'b1; s1=1'b0; din=1'b1; s2=1'b1;

#30

s0=1'b1; s1=1'b1; din=1'b1; s2=1'b1;

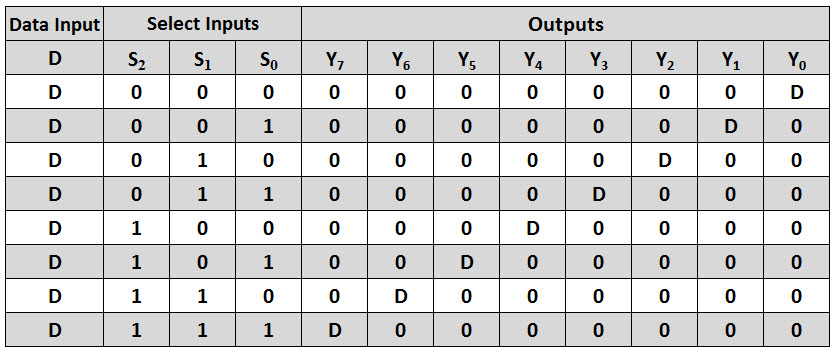
#30

$finish;

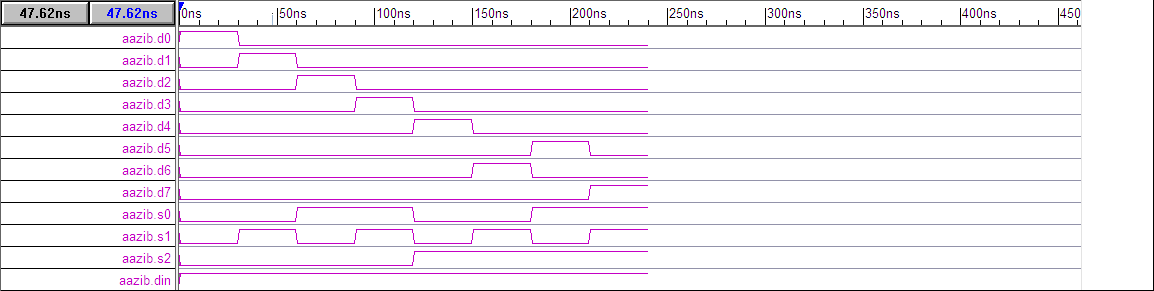
end

endmodule

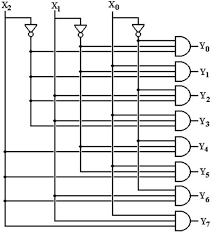
**TRUTH TABLE:**

****

**RESULT:**



**TASK3:** To design 3x8 DECODER using Data flow modelling.

**CIRCUIT DIAGRAM:**

**CODING:**

Task3:

//Design module

module demux4(d0,d1,d2,d3,d4,d5,d6,d7,s0,s1,s2);

input s0,s1,s2;

output d0,d1,d2,d3,d4,d5,d6,d7;

wire a,b,c;

assign a=!s0;

assign b=!s1;

assign c=!s2;

assign d0=a&b&c;

assign d1=a&s1&c;

assign d2=s0&b&c;

assign d3=s0&s1&c;

assign d4=a&b&s2;

assign d5=s0&b&s2;

assign d6=a&s1&s2;

assign d7=s0&s1&s2;

endmodule

//stimulus module

module aazib;

reg s0,s1,s2;

wire d0,d1,d2,d3,d4,d5,d6,d7;

demux4 d(d0,d1,d2,d3,d4,d5,d6,d7,s0,s1,s2);

initial

begin

s0=1'b0; s1=1'b0; s2=1'b0;

#30

s0=1'b0; s1=1'b1; s2=1'b0;

#30

s0=1'b1; s1=1'b0; s2=1'b0;

#30

s0=1'b1; s1=1'b1; s2=1'b0;

#30

s0=1'b0; s1=1'b0; s2=1'b1;

#30

s0=1'b0; s1=1'b1; s2=1'b1;

#30

s0=1'b1; s1=1'b0; s2=1'b1;

#30

s0=1'b1; s1=1'b1; s2=1'b1;

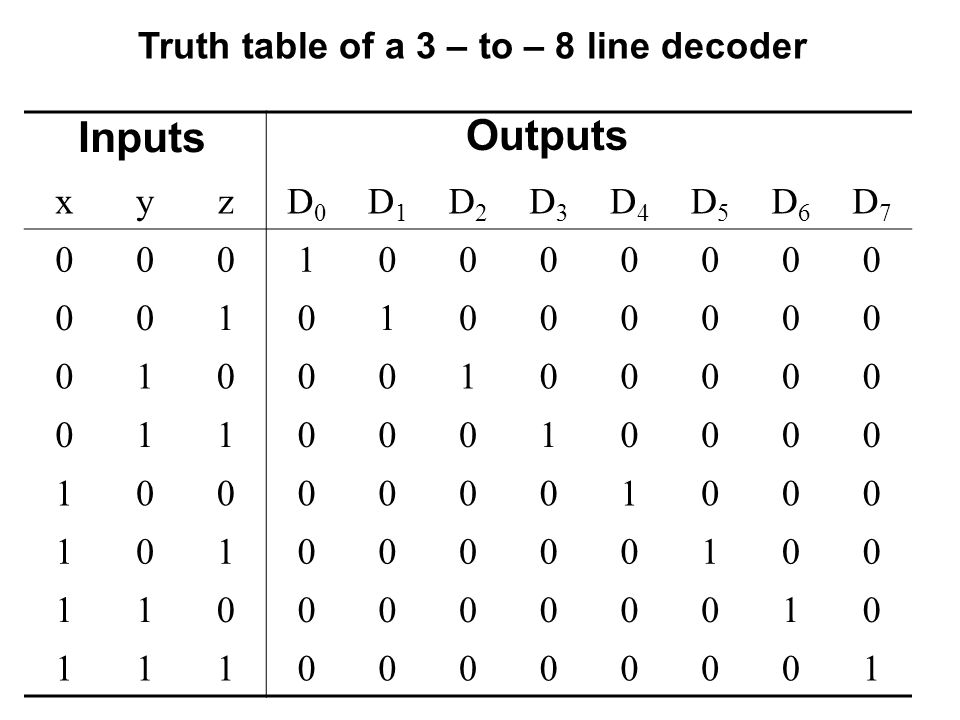
#30

$finish;

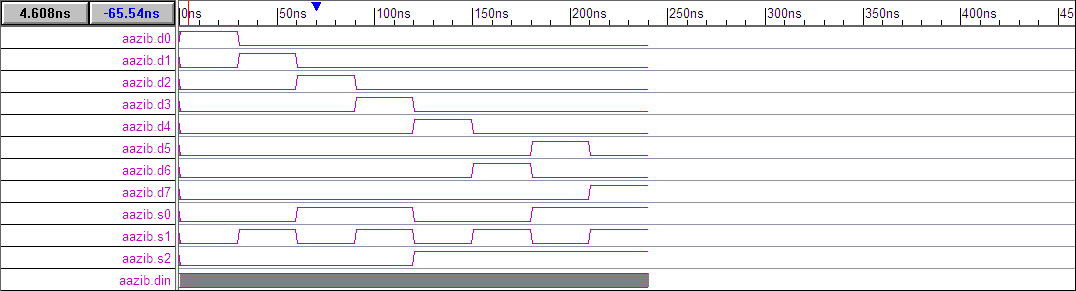
end

endmodule

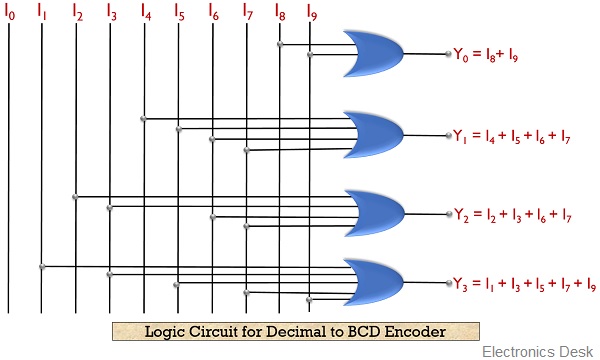
**TRUTH TABLE:**



**RESULT:**



**TASK4:** Implement Decimal to BCD ENCODER where input & output is given in truth table using Data flow modelling.

**CIRCUIT DIAGRAM:**

**CODING:**

//Design module

module bcd(a0,a1,a2,a3,x);

input [9:1]x;

output a0,a1,a2,a3;

assign a0=(x[1]|x[3]|x[5]|x[7]|x[9]);

assign a1=(x[2]|x[3]|x[6]|x[7]);

assign a2=(x[4]|x[5]|x[6]|x[7]);

assign a3=(x[8]|x[9]);

endmodule

//stimulus module

module aazib;

reg [9:1]x;

wire a0,a1,a2,a3;

bcd d1(a0,a1,a2,a3,x);

initial

begin

x=1'b1;

#30

x=4'b0010;

#30

x=4'b0011;

#30

x=4'b0100;

#30

x=4'b0101;

#30

x=4'b0110;

#30

x=4'b0111;

#30

x=4'b1000;

#30

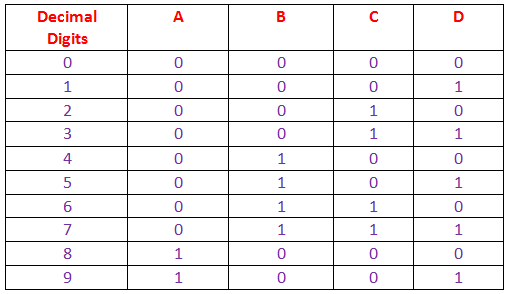
x=4'b1001;

#30

$finish;

end

endmodule

**TRUTH TABLE:**

**RESULT:**

