**Experiment 3: *8x1 Multiplexer and 1x8 Demultiplexer Implementation in VHDL.***

**Digital Systems Design Lab**

****

**Submitted To: Submitted By:**

**Dr. Rajiv Verma Gaurav Yadav**

**Assistant Professor CSE**

**IIIT Sonepat 11911038**

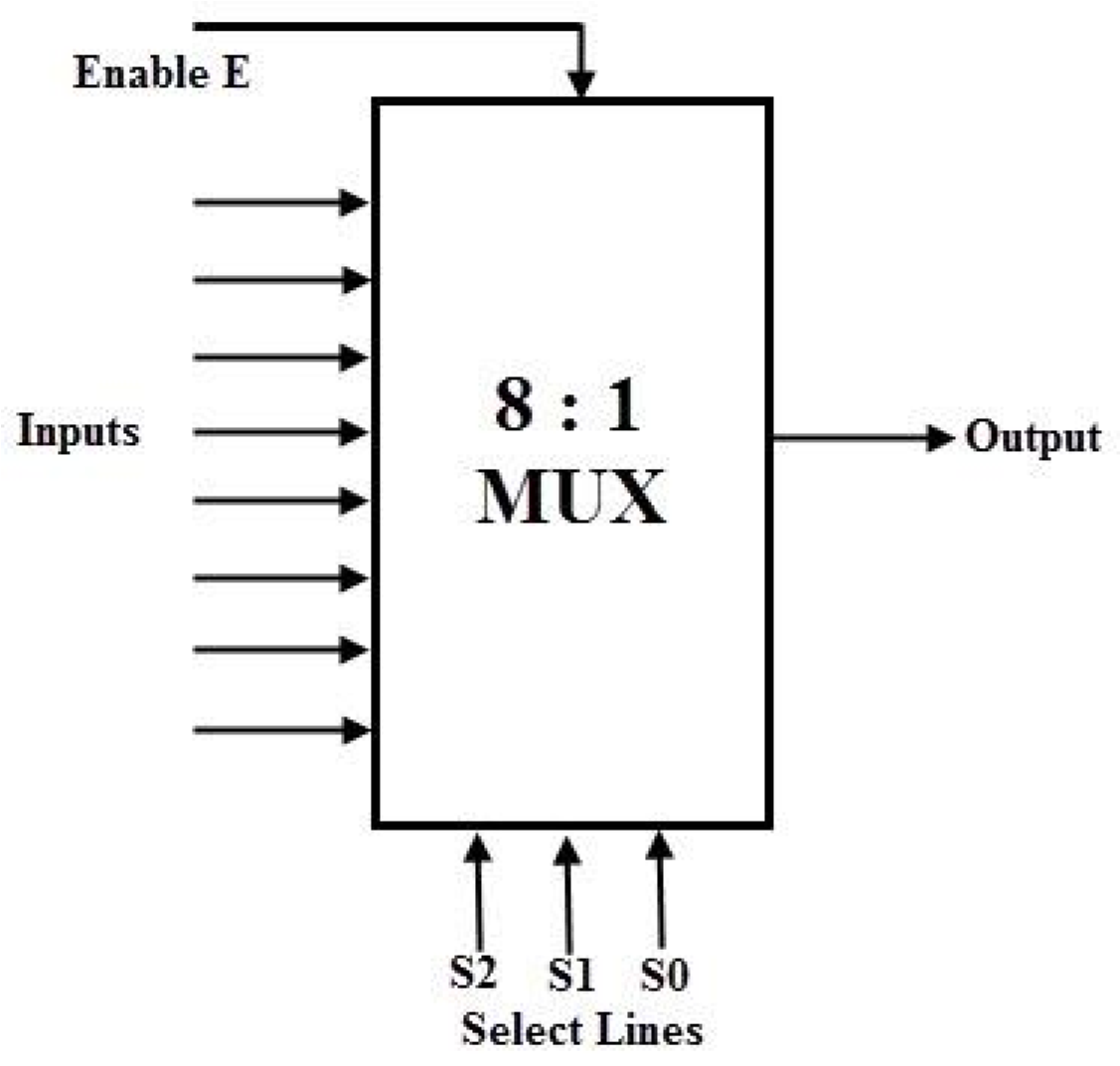
**AIM: *Implementation of 8x1 Multiplexer and 1x8 Demultiplexer in VHDL.***

**THEORY:**

* **8x1 MULTIPLEXER :-**

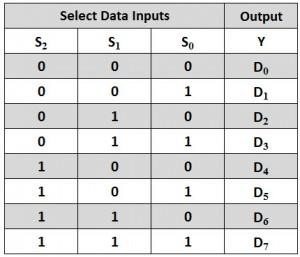
An 8 to 1 multiplexer consists of eight data inputs D0 through D7, three input select lines S2 through S0 and a single output line Y. Depending on the select lines combinations, multiplexer decodes the inputs.

The below figure shows the block diagram of an 8-to-1 multiplexer with enable input that enable or disable the multiplexer. Since the number data bits given to the MUX are eight then 3 bits (23=8) are needed to select one of the eight data bits.

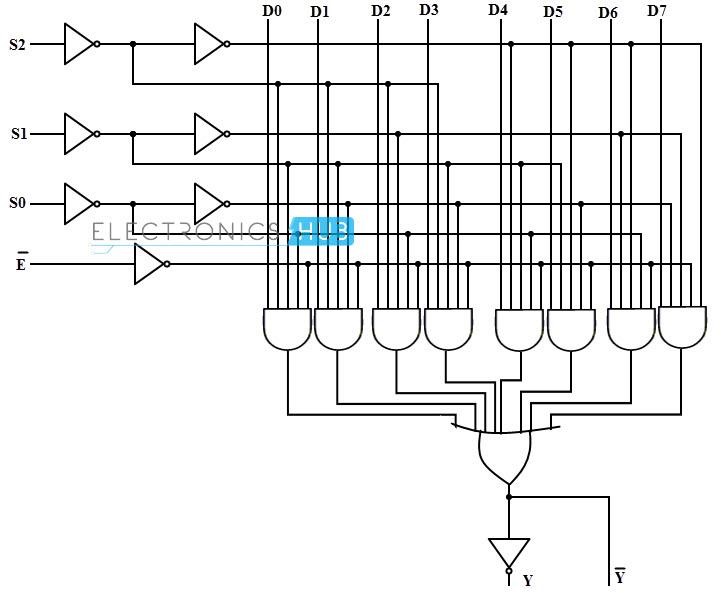
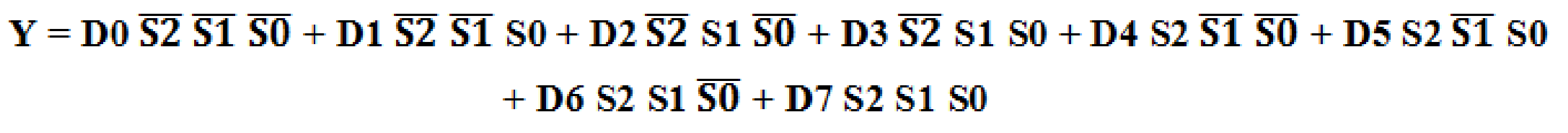


The truth table for an 8-to1 multiplexer is given below with eight combinations of inputs so as to generate each output corresponds to input.

For example, if S2= 0, S1=1 and S0=0 then the data output Y is equal to D2. Similarly the data outputs D0 to D7 will be selected through the combinations of S2, S1 and S0 as shown in below figure.



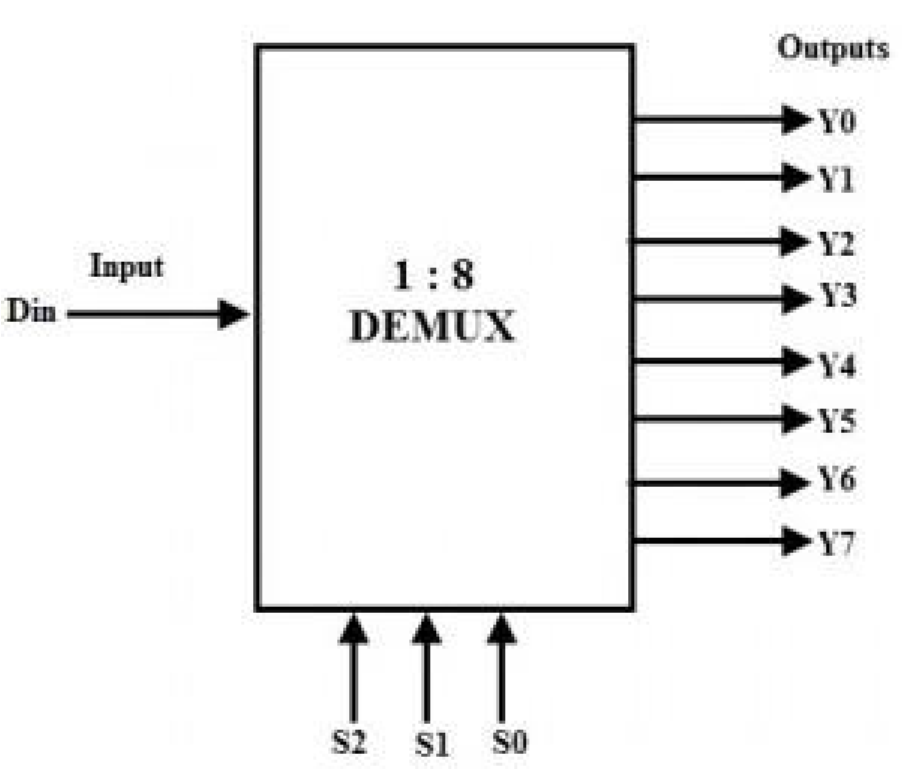
From the above truth table, the Boolean equation for the output is given as



* **8 x1 DEMULTIPLEXER : -**

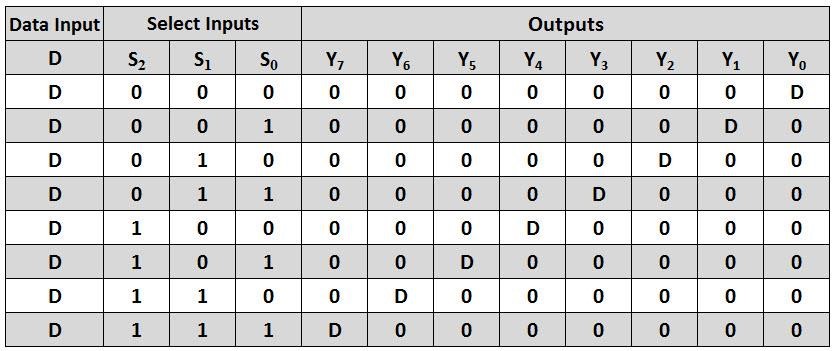
The below figure shows the block diagram of a 1 to 8 demultiplexer that consists of single input D, three select inputs S2, S1 and S0 and eight outputs from Y0 to Y7.

It is also called as 3-to-8 demultiplexer due to three select input lines. It distributes one input line to one of 8 output lines depending on the combination of select inputs.

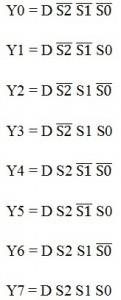


The truth table for this type of demultiplexer is shown below. The input D is connected with one of the eight outputs from Y0 to Y7 based on the select lines S2, S1 and S0.

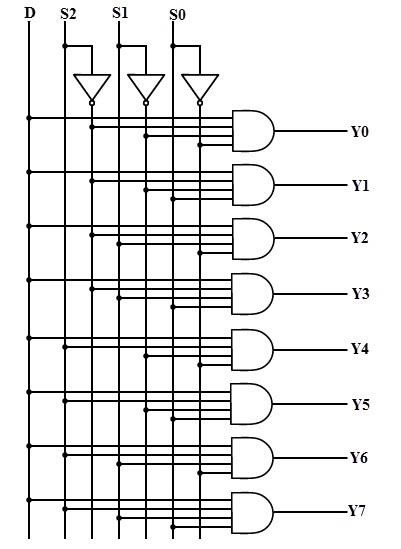
For example, if S2S1S0=000, then the input D is connected to the output Y0 and so on.



From this truth table, the Boolean expressions for all the outputs can be written as follows.



From these obtained equations, the logic diagram of this demultiplexer can be implemented by using eight AND gates and three NOT gates as shown in below figure. The different combinations of the select lines, select one AND gate at given time , such that data input will appear at a particular output.



* **VHDL Source Code for 8x1 Multiplexer:**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity multiplexer is

Port ( x:in STD\_LOGIC\_VECTOR (7 downto 0);

sel:in STD\_LOGIC\_VECTOR (2 downto 0);

y : out STD\_LOGIC);

end multiplexer;

architecture Behavioral of multiplexer is

begin

process (x,sel)

begin

case sel is

when "000"=>y<=x(0);

when "001"=>y<=x(1);

when "010"=>y<=x(2);

when "011"=>y<=x(3);

when "100"=>y<=x(4);

when "101"=>y<=x(5);

when "110"=>y<=x(6);

when "111"=>y<=x(7);

when others=> null;

end case;

end process;

end Behavioral;

* **VHDL Source Code for 1x8 Demultiplexer:**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

Entity Demux is

Port (S0: in STD\_LOGIC;

S1 :in STD\_LOGIC;

S2 :in STD\_LOGIC;

d0 :out STD\_LOGIC;

d1 :out STD\_LOGIC;

d2 :out STD\_LOGIC;

d3 :out STD\_LOGIC;

d4 :out STD\_LOGIC;

d5 :out STD\_LOGIC;

d6 :out STD\_LOGIC;

d7 :out STD\_LOGIC);

end Demux ;

Architecture behavioral of Demux is

begin

Process(S2 , S1 , S0 )

begin

d0 <= (not s2) and (not s1) and (not s0);

d1 <= (not s2) and (not s1) and s0 ;

d2 <= (not s2) and s1 and (not s0);

d3 <= (not s2) and s1 and s0 ;

d4 <= s2 and (not s1) and (not s0);

d5 <= s2 and (not s1) and s0 ;

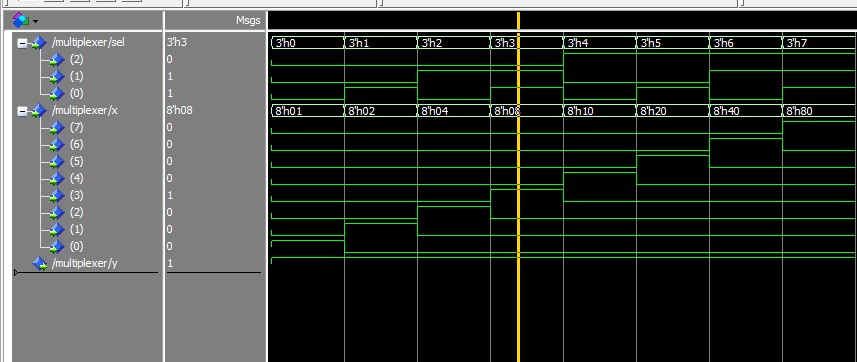
d6 <= s2 and s1 and (not s0) ;

d7 <= s2 and s1 and s0 ;

end Process;

end behavioral;

* **Output Waveform of 8x1 Multiplexer:**

****

* **Output Waveform of 1x8 Demultiplexer:**

****

* **Result : -**

From output of simulation we observe that for

* MULTIPLEXER :-

1. When all the input select lines S2, S1 and S0 are low then output Y only depend on I0.
2. When S2 and S1 are low and S0 is high then output Y only depend on I1.
3. When S2=0, S1=1 and S0=0 then output Y only depend on I2.
4. When S2=0, S1=1 and S0=1 then output Y only depend on I3.
5. When S2=1, S1=0 and S0=0 then output Y only depend on I4.
6. When S2=1, S1=0 and S0=1 then output Y only depend on I5.
7. When S2=1, S1=1 and S0=0 then output Y only depend on I6.
8. When S2=1, S1=1 and S0=1 then output Y only depend on I7.

* DEMULTIPLEXER :-

1. When all the input select lines S2, S1 and S0 are low then output D0 only depends on input I and D1, D2, D3, D4,D5, D6 and D7 all are low.
2. When S2 and S1 are low and S0 is high then only output D1 depends on input I and rest all output are must be low.
3. When S2=0, S1=1 and S0=0 then only output D2 depends on input I and rest all output are must be low.
4. When S2=0, S1=1 and S0=1 then only output D3 depends on input I and rest all output are must be low.
5. When S2=1, S1=0 and S0=0 then only output D4 depends on input I and rest all output are must be low.
6. When S2=1, S1=0 and S0=1 then output only D5 depends on input I and rest all output are must be low.
7. When S2=1, S1=1 and S0=0 then output only D6 depends on input I and rest all output are must be low.
8. When S2=1, S1=1 and S0=1 then output only D7 depends on input I and rest all output are must be low.