EXPERIMENT NO. 2

8:1 MULTIPLEXER

AIM: To write VHDL code, simulate with test bench, synthesis, implement on PLD for 8:1 MUX.

SOFTWARE TOOL: Xilinx, FPGA Kit.

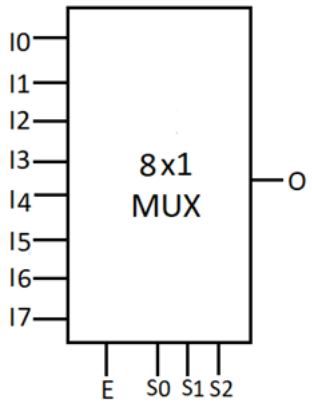
THEORY:

Multiplexing is the generic term used to describe the operation of sending one or more analogue or digital signals over a common transmission line at different times or speeds and as such, the device we use to do just that is called a **Multiplexer**.

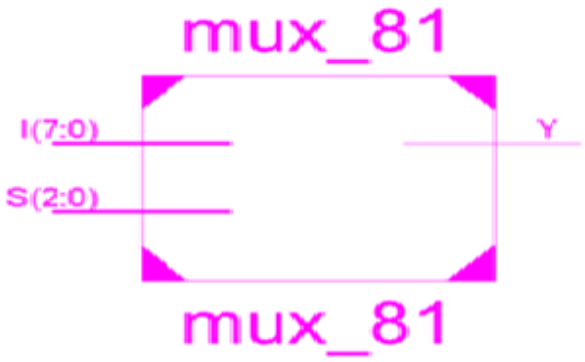
The multiplexer, shortened to "MUX" or "MPX", is a combinational logic circuit designed to switch one of several input lines through to a single common output line by the application of a control signal. Multiplexers operate like very fast acting multiple position rotary switches connecting or controlling multiple input lines called "channels" one at a time to the output. Multiplexers, or MUX's, can be either digital circuits made from high speed logic gates used to switch digital or binary data or they can be analogue types using transistors, MOSFET's or relays to switch one of the voltage or current inputs through to a single output.

Generally, the selection of each input line in a multiplexer is controlled by an additional set of inputs called *control lines* and according to the binary condition of these control inputs, either "HIGH" or "LOW" the appropriate data input is connected directly to the output. A multiplexer of 2^n inputs has n select lines, which are used to select which input line to send to the output.

8:1 MULTIPLEXER DIAGRAM:



RTL VIEW:



VHDL CODE:

1)Using Behavioral

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity mux8to1_behavioral is
   Port ( I : in  STD_LOGIC_vector(7 downto 0);
          S : in  STD_LOGIC_vector(2 downto 0);
          Y : out  STD_LOGIC);
end mux8to1_behavioral;
architecture Behavioral of mux8to1_behavioral is
begin
          process(I,S)
          begin
          if s="000" then
          Y<=I(0);
          elsif s="001" then
          Y<=I(1);
```

```
elsif s="010" then
Y<=I(2);
elsif s="011" then
Y<=I(3);
elsif s="100" then
Y<=I(4);
elsif s="101" then
Y<=I(5);
elsif s="110" then
Y<=I(6);
else Y<=I(7);
end if;
end process;
end Behavioral;
```

2)USING DATAFLOW

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity MUX8TO1DATAFLOW is
    Port (I : in  STD_LOGIC_VECTOR(7 DOWNTO 0);
          S : in  STD_LOGIC_VECTOR(2 DOWNTO 0);
          Y : out  STD_LOGIC);
end MUX8TO1DATAFLOW;

architecture DATAFLOW of MUX8TO1DATAFLOW is

begin

with S select
```

```
Y<=I(0) when "000",
I(1) when "001",
I(2) when "010",
I(3) when "011",
I(4) when "100",
I(5) when "101",
I(6) when "110",
I(7) when OTHERS;
end DATAFLOW;
```

CONCLUSION: