8:1 MUX Simple code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx primitives in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity mux is
   Port ( I : in  STD_LOGIC_VECTOR (7 downto 0);
          S : in  STD_LOGIC_VECTOR (2 downto 0);
          Y : out  STD_LOGIC);
end mux;

architecture Behavioral of mux 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;
```

8:1 mux Test bench

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--USE ieee.numeric_std.ALL;

ENTITY and_tb IS
END and_tb;

ARCHITECTURE behavior OF and_tb IS

   -- Component Declaration for the Unit Under Test (UUT)

   COMPONENT mux
   PORT(
      I : IN  std_logic_vector(7 downto 0);
      S : IN  std_logic_vector(2 downto 0);
```

```
        Y : OUT  std_logic
       );
   END COMPONENT;


  --Inputs
  signal I : std_logic_vector(7 downto 0) := (others => '0');
  signal S : std_logic_vector(2 downto 0) := (others => '0');

          --Outputs
  signal Y : std_logic;
  -- No clocks detected in port list. Replace <clock> below with
  -- appropriate port name

--   constant <clock>_period : time := 10 ns;

BEGIN

          -- Instantiate the Unit Under Test (UUT)
  uut: mux PORT MAP (
        I => I,
        S => S,
        Y => Y
       );

  -- Clock process definitions
--   <clock>_process :process
--   begin
--                  <clock> <= '0';
--                  wait for <clock>_period/2;
--                  <clock> <= '1';
```

```
--                      wait for <clock>_period/2;
--   end process;


   -- Stimulus process
   stim_proc_I: process
   begin
      I<="10101010";

     wait for 100 ns;
     end process;
   stim_proc_S: process
   begin
      S<="000";
     wait for 100 ns;
                      S<="001";
     wait for 100 ns;
                      S<="010";
     wait for 100 ns;
                      S<="011";
     wait for 100 ns;
                      S<="100";
     wait for 100 ns;
                      S<="101";
     wait for 100 ns;


                      S<="110";
     wait for 100 ns;
                      S<="111";
     wait for 100 ns;
--      wait for <clock>_period*10;
```

```
    -- insert stimulus here
    wait;
  end process;
END;
```