Ian Parker

Homework 4

1. Design a 4-to-1 multiplexer, where each input is 4 bits wide. Therefore, the output is also four bits.

VHDL Code for a 4:1 MUX

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity FourToOneMUX is

Port ( A : in STD\_LOGIC\_VECTOR(3 downto 0);

B : in STD\_LOGIC\_VECTOR(3 downto 0);

C : in STD\_LOGIC\_VECTOR(3 downto 0);

D : in STD\_LOGIC\_VECTOR(3 downto 0);

S0 : in STD\_LOGIC;

S1 : in STD\_LOGIC;

F : out STD\_LOGIC\_VECTOR(3 downto 0));

end FourToOneMUX;

architecture Behavioral of FourToOneMUX is

begin

process(A,B,C,D,S0,S1) is

begin

if(S0 = '0' AND S1 = '0')then

F <= A;

elsif(S0= '1' AND S1 = '0')then

F <= B;

elsif(S0 = '0' AND S1 = '1')then

F <= C;

else

F <= D;

end if;

end process;

end Behavioral;

TESTBENCH 1 for 4:1 MUX

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

use ieee.numeric\_std.all;

entity TB\_FourToOneMUX is

end TB\_FourToOneMUX;

architecture test of TB\_FourToOneMUX is

--INPUTS

signal A, B, C, D: STD\_LOGIC\_VECTOR(3 downto 0);

signal S0, S1 : STD\_LOGIC;

--OUTPUT

signal F : STD\_LOGIC\_VECTOR(3 downto 0);

--COMPONENT DECLARATION

component FourToOneMUX is

port( A,B,C,D : in STD\_LOGIC\_VECTOR(3 downto 0);

S0,S1 : in STD\_LOGIC;

F : out STD\_LOGIC\_VECTOR(3 downto 0));

end component;

-- INSTANTIATE the Module (Unit Under Test)

begin

UUT : FourToOneMUX PORT MAP (A,B,C,D,S0,S1,F);

process

begin

--WAIT TIME

wait for 100 ns;

A <= "0001";

B <= "0010";

C <= "0011";

D <= "0100";

--CASES

S1 <= '0'; S0 <= '0'; wait for 100 ns;

S1 <= '0'; S0 <= '1'; wait for 100 ns;

S1 <= '1'; S0 <= '0'; wait for 100 ns;

S1 <= '1'; S0 <= '1'; wait for 100 ns;

end process;

end test;

TESTBENCH 2 for 4:1 MUX

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

use ieee.numeric\_std.all;

entity TB\_FourToOneMUX2 is

end TB\_FourToOneMUX2;

architecture test of TB\_FourToOneMUX2 is

--INPUTS

signal A, B, C, D: STD\_LOGIC\_VECTOR(3 downto 0);

signal S0, S1 : STD\_LOGIC;

--OUTPUT

signal F : STD\_LOGIC\_VECTOR(3 downto 0);

--COMPONENT DECLARATION

component FourToOneMUX is

port( A,B,C,D : in STD\_LOGIC\_VECTOR(3 downto 0);

S0,S1 : in STD\_LOGIC;

F : out STD\_LOGIC\_VECTOR(3 downto 0));

end component;

-- INSTANTIATE the Module (Unit Under Test)

begin

UUT : FourToOneMUX PORT MAP (A,B,C,D,S0,S1,F);

process

begin

--WAIT TIME

wait for 100 ns;

A <= "0010";

B <= "0011";

C <= "0100";

D <= "0101";

--CASES

S1 <= '0'; S0 <= '0'; wait for 100 ns;

S1 <= '0'; S0 <= '1'; wait for 100 ns;

S1 <= '1'; S0 <= '0'; wait for 100 ns;

S1 <= '1'; S0 <= '1'; wait for 100 ns;

end process;

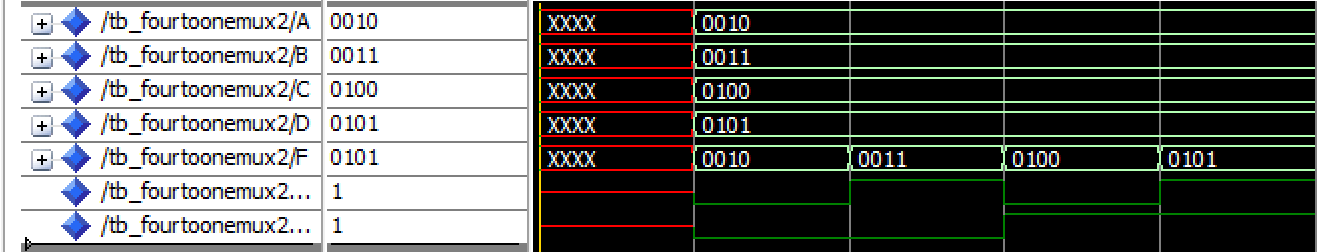
end test;

QUESTA SIMULATION FOR 4:1 MUX

1



2



1. Use the multiplexer created in problem 1 as a component in a new system that accepts two four-bit inputs and has one four-bit output. The new system performs 4 different operations on the two inputs. The results of these operations are tied to the four inputs of the multiplexer. See picture below. You are to choose the 4 operations performed. However, they should be distinct, use 2 4-bit inputs and produce a 4-bit result.

VHDL CODE for Operations feeding into 4:1 MUX

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity OperationsToMUX is

Port ( Input1,Input2: in STD\_LOGIC\_VECTOR(3 downto 0);

Sel0, Sel1: in STD\_LOGIC;

Output: out STD\_LOGIC\_VECTOR(3 downto 0));

end;

architecture Behavioral of OperationsToMUX is

--SIGNALS for Operations

signal Operation1Temp,Operation2Temp,Operation3Temp,Operation4Temp: STD\_LOGIC\_VECTOR( 3 downto 0);

--COMPONENT DECLARATION

component FourToOneMUX is

port( A,B,C,D : in STD\_LOGIC\_VECTOR(3 downto 0);

S0,S1 : in STD\_LOGIC;

F : out STD\_LOGIC\_VECTOR(3 downto 0));

end component;

begin

--Operation 1(OR)

Operation1Temp <= (Input1 OR Input2);

--Operation 2(NOR)

Operation2Temp <= (Input1 NOR Input2);

--Operation 3(AND)

Operation3Temp <= (Input1 AND Input2);

--Operation 4(NAND)

Operation4Temp <= (Input1 NAND Input2);

-- INSTANTIATE the Component (MUX)

MUX : FourToOneMUX PORT MAP ( A => Operation1Temp,

B => Operation2Temp,

C => Operation3Temp,

D => Operation4Temp,

S0 => Sel0,

S1 => Sel1,

F => Output);

end Behavioral;

TESTBENCH 1 for Operations feeding into 4:1 MUX

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

use ieee.numeric\_std.all;

entity TB\_OperationstoMUX is

end TB\_OperationstoMUX;

architecture test of TB\_OperationstoMUX is

--INPUTS

signal A, B : STD\_LOGIC\_VECTOR(3 downto 0);

signal S0, S1 : STD\_LOGIC;

--OUTPUT

signal F : STD\_LOGIC\_VECTOR(3 downto 0);

--COMPONENT DECLARATION

component OperationsToMUX is

port( Input1,Input2 : in STD\_LOGIC\_VECTOR(3 downto 0);

Sel0,Sel1 : in STD\_LOGIC;

Output : out STD\_LOGIC\_VECTOR(3 downto 0));

end component;

-- INSTANTIATE the Module (Unit Under Test)

begin

UUT : OperationsToMUX PORT MAP (A,B,S0,S1,F);

process

begin

--WAIT TIME

wait for 100 ns;

A <= "0001";

B <= "0010";

--CASES

S1 <= '0'; S0 <= '0'; wait for 100 ns;

S1 <= '0'; S0 <= '1'; wait for 100 ns;

S1 <= '1'; S0 <= '0'; wait for 100 ns;

S1 <= '1'; S0 <= '1'; wait for 100 ns;

end process;

end test;

TESTBENCH 2 for Operations feeding into 4:1 MUX

library ieee;

use ieee.std\_logic\_1164.all;

use ieee.std\_logic\_unsigned.all;

use ieee.numeric\_std.all;

entity TB\_OperationstoMUX2 is

end TB\_OperationstoMUX2;

architecture test of TB\_OperationstoMUX2 is

--INPUTS

signal A, B : STD\_LOGIC\_VECTOR(3 downto 0);

signal S0, S1 : STD\_LOGIC;

--OUTPUT

signal F : STD\_LOGIC\_VECTOR(3 downto 0);

--COMPONENT DECLARATION

component OperationsToMUX is

port( Input1,Input2 : in STD\_LOGIC\_VECTOR(3 downto 0);

Sel0,Sel1 : in STD\_LOGIC;

Output : out STD\_LOGIC\_VECTOR(3 downto 0));

end component;

-- INSTANTIATE the Module (Unit Under Test)

begin

UUT : OperationsToMUX PORT MAP (A,B,S0,S1,F);

process

begin

--WAIT TIME

wait for 100 ns;

A <= "0011";

B <= "0100";

--CASES

S1 <= '0'; S0 <= '0'; wait for 100 ns;

S1 <= '0'; S0 <= '1'; wait for 100 ns;

S1 <= '1'; S0 <= '0'; wait for 100 ns;

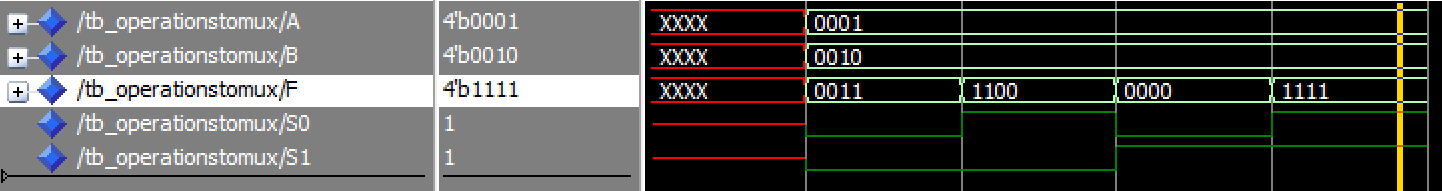
S1 <= '1'; S0 <= '1'; wait for 100 ns;

end process;

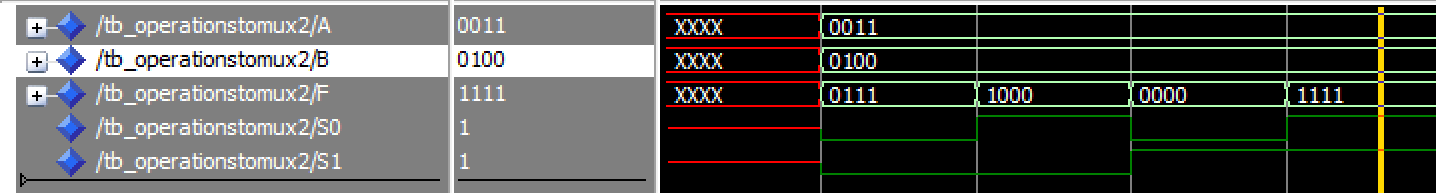
end test;

QUESTA SIMULATION FOR Operations feeding into 4:1 MUX

1



2



YOUTUBE LINK

https://youtu.be/yYbvHWcgkQ0