INSTITUTO
FEDERAL
Paraíba

Curso Superior de Bacharelado em Engenharia Elétrica

Disciplina: Sistemas Digitais
Professor: Lincoln Machado de Araújo

Exercício para quarta semana de atividades remotas

1 -(4 pontos) Desenvolva o circuito multiplicador de 4 bits descrito no final do vídeo intitulado “Binary Multiplication” capaz de operar também com números negativos (complemento de 2). Valide seus funcionamento através de um testbench testando a multiplicação de dois números positivos, dois números negativos e dois números com sinais diferentes. (Cole o seu código aqui nesse documento e o endereço do EDA Playground)

Professor, eu não consegui identificar o erro que está ocorrendo nas bibliotecas fullAdder e halfAdder

link do código:
[Edit code - EDA Playground]

DESING

```
------------------------------
library IEEE;
use IEEE.std_logic_1164.all;

entity multiplier is
port(a, b: in std_logic_vector(3 downto 0);
    s: out std_logic_vector(7 downto 0));
end multiplier;

architecture nu of multiplier is

        type grid is array(integer range <>) of std_logic_vector(3 downto 0);

   component fullAdder is
        port(a,b,cin: in std_logic;
         s, cout: out std_logic);
   end component;

   component halfAdder is
        port(a,b: in std_logic;
         s, cout: out std_logic);
   end component;

   signal sum: grid (2 downto 0);
   signal cout: grid (2 downto 0);
   signal ss, cc: std_logic;
begin


--Linha 1
adder00 : halfAdder port map( a(1)and b(0), a(0) and b(1), sum(0)(0), cout (0)(0));
adder01 : fullAdder port map( a(2) and b(0), a(1) and b(1), cout(0)(0), sum(0)(1), cout(0)(1));
adder02 : fullAdder port map( not(a(3) and b(0)), a(2) and b(1), cout(0)(1), sum(0)(2),
cout(0)(2));
adder03 : fullAdder port map (a(3) nand b(1),                    '1',        cout(0)(2), sum(0)(3),
cout(0)(3));

--Linha 2
adder10 : halfAdder port map( a(0) and b(2), sum(0)(1), sum(1)(0), cout(1)(0));
adder11 : fullAdder port map( a(1) and b(2), sum(0)(2), cout(1)(0), sum(1)(1), cout(1)(1));
adder12 : fullAdder port map( a(2) and b(2), sum(0)(3), cout(1)(1), sum(1)(2), cout(1)(2));
adder13 : fullAdder port map( not(a(3) and b(2)), cout(0)(3), cout(1)(2), sum(1)(3),
cout(1)(3));
```

```
--Linha 3
adder20 : halfAdder port map( not(a(0) and b(3)), sum(1)(1), sum(2)(0), cout(2)(0));
adder21 : fullAdder port map( not(a(1) and b(3)), sum(1)(2), cout(2)(0), sum(2)(1),
cout(2)(1));
adder22 : fullAdder port map( not(a(2) and b(3)), sum(1)(3), cout(2)(1), sum(2)(2),
cout(2)(2));
adder23 : fullAdder port map( a(3) and b(3), cout(1)(3), cout(2)(2), sum(2)(3), cout(2)(3));

--Saída

s(0) <= a(0) and b(0);
s(1) <= sum(0)(0);
s(2) <= sum(1)(0);
s(6 downto 3) <= sum(2)(3 downto 0);
adderf : halfAdder port map ('1', cout(2)(3), ss, cc);

   s(7) <= ss;

end nu;




---------------------------------------------------------------------------------------
--fullAdder

library IEEE;
use IEEE.std_logic_1164.all;

entity fullAdder is
        port(a, b, cin:in std_logic;
         s,   cout:out std_logic);
end fullAdder;

architecture fa of fullAdder is

        signal aux: std_logic;

begin
        aux <= a xor b;
        s <= aux xor cin;
   cout <= (a and b) or (aux and cin);

end fa;
```

```
--------------------------------------------------------------------------------


--halfAdder
library IEEE;
use IEEE.std_logic_1164.all;

entity halfAdder is
        port(a, b: in std_logic;
      s, cout: out std_logic);
end halfAdder;

architecture ha of halfAdder is
begin

        cout <= a and b;
   s <= a xor b;

end ha;


--------------------------------------------------------------------------------

TESTBENCH

library IEEE;
use IEEE.std_logic_1164.all;

entity testbench is
end testbench;

architecture tb of testbench is
        component fullAdder is
        port(a,b,cin: in std_logic;
         s, cout: out std_logic);
   end component;
   component multiplier is
                port( a, b: in std_logic_vector(3 downto 0);
              s: out std_logic_vector(7 downto 0));
        end component;

   signal a, b: std_logic_vector(3 downto 0) := "0000";
   signal s: std_logic_vector(7 downto 0) := x"00";
```

```
begin

            dut: multiplier port map(a, b, s);

    a <= "0101",
             "0110" after 100 ns,
                         "1001" after 200 ns,
                         "1110" after 300 ns,
                         "1101" after 400 ns,
         "1100" after 500 ns,
         "0010" after 600 ns;

            b <= "0000",
             "0011" after 100 ns,
                         "1001" after 200 ns,
                         "1110" after 300 ns,
                         "0111" after 400 ns,
         "0001" after 500 ns,
         "0100" after 600 ns;

end architecture;
```