4 bit Adder VHDL models

Tuesday, March 23, 2021 4:14 PM

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.NUMERIC_STD.all;
entity adder4 is
     port (op0: in std_logic_vector(3 downto 0);
           op1: in std_logic_vector(3 downto 0);
           cin: in std_logic;
           s:   out std_logic_vector(3 downto 0);
           cout: out std_logic);
end adder4;
-- Several architectures. Only the last one is synthetized.
architecture Structural of adder4 is
signal i_carry: std_logic_vector(2 downto 0);
begin
     FA0: entity work.FullAdder(DataFlow)
          port map (ai => op0(0),
                    bi => op1(0),
                          ci => cin,
                          sum => s(0),
                          co => i_carry(0));
     FA1: entity work.FullAdder(DataFlow)
          port map (ai => op0(1),
                    bi => op1(1),
                          ci => i_carry(0),
                          sum => s(1),
                          co => i_carry(1));
     FA2: entity work.FullAdder(DataFlow)
          port map (ai => op0(2),
                    bi => op1(2),
                          ci => i_carry(1),
                          sum => s(2),
                          co => i_carry(2));
     FA3: entity work.FullAdder(DataFlow)
          port map (ai => op0(3),
                    bi => op1(3),
                          ci => i_carry(2),
                          sum => s(3),
                          co => cout);
end Structural;
architecture DataFlow of adder4 is
signal i_carry: std_logic_vector(2 downto 0);
begin
          s(0)<= op0(0) xor op1(0) xor cin;
  i_carry(0) <= (op0(0) and op1(0)) or (cin and (op0(0) or op1(0))) ;

        s(1)<= op0(1) xor op1(1) xor i_carry(0);
  i_carry(1) <= (op0(1) and op1(1)) or (i_carry(0) and (op0(1) or op1(1))) ;

        s(2)<= op0(2) xor op1(2) xor i_carry(1);
  i_carry(2) <= (op0(2) and op1(2)) or (i_carry(1) and (op0(2) or op1(2))) ;

        s(3)<= op0(3) xor op1(3) xor i_carry(2);
       cout <= (op0(3) and op1(3)) or (i_carry(2) and (op0(3) or op1(3))) ;
end DataFlow;
architecture Arithm of Adder4 is
signal s_operand0, s_operand1, s_result : unsigned(4 downto 0);
begin
    s_operand0 <= '0' & unsigned(op0);
    s_operand1 <= '0' & unsigned(op1);
    s_result   <= s_operand0 + s_operand1 + 1 when cin ='1' else
                  s_operand0 + s_operand1;
             s <= std_logic_vector(s_result(3 downto 0));
         cout  <= s_result(4);
end Arithm;
```

Example Simulation