ITI Summer Training

LAB 1_VHDL

1- Behavioural

```
5   LIBRARY std;
6
7   USE IEEE.STD_LOGIC_1164.ALL;
8
9   USE ieee.numeric_bit.ALL;
10
11  USE IEEE.std_logic_arith.all;
12
13  USE IEEE.numeric_std.all;
14
15  USE IEEE.std_logic_signed.all;
16
17  use std.textio.all;
18
19
20  entity ADD_SUB_circuit is
21    port (
22      A, B: in std_logic_vector(3 downto 0);
23      Mode: in std_logic;
24      Cout: out std_logic;
25      S: out std_logic_vector(3 downto 0)
26    );
27  end entity ADD_SUB_circuit;
28
29  architecture behav of ADD_SUB_circuit is
30
31  begin
32    p1: process (A, B, Mode)
33    variable Temp: std_logic_vector(4 downto 0);
34    begin
35      if Mode = '0' then
36        Temp := ('0' & A) + B;
37      else
38        Temp := ('0' & A) - B;
39      end if;
40
41      S <= Temp(3 downto 0);
42      Cout <= Temp(4);
43    end process p1;
44  end architecture behav;
```

2- Data flow

```vhdl
entity ADD_SUB_circuit is
  port (
    A, B: in std_logic_vector(3 downto 0);
    Mode: in std_logic;
    Cout: out std_logic;
    S: out std_logic_vector(3 downto 0)
  );
end entity ADD_SUB_circuit;

architecture behav of ADD_SUB_circuit is

begin


    S <= A + B when Mode = '0' else A - B;
    Cout <= '1' when ('0' & A) < B(3 downto 0) else '0';

end architecture behav;

```

3- Structural

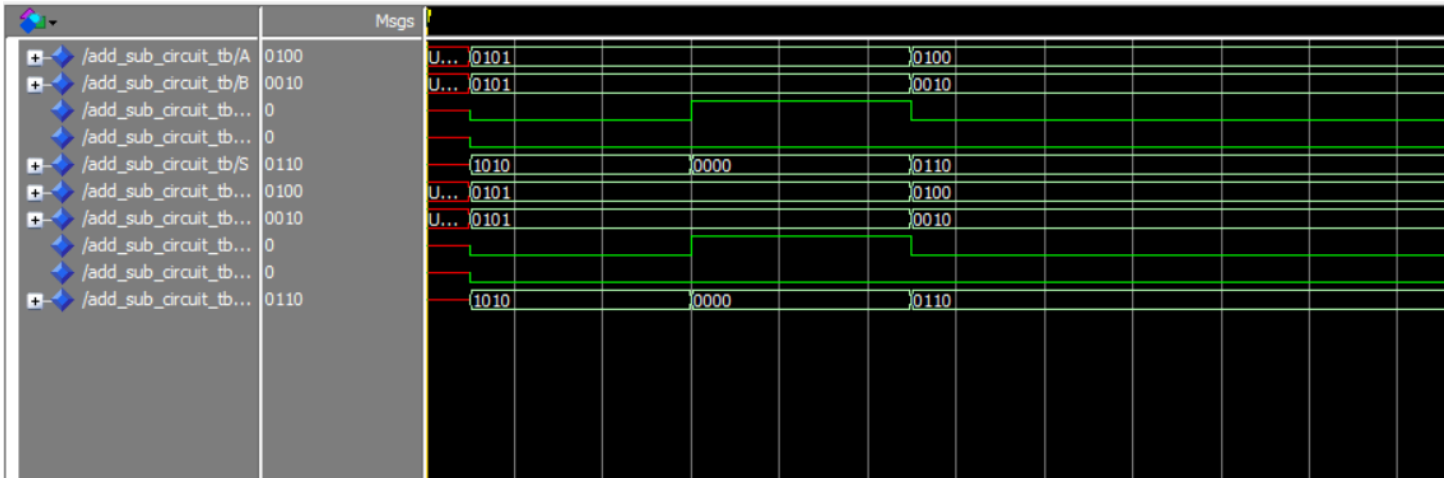
```vhdl
USE IEEE.std_logic_signed.all;

use std.textio.all;


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;



entity Full_Adder is
        Port (
            X, Y, Cin : in  STD_LOGIC;
            S, Cout   : out STD_LOGIC
        );
    end entity Full_Adder;

architecture behavioral of Full_Adder is
begin
    S <= X xor Y xor Cin;
    Cout <= (X and Y) or (Cin and (X xor Y));
end behavioral;

```

```vhdl
entity ADD_SUB_circuit is
    Port (
        A    : in  STD_LOGIC_VECTOR(3 downto 0);
        B    : in  STD_LOGIC_VECTOR(3 downto 0);
        Mode : in  STD_LOGIC;  -- Control signal: '1' for subtraction, '0' for addition
        Cout : out STD_LOGIC;  -- Carry-out signal
        S    : out STD_LOGIC_VECTOR(3 downto 0)  -- Sum/difference output
    );
end entity ADD_SUB_circuit;

architecture structural of ADD_SUB_circuit is
    component Full_Adder is
        Port (
            X, Y, Cin : in  STD_LOGIC;
            S, Cout   : out STD_LOGIC
        );
    end component;

    signal C0, C1, C2, C3, C4 : STD_LOGIC;
    signal S0, S1, S2, S3     : STD_LOGIC;
begin
    FA0: Full_Adder port map (A(0), B(0) xor Mode, Mode, S(0), C0);
    FA1: Full_Adder port map (A(1), B(1) xor Mode, C0, S(1), C1);
    FA2: Full_Adder port map (A(2), B(2) xor Mode, C1, S(2), C2);
    FA3: Full_Adder port map (A(3), B(3) xor Mode, C2, S(3), C3);

    Cout <= C3;
end architecture structural;
```

-Testbench-

```
53  entity ADD_SUB_circuit_TB is
54  end entity ADD_SUB_circuit_TB;
55
56  architecture TB of ADD_SUB_circuit_TB is
57      COMPONENT ADD_SUB_circuit
58          port (
59              A, B  : in  std_logic_vector(3 downto 0);
60              Mode  : in  std_logic;
61              Cout  : out std_logic;
62              S     : out std_logic_vector(3 downto 0)
63          );
64      END COMPONENT;
65
66      signal A, B   : std_logic_vector(3 downto 0);
67      signal Mode   : std_logic;
68      signal Cout   : std_logic;
69      signal S      : std_logic_vector(3 downto 0);
70
71  begin
72      dut: ADD_SUB_circuit
73          port map (
74              A => A,
75              B => B,
76              Mode => Mode,
77              S => S,
78              Cout => Cout
79          );
80
81      stimul: process
82
83
84      file in_file  : text open read_mode is "D:/in_file.txt";
85      file out_file : text open write_mode is "D:/out_file.txt";
86      variable input_line : line;
87      variable output_line : line;
88      variable A_f, B_f   : std_logic_vector(3 downto 0);
89      variable Mode_f     : std_logic;
90      variable Cout_f     : std_logic;
91      variable S_f        : std_logic_vector(3 downto 0);
```

```
93      begin
94          while not endfile(in_file) loop
95              wait for 5 ns;
96              readline(in_file, input_line);
97              read(input_line, A_f);
98              read(input_line, B_f);
99              read(input_line, Mode_f);
100
101             A <= A_f;
102             B <= B_f;
103             Mode <= Mode_f;
104
105             wait for 15 ns;  -- Ensure the signals have time to propagate
106
107             Cout_f := Cout;
108             S_f := S;
109
110             write(output_line, string'("Time = "));
111             write(output_line, now);
112             write(output_line, string'("  in1 = "));
113             write(output_line, A_f);
114             write(output_line, string'("  in2 = "));
115             write(output_line, B_f);
116             write(output_line, string'(" mode =  "));
117             write(output_line, Mode_f);
118             writeline(out_file, output_line);
119
120             write(output_line, string'("carry_out =  "));
121             write(output_line, Cout_f);
122             write(output_line, string'(" out =  "));
123             write(output_line, S_f);
124             writeline(out_file, output_line);
125             wait for 5 ns ;
126         end loop;
127
128         wait;
129     end process stimul;
130 end architecture TB;
```



out_file - Notepad

File Edit Format View Help

```
Time = 20 ns  in1 = 0101  in2 = 0101 mode =  0
carry_out =  0 out =  1010
Time = 45 ns  in1 = 0101  in2 = 0101 mode =  1
carry_out =  0 out =  0000
Time = 70 ns  in1 = 0100  in2 = 0010 mode =  0
carry_out =  0 out =  0110
```

in_file - Notepad

File Edit Format View Help

```
0101 0101 0
0101  0101  1
0100 0010 0
```