LAP1

Behavioral

```
Ln#
 1  LIBRARY IEEE;
 2  LIBRARY WORK;
 3  LIBRARY std;
 4  USE IEEE.STD_LOGIC_1164.ALL;
 5  USE ieee.numeric_bit.ALL;
 6  USE IEEE.std_logic_arith.all;
 7  USE IEEE.numeric_std.all;
 8  USE IEEE.std_logic_signed.all;
 9
10  ENTITY adder_sub_beh IS
11  PORT(a,b:IN std_logic_vector(3 downto 0);
12   mode:IN std_logic;
13   c_out:OUT std_logic;
14   sum :OUT std_logic_vector(3 downto 0));
15  END ENTITY adder_sub_beh;
16
17  ARCHITECTURE adder_sub_beh OF adder_sub_beh IS
18  signal result:std_logic_vector(4 downto 0);
19  BEGIN
20  M1:PROCESS(a,b,mode,result) IS
21
```

```
22    BEGIN
23    if(mode='0') THEN
24    result<=(("0"& a) + ("0"&b));
25    sum<=result(3 downto 0);
26    c_out<=result(4);
27    else
28    result<=(("0"& a) - ("0"&b));
29    sum<=result(3 downto 0);
30    c_out<=result(4);
31    END IF;
32    END PROCESS M1;
33    END ARCHITECTURE adder_sub_beh;
34
```

Data_flow

```
1  LIBRARY IEEE;
2  LIBRARY WORK;
3  LIBRARY std;
4  USE Ieee.std_logic_1164.ALL;
5  USE ieee.numeric_bit.ALL;
6  USE IEEE.std_logic_arith.all;
7  USE IEEE.numeric_std.all;
8  USE IEEE.std_logic_signed.all;
9
10 ENTITY adder_sub_beh IS
11 PORT(a,b:IN std_logic_vector(3 downto 0);
12  mode:IN std_logic;
13  c_out:OUT std_logic;
14  sum :OUT std_logic_vector(3 downto 0));
15 END ENTITY adder_sub_beh;
16 ARCHITECTURE adder_sub_beh OF adder_sub_beh IS
17 signal result:std_logic_vector(4 downto 0);
18 BEGIN
19 result<=(("0"& a) + ("0"&b)) WHEN (mode='0')
20 ELSE (("0"& a) - ("0"&b));
21 sum<=result(3 downto 0);
22 c_out<=result(4);
23 END ARCHITECTURE adder_sub_beh;
24
```

Structural

```
Ln#
 1    ENTITY full_adder IS
 2    PORT(a,b,mode:IN bit;
 3           c_out,sum:OUT bit);
 4    END ENTITY full_adder;
 5
 6    ARCHITECTURE full_adder OF full_adder IS
 7    BEGIN
 8    p1:process(a,b,mode) IS
 9    BEGIN
10    sum<= a XOR b XOR mode;
11    c_out<=(a AND b) OR (mode AND (a XOR b));
12    END PROCESS p1;
13
14    END ARCHITECTURE full_adder;
15
```

```
 1  LIBRARY IEEE;
 2  LIBRARY WORK;
 3  LIBRARY std;
 4  USE IEEE.STD_LOGIC_1164.ALL;
 5  USE ieee.numeric_bit.ALL;
 6  USE IEEE.std_logic_arith.all;
 7  USE IEEE.numeric_std.all;
 8  USE IEEE.std_logic_signed.all;
 9
10  ENTITY adder_sub_struct IS
11  PORT(a,b:IN bit_vector(3 downto 0);
12         mode:IN bit;
13         c_out:OUT bit;
14         sum :OUT bit_vector(3 downto 0));
15  END ENTITY adder_sub_struct;
16
17  ARCHITECTURE adder_sub_struct OF adder_sub_struct IS
18
```

```
17  ARCHITECTURE adder_sub_struct OF adder_sub_struct IS
18
19  COMPONENT add_sub_1bit IS
20  PORT(a,b,mode:IN bit;
21         c_out,sum:OUT bit);
22  END COMPONENT add_sub_1bit;
23  FOR ALL: add_sub_1bit USE ENTITY WORK.add_sub_1bit (add_sub_1bit);
24   SIGNAL carry :bit_vector(3 downto 0);
25   SIGNAL x :bit_vector(3 downto 0);
26   BEGIN
27   x(0)<=b(0) XOR mode;
28   x(1)<=b(1) XOR mode;
29   x(2)<=b(2) XOR mode;
30   x(3)<=b(3) XOR mode;
```

```
26   BEGIN
27   x(0)<=b(0) XOR mode;
28   x(1)<=b(1) XOR mode;
29   x(2)<=b(2) XOR mode;
30   x(3)<=b(3) XOR mode;
31   bit3:add_sub_1bit
32   PORT MAP(a(0),x(0) ,mode,carry(0),sum(0));
33   bit2:add_sub_1bit
34   PORT MAP(a(1),x(1) ,carry(0),carry(1),sum(1));
35   bit1:add_sub_1bit
36   PORT MAP(a(2),x(2) ,carry(1),carry(2),sum(2));
37   bit0:add_sub_1bit
38   PORT MAP(a(3),x(3) ,carry(2),carry(3),sum(3));
39   c_out<=carry(3);
40
41
42  END ARCHITECTURE adder_sub_struct;
43
```

```
in1= 1100 in2= 0111 time= 10 ns out= 0011 carry= 1 mode= 0
in1= 1011 in2= 0001 time= 20 ns out= 1010 carry= 0 mode= 1
in1= 0101 in2= 0010 time= 30 ns out= 0011 carry= 0 mode= 1
```

test_vectors.txt - Notepad

File Edit Format View Help

```
1100 0111 0
1011 0001 1
0101 0010 1
```

Adder_sub_tb

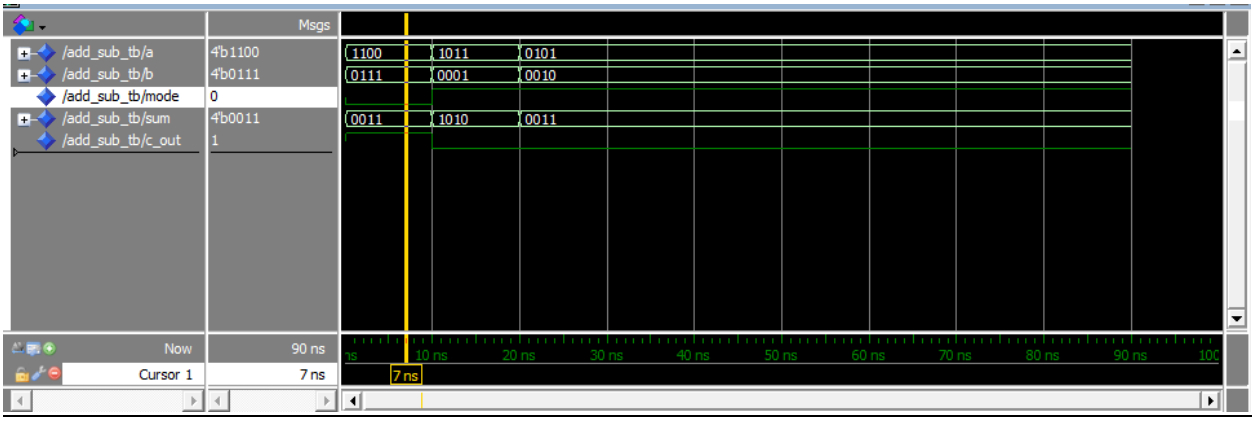
```
1  USE std.textio.ALL;
2  LIBRARY IEEE;
3  LIBRARY WORK;
4  LIBRARY std;
5  USE IEEE.STD_LOGIC_1164.ALL;
6  USE ieee.numeric_bit.ALL;
7  USE IEEE.std_logic_arith.all;
8  USE IEEE.numeric_std.all;
9  USE IEEE.std_logic_signed.all;
10
11 ENTITY add_sub_tb IS
12 END ENTITY add_sub_tb ;
13
14 ARCHITECTURE add_sub_tb OF add_sub_tb  IS
15
16 COMPONENT adder_sub_beh IS
17
18 PORT(a,b:IN std_logic_vector(3 downto 0);
19  mode:IN std_logic;
20  c_out:OUT std_logic;
```

```
18 PORT(a,b:IN std_logic_vector(3 downto 0);
19  mode:IN std_logic;
20  c_out:OUT std_logic;
21  sum :OUT std_logic_vector(3 downto 0));
22 END COMPONENT;
23
24 FOR dut: adder_sub_beh USE ENTITY WORK.adder_sub_beh (adder_sub_beh);
25
26 SIGNAL a,b:std_logic_vector(3 downto 0);
27 SIGNAL mode:std_logic;
28 SIGNAL sum:std_logic_vector(3 downto 0);
29 SIGNAL c_out:std_logic;
30 BEGIN
31 dut: adder_sub_beh PORT MAP(a,b,mode,c_out,sum);
32 pr: PROCESS IS
33 FILE vectors_f: text OPEN read_mode IS "E:\test_vectors.txt";
34 FILE results_f: text OPEN write_mode IS "E:\test_results.txt";
35 VARIABLE ln_file: line;
36 VARIABLE mode_t: std_logic;
```

```
VARIABLE ln_file: line;
VARIABLE mode_t: std_logic;
VARIABLE c_out_t: std_logic;
VARIABLE a_x,b_x: std_logic_vector(3 downto 0);
VARIABLE sum_t: std_logic_vector (3 downto 0);

VARIABLE out_file: line;

BEGIN
WHILE NOT endfile (vectors_f) LOOP
READLINE (vectors_f,ln_file);
READ (ln_file, a_x);
READ (ln_file, b_x);
READ (ln_file, mode_t);

-- writing varible in a,b signal

a<=a_x;
b<=b_x;
mode<=mode_t;
```

```
--now we get result
Wait for 10 ns;
c_out_t:=c_out;
sum_t:=sum;
write(out_file,string'(" in1= "));
write(out_file,a_x);
write(out_file,string'(" in2= "));
write(out_file,b_x);
write(out_file,string'(" time= "));
write(out_file,now);
write(out_file,string'(" out= "));
write(out_file,sum_t);
write(out_file,string'(" carry= "));
write(out_file,c_out_t);
write(out_file,string'(" mode= "));
write(out_file,mode_t);
writeline(results_f,out_file);
```

```
END LOOP;
WAIT;
END PROCESS ;
END ARCHITECTURE ;
```




```
in1= 1100 in2= 0111 time= 10 ns out= 0011 carry= 1 mode= 0
in1= 1011 in2= 0001 time= 20 ns out= 1010 carry= 0 mode= 1
in1= 0101 in2= 0010 time= 30 ns out= 0011 carry= 0 mode= 1
```

---

test_vectors.txt - Notepad

File Edit Format View Help

```
1100 0111 0
1011 0001 1
0101 0010 1
```
