VHDL LAB1

Digital verification

OMNIA MOHAMED SELIM

Table of Contents

behavioral architecture 2
design code: 2
The first test bench 3
RESULTS 4
The second test bench 4
RESULTS 5
Structural architecture 6
Design code 6
The first test bench 7
Results 7
The second test bench 8
Results 8
Dataflow architecture 9
Design code 9
The first test bench 9
Results 10
The second test bench 10
Results 11
The third test bench 11
Results 13

behavioral architecture

design code:

```
library IEEE;
LIBRARY std;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.numeric_bit.ALL;
use IEEE.std_logic_arith.all;
use IEEE.numeric_std.all;
use IEEE.std_logic_signed.all;

entity adder_subtractor  is
  port( A: IN STD_LOGIC_VECTOR(3 DOWNTO 0);
    B: IN STD_LOGIC_VECTOR(3 DOWNTO 0);
    M: IN STD_LOGIC;
    S: OUT STD_LOGIC_VECTOR(3 DOWNTO 0);
    Cout: OUT STD_LOGIC);
END adder_subtractor;

architecture behavioral of adder_subtractor is
begin
   process(A,B,M) IS
   VARIABLE temp:STD_LOGIC_VECTOR(4 DOWNTO 0);
   begin
      if(M='0') then
        temp:=('0' & A)+('0' & B);
        S<=temp(3 downto 0);
        Cout<= temp(4);
      else
     temp:=('0' & A)-('0' & B);
     S<=temp(3 downto 0);
        Cout<= temp(4);
       end if;
   end process;
end behavioral;
```

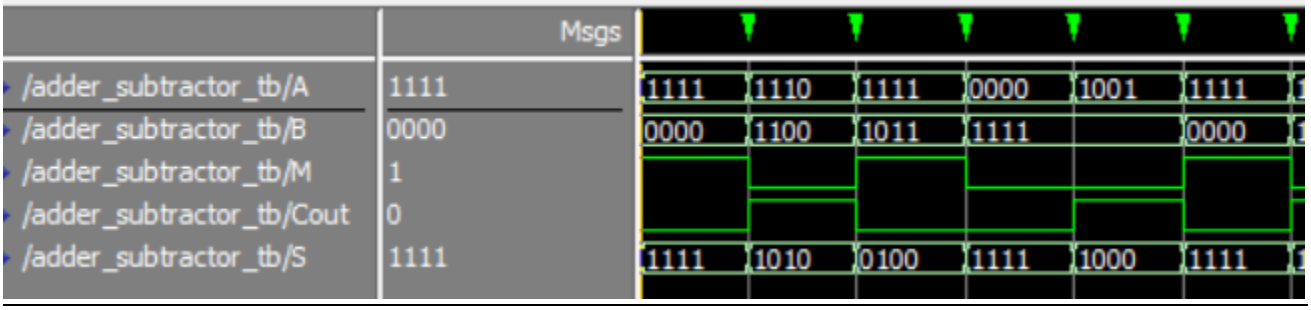
The first test bench

for testing the behavioral architecture

```
library IEEE;
LIBRARY std;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.numeric_bit.ALL;
use IEEE.std_logic_arith.all;
use IEEE.numeric_std.all;
use IEEE.std_logic_signed.all;

entity adder_subtractor_tb IS
end adder_subtractor_tb;

architecture firsttest of adder_subtractor_tb IS
 component adder_subtractor
  port(A: IN std_logic_vector(3 downto 0);
    B: IN std_logic_vector(3 downto 0);
    M: IN STD_LOGIC;
    S: OUT std_logic_vector(3 downto 0);
    Cout: OUT STD_LOGIC);
  end component;
for dut: adder_subtractor USE ENTITY work.adder_subtractor(behavioral);
--inputs
    signal A,B: std_logic_vector(3 downto 0):="0000";
    signal M: std_logic:='0';
--outputs
        signal S: std_logic_vector(3 downto 0);
    signal Cout: std_logic;

 begin
   dut:adder_subtractor port map(A,B,M,S,Cout);
   process
   begin

   A<="1111";
   B<="0000";
   M<='1';
   wait for 20 ns;
   assert S="1111" report" FAILED: A=1111,B=0000,M=1 (sum is incorrrect)" severity error;
   assert Cout='0' report" FAILED: A=1111,B=0000,M=1(carry out is incorrrect)" severity error;

A<="1110";
B<="1100";
M<='0';
wait for 20 ns;
assert S="1010" report" FAILED: A=1110,B=1100,M=0 (sum is incorrrect)" severity error;
assert Cout='1' report" FAILED: A=1110,B=1100,M=0(carry out is incorrrect)" severity error;
report" passed: A=1110,B=1100,M=0 ===> sum=1010 ,cout=1";

A<="1111";
B<="1011";
M<='1';
wait for 20 ns;
assert S="0100" report" FAILED: A=1111,B=1011,M=1 (sum is incorrrect)" severity error;
assert Cout='0' report" FAILED: A=1111,B=1011,M=1(carry out is incorrrect)" severity error;
report" passed: A=1111,B=1011,M=1 ===> sum=0100 ,cout=0";

A<="0000";
B<="1111";
M<='0';
wait for 20 ns;
assert S="1111" report" FAILED: A=0000,B=1111,M=0 (sum is incorrrect)" severity error;
assert Cout='0' report" FAILED: A=0000,B=1111,M=0(carry out is incorrrect)" severity error;
report" passed: A=0000,B=1111,M=0 ===> sum=1111 ,cout=0";

A<="1001";
B<="1111";
M<='0';
wait for 20 ns;
assert S="1000" report" FAILED: A=1001,B=1000,M=0 (sum is incorrrect)" severity error;
assert Cout='1' report" FAILED: A=1001,B=1000,M=0(carry out is incorrrect)" severity error;
report" passed: A=1001,B=1111,M=0 ===> sum=1000 ,cout=1";

end process;
end firsttest;
```

RESULTS

```
VSIM 18> run
# ** Note:  PASSED: A=1111,B=0000,M=1 ===> sum=1111,cout=0
#    Time: 20 ns  Iteration: 0  Instance: /adder_subtractor_tb
# ** Note:  passed: A=1110,B=1100,M=0 ===> sum=1010 ,cout=1
#    Time: 40 ns  Iteration: 0  Instance: /adder_subtractor_tb
# ** Note:  passed: A=1111,B=1011,M=1 ===> sum=0100 ,cout=0
#    Time: 60 ns  Iteration: 0  Instance: /adder_subtractor_tb
# ** Note:  passed: A=0000,B=1111,M=0 ===> sum=1111 ,cout=0
#    Time: 80 ns  Iteration: 0  Instance: /adder_subtractor_tb
# ** Note:  passed: A=1001,B=1111,M=0 ===> sum=1000 ,cout=1
#    Time: 100 ns  Iteration: 0  Instance: /adder_subtractor_tb
```



The second test bench

for testing the behavioral architecture

```
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_arith.all;
USE std.textio.ALL;

entity second_tb_adder_subtractor is
end entity second_tb_adder_subtractor;

architecture secondtest of second_tb_adder_subtractor is

component adder_subtractor IS
  port( A: IN std_logic_vector(3 downto 0);
     B: IN std_logic_vector(3 downto 0);
     M: IN STD_LOGIC;
     S: OUT std_logic_vector(3 downto 0);
     Cout: OUT STD_LOGIC);
end component adder_subtractor;
for dut: adder_subtractor USE ENTITY work.adder_subtractor(behavioral);
signal m_tb,cout_tb: STD_LOGIC;
signal a_tb, b_tb, s_tb : std_logic_vector(3 downto 0);
begin

  DUT: adder_subtractor port map(a_tb, b_tb,m_tb,s_tb, cout_tb);
 p1:process
     FILE test_f: text OPEN READ_MODE IS "add_sub_test_input.txt";
     FILE results_f: text OPEN WRITE_MODE IS "add_sub_test_results.txt";
     VARIABLE L: line;
     VARIABLE res_l: line;
     VARIABLE m_var,cout_expected: STD_LOGIC;
     VARIABLE a_var, b_var, s_expected:std_logic_vector(3 downto 0);
     VARIABLE c: character;
     VARIABLE pause: time;
        VARIABLE message: string (1 TO 14);
 begin
     WHILE NOT endfile(test_f) LOOP
     READLINE (test_f, L);
     READ (L, a_var);
     READ (L, b_var);
 READ (L, m_var);
     READ (L, pause);
     READ (L, cout_expected);
 READ (L, s_expected);
     READ (L, message);
```

```
        wait for pause;

        WRITE (res_l, string'(". a = "));
          WRITE (res_l, a_tb);

          WRITE (res_l, string'(", b = "));
          WRITE (res_l, b_tb);

          WRITE (res_l, string'(", m = "));
          WRITE (res_l, m_tb);

          WRITE (res_l, string'(", Expected cout = "));
          WRITE (res_l, cout_expected);

          WRITE (res_l, string'(", Actual cout = "));
          WRITE (res_l, cout_tb);

             WRITE (res_l, string'(", Expected sum = "));
          WRITE (res_l, s_expected);

          WRITE (res_l, string'(", Actual sum = "));
          WRITE (res_l, s_tb);
          IF cout_tb /= cout_expected or s_tb /= s_expected THEN
              WRITE (res_l, string'(". Test failed! Error message:"));
              WRITE (res_l, message);
          ELSE
              WRITE (res_l, string'(". Test passed."));
          END IF;

          WRITELINE (results_f, res_l);

    end loop;
    wait;
  end process p1;
end architecture secondtest;
```

add_sub_test_input - Notepad

File Edit Format View Help

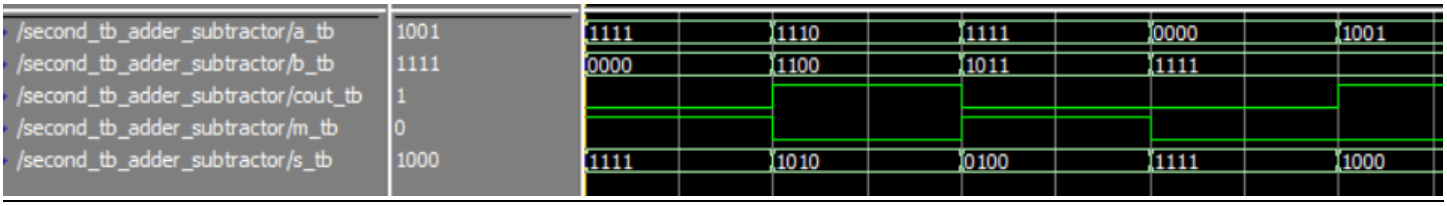
```
1111 0000 1 10 ns 0 1111 FAILED IN SUB
1110 1100 0 10 ns 1 1010 FAILED IN ADD
1111 1011 1 10 ns 0 0100 FAILED IN SUB
0000 1111 0 10 ns 0 1111 FAILED IN ADD
1001 1111 0 10 ns 1 1000 FAILED IN ADD
```

RESULTS

add_sub_test_results - Notepad

File Edit Format View Help

```
. a = 1111, b = 0000, m = 1, Expected cout = 0, Actual cout = 0, Expected sum = 1111, Actual sum = 1111. Test passed.
. a = 1110, b = 1100, m = 0, Expected cout = 1, Actual cout = 1, Expected sum = 1010, Actual sum = 1010. Test passed.
. a = 1111, b = 1011, m = 1, Expected cout = 0, Actual cout = 0, Expected sum = 0100, Actual sum = 0100. Test passed.
. a = 0000, b = 1111, m = 0, Expected cout = 0, Actual cout = 0, Expected sum = 1111, Actual sum = 1111. Test passed.
. a = 1001, b = 1111, m = 0, Expected cout = 1, Actual cout = 1, Expected sum = 1000, Actual sum = 1000. Test passed.
```



Structural architecture

Design code

```
library IEEE;
LIBRARY std;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.numeric_bit.ALL;
use IEEE.std_logic_arith.all;
use IEEE.numeric_std.all;
use IEEE.std_logic_signed.all;


entity FULLADDER  is
  port( x,y,cin: IN std_logic;
     s,cout: OUT std_logic);
end FULLADDER;

architecture behavioral of FULLADDER is
begin
        S<=x XOR y XOR cin;
     cout<= (x AND (y OR cin)) OR (cin AND y);
end behavioral;
```

```
library IEEE;
LIBRARY std;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.numeric_bit.ALL;
use IEEE.std_logic_arith.all;
use IEEE.numeric_std.all;
use IEEE.std_logic_signed.all;

entity adder_subtractor IS
   port( A: IN std_logic_vector(3 downto 0);
      B: IN std_logic_vector(3 downto 0);
      M: IN STD_LOGIC;
      S: OUT std_logic_vector(3 downto 0);
      Cout: OUT STD_LOGIC);
end adder_subtractor;


architecture structural of adder_subtractor is
   component FULLADDER
     port( x,y,cin: in std_logic;
           s,cout:out std_logic);
   end component;
   signal C0,C1,C2,C3:std_logic;
   signal b0,b1,b2,b3:std_logic;
   begin
    b0<= M XOR B(0);
    b1<= M XOR B(1);
    b2<= M XOR B(2);
    b3<= M XOR B(3);
     FA0 :FULLADDER port map(A(0),b0,M,S(0),C0);
     FA1 :FULLADDER port map(A(1),b1,C0,S(1),C1);
     FA2 :FULLADDER port map(A(2),b2,C1,S(2),C2);
     FA3 :FULLADDER port map(A(3),b3,C2,S(3),C3);
     Cout <= C3 xor M;

   end structural;
```

The first test bench

for testing the structural architecture

```
library IEEE;
LIBRARY std;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.numeric_bit.ALL;
use IEEE.std_logic_arith.all;
use IEEE.numeric_std.all;
use IEEE.std_logic_signed.all;

entity adder_subtractor_tb IS
end adder_subtractor_tb;

architecture firsttest of adder_subtractor_tb IS
 component adder_subtractor
  port(A: IN std_logic_vector(3 downto 0);
    B: IN std_logic_vector(3 downto 0);
    M: IN STD_LOGIC;
    S: OUT std_logic_vector(3 downto 0);
    Cout: OUT STD_LOGIC);
  end component;
for dut: adder_subtractor USE ENTITY work.adder_subtractor(structural);
```

Results


```
SIM 31> run
** Note:  PASSED: A=1111,B=0000,M=1 ===> sum=1111,cout=0
   Time: 120 ns  Iteration: 0  Instance: /adder_subtractor_tb
** Note:  passed: A=1110,B=1100,M=0 ===> sum=1010 ,cout=1
   Time: 140 ns  Iteration: 0  Instance: /adder_subtractor_tb
** Note:  passed: A=1111,B=1011,M=1 ===> sum=0100 ,cout=0
   Time: 160 ns  Iteration: 0  Instance: /adder_subtractor_tb
** Note:  passed: A=0000,B=1111,M=0 ===> sum=1111 ,cout=0
   Time: 180 ns  Iteration: 0  Instance: /adder_subtractor_tb
** Note:  passed: A=1001,B=1111,M=0 ===> sum=1000 ,cout=1
   Time: 200 ns  Iteration: 0  Instance: /adder_subtractor_tb
** Note:  PASSED: A=1111,B=0000,M=1 ===> sum=1111,cout=0
   Time: 220 ns  Iteration: 0  Instance: /adder_subtractor_tb
```

The second test bench

for testing the structural architecture

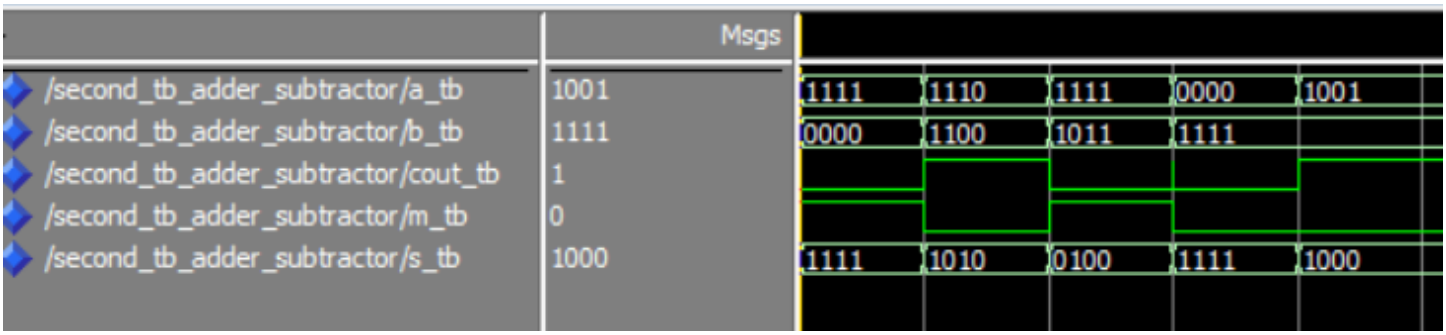
```
library IEEE;
LIBRARY std;
USE ieee.numeric_std.ALL;
--USE ieee.numeric_bit.ALL;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_arith.all;
USE std.textio.ALL;


entity second_tb_adder_subtractor is
end entity second_tb_adder_subtractor;

architecture secondtest of second_tb_adder_subtractor is

component adder_subtractor IS
  port( A: IN std_logic_vector(3 downto 0);
    B: IN std_logic_vector(3 downto 0);
    M: IN STD_LOGIC;
    S: OUT std_logic_vector(3 downto 0);
    Cout: OUT STD_LOGIC);
end component adder_subtractor;
for dut: adder_subtractor USE ENTITY work.adder_subtractor(structural);
signal m_tb,cout_tb: STD_LOGIC;
signal a_tb, b_tb, s_tb : std_logic_vector(3 downto 0);
begin

  DUT: adder_subtractor port map(a_tb, b_tb,m_tb,s_tb, cout_tb);
```

Results

add_sub_test_results - Notepad

File Edit Format View Help

```
a = 1111, b = 0000, m = 1, Expected cout = 0, Actual cout = 0, Expected sum = 1111, Actual sum = 1111. Test passed.
a = 1110, b = 1100, m = 0, Expected cout = 1, Actual cout = 1, Expected sum = 1010, Actual sum = 1010. Test passed.
a = 1111, b = 1011, m = 1, Expected cout = 0, Actual cout = 0, Expected sum = 0100, Actual sum = 0100. Test passed.
a = 0000, b = 1111, m = 0, Expected cout = 0, Actual cout = 0, Expected sum = 1111, Actual sum = 1111. Test passed.
a = 1001, b = 1111, m = 0, Expected cout = 1, Actual cout = 1, Expected sum = 1000, Actual sum = 1000. Test passed.
```



Dataflow architecture

Design code

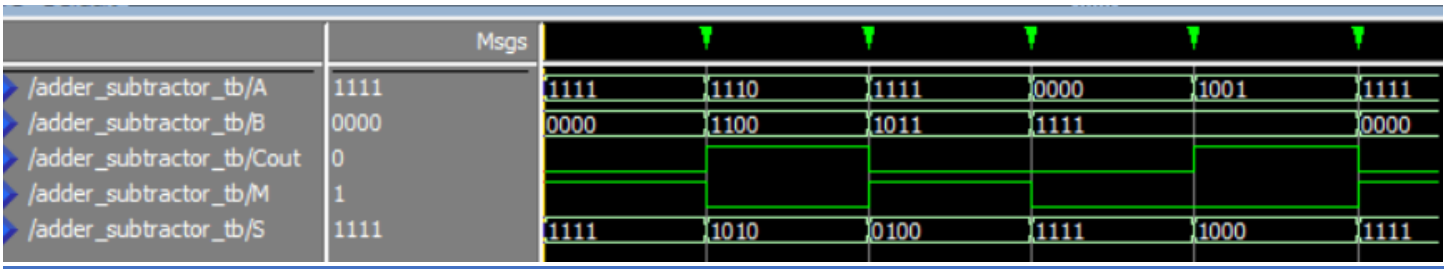
```
library IEEE;
LIBRARY std;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.numeric_bit.ALL;
use IEEE.std_logic_arith.all;
use IEEE.numeric_std.all;
use IEEE.std_logic_signed.all;

entity adder_subtractor  is
  port( A: IN std_logic_vector(3 downto 0);
    B: IN std_logic_vector(3 downto 0);
    M: IN STD_LOGIC;
    S: OUT std_logic_vector(3 downto 0);
    Cout: OUT STD_LOGIC);
END adder_subtractor;

architecture dataflow of adder_subtractor is
  signal temp:STD_LOGIC_VECTOR(4 DOWNTO 0);
begin
   temp<=('0' & A)+('0' & B) when M='0' else ('0' & A)-('0' & B);
   S<=temp(3 downto 0);
   Cout<= temp(4);

end dataflow;
```

The first test bench

for testing the dataflow architecture

```
library IEEE;
LIBRARY std;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.numeric_bit.ALL;
use IEEE.std_logic_arith.all;
use IEEE.numeric_std.all;
use IEEE.std_logic_signed.all;

entity adder_subtractor_tb IS
end adder_subtractor_tb;

architecture firsttest of adder_subtractor_tb IS
 component adder_subtractor
  port(A: IN std_logic_vector(3 downto 0);
    B: IN std_logic_vector(3 downto 0);
    M: IN STD_LOGIC;
    S: OUT std_logic_vector(3 downto 0);
    Cout: OUT STD_LOGIC);
  end component;
for dut: adder_subtractor USE ENTITY work.adder_subtractor(dataflow);
--inputs
    signal A,B: std_logic_vector(3 downto 0):="0000";
    signal M: std_logic:='0';
--outputs
        signal S: std_logic_vector(3 downto 0);
    signal Cout: std_logic;

 begin
   dut:adder_subtractor port map(A,B,M,S,Cout);
   process
```

Results




```
** Note:  PASSED: A=1111,B=0000,M=1 ===> sum=1111,cout=0
   Time: 20 ns  Iteration: 0  Instance: /adder_subtractor_tb
** Note:  passed: A=1110,B=1100,M=0 ===> sum=1010 ,cout=1
   Time: 40 ns  Iteration: 0  Instance: /adder_subtractor_tb
** Note:  passed: A=1111,B=1011,M=1 ===> sum=0100 ,cout=0
   Time: 60 ns  Iteration: 0  Instance: /adder_subtractor_tb
** Note:  passed: A=0000,B=1111,M=0 ===> sum=1111 ,cout=0
   Time: 80 ns  Iteration: 0  Instance: /adder_subtractor_tb
** Note:  passed: A=1001,B=1111,M=0 ===> sum=1000 ,cout=1
   Time: 100 ns  Iteration: 0  Instance: /adder_subtractor_tb
```

The second test bench

for testing the dataflow architecture

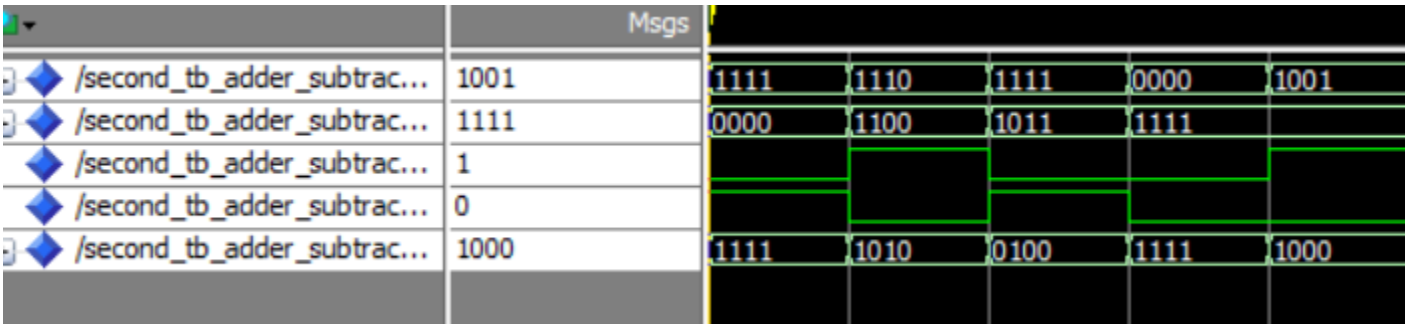
```
library IEEE;
LIBRARY std;
USE ieee.numeric_std.ALL;
--USE ieee.numeric_bit.ALL;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_arith.all;
USE std.textio.ALL;


entity second_tb_adder_subtractor is
end entity second_tb_adder_subtractor;

architecture secondtest of second_tb_adder_subtractor is

component adder_subtractor IS
  port( A: IN std_logic_vector(3 downto 0);
     B: IN std_logic_vector(3 downto 0);
     M: IN STD_LOGIC;
     S: OUT std_logic_vector(3 downto 0);
     Cout: OUT STD_LOGIC);
end component adder_subtractor;
for dut: adder_subtractor USE ENTITY work.adder_subtractor(dataflow) ;
signal m_tb,cout_tb: STD_LOGIC;
signal a_tb, b_tb, s_tb : std_logic_vector(3 downto 0);
begin

  DUT: adder_subtractor port map(a_tb, b_tb,m_tb,s_tb, cout_tb);
```

Results

add_sub_test_results - Notepad

File Edit Format View Help

```
. a = 1111, b = 0000, m = 1, Expected cout = 0, Actual cout = 0, Expected sum = 1111, Actual sum = 1111. Test passed.
. a = 1110, b = 1100, m = 0, Expected cout = 1, Actual cout = 1, Expected sum = 1010, Actual sum = 1010. Test passed.
. a = 1111, b = 1011, m = 1, Expected cout = 0, Actual cout = 0, Expected sum = 0100, Actual sum = 0100. Test passed.
. a = 0000, b = 1111, m = 0, Expected cout = 0, Actual cout = 0, Expected sum = 1111, Actual sum = 1111. Test passed.
. a = 1001, b = 1111, m = 0, Expected cout = 1, Actual cout = 1, Expected sum = 1000, Actual sum = 1000. Test passed.
```



The third test bench

It compares the outputs of the different architecture implementations to each other.

```
library IEEE;
LIBRARY std;
use IEEE.STD_LOGIC_1164.ALL;
USE ieee.numeric_bit.ALL;
use IEEE.std_logic_arith.all;
use IEEE.numeric_std.all;
use IEEE.std_logic_signed.all;

entity third_tb_adder_subtractor IS
end third_tb_adder_subtractor;

architecture thirdtest of third_tb_adder_subtractor IS
 component adder_subtractor
  port(A: IN std_logic_vector(3 downto 0);
    B: IN std_logic_vector(3 downto 0);
    M: IN STD_LOGIC;
    S: OUT std_logic_vector(3 downto 0);
    Cout: OUT STD_LOGIC);
  end component;

for add_sub_behav: adder_subtractor USE ENTITY work.adder_subtractor(behavioral);
for add_sub_struct: adder_subtractor USE ENTITY work.adder_subtractor(structural);
for add_sub_dataflow: adder_subtractor USE ENTITY work.adder_subtractor(dataflow);

--inputs
    signal A,B: std_logic_vector(3 downto 0):="0000";
    signal M: std_logic:='0';
--outputs
        signal S_behav,S_struct,S_df: std_logic_vector(3 downto 0);
    signal Cout_behav,Cout_struct,Cout_df: std_logic;

 begin
  add_sub_behav:adder_subtractor port map(A,B,M,S_behav,Cout_behav);
  add_sub_struct:adder_subtractor port map(A,B,M,S_struct,Cout_struct);
  add_sub_dataflow:adder_subtractor port map(A,B,M,S_df,Cout_df);
   process
   begin
```

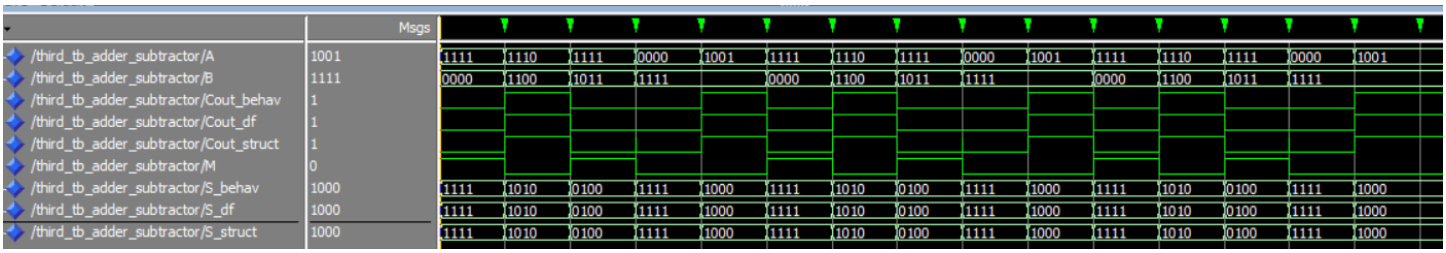
```
-- compare the outputs of  behavioral architecture and structural architecture
   A<="1111";
   B<="0000";
   M<='1';
   wait for 20 ns;
   assert S_behav=S_struct report" FAILED: A=1111,B=0000,M=1 (the output (sum) of the behavioral architecture and structural architecture is not the same ) " severity error;
   assert Cout_behav=Cout_struct report" FAILED: A=1111,B=0000,M=1(the output (carry out) of the behavioral architecture and structural architecture is not the same) " severity error;
   report" PASSED: A=1111,B=0000,M=1 ===>(the outputs (sum, carry out) of the behavioral architecture and structural architecture are the same ";

   A<="1110";
   B<="1100";
   M<='0';
   wait for 20 ns;
   assert S_behav=S_struct report" FAILED: A=1110,B=1100,M=0 (the output (sum) of the behavioral architecture and structural architecture is not the same) " severity error;
   assert Cout_behav=Cout_struct report" FAILED: A=1110,B=1100,M=0(the output (carry out) of the behavioral architecture and structural architecture is not the same ) " severity error;
   report" passed: A=1110,B=1100,M=0 ===> (the outputs (sum, carry out) of the behavioral architecture and structural architecture are the same ) ";

   A<="1111";
   B<="1011";
   M<='1';
   wait for 20 ns;
   assert S_behav=S_struct report" FAILED: A=1111,B=1011,M=1 (the output (sum) of the behavioral architecture and structural architecture is not the same)" severity error;
   assert Cout_behav=Cout_struct report" FAILED: A=1111,B=1011,M=1(the output (carry out) of the behavioral architecture and structural architecture is not the same)" severity error;
   report" passed: A=1111,B=1011,M=1 ===> sum=0100 ,cout=0 the outputs (sum, carry out) of the behavioral architecture and structural architecture are the same";

   A<="0000";
   B<="1111";
   M<='0';
   wait for 20 ns;
   assert S_behav=S_struct report" FAILED: A=0000,B=1111,M=0 (the output (sum) of the behavioral architecture and structural architecture is not the same)" severity error;
   assert Cout_behav=Cout_struct report" FAILED: A=0000,B=1111,M=0(the output (carry out) of the behavioral architecture and structural architecture is not the same)" severity error;
   report" passed: A=0000,B=1111,M=0 ===> sum=1111 ,cout=0 the outputs (sum, carry out) of the behavioral architecture and structural architecture are the same";

   A<="1001";
   B<="1111";
   M<='0';
   wait for 20 ns;
   assert S_behav=S_struct report" FAILED: A=1001,B=1000,M=0 (the output (sum) of the behavioral architecture and structural architecture is not the same)" severity error;
   assert Cout_behav=Cout_struct report" FAILED: A=1001,B=1000,M=0(the output (carry out) of the behavioral architecture and structural architecture is not the same)" severity error;
   report" passed: A=1001,B=1111,M=0 ===> sum=1000 ,cout=1 the outputs (sum, carry out) of the behavioral architecture and structural architecture are the same";
```

```
-- compare the outputs of  behavioral architecture and dataflow architecture
   A<="1111";
   B<="0000";
   M<='1';
   wait for 20 ns;
   assert S_behav=S_df report" FAILED: A=1111,B=0000,M=1 (the output (sum) of the behavioral architecture and dataflow architecture is not the same ) " severity error;
   assert Cout_behav=Cout_df report" FAILED: A=1111,B=0000,M=1(the output (carry out) of the behavioral architecture and dataflow architecture is not the same) " severity error;
   report" PASSED: A=1111,B=0000,M=1 ===>(the outputs (sum, carry out) of the behavioral architecture and dataflow architecture are the same ";

   A<="1110";
   B<="1100";
   M<='0';
   wait for 20 ns;
   assert S_behav=S_df report" FAILED: A=1110,B=1100,M=0 (the output (sum) of the behavioral architecture and dataflow architecture is not the same) " severity error;
   assert Cout_behav=Cout_df report" FAILED: A=1110,B=1100,M=0(the output (carry out) of the behavioral architecture and dataflow architecture is not the same ) " severity error;
   report" passed: A=1110,B=1100,M=0 ===> (the outputs (sum, carry out) of the behavioral architecture and dataflow architecture are the same ) ";

   A<="1111";
   B<="1011";
   M<='1';
   wait for 20 ns;
   assert S_behav=S_df report" FAILED: A=1111,B=1011,M=1 (the output (sum) of the behavioral architecture and dataflow architecture is not the same)" severity error;
   assert Cout_behav=Cout_df report" FAILED: A=1111,B=1011,M=1(the output (carry out) of the behavioral architecture and dataflow architecture is not the same)" severity error;
   report" passed: A=1111,B=1011,M=1 ===> sum=0100 ,cout=0 the outputs (sum, carry out) of the behavioral architecture and dataflow architecture are the same";

   A<="0000";
   B<="1111";
   M<='0';
   wait for 20 ns;
   assert S_behav=S_df report" FAILED: A=0000,B=1111,M=0 (the output (sum) of the behavioral architecture and dataflow architecture is not the same)" severity error;
   assert Cout_behav=Cout_df report" FAILED: A=0000,B=1111,M=0(the output (carry out) of the behavioral architecture and dataflow architecture is not the same)" severity error;
   report" passed: A=0000,B=1111,M=0 ===> sum=1111 ,cout=0 the outputs (sum, carry out) of the behavioral architecture and dataflow architecture are the same";

   A<="1001";
   B<="1111";
   M<='0';
   wait for 20 ns;
   assert S_behav=S_df report" FAILED: A=1001,B=1000,M=0 (the output (sum) of the behavioral architecture and dataflow architecture is not the same)" severity error;
   assert Cout_behav=Cout_df report" FAILED: A=1001,B=1000,M=0(the output (carry out) of the behavioral architecture and dataflow architecture is not the same)" severity error;
   report" passed: A=1001,B=1111,M=0 ===> sum=1000 ,cout=1 the outputs (sum, carry out) of the behavioral architecture and dataflow architecture are the same";
```

```
-- compare the outputs of  dataflow architecture and structural architecture
   A<="1111";
   B<="0000";
   M<='1';
   wait for 20 ns;
   assert S_df=S_struct report" FAILED: A=1111,B=0000,M=1 (the output (sum) of the dataflow architecture and structural architecture is not the same ) " severity error;
   assert Cout_df=Cout_struct report" FAILED: A=1111,B=0000,M=1(the output (carry out) of the dataflow architecture and structural architecture is not the same) " severity error;
   report" PASSED: A=1111,B=0000,M=1 ===>(the outputs (sum, carry out) of the dataflow architecture and structural architecture are the same ";

   A<="1110";
   B<="1100";
   M<='0';
   wait for 20 ns;
   assert S_df=S_struct report" FAILED: A=1110,B=1100,M=0 (the output (sum) of the dataflow architecture and structural architecture is not the same) " severity error;
   assert Cout_df=Cout_struct report" FAILED: A=1110,B=1100,M=0(the output (carry out) of the dataflow architecture and structural architecture is not the same ) " severity error;
   report" passed: A=1110,B=1100,M=0 ===> (the outputs (sum, carry out) of the dataflow architecture and structural architecture are the same ) ";

   A<="1111";
   B<="1011";
   M<='1';
   wait for 20 ns;
   assert S_df=S_struct report" FAILED: A=1111,B=1011,M=1 (the output (sum) of the dataflow architecture and structural architecture is not the same)" severity error;
   assert Cout_df=Cout_struct report" FAILED: A=1111,B=1011,M=1(the output (carry out) of the dataflow architecture and structural architecture is not the same)" severity error;
   report" passed: A=1111,B=1011,M=1 ===> sum=0100 ,cout=0 the outputs (sum, carry out) of the dataflow architecture and structural architecture are the same";

   A<="0000";
   B<="1111";
   M<='0';
   wait for 20 ns;
   assert S_df=S_struct report" FAILED: A=0000,B=1111,M=0 (the output (sum) of the dataflow architecture and structural architecture is not the same)" severity error;
   assert Cout_df=Cout_struct report" FAILED: A=0000,B=1111,M=0(the output (carry out) of the dataflow architecture and structural architecture is not the same)" severity error;
   report" passed: A=0000,B=1111,M=0 ===> sum=1111 ,cout=0 the outputs (sum, carry out) of the dataflow architecture and structural architecture are the same";

   A<="1001";
   B<="1111";
   M<='0';
   wait for 20 ns;
   assert S_df=S_struct report" FAILED: A=1001,B=1000,M=0 (the output (sum) of the dataflow architecture and structural architecture is not the same)" severity error;
   assert Cout_df=Cout_struct report" FAILED: A=1001,B=1000,M=0(the output (carry out) of the dataflow architecture and structural architecture is not the same)" severity error;
   report" passed: A=1001,B=1111,M=0 ===> sum=1000 ,cout=1 the outputs (sum, carry out) of the dataflow architecture and structural architecture are the same";

   wait;
   end process;
   end thirdtest;
```

Results

```
sim:/third_tb_adder_subtractor/S_struct
VSIM 50> run
# ** Note:  PASSED: A=1111,B=0000,M=1 ===>(the outputs (sum, carry out) of the behavioral architecture and structural architecture are the same
#    Time: 20 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
# ** Note:  passed: A=1110,B=1100,M=0 ===> (the outputs (sum, carry out) of the behavioral architecture and structural architecture are the same )
#    Time: 40 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
# ** Note:  passed: A=1111,B=1011,M=1 ===> sum=0100 ,cout=0 the outputs (sum, carry out) of the behavioral architecture and structural architecture are the same
#    Time: 60 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
# ** Note:  passed: A=0000,B=1111,M=0 ===> sum=1111 ,cout=0 the outputs (sum, carry out) of the behavioral architecture and structural architecture are the same
#    Time: 80 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
# ** Note:  passed: A=1001,B=1111,M=0 ===> sum=1000 ,cout=1 the outputs (sum, carry out) of the behavioral architecture and structural architecture are the same
#    Time: 100 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
VSIM 51> run
# ** Note:  PASSED: A=1111,B=0000,M=1 ===>(the outputs (sum, carry out) of the behavioral architecture and dataflow architecture are the same
#    Time: 120 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
# ** Note:  passed: A=1110,B=1100,M=0 ===> (the outputs (sum, carry out) of the behavioral architecture and dataflow architecture are the same )
#    Time: 140 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
# ** Note:  passed: A=1111,B=1011,M=1 ===> sum=0100 ,cout=0 the outputs (sum, carry out) of the behavioral architecture and dataflow architecture are the same
#    Time: 160 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
# ** Note:  passed: A=0000,B=1111,M=0 ===> sum=1111 ,cout=0 the outputs (sum, carry out) of the behavioral architecture and dataflow architecture are the same
#    Time: 180 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
# ** Note:  passed: A=1001,B=1111,M=0 ===> sum=1000 ,cout=1 the outputs (sum, carry out) of the behavioral architecture and dataflow architecture are the same
#    Time: 200 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
# ** Note:  PASSED: A=1111,B=0000,M=1 ===>(the outputs (sum, carry out) of the dataflow architecture and structural architecture are the same
#    Time: 220 ns  Iteration: 0  Instance: /third_tb_adder_subtractor

** Note:  passed: A=1110,B=1100,M=0 ===> (the outputs (sum, carry out) of the dataflow architecture and structural architecture are the same )
   Time: 240 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
** Note:  passed: A=1111,B=1011,M=1 ===> sum=0100 ,cout=0 the outputs (sum, carry out) of the dataflow architecture and structural architecture are the same
   Time: 260 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
** Note:  passed: A=0000,B=1111,M=0 ===> sum=1111 ,cout=0 the outputs (sum, carry out) of the dataflow architecture and structural architecture are the same
   Time: 280 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
** Note:  passed: A=1001,B=1111,M=0 ===> sum=1000 ,cout=1 the outputs (sum, carry out) of the dataflow architecture and structural architecture are the same
   Time: 300 ns  Iteration: 0  Instance: /third_tb_adder_subtractor
```


Msgs
/third_tb_adder_subtractor/A
1001
1111 1110 1111 0000 1001 1111 1110 1111 0000 1001 1111 1110 1111 0000 1001
/third_tb_adder_subtractor/B
1111
0000 1100 1011 1111 0000 1100 1011 1111 0000 1100 1011 1111
/third_tb_adder_subtractor/Cout_behav
1
/third_tb_adder_subtractor/Cout_df
1
/third_tb_adder_subtractor/Cout_struct
1
/third_tb_adder_subtractor/M
0
/third_tb_adder_subtractor/S_behav
1000
1111 1010 0100 1111 1000 1111 1010 0100 1111 1000 1111 1010 0100 1111 1000
/third_tb_adder_subtractor/S_df
1000
1111 1010 0100 1111 1000 1111 1010 0100 1111 1000 1111 1010 0100 1111 1000
/third_tb_adder_subtractor/S_struct
1000
1111 1010 0100 1111 1000 1111 1010 0100 1111 1000 1111 1010 0100 1111 1000