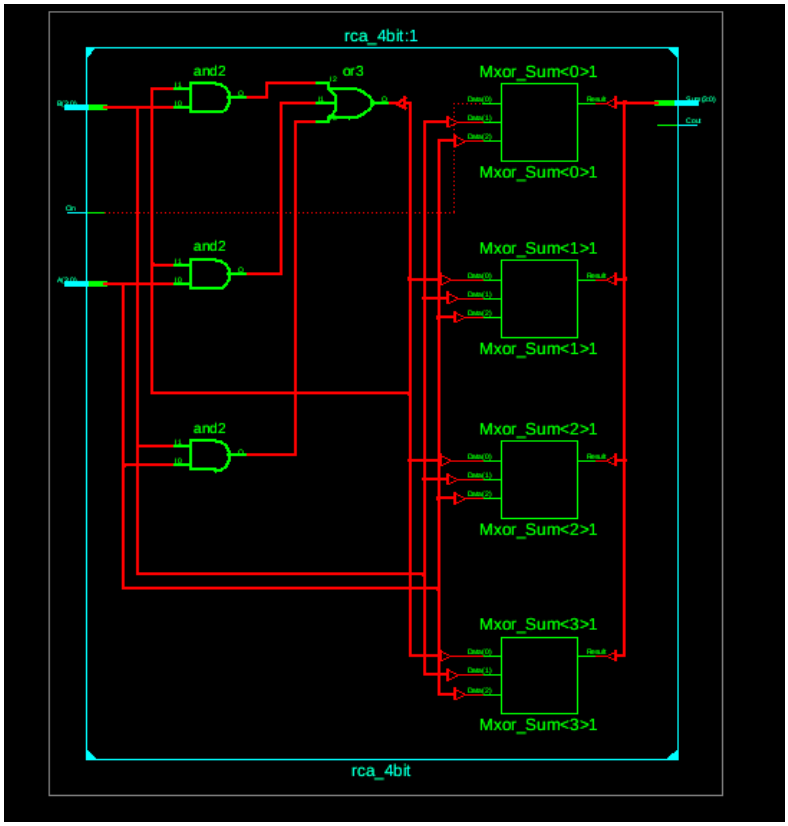
Mohamed Manessouri
SSE

Tp4: Test Benches

Exercice 1:

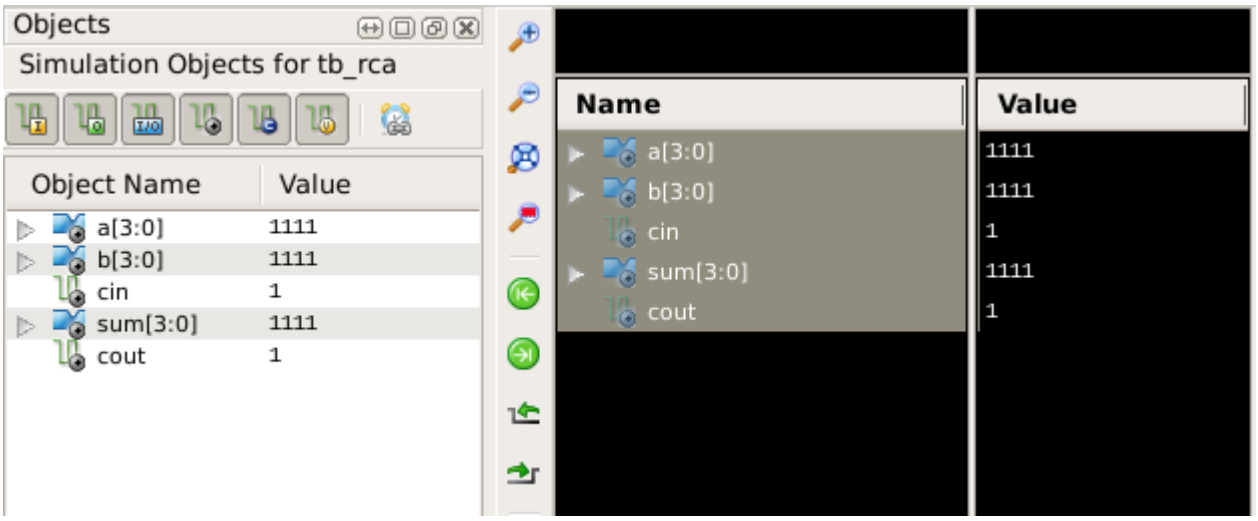
Rca_4bit:

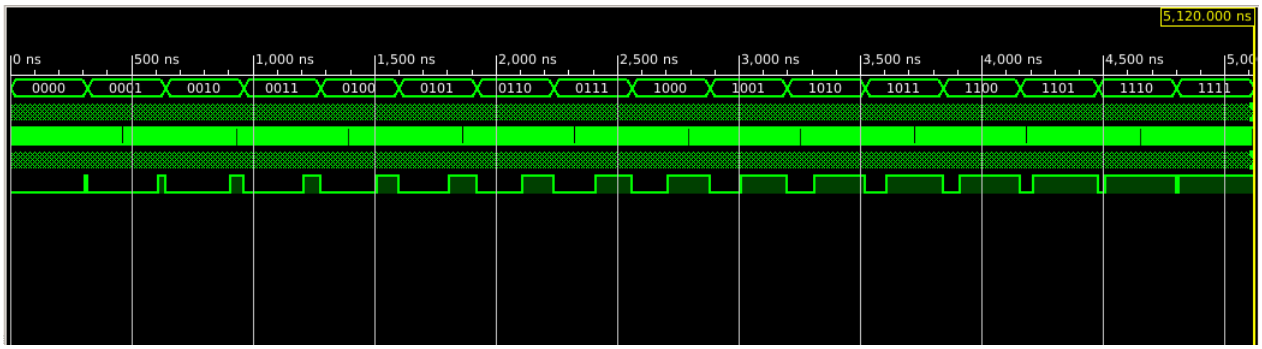
```
entity rca_4bit is
    Port ( A : in  STD_LOGIC_VECTOR(3 downto 0);
           B : in  STD_LOGIC_VECTOR(3 downto 0);
           Cin : in  STD_LOGIC;
           Sum : out  STD_LOGIC_VECTOR(3 downto 0);
           Cout : out  STD_LOGIC);
end rca_4bit;

architecture Behavioral of rca_4bit is
    signal carry : std_logic_vector(4 downto 0);
begin
    carry(0) <= Cin;

    gen_addition: for i in 0 to 3 generate
        Sum(i) <= A(i) xor B(i) xor carry(i);
        carry(i + 1) <= (A(i) and B(i)) or (A(i) and carry(i)) or (B(i) and carry(i));
    end generate;

    Cout <= carry(4);

end Behavioral;


```



Test Benche:

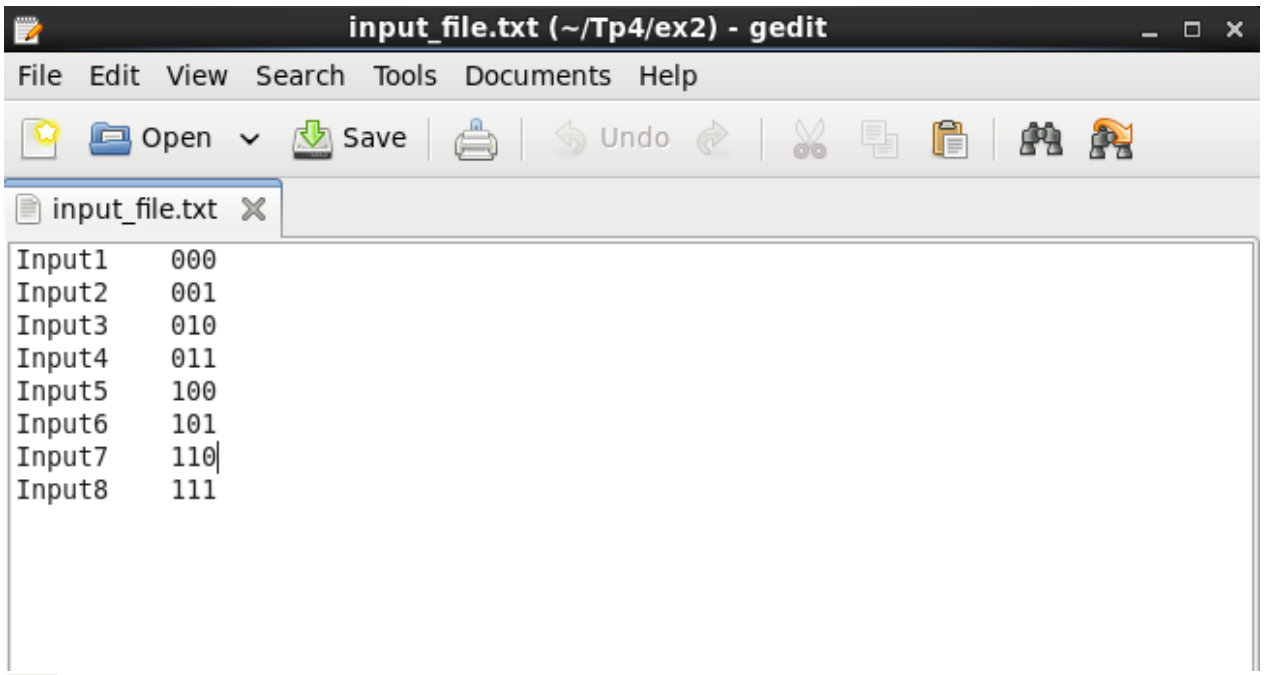
```
39  stimulus: process
40      variable i, j : integer;
41    begin
42      for i in 0 to 15 loop
43        for j in 0 to 15 loop
44          A <= std_logic_vector(to_unsigned(i, 4));
45          B <= std_logic_vector(to_unsigned(j, 4));
46          Cin <= '0'; wait for 10 ns;
47          Cin <= '1'; wait for 10 ns;
48        end loop;
49      end loop;
50
51      -- Terminer la simulation
52      wait;
53    end process;
54
55
56    END;
57
```





Exercice 2:

- **Préparer le fichier d'entrée:**




```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE ieee.std_logic_textio.ALL;
USE std.textio.ALL;

 ENTITY tb_stimulus IS
 END tb_stimulus;
ARCHITECTURE behavior OF tb_stimulus IS

        COMPONENT MyDUT
      PORT(
          ABC : IN std_logic_vector(2 downto 0);
          F : OUT std_logic
      );
  END COMPONENT;

    SIGNAL ABC_TB : std_logic_vector(2 downto 0);
    SIGNAL F_TB : std_logic;

   FILE Fin : TEXT OPEN READ_MODE IS "input_file.txt";
   FILE Fout : TEXT OPEN WRITE_MODE IS "output_file.txt";

  VARIABLE current_read_line : LINE;
  VARIABLE current_write_line : LINE;
  VARIABLE current_read_field1 : STRING(1 TO 7);
  VARIABLE current_read_field2 : std_logic_vector(2 downto 0);
```

```
BEGIN

      uut: MyDUT PORT MAP (
     ABC => ABC_TB,
     F => F_TB
 );

 STIMULUS: PROCESS
  BEGIN
      WHILE NOT endfile(Fin) LOOP
          readline(Fin, current_read_line);
          read(current_read_line, current_read_field1);
          read(current_read_line, current_read_field2);

          ABC_TB <= current_read_field2;
          WAIT FOR 125 ns;

          write(current_write_line, current_read_field1);
          write(current_write_line, string'(" "));
          write(current_write_line, current_read_field2);
          write(current_write_line, string'(" DUT Output: F_TB = "));
          write(current_write_line, F_TB);
          writeline(Fout, current_write_line);
      END LOOP;
      WAIT;
  END PROCESS;
END;
```

input_file.txt | output_file.txt

```
Input1  000 DUT Output: F_TB = U
Input2  001 DUT Output: F_TB = U
Input3  010 DUT Output: F_TB = U
Input4  011 DUT Output: F_TB = U
Input5  100 DUT Output: F_TB = U
Input6  101 DUT Output: F_TB = U
Input7  110 DUT Output: F_TB = U
Input8  111 DUT Output: F_TB = U
Input8  UUU DUT Output: F_TB = U
```

1) Utilisation de l'écriture dans un fichier externe

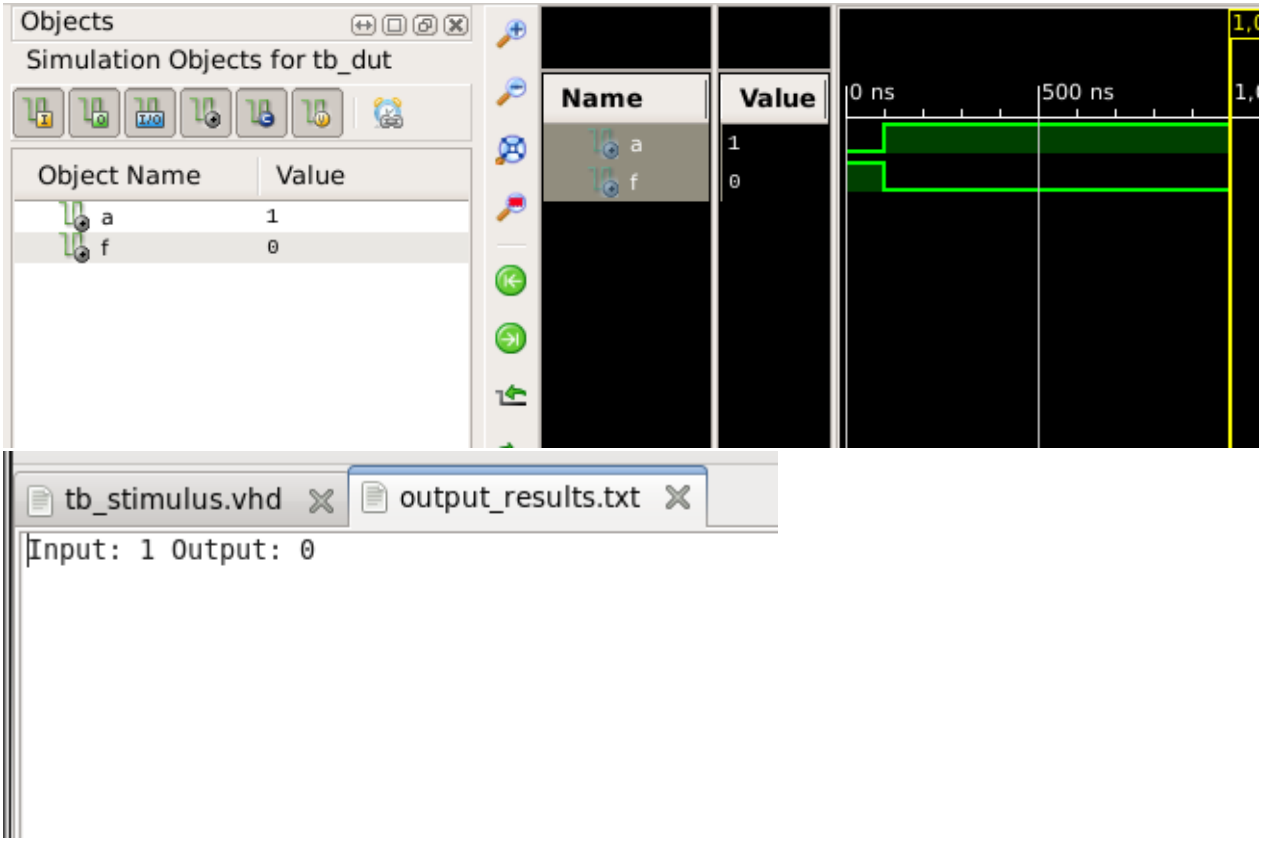
Dut.vhdl

```
19  ----------------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declaration if
24  -- arithmetic functions with Signed or Unsigned v
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity dut is
33      Port ( A : in  STD_LOGIC;
34             F : out  STD_LOGIC);
35  end dut;
36
37  architecture Behavioral of dut is
38
39  begin
40
41  F <= not A;
42
43  end Behavioral;
44
```

Tb_dut.vhdl

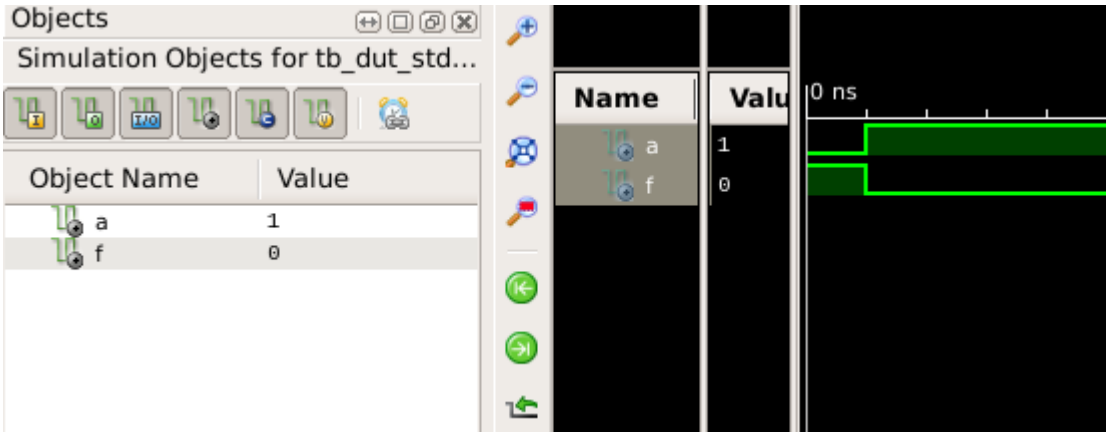
```
ENTITY tb_dut IS
END tb_dut;

ARCHITECTURE behavior OF tb_dut IS

    -- Component Declaration for the Unit Under Test (U

    COMPONENT dut
    PORT(
         A : IN  std_logic;
         F : OUT  std_logic
        );
    END COMPONENT;
   signal A : STD_LOGIC;
   signal F : STD_LOGIC;
    file Fout : TEXT open WRITE_MODE is "output_results

BEGIN
   uut: dut PORT MAP (
          A => A,
          F => F
        );
```

```
60
61      stimulus: process
62      variable Lout : line;
63      begin
64      a <= '0';
65      wait for 100 ns;
66      a <= '1';
67      wait for 100 ns;
68
69      write(Lout, string'("Input: "));
70      write(Lout, a);
71      write(Lout, string'(" Output: "));
72      write(Lout, f);
73      writeline(Fout, Lout);
74
75      wait;
76    end process;
77  END;
78
```



2) L'écriture dans STD_OUTPUT à partir d'un test bensh :

```
57  BEGIN
58      uut: dut PORT MAP (
59             A => A,
60             F => F
61           );
62      STIMULUS: process
63      variable Lout : line;
64    begin
65      a <= '0';
66      wait for 100 ns;
67      write(Lout, string'("Input: "));
68      write(Lout, a);
69      write(Lout, string'(" Output: "));
70      write(Lout, f);
71      writeline(OUTPUT, Lout);
72      a <= '1';
73      wait for 100 ns;
74      write(Lout, string'("Input: "));
75      write(Lout, a);
76      write(Lout, string'(" Output: "));
77      write(Lout, f);
78      writeline(OUTPUT, Lout);
79
80      wait;
81    end process;
82
83
84  END;
```



Console

```
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
Input: 0 Output: 1
Input: 1 Output: 0
ISim>
```

3) Lecture à partir d'un fichier externe dans un test Bench :

input_to_read.txt

```
0101
0011
0110
1001
```

```
    file input_file : text open read_mode is "input_to_read.txt";
BEGIN
    uut: dut PORT MAP (
          a => a,
          b => b,
          sum => sum
        );

   process
   variable line_content : line;
   variable data_read : std_logic_vector(3 downto 0);
   begin
   while not endfile(input_file) loop
            readline(input_file, line_content);
            read(line_content, data_read);
            a <= data_read;
            b <= "0001";

            wait for 100 ns;
            end loop;
        wait;
   end process;
END;
```

4) Lecture de données déliùitées par des espaces à partir d'un fichier externe dans un test Bench :

Dut.vhdl

```
19  ----------------------------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22  use IEEE.NUMERIC_STD.ALL;
23  -- Uncomment the following library declaration if using
24  -- arithmetic functions with Signed or Unsigned values
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declaration if insta
28  -- any Xilinx primitives in this code.
29  --library UNISIM;
30  --use UNISIM.VComponents.all;
31
32  entity dut is
33      Port ( a : in  STD_LOGIC_VECTOR(3 DOWNTO 0);
34             b : in  STD_LOGIC_VECTOR(3 DOWNTO 0);
35             sum : out  STD_LOGIC_VECTOR(3 DOWNTO 0)
36             );
37  end dut;
38
39  architecture Behavioral of dut is
40
41  begin
42  sum <= std_logic_vector(unsigned(a) + unsigned(b));
43  end Behavioral;
44
45
```

Tb_dut.vhdl

```
ARCHITECTURE behavior OF tb_dutRead IS

    -- Component Declaration for the Unit Under Test (UUT)

    COMPONENT dut
    PORT(
         a : IN  std_logic_vector(3 downto 0);
         b : IN  std_logic_vector(3 downto 0);
         sum : OUT  std_logic_vector(3 downto 0)
        );
    END COMPONENT;


   --Inputs
   signal a : std_logic_vector(3 downto 0) := (others => '0');
   signal b : std_logic_vector(3 downto 0) := (others => '0');

   --Outputs
   signal sum : std_logic_vector(3 downto 0);
   -- No clocks detected in port list. Replace <clock> below with
   -- appropriate port name
   file input_file : text open read_mode is "input_data.txt";
```

```
66    -- Instantiate the Unit Under Test (UUT)
67    uut: dut PORT MAP (
68           a => a,
69           b => b,
70           sum => sum
71         );
72
73    Stimulus : process
74
75     variable current_line : line;
76     variable input_a : STD_LOGIC_VECTOR(3 downto 0);
77     variable input_b : STD_LOGIC_VECTOR(3 downto 0);
78     begin
79     while not endfile(input_file) loop
80             readline(input_file, current_line);
81
82             read(current_line, input_a);
83             read(current_line, input_b);
84
85             a <= input_a;
86             b <= input_b;
87
88             wait for 100 ns;
89         end loop;
90
91         wait;
92     end process;
93  END;
```

input_data.txt

```
0000 0001
0010 0011
0100 0101
0110 0111
```