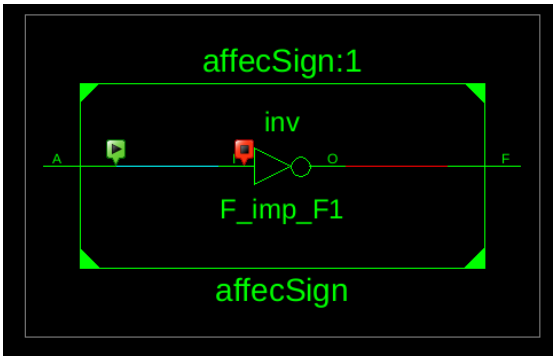
Mohamed Manessouri
SSE

Tp 3 : Modélisation séquentielle

Exercice 1 :

1) Approche d'affectation des signaux :

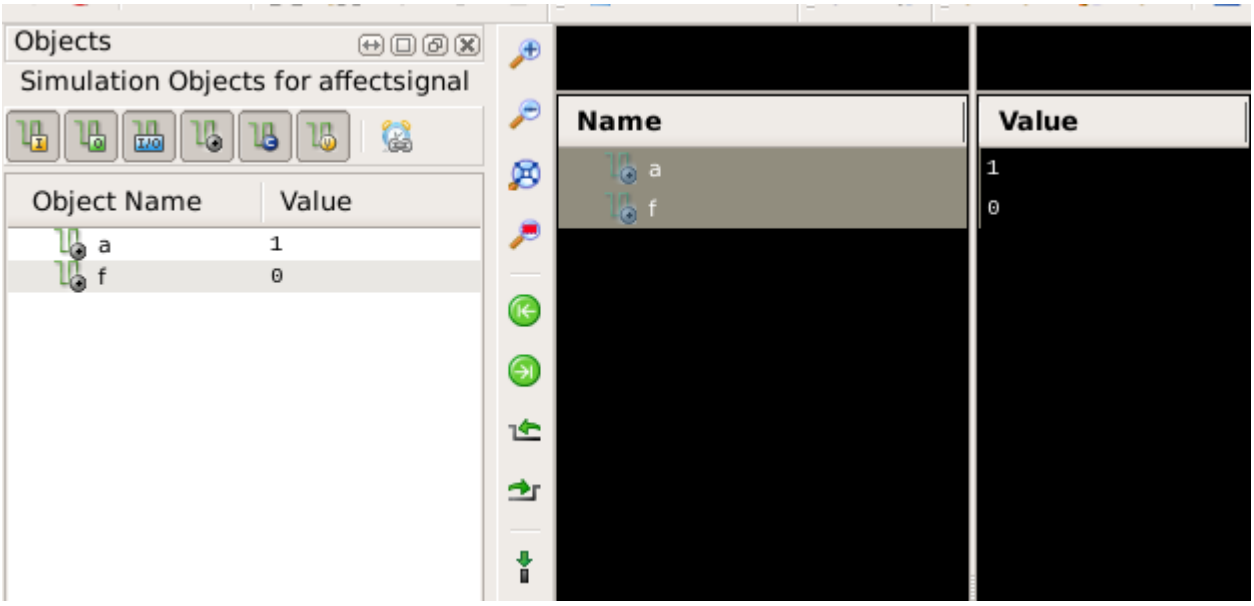
```
19  ----------------------------------------
20  library IEEE;
21  use IEEE.STD_LOGIC_1164.ALL;
22
23  -- Uncomment the following library declara
24  -- arithmetic functions with Signed or Uns
25  --use IEEE.NUMERIC_STD.ALL;
26
27  -- Uncomment the following library declara
28  -- any Xilinx primitives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

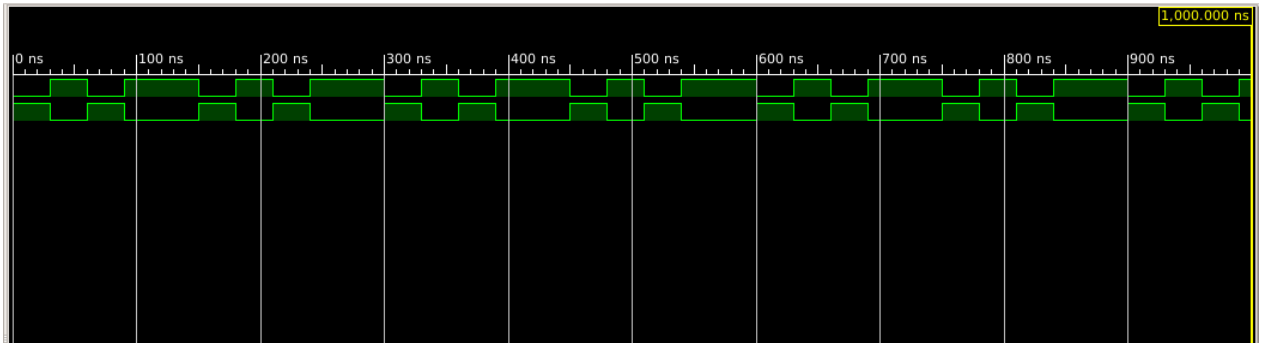


2) Visualiser les chronogrammes des signaux A, B, F :

```
26  -- simulation model.
27  --------------------------------------------------------------------
28  LIBRARY ieee;
29  USE ieee.std_logic_1164.ALL;
30
31  -- Uncomment the following library declaration if using
32  -- arithmetic functions with Signed or Unsigned values
33  --USE ieee.numeric_std.ALL;
34
35  ENTITY affectSignal IS
36  END affectSignal;
37
38  ARCHITECTURE behavior OF affectSignal IS
39
40      -- Component Declaration for the Unit Under Test (UUT)
41
42      COMPONENT affecSign
43      PORT(
44           A : IN  bit;
45           F : OUT  bit
46          );
47      END COMPONENT;
48
49
50     --Inputs
51     signal A : bit := '0';
52
53     --Outputs
54     signal F : bit;
```

```
57
58
59  BEGIN
60
61     -- Instantiate the Unit Under Test (UUT)
62     uut: affecSign PORT MAP (
63            A => A,
64            F => F
65          );
66
67       -- Stimulus process
68     stim_proc: process
69     begin
70        -- hold reset state for 100 ns.
71        A <= '0';  wait for 30 ns;
72
73        A <= '1';  wait for 30 ns;
74
75        A <= '0';  wait for 30 ns;
76
77        A <= '1';  wait for 30 ns;
78
79        A <= '1';  wait for 30 ns;
80
81     end process;
82
83
84  END;
85
```

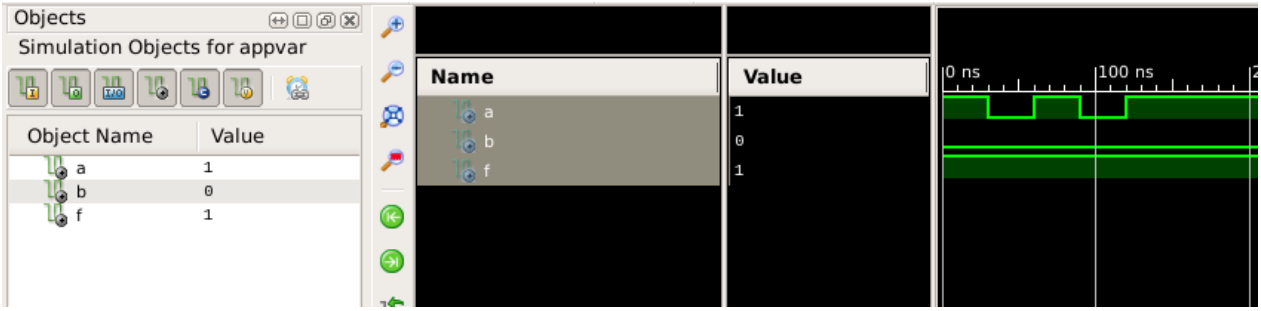

Objects
Simulation Objects for affectsignal
Object Name
Value
a
1
f
0
Name
Value
a
f
1
0


1,000.000 ns
0 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns
800 ns
900 ns

3) Approche variable :

```
30  --use UNISIM.VComponents.all;
31
32  entity approcheVar is
33      Port ( A : in  STD_LOGIC;
34             B : in  STD_LOGIC;
35             F : out  STD_LOGIC);
36  end approcheVar;
37
38  architecture Behavioral of approcheVar is
39
40  begin
41
42  var : process
43  variable B : STD_LOGIC ;
44  begin
45   B := not A;
46   F <= B;
47  end process;
48
49
50  end Behavioral;
51
```

4) Visualiser les chronogrammes des signaux A, B, F :



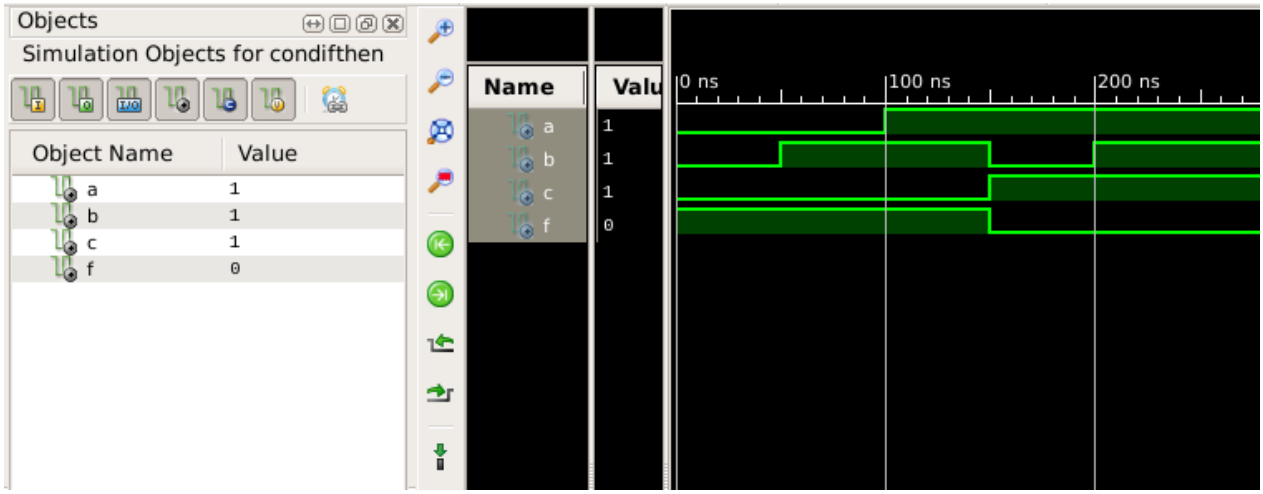
Exercice 2 :

1)

```
entity ifthen is
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           C : in  STD_LOGIC;
           F : out  STD_LOGIC);
end ifthen;

architecture Behavioral of ifthen is

begin
  process(A,B,C)
  begin
     if (A ='0' and B = '0' and C = '0') then
            F <= '1';
     elsif (A = '0' and B = '1' and C = '0') then
            F <= '1';
     elsif (A = '1' and B = '1' and C = '0') then
            F <= '1';
     else
         F <= '0';
     end if;

  end process;

end Behavioral;
```

2)

```
  stimulus: process
   begin

      A <= '0'; B <= '0'; C <= '0';
      wait for 50 ns;


      A <= '0'; B <= '1'; C <= '0';
      wait for 50 ns;


      A <= '1'; B <= '1'; C <= '0';
      wait for 50 ns;


      A <= '1'; B <= '0'; C <= '1';
      wait for 50 ns;


      A <= '1'; B <= '1'; C <= '1';
      wait for 50 ns;

      wait;
   end process;

END;
```

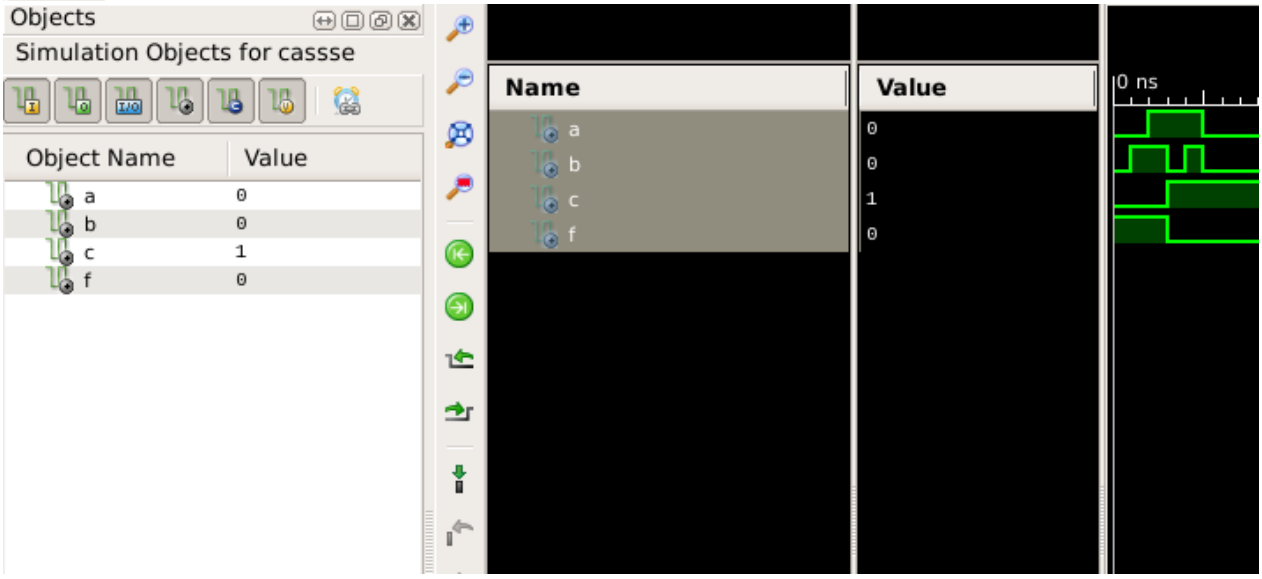


3)

```
entity cas is
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           C : in  STD_LOGIC;
           F : out  STD_LOGIC);
end cas;

architecture Behavioral of cas is

begin
process(A, B, C)
    variable input_case : std_logic_vector(2 downto 0);
  begin

    input_case := A & B & C;
    case input_case is
      when "000" => F <= '1';
      when "010" => F <= '1';
      when "110" => F <= '1';
      when others => F <= '0';
     end case;
    end process;

end Behavioral;
```

4)

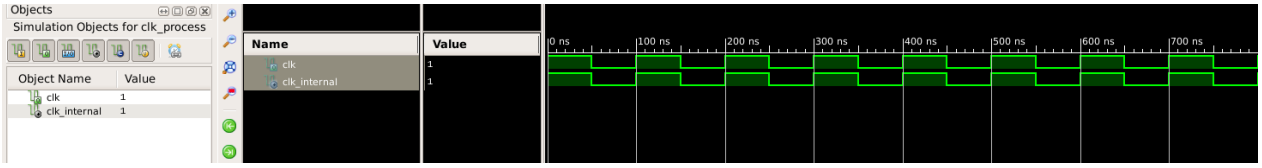
```
62
63  BEGIN
64
65     -- Instantiate the Unit Under Test (UUT)
66     uut: cas PORT MAP (
67            A => A,
68            B => B,
69            C => C,
70            F => F
71          );
72      stimulus: process
73      begin
74      A <= '0'; B <= '0'; C <= '0'; wait for 10 ns;
75
76      A <= '0'; B <= '1'; C <= '0'; wait for 10 ns;
77
78      A <= '1'; B <= '1'; C <= '0'; wait for 10 ns;
79
80      A <= '1'; B <= '0'; C <= '1'; wait for 10 ns;
81
82      A <= '1'; B <= '1'; C <= '1'; wait for 10 ns;
83
84      A <= '0'; B <= '0'; C <= '1'; wait for 10 ns;
85
86      wait;
87    end process;
88
89  END;
90
```



Example 1 :

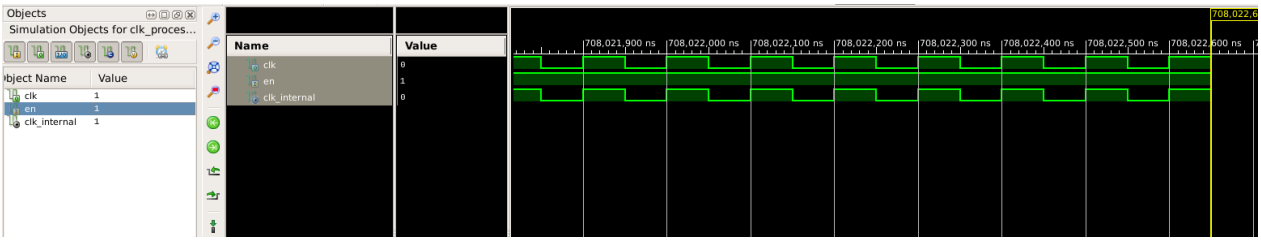
Clock process 1 :

```
31
32  entity clk_Process is
33      Port ( CLK : out  STD_LOGIC
34      );
35  end clk_Process;
36
37  architecture Behavioral of clk_Process is
38  signal clk_internal : STD_LOGIC := '0';
39  begin
40
41  clock_Proc1 : process
42    begin
43    loop
44      clk_internal <= not clk_internal;
45      wait for 50 ns;
46    end loop;
47      wait;
48
49    end process;
50
51  CLK <= clk_internal;
52  end Behavioral;
53
54
```



Clock process 2 :

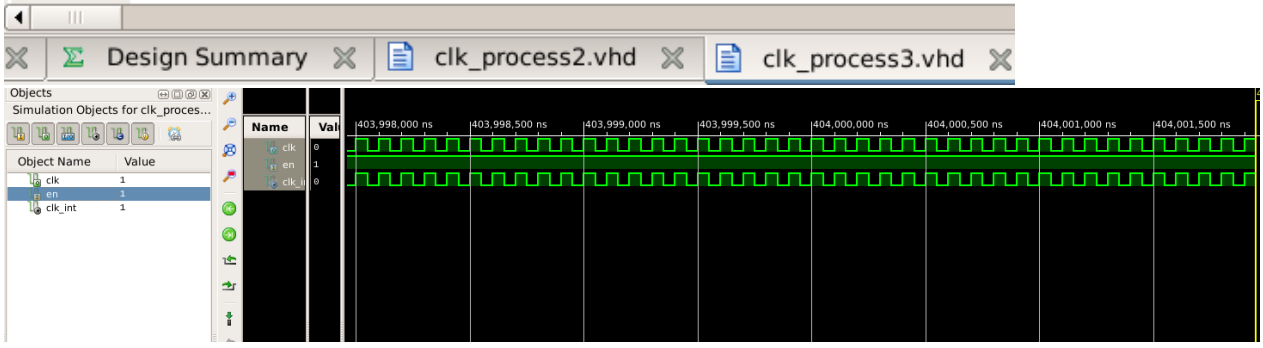
```
31
32  entity clk_process2 is
33      Port ( clk : out  STD_LOGIC;
34             EN : in  STD_LOGIC);
35  end clk_process2;
36
37  architecture Behavioral of clk_process2 is
38  signal clk_internal : STD_LOGIC := '0';
39  begin
40  clock_Proc2 : process
41      begin
42          wait until EN = '1';
43          loop
44              exit when EN = '0';
45              clk_internal <= not clk_internal;
46              wait for 50 ns;
47          end loop;
48
49          clk_internal <= '0';
50          wait until EN = '1';
51          end process;
52  clk <= clk_internal;
53
54  end Behavioral;
55
```



Example 2 :

Clock process 3 :

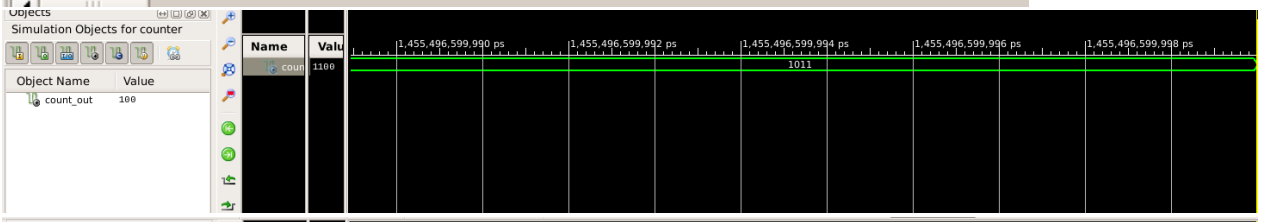
```
32  entity clk_process3 is
33      Port ( clk : out  STD_LOGIC;
34             EN : in  STD_LOGIC);
35  end clk_process3;
36
37  architecture Behavioral of clk_process3 is
38  signal clk_int : STD_LOGIC := '0';
39  begin
40  clk_process3: process
41  begin
42      while true loop
43          if EN = '1' then
44              clk_int <= not clk_int;
45              wait for 50 ns;
46          else
47              clk_int <= '0';
48              wait until EN = '1';
49          end if;
50      end loop;
51  end process;
52  clk <= clk_int;
53  end Behavioral;
54
55
```

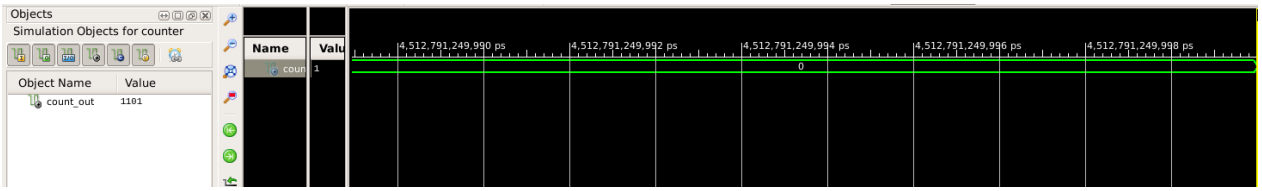


Example 3 :

Counter :

```
31
32  entity counter is
33  end counter;
34
35  architecture Behavioral of counter is
36  signal Count_Out : INTEGER := 0;
37  begin
38  COunter_proc : process
39  begin
40    for i in 0 to 15 loop
41      Count_Out <= i;
42      wait for 50 ns;
43    end loop;
44  end process;
45
46  end Behavioral;
47
48
```





Example 4:

D_flip_flop :

```
31
32  entity D_bascule is
33      Port ( Clock : in  STD_LOGIC;
34             D : in  STD_LOGIC;
35             Q : out  STD_LOGIC);
36  end D_bascule;
37
38  architecture Behavioral of D_bascule is
39
40  begin
41  D_flip_flop : process(CLock)
42   begin
43      if (CLock'event and Clock = '1') then
44        Q <= D;
45      end if;
46   end process;
47
48  end Behavioral;
49
50
```