Mohamed Manessouri
SSE

TP1 : Modélisation << Concurrent >>

Affectations simultanées de signaux << concurent Signal Assignements >> avec des opérateurs logiques :

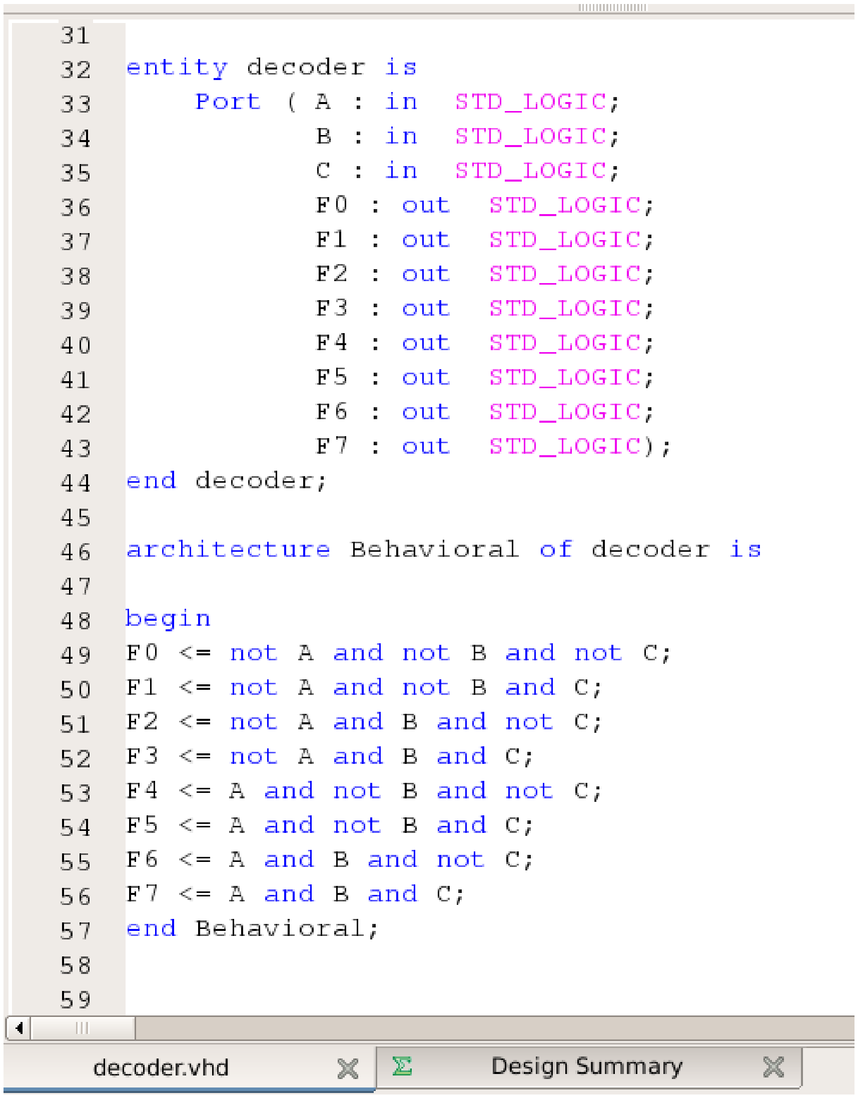
Exercice 1 : Décodeur 3-to-8 :

1) Les équations de sortie F0-F7 :

F0 = not A & not B & not C; F1 = not A & not B & C; F2 = not A & B & not C; F3 = not A & B & C;

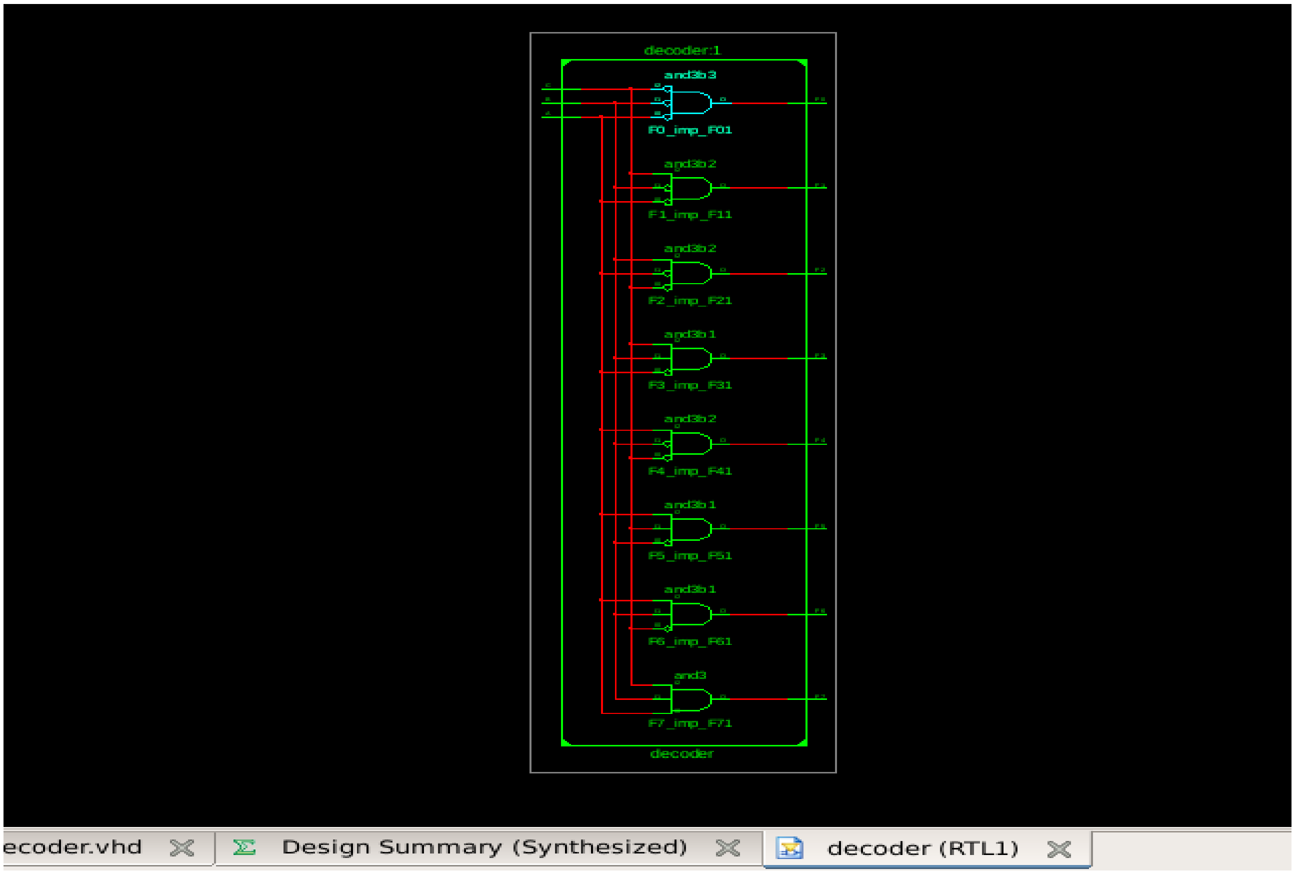
F4 = A & not B & not C; F5 = A & not B & C; F6 = A & B & not C; F7 = A & B & C;

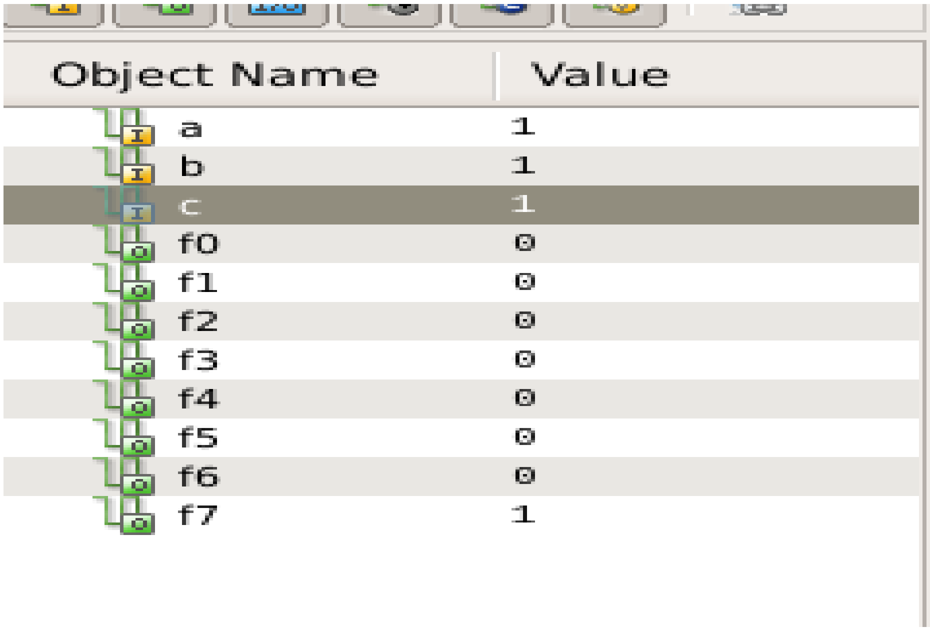
2) Modélisation VHDL du Décodeur 3-to-8 à l'aide des opérateurs logiques :



```
entity decoder is
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           C : in  STD_LOGIC;
           F0 : out  STD_LOGIC;
           F1 : out  STD_LOGIC;
           F2 : out  STD_LOGIC;
           F3 : out  STD_LOGIC;
           F4 : out  STD_LOGIC;
           F5 : out  STD_LOGIC;
           F6 : out  STD_LOGIC;
           F7 : out  STD_LOGIC);
end decoder;

architecture Behavioral of decoder is

begin
F0 <= not A and not B and not C;
F1 <= not A and not B and C;
F2 <= not A and B and not C;
F3 <= not A and B and C;
F4 <= A and not B and not C;
F5 <= A and not B and C;
F6 <= A and B and not C;
F7 <= A and B and C;
end Behavioral;
```

3) Vérification des résultats à l’aide de Isim :





Object Name	Value
a	1
b	1
c	1
f0	0
f1	0
f2	0
f3	0
f4	0
f5	0
f6	0
f7	1

Exercice 2 : Décodeur d’affichage à 7 segments :

1) Les équations Fa…Fb :

   Fa = not A and not B and not C or not A and B and not C or not A and B and C or A and not B and C or A and B and not C or A and B and C;

   Fb = not A and not B and not C or not A and B and not C or not A and B and C or A and B and not C or A and not B and not C or A and B and C;

   Fc = not A and not B and not C or not A and not B and C or not A and B and C or A and not B and not C or A and not B and C or A and B and not C or A and B and C;

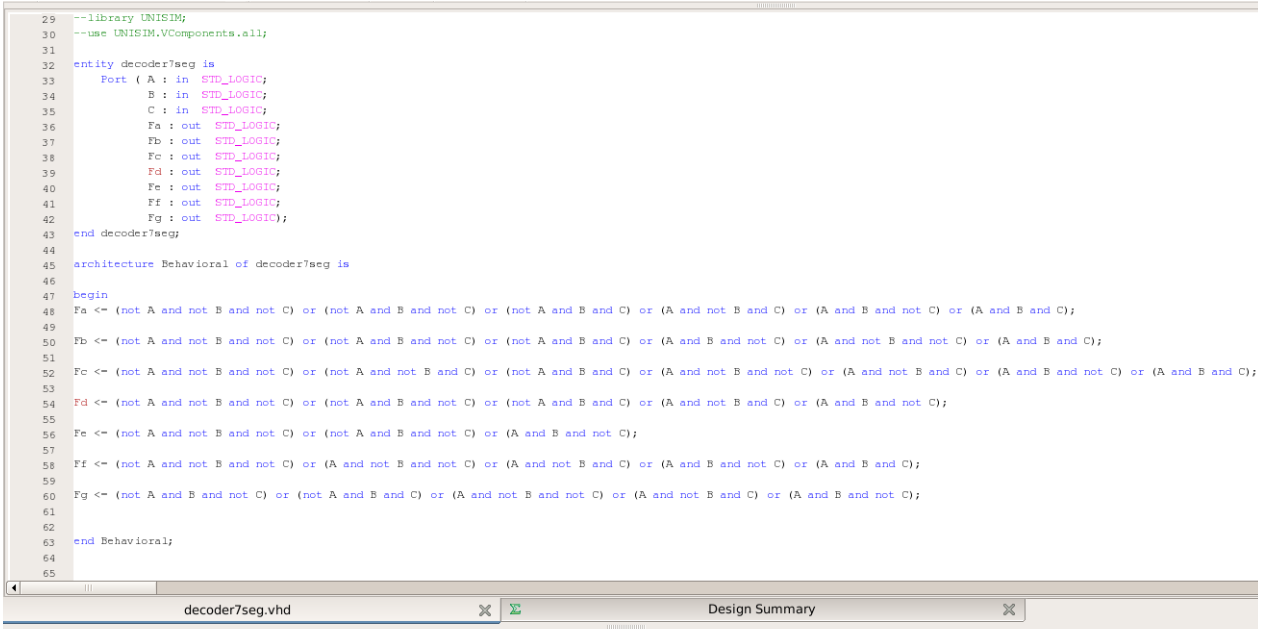
   Fd = not A and not B and not C or not A and B and not C or not A and B and C or A and not B and C or A and B and not C;

   Fe = not A and not B and not C or not A and B and not C or A and B and not C;

   Ff = not A and not B and not C or A and not B and not C or A and not B and C or A and B and not C or A and B and C;

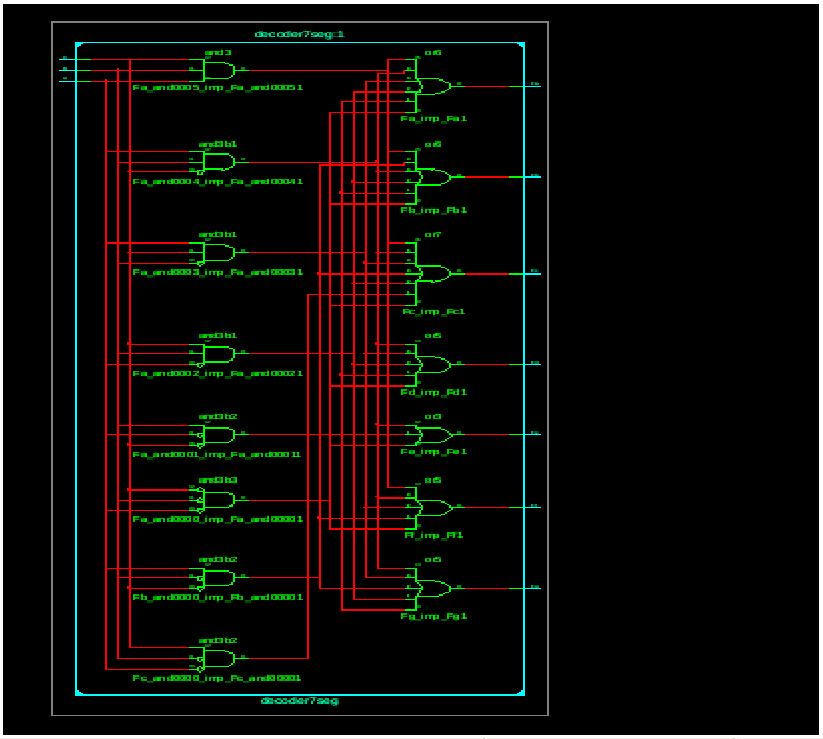
   Fg = not A and B and not C or not A and B and C or A and not B and not C or A and not B and C or A and B and not C;

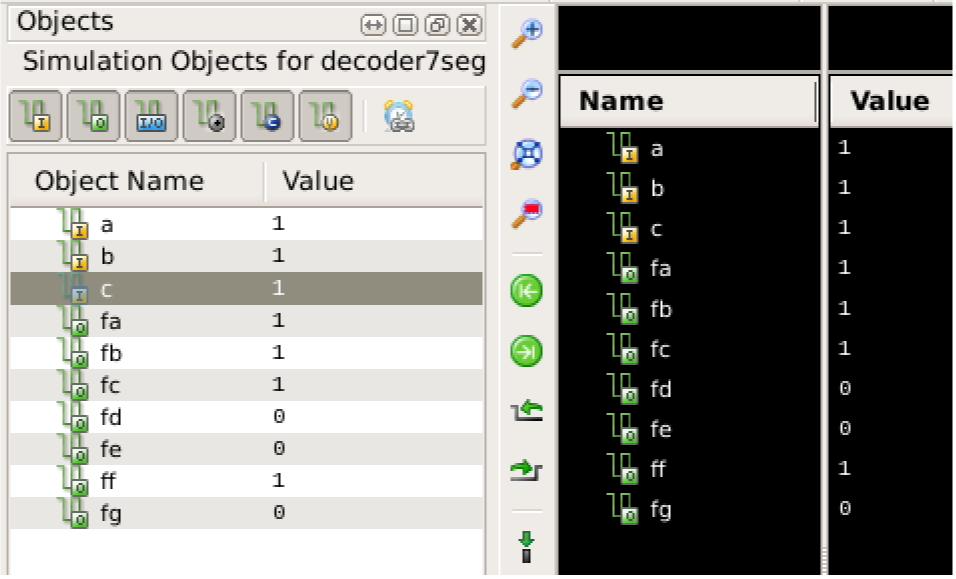
2) Modélisation VHDL du decoder 7 segments à l’aide des opérateurs logiques :



```
--library UNISIM;
--use UNISIM.VComponents.all;

entity decoder7seg is
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           C : in  STD_LOGIC;
           Fa : out  STD_LOGIC;
           Fb : out  STD_LOGIC;
           Fc : out  STD_LOGIC;
           Fd : out  STD_LOGIC;
           Fe : out  STD_LOGIC;
           Ff : out  STD_LOGIC;
           Fg : out  STD_LOGIC);
end decoder7seg;

architecture Behavioral of decoder7seg is

begin
Fa <= (not A and not B and not C) or (not A and B and not C) or (not A and B and C) or (A and not B and C) or (A and B and not C) or (A and B and C);

Fb <= (not A and not B and not C) or (not A and B and not C) or (not A and B and C) or (A and B and not C) or (A and not B and not C) or (A and B and C);

Fc <= (not A and not B and not C) or (not A and not B and C) or (not A and B and C) or (A and not B and not C) or (A and not B and C) or (A and B and not C) or (A and B and C);

Fd <= (not A and not B and not C) or (not A and B and not C) or (not A and B and C) or (A and not B and C) or (A and B and not C);

Fe <= (not A and not B and not C) or (not A and B and not C) or (A and B and not C);

Ff <= (not A and not B and not C) or (A and not B and not C) or (A and not B and C) or (A and B and not C) or (A and B and C);

Fg <= (not A and B and not C) or (not A and B and C) or (A and not B and not C) or (A and not B and C) or (A and B and not C);


end Behavioral;


```

3) Vérification des résultats à l'aide de Isim :


decoder7seg:1
and3
Fa_and0005_imp_Fa_and00051
and3b1
Fa_and0004_imp_Fa_and00041
and3b1
Fa_and0003_imp_Fa_and00031
and3b1
Fa_and0002_imp_Fa_and00021
and3b2
Fa_and0001_imp_Fa_and00011
and3b3
Fa_and0000_imp_Fa_and00001
and3b2
Fb_and0000_imp_Fb_and00001
and3b2
Fc_and0000_imp_Fc_and00001
or6
Fa_imp_Fa1
or6
Fb_imp_Fb1
or7
Fc_imp_Fc1
or6
Fd_imp_Fd1
or3
Fe_imp_Fe1
or5
Ff_imp_Ff1
or6
Fg_imp_Fg1
decoder7seg



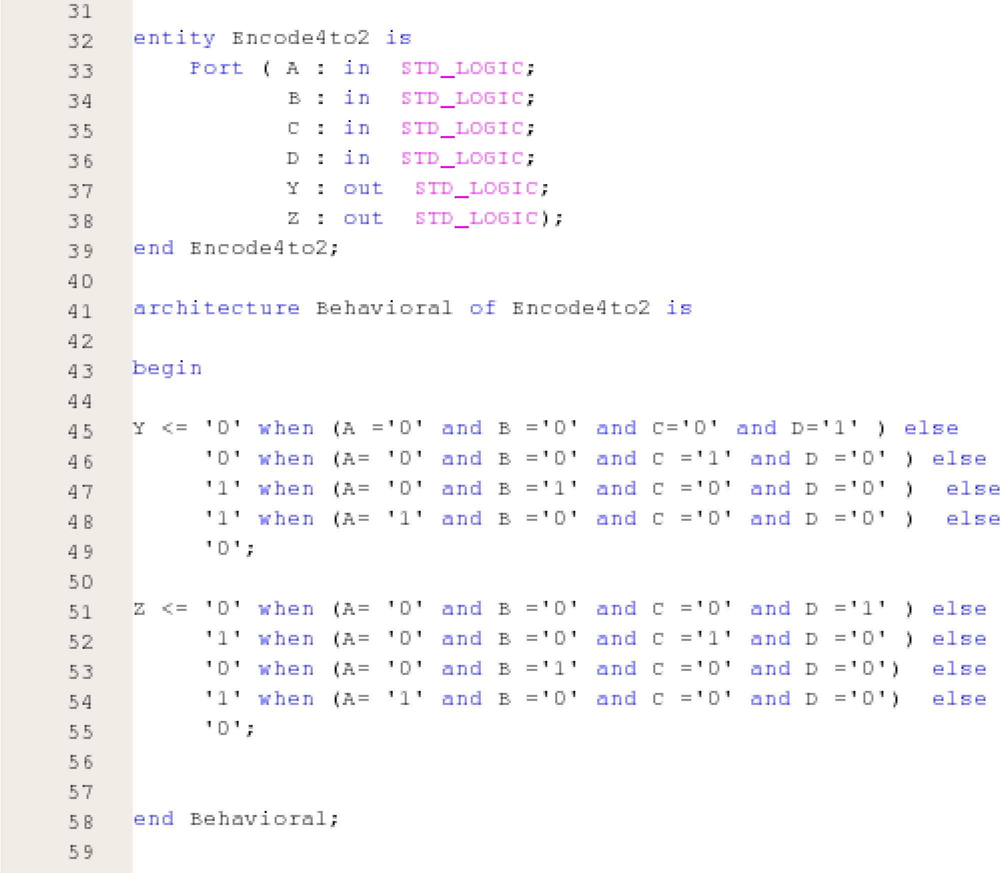
Object Name	Value
a	1
b	1
c	1
fa	1
fb	1
fc	1
fd	0
fe	0
ff	1
fg	0

Name	Value
a	1
b	1
c	1
fa	1
fb	1
fc	1
fd	0
fe	0
ff	1
fg	0

Affectations conditionnelles de signaux :

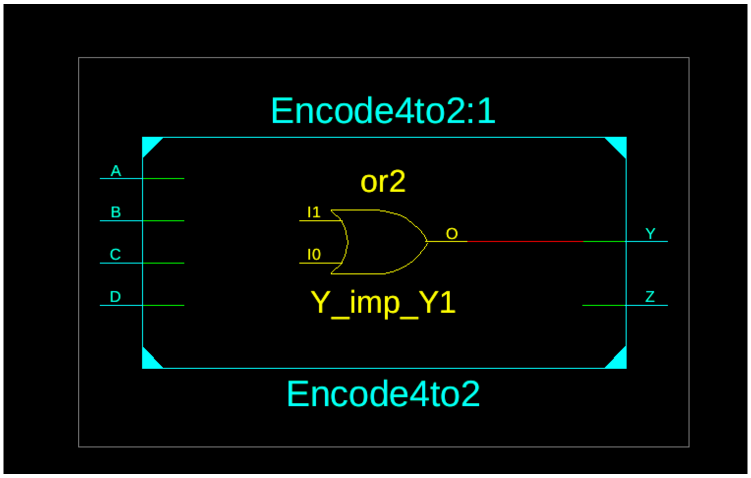
Exercice 1 :

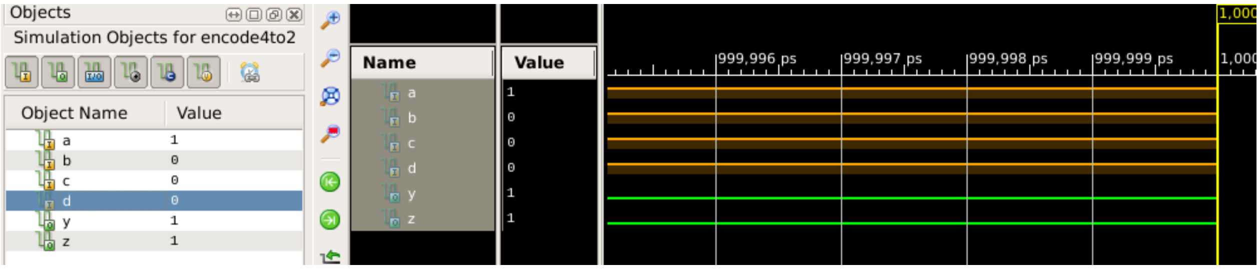
1) Les équations de sorties Y et Z :
   Y = not A and B and not C and not D or A and not B and not C and not D;
   Z = not A and not B and C and not D or A and not B and not C and not D;
2) Modélisation VHDL de l'encodeur 4-to-2 :



```

entity Encode4to2 is
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           C : in  STD_LOGIC;
           D : in  STD_LOGIC;
           Y : out  STD_LOGIC;
           Z : out  STD_LOGIC);
end Encode4to2;

architecture Behavioral of Encode4to2 is

begin

Y <= '0' when (A ='0' and B ='0' and C='0' and D='1' ) else
     '0' when (A= '0' and B ='0' and C ='1' and D ='0' ) else
     '1' when (A= '0' and B ='1' and C ='0' and D ='0' )  else
     '1' when (A= '1' and B ='0' and C ='0' and D ='0' )  else
     '0';

Z <= '0' when (A= '0' and B ='0' and C ='0' and D ='1' ) else
     '1' when (A= '0' and B ='0' and C ='1' and D ='0' ) else
     '0' when (A= '0' and B ='1' and C ='0' and D ='0')  else
     '1' when (A= '1' and B ='0' and C ='0' and D ='0')  else
     '0';


end Behavioral;

```

3) Vérification des résultats :





Objects

Simulation Objects for encode4to2

Object Name	Value
a	1
b	0
c	0
d	0
y	1
z	1

Name	Value
a	1
b	0
c	0
d	0
y	1
z	1

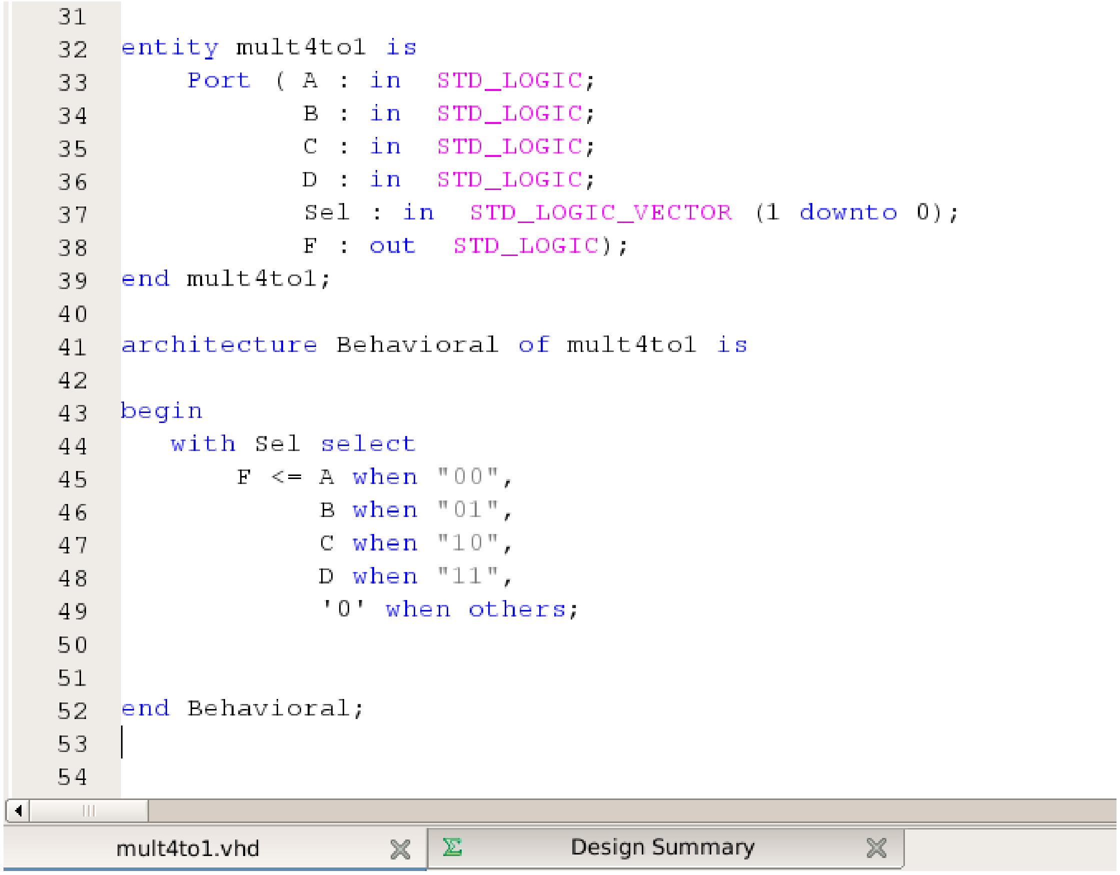
Affectations sélectionnées de signaux :

Exercice1 :

1) L'équation de F :

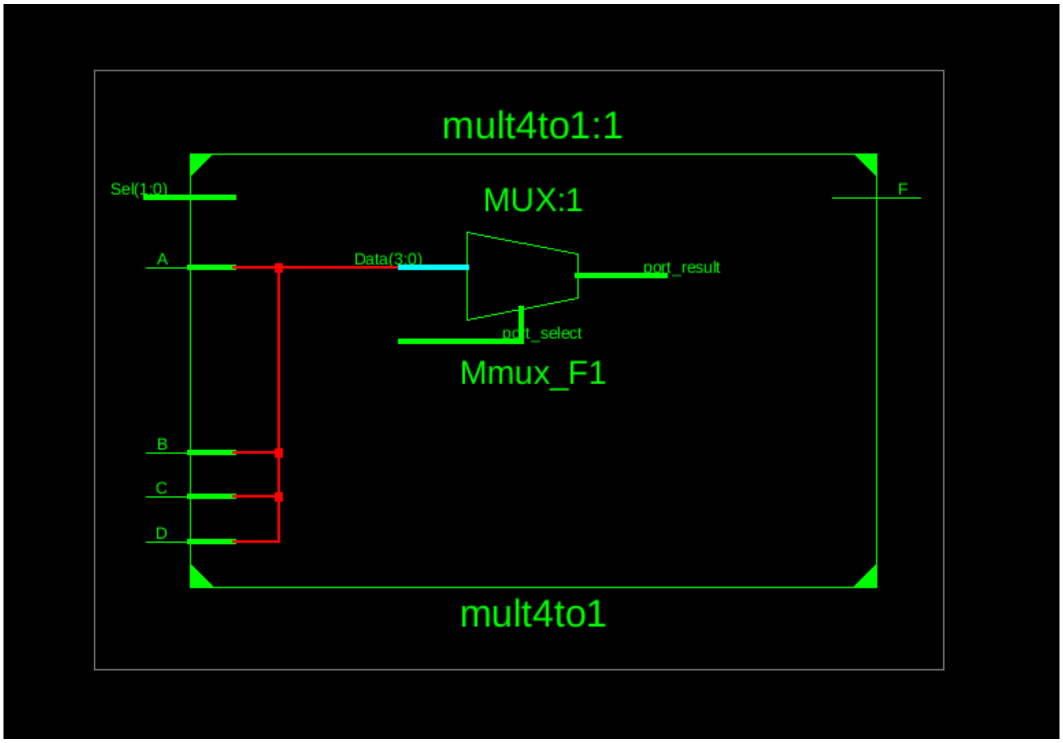
F = not sel1 and not sel0 and A
Or not sel1 and sel0 and B
Or sel1 and not sel0 and C
Or sel1 and sel0 and D;

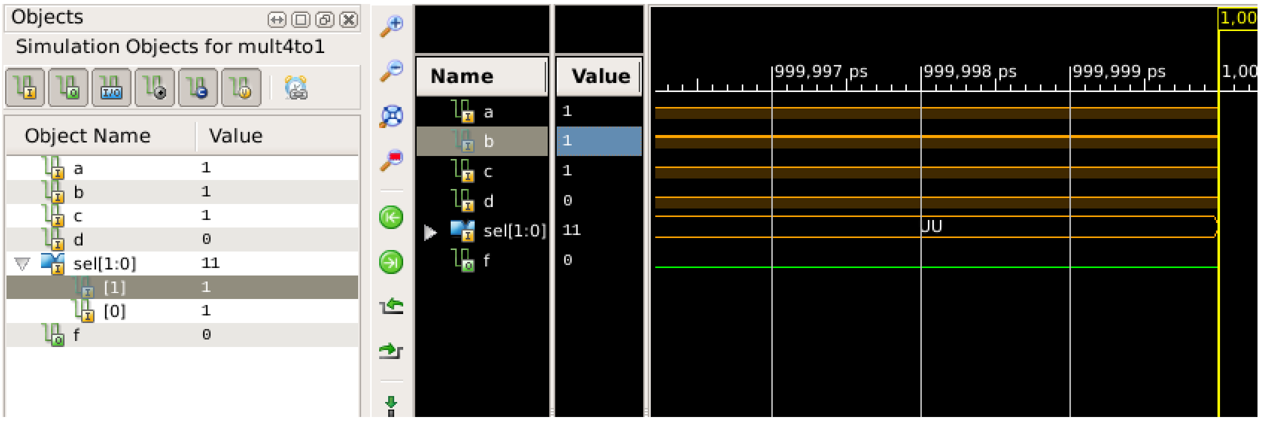
2) Modélisation VHDL:



```
entity mult4to1 is
    Port ( A : in  STD_LOGIC;
           B : in  STD_LOGIC;
           C : in  STD_LOGIC;
           D : in  STD_LOGIC;
           Sel : in  STD_LOGIC_VECTOR (1 downto 0);
           F : out  STD_LOGIC);
end mult4to1;

architecture Behavioral of mult4to1 is

begin
   with Sel select
       F <= A when "00",
            B when "01",
            C when "10",
            D when "11",
            '0' when others;


end Behavioral;

```

3) Vérification des résultats:




Objects

Simulation Objects for mult4to1

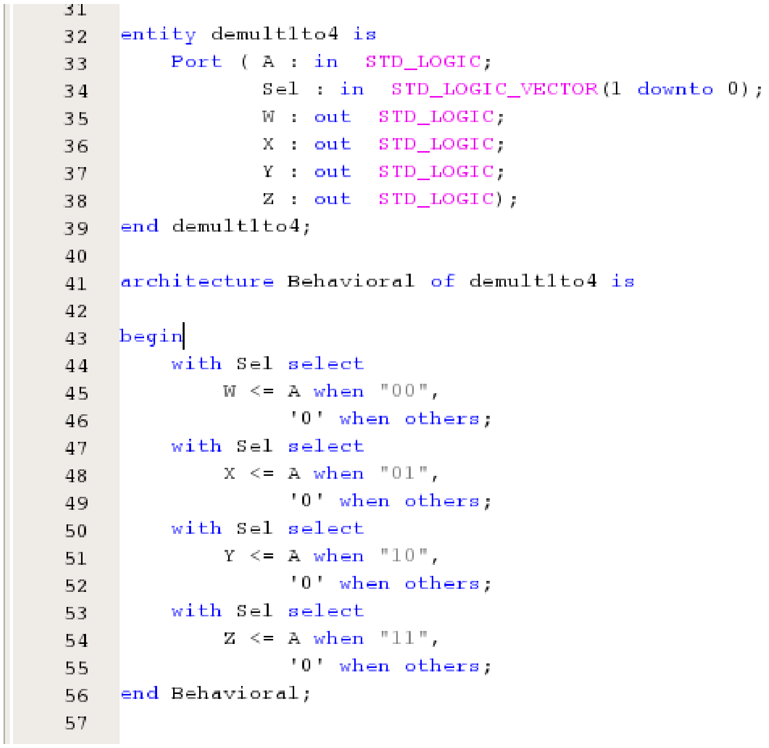
Object Name	Value
a	1
b	1
c	1
d	0
sel[1:0]	11
[1]	1
[0]	1
f	0

Name	Value
a	1
b	1
c	1
d	0
sel[1:0]	11
f	0

Exercice 2:

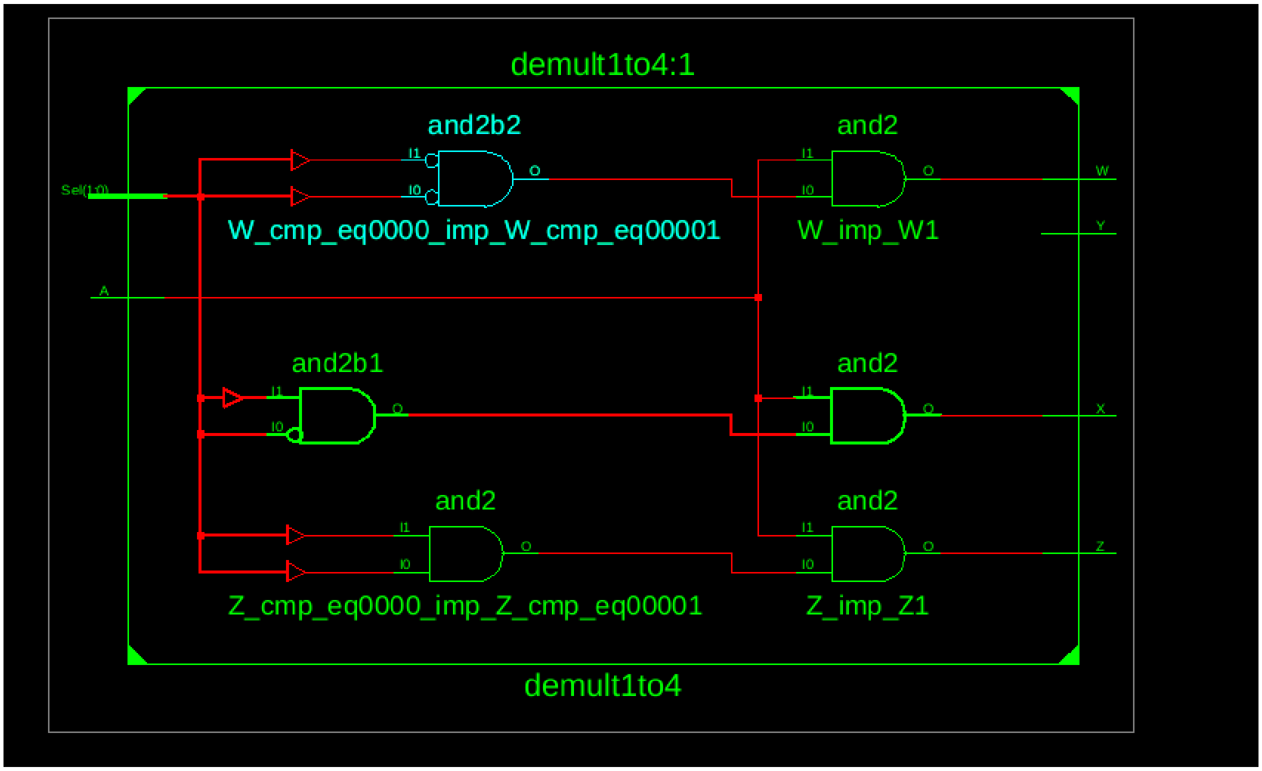
1) Les équations de sortie de F :
W = not sel1 and not sel0 and A;
X = not sel1 and sel0 and A;
Y = sel1 and not sel0 and A;
Z = sel1 and sel0 and A;

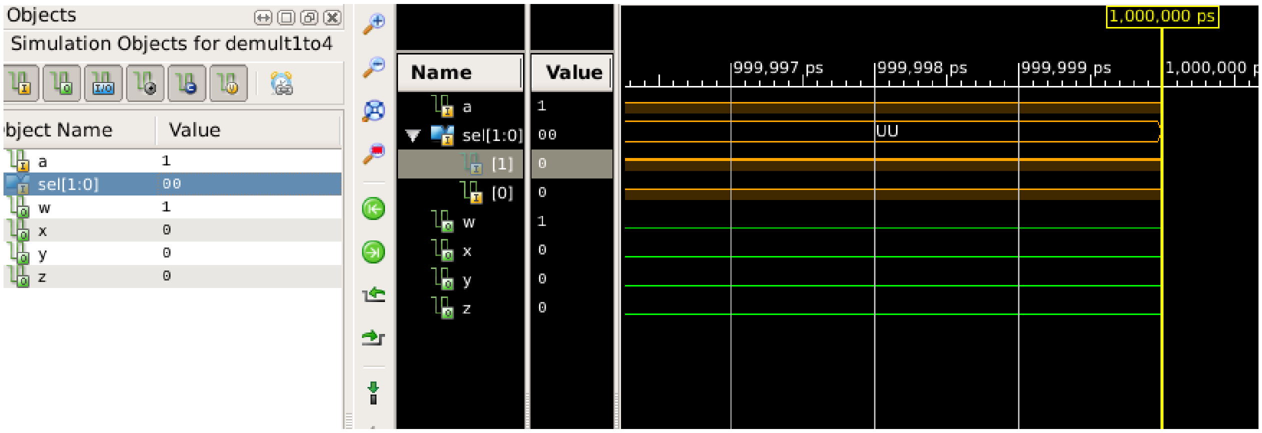
2) Modélisation VHDL :



```
entity demult1to4 is
    Port ( A : in  STD_LOGIC;
           Sel : in  STD_LOGIC_VECTOR(1 downto 0);
           W : out  STD_LOGIC;
           X : out  STD_LOGIC;
           Y : out  STD_LOGIC;
           Z : out  STD_LOGIC);
end demult1to4;

architecture Behavioral of demult1to4 is

begin
    with Sel select
        W <= A when "00",
             '0' when others;
    with Sel select
        X <= A when "01",
             '0' when others;
    with Sel select
        Y <= A when "10",
             '0' when others;
    with Sel select
        Z <= A when "11",
             '0' when others;
end Behavioral;
```

3) Vérification des résultats :





Objects

Simulation Objects for demult1to4

bject Name	Value
a	1
sel[1:0]	00
w	1
x	0
y	0
z	0

Name	Value
a	1
sel[1:0]	00
[1]	0
[0]	0
w	1
x	0
y	0
z	0