Mohamed Manessouri
SSE

Tp2 : Design Structurel et hiérarchie

1) Components en VHDL

En VHDL, un component est une abstraction qui représente un module matériel réutilisable dans une conception hiérarchique. Un component décrit l'interface d'un bloc logique, notamment ses entrées et sorties, sans définir directement son comportement interne. Il est utilisé pour instancier des modules dans une architecture supérieure.

Exercice :

1)Shéma et les équations de sortie d'un full_adder , à partir d'un half adder :

Sum = A xor B xor Cin ;
Cout = A and B or B and Cin or A and Cin ;

2) Programme Vhdl pour la modélisation :

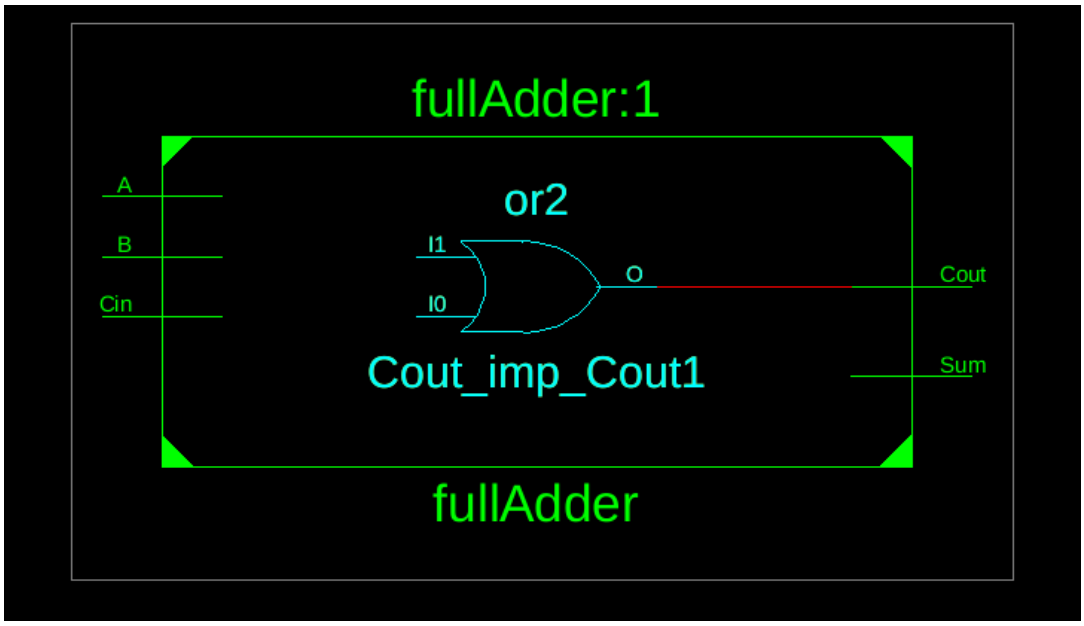
Half_adder.vhdl

```
32  entity halfAdder is
33      Port ( A : in  STD_LOGIC;
34             B : in  STD_LOGIC;
35             Sum : out  STD_LOGIC;
36             Cout : out  STD_LOGIC);
37  end halfAdder;
38
39  architecture Behavioral of halfAdder is
40
41  begin
42     Sum <= A xor B;
43     Cout <= A and B;
44  end Behavioral;
45
46
```

Full_adder.vhdl

```
31
32  entity fullAdder is
33      Port ( A : in  STD_LOGIC;
34             B : in  STD_LOGIC;
35             Cin : in  STD_LOGIC;
36             Sum : out  STD_LOGIC;
37             Cout : out  STD_LOGIC);
38  end fullAdder;
39
40  architecture Behavioral of fullAdder is
41
42  component halfAdder is
43          Port (
44              A : in STD_LOGIC;
45              B : in STD_LOGIC;
46              Sum : out STD_LOGIC;
47              Cout : out STD_LOGIC
48          );
49      end component;
50
51  signal Sum1, Cout1, Cout2 : STD_LOGIC;
52
```

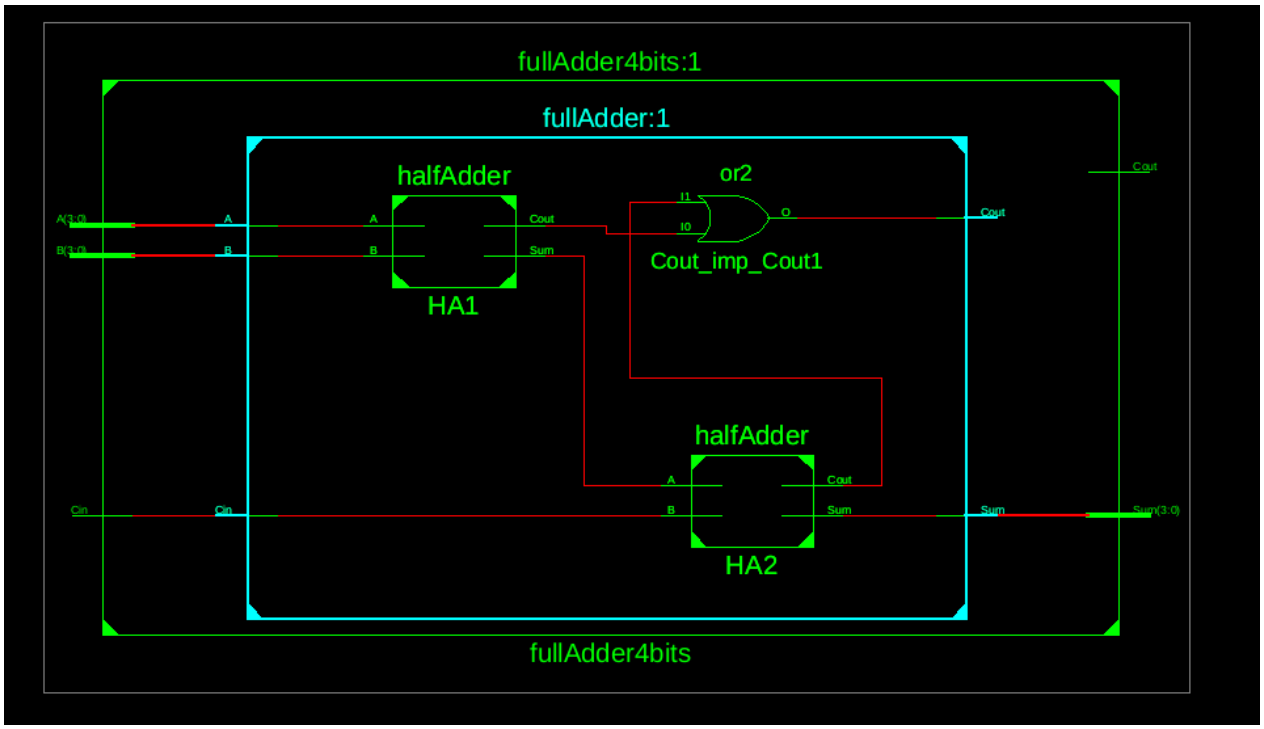
```
54  begin
55
56  HA1: halfAdder
57          port map (
58              A     => A,
59              B     => B,
60              Sum   => Sum1,
61              Cout  => Cout1
62          );
63
64
65  HA2: halfAdder
66          port map (
67              A     => Sum1,
68              B     => Cin,
69              Sum   => Sum,
70              Cout  => Cout2
71          );
72
73  Cout <= Cout1 or Cout2;
74
75  end Behavioral;
```



3) Réalisation d'un adder 4 bits à partir des full_adder :

```
32  entity fullAdder4bits is
33      Port (
34          A : in STD_LOGIC_VECTOR(3 downto 0);
35          B : in STD_LOGIC_VECTOR(3 downto 0);
36          Cin : in STD_LOGIC;
37          Sum : out STD_LOGIC_VECTOR(3 downto 0);
38          Cout : out STD_LOGIC
39          );
40
41  end fullAdder4bits;
42
43  architecture Behavioral of fullAdder4bits is
44
45  component fullAdder is
46          Port (
47              A    : in STD_LOGIC;
48              B    : in STD_LOGIC;
49              Cin  : in STD_LOGIC;
50              Sum  : out STD_LOGIC;
51              Cout : out STD_LOGIC
52          );
53      end component;
54  signal Cout1, Cout2, Cout3 : STD_LOGIC;
55
```

```
56  begin
57
58  FA1: fullAdder
59          port map (
60              A    => A(0),
61              B    => B(0),
62              Cin  => Cin,
63              Sum  => Sum(0),
64              Cout => Cout1
65          );
66  FA2: fullAdder
67          port map (
68              A    => A(1),
69              B    => B(1),
70              Cin  => Cout1,
71              Sum  => Sum(1),
72              Cout => Cout2
73          );
74  FA3: fullAdder
75          port map (
76              A    => A(2),
77              B    => B(2),
78              Cin  => Cout2,
79              Sum  => Sum(2),
80              Cout => Cout3
81          );
82  FA4: fullAdder
83          port map (
84              A    => A(3),
85              B    => B(3),
86              Cin  => Cout3,
87              Sum  => Sum(3),
88              Cout => Cout
89          );
90
91
92  end Behavioral;
```



Objects

Simulation Objects for fulladder4...

Object Name	Value
a[3:0]	1100
[3]	1
[2]	1
[1]	0
[0]	0
b[3:0]	1010
[3]	1
[2]	0
[1]	1
[0]	0
cin	0
sum[3:0]	0110
cout	1
cout1	0
cout2	0
cout3	0

Name	Value
a[3:0]	UUUU
b[3:0]	UUUU
cin	U
sum[3:0]	UUUU
cout	U
cout1	U
cout2	U
cout3	U