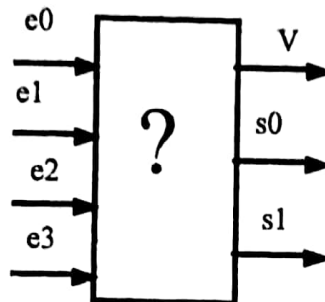


Exercise 1 :

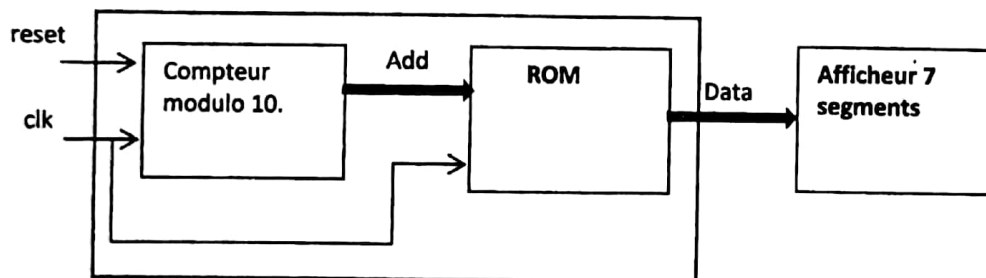
Si une seule des entités est présente, l'encodeur de priorité suivant :

Si une seule des entrées est au niveau 1, alors $V = 1$ et S150 indique en binaire le rang (i) de cette variable d'entrée. Si plusieurs entrées sont au niveau 1, alors $V = 1$ et S150 indique en binaire le rang (i) le plus élevé des variables d'entrées égales à 1. Si toutes les entrées sont au niveau 0 alors $V = 0$ et la valeur de S150 est quelconque.



Exercice 2 :

On désire réaliser un compteur modulo 10 (qui compte de 0 à 9 et revient à 0) avec affichage 7 segments en utilisant une ROM. Proposer un programme VHDL qui réalise cette fonction.



Exercice 3 :

Le processus général de multiplication binaire est le suivant (sur deux bits)

		A1	A0	
		B1	B0	Multiplicande
		<hr/>		Multiplieur
		A1B0	A0B0	Produit partiel
	A1B1	A0B1		Produit partiel
	<hr/>			
P3	P2	P1	P0	Résultat

Cette opération se résume en une somme de produits partiels. Elle peut être réalisée par une structure combinatoire en cascade d'additionneurs. Proposer un programme VHDL qui réalise ce circuit.

Solution Examen FPGA et VHDL.

①

Exo 1 :

table de vérité de l'encodeur

e3	e2	e1	e0	V	S1	S0
0	0	0	0	0	x	x
0	0	0	1	1	0	0
0	0	1	x	1	0	1
0	1	x	x	1	1	0
1	x	x	x	1	1	1

1

programme VHDL de l'encodeur

6 pts

Library ieee;

use ieee.std_logic_1164.all;

Entity encodeur is

Port (

E : in std_logic_vector (3 downto 0);

S : out std_logic_vector (2 downto 0);

end encodeur

-- E = e3e2e1e0

-- S = V S2 S1

Architecture description of encoder is
begin

S <= "0--" when E = "0000" else

"100" when E = "0001" else

"101" when E = "001-" else

"110" when E = "01--" else

"111";

4 pts

end description;

Exo2: (7pts)

library ieee;
use ieee.std_logic_1164.all;

Entity projet is

port (

reset, clk : in std_logic;

data : out std_logic_vector(7 downto 0)); 1pt

end projet;

Architecture aproj of projet is

signal add : std_logic_vector(3 downto 0); 0,5

Component Compt is

port (

reset, clk : in std_logic;

add : out std_logic_vector); 1pt

end component;

Component ProgROM is

port (

clk : in std_logic;

add : in std_logic_vector(3 downto 0); 1pt

data : out std_logic_vector(6 downto 0));

end component;

begin

(4)

0.5 C_1 : compt Port map (reset \Rightarrow reset, clk \Rightarrow clk, add \Rightarrow add);

0.5 C_2 : Prog ROT Port map (clk \Rightarrow clk, add \Rightarrow add, data \Rightarrow data);

end projet;

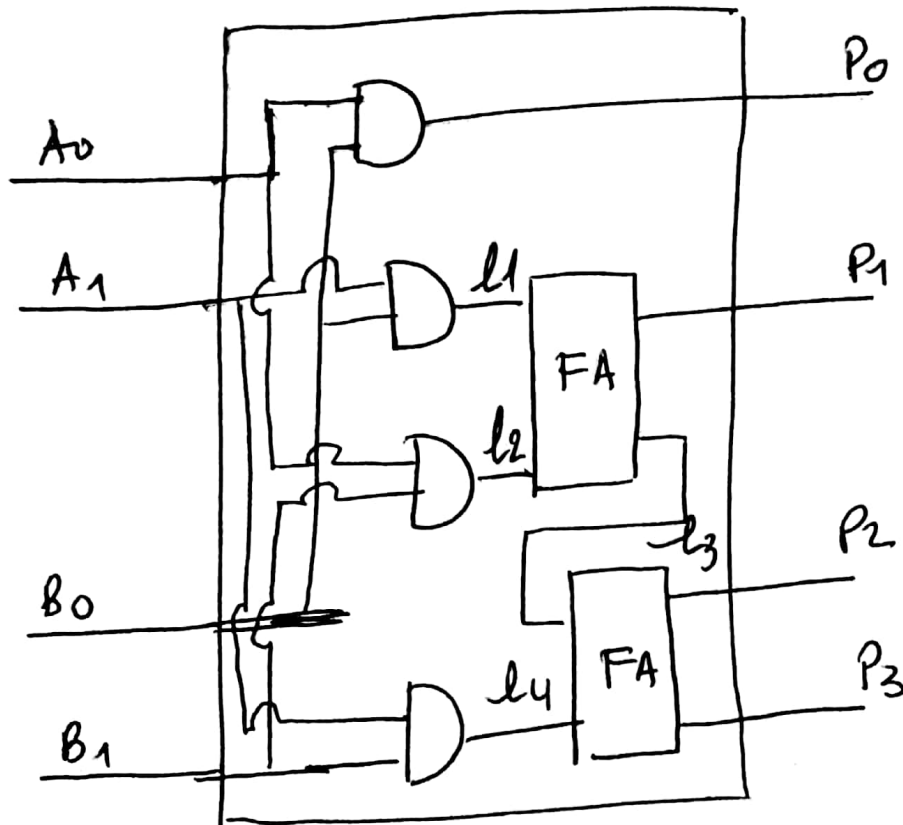
pour les programmes du compteur et de la ROT
voir le cours.

1.5 compteur

1.5 ROT.

Ex03: (7pts)

(5)



1 pr

Programme VHDL de la multiplication 2 bits

Library ieee;
use ieee.std_logic_1164.all;

Entity projet2 is

Port (

A, B : in std_logic_vector (1 down to 0);

P : out std_logic_vector (3 down to 0));

end projet2;

1 pr

Architecture a project of project 2 is

⑥

signal l1, l2, l3, l4: std_logic; 0,5

Component ET is

```
port(  
  e1, e2 : in std_logic;  
  s : out std_logic); 0,5  
end Component;
```

Component FA is

```
port(  
  a, b and : in std_logic;  
  s, co : out std_logic); 0,5  
end Component;
```

Begin 2/4

```
c1: ET Port map (e1 => A0, e2 => B0, s => P0);  
c2: ET Port map (e1 => A1, e2 => B0, s => l1);  
c3: ET Port map (e1 => A0, e2 => B1, s => l2);  
c4: ET Port map (e1 => A1, e2 => B1, s => l4);  
c5: FA Port map (a => l1, b => l2, s => P1, co => l3);  
c6: FA Port map (a => l3, b => l4, s => P2, co => P3);  
end architecture;
```