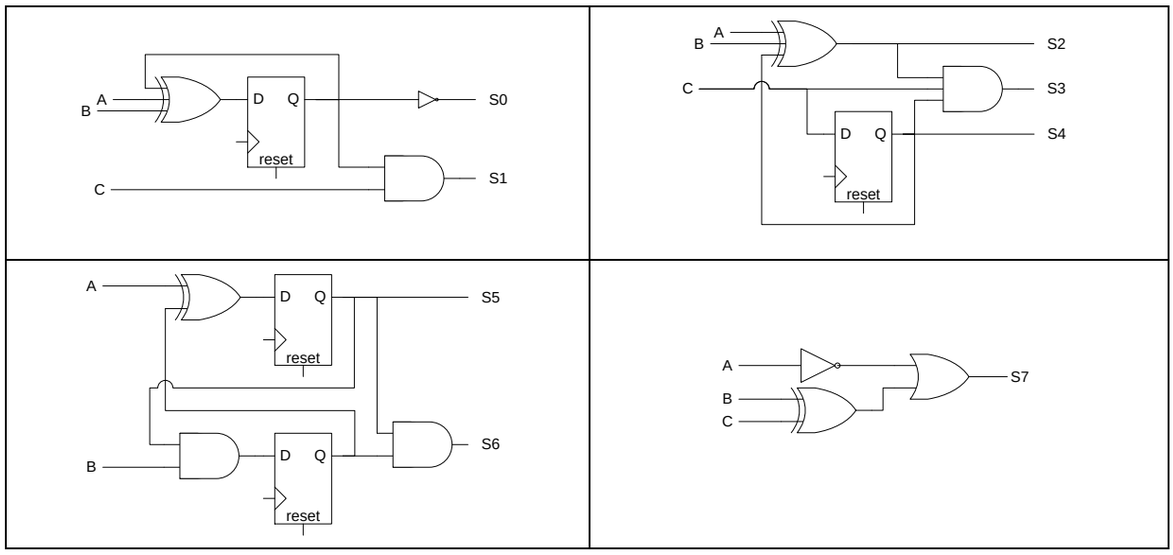
0401 Composantes séquentielles pour circuits numériques

1. Pour chacun des circuits suivants, indiquez, pour chacune des sorties, s'il s'agit d'une sortie combinatoire, d'une sortie de Moore ou d'une sortie de Mealy.



0402 VHDL pour circuits séquentiels

2. Donnez le code VHDL pour le registre suivant : 8 bits; chargement sur une transition négative de l'horloge si le signal charge = '1'; réinitialisation synchrone à F4h quand le signal reset est égal à '0'.

0403 Synthèse d'un circuit séquentiel à partir de sa description en VHDL

3. Donnez un diagramme du circuit modélisé par les codes VHDL suivants.

```
library ieee;
use ieee.std_logic_1164.all;

entity mystere1 is
   port (a, b, c: in std_logic;
   s : in std_logic_vector (1 downto 0);
   o : out std_logic);
end mystere1;

architecture archi of mystere1 is
begin
   process (a, b, c, s)
   begin
      if (s = "00") then o <= a;
         elsif (s = "01") then o <= b;
         elsif (s = "10") then o <= c;
      end if;
   end process;
end archi;
```

```
library ieee;
use ieee.std_logic_1164.all;

entity mystere2 is
   port (a, b, c: in std_logic;
   s : in std_logic_vector (1 downto 0);
   o : out std_logic);
end mystere2;

architecture archi of mystere2 is
begin
   process (a, b, c, s)
   begin
      if (s = "00") then o <= a;
         elsif (s = "01") then o <= b;
         elsif (s = "10") then o <= c;
         else o <= c;
      end if;
   end process;
end archi;
```

4. Considérez les codes VHDL suivants, et indiquez pourquoi, bien qu'ils puissent être compilés correctement et même simulés, il ne faut pas les utiliser.

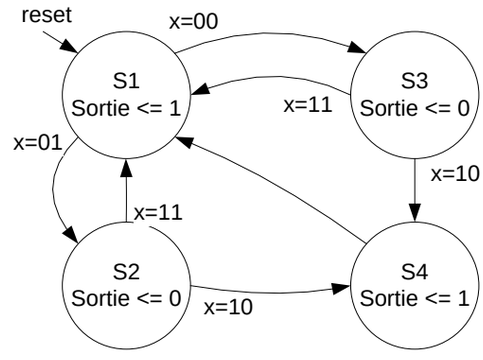
```
library ieee;
use ieee.std_logic_1164.all;

entity basculeBAD is
   port (
      CLK, reset, D : in STD_LOGIC;
      Q : out STD_LOGIC
   );
end basculeBAD;

architecture archiBAD1 of basculeBAD is
begin
   process(CLK) is
   begin
      if (CLK = '1' and CLK'event
         and reset = '1') then
         Q <= D;
      end if;
   end process;
end;

architecture archiBAD2 of basculeBAD is
begin
   process(CLK, reset) is
   begin
      if reset = '0'
         and not(CLK'event) then
         Q <= '0';
      elsif falling_edge(CLK) then
         Q <= D;
      end if;
   end process;
end;
```

```
architecture archiBAD3 of basculeBAD is
begin
   process(CLK) is
   begin
      if rising_edge(CLK)
         and reset /= '0' then
         Q <= D;
      end if;
   end process;
end;

architecture archiBAD4 of basculeBAD is
begin
   process(CLK) is
   begin
      Q <= D;
   end process;
end;
```

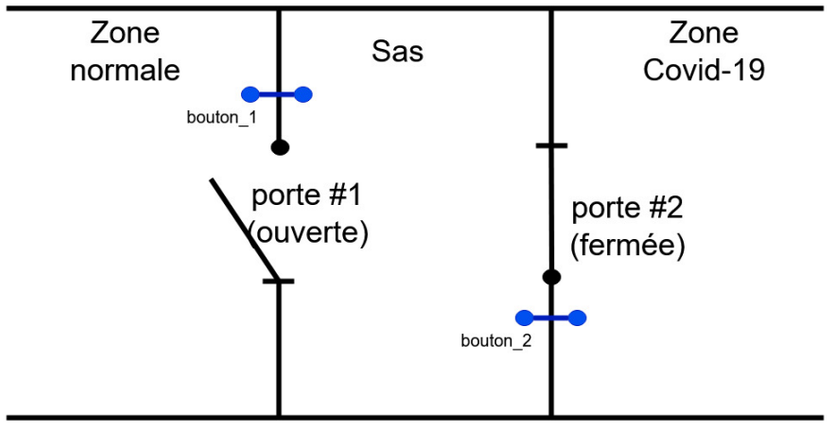
0405 Analyse de machines à états et description en VHDL

5. Donnez une architecture pour l'entité suivante en VHDL synthétisable afin qu'elle corresponde au diagramme d'états donné. Utilisez un signal de réinitialisation asynchrone actif sur le niveau '0'.

```
library IEEE;
use IEEE.std_logic_1164.all;

entity machineAEtats is
   port (
   reset, CLK : in STD_LOGIC;
   x : in STD_LOGIC_VECTOR(1 downto 0);
   sortie : out STD_LOGIC
   );
end machineAEtats;
```



6. L'hôpital Maisonneuve-Rosemont a besoin d'un système de contrôle pour un sas pour circuler entre la zone normale et la zone Covid-19 de l'hôpital. Le sas est formé d'une petite pièce avec deux portes, une vers chaque zone, et est montré ici.



Le système a quatre entrées :

- porte_1_fermée pour indiquer si la porte #1 est fermée ou ouverte
- porte_2_fermée pour indiquer si la porte #2 est fermée ou ouverte
- bouton_1 pour demander de déverrouiller la porte #1 (il y a deux instances du bouton #1, une à l'intérieur du sas et une à l'extérieur près de la porte #1)
- bouton_2 pour demander de déverrouiller la porte #2 (il y a deux instances du bouton #2, une à l'intérieur du sas et une à l'extérieur près de la porte #2)

Le système a trois sorties :

- barrer_1, pour verrouiller la porte #1
- barrer_2, pour verrouiller la porte #2
- alarme, pour activer une alarme si les deux portes sont ouvertes en même temps

Au départ, toutes les portes sont fermées, ou bien l'alarme doit sonner. Ensuite, le système de contrôle ne doit permettre que l'ouverture d'une seule porte à la fois.

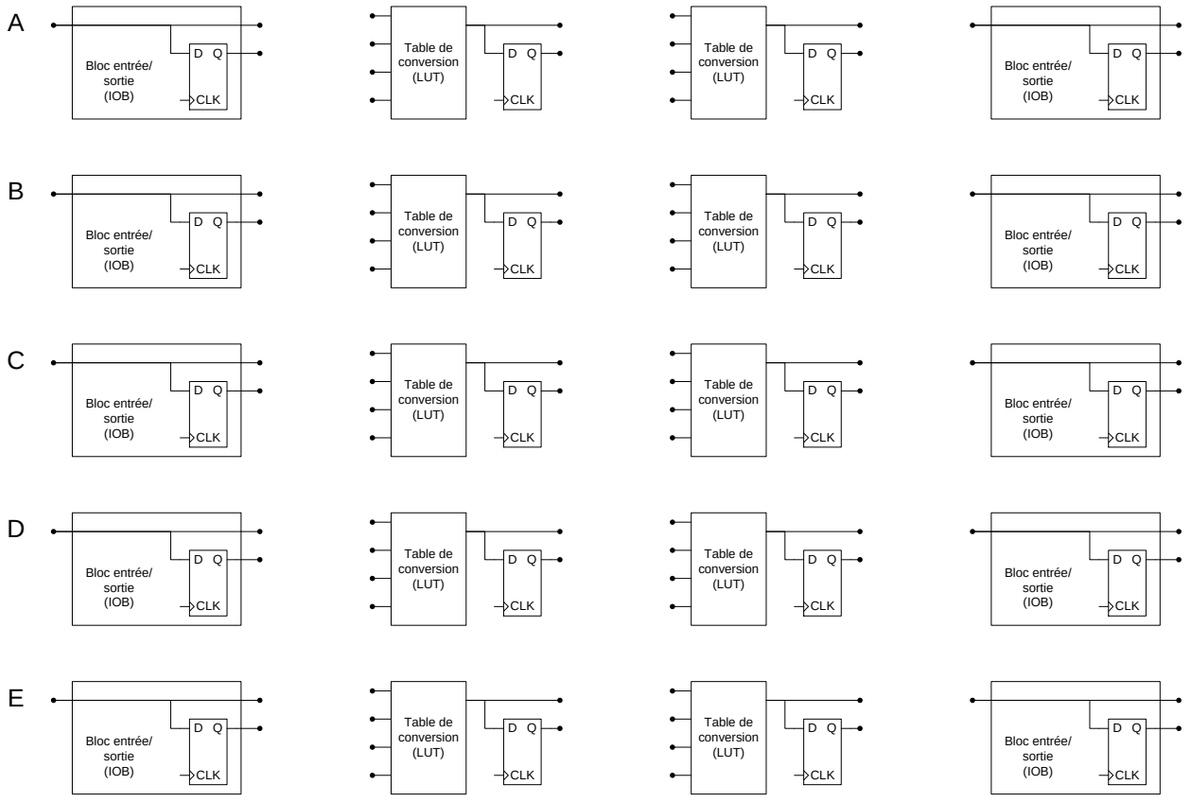
Donnez le diagramme d'une machine à états pour le système de contrôle. Utilisez une machine de Moore. Indice : il est possible de trouver une solution avec quatre états ou moins.

Donnez le code VHDL pour la machine à états. Complétez le code suivant.

```
library IEEE;
use IEEE.std_logic_1164.all;

entity sas is
    port (
        reset, CLK : in STD_LOGIC;
        porte_1_fermee : in std_logic;
        porte_2_fermee : in std_logic;
        bouton_1 : in std_logic;
        bouton_2 : in std_logic;
        barrer_1 : out std_logic;
        barrer_2 : out std_logic;
        alarme_out : out std_logic
    );
end sas;

architecture arch of sas is
-- votre code ici
begin
-- votre code ici
end arch;
```

7. Considérez le code VHDL et le modèle de FPGA suivants. Montrez, sur le modèle du FPGA, un résultat possible de la synthèse et de l'implémentation de ce code. Indiquez directement sur le dessin où chaque signal et port de sortie se situe ainsi que les interconnexions entre les blocs. Les interconnexions peuvent contourner les blocs. Indiquez quand une bascule doit être utilisée. Indiquez par une équation la fonction logique réalisée par chaque LUT que vous utilisez. Respectez l'assignation donnée pour les ports d'entrée.

```
library ieee;
use ieee.std_logic_1164.all;

entity module10 is
   port (
      clk, A, B, C, D, E: in std_logic;
      X, Y, Z: out std_logic
   );
end module10;

architecture arch of module10 is

signal F, G, H : std_logic;

begin

   X <= not(A and B and E);
   Y <= G xor H;

   process(clk) is
   begin
      if rising_edge(CLK) then
         F <= A and B and C and D;
         G <= F xor E;
         H <= B or C or D;
      end if;
   end process;

   process(A, B, C)
   begin
      if A = '1' then
         Z <= B or C;
      else
         Z <= B and C;
      end if;
   end process;

end arch;
```



Solutions

1. Combinatoires: S7; Mealy: S1, S2, S3; Moore: S0, S4, S5, S6

2. Réponse

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity registre2 is
   generic (
      W : integer := 8
   );
   port(
      reset, CLK, charge : in STD_LOGIC;
      D : in STD_LOGIC_VECTOR(W - 1 downto 0);
      Q : out STD_LOGIC_VECTOR(W - 1 downto 0)
   );
end registre2;

architecture arch of registre2 is
begin

   assert W = 8
   report "ne fonctionne que pour W = 8"
   severity failure;
```

```
   process (CLK, reset)
   begin
       if falling_edge(CLK) then
          if reset = '0' then
             Q <= X"F4";
          elsif charge = '1' then
             Q <= D;
          end if;
       end if;
   end process;

end arch;
```

3. Réponses.
   a. Pour mystere2, le code VHDL modélise un multiplexeur à 4 entrées avec comme signal de contrôle s. Les entrées 2 et 3 du multiplexeur sont reliées au signal c.
   b. Pour mystere1, le code VHDL modélise aussi un multiplexeur à 4 entrées, mais dont la sortie est reliée à un loquet. Le signal s contrôle le multiplexeur. L'entrée 3 (s = « 11 ») du multiplexeur n'est jamais utilisée. Le code indique un mode mémoire quand s = 11 – la sortie o ne doit pas changer. Donc le signal de contrôle G du loquet doit être mené par la fonction G = not (s(1) and s(0)), c'est-à-dire une porte NON-ET.

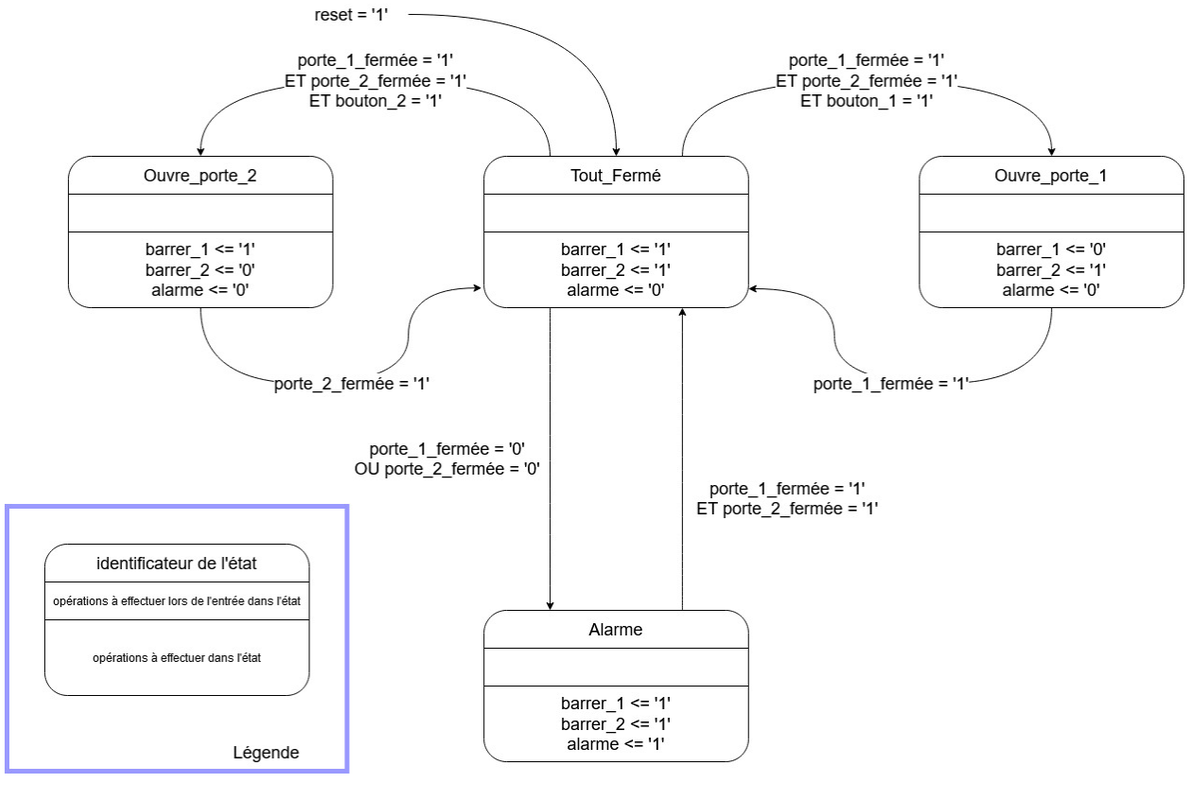
4. Réponses
   a. archiBAD1 : La condition de réinitialisation est dans la même condition que la transition d'horloge. Le patron de code n'est pas conforme.
   b. archiBAD2 : 1. On combine une condition de transition d'horloge et de réinitialisation. 2. Il y a une condition sur une transition d'horloge (CLK'event) sans indiquer de direction. 3. Il y a deux conditions de transition d'horloge. Le patron de code n'est pas conforme.
   c. archiBAD3 : La condition de réinitialisation est dans la même condition que la transition d'horloge. Le patron de code n'est pas conforme.
   d. archiBAD4 : Il n'y a pas de conditions d'indiquées sur l'horloge dans le processus. En simulation, on aurait le comportement d'une bascule sensible aux deux transitions, parce que le signal CLK est dans la liste de sensitivité. En synthèse, on aurait le fil D relié au fil Q, sans élément à mémoire.

5. Solution

```
architecture arch of machineAEtats is
type type_etat is (S1, S2, S3, S4);
signal etat : type_etat := S1;
begin

   process(CLK, reset) is
   begin
       if (reset = '0') then
          etat <= S1;
       elsif (rising_edge(CLK)) then
          case etat is
             when S1 =>
                if x = "00" then
                   etat <= S3;
                elsif x = "01" then
                   etat <= S2;
                end if;
             when S2 | S3 =>
                if x = "10" then
                   etat <= S4;
                elsif x = "11" then
                   etat <= S1;
                end if;
             when S4 =>
                etat <= S1;
          end case;
       end if;
   end process;
```

```
process(etat)
   begin
      case etat is
         when S1 | S4 =>
            sortie <= '1';
         when S2 | S3 =>
            sortie <= '0';
      end case;
   end process;

end arch;
```

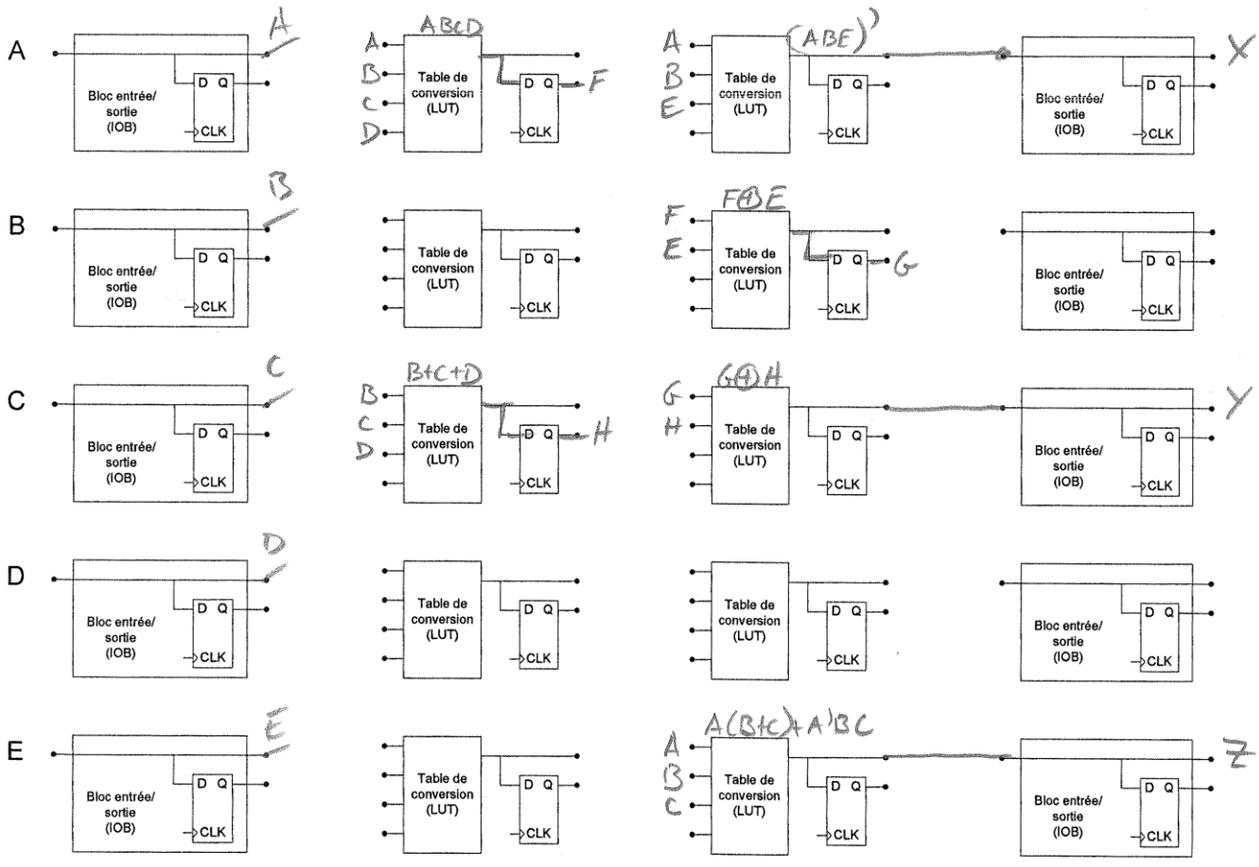
6. Solution




```
library IEEE;
use IEEE.std_logic_1164.all;

entity sas is
  port (
    reset, CLK : in STD_LOGIC;
    porte_1_fermee : in std_logic;
    porte_2_fermee : in std_logic;
    bouton_1 : in std_logic;
    bouton_2 : in std_logic;
    barrer_1 : out std_logic;
    barrer_2 : out std_logic;
    alarme_out : out std_logic
  );
end sas;
```

```
architecture arch of sas is
type type_etat is (tout_ferme, ouvre_porte_1, ouvre_porte_2, alarme);
signal etat : type_etat := tout_ferme;
begin

  process(CLK, reset) is
  begin
    if (reset = '1') then
      etat <= tout_ferme;
    elsif (rising_edge(CLK)) then
      case etat is
        when tout_ferme =>
          if porte_1_fermee = '0' or porte_2_fermee = '0' then
            etat <= alarme;
          elsif porte_1_fermee = '1' and porte_2_fermee = '1' and bouton_1 = '1' then
            etat <= ouvre_porte_1;
          elsif porte_1_fermee = '1' and porte_2_fermee = '1' and bouton_2 = '1' then
            etat <= ouvre_porte_2;
          end if;
        when ouvre_porte_1 =>
          if porte_1_fermee = '1' then
            etat <= tout_ferme;
          end if;
        when ouvre_porte_2 =>
          if porte_2_fermee = '1' then
            etat <= tout_ferme;
          end if;
        when alarme =>
          if porte_1_fermee = '1' or porte_2_fermee = '1' then
            etat <= tout_ferme;
          end if;
      end case;
    end if;
  end process;

  process(etat)
  begin
    case etat is
      when tout_ferme =>
        barrer_1 <= '1';
        barrer_2 <= '1';
        alarme_out <= '0';
      when ouvre_porte_1 =>
        barrer_1 <= '0';
        barrer_2 <= '1';
        alarme_out <= '0';
      when ouvre_porte_2 =>
        barrer_1 <= '1';
        barrer_2 <= '0';
        alarme_out <= '0';
      when alarme =>
        barrer_1 <= '1';
        barrer_2 <= '1';
        alarme_out <= '1';
    end case;
  end process;

end arch;
```

7. Solution


A
B
C
D
E
Bloc entrée/ sortie (IOB)
D Q
CLK
Table de conversion (LUT)
ABCD
F
(ABE)'
X
FOE
G
B+C+D
H
GOH
Y
A(B+C)+A'BC
Z