TP4 VLSI :

Conception et implementation d'une application en VHDL

Projet :

Commande d'un Ascenseur

Réalisé par :

Rym Bouabid
Kais Ben Yahia
Yesmine Ben Ameur

IIA4 G12

ANNÉE UNIVERSITAIRE 2020-2021

I. Le cahier des charges :

Pour une première version finale l'ascenseur circule entre 4 étages.
La montée et la descente de l'ascenseur s'effectue respectivement par activation des actionneurs MH et MB. Notre ascenseur dispose d'une alarme **ALARME** initialement au repos et qui est déclenché dès qu'en appui sur le bouton **A**. Le bouton **stopalarme** est responsable de son arrêt.

- **Initialement l'ascenseur est en état de repos à l'étage j: (j<i)**

Cas 1:

1. Un utilisateur à l'étage j fait appel à l'ascenseur en appuyant sur le bouton **APPEL**[j]
2. La porte de l'ascenseur s'ouvre et reste ouverte pendant une période T:

- Si **BE**[j] est actionné avant écoulement de T alors la porte reste ouverte pendant une nouvelle période T
- Si **BE**[i] est actionné la porte se ferme après écoulement de T et l'ascenseur monte à l'étage i.
- Sinon : une fois T est écoulée la porte se ferme et l'ascenseur reste à l'étage j.

Cas2 :

1. Un utilisateur est à l'étage i fait appel à l'ascenseur en appuyant sur le bouton **APPEL**[i]
2. L'ascenseur monte vers l'étage i.
3. Une fois arrivé, la porte de l'ascenseur s'ouvre et reste ouverte pendant une période T:

- Si **BE**[i] est actionné avant écoulement de T alors la porte reste ouverte pendant une nouvelle période T
- Si **BE**[n] est actionné la porte se ferme après écoulement de T et l'ascenseur se dirige vers l'étage n.
- Sinon : une fois T est écoulée la porte se ferme et l'ascenseur reste à l'étage i.

- **Si l'ascenseur est en état de repos l'étage j: (i<j)**

Cas 1:

1. Un utilisateur à l'étage j fait appel à l'ascenseur en appuyant sur le bouton **APPEL**[j]
2. La porte de l'ascenseur s'ouvre et reste ouverte pendant une période T:

- Si **BE**[j] est actionné avant écoulement de T alors la porte reste ouverte pendant une nouvelle période T
- Si **BE**[i] est actionné la porte se ferme après écoulement de T et l'ascenseur descend à l'étage i.
- Sinon : une fois T est écoulée la porte se ferme et l'ascenseur reste à l'étage j.

Cas2 :

1. Un utilisateur est à l'étage i fait appel à l'ascenseur en appuyant sur le bouton **APPEL**[i]
2. L'ascenseur descend à l'étage j.
3. Une fois arrivé, la porte de l'ascenseur s'ouvre et reste ouverte pendant une période T:

- Si **BE**[i] est actionné avant écoulement de T alors la porte reste ouverte pendant une nouvelle période T

- Si **BE**[n] est actionné la porte se ferme après écoulement de T et l'ascenseur se dirige vers l'étage n.
- Sinon : une fois T est écoulée la porte se ferme et l'ascenseur reste à l'étage i.

- **Initialement l'ascenseur est en état de repos à l'étage j et un utilisateur appui sur BE[i]:**

  Cas 1: **(j<i)**

  L'ascenseur monte vers l'étage i.

  Cas2 : **(i<j)**

  L'ascenseur descend vers l'étage j

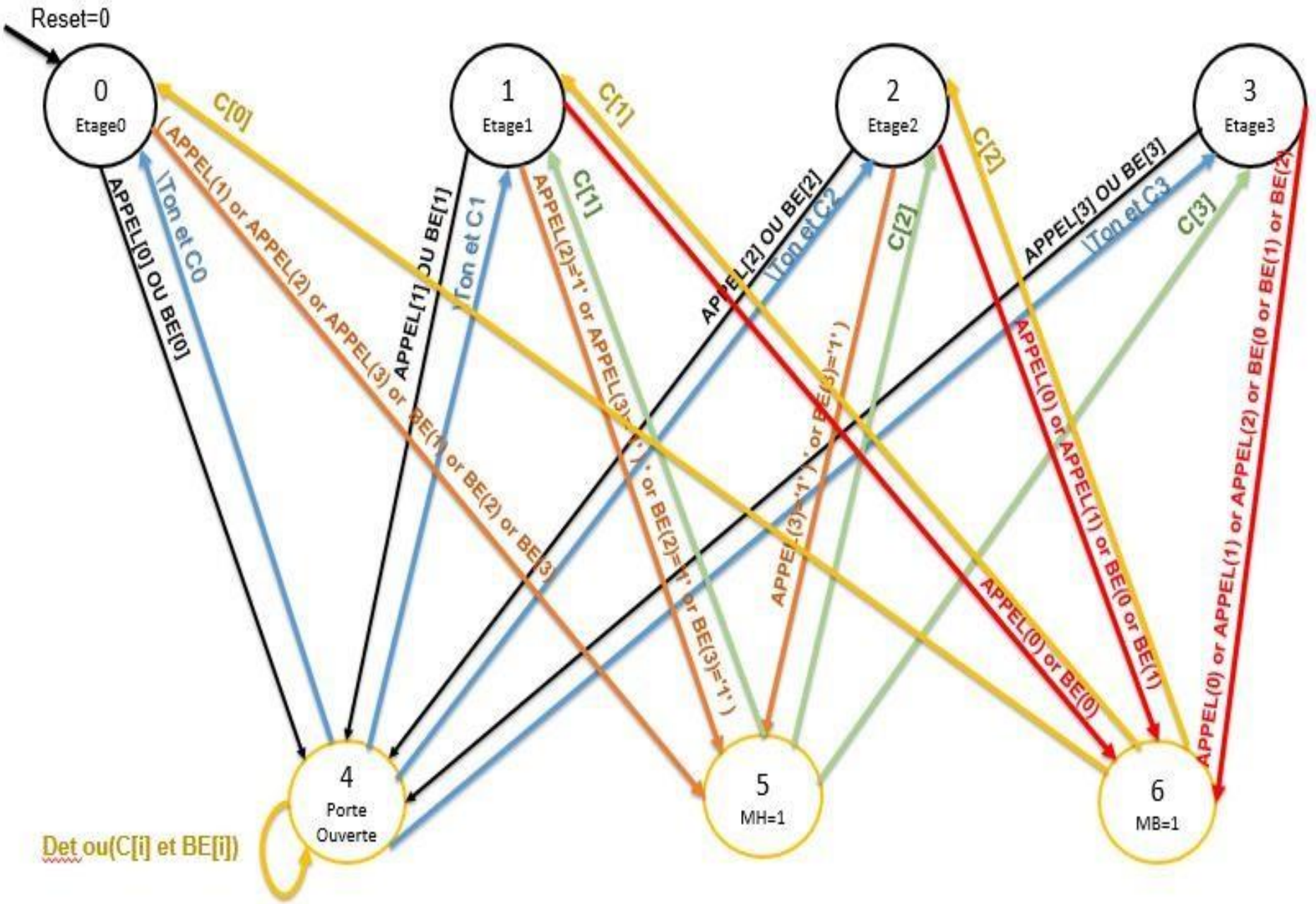
Soit :

Capteur	Description
APPEL0	Bouton poussoir d'appel de l'ascenseur au RC
APPEL1	Bouton poussoir d'appel de l'ascenseur au 1er étage
APPEL1	Bouton poussoir d'appel de l'ascenseur au 2eme étage
APPEL1	Bouton poussoir d'appel de l'ascenseur au 3eme étage
BE0	Bouton poussoir: Demande d'aller au RC
BE1	Bouton poussoir: Demande d'aller au 1er étage
BE2	Bouton poussoir: Demande d'aller au 2eme étage
BE3	Bouton poussoir: Demande d'aller au 3eme étage
C0	Capteur position: Ascenseur au RC
C1	Capteur position: Ascenseur au 1er étage
C2	Capteur position: Ascenseur au 2eme étage
C3	Capteur position: Ascenseur au 3eme étage
DET	Capteur pour détecter la présence d'un objet ou d'une personne

Actionneur	Description
P	Si P=1 alors la porte s'ouvre / Si P=0 alors la porte se ferme
MH	Moteur vers le haut pour faire monter vers l'ascenseur
MB	Moteur vers le bas pour faire descendre vers l'ascenseur

Temporisateur	Description
Ton	De type Ton et de durée= période pendant laquelle la porte est ouverte

II. le modèle de spécification (l'automate) : FS



III. Code commenté :

1 . Code VHDL du module de la machine d'état

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity Ascenseur is
        -- generic (clockFrequencyHZ : integer );
    Port ( reset : in  STD_LOGIC   ;
           CLK : in  STD_LOGIC     ;
                        APPEL : IN  std_logic_vector (3 downto 0);
-- bouton d'appel de l'ascenseur d'une personne à l'étage i
                        BE: IN  std_logic_vector (3 downto 0); -- choix de
l'étage à l'interieur de l'ascenseur
                        C: IN  std_logic_vector (3 downto 0); -- capteur
de l'ascenseur
           DET : in  STD_LOGIC     ;
           P : out  STD_LOGIC     ;
           MH : out  STD_LOGIC    ;
           MB : out STD_LOGIC  ;
                        a, stopalarme : in STD_LOGIC ;
                        ALARME : out STD_LOGIC );
end Ascenseur;

architecture Behavioral of Ascenseur is

constant clockFrequencyHZ : integer:=50;
type Etat is (Etat0 ,Etat1 ,Etat2 ,Etat3 ,Etat4_PorteOuverte,
Etat5_MonteeAscenseur,Etat6_DescenteAscenseur);
signal Etati : Etat := Etat0; -- état de repos de l'ascenseur à l'étage 0
et portes fermées
signal counter : integer range 0 to clockFrequencyHZ*1000; --50ms
signal h : STD_LOGIC := '0';
signal d : STD_LOGIC := '0' ;
begin

process(CLK, reset) is
variable n: integer :=0 ;
begin
 if (reset = '0') then
       etati <= Etat0;
       MH <= '0' ; MB <= '0' ; P<='0'; h<='0'; d<='0';
       ALARME<= '0';
       counter <= 0;
elsif (rising_edge(CLK)) then
       case etati is
              when Etat0 =>
                     -- appel de l'étage 0 à partir de l'étage 0
                     if ( APPEL(0)='1' or BE(0)= '1' ) then
                            counter <= 0;
                            etati <= Etat4_PorteOuverte;
```

```
                        -- appel d'un etage supérieur
                        elsif ( ( APPEL(1)='1' or APPEL(2)='1' or
APPEL(3)='1' ) or ( BE(1)='1' or BE(2)='1' or BE(3)='1' ) ) then
                        etati <= Etat5_MonteeAscenseur;
                        end if;

                when Etat1 =>
                        -- appel de l'étage 1 à partir de l'étage 1
                        if ( APPEL(1)='1' or BE(1)= '1' ) then
                                counter <= 0;
                                etati <= Etat4_PorteOuverte;

                        elsif ( BE(2)='1' or BE(3)='1' ) and h='1' then
etati <= Etat5_MonteeAscenseur;

                        elsif (BE(0)= '1' and d='1') then  etati <=
Etat6_DescenteAscenseur ;

                        elsif ( BE(2)='1' or BE(3)='1' ) and APPEL="0010"
then etati <= Etat5_MonteeAscenseur;

                        elsif (BE(0)= '1' and APPEL="0010" ) then  etati <=
Etat6_DescenteAscenseur ;

                        -- appel d'un etage supérieur
                        elsif (( APPEL(2)='1' or APPEL(3)='1' )or (
BE(2)='1' or BE(3)='1' ) )   then etati <= Etat5_MonteeAscenseur;

                        -- appel d'un etage inférieur
                        elsif (  (APPEL(0)='1') or BE(0)= '1' ) then  etati
<= Etat6_DescenteAscenseur ;
                        end if;


                when Etat2 =>
                        -- appel de l'étage 2 à partir de l'étage 2
                        if ( APPEL(2)='1' or BE(2)= '1' ) then
                        counter <= 0;
                        etati <= Etat4_PorteOuverte;

                        elsif ( ( BE(3)='1' ) and h='1') then etati <=
Etat5_MonteeAscenseur;

                        elsif ( ( BE(0)= '1' or BE(1)='1' ) and d='1' ) then
etati <= Etat6_DescenteAscenseur ;

                        elsif ( ( BE(3)='1' )and APPEL="0100" ) then etati
<= Etat5_MonteeAscenseur;

                        elsif ( ( BE(0)= '1' or BE(1)='1' ) and APPEL="0100"
) then  etati <= Etat6_DescenteAscenseur ;  -- appel d'un etage supérieur
```

```
                        elsif ( APPEL(3)='1' or ( BE(3)='1' ) )  then etati
<= Etat5_MonteeAscenseur;

                        -- appel d'un etage inférieur
                        elsif ( APPEL(0)='1' or APPEL(1)='1' or ( BE(0)= '1'
or BE(1)='1' ) )  then  etati <= Etat6_DescenteAscenseur ;

                        end if;

                when Etat3 =>
                        -- appel de l'étage 3 à partir de l'étage 3
                        if ( APPEL(3)='1' or BE(3)= '1' ) then
                        counter <= 0;
                        etati <= Etat4_PorteOuverte;

                        -- appel d'un etage inférieur
                        elsif ( ( APPEL(0)='1' or APPEL(1)='1' or
APPEL(2)='1' ) or ( BE(0)='1' or BE(1)='1' or BE(2)='1' ) ) then  etati <=
Etat6_DescenteAscenseur ;

                        end if;

                when Etat4_PorteOuverte =>
                        counter <= counter + 1;

                        if ( ( BE(0)='1' AND C(0)='1' ) or ( BE(1)='1' AND
C(1)='1') or( BE(2)='1' AND C(2)='1' ) or ( BE(3)='1' AND C(3)='1'))then
                        counter<=0;
                        etati <= Etat4_PorteOuverte;

                        elsif det = '1' then
                        counter<=0 ;
                        etati <= Etat4_PorteOuverte ;

                        elsif ( (counter= clockFrequencyHZ*1000-1 ) and
C(0)='1' ) then
                        etati <= Etat0 ;

                        elsif ( (counter= clockFrequencyHZ*1000-1 ) and
C(1)='1' ) then
                        etati <= Etat1 ;

                        elsif ( (counter= clockFrequencyHZ*1000-1 ) and
C(2)='1' ) then
                        etati <= Etat2 ;

                        elsif ( (counter= clockFrequencyHZ*1000-1 ) and
C(3)='1' ) then
                        etati <= Etat3;

                        end if;

                when Etat5_MonteeAscenseur =>
                        if ( C(1) = '1' ) then etati <= Etat1;
```

```
                        elsif ( C(2) = '1' ) then etati <= Etat2;
                        elsif ( C(3) = '1' ) then etati <= Etat3;
                        end if;

                when Etat6_DescenteAscenseur =>
                        if ( C(0) = '1' ) then etati <= Etat0;
                        elsif ( C(1) = '1' ) then etati <= Etat1;
                        elsif ( C(2) = '1' ) then etati <= Etat2;
                        end if;
        end case;

 end if;


 case etati is
                when Etat0 =>
                        MH <= '0' ; MB <= '0' ; P<='0';
                when Etat1 =>
                        MH <= '0' ; MB <= '0' ; P<='0';
                when Etat2 =>
                        MH <= '0' ; MB <= '0' ; P<='0';
                when Etat3 =>
                        MH <= '0' ; MB <= '0' ; P<='0';
                when Etat4_PorteOuverte =>
                        MH <= '0' ; MB <= '0' ; P<='1';
                when Etat5_MonteeAscenseur =>
                        MH <= '1' ; MB <= '0' ; P<='0';
                        h<='1'; d<='0';
                when Etat6_DescenteAscenseur =>
                        MH <= '0' ; MB <= '1' ; P<='0';
                        h<='0'; d<='1';
end case;

-- Fonctionnement de l'Alarme de secours :
        -- remarque : a est initialisé au test bench à '0'
        if ( a = '0')  AND ( n = 0 ) then alarme <= '0'; -- si le bouton
poussoir (a), simulant l'alarme, n'est pas activé: l'alarme n'est pas
activée
        elsif a = '1' then    -- une fois (a) est activé:
                alarme <= '1' ;       --     l'alarme s'active
                n:=1 ;
        end if;
        if stopalarme = '1' then
                alarme <= '0';        -- si le bouton poussoir stopalarme
s'active alors l'alarme est arrêtée
                n:=0 ;
        end if;

end process;

end Behavioral;
```

2 . Code test bench :

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY TesAscenseur IS
END TesAscenseur;

ARCHITECTURE behavior OF TesAscenseur IS

    COMPONENT Ascenseur
    PORT(
         reset : INstd_logic;
         CLK : IN  std_logic;
         APPEL : IN  std_logic_vector (3 downto 0); -- bouton d'appel de
l'ascenseur d'une personne à l'étage i
                      BE: IN  std_logic_vector (3 downto 0); -- choix de
l'étage à l'interieur de l'ascenseur
                      C: IN  std_logic_vector (3 downto 0); -- capteur de
l'ascenseur
         DET : IN  std_logic;
                      P : OUT  std_logic;
         MH : OUT  std_logic;
         MB : OUTstd_logic;
         a : IN  std_logic;
         stopalarme : IN  std_logic;
         ALARME : OUT  std_logic );
    END COMPONENT;


   --Inputs
   signal reset : std_logic := '1';
   signal CLK : std_logic := '0';
       signal APPEL : std_logic_vector (3 downto 0):= "0000";
   signal BE : std_logic_vector (3 downto 0):= "0000";
       signal C : std_logic_vector (3 downto 0):= "0001"; --capteur etage
0 activé c(0)='1'
   signal DET : std_logic := '0';
   signal a : std_logic := '0';
   signal stopalarme : std_logic := '0';

       --Outputs
   signal P : std_logic;
   signal MH : std_logic;
   signal MB : std_logic;
   signal ALARME : std_logic ;

   -- Clock period definitions
   constant CLK_period : time := 1000ns;

BEGIN
```

```
        -- Instantiate the Unit Under Test (UUT)
   uut: Ascenseur PORT MAP (
          reset => reset,
          CLK => CLK,
                      APPEL => APPEL,
                      BE => BE,
                      C => C,
          DET => DET,
          P => P,
          MH => MH,
          MB => MB,
          a => a,
          stopalarme => stopalarme,
          ALARME => ALARME
        );

   -- Clock process definitions
   CLK_process :process
   begin
               CLK <= '0';
               wait for CLK_period/2;
               CLK <= '1';
               wait for CLK_period/2;
   end process;


   -- Stimulus process
   stim_proc: process
   begin

----re-initialiser
--      wait for 100 ms ;
--      reset<='0';
--      reset<='1' after 50 ms;


--Scenario 1 : ____________________________________________

-- demande d'appel de l'etage 3
   wait for 20 ms;
        APPEL <="1000";

-- l'ascenseur a quitté l'étage 0 donc C(0) devient nulle
   wait for 20 ms;
        C <="0000";
-- l'ascenseur est arrivé à l'etage 1
        wait for 50 ms;
        C <="0010";
        wait for 1000 ns;
        C <="0000";

-- l'ascenseur est arrivé à l'etage 2
        wait for 50 ms;
        C <="0100";
```

```
        wait for 1000 ns;
        C <="0000";

-- l'ascenseur est arrivé à l'etage 3
   wait for 50 ms;
        C <="1000";

--mission accomplie donc desactivation de appel(3)
   wait for 50 ms;
        APPEL(3)<='0';

--Scenario 2 : _____________________________________________
-- APPEL DE l'etage 2 et 1
        wait for 100 ms;
        appel(2)<='1';
        appel(1)<='1';

        -- l'ascenseur a quitté l'étage 3 donc C(3) devient nulle
        wait for 10 ms;
        C <="0000";

        wait for 90 ms;
        C <="0100";

        --une personne entre ( le détecteur de présence (capteur) det passe
à '1'
        wait for 2 ms ;
        det<='1';

        -- la personne n'est plus devant le detecteur donc det revient à
'0'
        wait for 2 ms ;
        det<='0';

        -- demande de l'étage 0 par la personne dans l'ascenseur
        wait for 5 ms;
        BE(0) <= '1';

--mission accomplie donc desactivation de appel(2)
        wait for 10 ms;
        appel(2)<='0';

        -- l'ascenseur a quitté l'étage 2 donc C(2) devient nulle
        wait for 35 ms;
        C <="0000";

-- l'ascenseur est arrivé à l'etage 1
        wait for 100 ms;
        C<="0010";

--mission accomplie donc desactivation de APPEL(1)
        wait for 50 ms;
        appel(1)<='0';
```

```
        -- l'ascenseur a quitté l'étage 1 donc C(1) devient nulle
        wait for 3000 ns;
        C <="0000";


-- l'ascenseur est arrivé à l'etage 0
        wait for 100ms;
        C <="0001";

        wait for 2000 ns;
        BE(0)<='0';

        --les personnes sortent de l'ascenseur
        wait for 2 ms ;
        det<='1';

        wait for 5 ms ;
        det<='0';

--Scenario 3 : _____________________________________________
        wait for 300 ms;
        appel(2)<='1';
        appel(1)<='1';

        -- l'ascenseur a quitté l'étage 0 donc C(0) devient nulle
   wait for 1000 ns;
        C <="0000";

        wait for 100 ms;
        C <="0010";  -- arrivée à l'étage1

        --une personne entre ( le détecteur de présence (capteur) det passe
à '1'
        wait for 2 ms ;
        det<='1';

        -- la personne n'est plus devant le detecteur donc det revient à
'0'
        wait for 2 ms ;
        det<='0';

        wait for 10 ms;--choix de l'etage 0 par la personne
        BE(0) <= '1';

        wait for 40 ms;--mission accomplie donc desactivation de appel(1)
        appel(1)<='0';

        -- l'ascenseur a quitté l'étage 1 donc C(1) devient nulle
  -- wait for 2000 ns;
        C <="0000";

        wait for 100 ms;
        C <="0100";  -- arrivée à l'étage2
```

```
        wait for 50 ms;--mission accomplie donc desactivation de appel(2)
        appel(2)<='0';

        -- l'ascenseur a quitté l'étage 2 donc C(2) devient nulle
    wait for 2000 ns;
        C <="0000";

        wait for 100 ms;
        C <="0010";

        wait for 1000 ns;
        C <="0000";  -- l'ascenseur a quitté l'étage 1 donc C(1) devient
nulle

        -- quelqu'un appuie sur le bouton poussoir (a) simulant l'alarme
        wait for 50 ms ;
        a<= '1' ;
        wait for 2000 ns;
        a<= '0' ;

        wait for 50 ms;
        C <="0001";  -- arrivée à l'étage0
        stopalarme<='1';

        wait for 2000 ns;
        stopalarme<= '0' ;

        wait for 2 ms ;
        BE(0)<= '0';  --mission accomplie donc desactivation de BE(0)
        det<='1'; --les personnes sortent de l'ascenseur

        wait for 5 ms ;
        det<='0'; -- les personnes ne sont plus devant le detecteur donc
det revient à '0'



wait for CLK_period*10;
wait;
end process;
END;
```

VI. Résultats des tests :

Scénario1:

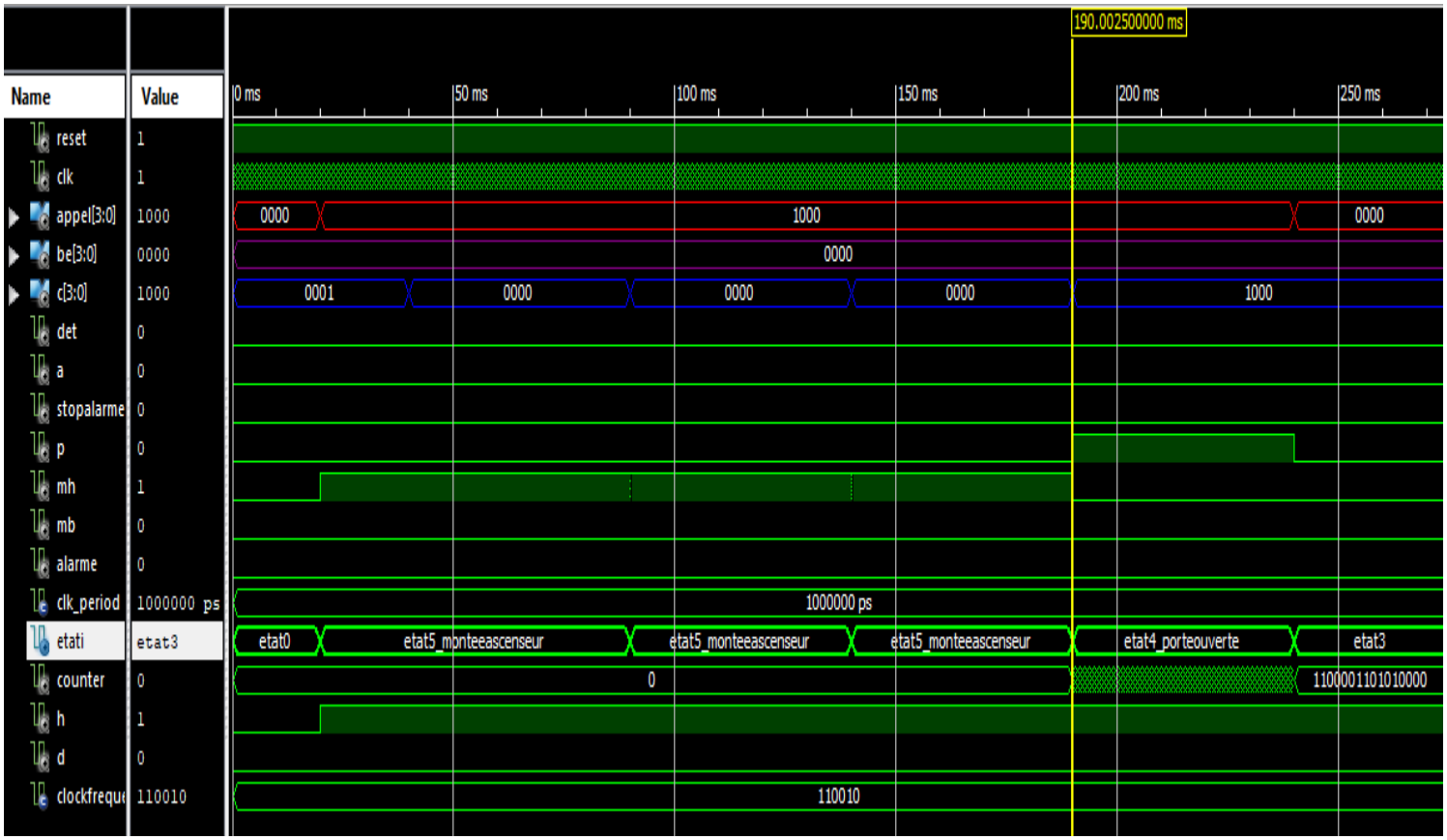
En repos l'ascenseur est à l'étage 0 (le vecteur C (input) indiquant à quel étage se trouve l'ascenseur prend la valeur "0001" avec porte fermée "**etat0**".
Suite à une demande extérieure de l'étage 3 obtenue par le bit le plus fort du vecteur APPEL "1000" (càd celui qui a demandé l'ascenseur se trouve à l'étage 3):

1) l'ascenseur commence à monter (passage à l'état: "**etat5_monteeascenseur**")
2) jusqu'à arriver à l'étage 1, avec porte fermée (passage à l'état: "**etat1**")
3) pas de demande de l'étage 1 donc l'ascenseur ne s'arrête pas et continue à monter (passage à l'état: "**etat5_monteeascenseur**")
4) jusqu'à arriver à l'étage 2, avec porte fermée (passage à l'état: "**etat2**")
5) pas de demande de l'étage 2 donc l'ascenseur ne s'arrête pas et continue à monter (passage à l'état: "**etat5_monteeascenseur**")
6) jusqu'à arriver à l'étage 3, avec porte fermée (passage à l'état: "**etat3**")
7) le vecteur APPEL indique la demande de l'étage 3 "1000" donc désactivation de APPEL(3) alors l'ascenseur s'arrête et ouvre la porte (passage à l'état: "**Etat4_PorteOuverte**")
8) la porte reste ouverte pendant T=50ms, une fois cette période écoulée (cad counter arrive au nombre de cycles d'horloge passés équivalent à 50ms), la porte se ferme (passage à l'état: "**etat3**").

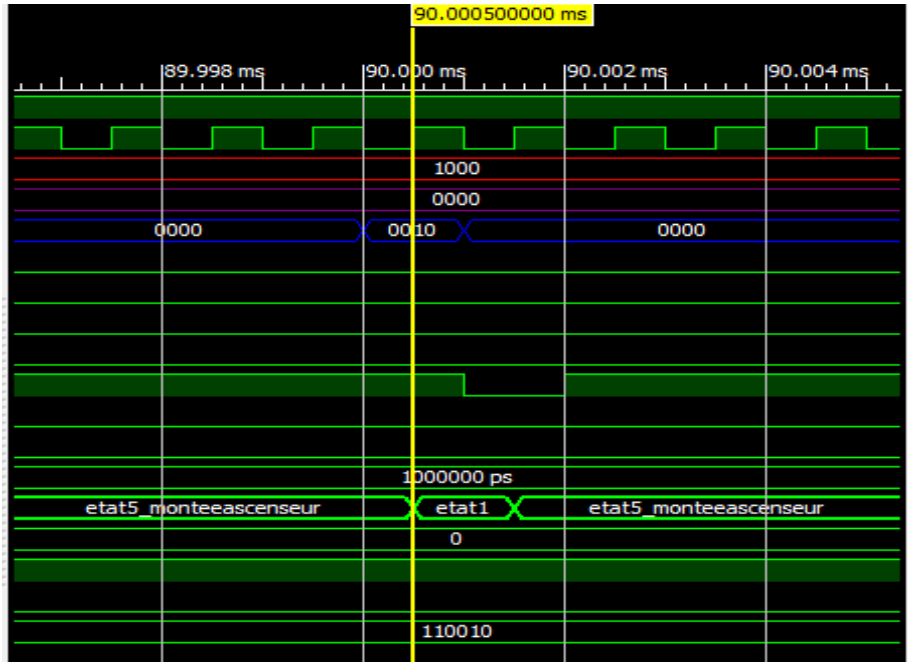
Remarque: personne n'a monté dans l'ascenseur (det le capteur de détection d'un objet reste à '0') et l'ascenseur reste à l'étage 3 avec porte fermée "etat3" jusqu'à nouvelle demande

- **Donc le cycle sera comme suit** :

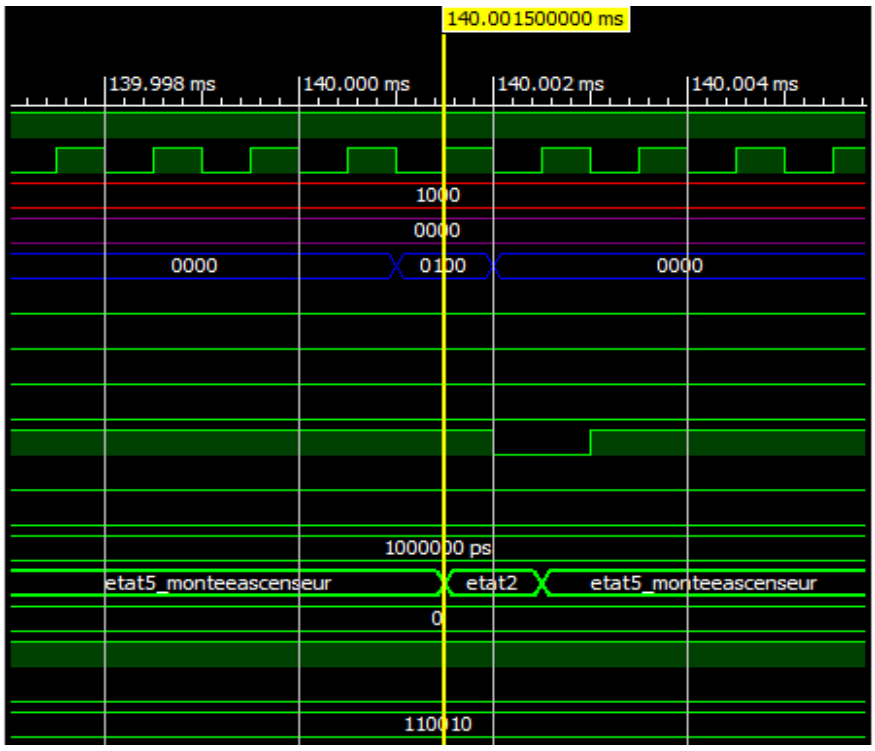
Etat0 → Etat5MonteAscenseur → etat1 → etat5MonteAscenseur → etat2 → etat5MonteAscenseur → etat3 → Etat4PorteOuverte → etat3 (fig : etat3 2.0)



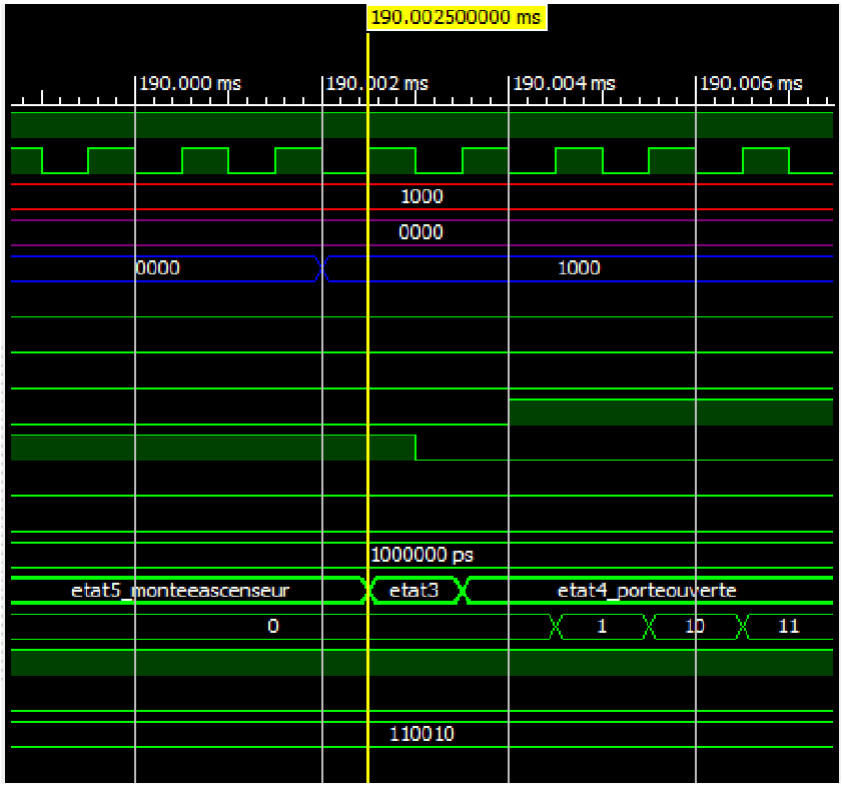
Un zoom est nécessaire pour visualiser les états de petites durées : Etat1:



Etat2:


140.001500000 ms
139.998 ms
140.000 ms
140.002 ms
140.004 ms
1000
0000
0000
0100
0000
1000000 ps
etat5_monteeascenseur
etat2
etat5_monteeascenseur
0
110010

Etat3:


190.002500000 ms
190.000 ms
190.002 ms
190.004 ms
190.006 ms
1000
0000
0000
1000
1000000 ps
etat5_monteeascenseur
etat3
etat4_porteouverte
0
1
10
11
110010

Scénario2:

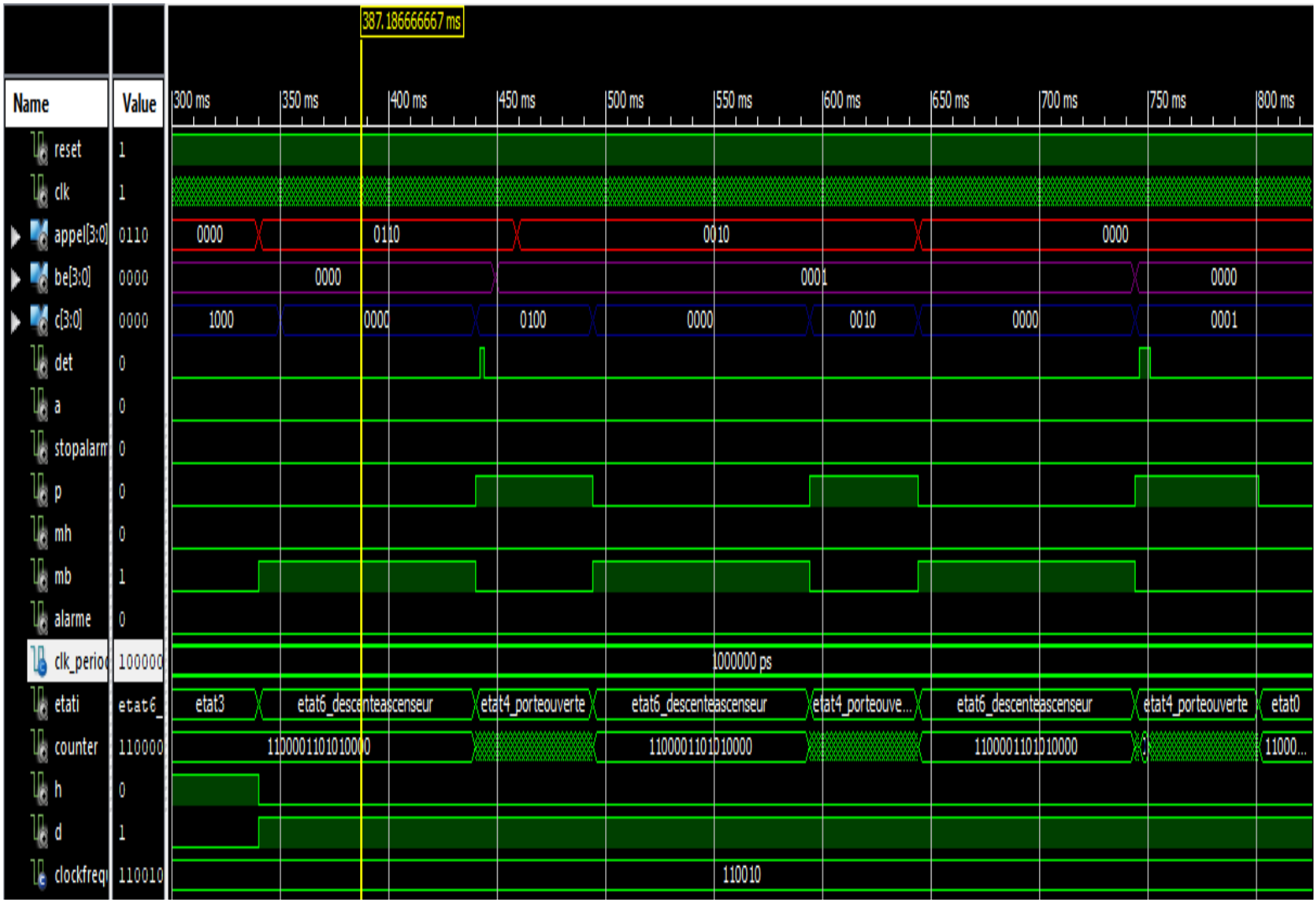
L'ascenseur est à l'étage 3 (C = "1000") avec porte fermée ("**etat3**")
Suite à des demandes extérieures des étages 2 et 1 obtenues par les bits 2 et 1 du vecteur APPEL "0110" (càd ceux qui ont demandé l'ascenseur se trouvent à l'étage 2 et 1) :

1) l'ascenseur commence à descendre (passage à l'état: " **Etat6_DescenteAscenseur**")
2) jusqu'à arriver à l'étage 2 avec porte fermée (passage à l'état: "**etat2**")
3) le vecteur APPEL indique la demande de l'étage 2 APPEL(2)='1' alors l'ascenseur s'arrête et ouvre la porte (passage à l'état: "**Etat4_PorteOuverte**")
4) la porte s'ouvre et une personne entre (le détecteur de présence (capteur) det passe à '1' pendant 2ms puis revient à '0'). Cette personne demande l'étage 0 par le bouton à l'intérieur de l'ascenseur (BE(0) passe à '1'). Une fois la période d'ouverture écoulée la porte se ferme (passage à l'état: "**etat2**")
5) l'ascenseur commence à descendre (passage à l'état: " **Etat6_DescenteAscenseur**")
6) jusqu'à arriver à l'étage 1 avec porte fermée (passage à l'état: "**etat1**")
7) le vecteur APPEL indique la demande de l'étage 1 APPEL(1)='1' alors l'ascenseur s'arrête et ouvre la porte (passage à l'état: "**Etat4_PorteOuverte**")
8) la porte reste ouverte pendant T=50ms sans qu'aucune personne n'entre, une fois cette période écoulée la porte se ferme (passage à l'état: "**etat1**")
9) l'ascenseur commence à descendre (passage à l'état: " **Etat6_DescenteAscenseur**").
10) jusqu'à arriver à l'étage 0 avec porte fermée (passage à l'état: "**etat0**").
11) le vecteur BE indique la demande de l'étage 0 BE(0)='1' alors l'ascenseur s'arrête et ouvre la porte (passage à l'état: "**Etat4_PorteOuverte**")
12) la porte s'ouvre et les personnes sortent de l'ascenseur (le détecteur de présence (capteur) det passe à '1' pendant 5 ms puis revient à '0')

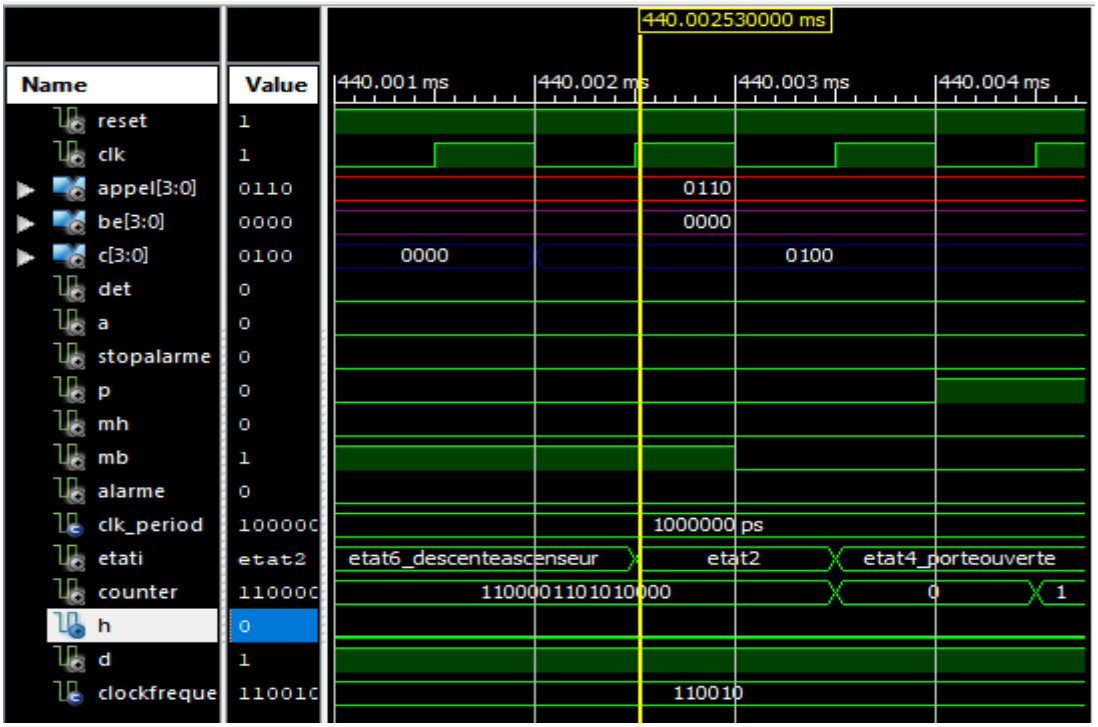
Une fois la période d'ouverture écoulée la porte se ferme (passage à l'état: "**etat0**")

❖ **<u>Donc le cycle sera comme suit</u>** :

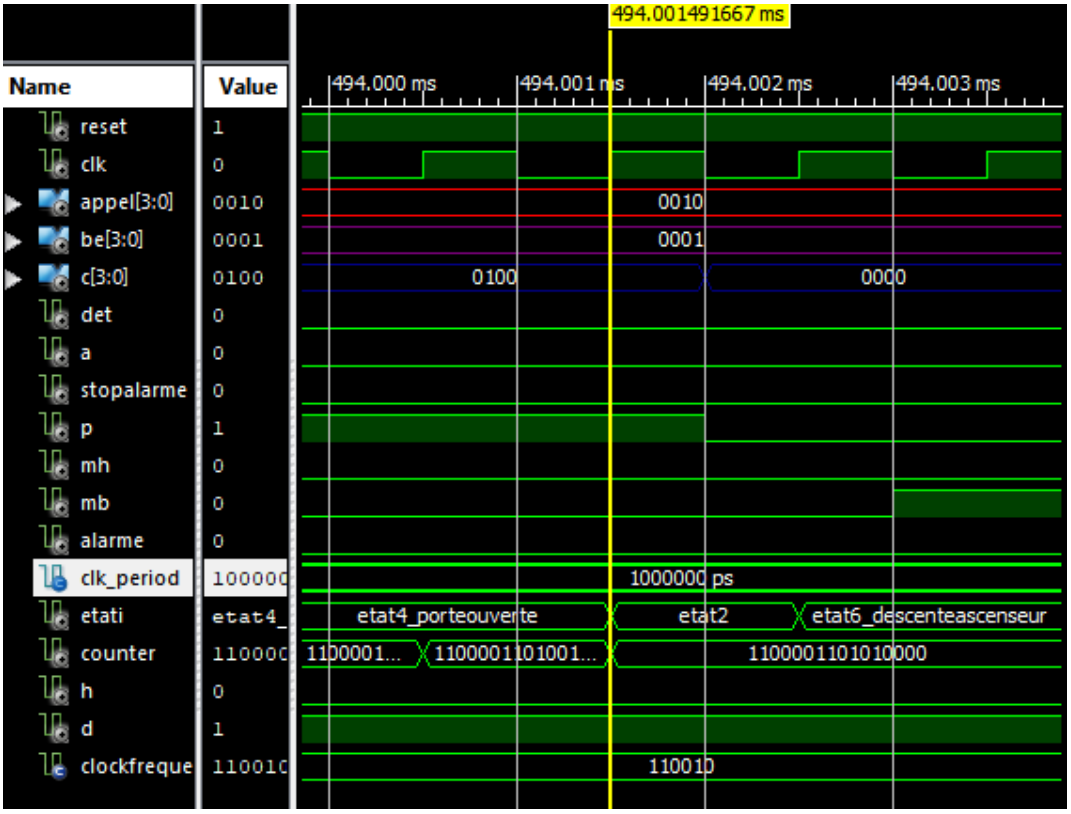
Etat3 → Etat6DescenteAscenseur → etat2 → Etat4PorteOuverte → etat2 (fig : etat2 2.0) → etat6DescenteAscenseur → etat1 → Etat4PorteOuverte → etat1 (fig : etat1 2.0) → etat6DescenteAscenseur → etat0 → Etat4PorteOuverte → etat0 (fig : etat0 2.0)



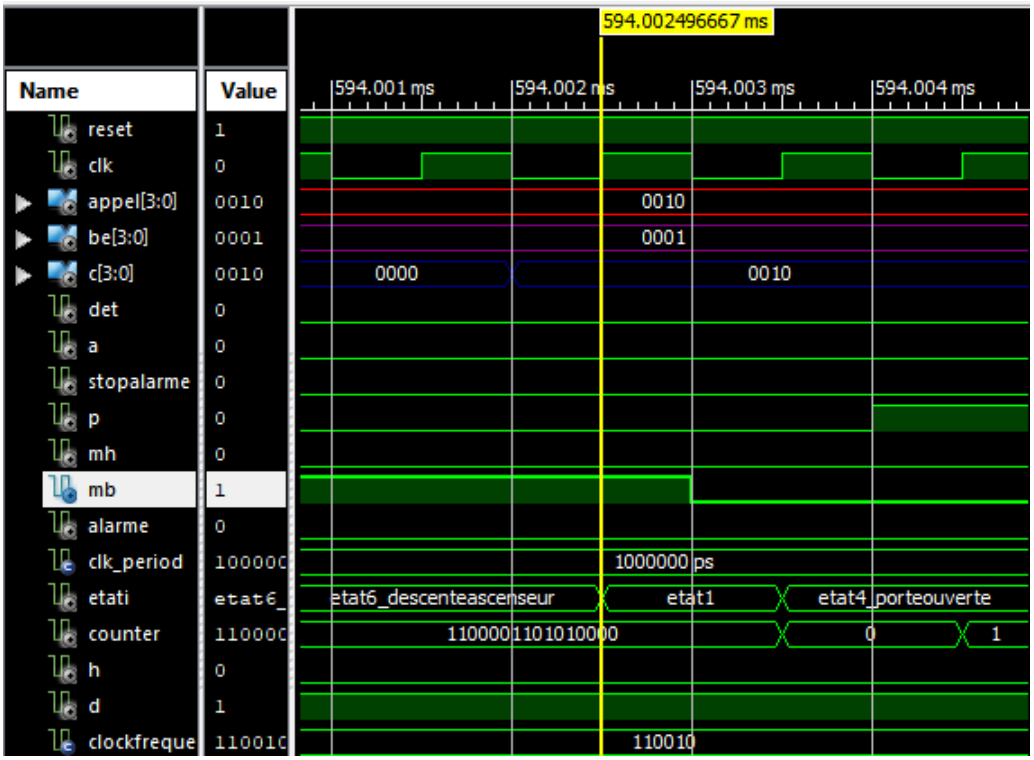
Etat2:



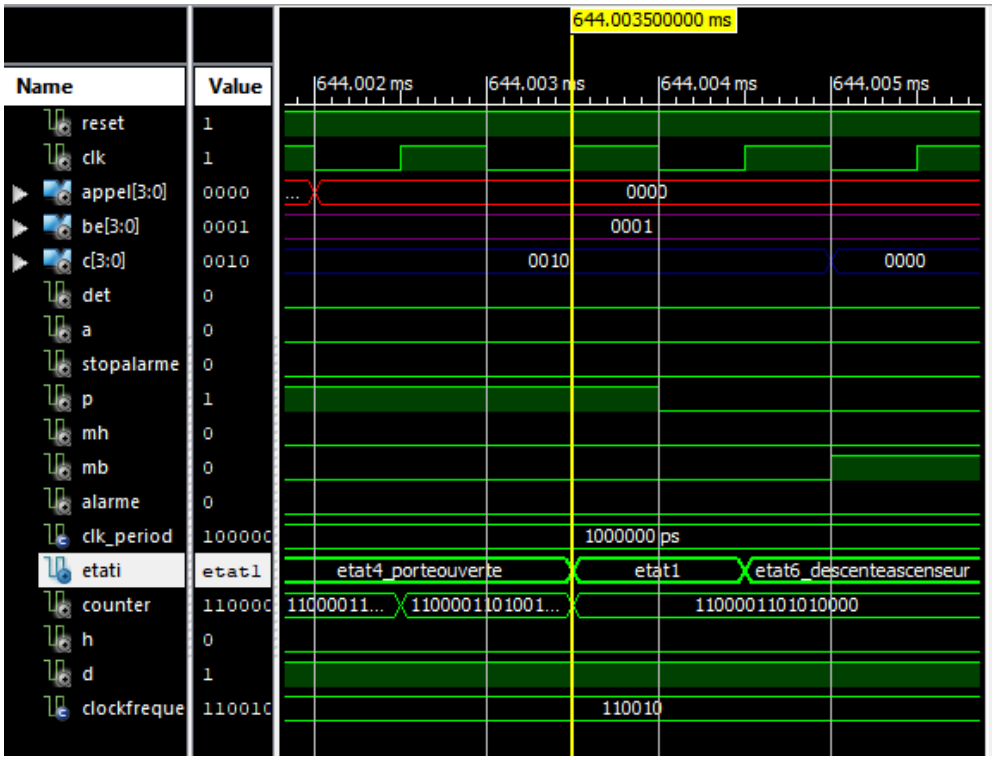
Etat2 2.0:



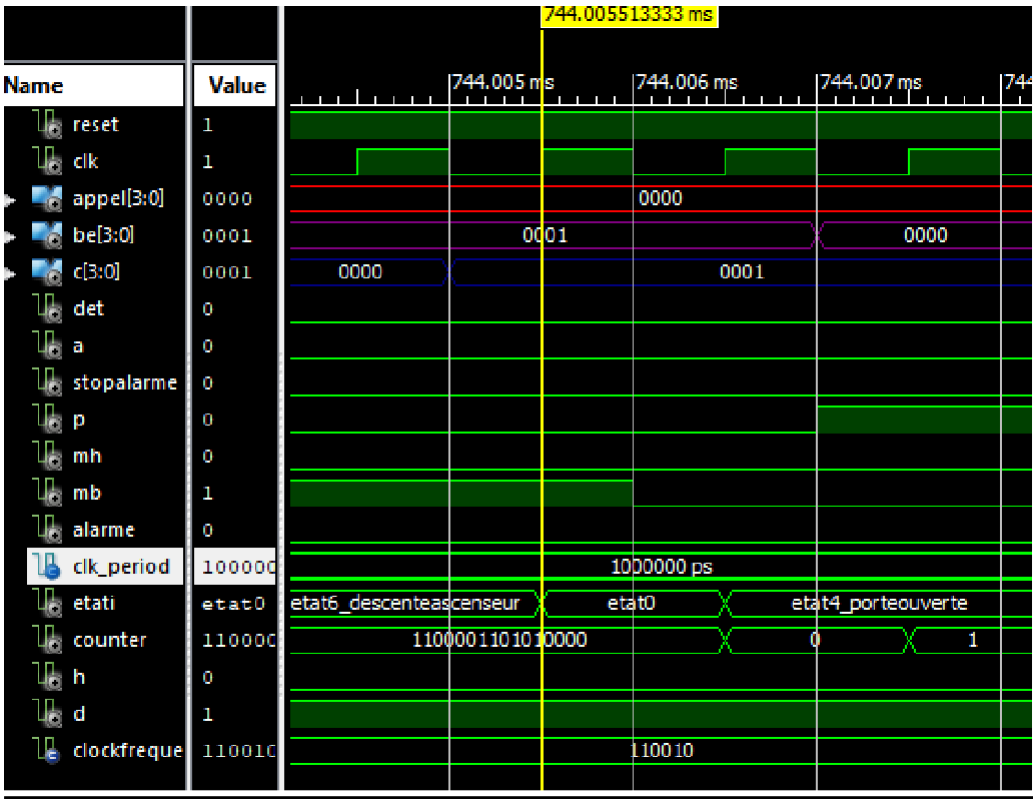
Etat1:



Etat1 2.0:



Etat0:


744.005513333 ms
744.005 ms
744.006 ms
744.007 ms
Name
Value
reset
1
clk
1
appel[3:0]
0000
be[3:0]
0001
c[3:0]
0001
det
0
a
0
stopalarme
0
p
0
mh
0
mb
1
alarme
0
clk_period
100000
etati
etat0
counter
110000
h
0
d
1
clockfreque
110010
0000
0001
0000
0000
0001
1000000 ps
etat6_descenteascenseur
etat0
etat4_porteouverte
1100001101010000
0
1
110010

Scénario3:

En repos l'ascenseur est à l'étage 0 (C = "0001") avec porte fermée "**etat0**".
Suite à une demande extérieure de l'étage 2 et 1 obtenues par les bits 2 et 1 du vecteur APPEL "0110" (càd ceux qui ont demandé l'ascenseur se trouvent à l'étage 2 et 1) :

1) l'ascenseur commence à monter (passage à l'état: "**etat5_monteeascenseur**")
2) jusqu'à arriver à l'étage 1, avec porte fermée (passage à l'état: "**etat1**")
3) le vecteur APPEL indique la demande de l'étage 1 APPEL(1)='1' alors l'ascenseur s'arrête et ouvre la porte (passage à l'état: "**Etat4_PorteOuverte**")
4) la porte s'ouvre et une personne entre (le détecteur de présence (capteur) det passe à '1' pendant 2ms puis revient à '0'). Cette personne demande l'étage 0 par le bouton à l'intérieur de l'ascenseur BE(0) passe à '1'). Une fois la période d'ouverture (50ms) écoulée la porte se ferme (passage à l'état: "**etat1**")

=> On a pour le moment une demande de l'étage 2 dès le début APPEL(2)='1' et la personne qui a monté tout à l'heure a demandé l'étage 0 BE(0)='1'

5) Comme l'ascenseur était déjà en montée il continue alors à monter et donne la priorité à celui qui a demandé l'ascenseur de l'étage n°2 (passage à l'état: "**etat5_monteeascenseur**")
6) jusqu'à arriver à l'étage 2 avec porte fermée (passage à l'état: "**etat2**")
7) le vecteur APPEL indique la demande de l'étage 2 APPEL(2)='1' alors l'ascenseur s'arrête et ouvre la porte (passage à l'état: "**Etat4_PorteOuverte**")
8) la porte reste ouverte pendant T=50ms sans qu'aucune personne n'entre, une fois cette période écoulée la porte se ferme (passage à l'état: "**etat2**")

=> On se trouve maintenant avec une seule demande de l'étage 0 lors de l'étape 4 :

9) l'ascenseur commence à descendre (passage à l'état: " **Etat6_DescenteAscenseur** ")
10) jusqu'à arriver à l'étage 1, avec porte fermée (passage à l'état: **"etat1**")
11) pas de demande de l'étage 1 donc l'ascenseur ne s'arrête pas et continue à descendre (passage à l'état: " **Etat6_DescenteAscenseur** ")
12) Au cours de la descente quelqu'un appuie dur le bouton poussoir (capteur) a (a passé à '1') ce qui active l'alarme (actionneur) (alarme passe à '1')

13)-l'ascenseur descend jusqu'à arriver à l'étage 0 avec porte fermée (passage à l'état: "**etat0**")

- le problème est résolu donc le responsable appuie sur le bouton poussoir stopalarme (stopalarme passe à '1') ce qui désactive l'alarme (actionneur) (alarme passe à '0')

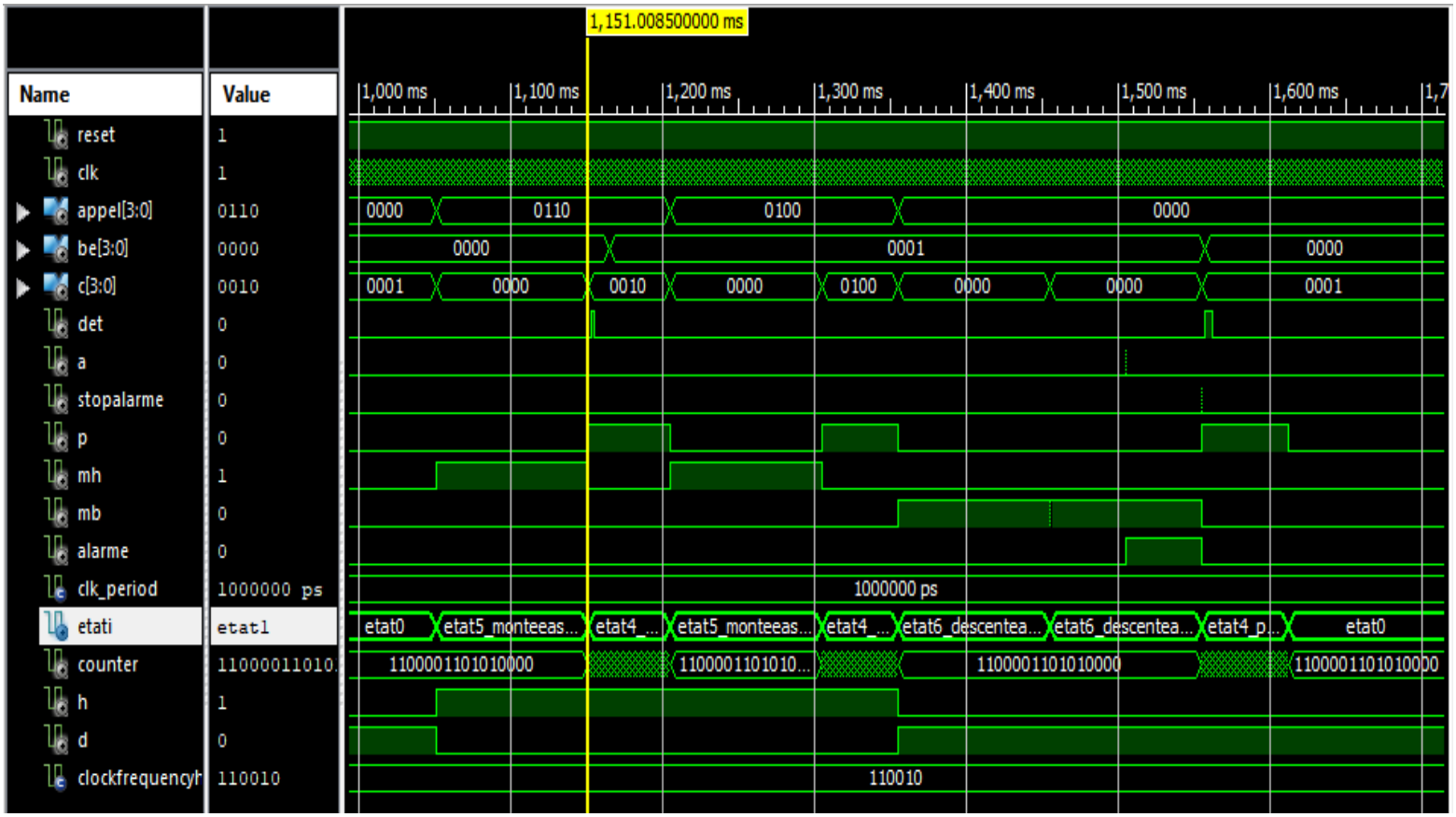
14) le vecteur BE indique la demande de l'étage 0 BE(0)='1' alors l'ascenseur s'arrête et ouvre la porte (passage à l'état: "**Etat4_PorteOuverte**")

15) la porte s'ouvre et les personnes sortent de l'ascenseur (le détecteur de présence (capteur) det passe à '1' pendant 5 ms puis revient à '0')

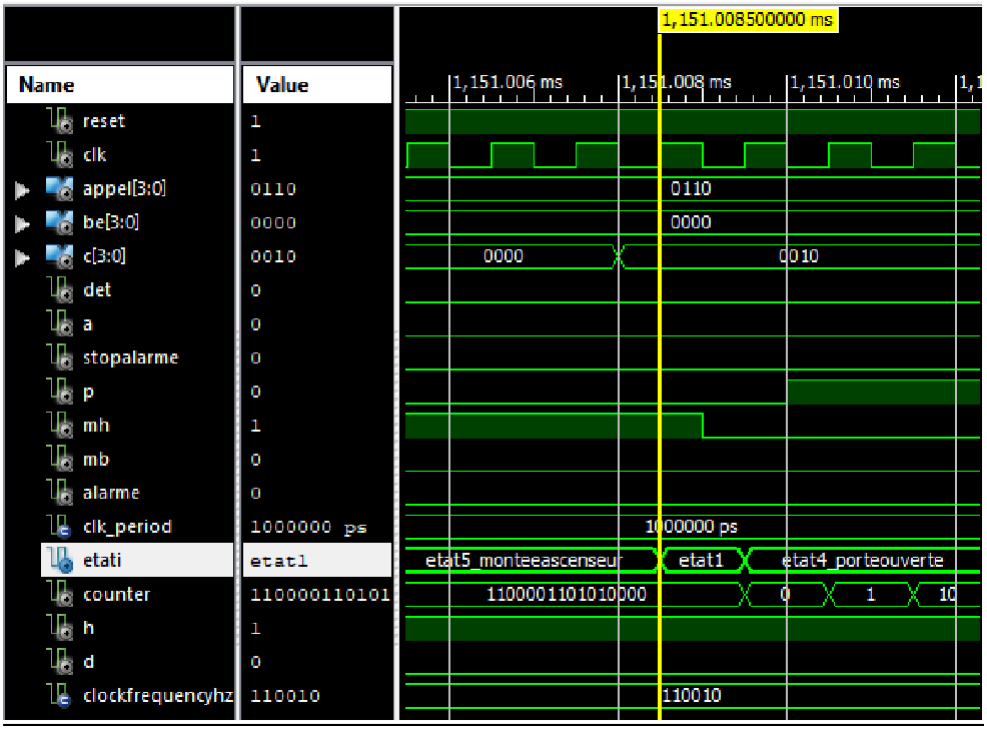
Une fois la période d'ouverture écoulée la porte se ferme (passage à l'état: "**etat0**")

- **<u>Donc le cycle sera comme suit</u>** :

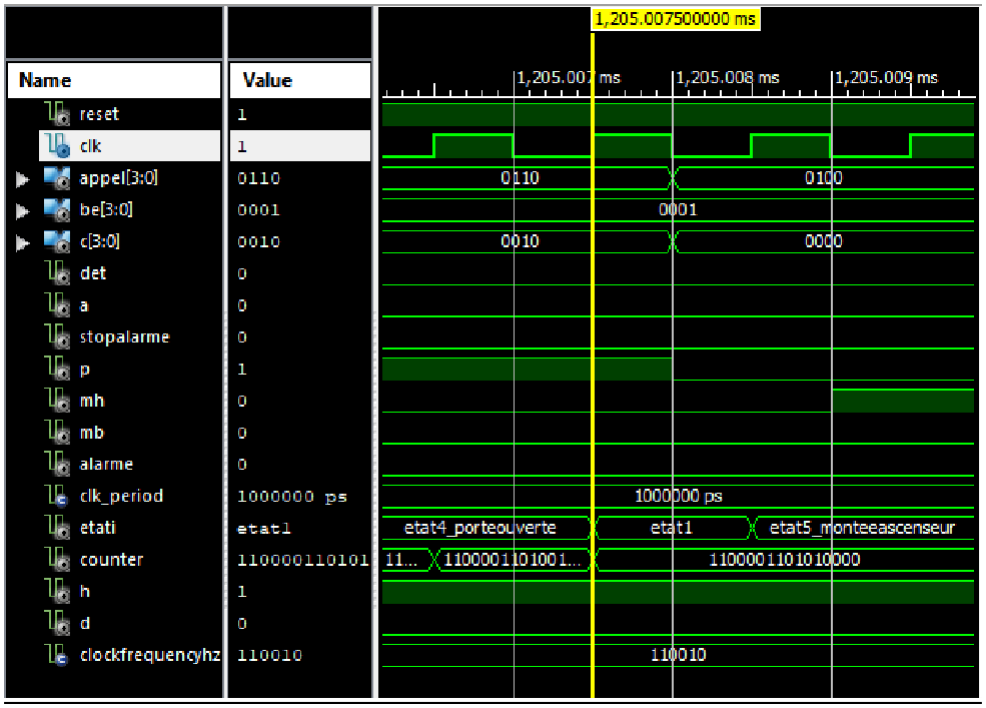
Etat0 → Etat5MonteAscenseur → etat1 → etat4PorteOuverte → etat1 (fig : etat1 2.0) → etat5MonteAscenseur → etat2 → etat4PorteOuverte → etat2 (fig : etat2 2.0) → etat6DescenteAscenseur → etat1 (fig : etat1 3.0) → etat6DescenteAscenseur → etat0 (fig : etat 2.0) → Etat4PorteOuverte → etat0



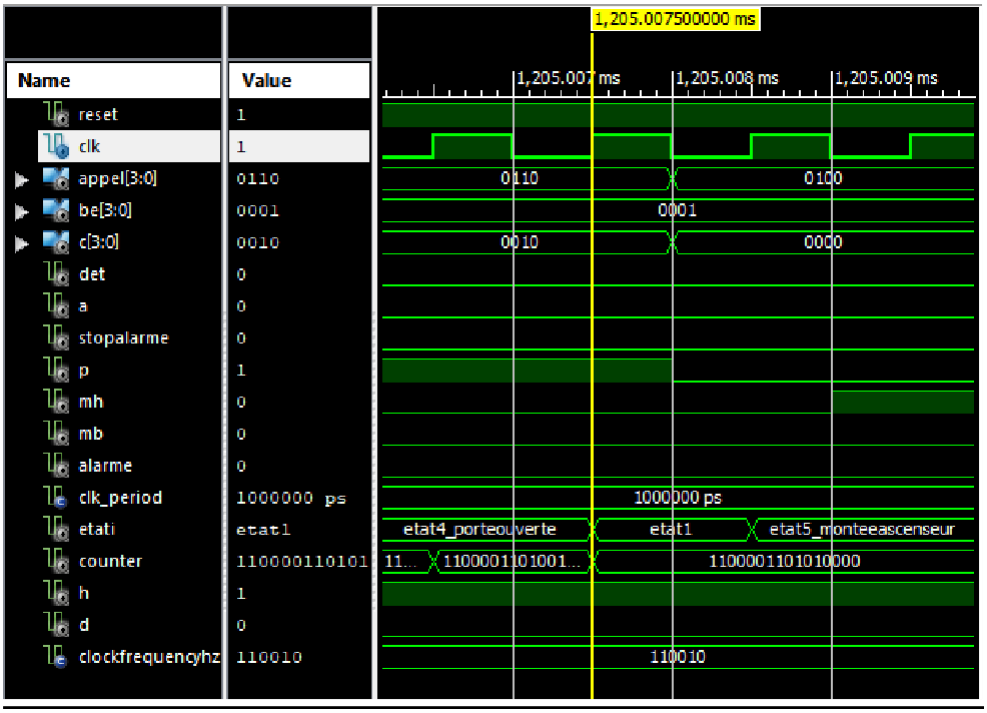
Etat1:



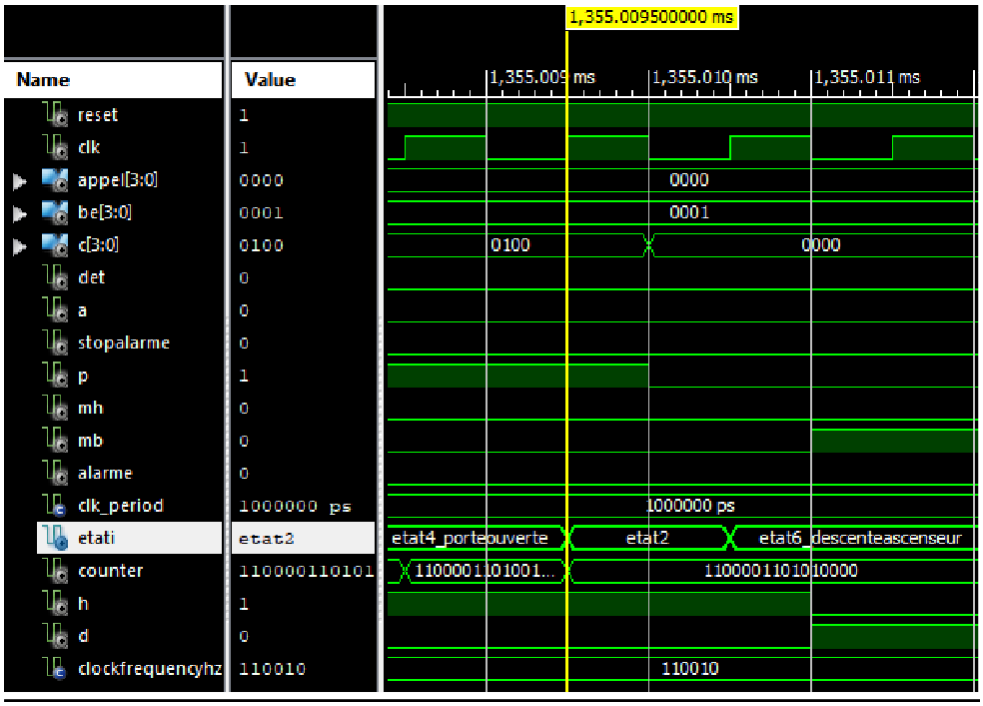
Etat1 2.0:



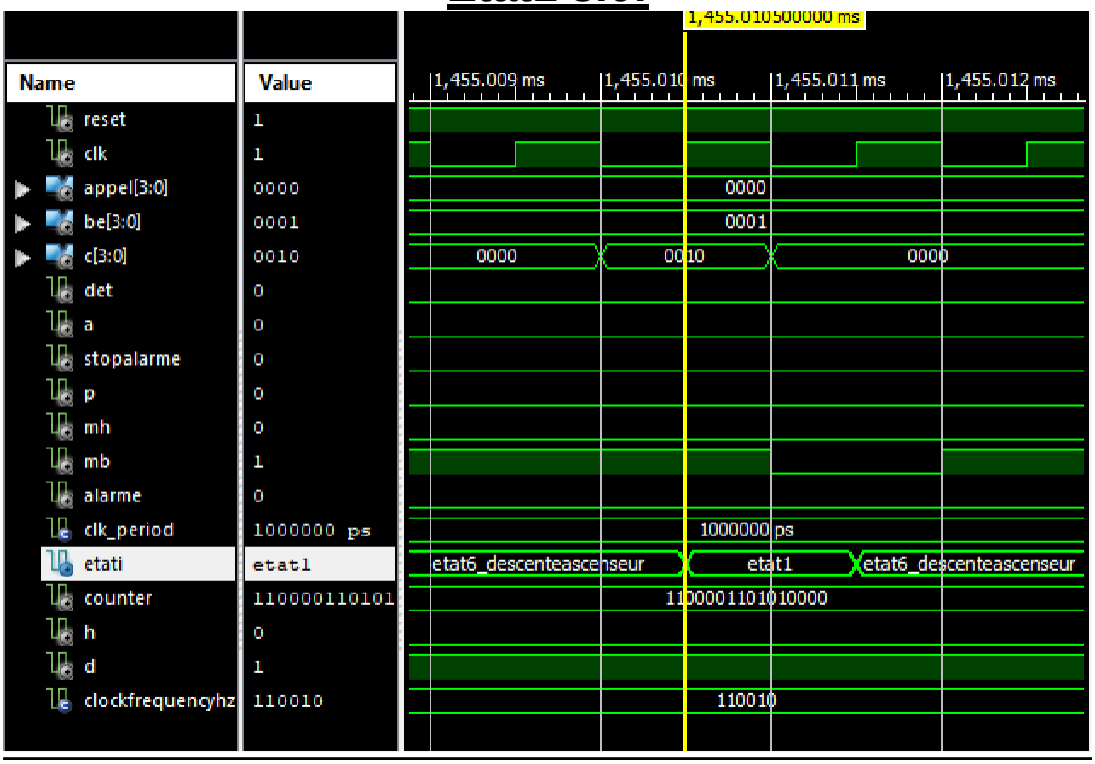
Etat2:



Etat2 2.0:



Etat1 3.0:



Etat0 2.0: