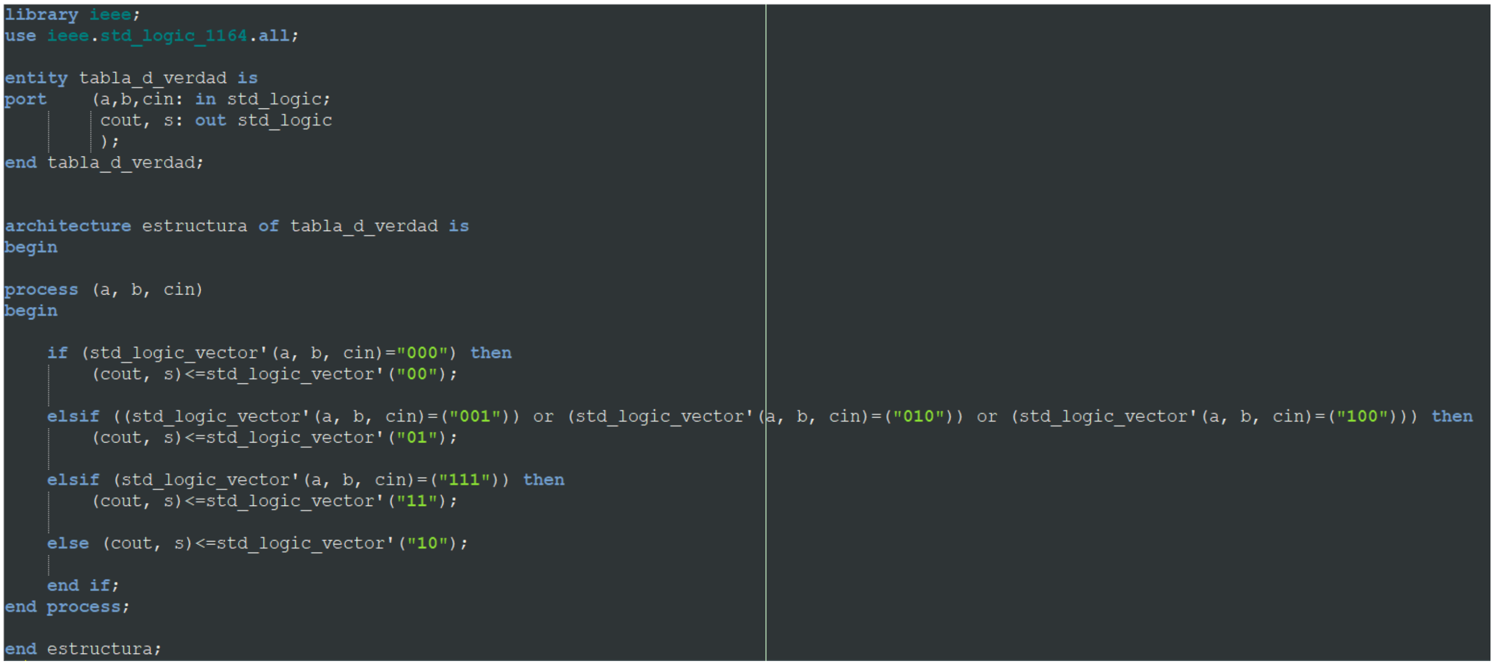
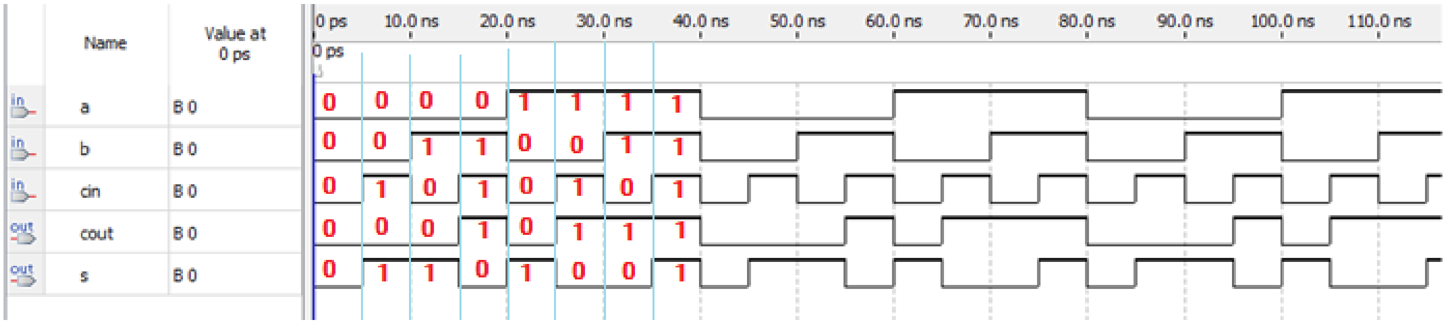
1)

a-(pego el Código desde otro ide porque, si no, no me entraba en la captura)

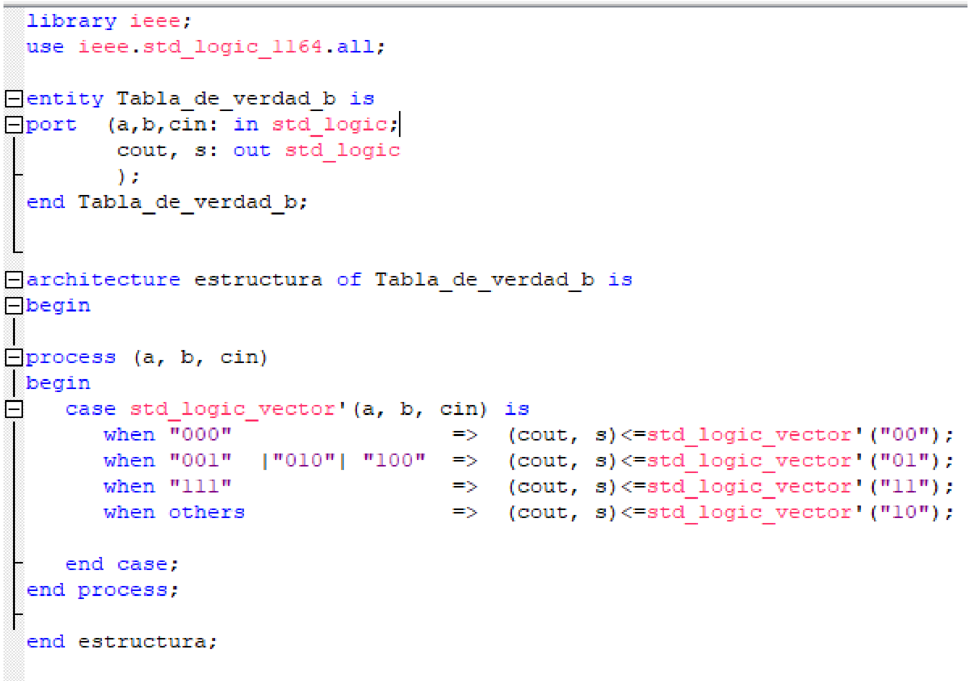


```
library ieee;
use ieee.std_logic_1164.all;

entity tabla_d_verdad is
port     (a,b,cin: in std_logic;
          cout, s: out std_logic
          );
end tabla_d_verdad;


architecture estructura of tabla_d_verdad is
begin

process (a, b, cin)
begin

    if (std_logic_vector'(a, b, cin)="000") then
        (cout, s)<=std_logic_vector'("00");

    elsif ((std_logic_vector'(a, b, cin)=("001")) or (std_logic_vector'(a, b, cin)=("010")) or (std_logic_vector'(a, b, cin)=("100"))) then
        (cout, s)<=std_logic_vector'("01");

    elsif (std_logic_vector'(a, b, cin)=("111")) then
        (cout, s)<=std_logic_vector'("11");

    else (cout, s)<=std_logic_vector'("10");

    end if;
end process;

end estructura;
```

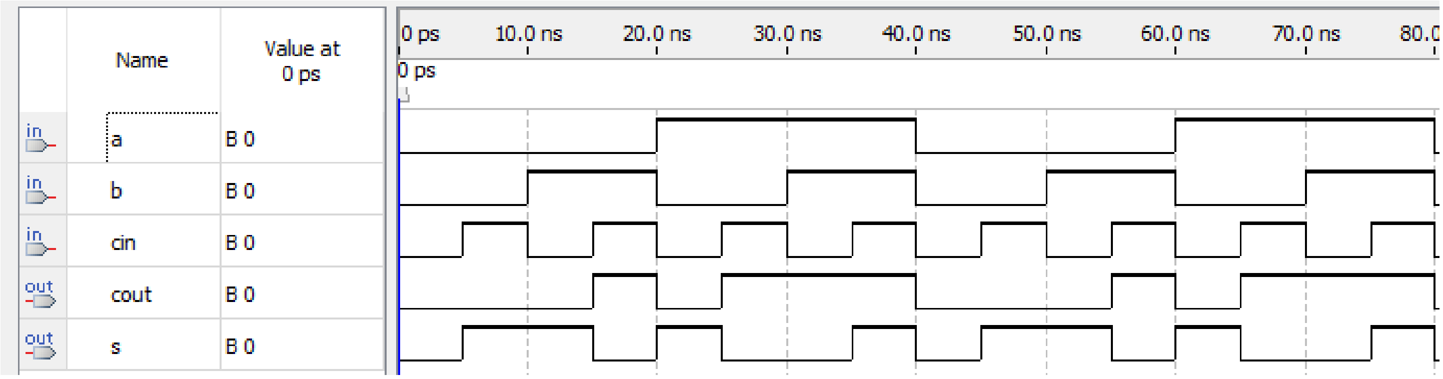


Name	Value at 0 ps
a	B 0
b	B 0
cin	B 0
cout	B 0
s	B 0

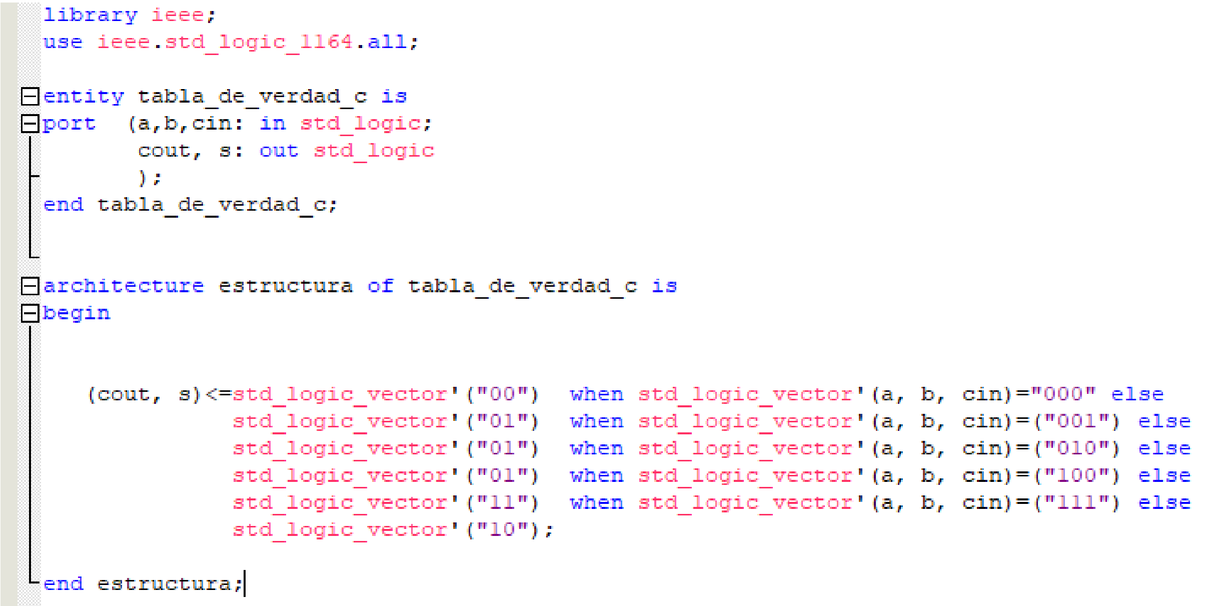
b-



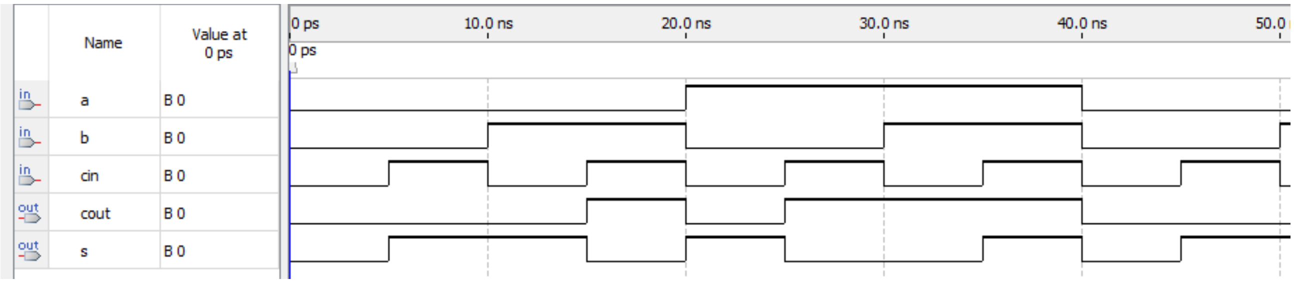
```
library ieee;
use ieee.std_logic_1164.all;

entity Tabla_de_verdad_b is
port   (a,b,cin: in std_logic;
        cout, s: out std_logic
        );
end Tabla_de_verdad_b;


architecture estructura of Tabla_de_verdad_b is
begin

process (a, b, cin)
begin
   case std_logic_vector'(a, b, cin) is
      when "000"                 =>  (cout, s)<=std_logic_vector'("00");
      when "001"  |"010"| "100"  =>  (cout, s)<=std_logic_vector'("01");
      when "111"                 =>  (cout, s)<=std_logic_vector'("11");
      when others                =>  (cout, s)<=std_logic_vector'("10");

   end case;
end process;

end estructura;
```



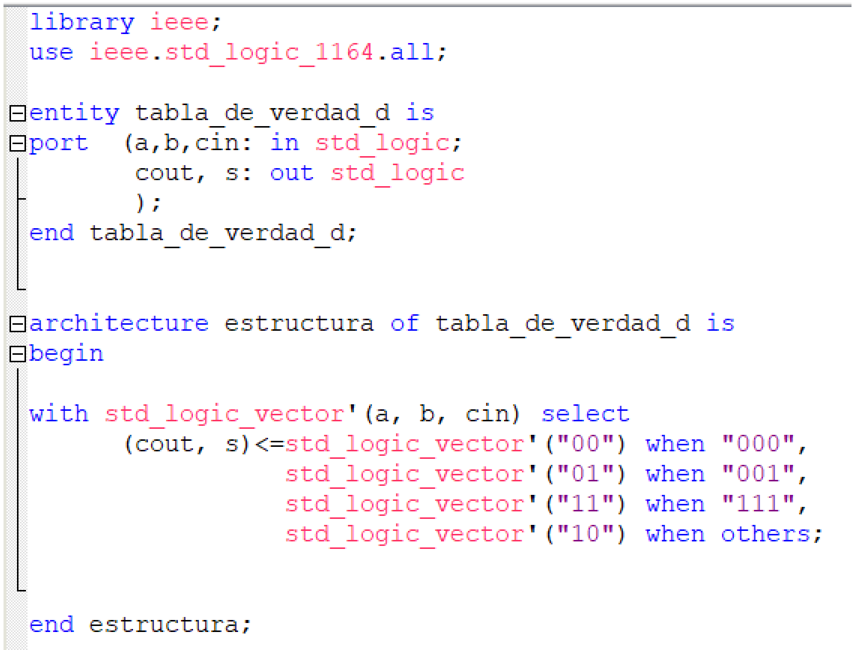
c-



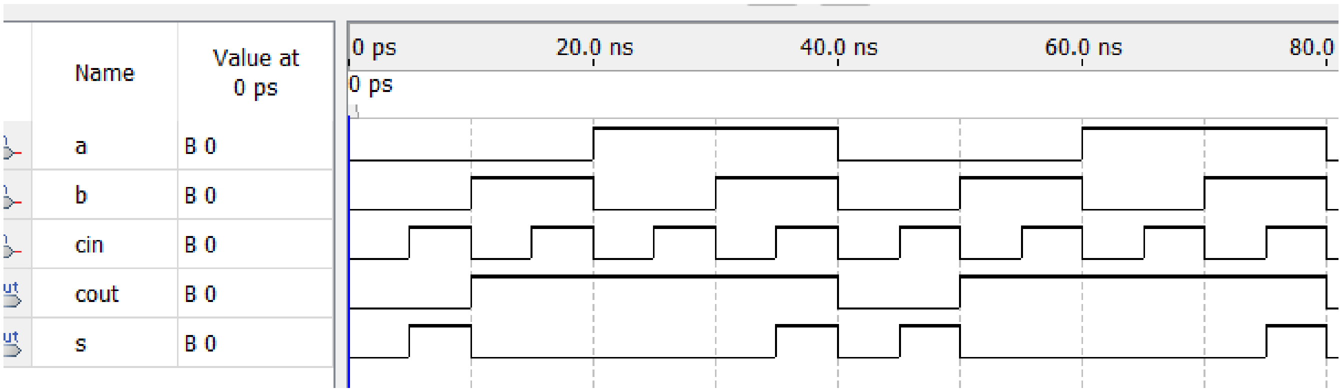
```
library ieee;
use ieee.std_logic_1164.all;

entity tabla_de_verdad_c is
port   (a,b,cin: in std_logic;
        cout, s: out std_logic
        );
end tabla_de_verdad_c;

architecture estructura of tabla_de_verdad_c is
begin


   (cout, s)<=std_logic_vector'("00")  when std_logic_vector'(a, b, cin)="000" else
              std_logic_vector'("01")  when std_logic_vector'(a, b, cin)=("001") else
              std_logic_vector'("01")  when std_logic_vector'(a, b, cin)=("010") else
              std_logic_vector'("01")  when std_logic_vector'(a, b, cin)=("100") else
              std_logic_vector'("11")  when std_logic_vector'(a, b, cin)=("111") else
              std_logic_vector'("10");

end estructura;
```



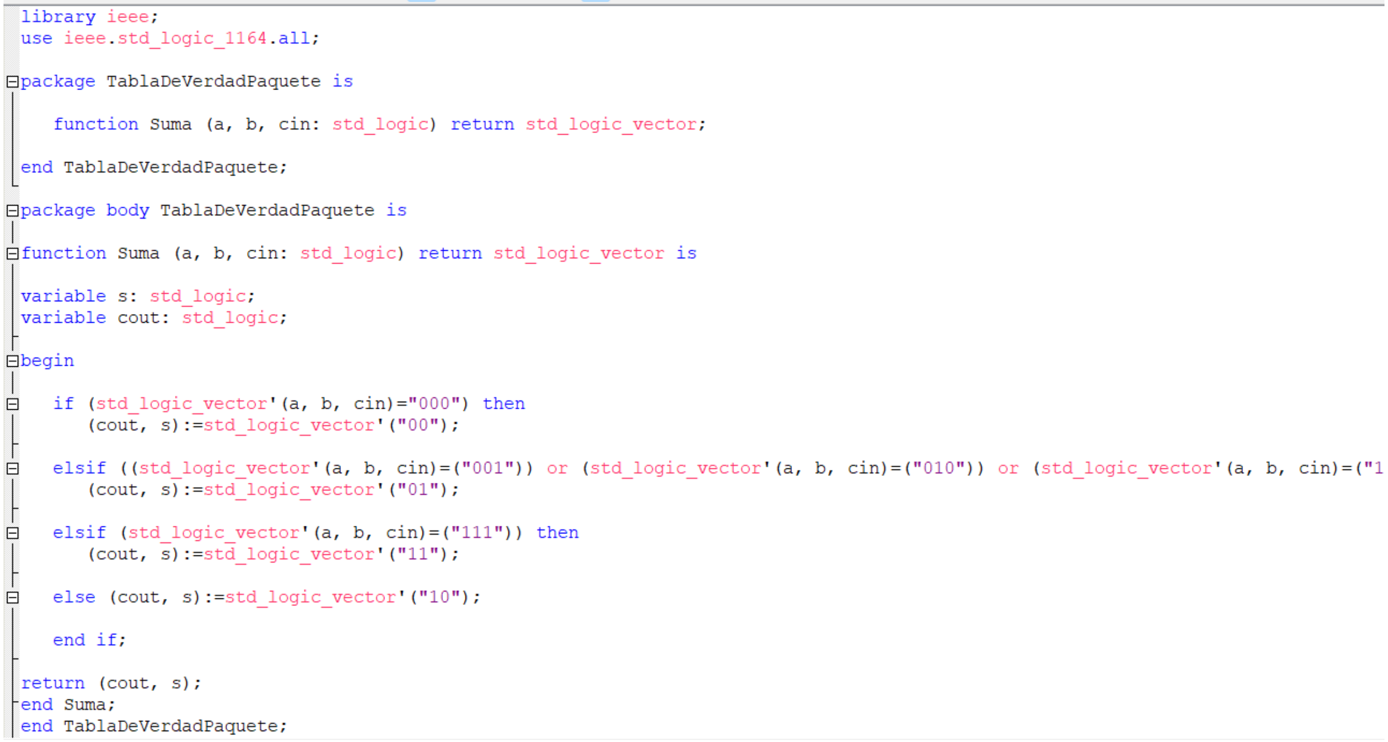
d-



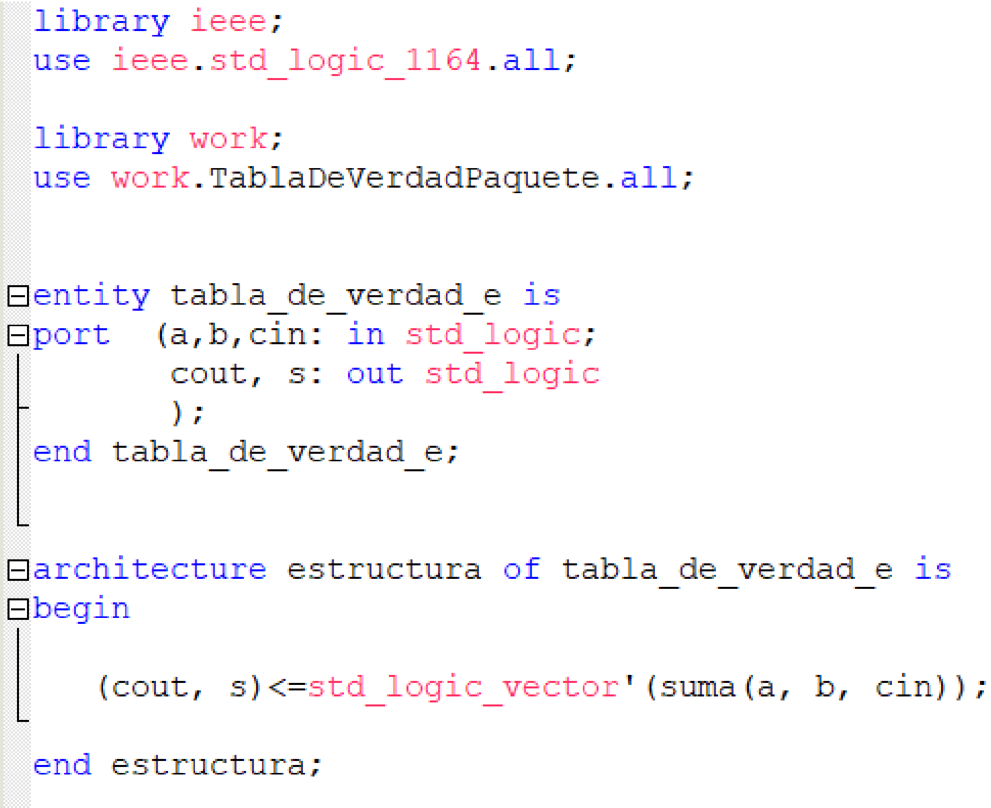
```vhdl
library ieee;
use ieee.std_logic_1164.all;

entity tabla_de_verdad_d is
port   (a,b,cin: in std_logic;
        cout, s: out std_logic
        );
end tabla_de_verdad_d;


architecture estructura of tabla_de_verdad_d is
begin

with std_logic_vector'(a, b, cin) select
      (cout, s)<=std_logic_vector'("00") when "000",
                 std_logic_vector'("01") when "001",
                 std_logic_vector'("11") when "111",
                 std_logic_vector'("10") when others;


end estructura;
```



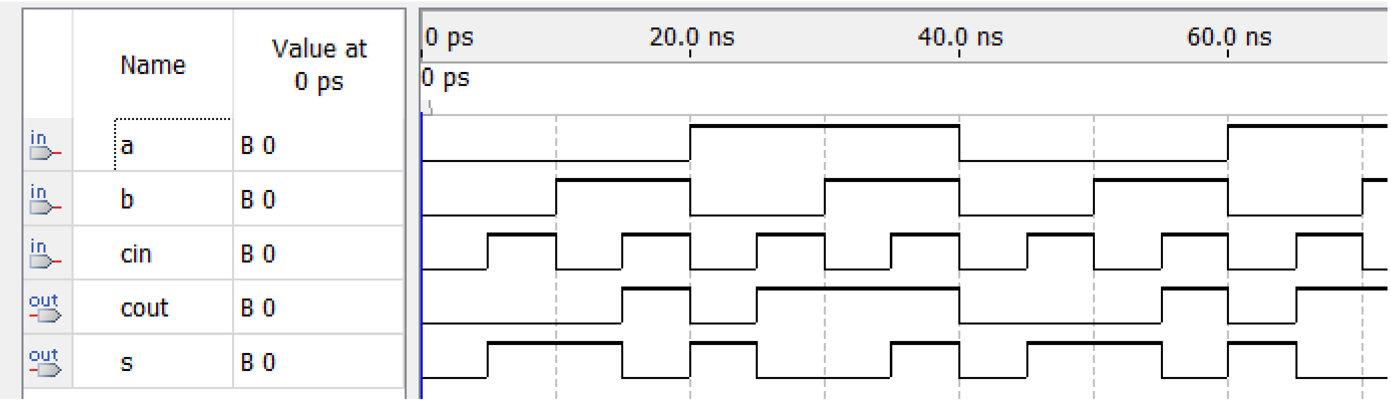
e-



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

package TablaDeVerdadPaquete is

   function Suma (a, b, cin: std_logic) return std_logic_vector;

end TablaDeVerdadPaquete;

package body TablaDeVerdadPaquete is

function Suma (a, b, cin: std_logic) return std_logic_vector is

variable s: std_logic;
variable cout: std_logic;

begin

   if (std_logic_vector'(a, b, cin)="000") then
      (cout, s):=std_logic_vector'("00");

   elsif ((std_logic_vector'(a, b, cin)=("001")) or (std_logic_vector'(a, b, cin)=("010")) or (std_logic_vector'(a, b, cin)=("1
      (cout, s):=std_logic_vector'("01");

   elsif (std_logic_vector'(a, b, cin)=("111")) then
      (cout, s):=std_logic_vector'("11");

   else (cout, s):=std_logic_vector'("10");

   end if;

return (cout, s);
end Suma;
end TablaDeVerdadPaquete;
```

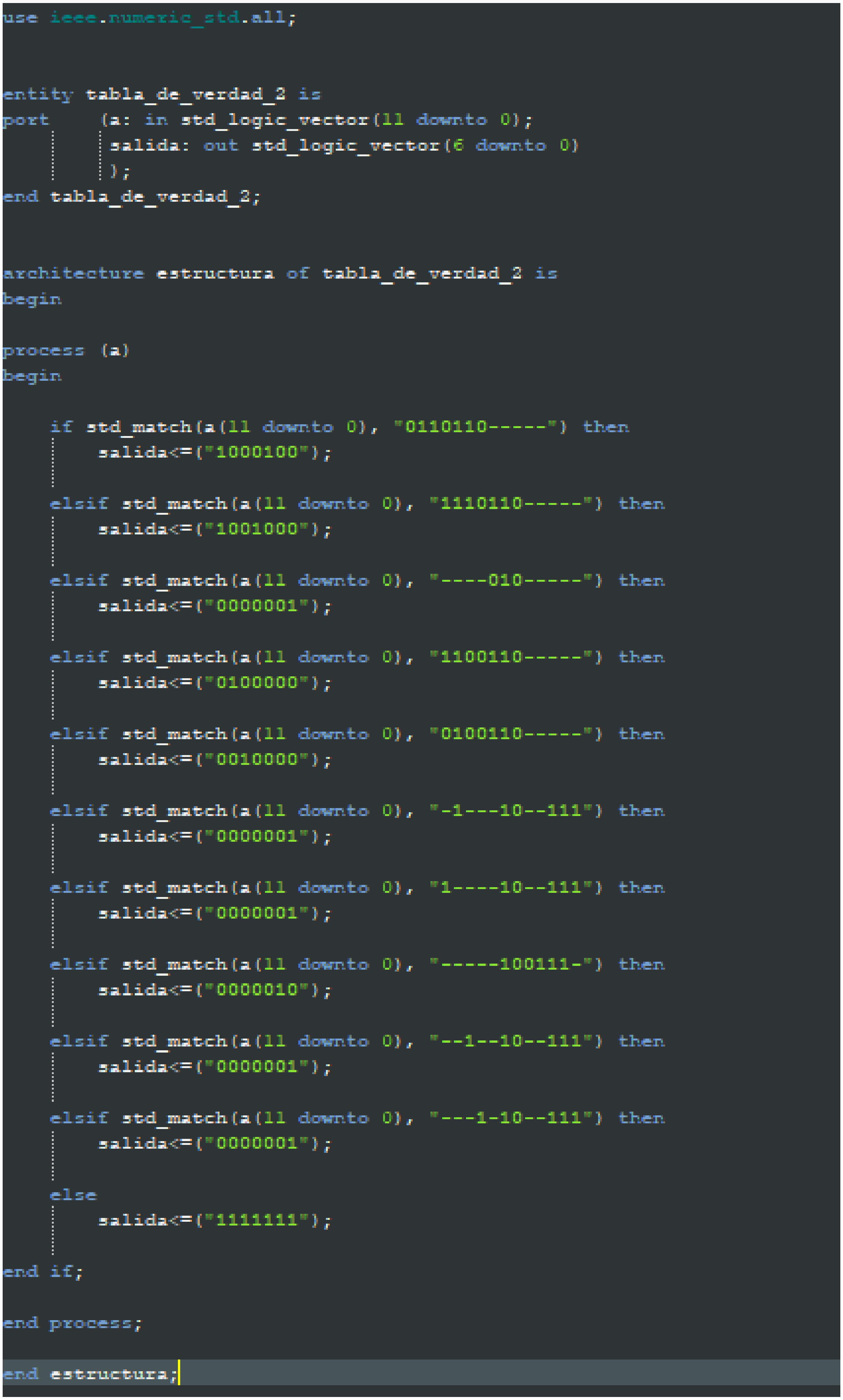


```vhdl
library ieee;
use ieee.std_logic_1164.all;

library work;
use work.TablaDeVerdadPaquete.all;


entity tabla_de_verdad_e is
port   (a,b,cin: in std_logic;
        cout, s: out std_logic
        );
end tabla_de_verdad_e;


architecture estructura of tabla_de_verdad_e is
begin

   (cout, s)<=std_logic_vector'(suma(a, b, cin));

end estructura;
```

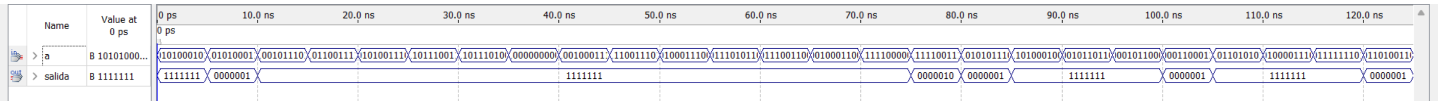


Name	Value at 0 ps
a	B 0
b	B 0
cin	B 0
cout	B 0
s	B 0

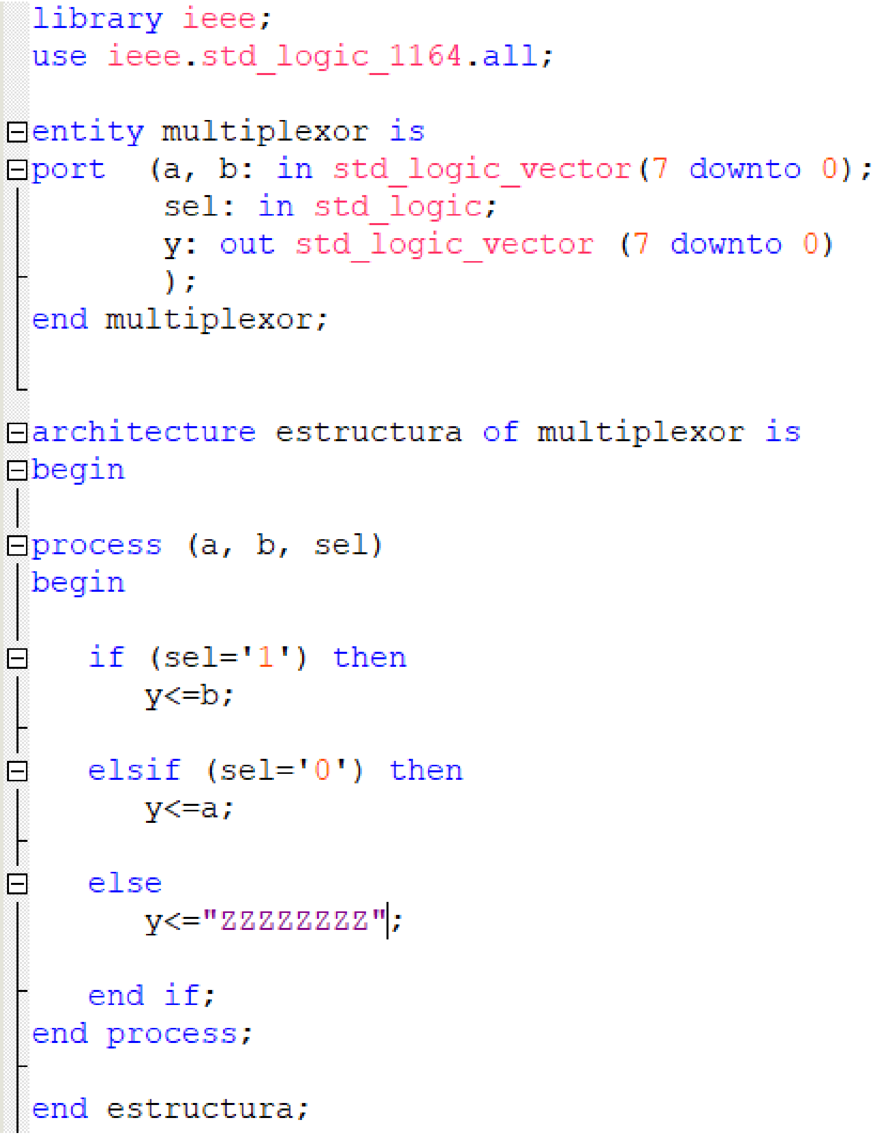
2)



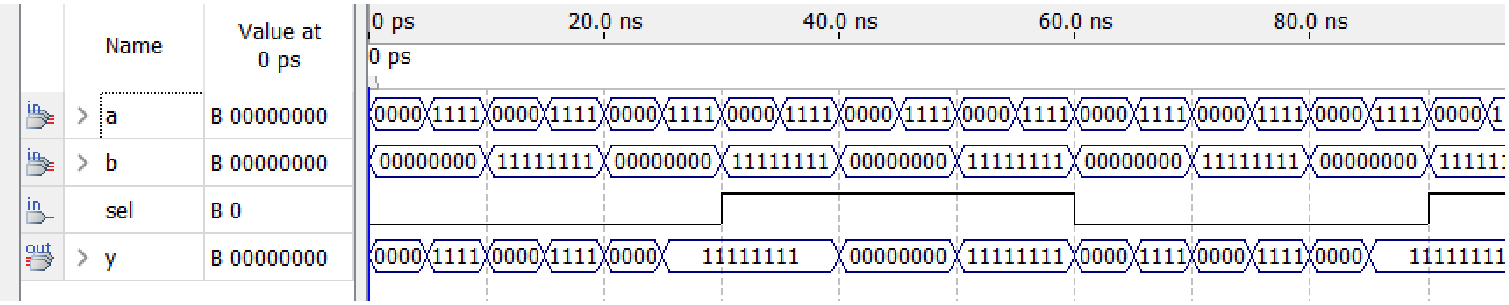
```
use ieee.numeric_std.all;


entity tabla_de_verdad_2 is
port    (a: in std_logic_vector(11 downto 0);
         salida: out std_logic_vector(6 downto 0)
         );
end tabla_de_verdad_2;


architecture estructura of tabla_de_verdad_2 is
begin

process (a)
begin

    if std_match(a(11 downto 0), "0110110-----") then
        salida<=("1000100");

    elsif std_match(a(11 downto 0), "1110110-----") then
        salida<=("1001000");

    elsif std_match(a(11 downto 0), "----010-----") then
        salida<=("0000001");

    elsif std_match(a(11 downto 0), "1100110-----") then
        salida<=("0100000");

    elsif std_match(a(11 downto 0), "0100110-----") then
        salida<=("0010000");

    elsif std_match(a(11 downto 0), "-1---10--111") then
        salida<=("0000001");

    elsif std_match(a(11 downto 0), "1----10--111") then
        salida<=("0000001");

    elsif std_match(a(11 downto 0), "-----100111-") then
        salida<=("0000010");

    elsif std_match(a(11 downto 0), "--1--10--111") then
        salida<=("0000001");

    elsif std_match(a(11 downto 0), "---1-10--111") then
        salida<=("0000001");

    else
        salida<=("1111111");

end if;

end process;

end estructura;
```



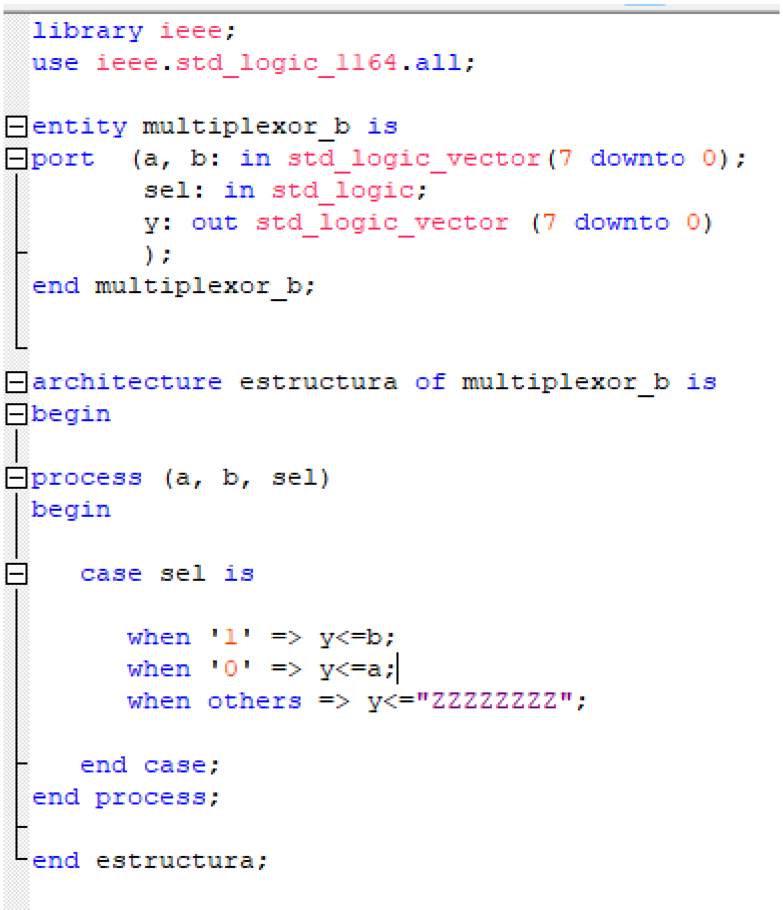
3) a-



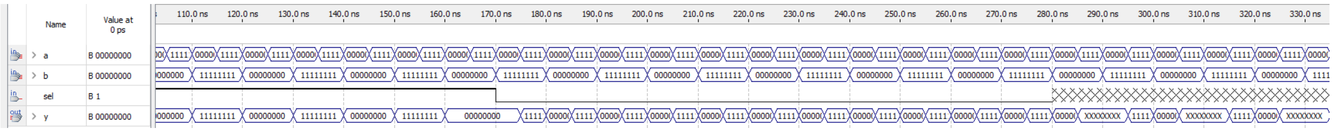
```
library ieee;
use ieee.std_logic_1164.all;

entity multiplexor is
port   (a, b: in std_logic_vector(7 downto 0);
        sel: in std_logic;
        y: out std_logic_vector (7 downto 0)
        );
end multiplexor;


architecture estructura of multiplexor is
begin

process (a, b, sel)
begin

   if (sel='1') then
      y<=b;

   elsif (sel='0') then
      y<=a;

   else
      y<="ZZZZZZZZ";

   end if;
end process;

end estructura;
```



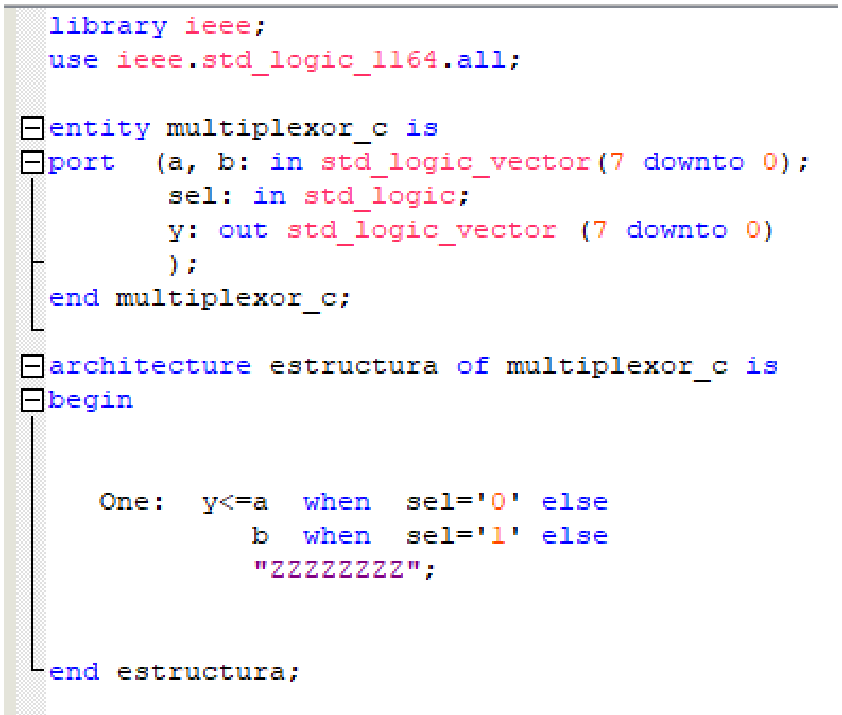
b-



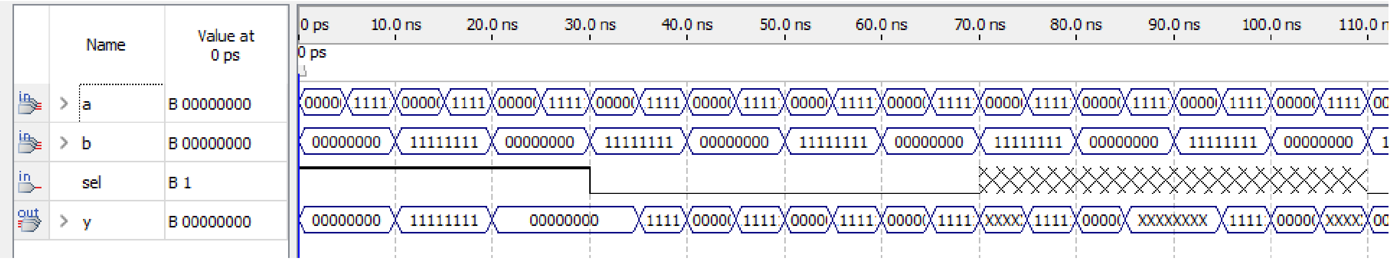
```
library ieee;
use ieee.std_logic_1164.all;

entity multiplexor_b is
port   (a, b: in std_logic_vector(7 downto 0);
        sel: in std_logic;
        y: out std_logic_vector (7 downto 0)
        );
end multiplexor_b;


architecture estructura of multiplexor_b is
begin

process (a, b, sel)
begin

   case sel is

      when '1' => y<=b;
      when '0' => y<=a;
      when others => y<="ZZZZZZZZ";

   end case;
end process;

end estructura;
```



c-

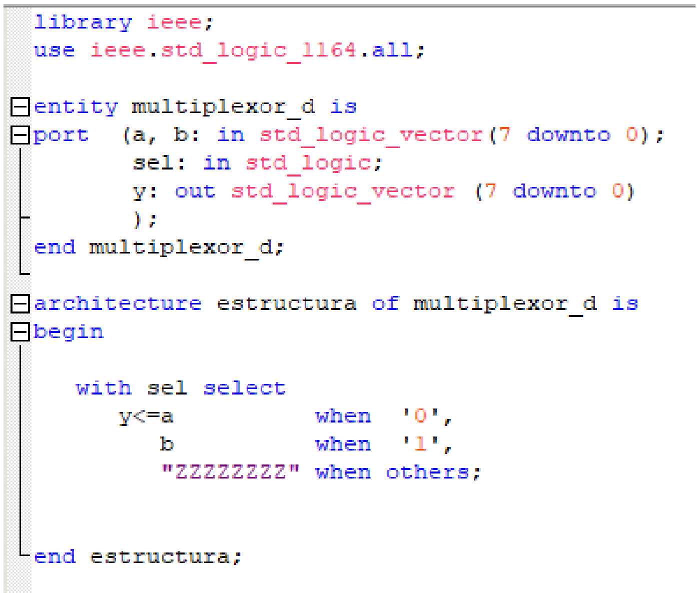


```
library ieee;
use ieee.std_logic_1164.all;

entity multiplexor_c is
port   (a, b: in std_logic_vector(7 downto 0);
        sel: in std_logic;
        y: out std_logic_vector (7 downto 0)
        );
end multiplexor_c;

architecture estructura of multiplexor_c is
begin


   One:  y<=a  when  sel='0' else
             b  when  sel='1' else
             "ZZZZZZZZ";


end estructura;
```

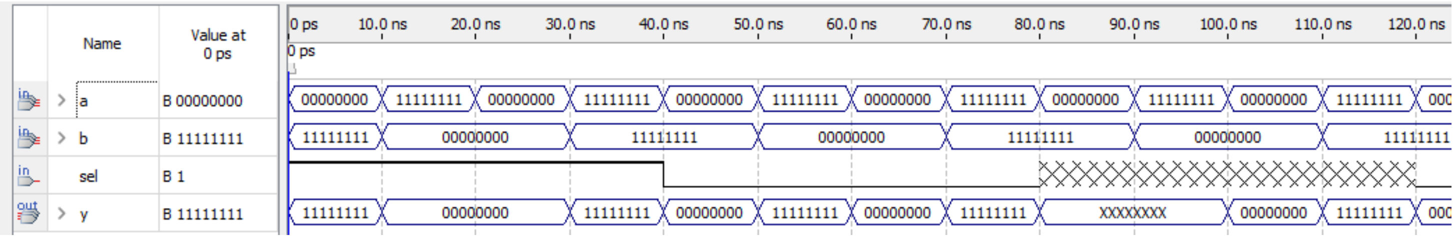


Name	Value at 0 ps
a	B 00000000
b	B 00000000
sel	B 1
y	B 00000000

d-

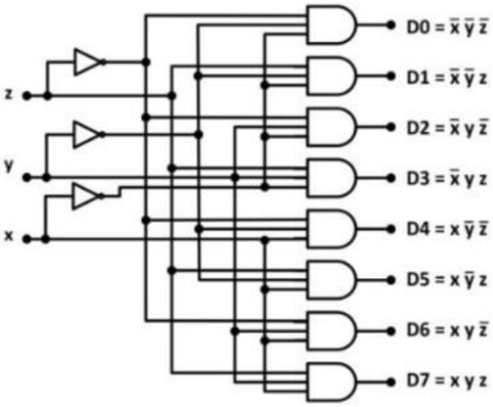


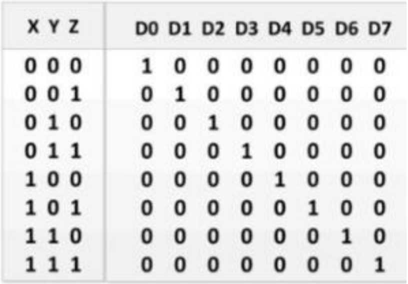
```
library ieee;
use ieee.std_logic_1164.all;

entity multiplexor_d is
port   (a, b: in std_logic_vector(7 downto 0);
        sel: in std_logic;
        y: out std_logic_vector (7 downto 0)
        );
end multiplexor_d;

architecture estructura of multiplexor_d is
begin

   with sel select
      y<=a          when  '0',
         b          when  '1',
         "ZZZZZZZZ" when others;


end estructura;
```



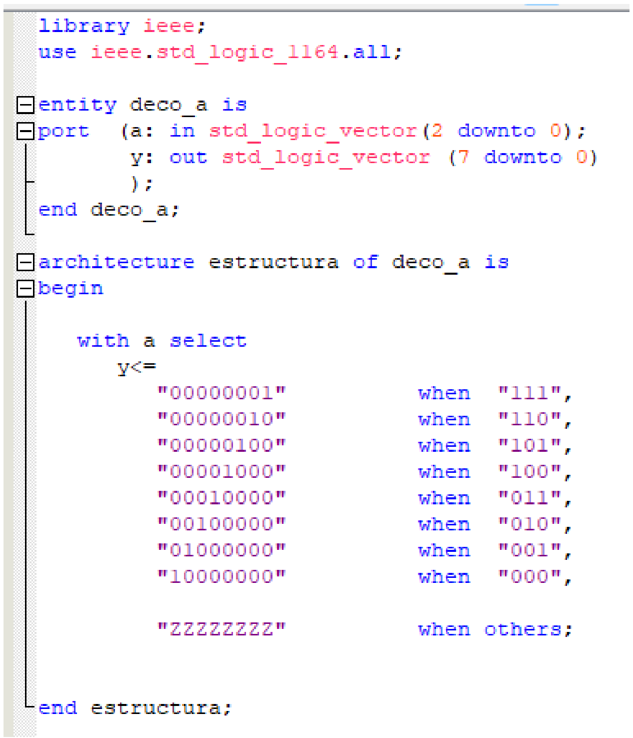
Name	Value at 0 ps
a	B 00000000
b	B 11111111
sel	B 1
y	B 11111111

4)

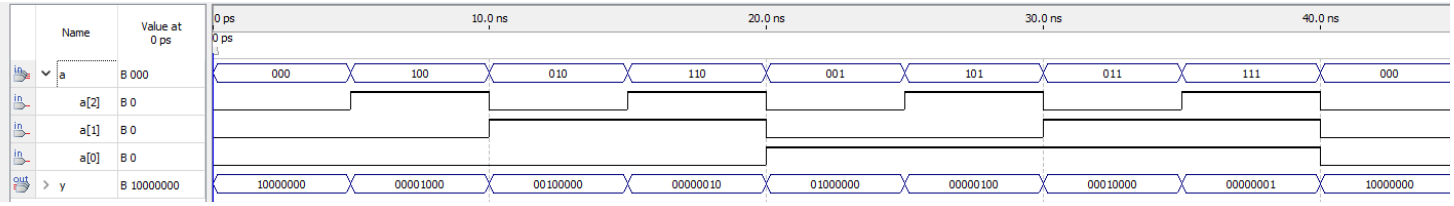

z
y
x
D0 = $\bar{x}\,\bar{y}\,\bar{z}$
D1 = $\bar{x}\,\bar{y}\,z$
D2 = $\bar{x}\,y\,\bar{z}$
D3 = $\bar{x}\,y\,z$
D4 = $x\,\bar{y}\,\bar{z}$
D5 = $x\,\bar{y}\,z$
D6 = $x\,y\,\bar{z}$
D7 = $x\,y\,z$



X Y Z	D0	D1	D2	D3	D4	D5	D6	D7
0 0 0	1	0	0	0	0	0	0	0
0 0 1	0	1	0	0	0	0	0	0
0 1 0	0	0	1	0	0	0	0	0
0 1 1	0	0	0	1	0	0	0	0
1 0 0	0	0	0	0	1	0	0	0
1 0 1	0	0	0	0	0	1	0	0
1 1 0	0	0	0	0	0	0	1	0
1 1 1	0	0	0	0	0	0	0	1

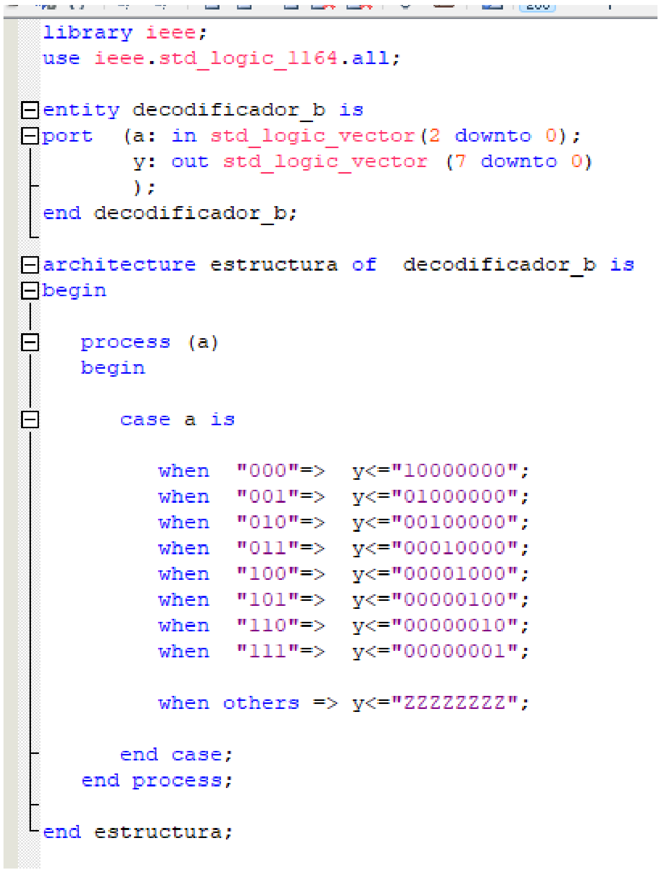


```
library ieee;
use ieee.std_logic_1164.all;

entity deco_a is
port   (a: in std_logic_vector(2 downto 0);
        y: out std_logic_vector (7 downto 0)
        );
end deco_a;

architecture estructura of deco_a is
begin

   with a select
      y<=
         "00000001"          when  "111",
         "00000010"          when  "110",
         "00000100"          when  "101",
         "00001000"          when  "100",
         "00010000"          when  "011",
         "00100000"          when  "010",
         "01000000"          when  "001",
         "10000000"          when  "000",

         "ZZZZZZZZ"          when others;


end estructura;
```

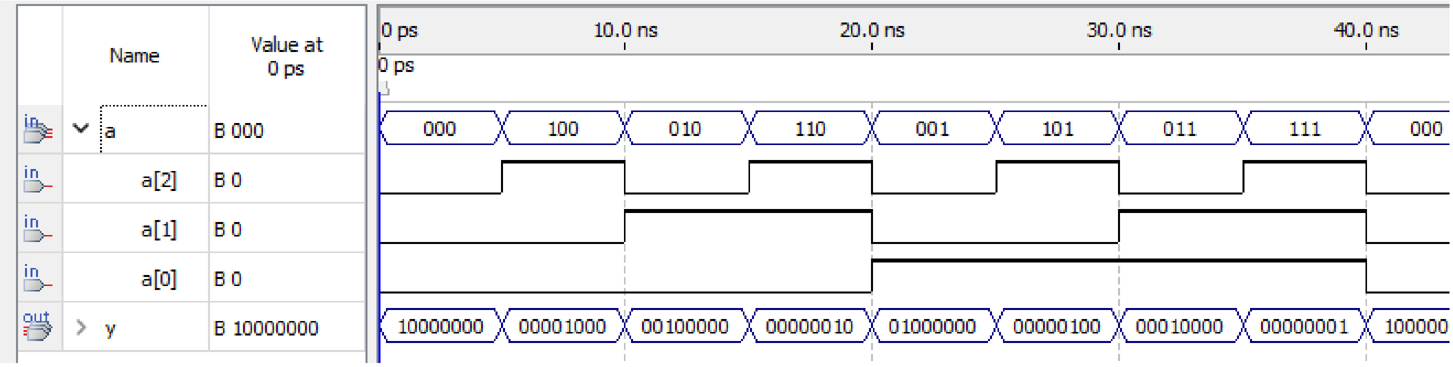


Name	Value at 0 ps
a	B 000
a[2]	B 0
a[1]	B 0
a[0]	B 0
y	B 10000000

b-

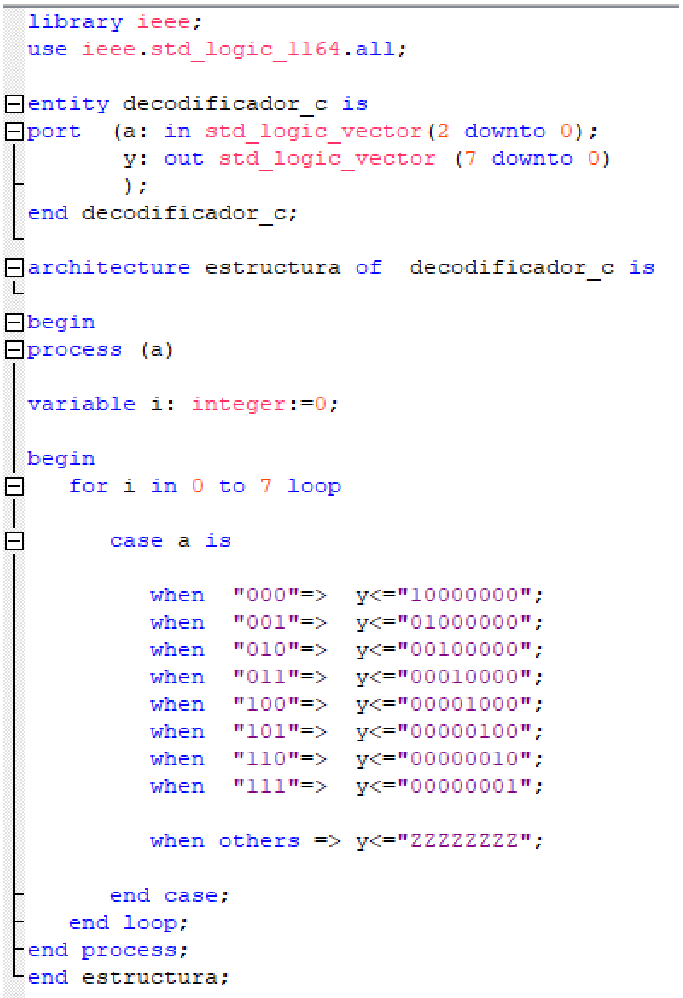


```
library ieee;
use ieee.std_logic_1164.all;

entity decodificador_b is
port   (a: in std_logic_vector(2 downto 0);
        y: out std_logic_vector (7 downto 0)
        );
end decodificador_b;

architecture estructura of  decodificador_b is
begin

   process  (a)
   begin

      case a is

         when  "000"=>  y<="10000000";
         when  "001"=>  y<="01000000";
         when  "010"=>  y<="00100000";
         when  "011"=>  y<="00010000";
         when  "100"=>  y<="00001000";
         when  "101"=>  y<="00000100";
         when  "110"=>  y<="00000010";
         when  "111"=>  y<="00000001";

         when others => y<="ZZZZZZZZ";

      end case;
   end process;

end estructura;
```

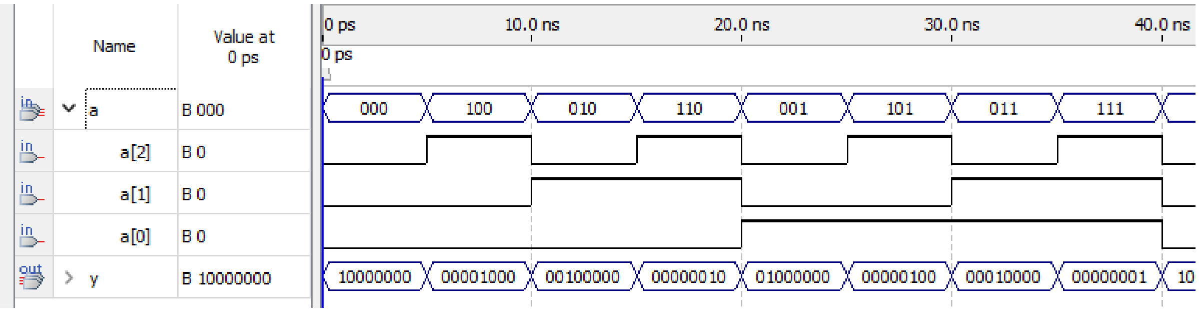


Name	Value at 0 ps
a	B 000
a[2]	B 0
a[1]	B 0
a[0]	B 0
y	B 10000000

c-



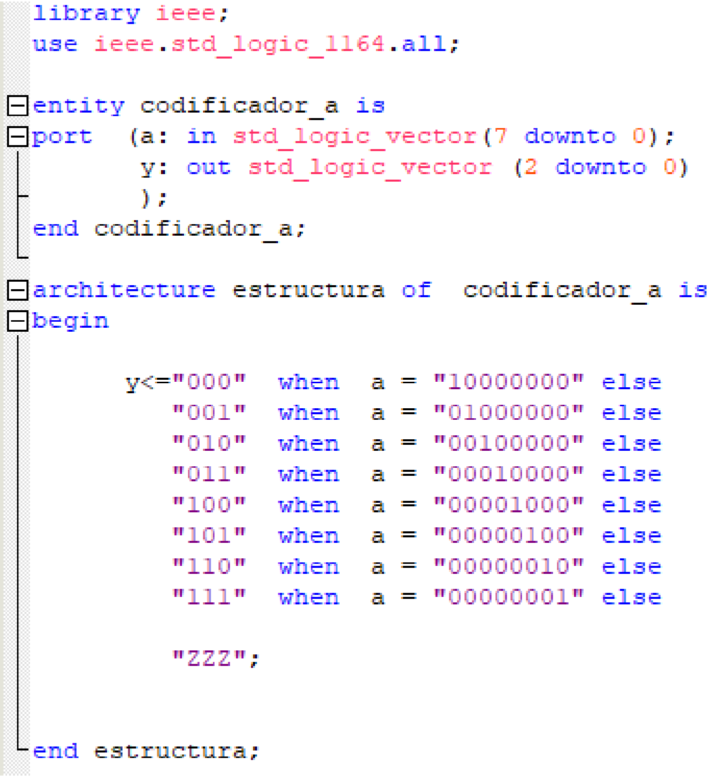
```
library ieee;
use ieee.std_logic_1164.all;

entity decodificador_c is
port  (a: in std_logic_vector(2 downto 0);
       y: out std_logic_vector (7 downto 0)
       );
end decodificador_c;

architecture estructura of  decodificador_c is

begin
process (a)

variable i: integer:=0;

begin
   for i in 0 to 7 loop

      case a is

         when  "000"=>  y<="10000000";
         when  "001"=>  y<="01000000";
         when  "010"=>  y<="00100000";
         when  "011"=>  y<="00010000";
         when  "100"=>  y<="00001000";
         when  "101"=>  y<="00000100";
         when  "110"=>  y<="00000010";
         when  "111"=>  y<="00000001";

         when others => y<="ZZZZZZZZ";

      end case;
   end loop;
end process;
end estructura;
```



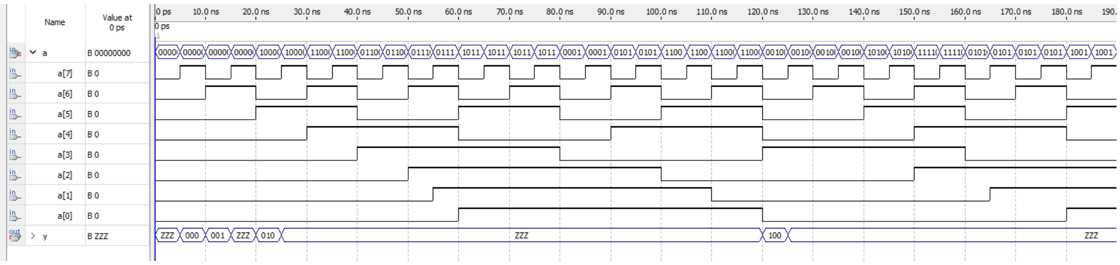
Name	Value at 0 ps
a	B 000
a[2]	B 0
a[1]	B 0
a[0]	B 0
y	B 10000000

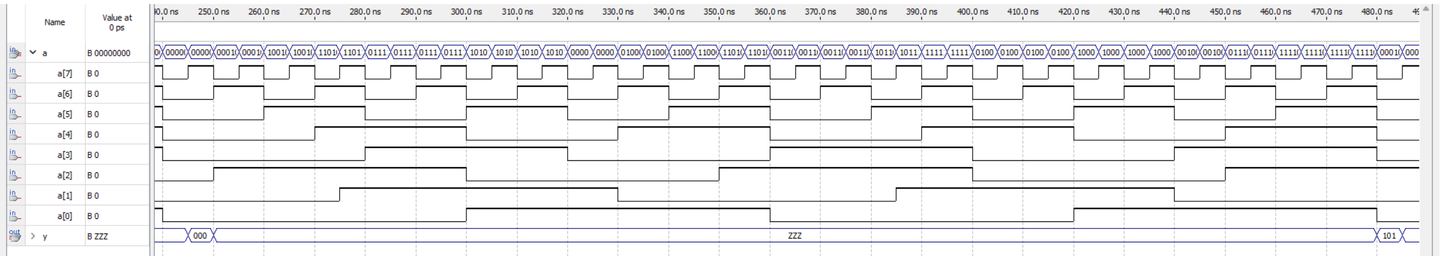
5)

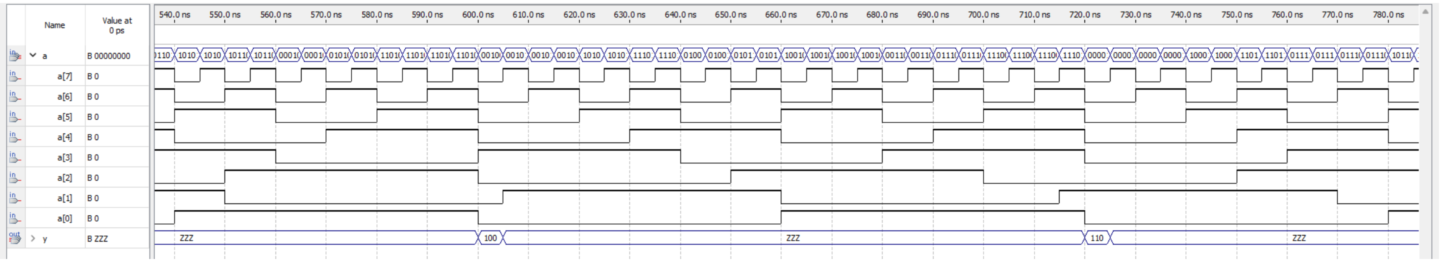
a-

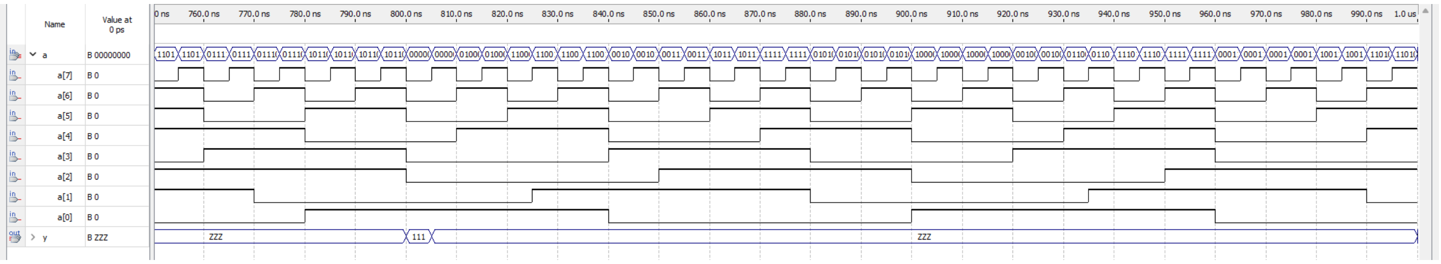


```
library ieee;
use ieee.std_logic_1164.all;

entity codificador_a is
port   (a: in std_logic_vector(7 downto 0);
        y: out std_logic_vector (2 downto 0)
        );
end codificador_a;

architecture estructura of  codificador_a is
begin

      y<="000"  when  a = "10000000" else
         "001"  when  a = "01000000" else
         "010"  when  a = "00100000" else
         "011"  when  a = "00010000" else
         "100"  when  a = "00001000" else
         "101"  when  a = "00000100" else
         "110"  when  a = "00000010" else
         "111"  when  a = "00000001" else

         "ZZZ";


end estructura;
```

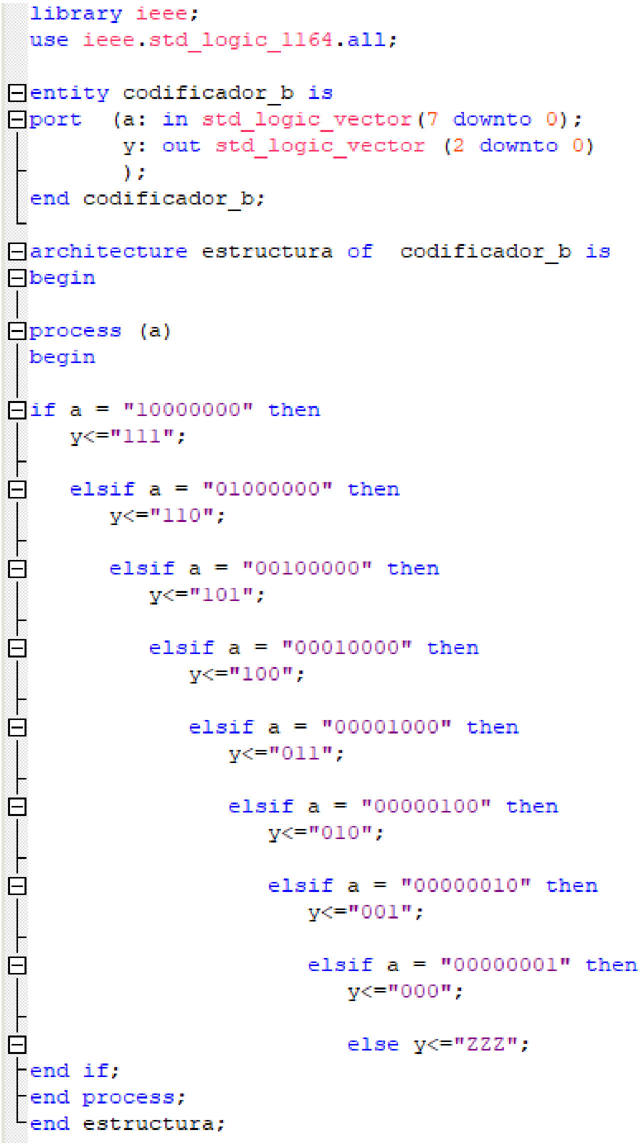




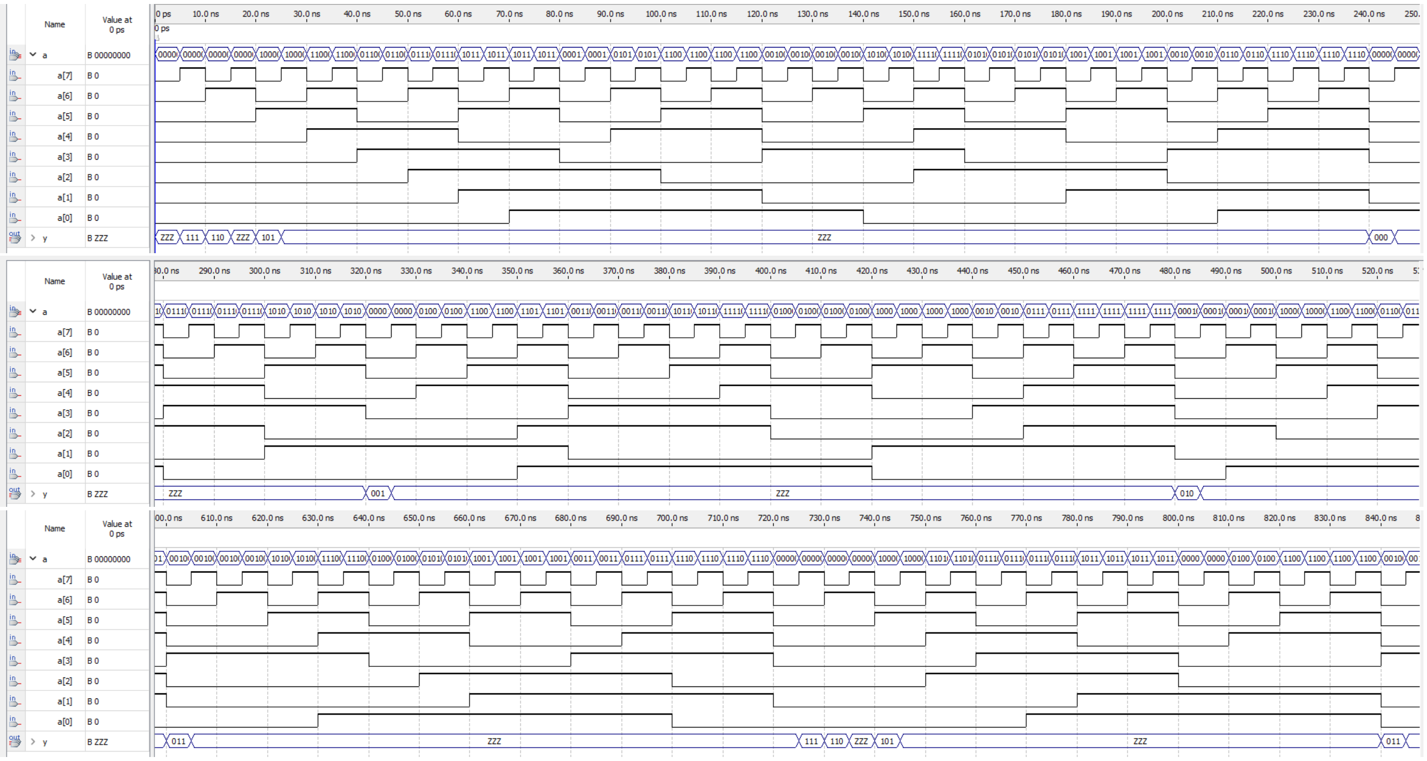




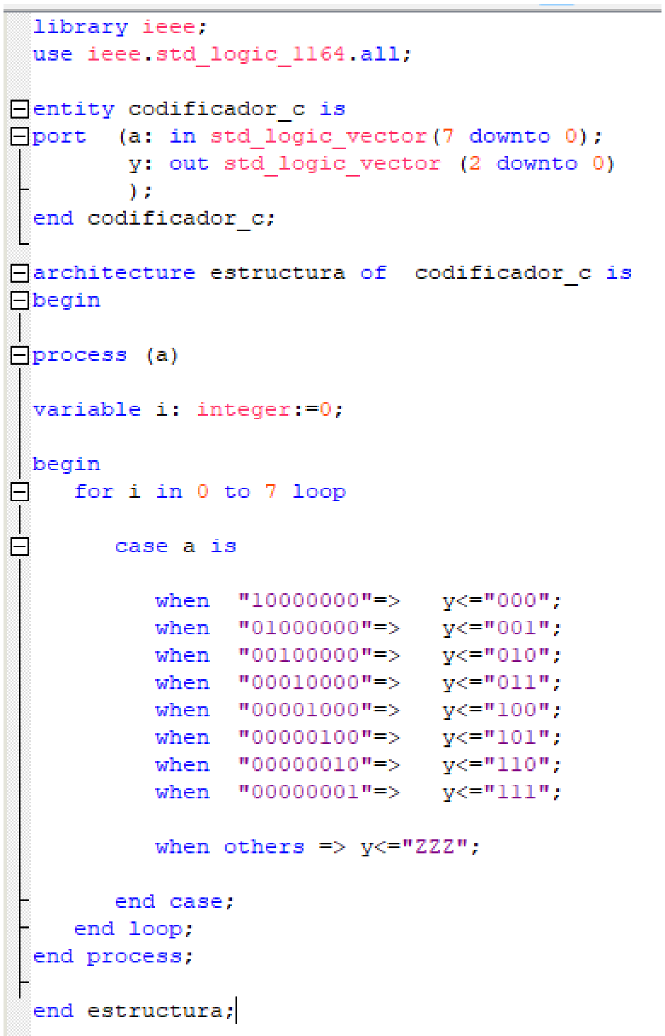
b-



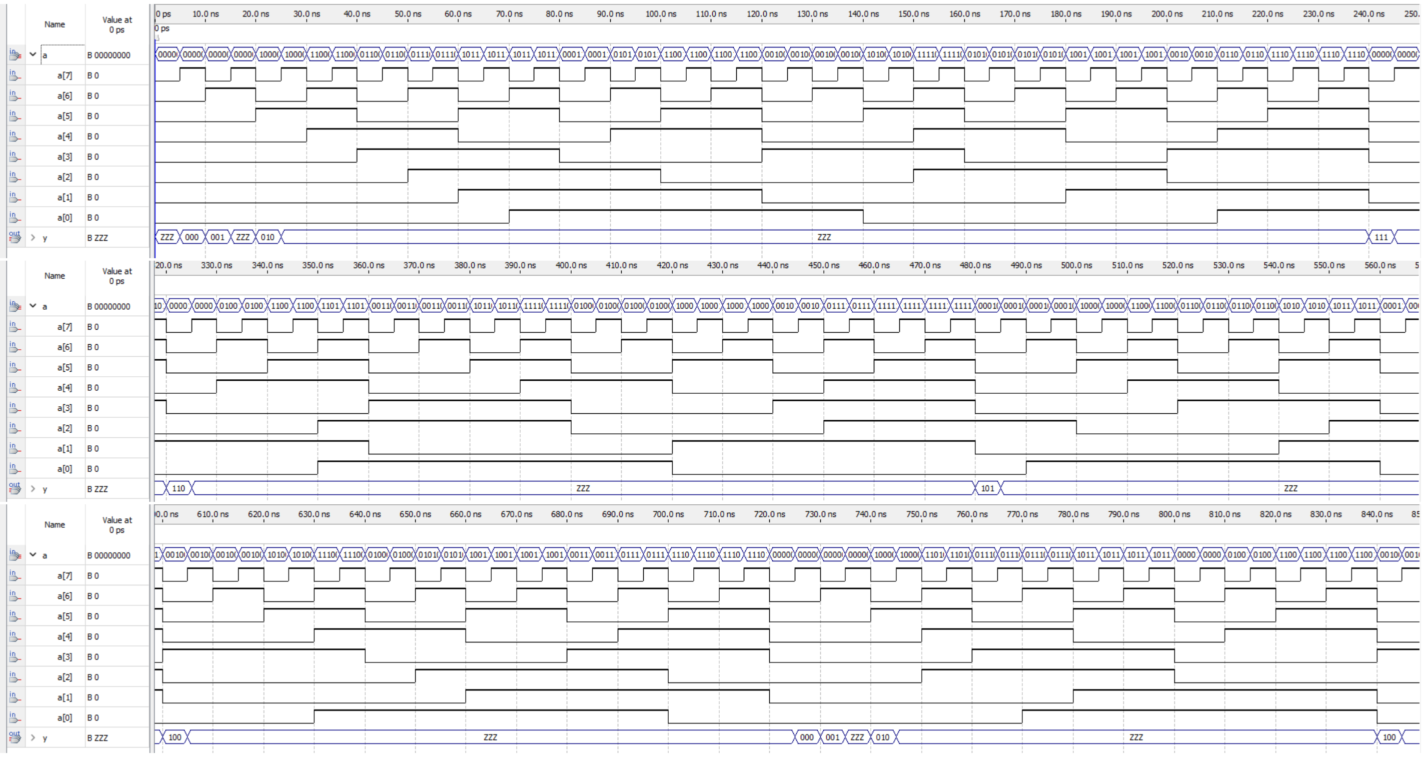
```
library ieee;
use ieee.std_logic_1164.all;

entity codificador_b is
port  (a: in std_logic_vector(7 downto 0);
       y: out std_logic_vector (2 downto 0)
       );
end codificador_b;

architecture estructura of  codificador_b is
begin

process (a)
begin

if a = "10000000" then
   y<="111";

   elsif a = "01000000" then
      y<="110";

      elsif a = "00100000" then
         y<="101";

         elsif a = "00010000" then
            y<="100";

            elsif a = "00001000" then
               y<="011";

               elsif a = "00000100" then
                  y<="010";

                  elsif a = "00000010" then
                     y<="001";

                     elsif a = "00000001" then
                        y<="000";

                        else y<="ZZZ";
end if;
end process;
end estructura;
```



c-

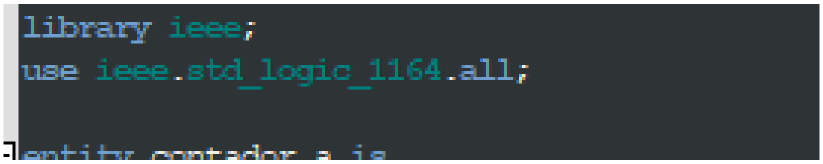


```
library ieee;
use ieee.std_logic_1164.all;

entity codificador_c is
port   (a: in std_logic_vector(7 downto 0);
        y: out std_logic_vector (2 downto 0)
        );
end codificador_c;

architecture estructura of  codificador_c is
begin

process (a)

variable i: integer:=0;

begin
   for i in 0 to 7 loop

      case a is

         when  "10000000"=>   y<="000";
         when  "01000000"=>   y<="001";
         when  "00100000"=>   y<="010";
         when  "00010000"=>   y<="011";
         when  "00001000"=>   y<="100";
         when  "00000100"=>   y<="101";
         when  "00000010"=>   y<="110";
         when  "00000001"=>   y<="111";

         when others => y<="ZZZ";

      end case;
   end loop;
end process;

end estructura;
```

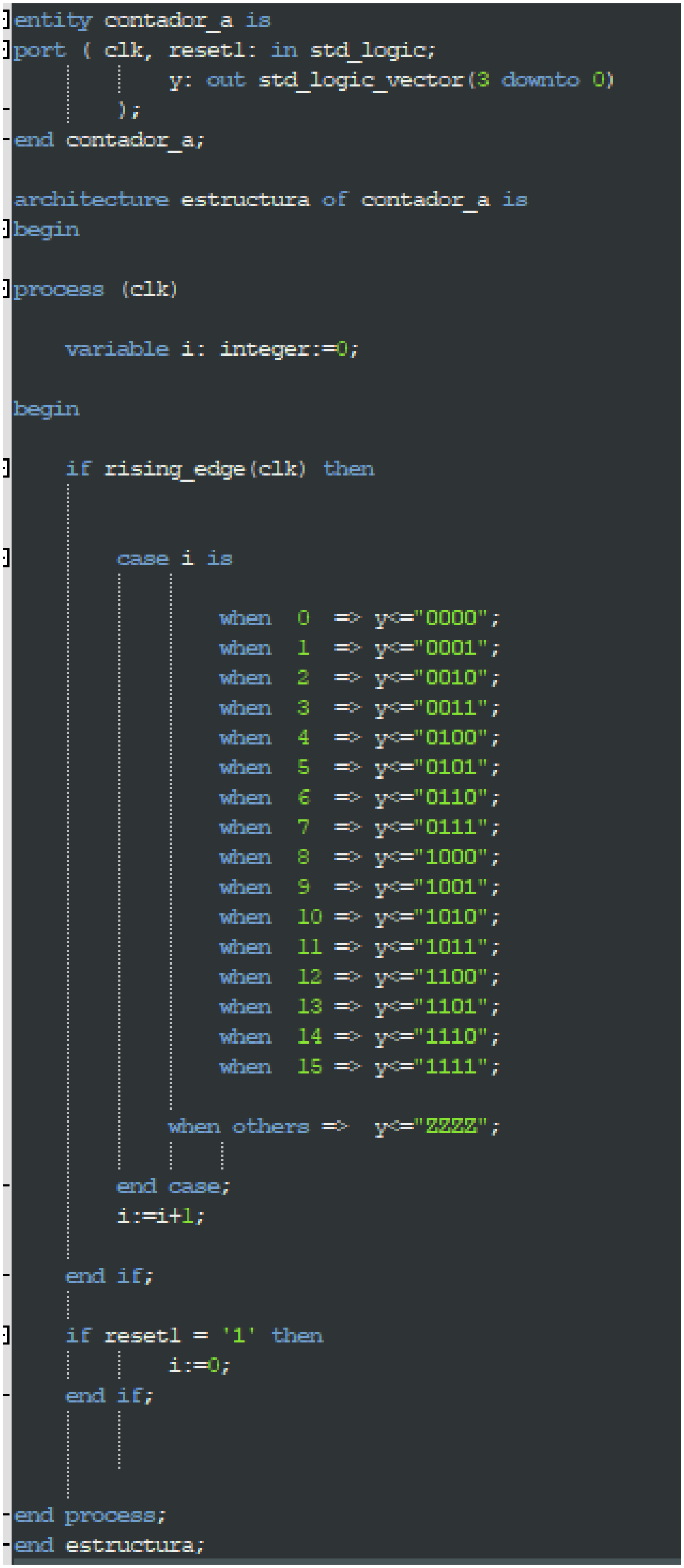


6)

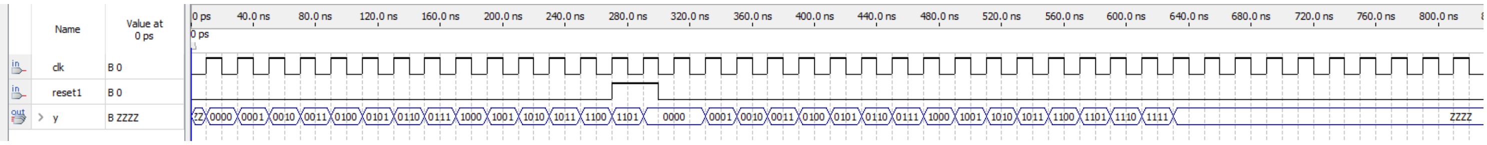
a-



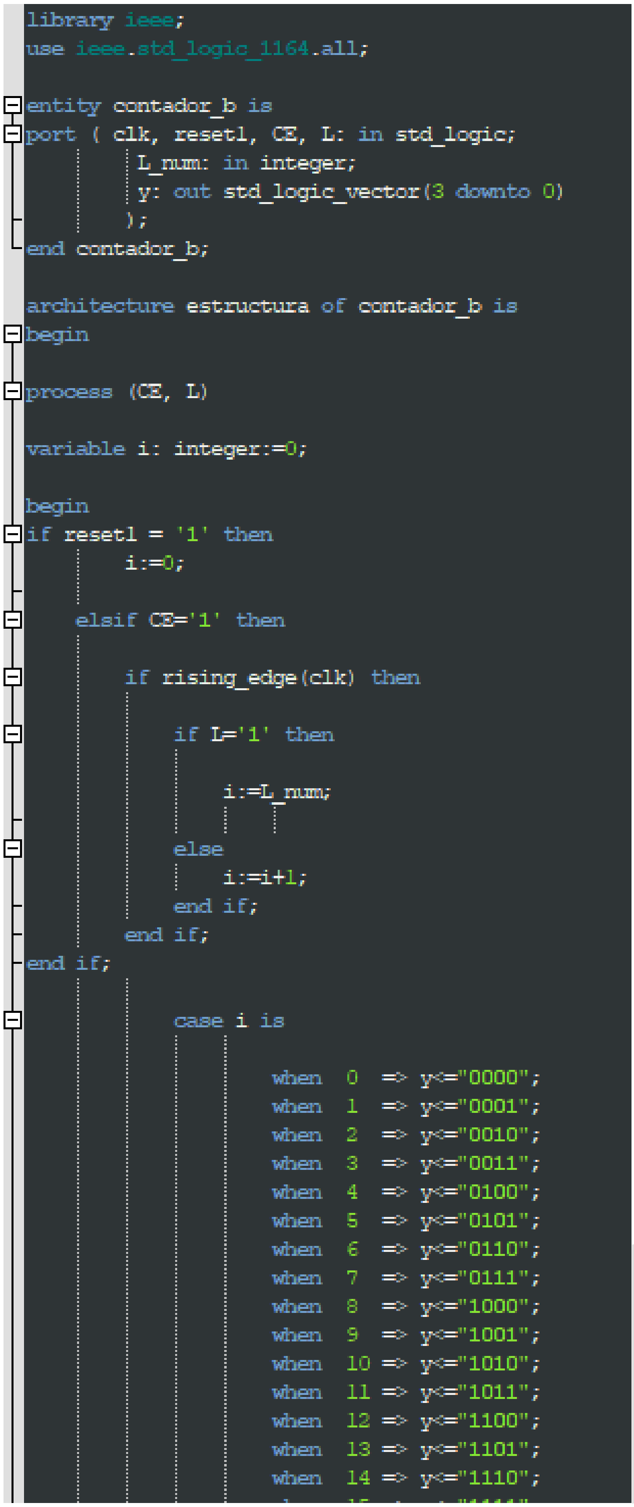
```
library ieee;
use ieee.std_logic_1164.all;


entity contador_a is
```



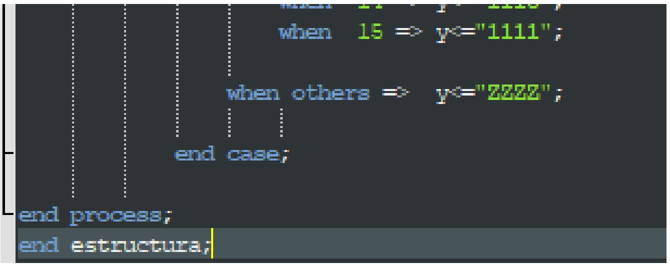
```vhdl
entity contador_a is
port ( clk, resetl: in std_logic;
            y: out std_logic_vector(3 downto 0)
        );
end contador_a;

architecture estructura of contador_a is
begin

process (clk)

    variable i: integer:=0;

begin

    if rising_edge(clk) then


        case i is

                when  0  => y<="0000";
                when  1  => y<="0001";
                when  2  => y<="0010";
                when  3  => y<="0011";
                when  4  => y<="0100";
                when  5  => y<="0101";
                when  6  => y<="0110";
                when  7  => y<="0111";
                when  8  => y<="1000";
                when  9  => y<="1001";
                when  10 => y<="1010";
                when  11 => y<="1011";
                when  12 => y<="1100";
                when  13 => y<="1101";
                when  14 => y<="1110";
                when  15 => y<="1111";

            when others =>  y<="ZZZZ";

        end case;
        i:=i+1;

    end if;

    if resetl = '1' then
            i:=0;
    end if;



end process;
end estructura;
```



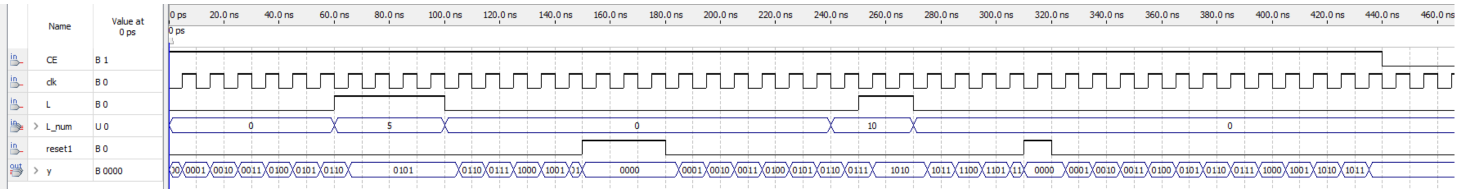
b-



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

entity contador_b is
port ( clk, resetl, CE, L: in std_logic;
         L_num: in integer;
         y: out std_logic_vector(3 downto 0)
        );
end contador_b;

architecture estructura of contador_b is
begin

process (CE, L)

variable i: integer:=0;

begin
if resetl = '1' then
        i:=0;

    elsif CE='1' then

        if rising_edge(clk) then

            if L='1' then

                i:=L_num;

            else
                i:=i+1;
            end if;
        end if;
end if;

            case i is

                when  0  => y<="0000";
                when  1  => y<="0001";
                when  2  => y<="0010";
                when  3  => y<="0011";
                when  4  => y<="0100";
                when  5  => y<="0101";
                when  6  => y<="0110";
                when  7  => y<="0111";
                when  8  => y<="1000";
                when  9  => y<="1001";
                when  10 => y<="1010";
                when  11 => y<="1011";
                when  12 => y<="1100";
                when  13 => y<="1101";
                when  14 => y<="1110";
                when  15 => y<="1111";
```



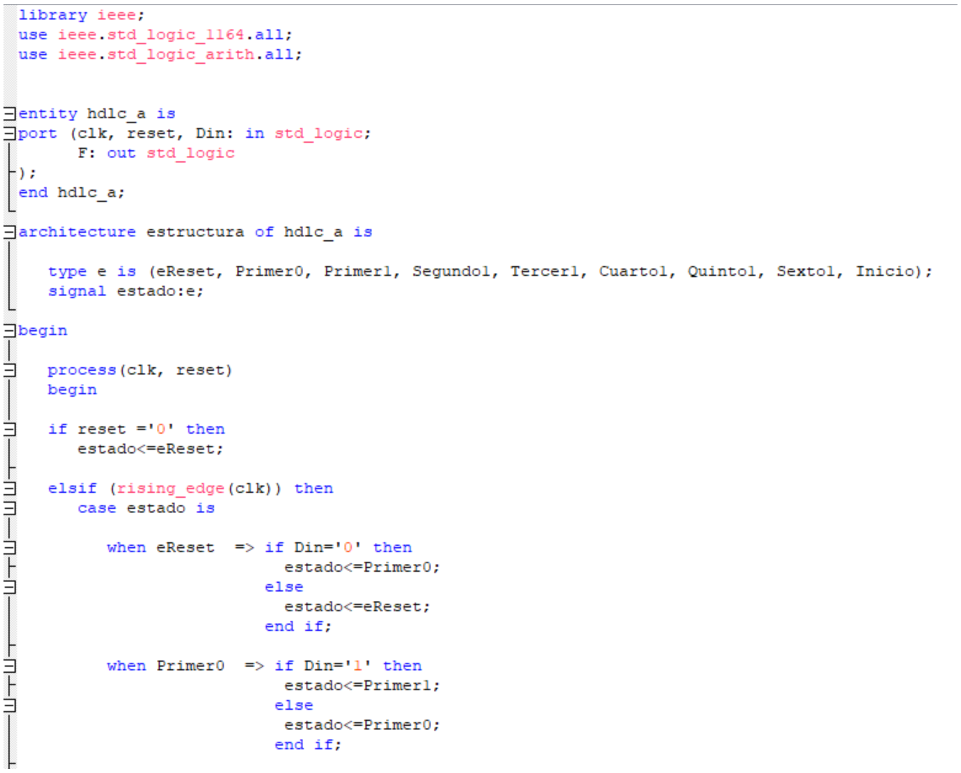
```
                when  15 => y<="1111";

                when others =>  y<="ZZZZ";

            end case;

end process;
end estructura;
```



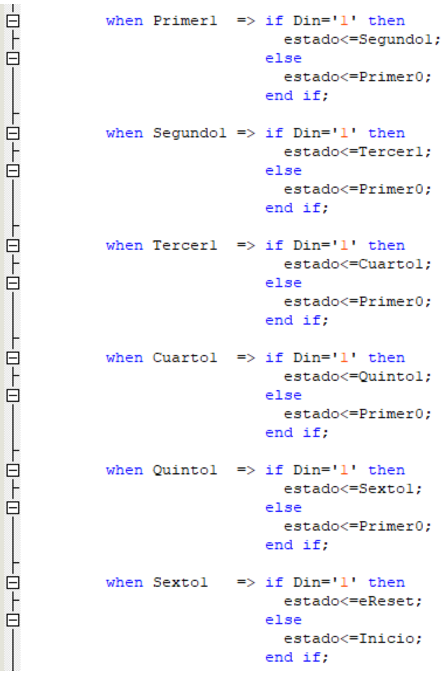
c-

7)

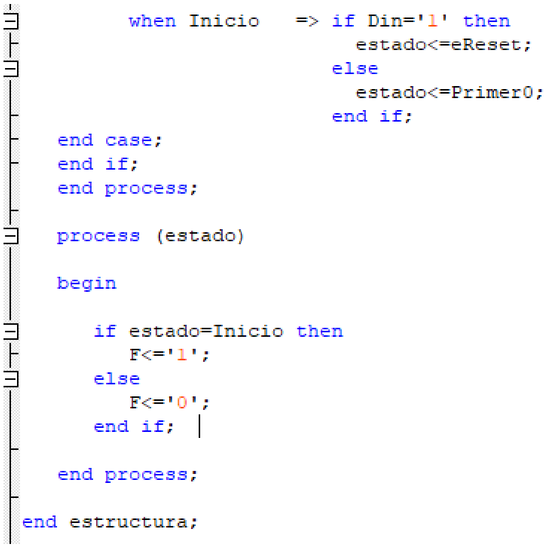
a-



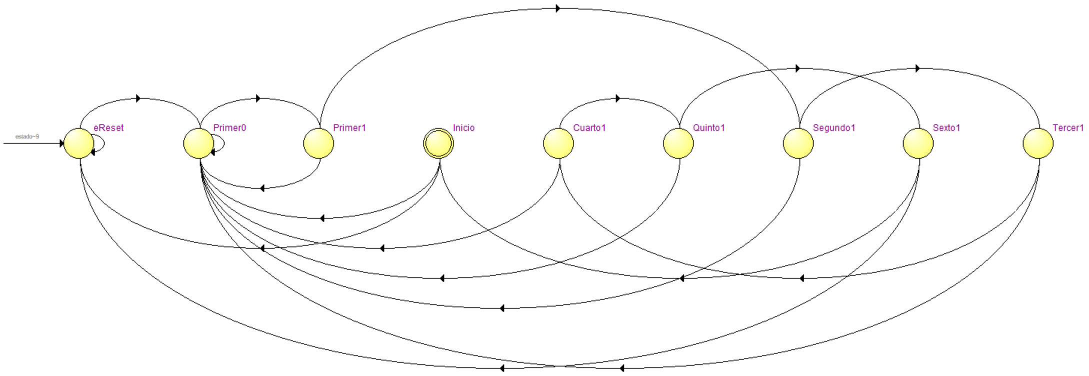
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;


entity hdlc_a is
port (clk, reset, Din: in std_logic;
        F: out std_logic
);
end hdlc_a;

architecture estructura of hdlc_a is

    type e is (eReset, Primer0, Primer1, Segundo1, Tercer1, Cuarto1, Quinto1, Sexto1, Inicio);
    signal estado:e;

begin

    process(clk, reset)
    begin

    if reset ='0' then
        estado<=eReset;

    elsif (rising_edge(clk)) then
        case estado is

            when eReset  => if Din='0' then
                               estado<=Primer0;
                            else
                               estado<=eReset;
                            end if;

            when Primer0  => if Din='1' then
                               estado<=Primer1;
                             else
                               estado<=Primer0;
                             end if;
```

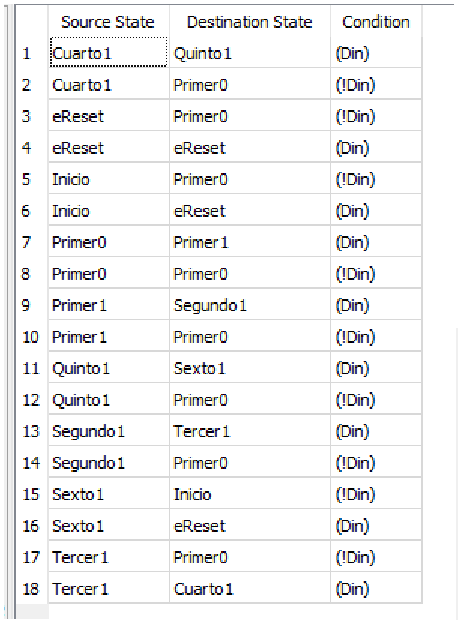


```
            when Primer1  => if Din='1' then
                               estado<=Segundo1;
                            else
                               estado<=Primer0;
                            end if;

            when Segundo1 => if Din='1' then
                               estado<=Tercer1;
                            else
                               estado<=Primer0;
                            end if;

            when Tercer1  => if Din='1' then
                               estado<=Cuarto1;
                            else
                               estado<=Primer0;
                            end if;

            when Cuarto1  => if Din='1' then
                               estado<=Quinto1;
                            else
                               estado<=Primer0;
                            end if;

            when Quinto1  => if Din='1' then
                               estado<=Sexto1;
                            else
                               estado<=Primer0;
                            end if;

            when Sexto1   => if Din='1' then
                               estado<=eReset;
                            else
                               estado<=Inicio;
                            end if;
```

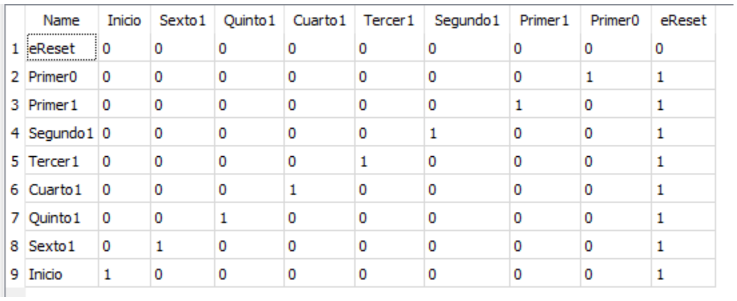


```
         when Inicio   => if Din='1' then
                            estado<=eReset;
                          else
                            estado<=Primer0;
                          end if;
   end case;
   end if;
   end process;

   process  (estado)

   begin

      if estado=Inicio then
         F<='1';
      else
         F<='0';
      end if;

   end process;

end estructura;
```



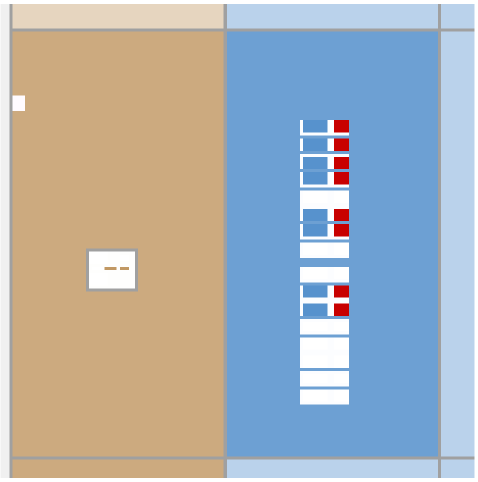


	Source State	Destination State	Condition
1	Cuarto1	Quinto1	(Din)
2	Cuarto1	Primer0	(!Din)
3	eReset	Primer0	(!Din)
4	eReset	eReset	(Din)
5	Inicio	Primer0	(!Din)
6	Inicio	eReset	(Din)
7	Primer0	Primer1	(Din)
8	Primer0	Primer0	(!Din)
9	Primer1	Segundo1	(Din)
10	Primer1	Primer0	(!Din)
11	Quinto1	Sexto1	(Din)
12	Quinto1	Primer0	(!Din)
13	Segundo1	Tercer1	(Din)
14	Segundo1	Primer0	(!Din)
15	Sexto1	Inicio	(!Din)
16	Sexto1	eReset	(Din)
17	Tercer1	Primer0	(!Din)
18	Tercer1	Cuarto1	(Din)



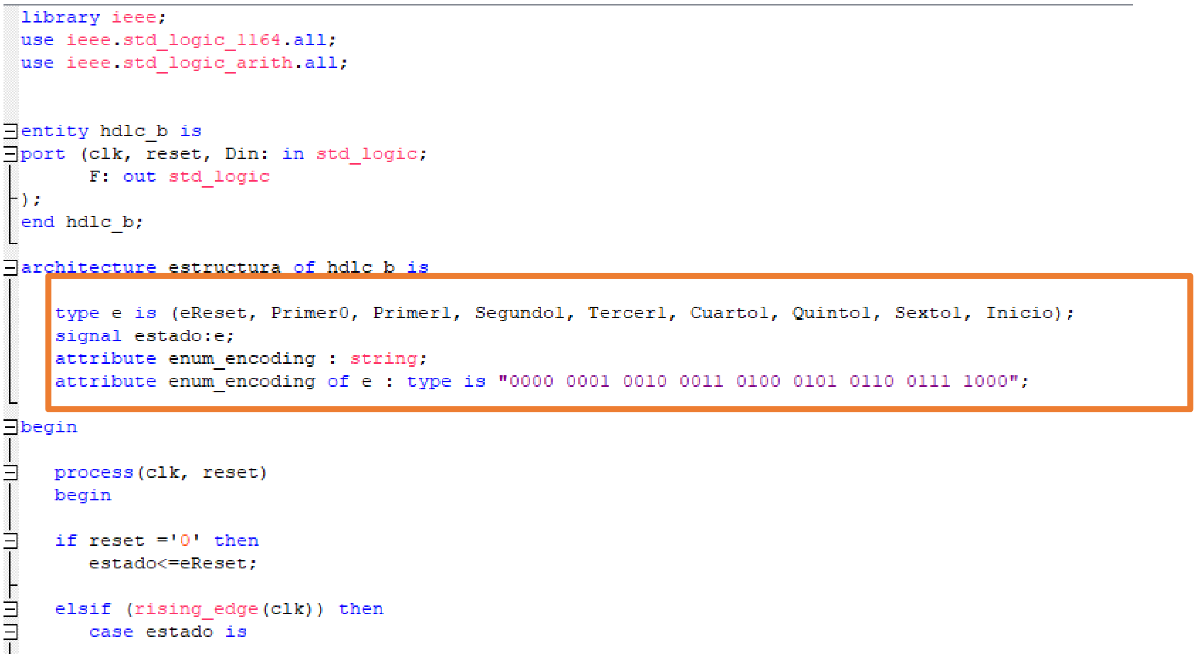
	Name	Inicio	Sexto1	Quinto1	Cuarto1	Tercer1	Segundo1	Primer1	Primer0	eReset
1	eReset	0	0	0	0	0	0	0	0	0
2	Primer0	0	0	0	0	0	0	0	1	1
3	Primer1	0	0	0	0	0	0	1	0	1
4	Segundo1	0	0	0	0	0	1	0	0	1
5	Tercer1	0	0	0	0	1	0	0	0	1
6	Cuarto1	0	0	0	1	0	0	0	0	1
7	Quinto1	0	0	1	0	0	0	0	0	1
8	Sexto1	0	1	0	0	0	0	0	0	1
9	Inicio	1	0	0	0	0	0	0	0	1

Elementos lógicos: 8

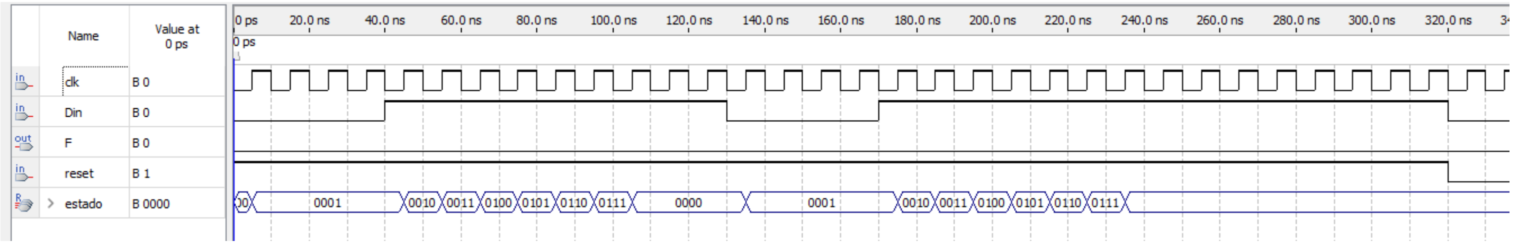


b-

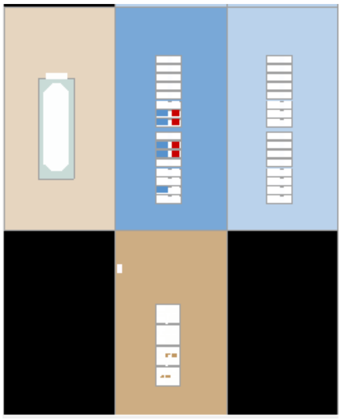
Copio solo la descripción de hardware que se incluyó a la descripción anterior.



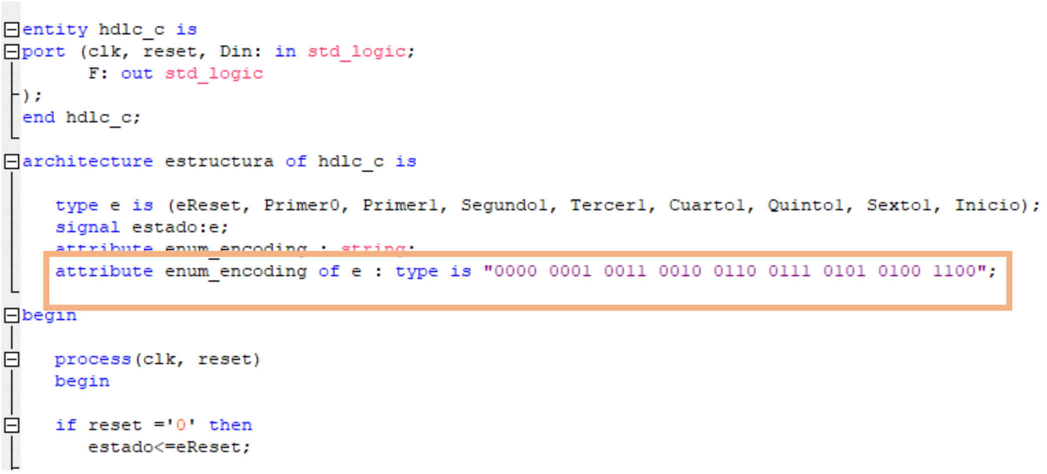
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;


entity hdlc_b is
port (clk, reset, Din: in std_logic;
      F: out std_logic
);
end hdlc_b;

architecture estructura of hdlc_b is

   type e is (eReset, Primer0, Primer1, Segundo1, Tercer1, Cuarto1, Quinto1, Sexto1, Inicio);
   signal estado:e;
   attribute enum_encoding : string;
   attribute enum_encoding of e : type is "0000 0001 0010 0011 0100 0101 0110 0111 1000";
begin

   process(clk, reset)
   begin

   if reset ='0' then
      estado<=eReset;

   elsif (rising_edge(clk)) then
      case estado is
```



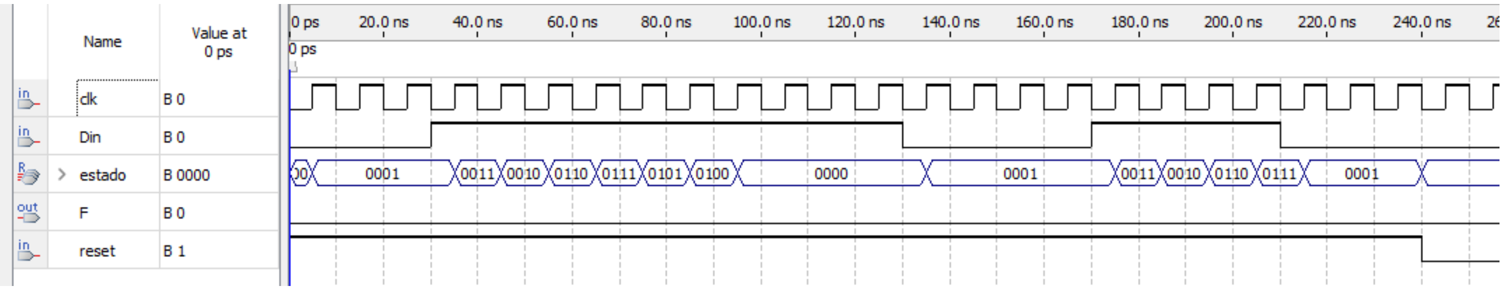
Elementos lógicos: 5



c-

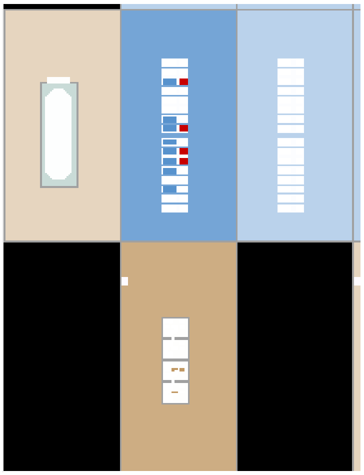


```
entity hdlc_c is
port (clk, reset, Din: in std_logic;
        F: out std_logic
);
end hdlc_c;

architecture estructura of hdlc_c is

    type e is (eReset, Primer0, Primer1, Segundo1, Tercer1, Cuarto1, Quinto1, Sexto1, Inicio);
    signal estado:e;
    attribute enum_encoding : string;
    attribute enum_encoding of e : type is "0000 0001 0011 0010 0110 0111 0101 0100 1100";

begin

    process(clk, reset)
    begin

    if reset ='0' then
        estado<=eReset;
```

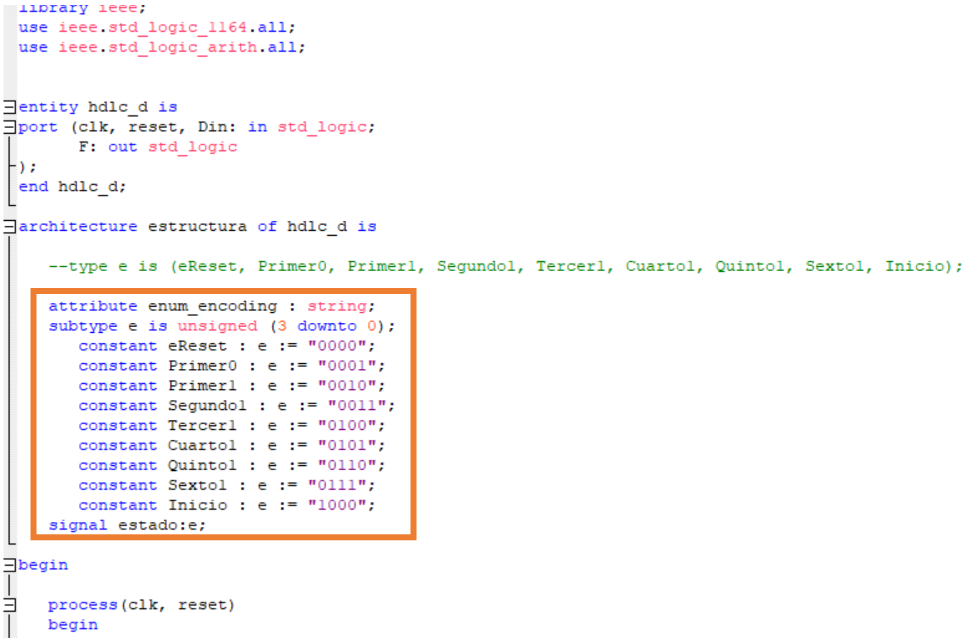


Name	Value at 0 ps
clk	B 0
Din	B 0
estado	B 0000
F	B 0
reset	B 1

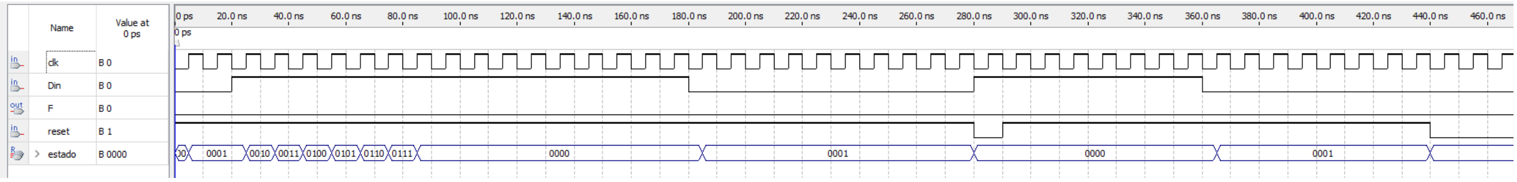
Elementos lógicos: 8



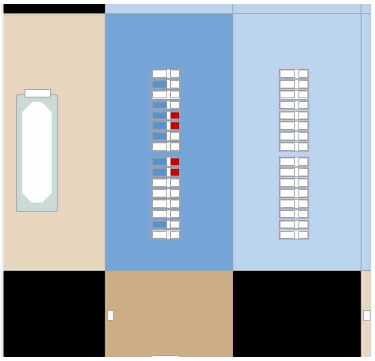
d-



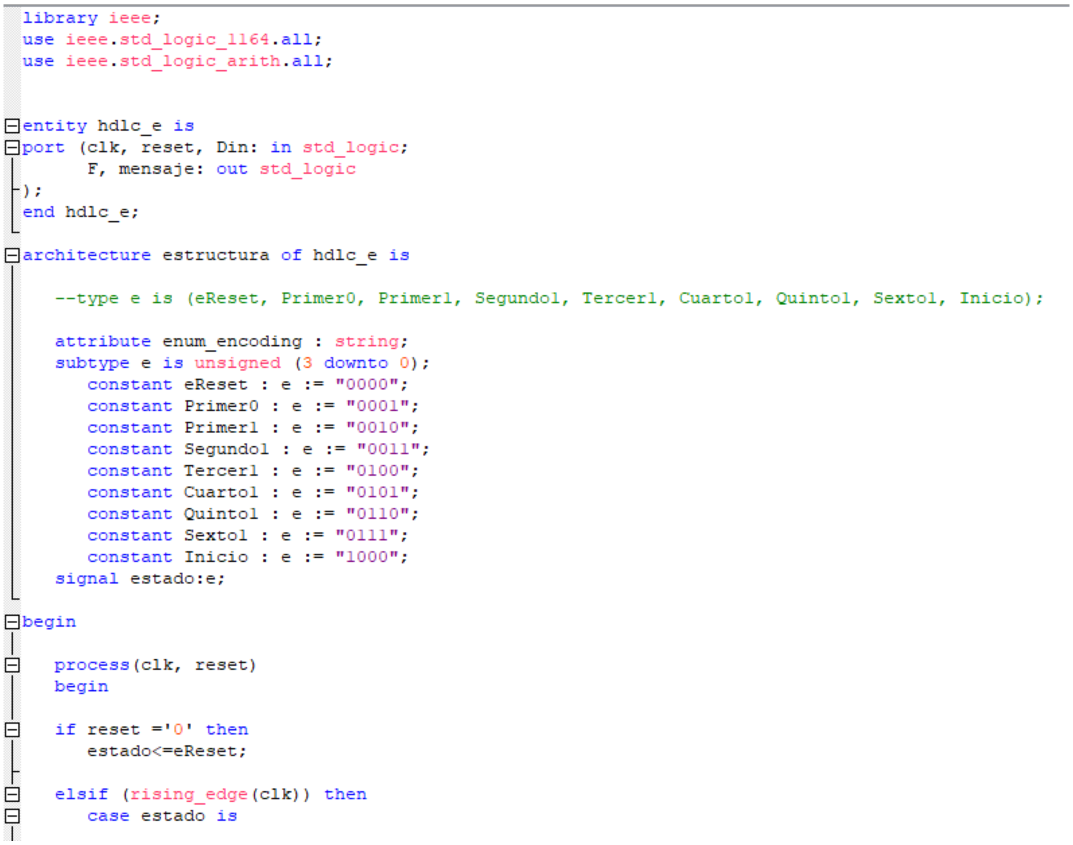
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;


entity hdlc_d is
port (clk, reset, Din: in std_logic;
      F: out std_logic
);
end hdlc_d;

architecture estructura of hdlc_d is

   --type e is (eReset, Primer0, Primer1, Segundo1, Tercer1, Cuarto1, Quinto1, Sexto1, Inicio);

   attribute enum_encoding : string;
   subtype e is unsigned (3 downto 0);
      constant eReset : e := "0000";
      constant Primer0 : e := "0001";
      constant Primer1 : e := "0010";
      constant Segundo1 : e := "0011";
      constant Tercer1 : e := "0100";
      constant Cuarto1 : e := "0101";
      constant Quinto1 : e := "0110";
      constant Sexto1 : e := "0111";
      constant Inicio : e := "1000";
   signal estado:e;

begin

   process(clk, reset)
   begin
```



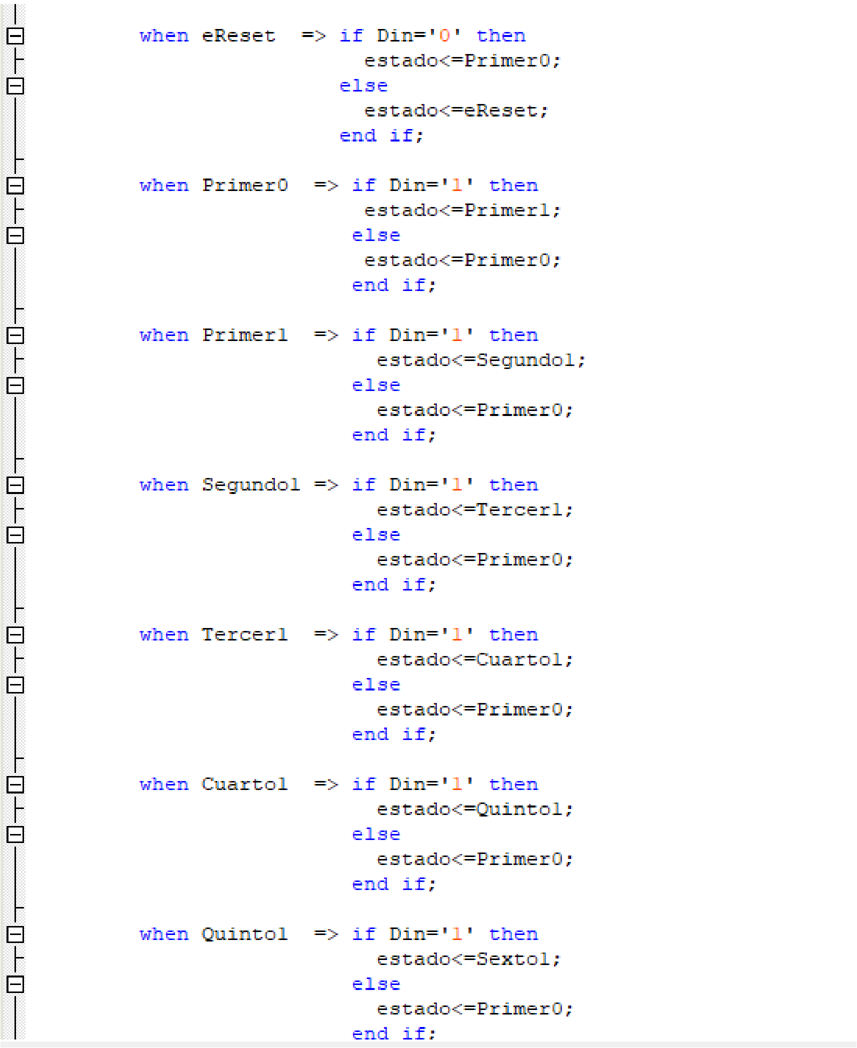
Elementos lógicos: 8



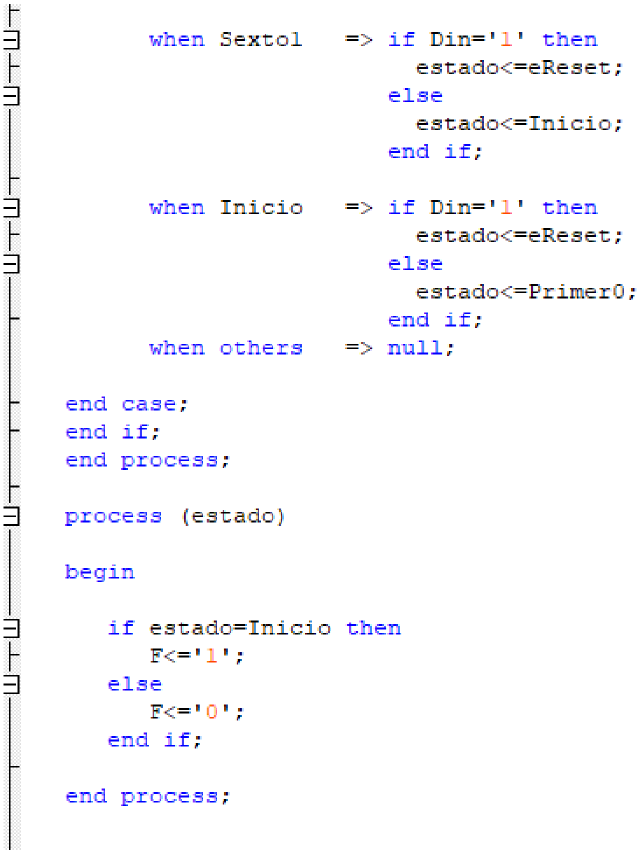
e-



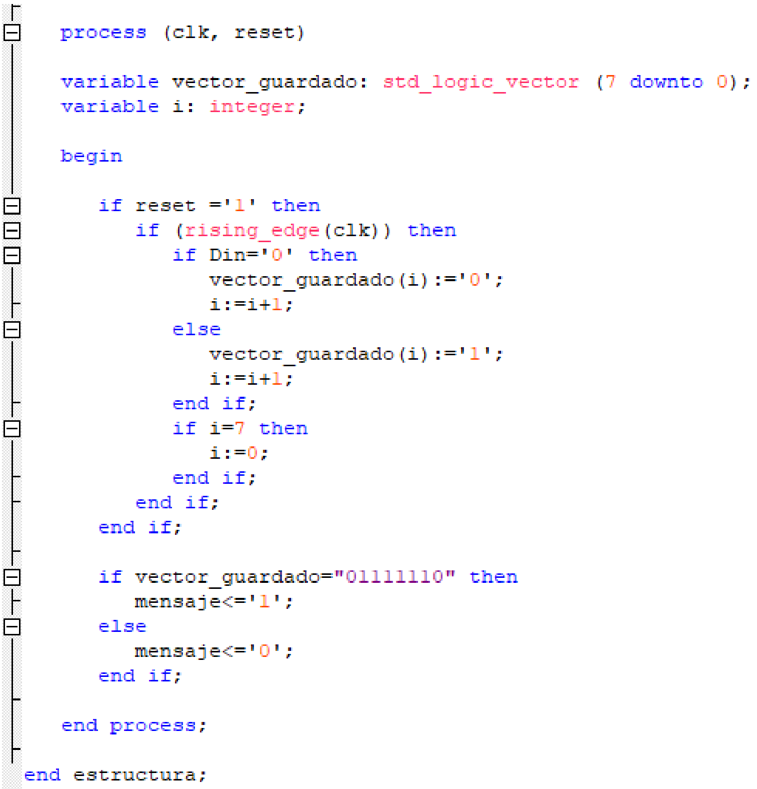
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;


entity hdlc_e is
port (clk, reset, Din: in std_logic;
      F, mensaje: out std_logic
);
end hdlc_e;

architecture estructura of hdlc_e is

   --type e is (eReset, Primer0, Primer1, Segundo1, Tercer1, Cuarto1, Quinto1, Sexto1, Inicio);

   attribute enum_encoding : string;
   subtype e is unsigned (3 downto 0);
      constant eReset : e := "0000";
      constant Primer0 : e := "0001";
      constant Primer1 : e := "0010";
      constant Segundo1 : e := "0011";
      constant Tercer1 : e := "0100";
      constant Cuarto1 : e := "0101";
      constant Quinto1 : e := "0110";
      constant Sexto1 : e := "0111";
      constant Inicio : e := "1000";
   signal estado:e;

begin

   process(clk, reset)
   begin

   if reset ='0' then
      estado<=eReset;

   elsif (rising_edge(clk)) then
      case estado is
```



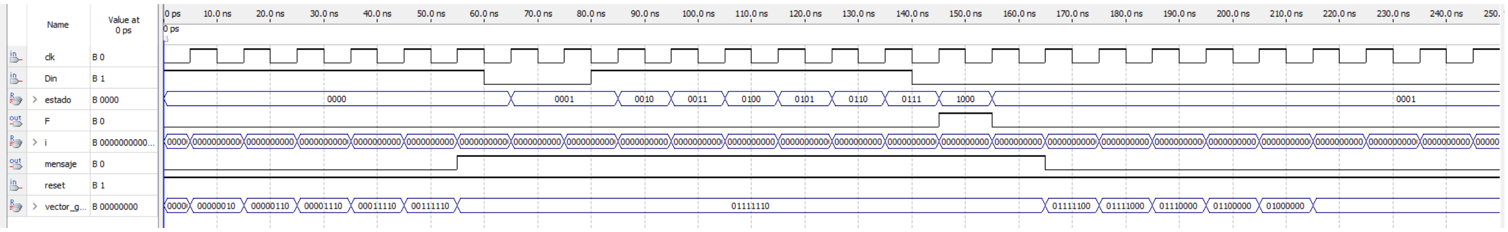
```

         when eReset  => if Din='0' then
                           estado<=Primer0;
                         else
                           estado<=eReset;
                         end if;

         when Primer0  => if Din='1' then
                           estado<=Primer1;
                          else
                           estado<=Primer0;
                          end if;

         when Primer1  => if Din='1' then
                            estado<=Segundo1;
                          else
                            estado<=Primer0;
                          end if;

         when Segundo1 => if Din='1' then
                            estado<=Tercer1;
                          else
                            estado<=Primer0;
                          end if;

         when Tercer1  => if Din='1' then
                            estado<=Cuarto1;
                          else
                            estado<=Primer0;
                          end if;

         when Cuarto1  => if Din='1' then
                            estado<=Quinto1;
                          else
                            estado<=Primer0;
                          end if;

         when Quinto1  => if Din='1' then
                            estado<=Sexto1;
                          else
                            estado<=Primer0;
                          end if;
```

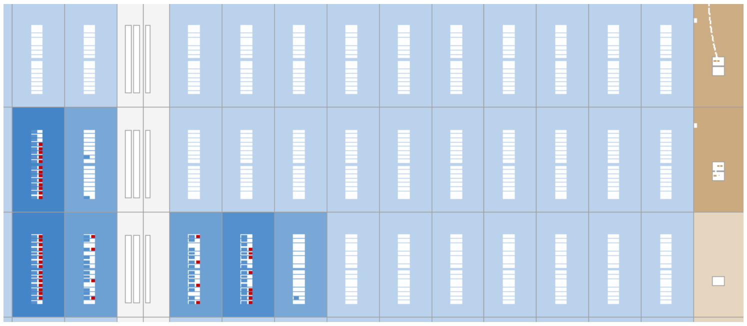


```
      when Sextol    => if Din='1' then
                          estado<=eReset;
                        else
                          estado<=Inicio;
                        end if;

      when Inicio    => if Din='1' then
                          estado<=eReset;
                        else
                          estado<=Primer0;
                        end if;
      when others    => null;

end case;
end if;
end process;

process (estado)

begin

   if estado=Inicio then
      F<='1';
   else
      F<='0';
   end if;

end process;
```



```
   process (clk, reset)

   variable vector_guardado: std_logic_vector (7 downto 0);
   variable i: integer;

   begin

      if reset ='1' then
         if (rising_edge(clk)) then
            if Din='0' then
               vector_guardado(i):='0';
               i:=i+1;
            else
               vector_guardado(i):='1';
               i:=i+1;
            end if;
            if i=7 then
               i:=0;
            end if;
         end if;
      end if;

      if vector_guardado="01111110" then
         mensaje<='1';
      else
         mensaje<='0';
      end if;

   end process;

end estructura;
```





Elementos lógicos: 77