



Universidad Tecnológica de la Mixteca

Cómputo Reconfigurable

Profesor: M.C. Arturo Pablo Sandoval García

Tercer Parcial

Ingeniería en Computación || Grupo: 402-B

17 de junio de 2023

Integrantes:

Elorza Velásquez Jorge Aurelio
González Martínez Ruben Eduardo
Martínez Lorenzo Frida Ximena
Robles Jimenez José Guadalupe

- Nota: Implementar en ISE DESING a nivel de VHDL y descarga en la tarjeta Nexys 2.

Actividad 1.- SELECTOR DE FUNCIONES EN VHDL.

- Representar, simular e implementar en ISE-Design el siguiente circuito considerando que (A,B,C) son entradas del módulo de **funciones** (Y_0, Y_1, Y_2, Y_3) y (D0, D1) corresponde a las entradas del decodificador de 2 a 4 donde la línea (D0) corresponde al LSB. La implementación del circuito será de manera combinada en un archivo las compuertas **en esquemático y el decodificador y el módulo de funciones (Y_0, Y_1, Y_2, Y_3) en un módulo en VHDL**.

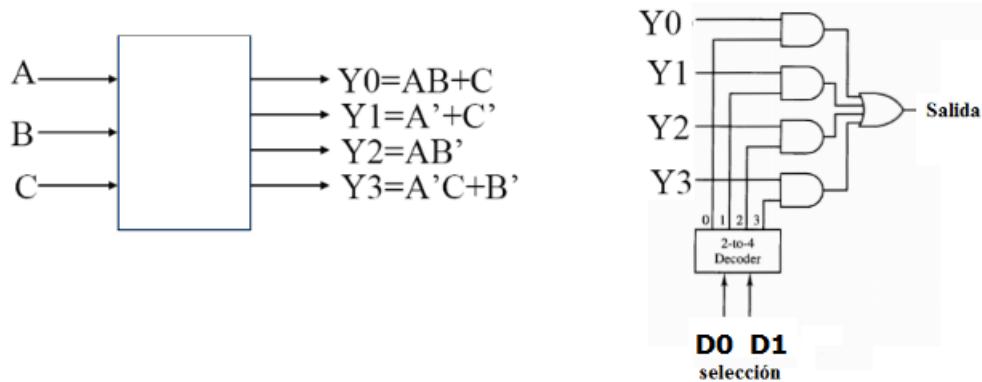
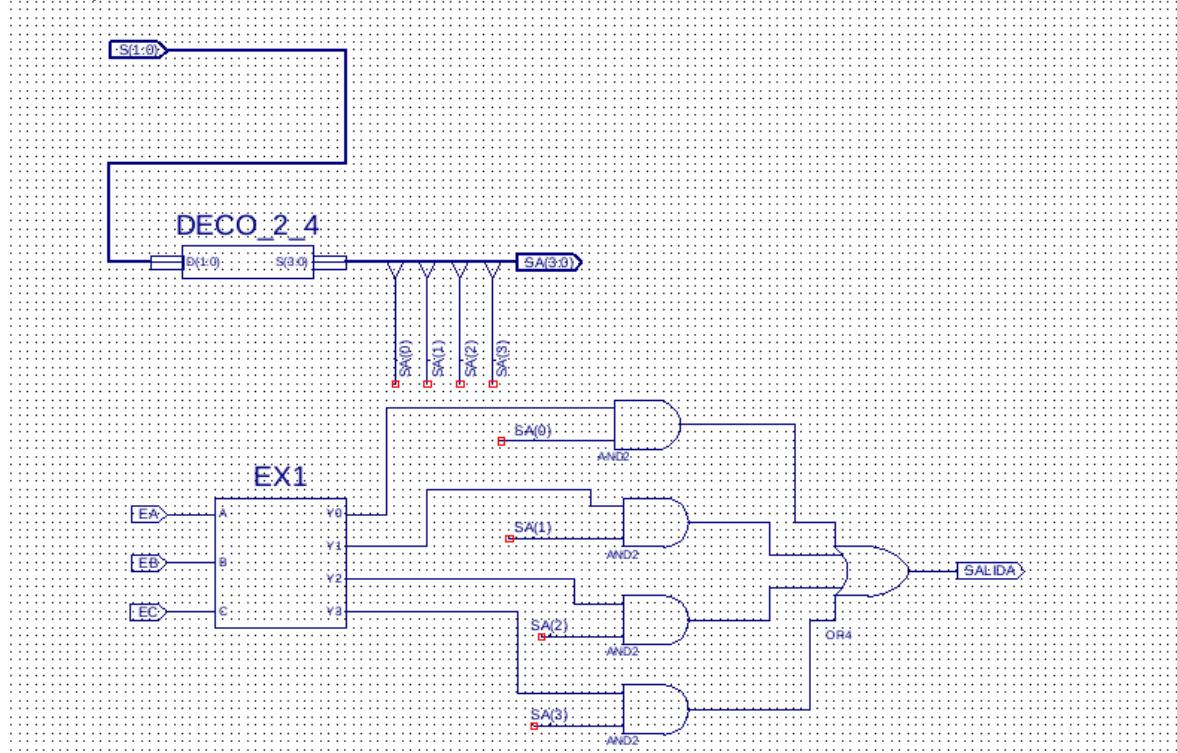
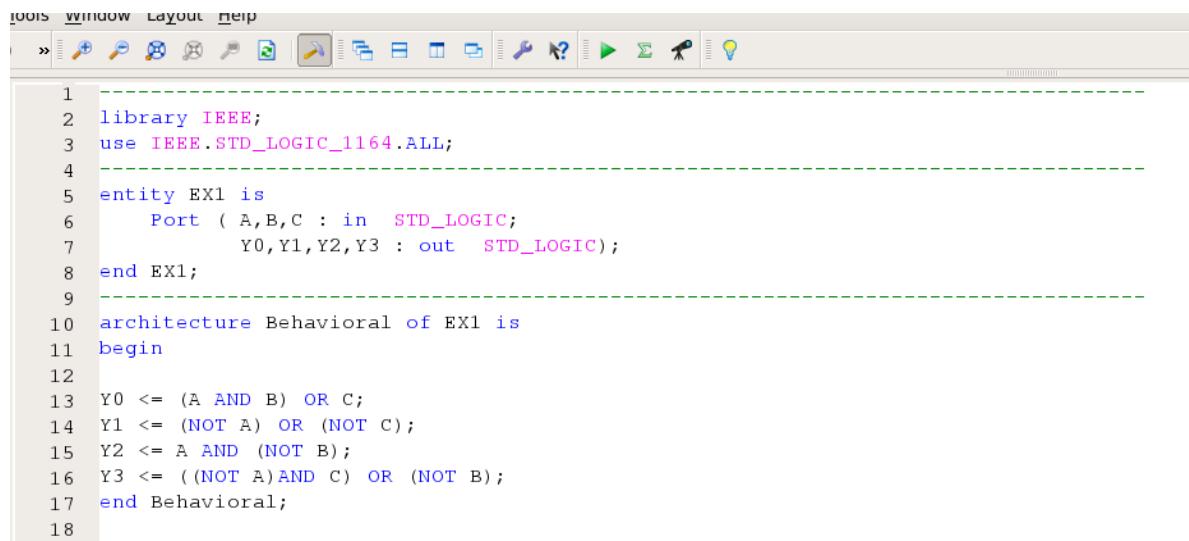


Figura 1

- Adicionar los códigos utilizados en VHDL
- Representar la simulación.

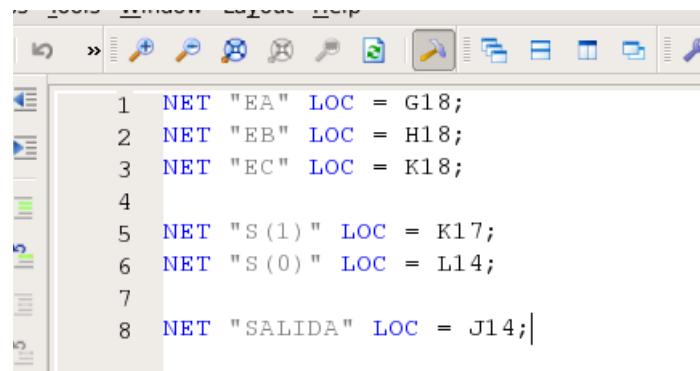


```
1
2 library IEEE;
3 use IEEE.STD_LOGIC_1164.ALL;
4 -----
5 entity DECO_2_4 is
6     Port ( D : in STD_LOGIC_VECTOR (1 downto 0);
7             S : out STD_LOGIC_VECTOR (3 downto 0));
8 end DECO_2_4;
9 -----
10 architecture Behavioral of DECO_2_4 is
11 begin
12
13 WITH D SELECT
14 S <= "0001" WHEN "00",
15         "0010" WHEN "01",
16         "0100" WHEN "10",
17         "1000" WHEN OTHERS;
18
19 end Behavioral;
20
```



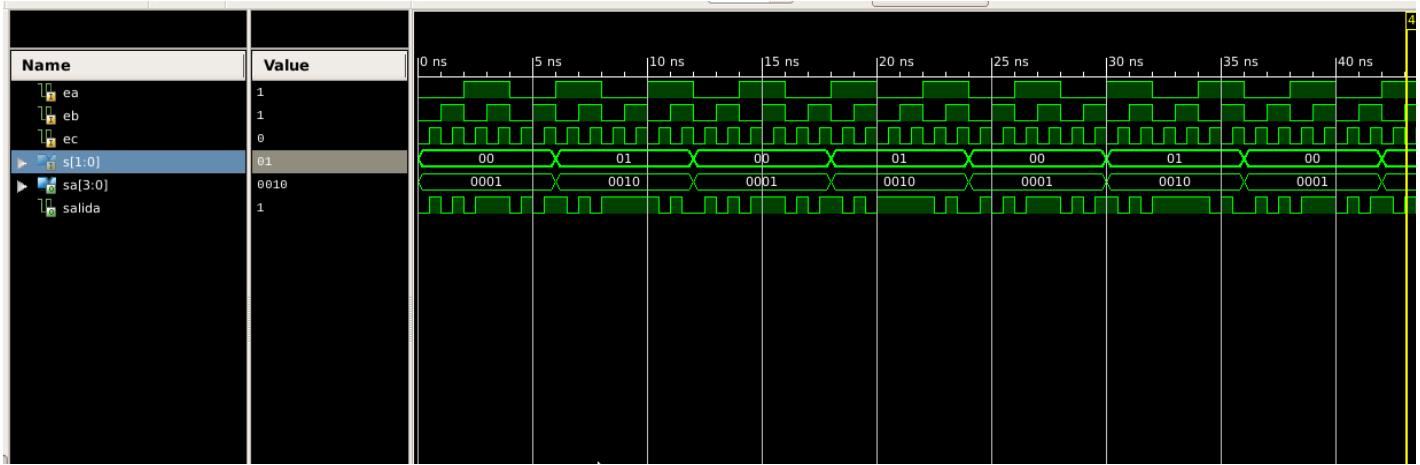
The screenshot shows a VHDL editor interface with a menu bar (Tools, Window, Layout, Help) and a toolbar with various icons. The code area displays the following VHDL code:

```
1
2 library IEEE;
3 use IEEE.STD_LOGIC_1164.ALL;
4 -----
5 entity EX1 is
6     Port ( A,B,C : in STD_LOGIC;
7             Y0,Y1,Y2,Y3 : out STD_LOGIC);
8 end EX1;
9 -----
10 architecture Behavioral of EX1 is
11 begin
12
13 Y0 <= (A AND B) OR C;
14 Y1 <= (NOT A) OR (NOT C);
15 Y2 <= A AND (NOT B);
16 Y3 <= ((NOT A)AND C) OR (NOT B);
17
18 end Behavioral;
```



The screenshot shows a logic editor interface with a toolbar and a main window displaying a netlist. The netlist contains the following connections:

```
1 NET "EA" LOC = G18;
2 NET "EB" LOC = H18;
3 NET "EC" LOC = K18;
4
5 NET "S(1)" LOC = K17;
6 NET "S(0)" LOC = L14;
7
8 NET "SALIDA" LOC = J14;
```



Actividad 2.- ALU (Unidad lógica Aritmética)

Utilizando ISE DESIGN en VHDL y la tarjeta NEXYS 2. Represente una ALU, la cual contiene dos números de dos entradas de 2 bits, dos líneas de selección y 4 bits de salida. Las operaciones a realizar son : AND, OR , suma, multiplicación .. Esté circuito debe aceptar dos números decimales (a y b). Cada uno de los números decimales está limitado a los valores (0, 1, 2, 3). El circuito realizará la multiplicación ($Res = a * b$) y la suma ($Res = a+b$) adicionalmente las operaciones booleanas AND, OR. El resultado de la salida se visualiza en un display de la tarjeta.

- Los datos de salida se visualizarán en los leds de la tarjeta Nexys 2.
- Adicionar el códigos utilizados en VHDL
- Representar la simulación.

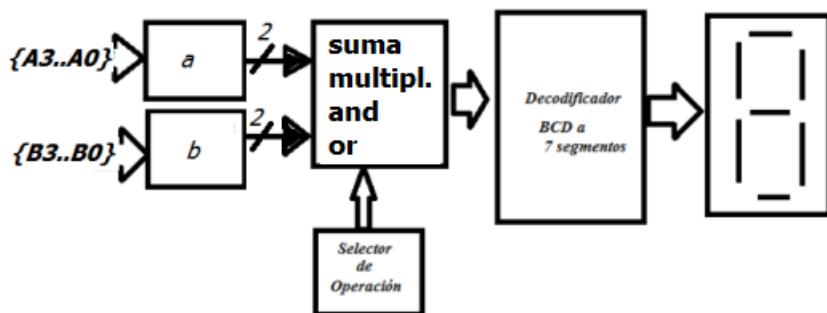
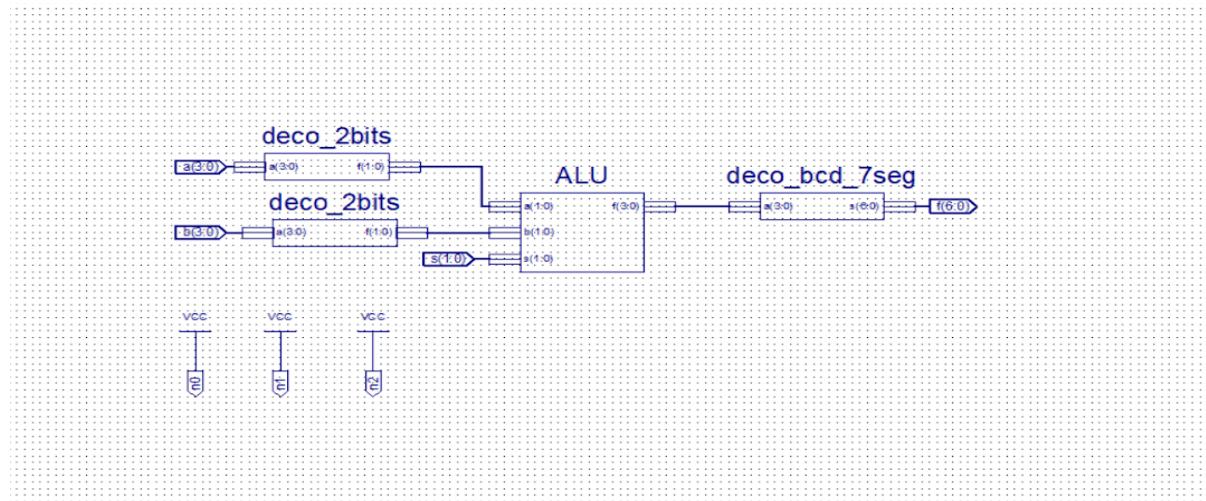


Figura 2. Representación en bloques de la operación de suma y multiplicación

- Adicionar los códigos utilizados en VHDL
- Representar la simulación.

Implementación en forma esquemática en ISE DESIGN:



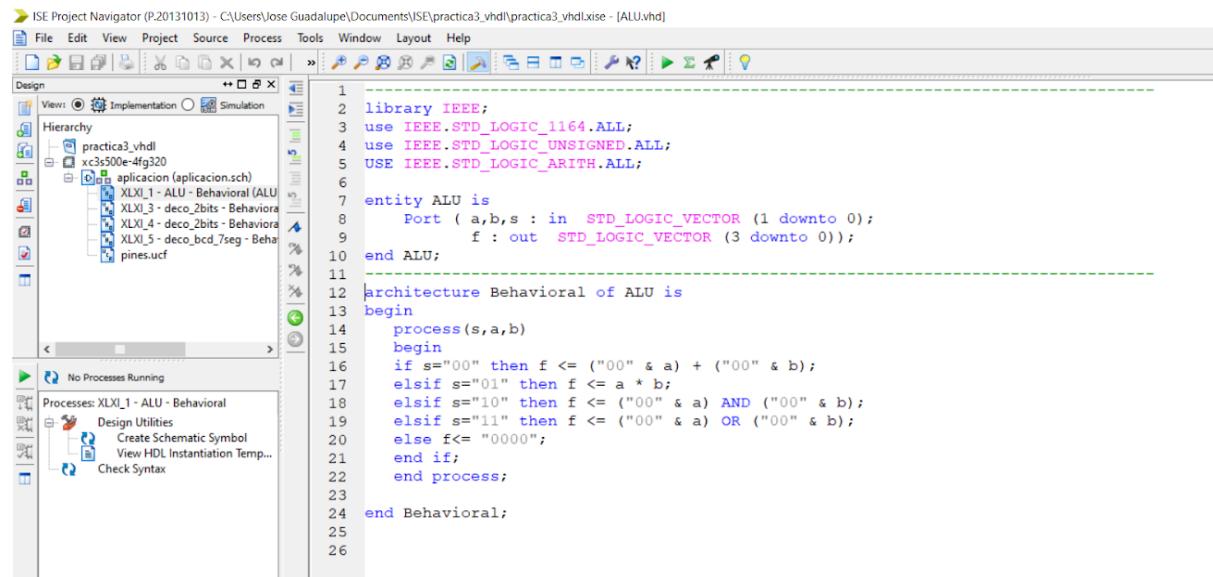
Código en VHDL del decodificador (cajitas pequeñas) :

The screenshot shows the ISE Design Software interface. The left pane displays the project hierarchy under 'Design' with files like 'practica3.vhdll', 'xc3s500e-4fg320', and 'aplicacion.aplicacion.sch'. The right pane shows the VHDL code for the 'deco_2bits' entity:

```
1 library IEEE;
2 use IEEE.STD_LOGIC_1164.ALL;
3
4 entity deco_2bits is
5     Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
6             f : out STD_LOGIC_VECTOR (1 downto 0));
7 end deco_2bits;
8
9 -----
10
11 architecture Behavioral of deco_2bits is
12
13 begin
14     process (a)
15     begin
16         if a="0001" then f <="00";
17         elsif a="0010" then f <="01";
18         elsif a="0100" then f <="10";
19         else f <="11";
20     end if;
21     end process;
22 end Behavioral;
```

The bottom left pane shows a context menu for the selected component 'XLXI_3 - deco_2bits - Behavioral' with options like 'Design Utilities', 'Create Schematic Symbol', 'View HDL Instantiation Temp...', and 'Check Syntax'.

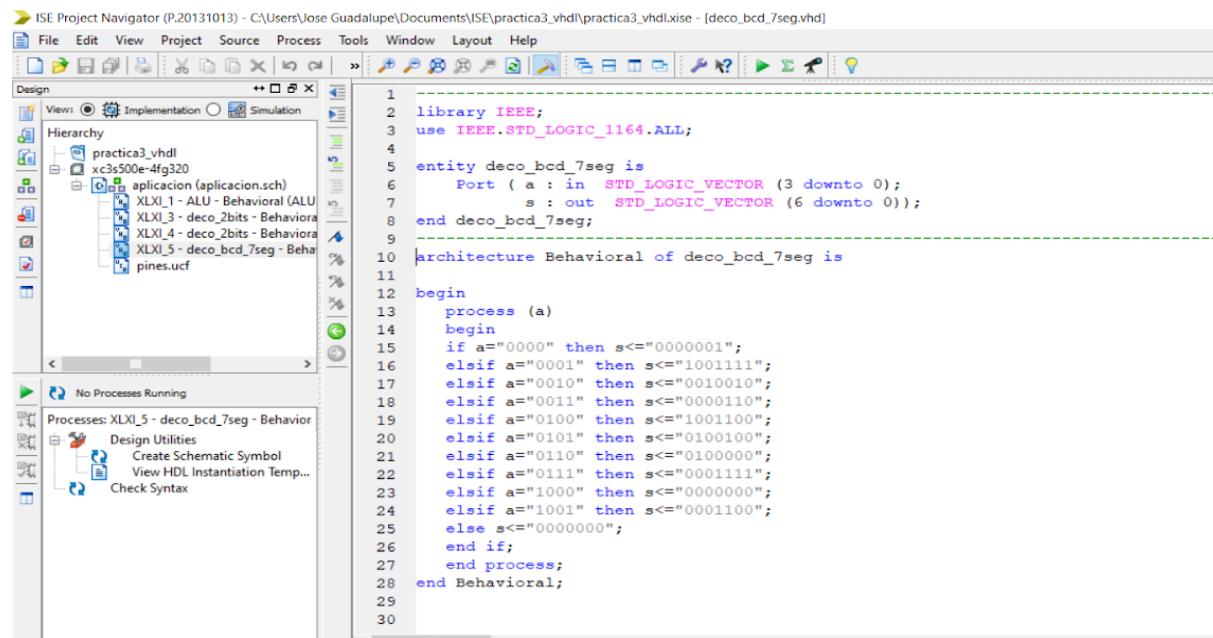
Código de la ALU:



The screenshot shows the ISE Project Navigator interface with the project 'practica3_vhdl' open. The left pane displays the project hierarchy, including the main file 'practica3_vhdl' and sub-files 'xc3s500e-4fg320', 'aplicacion (aplicacion.sch)', and 'XLXI_1 - ALU - Behavioral (ALU)'. The right pane shows the VHDL code for the ALU:

```
1 library IEEE;
2 use IEEE.STD_LOGIC_1164.ALL;
3 use IEEE.STD_LOGIC_UNSIGNED.ALL;
4 use IEEE.STD_LOGIC_ARITH.ALL;
5
6 entity ALU is
7     Port ( a,b,s : in STD_LOGIC_VECTOR (1 downto 0);
8             f : out STD_LOGIC_VECTOR (3 downto 0));
9 end ALU;
10
11 architecture Behavioral of ALU is
12 begin
13     process(s,a,b)
14     begin
15         if s="00" then f <= ("00" & a) + ("00" & b);
16         elsif s="01" then f <= a * b;
17         elsif s="10" then f <= ("00" & a) AND ("00" & b);
18         elsif s="11" then f <= ("00" & a) OR ("00" & b);
19         else f<= "0000";
20     end if;
21     end process;
22
23 end Behavioral;
24
25
26
```

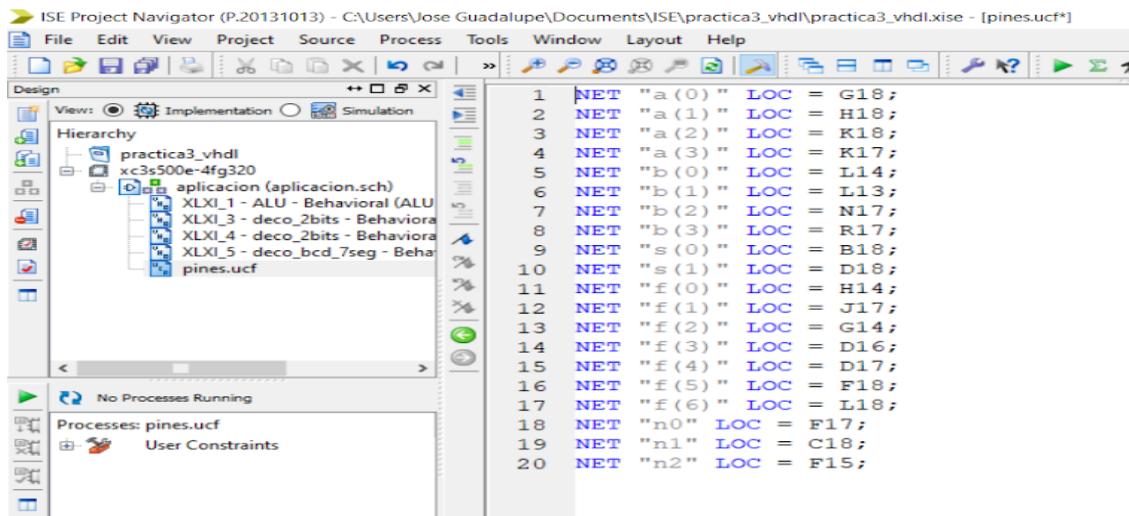
Código del codificador BCD 7 segmentos:



The screenshot shows the ISE Project Navigator interface with the project 'practica3_vhdl' open. The left pane displays the project hierarchy, including the main file 'practica3_vhdl' and sub-files 'xc3s500e-4fg320', 'aplicacion (aplicacion.sch)', and 'XLXI_5 - deco_bcd_7seg - Behavioral (deco_bcd_7seg)'. The right pane shows the VHDL code for the BCD-to-7segment converter:

```
1 library IEEE;
2 use IEEE.STD_LOGIC_1164.ALL;
3
4 entity deco_bcd_7seg is
5     Port ( a : in STD_LOGIC_VECTOR (3 downto 0);
6             s : out STD_LOGIC_VECTOR (6 downto 0));
7 end deco_bcd_7seg;
8
9 architecture Behavioral of deco_bcd_7seg is
10 begin
11     process (a)
12     begin
13         if a="0000" then s<="0000001";
14         elsif a="0001" then s<="1001111";
15         elsif a="0010" then s<="0010010";
16         elsif a="0011" then s<="0000110";
17         elsif a="0100" then s<="1001100";
18         elsif a="0101" then s<="0100100";
19         elsif a="0110" then s<="0100000";
20         elsif a="0111" then s<="0001111";
21         elsif a="1000" then s<="0000000";
22         elsif a="1001" then s<="0001100";
23         else s<="0000000";
24     end if;
25     end process;
26
27 end Behavioral;
28
29
30
```

Pines:



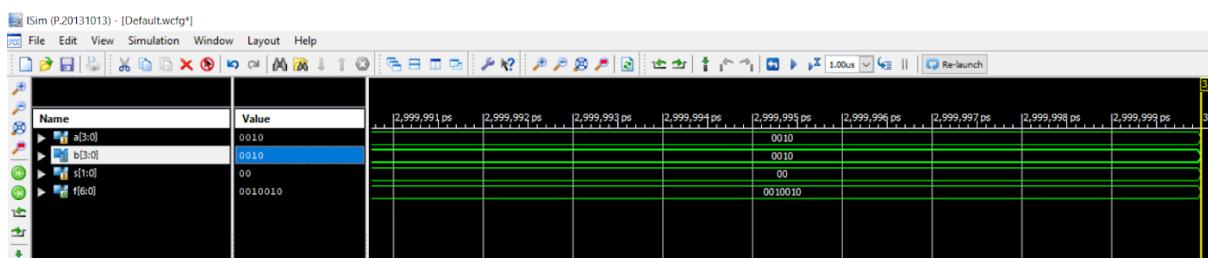
```

1 NET "a (0)" LOC = G18;
2 NET "a (1)" LOC = H18;
3 NET "a (2)" LOC = K18;
4 NET "a (3)" LOC = K17;
5 NET "b (0)" LOC = L14;
6 NET "b (1)" LOC = L13;
7 NET "b (2)" LOC = N17;
8 NET "b (3)" LOC = R17;
9 NET "s (0)" LOC = B18;
10 NET "s (1)" LOC = D18;
11 NET "f (0)" LOC = H14;
12 NET "f (1)" LOC = J17;
13 NET "f (2)" LOC = G14;
14 NET "f (3)" LOC = D16;
15 NET "f (4)" LOC = D17;
16 NET "f (5)" LOC = F18;
17 NET "f (6)" LOC = L18;
18 NET "n0" LOC = F17;
19 NET "n1" LOC = C18;
20 NET "n2" LOC = F15;

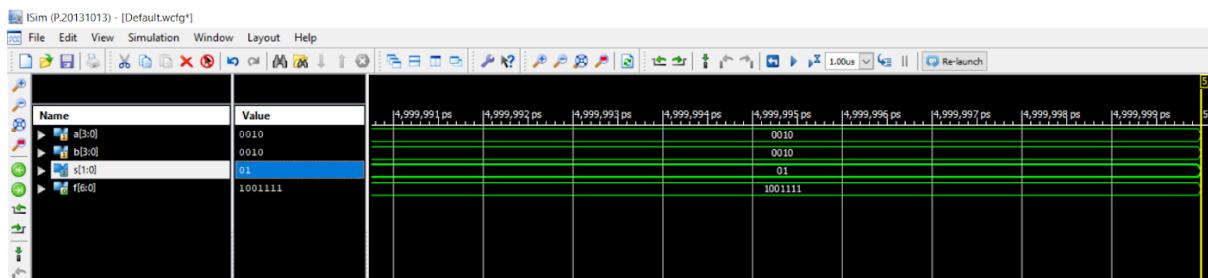
```

Simulación:

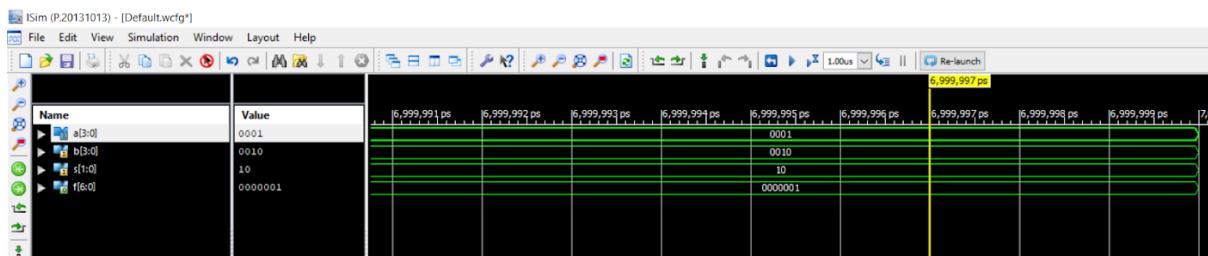
suma $1+1=2$



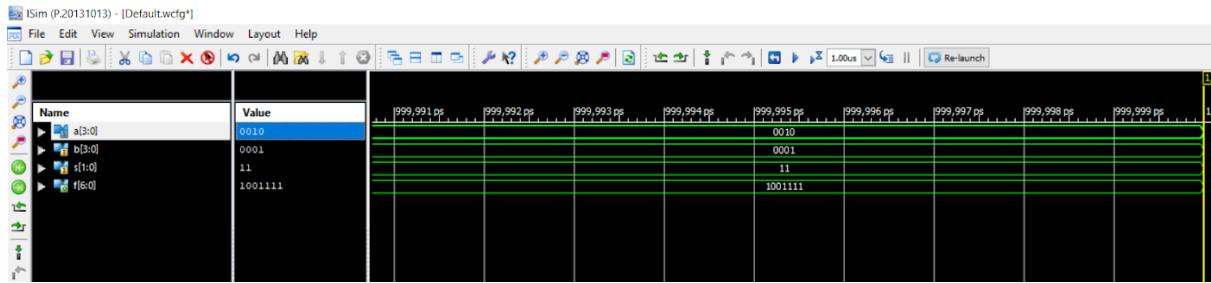
multiplicación $1*1=1$



AND ($0 \text{ AND } 1\right)=0$



OR (0 OR 1=1)



Actividad 3.MODULADOR PWM DIGITAL

La función PWM como abreviatura de la modulación por ancho de pulsos, La modulación por ancho de pulsos de una señal o fuente de energía es una técnica en la que se modifica el ciclo de trabajo de una señal periódica. Este circuito está conformado por los interruptores {D0..D7}, como entradas que ingresan a un comparador de 8 bits, así mismo un contador de 8 bits. La salida se visualiza en un led de la tarjeta Nexys2.

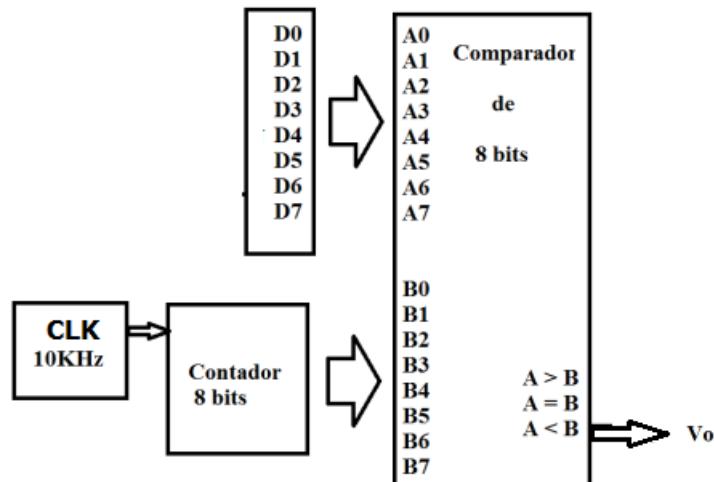


Figura 5. Modulador PWM digital.

Figura 3.- Modulador PWM

- Adicionar los códigos utilizados en VHDL

```

entity proyec3 is
    Port (clk,reset : in STD_LOGIC;
          ent : in STD_LOGIC_VECTOR (7 downto 0);
          sal : out STD_LOGIC);
end proyec3;

architecture Behavioral of proyec3 is
COMPONENT comparador8b
    PORT(a : IN std_logic_vector(7 downto 0);
         b : IN std_logic_vector(7 downto 0);
         s : OUT std_logic);
END COMPONENT;
COMPONENT c8bist
    PORT(
        reloj : IN std_logic;
        r : IN std_logic;
        salida : OUT std_logic_vector(7 downto 0));
END COMPONENT;
signal aux:std_logic_vector(7 downto 0);

begin
Inst_comparador8b: comparador8b PORT MAP(
    a => ent,
    b =>aux,
    s =>sal
);
Inst_c8bist: c8bist PORT MAP(
    reloj =>clk ,
    r =>reset,
    salida => aux
);

```

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity comparador8b is
    Port ( a,b : in STD_LOGIC_VECTOR (7 downto 0);
           s : out STD_LOGIC);
end comparador8b;

architecture Behavioral of comparador8b is begin
process (a,b) begin
    if a=b then s<='0';
    elsif (a<b) then s<='1';
    else s<= '0';
    end if;
end process;
end Behavioral;

```

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
-- 
entity c8bist is
    Port ( reloj,r : in STD_LOGIC;
           salida : out STD_LOGIC_vector(7 downto 0));
end c8bist;
-- 
architecture Behavioral of c8bist is
COMPONENT F10khz
    PORT(
        clk : IN std_logic;
        reset : IN std_logic;
        led : OUT std_logic
    );
END COMPONENT;
COMPONENT contadorde8bits
    PORT(
        c : IN std_logic;
        q : OUT std_logic_vector(7 downto 0)
    );
END COMPONENT;
signal aux:std_logic;
begin
Inst_F10khz: F10khz PORT MAP(
    clk => reloj,
    reset => r,
    led =>aux
);
Inst_contadorde8bits: contadorde8bits PORT MAP(
    c =>aux ,
    q =>salida
);
end Behavioral;

```

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity F10khz is
    Port ( clk : in STD_LOGIC;
           reset : in STD_LOGIC;
           led : out STD_LOGIC
    );
end F10khz;

architecture contador of F10khz is
    signal contar:integer range 0 to 2499:=0;
    signal Salida:std_logic;
begin
Divisor_frecuencia: process (reset,clk) begin
    if reset='0' then
        Salida <='0';
        contar <=0;
    elsif rising_edge(clk) then
        if contar = 2499 then
            Salida <= not Salida;
            contar<=0;
        else
            contar<=contar+1;
        end if;
    end if;
end process;
led<=Salida;
end contador ;

```

```

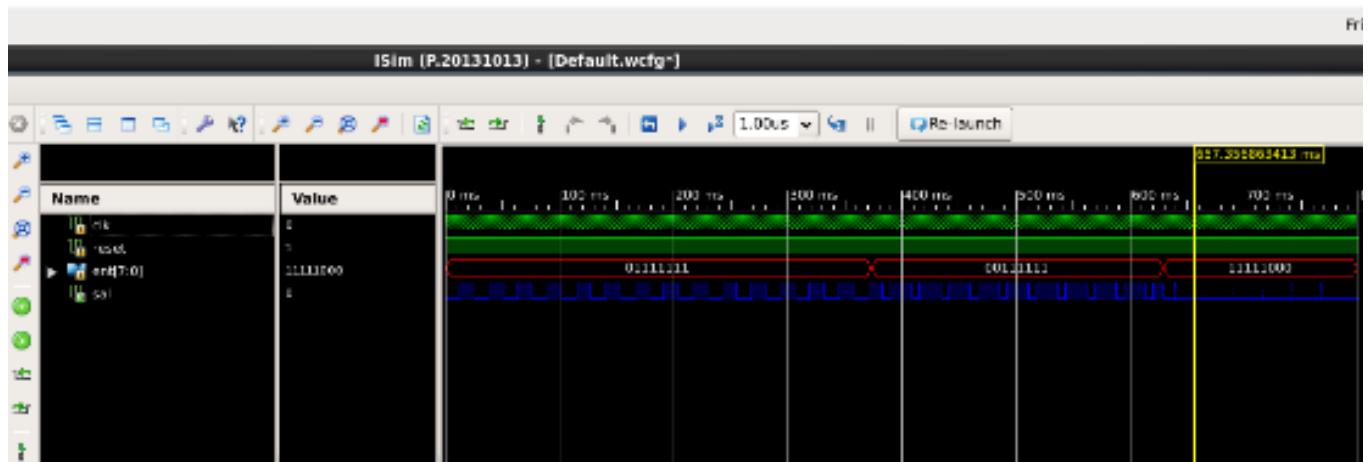
library IEEE;
use IEEE.STD_LOGIC.all;
use IEEE.STD_LOGIC_UNSIGNED.all;

entity contadorde8bits is
    Port ( c : in STD_LOGIC;
           q : out STD_LOGIC_VECTOR(7 downto 0));
end contadorde8bits;

architecture Behavioral of contadorde8bits is
    signal aux:STD_LOGIC_VECTOR (7 downto 0):="00000000";
begin
conta:process (aux,c) begin
    if (c'event and c ='1') then
        if (aux="11111111") then
            aux<="00000000";
        else
            aux<=aux+1;
        end if;
    end if;
    q<=aux;
end process;
end Behavioral;

```

- Representar la simulación.



ACTIVIDAD 4.- Circuito Secuencial en VHDL

Realizar un circuito secuencial en VHDL, el cual está conformado por un divisor de frecuencia, un contador en BCD y un decodificador de BCD a 7 segmentos como se muestra en la siguiente representación a bloques.

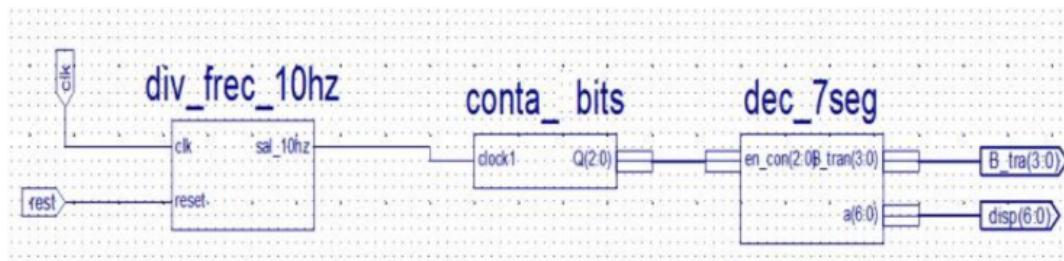


Fig. 4 Contador BCD.

- Habrá que añadir los códigos de los bloques utilizados.
- Representar la simulación.
- En seguida se muestra el archivo de configuración de los pines para descargarlo en la tarjeta.

CONFIGURACIÓN EN PINES

```

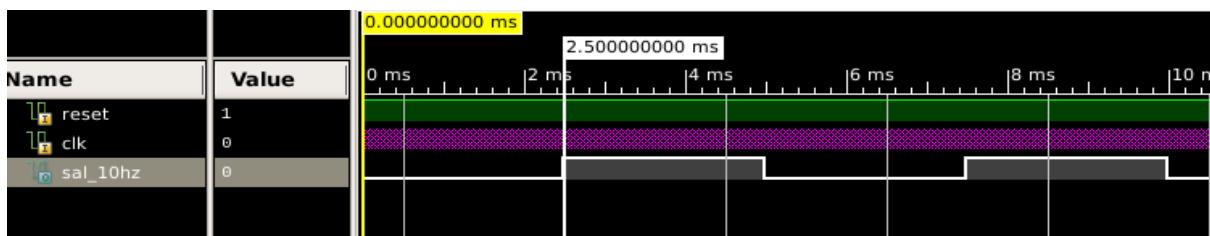
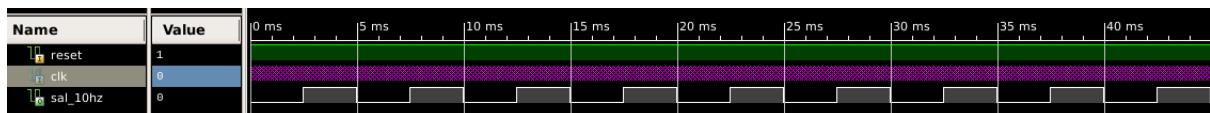
1  NET "clk_in" LOC = B8;
2  NET "rst_in" LOC = G18;
3
4  NET "B(0)" LOC = F15;
5  NET "B(1)" LOC = F17;
6  NET "B(2)" LOC = H17;
7  NET "B(3)" LOC = C18;
8
9  NET "Dis(6)" LOC = L18;
10 NET "Dis(5)" LOC = F18;
11 NET "Dis(4)" LOC = D17;
12 NET "Dis(3)" LOC = D16;
13 NET "Dis(2)" LOC = G14;
14 NET "Dis(1)" LOC = J17;
15 NET "Dis(0)" LOC = H14;
```

Divisor de Frecuencia 10 Hz.

```

1 library IEEE;
2 use IEEE.STD_LOGIC_1164.ALL;
3
4 entity DIVISOR2 is
5     Port ( CLK,RESET : in STD_LOGIC;
6             SAL_10HZ : out STD_LOGIC);
7 end DIVISOR2;
8
9
10 architecture Behavioral of DIVISOR2 is
11 signal conta: integer range 0 to 24999999:=0;
12 signal temp: std_logic:='0';
13 begin
14     process (RESET,CLK) is
15 begin
16     if(RESET='0') then
17         temp <= '0';
18         conta <= 0;
19     elsif (rising_edge(CLK)) then
20         if (conta=24999999) then      -- Comparator
21             temp <= not temp;
22             conta <= 0;
23         else
24             conta <= conta+1;        --Contador
25         end if;
26     end if;
27 end process;
28
29 SAL_10HZ <= temp;
30
31 end Behavioral;

```



Como se aprecia en la simulación se necesitan un total de 2.499999ms para que cambie de estado la salida, esto debido al factor de escalamiento, con un CLK a 1ns.

$$Escala = \frac{f_{entrada}}{f_{deseada}} = \frac{50\text{ MHz}}{10\text{ Hz}} = 5M$$

Por lo tanto, el contador para el divisor de frecuencia tiene como función generar la señal de salida de 10Hz cada 2500000 ciclos.

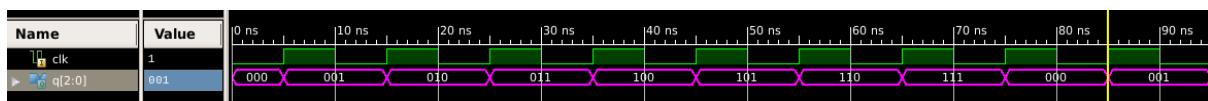
$$Escala = \frac{5M}{2} - 1 = 2,499,999$$

Contador de 3 bits.

```

1 library IEEE;
2 use IEEE.STD_LOGIC_1164.ALL;
3 use IEEE.STD_LOGIC_UNSIGNED.ALL;
4
5 entity CONTADOR is
6     Port ( CLK : in STD_LOGIC;
7             Q : out STD_LOGIC_VECTOR (2 downto 0));
8 end CONTADOR;
9
10 architecture Behavioral of CONTADOR is
11     signal aux : std_logic_vector (2 downto 0) := "000";
12 begin
13     conta: process (aux, CLK)
14     begin
15         if (rising_edge(CLK)) then
16             if (aux = "111") then
17                 aux <= "000";
18             else
19                 aux <= aux+1;
20             end if;
21         end if;
22         Q <= aux;
23     end process conta;
24 end Behavioral;
25

```



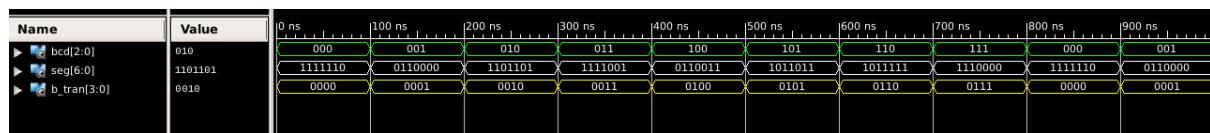
Esta entidad representa un contador de 3 bits sincronizado por un reloj. Toma una entrada de reloj clock y proporciona una salida de 3 bits llamada Q, que representa el estado actual del contador. En cada flanco de subida del reloj, el proceso dentro de la arquitectura incrementa el valor actual del contador en 1, siguiendo un patrón de conteo ascendente de 0 a 7. Si el contador alcanza el valor máximo (7), se reinicia a 0.

Display BCD 7 segmentos.

Para tarjetas de desarrollo con los displays de 7 segmentos conectados en paralelo, se

requiere adicionar un vector de salida y código extra. (Este vector es “*B_tran*”)

```
1 -----  
2 library IEEE;  
3 use IEEE.STD_LOGIC_1164.ALL;  
4 use IEEE.STD_LOGIC_UNSIGNED.ALL;  
5 -----  
6 entity BCD_to_7seg is  
7     Port ( BCD : in STD_LOGIC_VECTOR (2 downto 0);  
8             SEG : out STD_LOGIC_VECTOR (6 downto 0);  
9             B_tran : out STD_LOGIC_VECTOR (3 downto 0));  
10 end BCD_to_7seg;  
11 -----  
12 architecture Behavioral of BCD_to_7seg is  
13 begin  
14     process (BCD)  
15     begin  
16         case BCD is  
17             when "000" =>  
18                 SEG <= "1111110"; -- 0  
19                 B_tran <= "0000";  
20             when "001" =>  
21                 SEG <= "0110000"; -- 1  
22                 B_tran <= "0001";  
23             when "010" =>  
24                 SEG <= "1101101"; -- 2  
25                 B_tran <= "0010";  
26             when "011" =>  
27                 SEG <= "1111001"; -- 3  
28                 B_tran <= "0011";  
29             when "100" =>  
30                 SEG <= "0110011"; -- 4  
31                 B_tran <= "0100";  
32             when "101" =>  
33                 SEG <= "1011011"; -- 5  
34                 B_tran <= "0101";  
35             when "110" =>  
36                 SEG <= "1011111"; -- 6  
37                 B_tran <= "0110";  
38             when "111" =>  
39                 SEG <= "1110000"; -- 7  
40                 B_tran <= "0111";  
41             when others =>  
42                 SEG <= "0000000"; -- Invalid input  
43                 B_tran <= "0000";  
44         end case;  
45     end process;  
46 end Behavioral;
```



```

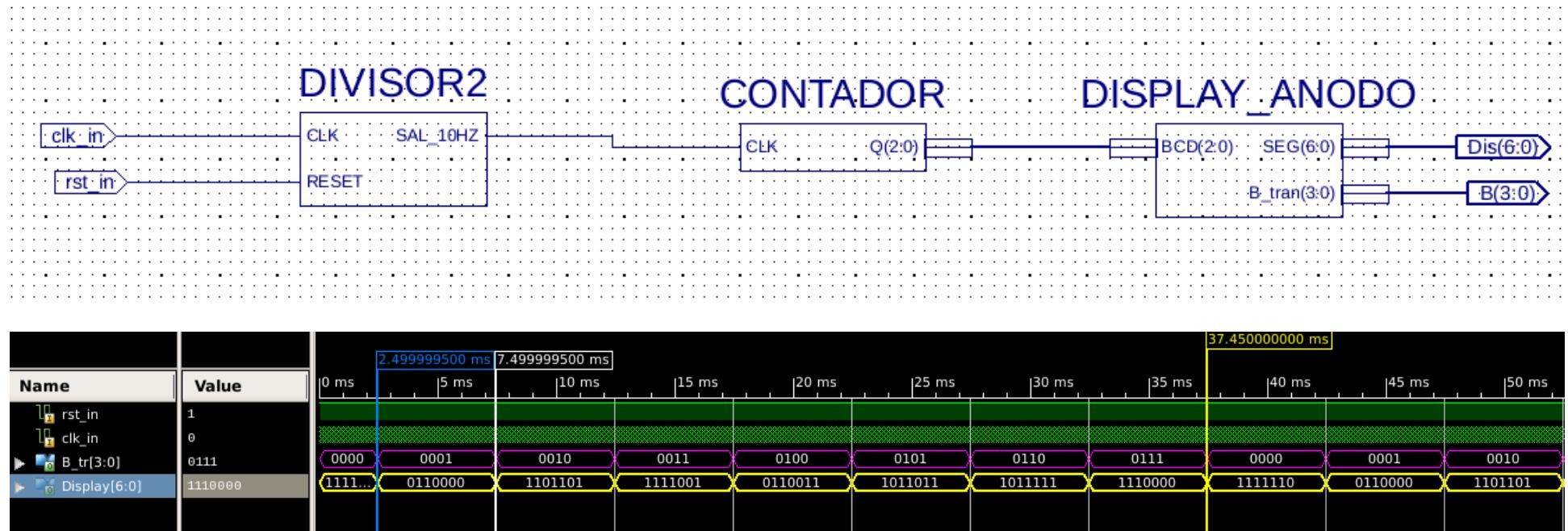
1 -----
2 LIBRARY ieee;
3 USE ieee.std_logic_1164.ALL;
4 -----
5 ENTITY TB_DECO IS
6 END TB_DECO;
7
8 ARCHITECTURE behavior OF TB_DECO IS
9   -- Component Declaration for the Unit Under Test (UUT)
10  COMPONENT BCD_to_7seg
11    PORT(
12      BCD : IN std_logic_vector(2 downto 0);
13      SEG : OUT std_logic_vector(6 downto 0);
14      B_tran : OUT std_logic_vector(3 downto 0)
15    );
16  END COMPONENT;
17
18  --Inputs
19  signal BCD : std_logic_vector(2 downto 0) := (others => '0');
20  --Outputs
21  signal SEG : std_logic_vector(6 downto 0);
22  signal B_tran : std_logic_vector(3 downto 0);
23
24 BEGIN
25   -- Instantiate the Unit Under Test (UUT)

26   uut: BCD_to_7seg PORT MAP (
27     BCD => BCD,
28     SEG => SEG,
29     B_tran => B_tran
30   );
31
32   -- Stimulus process
33   stim_proc0: process
34   begin
35     wait for 100 ns;
36     BCD(0) <= NOT BCD(0);
37   end process;
38
39   stim_proc1: process
40   begin
41     wait for 200 ns;
42     BCD(1) <= NOT BCD(1);
43   end process;
44
45   stim_proc2: process
46   begin
47     wait for 400 ns;
48     BCD(2) <= NOT BCD(2);
49   end process;
50 END;

```

Para este bloque el “BCD” genera los números en decimal del 0 al 7 (Como se muestra en SEG(6:0)) y vuelve a empezar en 0, esto sin importar haber generado todas las combinaciones de BCD(2:0).

FINAL:



*Nota: Para el bloque del “DIVISOR2” se le incrementó el tiempo debido a que la salida generaba cambios muy rápidos, así que al aumentar el tiempo ya era visible en la realidad los números en el display de 7 segmento