-- Incorporamos las librerias

library IEEE;

use IEEE.std\_logic\_1164.ALL;

--Entidad

entity TecladoMat is

PORT (clk : in STD\_LOGIC;

C1 : in STD\_LOGIC;

C2 : in STD\_LOGIC;

C3 : in STD\_LOGIC;

C4 : in STD\_LOGIC;

F1 : out Std\_LOGIC;

F2 : out Std\_LOGIC;

F3 : out Std\_LOGIC;

F4 : out Std\_LOGIC;

T : out std\_LOGIC;

salidas: out STD\_LOGIC\_VECTOR(3 downto 0));

end TecladoMat;

architecture Behavioral of TecladoMat is

type estados\_contador is(S0,S1,S2,S3);

signal estado\_Actual, estado\_Siguiente :estados\_contador;

signal contador: integer range 0 to 500000 := 0;

signal reloj\_aux: std\_LOGIC;

begin

Divisor\_frecuencia: process (clk) begin -- Lista de sensitividad

if rising\_edge(clk) then

if (contador = 50000) then

reloj\_aux <= NOT(reloj\_aux);

contador <= 0;

else

contador <= contador+1;

end if;

end if;

end process;

Proc\_Sec\_Contador: process(reloj\_aux)

begin

if rising\_edge(reloj\_aux) then

estado\_Actual <= estado\_Siguiente;

end if;

end process;

Proc\_Comb\_Contador: process(Estado\_Actual)

begin

case Estado\_Actual is

when S0 =>

F1 <= '1';

F2 <= '0';

F3 <= '0';

F4 <= '0';

Estado\_Siguiente <= S1;

when S1 =>

F1 <= '0';

F2 <= '1';

F3 <= '0';

F4 <= '0';

Estado\_Siguiente <= S2;

when S2 =>

F1 <= '0';

F2 <= '0';

F3 <= '1';

F4 <= '0';

Estado\_Siguiente <= S3;

when S3 =>

F1 <= '0';

F2 <= '0';

F3 <= '0';

F4 <= '1';

Estado\_Siguiente <= S0;

end case;

end process;

Proc\_Comb\_Salidas: process(Estado\_Actual)

begin

case Estado\_Actual is

when S0 =>

if(C1 = '1' ) then

Salidas <= "0001"; --1

T <= '1';

elsif(C2 = '1' ) then

Salidas <= "0010"; -- 2

T <= '1';

elsif(C3 = '1' ) then

Salidas <= "0011"; -- 3

T <= '1';

elsif(C4 = '1' ) then

Salidas <= "1010"; --10,A

T <= '1';

else

Salidas <= "0000";

T <= '0';

end if;

when S1 =>

if(C1 = '1') then

Salidas <= "0100"; -- 4

T <= '1';

elsif(C2 = '1') then

Salidas <= "0101"; -- 5

T <= '1';

elsif(C3 = '1') then

Salidas <= "0110"; -- 6

T <= '1';

elsif(C4 = '1') then

Salidas <= "1011"; -- 11,B

T <= '1';

else

Salidas <= "0000";

T <= '0';

end if;

when S2 =>

if(C1 = '1') then

Salidas <= "0111"; --7

T <= '1';

elsif(C2 = '1') then

Salidas <= "1000"; --8

T <= '1';

elsif(C3 = '1') then

Salidas <= "1001"; --9

T <= '1';

elsif(C4 = '1') then

Salidas <= "1100";--12,C

T <= '1';

else

Salidas <= "0000";

T <= '0';

end if;

when S3 =>

if(C1 = '1') then

Salidas <= "1110"; --14,asterisco

T <= '1';

elsif(C2 = '1') then

Salidas <= "0000"; --0

T <= '1';

elsif(C3 = '1') then

Salidas <= "1111"; --15,gato

T <= '1';

elsif(C4 = '1') then

Salidas <= "1101";--13,D

T <= '1';

else

Salidas <= "0000";

T <= '0';

end if;

end case;

end process;

end architecture;

--Comentario--

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

--Entidad--

entity dFF4Bits is

PORT(D : in std\_logic\_vector(3 downto 0);

clk\_In : in std\_logic;

Q : out std\_logic\_vector(3 downto 0));

end dFF4Bits;

--Arquitectura--

architecture Behavioral of dFF4Bits is

signal aux : std\_logic\_vector(3 downto 0) := "0000";

begin

dFF4Bits : process(D, clk\_In) begin

if rising\_edge(clk\_In) then

if(D /= "0000") then

aux <= D;

end if;

end if;

end process;

--Asignación de señales--

Q <= aux;

end Behavioral;

---Maquina de estados del elevador

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_ARITH.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

--Declaracion de variables--

entity Elevador is

Port (teclado : in std\_LOGIC\_vector(3 downto 0);

clk : in std\_logic;

piso : out std\_LOGIC\_vector(3 downto 0));

end Elevador;

--Arquitectura elevador--

architecture Behavioral of Elevador is

type estadosElevador is (piso1, piso2, piso3, piso4);

signal estadoActual : estadosElevador := piso1;

signal estadoSiguiente: estadosElevador := piso1;

signal pisos : std\_LOGIC\_vector(3 downto 0) := "0001";

begin

cambioPiso: process(clk)

begin

if rising\_edge(clk) then

estadoActual <= estadoSiguiente;

end if;

end process;

asignaSalida: process(estadoActual,teclado)

begin

case estadoActual is

when piso1 =>

if (teclado = "0101") then

pisos <= "0001";

estadoSiguiente <= piso1;

else

pisos <= "0010";

estadoSiguiente <= piso2;

end if;

when piso2 =>

if (teclado = "0110") then

pisos <= "0010";

estadoSiguiente <= piso2;

elsif (teclado = "0101") then

--piso <= "0001";

pisos <= "0001";

estadoSiguiente <= piso1;

else

estadoSiguiente <= piso3;

pisos <= "0011";

end if;

when piso3 =>

if (teclado = "1000") then

pisos <= "0011";

estadoSiguiente <= piso3;

elsif (teclado = "1001") then

pisos <= "0100";

estadoSiguiente <= piso4;

else

pisos <= "0010";

estadoSiguiente <= piso2;

end if;

when piso4 =>

if (teclado = "1001") then

pisos <= "0100";

estadoSiguiente <= piso4;

else

pisos <= "0011";

estadoSiguiente <= piso3;

end if;

end case;

end process;

piso <= pisos;

end Behavioral;

library IEEE;

use IEEE.NUMERIC\_STD.ALL;

use IEEE.STD\_LOGIC\_1164.ALL;

entity Retraso1Hz is

Port (

CLK\_IN : in STD\_LOGIC;

RESET : in STD\_LOGIC;

CLK\_OUT : out STD\_LOGIC

);

end Retraso1Hz;

architecture DivisorFrecuencia1HZ of Retraso1Hz is

signal auxiliar: STD\_LOGIC;

signal contador: integer range 0 to 24999999 := 0; -- valor: 24999999

begin

Divisor\_frecuencia: process (RESET, CLK\_IN) begin -- Lista de sensitividad

if (RESET = '0') then

auxiliar <= '0';

contador <= 0;

elsif rising\_edge(CLK\_IN) then

if (contador = 24999999) then

auxiliar <= NOT(auxiliar);

contador <= 0;

else

contador <= contador+1;

end if;

end if;

end process;

CLK\_OUT <= auxiliar;

end DivisorFrecuencia1HZ;