library IEEE;

use IEEE.std\_logic\_1164.all;

use IEEE.numeric\_std.all;

use IEEE.std\_logic\_unsigned.all;

-- Uncomment the following library declaration if instantiating

-- any Xilinx leaf cells in this code.

--library UNISIM;

--use UNISIM.VComponents.all;

entity main is

port (

clk : in std\_logic;

sw : in STD\_LOGIC\_VECTOR(15 DOWNTO 0); --interruptores

btnU : in STD\_LOGIC; --boton arriba

btnD : in STD\_LOGIC; --boton abajo

btnL : in STD\_LOGIC; --boton izquierda

btnR : in STD\_LOGIC; --boton derecha

btnC : in STD\_LOGIC; --boton central

led : out STD\_LOGIC\_VECTOR(15 DOWNTO 0); --leds

seg : out STD\_LOGIC\_VECTOR(6 DOWNTO 0); --siete segmentos

dp : out STD\_LOGIC; --punto decimal del seite segmentos

an : out STD\_LOGIC\_VECTOR(3 DOWNTO 0); -- control de 7-seg

dcmotor : out std\_logic\_vector (1 downto 0)

);

end main;

architecture Behavioral of main is

signal freq\_switch: std\_logic; -- sw(0)

signal estado: std\_logic\_vector (2 downto 0);

signal cont\_filtro: integer range 0 to 100000000;

signal salida: std\_logic;

signal flag\_suma: std\_logic;

signal flag\_resta: std\_logic;

signal freq\_min: integer range 0 to 100000000;

signal contador\_unidades\_seg: std\_logic\_vector (3 downto 0);

signal contador\_decenas\_seg: std\_logic\_vector (3 downto 0);

signal contador\_base\_enable: integer range 0 to 100000;

signal enable\_seg\_aux: std\_logic\_vector (3 downto 0);

signal dato: std\_logic\_vector (3 downto 0);

begin

freq\_switch <= sw(0);

-- process del automata

process(btnC, clk)

begin

if btnC = '1' then

estado <= "000";

cont\_filtro <= 0;

elsif rising\_edge(clk) then

case estado is

when "000" => -- INICIO

cont\_filtro <= 0;

if btnU = '1' or btnD = '1' then

estado <= "001";

else

estado <= "000";

end if;

when "001" => -- FILTRADO

cont\_filtro <= cont\_filtro + 1;

if (btnU = '1' or btnD = '1') and cont\_filtro < freq\_min then

estado <= "001";

elsif (btnU = '1' or btnD = '1') and cont\_filtro = freq\_min then

if btnU = '1'then

estado <= "010";

elsif btnD = '1' then

estado <= "100";

end if;

else

estado <= "000";

end if;

when "010" => -- UNO +

cont\_filtro <= 0;

if btnU = '1' then

estado <= "010";

else

estado <= "011";

end if;

when "011" => -- SUMA

cont\_filtro <= 0;

if btnU = '1' then

estado <= "001";

else

estado <= "000";

end if;

when "100" => -- UNO -

cont\_filtro <= 0;

if btnD = '1' then

estado <= "100";

else

estado <= "101";

end if;

when "101" => -- RESTA

cont\_filtro <= 0;

if btnD = '1' then

estado <= "001";

else

estado <= "000";

end if;

when others =>

cont\_filtro <= 0;

estado <= "000";

end case;

end if;

end process;

-- process de las salidas

process(estado)

begin

case estado is

when "000" =>

salida <= '0';

flag\_suma <= '0';

flag\_resta <= '0';

when "001" =>

salida <= '0';

flag\_suma <= '0';

flag\_resta <= '0';

when "010" =>

salida <= '0';

flag\_suma <= '0';

flag\_resta <= '0';

when "011" =>

salida <= '1';

flag\_suma <= '1';

flag\_resta <= '0';

when "100" =>

salida <= '0';

flag\_suma <= '0';

flag\_resta <= '0';

when "101" =>

salida <= '1';

flag\_suma <= '0';

flag\_resta <= '1';

when others => salida <= '0';

end case;

end process;

-- process de cambio de frecuencia minima para sumar/restar 1

process(freq\_switch)

begin

if freq\_switch = '0' then

freq\_min <= 100000;

else

freq\_min <= 100000000;

end if;

end process;

process(btnC, clk)

begin

if btnC = '1' then

contador\_unidades\_seg <= "0000";

elsif rising\_edge(clk) then

if salida = '1' and freq\_switch = '0' then

if flag\_suma = '1' then

if contador\_unidades\_seg = 9 then

contador\_unidades\_seg <= "1001";

else

contador\_unidades\_seg <= contador\_unidades\_seg + 1;

end if;

elsif flag\_resta = '1' then

if contador\_unidades\_seg = 0 then

contador\_unidades\_seg <= "0000";

else

contador\_unidades\_seg <= contador\_unidades\_seg - 1;

end if;

end if;

end if;

end if;

end process;

process(btnC, clk)

begin

if btnC = '1' then

contador\_decenas\_seg <= "0000";

elsif rising\_edge(clk) then

if salida = '1' and freq\_switch = '1' then

if flag\_suma = '1' then

if contador\_decenas\_seg = 9 then

contador\_decenas\_seg <= "1001";

else

contador\_decenas\_seg <= contador\_decenas\_seg + 1;

end if;

elsif flag\_resta = '1' then

if contador\_decenas\_seg = 0 then

contador\_decenas\_seg <= "0000";

else

contador\_decenas\_seg <= contador\_decenas\_seg - 1;

end if;

end if;

end if;

end if;

end process;

process(clk, btnC)

begin

if btnC='1' then

contador\_base\_enable<=0;

elsif rising\_edge(clk) then

if contador\_base\_enable=100000 then

contador\_base\_enable<=0;

else

contador\_base\_enable<=contador\_base\_enable+1;

end if;

end if;

end process;

process(clk, btnC)

begin

if btnC='1' then

enable\_seg\_aux<="0111";

elsif rising\_edge(clk) then

if contador\_base\_enable=100000 then

enable\_seg\_aux<=enable\_seg\_aux(2 downto 0)&enable\_seg\_aux(3);

end if;

end if;

end process;

an <= enable\_seg\_aux;

process(enable\_seg\_aux, contador\_unidades\_seg, contador\_decenas\_seg)

begin

case enable\_seg\_aux is

when "0111" => dato <= "1111";

when "1011" => dato <= "1111";

when "1101" => dato <= contador\_decenas\_seg;

when "1110" => dato <= contador\_unidades\_seg;

when others => dato <= "1111";

end case;

end process;

process(dato)

begin

case dato is

when "0000" => seg <= "0000001";

when "0001" => seg <= "1001111";

when "0010" => seg <= "0010010";

when "0011" => seg <= "0000110";

when "0100" => seg <= "1001100";

when "0101" => seg <= "0100100";

when "0110" => seg <= "1100000";

when "0111" => seg <= "0001111";

when "1000" => seg <= "0000000";

when "1001" => seg <= "0001100";

when others => seg <= "1111111";

end case;

end process;

end Behavioral;