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

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if
instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity Display_7seg is
    port(
        BCD : in STD_LOGIC_VECTOR(15 downto 0); -- valor a
mostrar
        RELOJ : in STD_LOGIC; -- reloj
        SEGMENTOS : out STD_LOGIC_VECTOR(6 downto 0); -- 7seg
        ANODOS : out STD_LOGIC_VECTOR(3 downto 0); -- anodos
7seg
        ENABLE : in STD_LOGIC
    );
end Display_7seg;

architecture Behavioral of Display_7seg is
    signal PRESCALER: STD_LOGIC_VECTOR(16 downto 0) :=
"11000011010100000"; -- 500 Hz
    signal PRESCALER_COUNTER: STD_LOGIC_VECTOR(16 downto 0) :=
(others => '0');
    signal COUNTER: STD_LOGIC_VECTOR(1 downto 0) := (others =>
'0');
    signal ANODOS_SHOW: STD_LOGIC_VECTOR(3 downto 0);
    signal DIGITO: STD_LOGIC_VECTOR(3 downto 0 );
begin

    -- asociamos los anodos
    ANODOS <= ANODOS_SHOW;

    -- mostrar los 4 digitos uno a uno
    DISPLAY: process(COUNTER, ENABLE, BCD, DIGITO)
    begin
        -- asignamos los anodos y extraemos los indices de
cada digito
        case COUNTER(1 downto 0) is
            when "00" =>
                ANODOS_SHOW <= "1110"; -- AN 0
                DIGITO <= BCD(3 downto 0);
            when "01" =>
                ANODOS_SHOW <= "1101"; -- AN 1
                DIGITO <= BCD(7 downto 4);
            when "10" =>
                ANODOS_SHOW <= "1011"; -- AN 2
                DIGITO <= BCD(11 downto 8);
            when "11" =>
                ANODOS_SHOW <= "0111"; -- AN 3
```

```
            DIGITO <= BCD(15 downto 12);
        when others =>
            ANODOS_SHOW <= "1111"; -- ninguno
    end case;

    if ENABLE='1' then
        -- mostramos valores para cada digito
        case DIGITO is
            when "0000" =>
                SEGMENTOS <= "1000000";
            when "0001" =>
                SEGMENTOS <= "1111001";
            when "0010" =>
                SEGMENTOS <= "0100100";
            when "0011" =>
                SEGMENTOS <= "0110000";
            when "0100" =>
                SEGMENTOS <= "0011001";
            when "0101" =>
                SEGMENTOS <= "0010010";
            when "0110" =>
                SEGMENTOS <= "0000010";
            when "0111" =>
                SEGMENTOS <= "1111000";
            when "1000" =>
                SEGMENTOS <= "0000000";
            when "1001" =>
                SEGMENTOS <= "0010000";
            when "1010" => --J
                SEGMENTOS <= "1100000";
            when "1011" => --T
                SEGMENTOS <= "1001110";
            when "1100" => --A
                SEGMENTOS <= "0001000";
            when "1101" => --G
                SEGMENTOS <= "1000010";
            when "1110" => --S
                SEGMENTOS <= "0010010";
            when others => --F
                SEGMENTOS <= "0001110";
        end case;
    else
        SEGMENTOS <= "1111111"; -- No mostrar nada
    end if;

end process;

-- calculamos cada cuanto cambiar de digito a mostrar
CONTADOR: process(RELOJ, COUNTER)
begin
    if rising_edge(RELOJ) then
        PRESCALER_COUNTER <= PRESCALER_COUNTER + 1;
        if(PRESCALER_COUNTER = PRESCALER) then
            COUNTER <= COUNTER + 1;
            PRESCALER_COUNTER <= (others => '0');
        end if;
    end if;
```

```
    end process;
end Behavioral;
```