

Práctica 1. **DISEÑO DE UN RELOJ DIGITAL**

OBJETIVO:

Demostrar a los estudiantes mediante el diseño de un reloj digital, que las declaraciones concurrentes se efectúan al mismo tiempo (en paralelo). El orden de escritura en las instrucciones concurrentes no afecta el resultado de síntesis o de simulación.

ESPECIFICACIONES:

Utilizando un FPGA y 4 displays de 7 segmentos, diseñar un reloj digital, el cual visualice en los dos primeros displays las horas y en los siguientes dos los minutos. Cada vez que se llegue a 23 horas con 59 minutos, se reiniciará el conteo de horas y minutos. La figura 1.1 muestra el diagrama del bloque de este sistema.

DIAGRAMA DE BLOQUES:

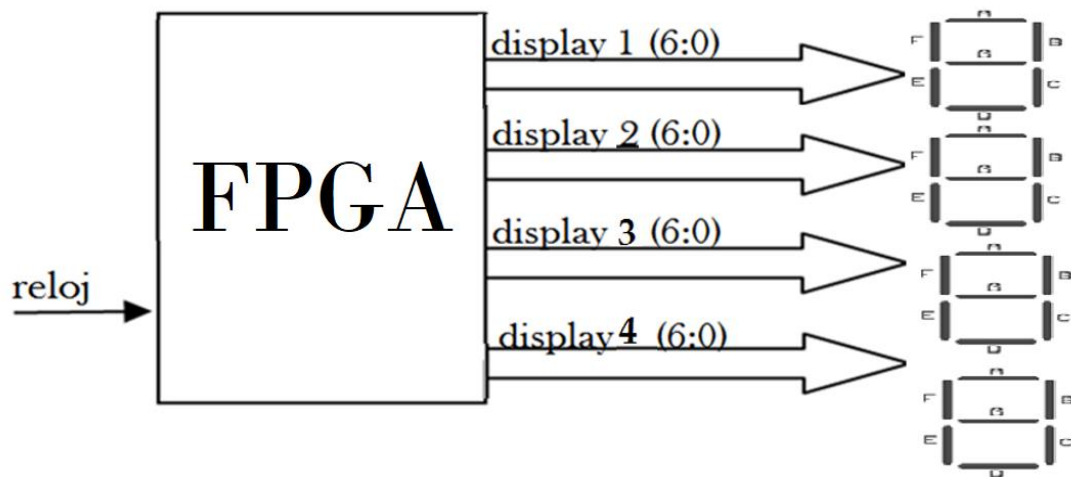


Figura 1.1. Diagrama de bloques del sistema Reloj Digital

Un FPGA puede configurarse con muchos bloques funcionales en lenguaje VHDL que estén ejecutando acciones a la vez. A estas acciones ejecutándose al mismo tiempo se le llama concurrencia.

Las señales son declaraciones necesarias cuando se ejecutan instrucciones concurrentes, debido a que ellas unen los bloques funcionales.

La figura 1.2 muestra los bloques funcionales del sistema Reloj Digital donde las señales se muestran con flechas de color azul.

BLOQUES FUNCIONALES:

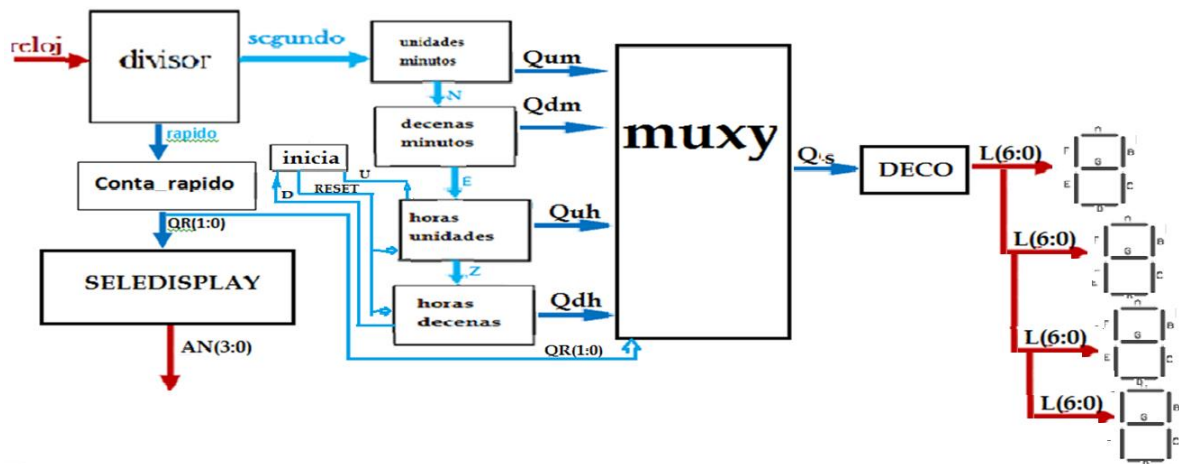


Figura 1.2. Bloques funcionales del sistema Reloj Digital

La figura 1.3 muestra la entidad del sistema Reloj Digital.

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;
entity reldig is
Port ( reloj : in  std_logic;
      AN : out  std_logic_vector (3 downto 0);
      L : out  std_logic_vector (6 downto 0));
end reldig;
```

Figura 1.3. Entidad del sistema Reloj Digital

La figura 1.4 muestra la parte declaratoria de la arquitectura en el sistema Reloj Digital.

```

architecture behavioral of reldig is
    signal segundo : std_logic;
    signal rapido: std_logic;
    signal n : std_logic;
    signal Qs: std_logic_vector(3 downto 0);
    signal Qum: std_logic_vector(3 downto 0);
    signal Qdm: std_logic_vector(3 downto 0);
    signal e : std_logic;
    signal Qr: std_logic_vector(1 downto 0);
    signal Quh: std_logic_vector(3 downto 0);
    signal Qdh: std_logic_vector(3 downto 0);
    signal z : std_logic;
    signal u : std_logic;
    signal d : std_logic;
    signal reset : std_logic;

```

Figura 1.4. Parte declaratoria en la arquitectura del sistema Reloj Digital

La figura 1.5 muestra la parte operatoria de la arquitectura en el sistema Reloj Digital.

```

begin
    divisor : process (reloj)
        variable CUENTA: STD_LOGIC_VECTOR(27 downto 0) := X"00000000";
    begin
        if rising_edge (reloj) then
            if CUENTA =X"48009E0" then
                CUENTA := X"00000000";
            else
                CUENTA := CUENTA +1;
            end if;
        end if;
        segundo <= CUENTA(22);
        rapido <= CUENTA(10);
    end process;

    UNIDADES: process (segundo)
        variable CUENTA: STD_LOGIC_VECTOR(3 downto 0) := "0000";
    begin
        if rising_edge (segundo) then
            if CUENTA ="1001" then
                CUENTA := "0000";
                N <= '1';
            else
                CUENTA := CUENTA +1;
                N <= '0';
            end if;
        end if;
        Qum <= CUENTA;
    end process;

```

Figura 1.5. Parte operatoria de la arquitectura del sistema Reloj Digital

```

DECENAS: process (N)
    variable CUENTA: STD_LOGIC_VECTOR(3 downto 0) := "0000";
begin
    if rising_edge (N) then
        if CUENTA ="0101" then
            CUENTA :="0000";
            E <= '1';
        else
            CUENTA := CUENTA +1;
            E <= '0';
        end if;
    end if;
    Qdm <= CUENTA;
end process;

HoraU: Process(E,reset)
    variable cuenta: std_logic_vector(3 downto 0):="0000";
begin
    if rising_edge(E) then
        if cuenta="1001" then
            cuenta:= "0000";
            Z<='1';
        else
            cuenta:=cuenta+1;
            Z<='0';
        end if;
    end if;
    if reset='1' then
        cuenta:="0000";
    end if;
    Quh<=cuenta;
    U<=cuenta(2);
end Process;

HORAD: Process(Z, reset)
    variable cuenta: std_logic_vector(3 downto 0):="0000";
begin
    if rising_edge(Z) then
        if cuenta="0010" then
            cuenta:= "0000";
        else
            cuenta:=cuenta+1;
        end if;
    end if;
    if reset='1' then
        cuenta:="0000";
    end if;
    Qdh<=cuenta;
    D <=cuenta(1);
end Process;

```

Figura 1.5. (continuación) Parte operatoria de la arquitectura del sistema Reloj Digital

```

        inicia: process (U,D)
        begin
            reset <= (U and D);
        end process;

        CONTRAPID: process (rapido)
            variable CUENTA: STD_LOGIC_VECTOR(1 downto 0) := "00";
        begin
            if rising_edge (rapido) then
                CUENTA := CUENTA +1;
            end if;
            Qr <= CUENTA;
        end process;

        MUXY: PROCESS (Qr)
        BEGIN
            if Qr = "00" then
                Qs<= Qum;
            elsif Qr = "01" then
                Qs<= Qdm;
            elsif Qr = "10" then
                Qs<= Quh;
            elsif Qr = "11" then
                Qs<= Qdh;
            end if;
        end process;

        seledisplay: process (Qr)
        BEGIN
            case Qr is
                when "00" =>
                    AN<= "1110";
                when "01" =>
                    AN<= "1101";
                when "10" =>
                    AN<= "1011";
                when others =>
                    AN<= "0111";
            end case;
        end process;

        with Qs SELECT
            L <= "1000000" when "0000",    --0
                "1111001" when "0001",    --1
                "0100100" when "0010",    --2
                "0110000" when "0011",    --3
                "0011001" when "0100",    --4
                "0010010" when "0101",    --5
                "0000010" when "0110",    --6
                "1111000" when "0111",    --7
                "0000000" when "1000",    --8
                "0010000" when "1001",    --9
                "1000000" when others;    --F
        end Behavioral;

```

Figura 1.5. (continuación) Parte operatoria de la arquitectura del sistema Reloj Digital

ACTIVIDADES COMPLEMENTARIAS:

- 1.- El alumno diseñará un reloj digital con alarma que active una señal sonora con un sonido que vaya aumentando la intensidad hasta que el usuario apague el sistema.
- 2.- El alumno diseñará un reloj digital que trabaje al contrario de las manecillas del reloj, lo que significa que va a descontar tiempo empezando en un valor preseleccionado por el usuario.