Práctica 3.

# GENERACIÓN DE FRECUENCIA, RELOJES Y TEMPORIZADORES

OBJETIVO:

El alumno aprenderá a diseñar temporizadores, divisores de frecuencia y demás bases de tiempo que permitan controlar la respuesta temporal de sistemas diseñados en el lenguaje VHDL.

ESPECIFICACIONES:

Diseñar los tres bloques funcionales básicos que generen las bases de tiempo que pueden ser utilizadas en sistemas desarrollados en el lenguaje VHDL para controlar su respuesta funcional. Estos bloques son:

a) Divisor de frecuencia. Este bloque genera una señal periódica con ciclo de trabajo del 50%, cuya frecuencia corresponde a la de la señal de reloj dividida por un factor de $2^{n+1}$, donde ***n*** es una constante entera definida por el usuario.
b) Señal de reloj. Este bloque genera una señal periódica, con ancho de pulso de 1 ms, cuyo período está dado en milisegundos por un valor entero definida por el usuario.
c) Temporizador. Este bloque genera un pulso de salida, el cual tendrá un ancho de pulso en milisegundos definido por un valor entero definido por el usuario.

En la figura 3.1 se muestra el diagrama de bloques de los módulos de temporización.

BLOQUES FUNCIONALES:

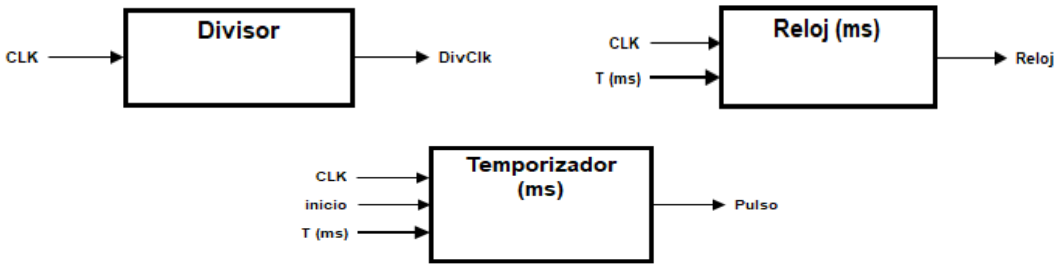


**Figura 3.1. Bloques funcionales de los módulos de temporización**

La figura 3.2 muestra el código para el divisor de frecuencia.

```
----------------------------------------------------------------------
-- Practica 3. Generación de Frecuencia, Relojes y Temporizadores.
-- Module Name:    divisor - Behavioral
-- Project Name: Temporización.
-- Description:
--          Divisor de frecuencia clk/2^(n+1)
--                                                                * RPM
----------------------------------------------------------------------

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

entity Divisor is
   Port ( clk     : in std_logic;
          div_clk : out std_logic);
end Divisor;

architecture Behavioral of Divisor is
begin
   process (clk)
      constant N : integer := 24;   -- N : Valor que define el indice
del divisor
      variable cuenta: std_logic_vector (27 downto 0) := X"0000000";
   begin
      if rising_edge (clk) then
         cuenta := cuenta + 1;
      end if;
      div_clk <= cuenta (N);
      end process;
end Behavioral;
-- Periodo de la salida en funcion del valor N para clk = 50 MHz:
-- 27 ~ 5.37s,   26 ~ 2.68s,   25 ~ 1.34s,   24 ~ 671ms,   23 ~ 336 ms
-- 22 ~ 168 ms,  21 ~ 83.9 ms, 20 ~ 41.9 ms, 19 ~ 21 ms,   18 ~ 10.5ms
-- 17 ~ 5.24 ms, 16 ~ 2.62 ms, 15 ~ 1.31 ms, 14 ~ 655 us,  13 ~ 328 us
-- 12 ~ 164 us,  11 ~ 81.9 us, 10 ~ 41 us,    9 ~ 20.5 us,  8 ~ 10.2us
```

**Figura 3.2. Divisor de frecuencia**

La figura 3.3 muestra el código para el generador de señal de reloj con período dado en milisegundos.

```
----------------------------------------------------------------------------
-- Practica 3. Generación de Frecuencia, Relojes y Temporizadores.
-- Module Name:    RelojMS - Behavioral
-- Project Name: Temporización.
-- Description:
--          Generación de pulso de reloj con periodo T en ms
--                                                                    * RPM
----------------------------------------------------------------------------

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

entity RelojMS is
   Port ( clk : in std_logic;
          Tms : in std_logic_vector(19 downto 0);
          reloj : out std_logic);
end RelojMS;

architecture Behavioral of RelojMS is
   constant fclk : integer := 50_000_000;
   signal clk1ms : std_logic;
begin
   process (clk)               -- Reloj de 1ms
      variable cuenta: integer := 0;
   begin
      if rising_edge (clk) then
         if cuenta >= fclk/1000-1 then
            cuenta := 0;
            clk1ms <= '1';
         else
            cuenta := cuenta + 1;
            clk1ms <= '0';
         end if;
      end if;
   end process;

   process (clk1ms)            -- Reloj con periodo T ms
      variable tiempo: std_logic_vector(19 downto 0) := X"00000";
   begin
      if rising_edge (clk1ms) then
         if tiempo >= Tms-1 then
            tiempo := X"00000";
            reloj <= '1';
         else
            tiempo := tiempo + 1;
            reloj <= '0';
         end if;
      end if;
   end process;
end Behavioral;
```

**Figura 3.3. Generador de señal de reloj con período dado en milisegundos**

La figura 3.4 muestra el código para el temporizador con ancho de pulso dado en milisegundos.

```
----------------------------------------------------------------------------
-- Practica 3. Generación de Frecuencia, Relojes y Temporizadores.
-- Module Name:    timer - Behavioral
-- Project Name: Temporización.
-- Description:
--          Temporizador con ancho de pulso en ms
--                                                                  * RPM
----------------------------------------------------------------------------

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

entity Timer is
   Port ( clk : in std_logic;
          start : in std_logic;
          Tms : in std_logic_vector (19 downto 0);
          P : out std_logic);
end Timer;

architecture Behavioral of Timer is
   constant fclk : integer := 50_000_000;
   signal clk1ms : std_logic;
   signal previo: std_logic := '0';
begin
   process (clk)               -- Reloj de 1ms
      variable cuenta: integer := 0;
   begin
      if rising_edge (clk) then
         if cuenta >= fclk/1000-1 then
            cuenta := 0;
            clk1ms <= '1';
         else
            cuenta := cuenta + 1;
            clk1ms <= '0';
         end if;
      end if;
   end process;
```

**Figura 3.4. Temporizador con ancho de pulso dado en milisegundos**

```
   process (clk1ms, start)     -- Temporizador en ms
      variable cuenta: std_logic_vector (19 downto 0) := X"00000";
            variable contando : bit :='0';
   begin
      if rising_edge (clk1ms) then
         if contando='0' then
            if start /= previo and start='1' then
               cuenta := X"00000";
               contando := '1';
               P <= '1';
            else
               P <= '0';
            end if;
         else
            cuenta := cuenta + 1;
            if cuenta < Tms then
               P <= '1';
            else
               P <= '0';
               contando := '0';
            end if;
         end if;
         previo <= start;
      end if;
   end process;
end Behavioral;
```

**Figura 3.4. (continuación) Temporizador con ancho de pulso dado en milisegundos**

ACTIVIDAD COMPLEMENTARIA:

1.- Genere los bloques funcionales descritos, los cuales pasarán a ser parte de la biblioteca de módulos funcionales del alumno.

2.- Diseñe un bloque que sea capaz de generar una señal de reloj con una frecuencia dada por el usuario, la cual estará dada en MHz, y que tenga un ciclo de trabajo definido por el usuario.

3.- Las demás actividades complementarias serán presentadas el día de la práctica.