INGENIERÍA DE COMPUTADORES 3

Solución al examen de Septiembre 2016

PREGUNTA 1 (2 puntos)

Tomando como base el siguiente código VHDL, dibuje el cronograma de evolución de las señales `x1`, `x2`, `x3` y `x4` entre los instantes 0 y 100 ns.

```
use IEEE.std_logic_1164.all;
entity crono1 is
end entity crono1;
architecture crono1 of crono1 is
  signal x1, x2, x3, x4 : std_logic;
begin
  proc0: process
  begin
     x1  <=  TRANSPORT '0' AFTER 10 NS;
     x1  <=  TRANSPORT '1' AFTER 20 NS;
     x1  <=  TRANSPORT '1' AFTER 15 NS;
     x1  <=  TRANSPORT '0' AFTER 30 NS;
     wait;
  end process;
  Proc1: process
  begin
    x2 <= '1';  wait for 10 ns;
    x2 <= '0';  wait for 20 ns;
    x2 <= '1';  wait for 5 ns;
    x2 <= '0';
  end process;
   x3 <= x1 xor x2 after 10 ns;
  Proc2: process
  begin
    for i in 0 to 3 loop
       x4 <= x2;  wait for 5 ns;
     end loop;
     wait;
  end process;
end architecture crono1;
```

Solución a la Pregunta 1

En la Figura 1.1 se muestra el cronograma de evolución de las señales.

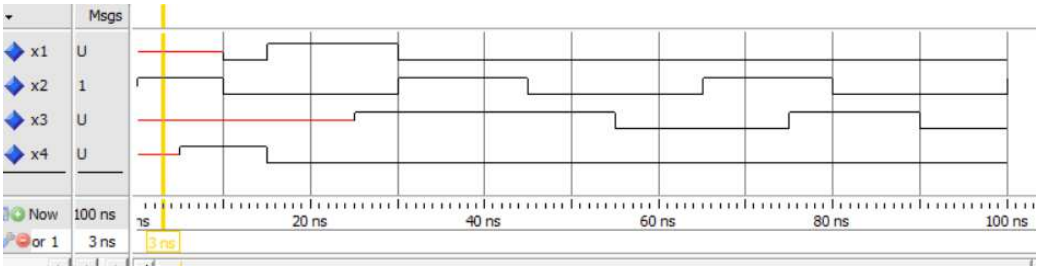

Figura 1.1: Cronograma de evolución de las señales.

PREGUNTA 2 (2.5 puntos)

Se pretende diseñar un circuito comparador cuya salida (F) vale '1' si el valor del número de cuatro bits de entrada (x) es mayor que el número decimal 9. Los cuatro bits de entrada se interpretan como un número binario sin signo. La **entity** del circuito se muestra a continuación.

```
entity comparaXmayor9 is port
 ( F : out std_logic;
   x : in  std_logic_vector(3 downto 0) );
end entity comparaXmayor9;
```

2.a) (1 punto) Escriba la tabla de verdad de la salida (F) en función de la entrada (x). Escriba la función lógica (F) en función de (x) obtenida a partir de dicha tabla de verdad. Escriba en VHDL la **architecture** del circuito comparador empleando únicamente sentencias de asignación concurrente y operadores lógicos.

2.b) (1.5 puntos) Programe el banco de pruebas del circuito combinacional que ha diseñado en la Pregunta 2.a. Explique detalladamente cómo el programa de test comprueba de manera sistemática el funcionamiento del circuito. El banco de pruebas debe comprobar que los valores obtenidos de la UUT

coinciden con los esperados, mostrando el correspondiente mensaje en caso de que no coincidan. Al final del test, debe mostrarse un mensaje indicando el número total de errores.

Solución a la Pregunta 2

El circuito tiene la siguiente tabla de verdad.

X	F
"0000"	'0'
"0001"	'0'
"0010"	'0'
"0011"	'0'
"0100"	'0'
"0101"	'0'
"0110"	'0'
"0111"	'0'
"1000"	'0'
"1001"	'0'
"1010"	'1'
"1011"	'1'
"1100"	'1'
"1101"	'1'
"1110"	'1'
"1111"	'1'

A partir de la anterior tabla de verdad obtenemos la siguiente función lógica:

```
F <= (x(3) and x(2)) or (x(3) and x(1));
```

La **architecture** del circuito comparador se muestra en el Código 1.1 y su banco de pruebas en el Código 1.2.

```
 library IEEE;
use IEEE.std_logic_1164.all;

architecture concurrente1 of comparaXmayor9 is
begin
   F <= (x(3) and x(2)) or (x(3) and x(1));
end architecture concurrente1;
```

Código VHDL 1.1: Architecture del circuito comparador.

```
  ------------------------------------
-- Banco de pruebas comparador X>9
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity bp_comparaXMayor9 is
end entity bp_comparaXMayor9;

architecture bp_comparaXMayor9 of bp_comparaXMayor9 is
   signal F : std_logic;   -- Conectar salidas UUT
   signal x : std_logic_vector(3 downto 0);   -- Conectar entradas UUT
   component comparaXmayor9 is port
      ( F : out std_logic;
        x : in   std_logic_vector(3 downto 0));
   end component comparaXmayor9;
begin
   -- Instanciar y conectar UUT
   uut : component comparaXmayor9 port map
         ( F => F, x => x );
   gen_vec_test : process
      variable test_in      : unsigned (3 downto 0):= B"0000"; -- Vector de test
      variable num_errores  : integer := 0;              -- Numero de errores
   begin
      for count in 0 to 15 loop
        x(3)  <= test_in(3);
        x(2)  <= test_in(2);
        x(1)  <= test_in(1);
        x(0)  <= test_in(0);
        wait for 10 ns;
-- Comprueba resultado
        if (F='0' and test_in>9) or (F='1' and test_in<=9) then
        report("Error. Valor "&integer'image(to_integer(test_in)));
        num_errores := num_errores + 1;
        end if;
        test_in := test_in + 1;
      end loop;
      report "Test completo. Hay "      &
             integer'image(num_errores) &
             " errores.";
      wait; --Final simulación
   end process gen_vec_test;
end architecture bp_comparaXMayor9;
------------------------------------
```

Código VHDL 1.2: Banco de pruebas del comparador.

PREGUNTA 3 (2.5 puntos)

Escriba el código VHDL de la **architecture** que describe el comportamiento de un circuito contador binario de 3 bits cuya salida sigue cíclicamente la secuencia "001", "100", "110" y "111", con señal de reset asíncrona activa a nivel alto. La **entity** del circuito se muestra a continuación.

```
entity contador is
 port (  count         : out std_logic_vector (2 downto 0);
         clk, reset, c : in  std_logic );
end contador;
```

Las entradas al circuito son la señal de reloj (`clk`), la señal de reset asíncrono activo a nivel alto (`reset`) y la señal que habilita la cuenta (`c`). La salida del circuito contador es la señal de 3 bits `count`.

El funcionamiento del circuito debe ser el siguiente:

- *Reset asíncrono activo a nivel alto*.

  Mientras `reset` vale '1', el contador tiene el valor "001".

- *Cuenta síncrona*.

  Mientras `reset` vale '0' y la señal `c` que habilita la cuenta vale'1', el contador pasa cíclicamente en cada flanco de subida de la señal de reloj por la secuencia "001", "100", "110" y "111".

- *Cuenta deshabilitada*.

  Mientras `reset` vale '0' y la señal `c` que habilita la cuenta vale'0', el contador mantiene el valor de la señal de salida `count`.

Describa el comportamiento del circuito en términos de una máquina de Moore. El estado del contador ha de tener el mismo valor que la señal de salida `count`. En caso de que el circuito esté en un estado distinto a los estados "001", "100", "110" y "111", debe pasar en el flanco de subida de la señal de reloj al estado "001". Dibuje el diagrama de estados en el que ha basado su diseño.

Solución a la Pregunta 3

En la Figura 1.2 se muestra el diagrama de estados del circuito contador.

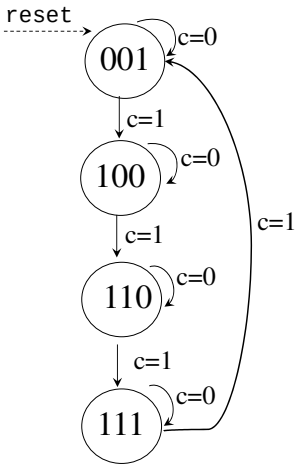

Figura 1.2: Diagrama de estados del circuito detector de secuencias.

El código VHDL del contador se muestra en Código VHDL 1.3.

```
-------------------------------------------------
---Contador implementado como máquina de Moore
-------------------------------------------------
library IEEE;
use IEEE.std_logic_1164.all;
entity contador is
   port (   count       : out std_logic_vector (2  downto 0);
            clk, reset, c  : in    std_logic );
end contador;
architecture contador of contador is
 signal internal_state: std_logic_vector( 2  downto 0);
begin
 count <= internal_state;
 --Cálculo del próximo estado
 proximo_estado: process (clk, reset)
 begin
      if (reset = '1') then
           internal_state <= "001";
      elsif (rising_edge(clk)) then
           case internal_state is
                when "001" =>
                   if (c = '1') then
                    internal_state <= "100";
                   end if;
                when "100" =>
                   if (c = '1') then
                    internal_state <= "110";
                   end if;
                when "110" =>
                   if (c = '1') then
                    internal_state <= "111";
                   end if;
                when "111" =>
                   if (c = '1') then
                    internal_state <= "001";
                   end if;
               when others =>
                    internal_state <= "001";
               end case;
      end if;
end process proximo_estado;
end architecture contador;
-----------------------------------------
```

Código VHDL 1.3: Diseño del circuito contador.

PREGUNTA 4 (3 puntos)

Diseñe un circuito secuencial síncrono capaz de detectar cuando le llega la secuencia de bits "1101" por su entrada serie de un bit. La **entity** del circuito se muestra a continuación.

```
entity detector is
   port( Y     : out std_logic;
         state : out std_logic_vector(1 downto 0);
         X     : in std_logic;
         reset : in std_logic;
         clk   : in std_logic);
end entity detector;
```

El circuito tiene una señal de reloj (`clk`), un entrada serie de un bit (`X`), una señal de reset asíncrona activa en '1' (`reset`), una señal que indica el estado en que se encuentra el circuito (`state`) y una señal de salida de un bit (`Y`).

La señal `Y` se pone a '1' si por los últimos cuatro bits de su entrada `X` ha llegado la secuencia "1101". El circuito ha de ser capaz de detectar secuencias solapadas.

La señal `reset` pone el circuito en su estado inicial. Todos los cambios tienen lugar en el flanco de subida de la señal de reloj.

Escriba en VHDL la **architecture** que describe el comportamiento del circuito en términos de una máquina de Mealy. Dibuje el diagrama de estados correspondiente al circuito que ha diseñado, indicando en cada arco de transición los valores de la señal de entrada `X` y de la señal de salida `Y`.

Solución a la Pregunta 4

En la Figura 1.3 se muestra el diagrama de estados del circuito detector de secuencias.

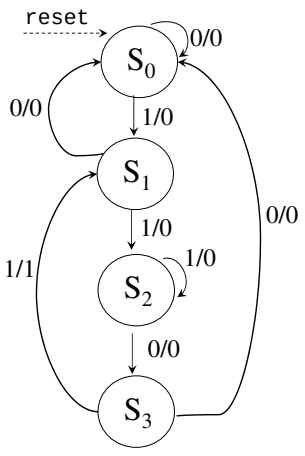

Figura 1.3: Diagrama de estados del circuito detector de secuencias.

El código VHDL del detector se muestra en Código VHDL 1.4.

```
-------------------------------------------------------
---Detector de la secuencia 1101 diseñado como
---maquina de Mealy
-------------------------------------------------------
library IEEE;
use IEEE.std_logic_1164.all;

entity detector is
    port( Y       : out std_logic;
          state   : out std_logic_vector(1 downto 0);
          X       : in std_logic;
          reset   : in std_logic;
          clk     : in std_logic);
end entity detector;

architecture detector of detector is
        signal internal_state: std_logic_vector(1 downto 0);
begin
 state <= internal_state;
--Cálculo salida
 salida: process (internal_state, X) is
 begin
    y <= '0';
    if (internal_state = "11" and x = '1') then
      y <= '1';
    end if;
 end process salida;
--Cálculo del próximo estado
 proximo_estado: process (clk, reset)
 begin
     if (reset = '1') then --reset asíncrono
          internal_state <= "00";
     elsif (rising_edge(clk)) then
          case internal_state is
               when "00" =>     -- Estado actual: S0
                    if X = '1' then
                         internal_state <= "01";
                    end if;
               when "01" =>    --Estado actual: S1
                    if X = '1' then
                         internal_state <= "10";
                    else
                         internal_state <= "00";
                    end if;
               when "10" =>    --Estado actual: S2
                    if X = '0' then
                         internal_state <= "11";
                    end if;
                when "11" =>    -- Estado actual: S3
                     if X = '1' then
                         internal_state <= "01";
                     else
                         internal_state <= "00";
                    end if;
                when others=>    -- Por completitud
                     internal_state <= "00";
               end case;
          end if;
end process proximo_estado;
end architecture detector;
-----------------------------------------
```

Código VHDL 1.4: Diseño del circuito detector de secuencias.