INGENIERÍA DE COMPUTADORES III

Solución al examen de Septiembre 2014

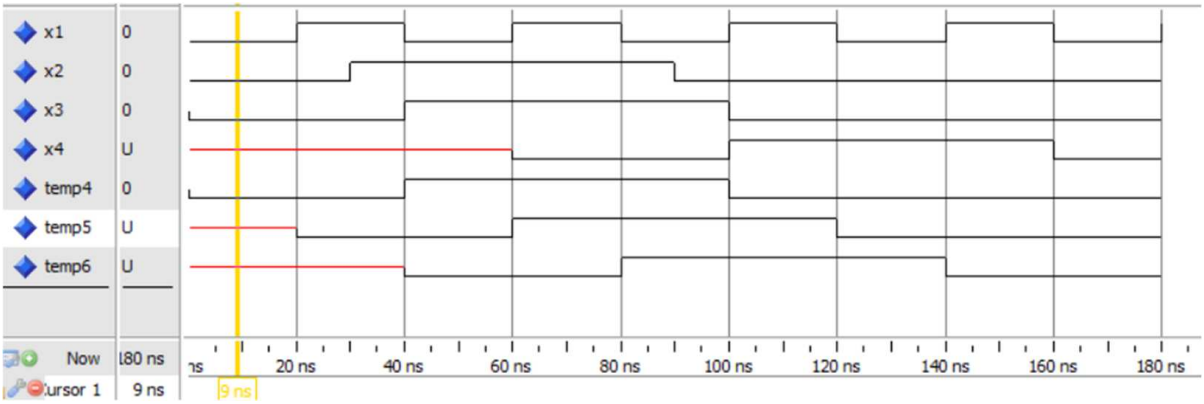
PREGUNTA 1 (2 puntos)

Tomando como base el siguiente código VHDL, dibuje el cronograma de evolución de las señales `x1`, `x2`, `x3`, `temp4`, `temp5`, `temp6` y `x4` entre los instantes 0 y 180 ns.

```
library IEEE; use IEEE.std_logic_1164.all;
entity crono1 is
end entity crono1;
architecture crono1 of crono1 is
   signal x1, x2 : std_logic := '0';
   signal x3, x4 : std_logic;
   signal temp4, temp5, temp6: std_logic;
begin
 process (x1)
    variable temp1, temp2, temp3: std_logic;
  begin
     temp1 := x2;
     temp2 := temp1;
     temp3 := temp2;
     x3 <= temp3;
     temp4 <= x2;
     temp5 <= temp4;
     temp6 <= temp5;
     x4 <= temp6;
  end process;
  x1 <= not x1 after 20 ns;
  x2 <= '0',  '1' after 30 ns, '0' after 90 ns;
end architecture crono1;
```

Solución a la Pregunta 1

En la siguiente figura se muestra el cronograma solución a la Pregunta 1.



PREGUNTA 2 (3 puntos)

Diseñe un circuito digital combinacional que tenga como entrada una señal de 4 bits y como salida una señal de un bit que se ponga a '1' si la señal de entrada es "0010", "0011", "1010", "1011" ó "1111" y se ponga a '0' en cualquier otro caso. La **entity** del circuito digital se muestra a continuación.

```
entity circ2 is
port (   y: out std_logic;
          x: in std_logic_vector(3 downto 0)    );
end entity circ2;
```

2.a) (0.25 puntos) Escriba en VHDL la **entity** y la **architecture** que describe el comportamiento de una puerta NOT.

2.b) (0.25 puntos) Escriba en VHDL la **entity** y la **architecture** que describe el comportamiento de una puerta AND de 3 entradas.

2.c) (0.25 puntos) Escriba en VHDL la **entity** y la **architecture** que describe el comportamiento de una puerta OR de 3 entradas.

2.d) (0.25 puntos) Dibuje el diagrama a nivel de puertas lógicas del circuito combinacional descrito. Emplee para ello únicamente puertas NOT, puertas AND de 3 entradas y puertas OR de 3 entradas.

2.e) (2 puntos) Escriba en VHDL la **architecture** que describe la estructura del circuito combinacional siguiendo el diagrama dibujado en el apartado anterior y empleando las puertas lógicas cuyo diseño ha realizado al contestar los tres primeros apartados.

Solución a la Pregunta 2

El código correspondiente a las puertas lógicas NOT, AND y OR se muestra en Código VHDL 1.1–1.3.

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

entity not1 is
    port (  y0      : out std_logic;
            x0      : in std_logic );
end entity not1;

architecture not1 of not1 is
begin
     y0 <= not x0;
end architecture not1;
```

Código VHDL 1.1: Puerta lógica NOT.

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

entity and3 is
    port (  y0      : out std_logic;
            x0, x1, x2 : in std_logic );
end entity and3;

architecture and3 of and3 is
begin
     y0 <= x2 and x1 and x0;
end architecture and3;
```

Código VHDL 1.2: Puerta lógica AND de 3 entradas.

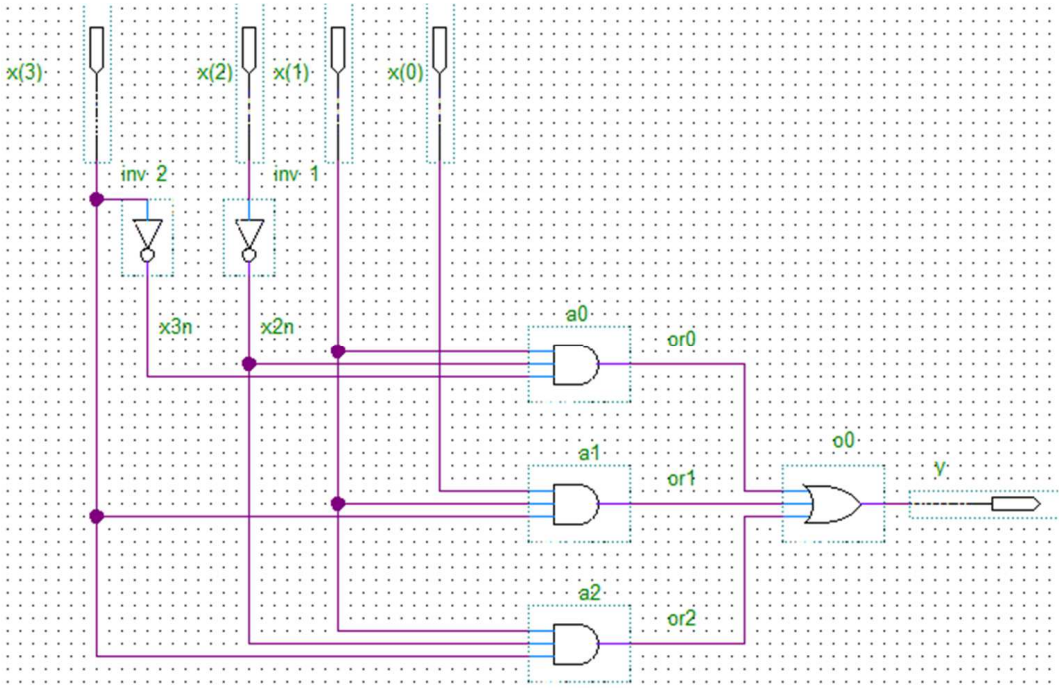
A continuación se muestra la figura del diagrama circuital del circuito combinacional, donde se han usado puertas NOT, puertas AND de 3 entradas y puertas OR de 3 entradas como se indicaba en el enunciado. El código VHDL de dicho circuito se muestra en Código VHDL 1.4.

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

entity or3 is
   port ( y0 : out std_logic;
        x0, x1, x2 : in std_logic );
end entity or3;

architecture or3 of or3 is
begin
   y0 <= x0 or x1 or x2;
end architecture or3;
--------------------------------------
```

Código VHDL 1.3: Puerta lógica OR de 3 entradas.



```
--------------------------------------
library IEEE;
use IEEE.std_logic_1164.all;
architecture circ2_Estruc of circ2 is
   signal x3n, x2n : std_logic;
   signal or0, or1, or2 : std_logic;
-- Declaración de las clases de los componentes
   component and3 is
      port (  y0 : out std_logic ;
      x0, x1, x2 : in std_logic);
   end component and3;
   component not1 is
      port ( y0 : out std_logic;
               x0 : in std_logic );
   end component not1;
   component or3 is
      port ( y0 : out std_logic;
          x0, x1, x2 : in std_logic );
   end component or3;
begin
-- Instanciación y conexión de los componentes
   inv_1 : component not1 port map (x2n, x(2));
   inv_2 : component not1 port map (x3n, x(3));
   a0 : component and3 port map (or0, x3n, x2n, x(1));
   a1 : component and3 port map (or1, x(3), x(1), x(0));
   a2 : component and3 port map (or2, x(3), x2n, x(1));
   o0 : component or3 port map (y, or0, or1, or2);
end architecture circ2_Estruc;
--------------------------------------
```

Código VHDL 1.4: Descripción estructural del circuito combinacional.

PREGUNTA 3 (2 puntos)

Programe en VHDL el banco de pruebas del circuito combinacional que ha diseñado al contestar a la Pregunta 2.e). 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.

Solución a la Pregunta 3

El código VHDL del banco de pruebas del circuito se muestra en Código VHDL 1.5.

```
-------------------------------------
-- Banco de pruebas
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
entity bp_circ2 is
end entity bp_circ2;
architecture bp_circ2 of bp_circ2 is
  signal y : std_logic; -- Conectar salida UUT
  signal x : std_logic_vector(3 downto 0); -- Conectar entrada UUT
  component circ2 is port
   ( y : out std_logic;
     x : in std_logic_vector(3 downto 0));
  end component circ2;
begin
-- Instanciar y conectar UUT
  uut : component circ2 port map( y=> y, x => x);
  gen_vec_test : process
    variable test_in : unsigned (3 downto 0); -- Vector de test
    variable y2: std_logic;
  begin
    test_in := B"0000";
    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;
      if ( test_in = "0010") or
       (test_in =  "0011") or (test_in = "1010" )
         or (test_in = "1011" ) or (test_in = "1111" ) then
         y2 := '1';
      else
         y2 := '0';
      end if;
     assert(y2 = y)
   report      "ERROR.La salida del circuito no corresponde con la salida
esperada";
      test_in := test_in + 1;
    end loop;
    wait;
  end process gen_vec_test;
end architecture bp_circ2;
-------------------------------------
```

Código VHDL 1.5: Banco de pruebas del circuito combinacional.

PREGUNTA 4 (3 puntos)

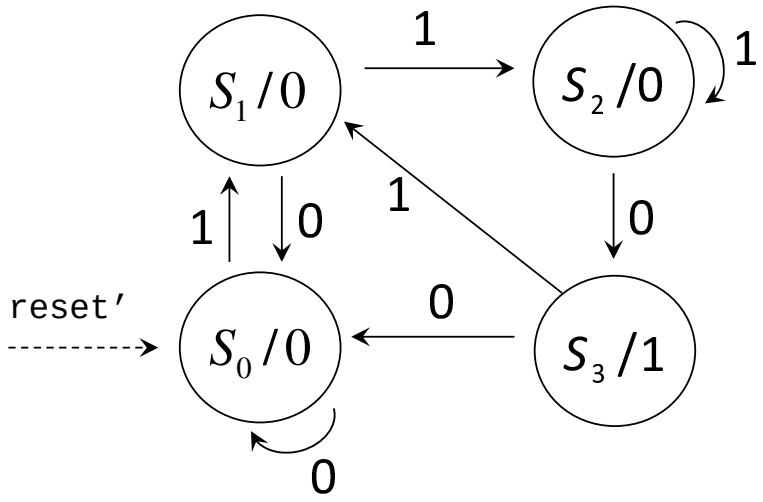
Diseñe un circuito secuencial síncrono capaz de detectar cuando le llega la secuencia "110" por su entrada. La **entity** del circuito se muestra a continuación.

```
entity detector is
   port( Y     : out std_logic;
         X     : in std_logic;
         reset : in std_logic;
         clk   : in std_logic);
end entity detector;
```

El circuito tiene una señal de reloj (`clk`), una entrada serie de un bit (`X`), una señal de reset asíncrona activa en '0' (`reset`) y una señal de salida de un bit (`Y`). La señal `Y` se pone a '1' si los últimos tres bits que han llegado por la entrada (`X`) se corresponden con la secuencia "110". 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 Moore. Dibuje el diagrama de estados correspondiente al circuito que ha diseñado.

Solución a la Pregunta 4

A continuación se muestra la figura del diagrama de estados del circuito dectector.



El código del circuito detector se muestra en Código VHDL 1.6–1.7.

```
-------------------------------------------------------
---Detector secuencia 110
-------------------------------------------------------
library IEEE;
use IEEE.std_logic_1164.all;
entity detector is
    port( Y      : out std_logic;
          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
 --Cálculo salida
 salida: process (internal_state) is
 begin
     case internal_state is
          when "00" => Y <= '0';
          when "01" => Y <= '0';
          when "10" => Y <= '0';
          when others => Y <= '1';
      end case;
 end process salida;
```

Código VHDL 1.6: Circuito detector.

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

Código VHDL 1.7: Continuación del circuito detector.