INGENIERÍA DE COMPUTADORES 3

Solución al Trabajo Práctico - Septiembre de 2019

EJERCICIO 1 (3 PUNTOS)

Se desea diseñar un circuito digital que implemente las funciones `F1` y `F2` cuyas tablas de verdad se muestran a continuación, que dependen de las tres variables `a`, `b` y `c`:

a	b	c	F1	F2
'0'	'0'	'0'	'0'	'0'
'0'	'0'	'1'	'0'	'0'
'0'	'1'	'0'	'1'	'0'
'0'	'1'	'1'	'1'	'1'
'1'	'0'	'0'	'1'	'0'
'1'	'0'	'1'	'1'	'0'
'1'	'1'	'0'	'1'	'0'
'1'	'1'	'1'	'1'	'1'

1.a) (0.25 puntos) Obtenga las funciones lógicas `F1` y `F2` a partir de la tabla de verdad. Escriba en VHDL la **entity** del circuito que implemente las dos funciones lógicas. Es decir, que tenga tres entradas `a`, `b` y `c`, y dos salidas `F1` y `F2`.

1.b) (0.75 puntos) Escriba en VHDL la **architecture** que describa el *comportamiento* del circuito.

1.c) (0.25 puntos) Dibuje el diagrama de un circuito que implemente estas funciones lógicas al nivel de puertas lógicas. No es necesario que el circuito esté simplificado. A continuación, escriba en VHDL la **entity** y la **architecture**

de cada una de las puertas lógicas que componen el circuito que acaba de dibujar.

1.d) (0.75 puntos) Escriba en VHDL una **architecture** que describa la *estructura* del circuito que ha dibujado, instanciando y conectando las puertas lógicas que ha diseñado anteriormente.

1.e) (1 punto) Escriba en VHDL un banco de pruebas que permita visualizar, para todos los posibles valores de las entradas, las salidas de los circuitos diseñados en los Apartados 1.b y 1.d. Compruebe mediante inspección visual que los dos diseños funcionan correctamente. Incluya en la memoria los dos cronogramas obtenidos al realizar la simulación del banco de pruebas para comprobar los circuitos diseñados en los Apartados 1.b y 1.d.

Solución al Ejercicio 1

La **entity** del circuito que implementa la función lógica se muestra en Código VHDL 1.1. El Código VHDL 1.2 muestra la **architecture** del circuito describiendo su comportamiento.

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

entity funcF1F2 is
   port ( F1, F2: out std_logic;
          a, b, c : in std_logic );
end entity funcF1F2;
-------------------------------------
```

Código VHDL 1.1: Solución al Apartado 1.a: **entity** del circuito.

```
-------------------------------------
architecture Comp of funcF1F2 is
begin
 F1 <= a or b;
 F2 <= b and c;
end architecture Comp;
-------------------------------------
```

Código VHDL 1.2: Solución al Apartado 1.b: **architecture** del circuito describiendo su comportamiento.

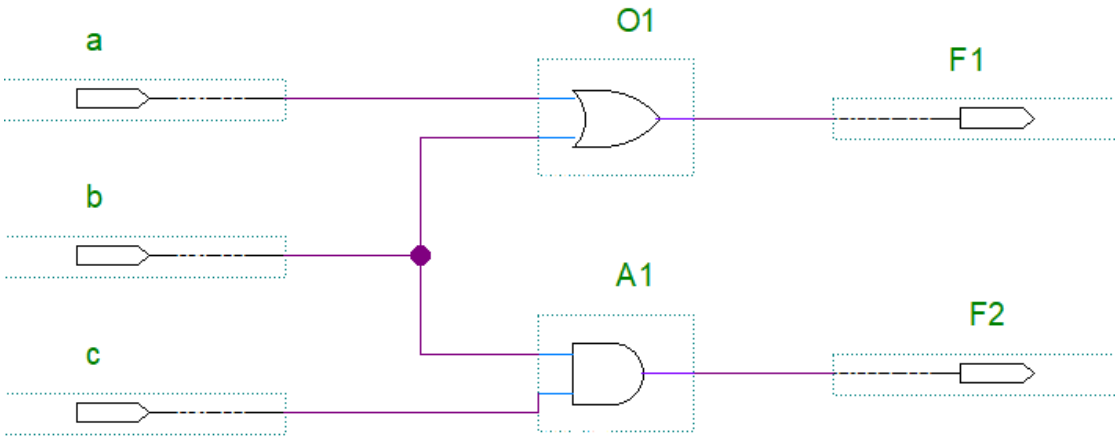

Figura 1.1: Solución al Apartado 1.c: diagrama al nivel de puertas lógicas.

La Figura 1.1 muestra el diagrama del circuito implementado empleando una puerta AND de dos entradas y una puerta OR de dos entradas.

El código VHDL de la **entity** y la **architecture** de estas dos puertas lógicas se muestra en Código VHDL 1.3 y 1.4, respectivamente. El Código VHDL 1.5 muestra la **architecture** del circuito describiendo su estructura.

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

entity and2 is
   port ( y0 : out std_logic;
          x0, x1 : in std_logic );
end entity and2;

architecture and2 of and2 is
begin
   y0 <= x0 and x1;
end architecture and2;
-----------------------------------
```

Código VHDL 1.3: Puerta AND de dos entradas.

El código VHDL del banco de pruebas del circuito se muestra en Código VHDL 1.6. Los dos cronogramas obtenidos al simular el banco de pruebas aplicado al diseño del Apartado 1.b. y 1.d se muestran, respectivamente, en las Figuras 1.2 y 1.3.

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

entity or2 is
   port ( y0 : out std_logic;
        x0, x1 : in std_logic );
end entity or2;

architecture or2 of or2 is
begin
   y0 <= x0 or x1;
end architecture or2;
-------------------------------------
```

Código VHDL 1.4: Puerta lógica OR de 2 entradas.

```
-------------------------------------
library IEEE;
use IEEE.std_logic_1164.all;
architecture circuito_Estruc of funcF1F2 is
-- Declaración de las clases de los componentes
   component and2 is
      port (   y0 : out std_logic ;
            x0, x1 : in std_logic);
   end component and2;

   component or2 is
      port ( y0 : out std_logic;
           x0, x1 : in std_logic );
   end component or2;
begin
-- Instanciación y conexión de los componentes
   O1 : component or2 port map (F1, a, b);
   A1 : component and2 port map (F2, b, c);
end architecture circuito_Estruc;
-----------------------------------
```

Código VHDL 1.5: Solución al Apartado 1.d: **architecture** del circuito describiendo su estructura.

```
-- Banco de pruebas del circuito Ejercicio 1
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity bp_funcF1F2 is
    constant DELAY      : time      := 20 ns; -- Retardo usado en el test
end entity bp_funcF1F2;

architecture bp_funcF1F2 of bp_funcF1F2 is
    signal F1, F2 : std_logic;
    signal a, b, c   : std_logic;

    component funcF1F2 is
     port (   F1, F2 : out std_logic;
            a, b, c  : in std_logic );
     end component funcF1F2;

begin
UUT : component funcF1F2 port map
          (F1, F2, a, b, c);

vec_test : process is
      variable valor  : unsigned (2 downto 0);
begin
       -- Generar todos los posibles valores de entrada
       for i in 0 to 7 loop
          valor := to_unsigned(i,3);
          a <= std_logic(valor(2));
          b <= std_logic(valor(1));
          c <= std_logic(valor(0));
          wait for DELAY;
       end loop;
       wait;      -- Final de la simulación
end process vec_test;
end architecture bp_funcF1F2;
-------------------------------------------------
```

Código VHDL 1.6: Solución al Apartado 1.e: banco de pruebas del circuito.

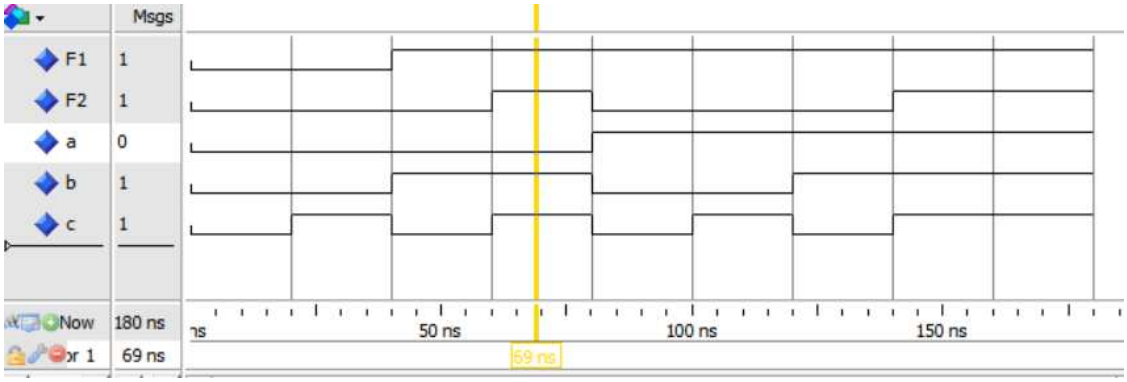

Figura 1.2: Cronograma del banco de pruebas aplicado al diseño del Apartado 1.b.

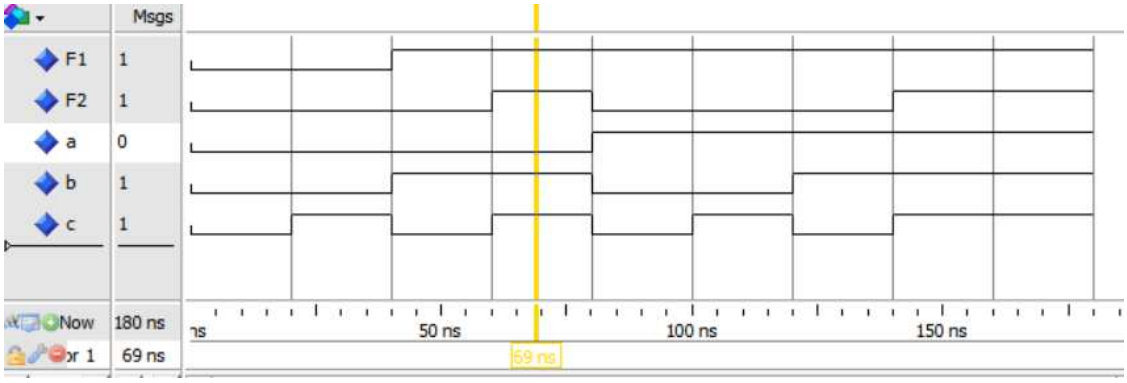

Figura 1.3: Cronograma del banco de pruebas aplicado al diseño del Apartado 1.d.

EJERCICIO 2 (7 PUNTOS)

Se pretende diseñar un circuito combinacional que tiene una señal de entrada `X` de 5 bits y una señal de salida `Y` de 5 bits. El bit i $(i = 0, \ldots, 4)$ de la señal `Y` (Y_i) tiene el valor '1' sólo si la cadena de bits $X_0X_1 \ldots X_{i-1}X_i$ es igual que la cadena de bits $X_iX_{i-1} \ldots X_1X_0$. En caso contrario, el valor del bit i de la señal `Y` (Y_i) es '0'.

Por ejemplo, si la señal de entrada tiene el valor "01100" la señal de salida toma el valor "00011". Y_0 tiene valor '1', ya que $X_0 = X_0$. Y_1 tiene valor '1', ya que $X_1X_0 = X_0X_1$. Y_2 tiene valor '0', ya que $X_2X_1X_0 \neq X_0X_1X_2$. Y_3 tiene valor '0', ya que $X_3X_2X_1X_0 \neq X_0X_1X_2X_3$. Y_4 tiene valor '0', ya que $X_4X_3X_2X_1X_0 \neq X_0X_1X_2X_3X_4$.

2.a) (0.5 puntos) Escriba en VHDL la **entity** del circuito combinacional. Las señales de entrada y salida del circuito han de ser del tipo `std_logic_vector` y tener la dimensión y los nombres especificados en el enunciado.

2.b) (2 puntos) Escriba en VHDL la **architecture** que describe el comportamiento del circuito empleando un bloque **process** y sentencias **if**.

2.c) (1 punto) Escriba en VHDL la **architecture** que describe el comportamiento del circuito empleando únicamente sentencias de asignación concurrente.

2.d) (2 puntos) Dibuje el diagrama del circuito al nivel de puertas lógicas. No es necesario que el circuito esté simplificado. A continuación, escriba en VHDL la **entity** y la **architecture** de cada una de las puertas lógicas que componen el circuito que acaba de dibujar.

Escriba en VHDL una **architecture** que describa la *estructura* del circuito que ha dibujado, instanciando y conectando las puertas lógicas que ha diseñado anteriormente.

2.e) (1.5 puntos) Escriba en VHDL un banco de pruebas que permita visualizar, para todos los posibles valores de las entradas, las salidas de los circuitos diseñados en los Apartados 2.b, 2.c y 2.d. El banco de pruebas debe comparar las salidas de la UUT con las salidas esperadas, mostrando el correspondiente mensaje de error en caso de que las salidas obtenidas de la UUT no correspondan con las esperadas.

Compruebe que los tres diseños funcionan correctamente. Incluya en la memoria los tres cronogramas obtenidos al realizar la simulación del banco de pruebas para comprobar los circuitos diseñados en los Apartados 2.b, 2.c y 2.d.

Solución al Ejercicio 2

El código VHDL de la **entity** del circuito se muestra en Código VHDL 1.7. La **architecture** que describe el comportamiento del circuito empleando un bloque **process** y sentencias **if** se muestra en Código VHDL 1.8. La **architecture** que describe el comportamiento del circuito empleando únicamente sentencias de asignación concurrentes se muestra en Código VHDL 1.9.

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

entity circ is
   port ( y   : out std_logic_vector(4  downto 0);
          x   : in   std_logic_vector(4  downto 0));
end entity circ;
-------------------------------------
```

Código VHDL 1.7: Solución al Apartado 2.a: **entity** del circuito detector.

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

architecture circ of circ is
begin
   process (x)
   begin
    y <= "00001";
    if(x(1  downto 0) = x(0)&x(1) ) then
       y(1) <= '1';
    end if;
    if (x(2  downto 0) = x(0)&x(1)&x(2)) then
       y(2) <= '1';
    end if;
    if (x(3  downto 0) =   x(0)&x(1)&x(2)&x(3)) then
       y(3) <= '1';
    end if;
    if (x =   x(0)&x(1)&x(2)&x(3)&x(4)) then
       y(4) <= '1';
    end if;
    end process;
end architecture circ;
-------------------------------------
```

Código VHDL 1.8: Solución al Apartado 2.b: **architecture** del circuito detector empleando un bloque **process**.

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

architecture circ of circ is
begin
   y(0) <= '1';
   y(1) <= (x(0) and x(1)) or (not x(0) and not x(1));
   y(2) <= (x(0) and x(2)) or (not x(0) and not x(2));
   y(3) <= ((x(0) and x(3)) or (not x(0) and not x(3))) and ((x(1) and x(2))or (not x(1)
and not x(2)));
   y(4) <= ((x(0) and x(4)) or (not x(0) and not x(4))) and
              ((x(1) and x(3))or (not x(1) and not x(3)));

end architecture circ;
-------------------------------------
```

Código VHDL 1.9: Solución al Apartado 2.c: **architecture** del circuito detector empleando sentencias de asignación concurrente.

La Figura 1.4 muestra el diagrama del circuito. Este circuito consta de puertas OR de dos entradas, puertas AND de dos entradas y puertas NOT. La **architecture** de las puertas AND de dos entradas, puertas OR de dos entradas, y puertas NOT se muestran respectivamente en Código VHDL 1.3, 1.4 y 1.10.

```
-------------------------------------
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 1.10: Puerta NOT lógica.

La **architecture** que describe la *estructura* del circuito se muestra en Código VHDL 1.11.

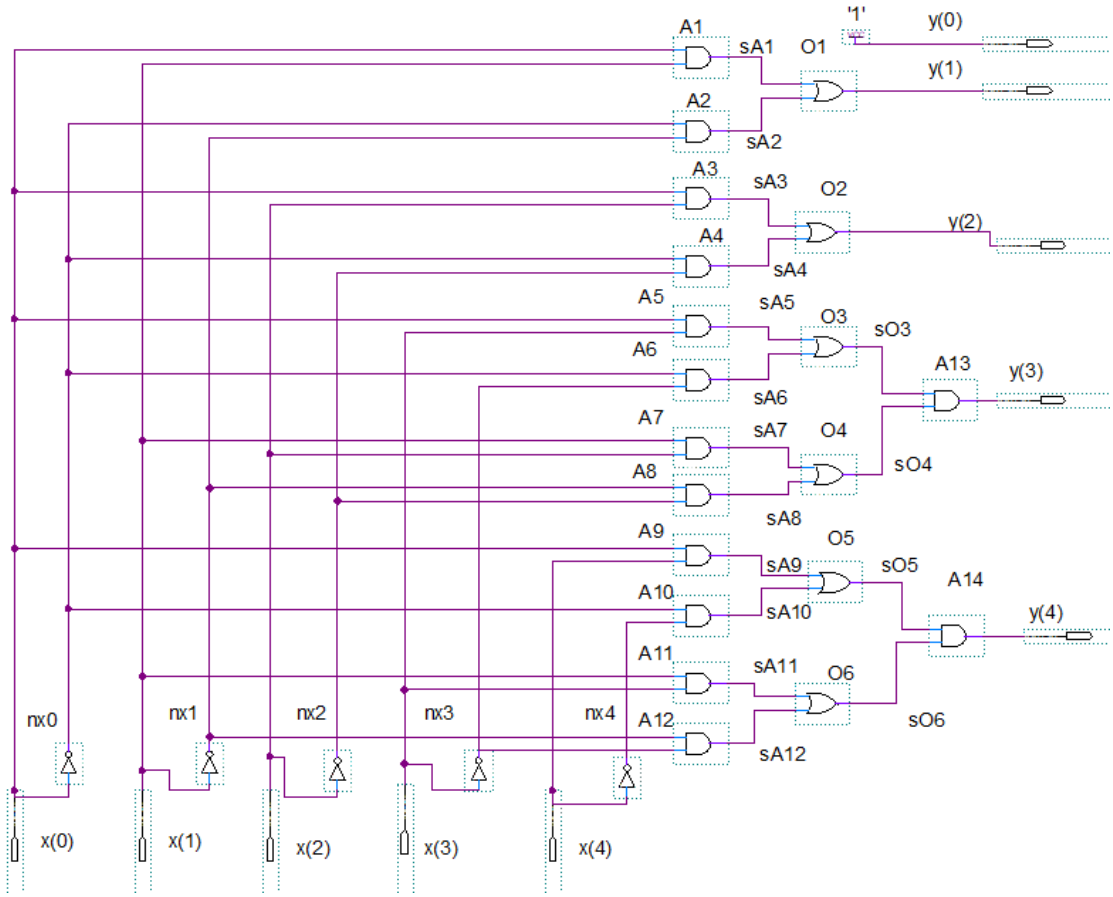

Figura 1.4: Diagrama circuital solución del Apartado 2.d.

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

architecture circuitoEST of circ is
   signal nx0, nx1, nx2, nx3, nx4: std_logic;
   signal sA1, sA2, sA3, sA4, sA5, sA6, sA7: std_logic;
   signal sA8, sA9, sA10, sA11, sA12: std_logic;
   signal s03, s04, s05, s06: std_logic;
-- Declaración de las clases de los componentes
   component and2 is
      port (   y0 : out std_logic ;
            x0, x1 : in std_logic);
   end component and2;
   component not1 is
      port ( y0 : out std_logic;
                x0 : in std_logic );
   end component not1;
   component or2 is
      port ( y0 : out std_logic;
           x0, x1 : in std_logic );
   end component or2;
begin
-- Instanciación y conexión de los componentes
   N0 : component not1 port map (nx0, x(0));
   N1 : component not1 port map (nx1, x(1));
   N2 : component not1 port map (nx2, x(2));
   N3 : component not1 port map (nx3, x(3));
   N4 : component not1 port map (nx4, x(4));
   A1 : component and2 port map (sA1, x(0), x(1));
   A2 : component and2 port map (sA2, nx0, nx1);
   A3 : component and2 port map (sA3, x(0), x(2));
   A4 : component and2 port map (sA4, nx0, nx2);
   A5 : component and2 port map (sA5, x(0), x(3));
   A6 : component and2 port map (sA6, nx0, nx3);
   A7: component and2 port map    (sA7, x(1), x(2));
   A8 : component and2 port map (sA8, nx1, nx2);
   A9 : component and2 port map (sA9, x(0), x(4));
   A10: component and2 port map (sA10, nx0, nx4);
   A11 : component and2 port map(sA11, x(1), x(3));
   A12 : component and2 port map(sA12, nx1, nx3);
   O1 : component or2 port map (y(1), sA1, sA2);
   O2 : component or2 port map (y(2), sA3, sA4);
   O3 : component or2 port map (s03, sA5, sA6);
   O4 : component or2 port map (s04, sA7, sA8);
   O5 : component or2 port map (s05, sA9, sA10);
   O6 : component or2 port map (s06, sA11, sA12);
   A13 : component and2 port map(y(3), s03, s04);
   A14 : component and2 port map(y(4), s05, s06);

   y(0) <= '1';

end architecture circuitoEST;
```

Código VHDL 1.11: Solución al Apartado 2.d: Diseño estructural del circuito detector.

El banco de pruebas del circuito se muestra en Código VHDL 1.12–1.13. El cronograma obtenido tras simular el banco de pruebas con cada uno de los 3 diseños del circuito de control realizado se muestran en las Figuras 1.5–1.7.

```
----------------------------------
-- Banco de pruebas
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity bp_circ is
end entity bp_circ;

architecture bp_circ of bp_circ is
   signal y       : std_logic_vector(4 downto 0);   -- Conectar salidas UUT
   signal x   : std_logic_vector(4 downto 0);   -- Conectar entradas UUT

   component circ is
      port ( y   : out std_logic_vector(4 downto 0);
             x   : in  std_logic_vector(4 downto 0));
   end component circ;

begin
   -- Instanciar y conectar UUT
   uut : component circ
         port map( y,x );
   gen_vec_test : process
      variable tx      : unsigned (4 downto 0); -- Vector de test
      variable esperado_Y     : std_logic_vector(4 downto 0);
      variable error_count          : integer     := 0;
   begin
```

Código VHDL 1.12: Solución al Apartado 2.e: Diseño del banco de pruebas.

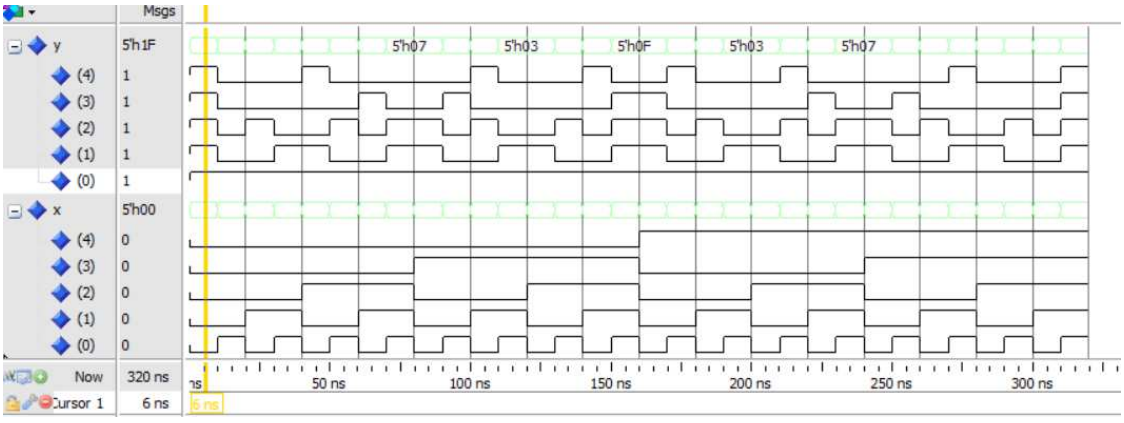

Figura 1.5: Cronograma obtenido al testear el circuito detector diseñado empleando un bloque **process**.

```
    report "Comienza la simulación";

    for i in 0 to 31 loop
        tx := TO_UNSIGNED(i,5);
        x <= std_logic_vector(tx);
        esperado_Y := "00001";
        if(tx(1 downto 0) = tx(0)&tx(1)) then
            esperado_Y(1) := '1';
        end if;
        if (tx(2 downto 0) = tx(0)&tx(1)&tx(2)) then
            esperado_Y(2) := '1';
        end if;
        if (tx(3 downto 0) =  tx(0)&tx(1)&tx(2)&tx(3)) then
            esperado_Y(3) := '1';
        end if;
        if (tx =  tx(0)&tx(1)&tx(2)&tx(3)&tx(4)) then
            esperado_Y(4) := '1';
        end if;
        wait for 10 ns;
        if ( esperado_Y /= y ) then
          report "ERROR en la salida valida. Valor esperado: "  &
                  std_logic'image(esperado_Y(4))                &
                  std_logic'image(esperado_Y(3))                &
                  std_logic'image(esperado_Y(2))                &
                  std_logic'image(esperado_Y(1))                &
                  std_logic'image(esperado_Y(0))                &

                  ", valor actual: "                              &
                  std_logic'image(y(4))                         &
                  std_logic'image(y(3))                         &
                  std_logic'image(y(2))                         &
                  std_logic'image(y(1))                         &
                  std_logic'image(y(0))                         &

                  " en el instante: "                             &
                  time'image(now);
          error_count := error_count + 1;
        end if;
      end loop;
    report   "ERROR: Hay "              &
               integer'image(error_count) &
               " errores.";
    wait;
  end process gen_vec_test;
end architecture bp_circ;
-----------------------------------
```

Código VHDL 1.13: Solución al Apartado 2.e: Continuación del diseño del banco de pruebas.

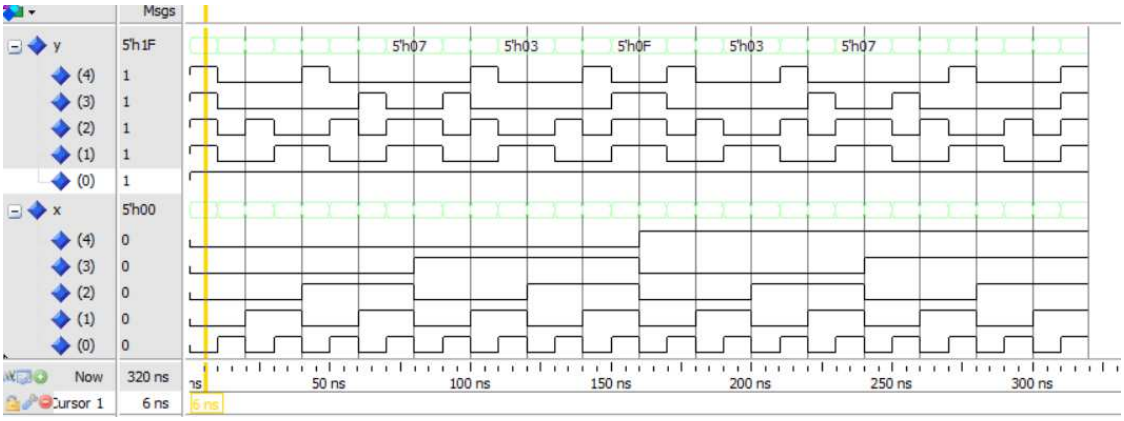

Figura 1.6: Cronograma obtenido al testear el circuito detector diseñado empleando sentencias de asignación concurrentes.

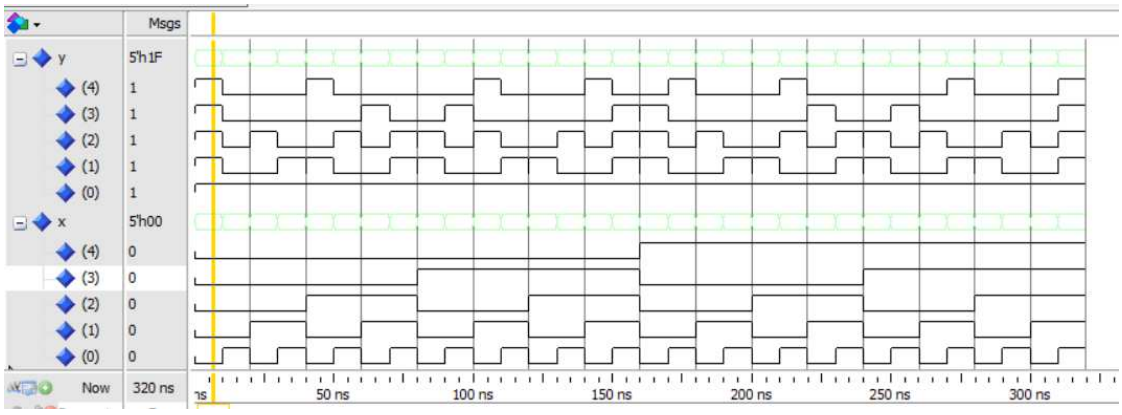

Figura 1.7: Cronograma obtenido al testear el circuito detector descrito empleando puertas lógicas.