INGENIERÍA DE COMPUTADORES 3

Solución al Trabajo Práctico - Septiembre de 2021

EJERCICIO 1

Dada las dos funciones lógicas (`F` y `G`) mostradas a continuación, que dependen de 3 variables (`x`, `y` y `z`):

```
F = not x or (y and z)
G = x and y and z
```

1.a) (0.5 puntos) Escriba en VHDL la **entity** del circuito.

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

1.c) (0.5 puntos) Dibuje el diagrama de un circuito que implemente estas dos funciones lógicas al nivel de puertas lógicas. 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) (1 punto) 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 del circuito diseñado en los Apartados 1.b y 1.d.

Solución al Ejercicio 1

La **entity** del circuito se muestra en Código VHDL 1.1.

```
-----------------------------------
library IEEE;
use IEEE.std_logic_1164.all;
entity funcLog_F_G is
   port ( F, G     : out std_logic;
          x, y, z : in std_logic );
end entity funcLog_F_G;
-----------------------------------
```

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

El Código VHDL 1.2 muestra la **architecture** del circuito describiendo su comportamiento.

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

architecture funcLog_F_G_Comp of funcLog_F_G is
begin
    F <= not x or (y and z);
    G <= x and y and z;
end architecture funcLog_F_G_Comp;
-----------------------------------
```

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

La Figura 1.1 muestra el diagrama del circuito empleando dos puertas AND, un inversor y una puerta OR.

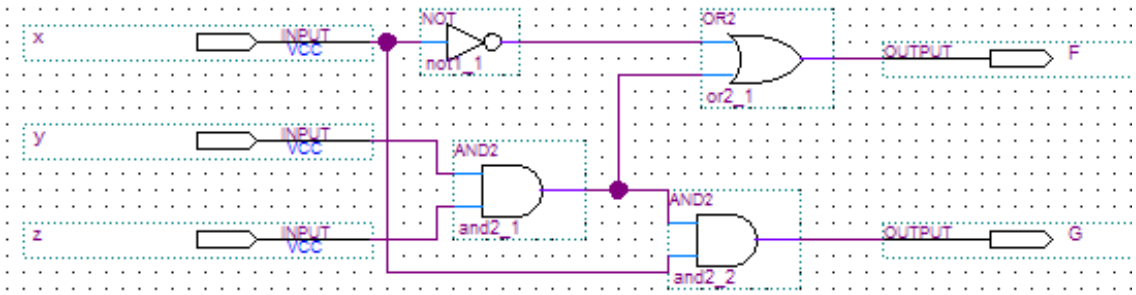

Figura 1.1: Solución al Apartado 1.c: diagrama al nivel de puertas lógicas.

El código VHDL de la **entity** y la **architecture** de estas tres puertas lógicas se muestra en Código VHDL 1.3, 1.4 y 1.5, respectivamene.

```
-------------------------------------
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 lógica.

```
-------------------------------------
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 1.4: Puerta NOT lógica.

```
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 1.5: Puerta OR lógica.

El Código VHDL 1.6 muestra la **architecture** del circuito describiendo estructura.

El código VHDL del banco de pruebas del circuito se muestra en Código VHDL 1.7.

```
-----------------------------------
library IEEE;
use IEEE.std_logic_1164.all;
architecture funcLog_F_G_Estruc of funcLog_F_G is
   signal not_x, yz: 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
   not1_1 : component not1 port map (not_x, x);
   and2_1 : component and2 port map (yz, y, z);
   or2_1 : component or2 port map (F, not_x, yz);
   and2_2 : component and2 port map (G, x, yz);
end architecture funcLog_F_G_Estruc;
-----------------------------------
```

Código VHDL 1.6: Solución al Apartado 1.d: **architecture** del circuito describiendo su estructura.

Los dos cronogramas obtenidos al simular el banco de pruebas con los diseños obtenidos en los Apartados 1.b y 1.d se muestran, respectivamente, en las Figuras 1.2 y 1.3.

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

entity bp_funcLog_F_G is
end entity bp_funcLog_F_G;

architecture bp_funcLog_F_G of bp_funcLog_F_G is
   signal F, G : std_logic; -- Conectar salidas UUT
   signal x0, x1, x2 : std_logic; -- Conectar entradas UUT

   component funcLog_F_G is port
    ( F, G : out std_logic;
      x, y, z: in std_logic);
   end component funcLog_F_G;

begin
-- Instanciar y conectar UUT
   uut : component funcLog_F_G port map
   ( F => F, G => G,
   x => x0, y => x1, z => x2);

   gen_vec_test : process
     variable test_in : unsigned (2 downto 0); -- Vector de test
   begin
     test_in := B"000";
     for count in 0 to 7 loop
       x2 <= test_in(2);
       x1 <= test_in(1);
       x0 <= test_in(0);
       wait for 10 ns;
       test_in := test_in + 1;
     end loop;
     wait;
   end process gen_vec_test;
end architecture bp_funcLog_F_G;
-------------------------------------
```

Código VHDL 1.7: Solución al Apartado 1.e: banco de pruebas del circuito.

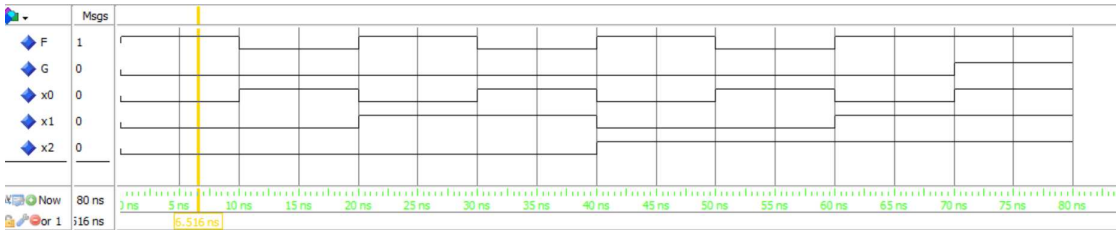

Figura 1.2: Cronograma del Apartado 1.e comprobando el comportamiento del circuito diseñado en el Apartado 1.b.

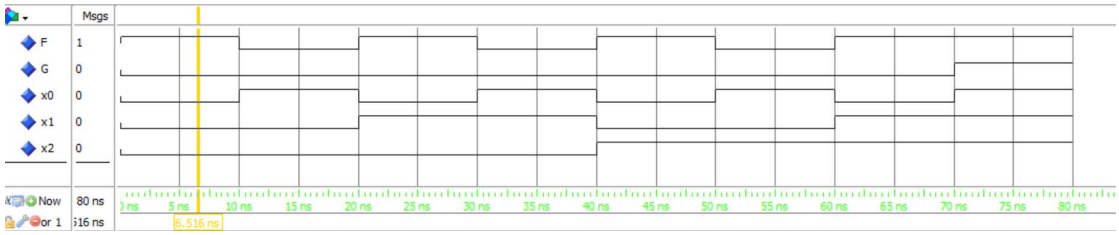

Figura 1.3: Cronograma del Apartado 1.e comprobando el comportamiento del circuito diseñado en el Apartado 1.d.

EJERCICIO 2

2.a (1 punto) Diseñe en VHDL la **architecture** de un decodificador 2 a 4 con entrada enable activa a nivel alto, describiendo la estructura del circuito usando para ello solo puertas AND de tres entradas y puertas NOT. Debe proporcionar el código VHDL de la **entity** y **architecture** de la puerta AND de tres entradas y la puerta NOT. El diseño estructural del decodificador se ha de realizar instanciando y conectando las puertas lógicas AND y NOT diseñadas. En la memoria ha de mostrar, además de todo el código VHDL diseñado, el diagrama circuital del diseño realizado.

La **entity** del circuito y la tabla de la verdad se muestran a continuación.

```
entity decodificador2a4 is
   port (  y0, y1, y2, y3  : out std_logic;
           w0, w1          : in std_logic;
           En              : in  std_logic);
end entity decodificador2a4;
```

En	w1	w0	y0	y1	y2	y3
'1'	'0'	'0'	'1'	'0'	'0'	'0'
'1'	'0'	'1'	'0'	'1'	'0'	'0'
'1'	'1'	'0'	'0'	'0'	'1'	'0'
'1'	'1'	'1'	'0'	'0'	'0'	'1'
'0'	'X'	'X'	'0'	'0'	'0'	'0'

Donde el valor 'X' en la tabla indica que se produce esa salida tanto para el valor '0' como el valor '1' de dicho bit.

2.b (1 punto) Programe en VHDL un banco de pruebas que testee todas las posibles entradas al circuito que ha diseñado en el Apartado 2.a. 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. Incluya en la memoria el cronograma obtenido al realizar la simulación del banco de pruebas del circuito diseñado en el Apartados 2.a.

2.c (2 puntos) Diseñe un decodificador 3 a 8 con entrada enable activa a nivel alto, describiendo su estructura empleando para ello únicamente dos decodificadores 2 a 4 como el diseñado en el Apartado 2.a, puertas NOT y puertas AND de dos entradas. El comportamiento de este decodificador es

análogo al descrito en el Apartado 2.a para el decodificador 2 a 4, pero con diferente número de entradas y salidas. Debe proporcionar el código VHDL de la **entity** y **architecture** de la puerta AND de dos entradas. El diseño estructural del decodificador se ha de realizar instanciando y conectando los decodificadores 2 a 4, las puertas lógicas AND y NOT diseñadas. En la memoria ha de mostrar, además de todo el código VHDL diseñado, el diagrama circuital del diseño realizado.

La **entity** del circuito se muestra a continuación.

```
entity decodificador3a8 is
   port (y0, y1, y2, y3, y4, y5, y6, y7  : out std_logic;
         w0, w1, w2          : in std_logic;
         En              : in  std_logic);
end entity decodificador3a8;
```

2.d) (2 puntos) Programe en VHDL un banco de pruebas que testee el circuito diseñado en el Apartado 2.c para todos los posibles valores de las entradas. 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.

Incluya en la memoria el cronograma obtenido al realizar la simulación del banco de pruebas del circuito diseñado en el Apartado 2.c.

Solución al Ejercicio 2

La Figura 1.4 muestra el diagrama del circuito implementado empleando cuatro puertas AND de tres entradas y 2 puertas NOT.

El código VHDL de la **entity** y la **architecture** de estas dos puertas lógicas se muestra en Código VHDL 1.8 y 1.4, respectivamente.

El Código VHDL 1.9 muestra el diseño del circuito describiendo su estructura. El Código VHDL 1.10–1.11 muestra el código VHDL banco de pruebas del circuito decodificador 2 a 4.

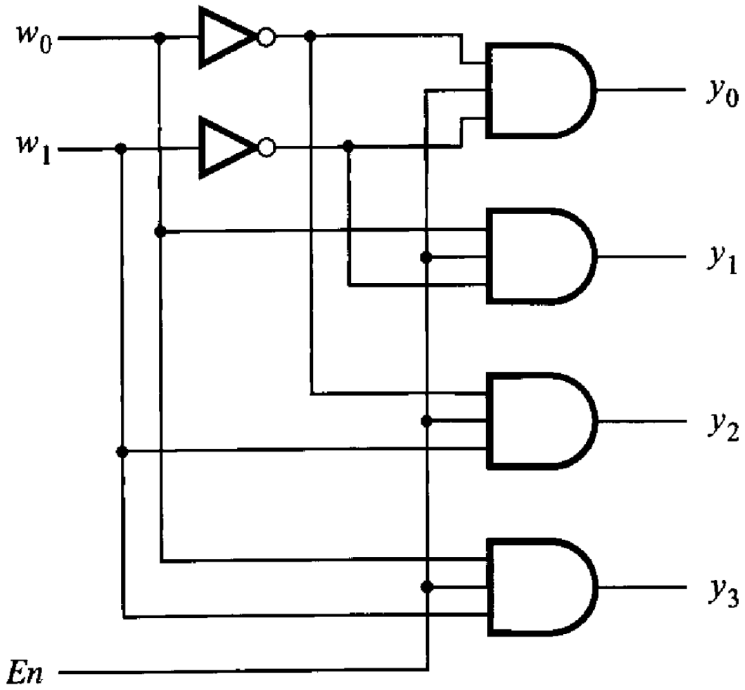

Figura 1.4: Solución al Apartado 2.a: diagrama al nivel de puertas lógicas.

```
---------------------------------------
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 <= x0 and x1 and x2;
end architecture and3;
---------------------------------------
```

Código VHDL 1.8: Puerta AND lógica de tres entradas.

```
-------------------------------------
--Diseño estructural decod. 2 a 4
library IEEE;
use IEEE.std_logic_1164.all;
architecture decodificador2a4 of decodificador2a4 is
   signal wn1, wn0 : 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;


begin
-- Instanciación y conexión de los componentes
   N1 : component not1 port map (wn1, w1);
   N0 : component not1 port map (wn0, w0);
   A0 : component and3 port map (y0, En, wn1, wn0);
   A1 : component and3 port map (y1, En, wn1, w0);
   A2 : component and3 port map (y2, En, w1, wn0);
   A3 : component and3 port map (y3, En, w1, w0);

end architecture decodificador2a4;
---------------------------------
```

Código VHDL 1.9: Diseño del decodificador 2 a 4 describiendo su estructura.

```
--------------------------------------------------
-- Banco de pruebas del codificador 4:2 con prioridad
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity bp_decod2a4 is
    constant MAX_COMB : integer := 8;       -- Num. combinaciones entrada
    constant DELAY    : time    := 10 ns;   -- Retardo usado en el test
end entity bp_decod2a4;


architecture bp_decod2a4 of bp_decod2a4 is
    -- Salidas UUT
    signal y0, y1, y2, y3          : std_logic;
     -- Entradas UUT
    signal w0, w1, En  : std_logic;

    component decodificador2a4 is
        port (   y0, y1, y2, y3   : out std_logic;
                 w0, w1           : in std_logic;
                 En               : in  std_logic );
    end component decodificador2a4;

begin   -- Cuerpo de la arquitectura
  UUT : component decodificador2a4 port map
          (y0, y1, y2, y3, w0, w1, En);

  main : process is
    variable temp : unsigned (3 downto 0); -- Usado en los cálculos
    variable esperado_salida : std_logic_vector(3 downto 0);
    variable error_count          : integer     := 0;
  begin
    report "Comienza la simulación";
    -- Generar todos los posibles valores de entrada
    for i in 0 to (MAX_COMB-1) loop
      temp := TO_UNSIGNED(i,4);
      En <= std_logic(temp(2));
      w1 <= std_logic(temp(1));
      w0 <= std_logic(temp(0));
      -- Calcular el valor esperado
      if (i<4) then
          esperado_salida := "0000";
      else
          if( i =4) then esperado_salida := "0001";
          elsif (i = 5) then esperado_salida := "0010";
          elsif (i=6) then esperado_salida := "0100";
          else esperado_salida := "1000";
          end if;
      end if;

      wait for DELAY; -- Espera y compara con las salidas de UUT
```

Código VHDL 1.10: Banco de pruebas del decodificador 2 a 4 (1/2).

```
        if ( esperado_salida /= y3&y2&y1&y0 ) then

            report "ERROR en la salida . Valor esperado:" &
                    std_logic'image(esperado_salida(3))          &
                    std_logic'image(esperado_salida(2))          &
                    std_logic'image(esperado_salida(1))          &
                    std_logic'image(esperado_salida(0))          &
                    ",valor actual:"                                    &

                    std_logic'image(y3)                          &
                    std_logic'image(y2)                          &

                    std_logic'image(y1)                          &
                    std_logic'image(y0)                          &
                    " en el instante:"                                  &
                    time'image(now);
            error_count := error_count + 1;
        end if;

    end loop; -- Final del bucle for de posibles valores de entrada

    -- Informe del número total de errores
    if (error_count = 0) then
        report "Simulación finalizada sin errores";
    else
        report   "ERROR: Hay "              &
                  integer'image(error_count) &
                  " errores.";
    end if;

    wait;      -- Final de la simulación
  end process main;
end architecture bp_decod2a4;
-------------------------------------------------
```

Código VHDL 1.11: Continuación del banco de pruebas del decodificador 2 a 4 (2/2).

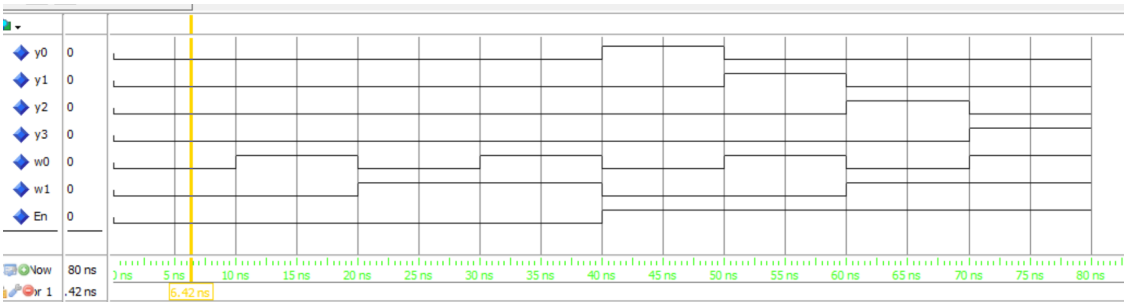

Figura 1.5: Cronograma del Apartado 2.b comprobando el comportamiento del circuito diseñado en el Apartado 2.a.

En la Figura 1.5 se muestra el cronograma obtenido al simular el banco de pruebas del decodificador 2 a 4.

La Figura 1.6 muestra el diagrama del circuito implementado empleando dos decodificadores 2 a 4, 2 puertas AND de dos entradas y 1 puerta NOT.

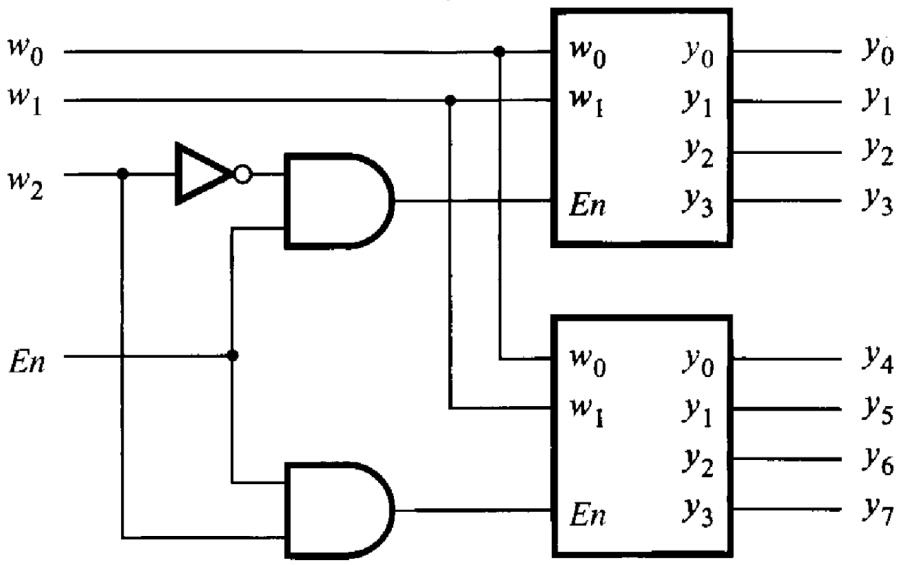

Figura 1.6: Solución al Apartado 2.c: diagrama estructural del decodificador 3 a 8.

El código VHDL de las puertas AND de dos entradas y la puerta NOT se muestra en Código VHDL 1.3 y 1.4, respectivamente. El Código VHDL 1.12 muestra el código VHDL del decodificador 3 a 8.

El banco de pruebas del decodificador 3 a 8 se muestra en Código VHDL 1.13–1.14. El cronograma obtenido al simular el banco de pruebas usando como circuito a testear el diseño del Apartado 2.c se muestra en la Figura 1.7.

```
-------------------------------------
--Diseño estructural decod. 3 a 8
library IEEE;
use IEEE.std_logic_1164.all;
architecture decodificador3a8 of decodificador3a8 is
   signal wn2, En0, En1 : 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 decodificador2a4 is
       port (   y0, y1, y2, y3   : out std_logic;
                w0, w1              : in std_logic;
                En                    : in    std_logic);
    end component decodificador2a4;
begin
-- Instanciación y conexión de los componentes
   N2 : component not1 port map (wn2, w2);
   A0 : component and2 port map (En0, wn2, En);
   A1 : component and2 port map (En1, w2, En);
   D0 : component decodificador2a4 port map (y0, y1, y2, y3, w0, w1, En0);
   D1 : component decodificador2a4 port map (y4, y5, y6, y7, w0, w1, En1);

end architecture decodificador3a8;
-----------------------------------
```

Código VHDL 1.12: Decodificador 3 a 8.

```
----------------------------------------------------
-- Banco de pruebas del decodificador 3 a8
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity bp_decod3a8 is
    constant MAX_COMB : integer := 16;      -- Num. combinaciones entrada
    constant DELAY    : time    := 10 ns;  -- Retardo usado en el test
end entity bp_decod3a8;



architecture bp_decod3a8 of bp_decod3a8 is
    -- Salidas UUT
    signal y0, y1, y2, y3, y4, y5, y6, y7          : std_logic;
     -- Entradas UUT
    signal w0, w1, w2, En : std_logic;

    component decodificador3a8 is
        port (  y0, y1, y2, y3, y4, y5, y6, y7  : out std_logic;
                w0, w1, w2                      : in std_logic;
                En                              : in  std_logic );
    end component decodificador3a8;

begin   -- Cuerpo de la arquitectura
  UUT : component decodificador3a8 port map
          (y0, y1, y2, y3, y4, y5, y6, y7, w0, w1, w2, En);

  main : process is
    variable temp : unsigned (7 downto 0); -- Usado en los cálculos
    variable esperado_salida : std_logic_vector(7 downto 0);
    variable error_count           : integer    := 0;
  begin
    report "Comienza la simulación";
    -- Generar todos los posibles valores de entrada
    for i in 0 to (MAX_COMB-1) loop
      temp := TO_UNSIGNED(i,8);
      En <= std_logic(temp(3));
      w2 <= std_logic(temp(2));
      w1 <= std_logic(temp(1));
      w0 <= std_logic(temp(0));
      -- Calcular el valor esperado
      if (i<8) then
          esperado_salida:= "00000000";
      else
          esperado_salida:= "00000000";
```

Código VHDL 1.13: Solución al Apartado 2.d: banco de pruebas del circuito (1/2).

```
            if( i = 8) then esperado_salida(0) := '1';
            elsif (i = 9) then esperado_salida(1) := '1';
            elsif (i=10) then esperado_salida(2) := '1';
            elsif (i = 11) then esperado_salida(3) := '1';
            elsif (i=12) then esperado_salida(4) := '1';
            elsif (i = 13) then esperado_salida(5) := '1';
            elsif (i=14) then esperado_salida(6):= '1';
            else esperado_salida(7) := '1';
            end if;
        end if;


        wait for DELAY; -- Espera y compara con las salidas de UUT

        if ( esperado_salida /= y7&y6&y5&y4&y3&y2&y1&y0 ) then

             report "ERROR en la salida . Valor esperado:" &
                    std_logic'image(esperado_salida(7))          &
                    std_logic'image(esperado_salida(6))          &
                    std_logic'image(esperado_salida(5))          &
                    std_logic'image(esperado_salida(4))          &

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


                    std_logic'image(y7)                          &
                    std_logic'image(y6)                          &
                    std_logic'image(y5)                          &
                    std_logic'image(y4)                          &
                    std_logic'image(y3)                          &
                    std_logic'image(y2)                          &
                    std_logic'image(y1)                          &
                    std_logic'image(y0)                          &
                    " en el instante:"                                  &
                    time'image(now);
             error_count := error_count + 1;
        end if;

    end loop; -- Final del bucle for de posibles valores de entrada

    -- Informe del número total de errores
    if (error_count = 0) then
         report "Simulación finalizada sin errores";
    else
         report   "ERROR: Hay "                &
                  integer'image(error_count) &
                  " errores.";
    end if;

    wait;      -- Final de la simulación
  end process main;
end architecture bp_decod3a8;
-------------------------------------------------
```

Código VHDL 1.14: Solución al Apartado 2.d: banco de pruebas del circuito (2/2).

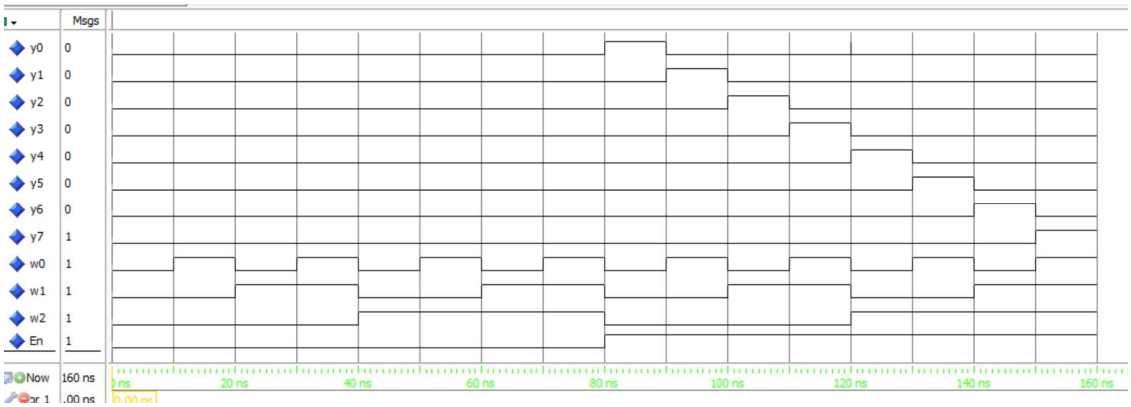

Figura 1.7: Cronograma del Apartado 2.d comprobando el comportamiento del circuito diseñado en el Apartado 2.c.