INGENIERÍA DE COMPUTADORES 3

Solución al Trabajo Práctico - Septiembre de 2018

EJERCICIO 1 (3 PUNTOS)

Se desea diseñar un circuito digital que implemente las funciones `F` y `G` cuyas tablas de verdad se muestran a continuación, que dependen de las tres variables `x`, `y` y `z`:

x	y	z	F	G
'0'	'0'	'0'	'1'	'1'
'0'	'0'	'1'	'0'	'1'
'0'	'1'	'0'	'0'	'0'
'0'	'1'	'1'	'1'	'1'
'1'	'0'	'0'	'0'	'1'
'1'	'0'	'1'	'1'	'1'
'1'	'1'	'0'	'0'	'1'
'1'	'1'	'1'	'1'	'1'

1.a) (0.25 puntos) Obtenga las funciones lógicas `F` y `G` 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 `x`, `y` y `z`, y dos salidas `F` y `G`.

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

1.c) (0.25 punto) 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 funcFG is
   port ( F, G: out std_logic;
          x, y, z : in std_logic );
end entity funcFG;
-------------------------------------
```

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

```
  --F= x'y'z'+yz+xz
--G=(x'yz')'
-------------------------------------
architecture Comp of funcFG is
begin
  F <= ((not x) and (not y) and (not z)) or (y and z) or (x and z);
  G <= not(not x and y and not z);
end architecture Comp;
-------------------------------------
```

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

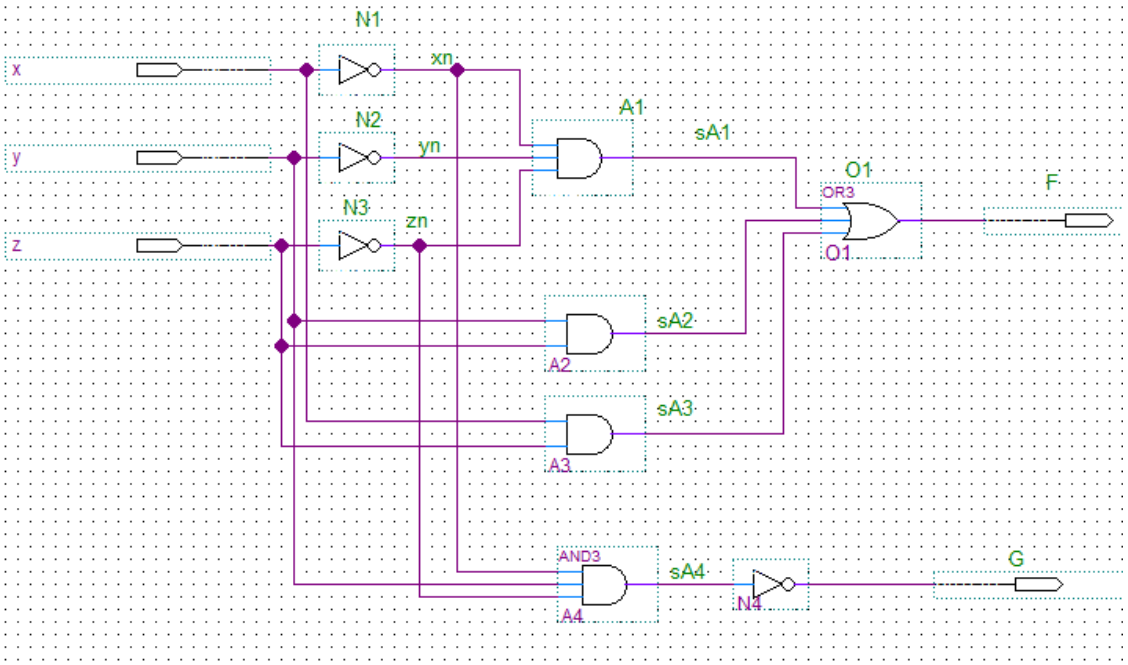

Figura 1.1: Solución al Apartado 1.c: diagrama al nivel de puertas lógicas.

La Figura 1.1 muestra el diagrama del circuito implementado empleando dos puertas AND de dos entradas, dos puertas AND de tres entradas, tres puertas NOT y una puerta OR de tres entradas.

El código VHDL de la **entity** y la **architecture** de estas cuatro puertas lógicas se muestra en Código VHDL 1.3, 1.4, 1.5 y 1.12, respectivamente. El Código VHDL 1.7 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.

```
----------------------------------
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 1.4: Puerta AND de tres entradas.

```
----------------------------------
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 1.5: Puerta NOT lógica.

```
-------------------------------------
-- OR de 3 entradas
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 1.6: Puerta OR lógica.

El código VHDL del banco de pruebas del circuito se muestra en Código VHDL 1.8. 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.

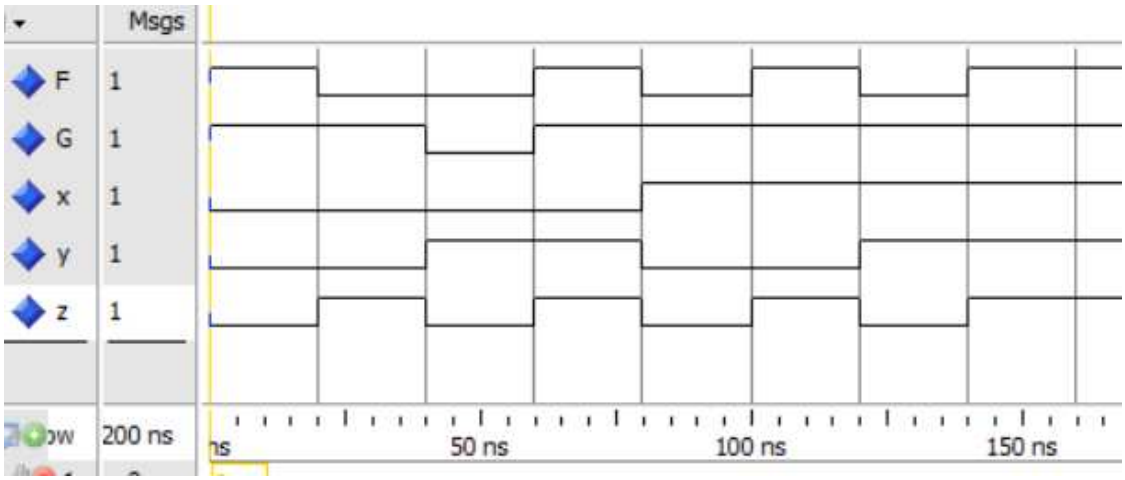

Figura 1.2: Cronograma del banco de pruebas aplicado al diseño del Apartado 1.b.

```
----------------------------------
library IEEE;
use IEEE.std_logic_1164.all;
architecture circuito_Estruc of funcFG is
   signal xn, yn, zn: std_logic;
   signal sA1, sA2, sA3,sA4: 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 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 or3 is
      port ( y0 : out std_logic;
           x0, x1, x2 : in std_logic );
   end component or3;
begin
-- Instanciación y conexión de los componentes
   N1 : component not1 port map (xn, x);
   N2 : component not1 port map (yn, y);
   N3 : component not1 port map (zn, z);
   A1 : component and3 port map (sA1, xn, yn, zn);
   A2 : component and2 port map (sA2, y, z);
   A3 : component and2 port map (sA3, x, z);
   A4 : component and3 port map (sA4, xn, y, zn);
   O1 : component or3 port map (F, sA1, sA2, sA3);
   N4 : component not1 port map (G, sA4);
end architecture circuito_Estruc;
---------------------------------
```

Código VHDL 1.7: 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_funcFG is
    constant DELAY      : time      := 20 ns; -- Retardo usado en el test
end entity bp_funcFG;

architecture bp_funcFG of bp_funcFG is
    signal F, G : std_logic;
    signal x, y, z   : std_logic;

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

begin
    UUT : component funcFG port map
          (F, G, x, y, z);

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);
          x <= std_logic(valor(2));
          y <= std_logic(valor(1));
          z <= std_logic(valor(0));
          wait for DELAY;
       end loop;
       wait;      -- Final de la simulación
   end process vec_test;
end architecture bp_funcFG;
-------------------------------------------------
```

Código VHDL 1.8: Solución al Apartado 1.e: banco de pruebas del circuito.

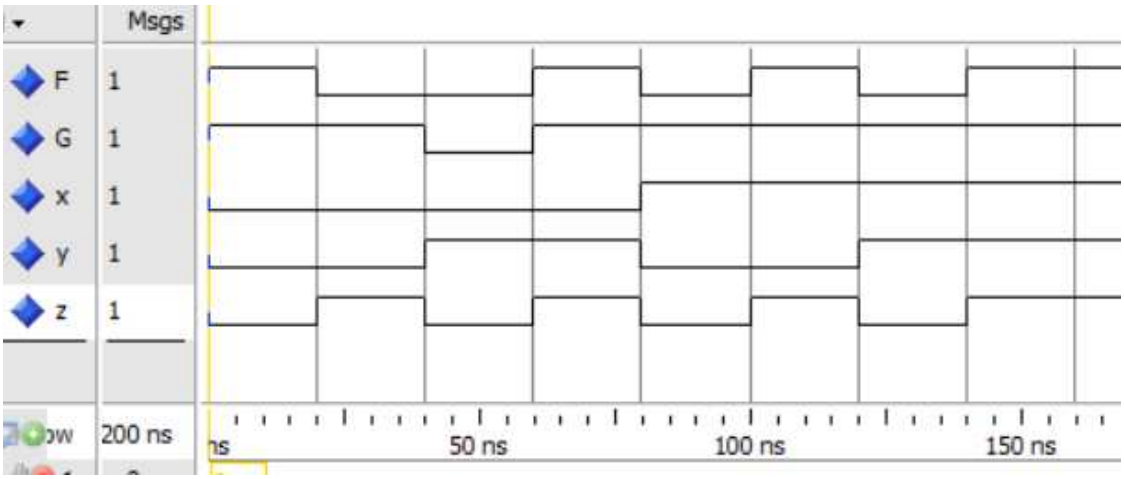

Figura 1.3: Cronograma del banco de pruebas aplicado al diseño del Apartado 1.d.

EJERCICIO 2 (7 PUNTOS)

Se pretende diseñar un circuito combinacional para el control de un aparato de aire acondicionado que consta de una bomba de frío, una bomba de calor y un ventilador. El circuito tiene 6 señales de entrada: 2 señales de entrada de un bit procedentes de sensores y 4 señales de entrada de un bit cuyo valor depende de las acciones del usuario del aparato de aire acondicionado. El circuito tiene 3 señales de salida de un bit cuyo objetivo es encender/apagar la bomba de frío, de calor y el ventilador.

A continuación se describe el significado de las 6 señales de entrada del circuito:

- Señal `temp_low`: tiene valor '1' sólo si la temperatura baja por debajo de cierta temperatura de referencia.
- Señal `temp_high`: tiene valor '1' sólo si la temperatura sube por encima de cierta temperatura de referencia.
- Señal `auto_temp`: está a '1' sólo si se desea controlar la temperatura ambiente de modo automático.
- Señal `manual_heat`: tiene valor '1' sólo si el usuario quiere encender manualmente la bomba de calor.
- Señal `manual_cool`: tiene valor '1' sólo si el usuario quiere encender manualmente la bomba de frío.
- Señal `manual_fan`: tiene valor '1' sólo si el usuario quiere encender manualmente el ventilador.

A continuación se describe el significado de las 3 las señales de salida del circuito:

- Señal `heater_on`: se pone a '1' sólo si se cumple cualquiera de las dos condiciones siguientes:
  1. La señal `temp_low` tiene valor '1' y el aparato se encuentra en el modo de control automático.
  2. La señal `manual_heat` tiene valor '1'.
- Señal `cooler_on`: se pone a '1' sólo si se cumple cualquiera de las dos siguientes condiciones:
  1. La señal `temp_high` tiene valor '1' y el aparato se encuentra en el modo de control automático.

2. La señal `manual_cool` tiene valor '1'.

– Señal `fan_on`: se pone a '1' sólo si se cumple cualquiera de las tres condiciones siguientes:

1. La bomba de frío está encendida.
2. La bomba de calor está encendida.
3. El usuario enciende manualmente el ventilador.

2.a) (0.5 puntos) Escriba en VHDL la **entity** del circuito de control. Todas las señales de entrada y de salida del circuito han de ser del tipo `std_logic` y han de tener 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) (2 puntos) 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) (0.5 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. Compruebe mediante inspección visual 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 de control se muestra en Código VHDL 1.9. La **architecture** que describe el comportamiento del circuito empleando un bloque **process** y sentencias **if** se muestra en Código VHDL 1.10. La **architecture** que describe el comportamiento del circuito empleando

únicamente sentencias de asignación concurrentes se muestra en Código VHDL 1.11.

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

entity controlAC is
   port ( heater_on, cooler_on, fan_on: out std_logic;
          temp_low, temp_high, auto_temp,
          manual_heat, manual_cool, manual_fan : in std_logic );
end entity controlAC;
-------------------------------------
```

Código VHDL 1.9: Solución al Apartado 2.a: **entity** del circuito de control.

```
-------------------------------------
--Sistema de control con IF
-------------------------------------
architecture Comp of controlAC is
begin
  process(temp_low, temp_high, auto_temp,
          manual_heat, manual_cool, manual_fan)
     variable heater_on_tmp: std_logic;
     variable cooler_on_tmp: std_logic;
  begin
     heater_on_tmp := '0';
     cooler_on_tmp := '0';
     fan_on <= '0';
     if (((temp_low ='1')and (auto_temp = '1')) or (manual_heat = '1')) then
        heater_on_tmp := '1';
     end if;
     if (((temp_high = '1') and (auto_temp = '1')) or (manual_cool = '1')) then
        cooler_on_tmp := '1';
     end if;
     if ( (heater_on_tmp = '1') or (cooler_on_tmp = '1') or (manual_fan =
'1')) then
        fan_on <='1';
     end if;
     heater_on <= heater_on_tmp;
     cooler_on <= cooler_on_tmp;
end process;
end architecture Comp;
-------------------------------------
```

Código VHDL 1.10: Solución al Apartado 2.b: **architecture** del circuito de control empleando un bloque **process**.

La Figura 1.4 muestra el diagrama del circuito de control. Este circuito consta de dos puertas OR de dos entradas, una puerta OR de 3 entradas y 2 puertas AND de dos entradas. La **architecture** de las puertas OR de dos

```
-----------------------------------
--Sistema de control emplenado
--sentencias asignacion concurrentes
-----------------------------------
architecture Concurrente of controlAC is
   signal heater_on_tmp, cooler_on_tmp: std_logic;
begin
   heater_on_tmp <= (temp_low and auto_temp) or manual_heat;
   cooler_on_tmp <= (temp_high and auto_temp) or manual_cool;
   heater_on <= heater_on_tmp;
   cooler_on <= cooler_on_tmp;
   fan_on <= heater_on_tmp or cooler_on_tmp or manual_fan;
end architecture Concurrente;
-----------------------------------
```

Código VHDL 1.11: Solución al Apartado 2.c: **architecture** del circuito de control empleando sentencias de asignación concurrente.

entradas, OR de tres entradas y AND de dos entradas se muestran respectivamente en Código VHDL 1.12, 1.6, y 1.3.

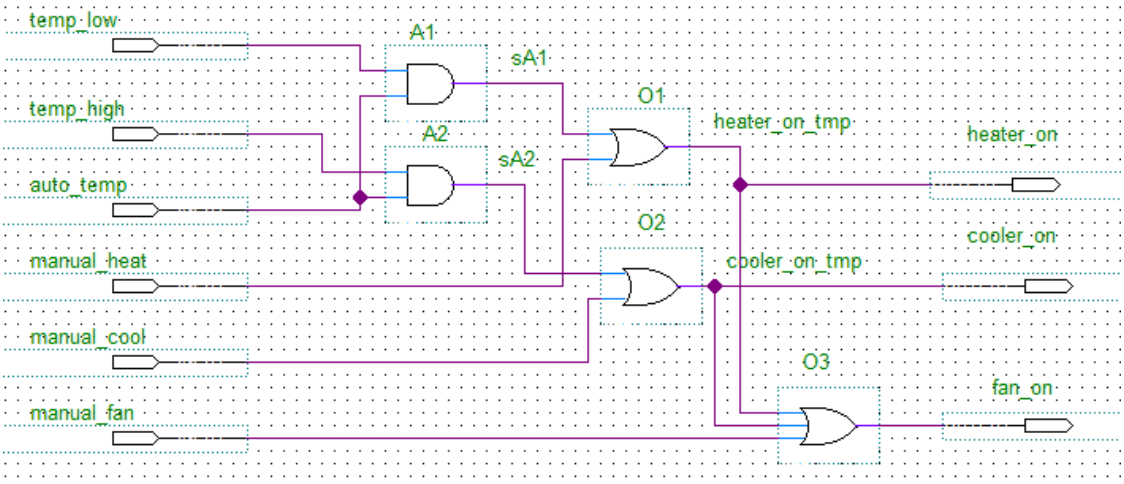

Figura 1.4: Diagrama circuital solución del Apartado 2.d.

La **architecture** que describe la *estructura* del circuito se muestra en Código VHDL 1.13.

```
-------------------------------------
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 1.12: Puerta lógica OR de 2 entradas.

```
-------------------------------------
--Sistema de control describiendo
--su estructura
-------------------------------------
architecture Estructural of controlAC is
   signal sA1, sA2: std_logic;
   signal heater_on_tmp, cooler_on_tmp: 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 or2 is
      port ( y0 : out std_logic;
               x0, x1 : in std_logic );
   end component or2;
   component or3 is
      port ( y0 : out std_logic;
           x0, x1, x2 : in std_logic );
   end component or3;
begin
-- Instanciación y conexión de los componentes
   A1 : component and2 port map (sA1, temp_low, auto_temp);
   A2 : component and2 port map (sA2, temp_high, auto_temp);
   O1 : component or2 port map (heater_on_tmp, sA1, manual_heat);
   O2 : component or2 port map (cooler_on_tmp, sA2, manual_cool);
   O3 : component or3 port map (fan_on, heater_on_tmp, cooler_on_tmp,
manual_fan);
   heater_on <= heater_on_tmp;
   cooler_on <= cooler_on_tmp;
end architecture Estructural;
-------------------------------------
```

Código VHDL 1.13: Solución al Apartado 2.d: Diseño estructural del circuito de control.

El banco de pruebas del circuito se muestra en Código VHDL 1.14. 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 del sistema de control
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

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

architecture bp_control of bp_control is
    signal heater_on, cooler_on, fan_on: std_logic;
    signal temp_low, temp_high, auto_temp  : std_logic;
    signal manual_heat, manual_cool, manual_fan  : std_logic;

    component controlAC is
     port ( heater_on, cooler_on, fan_on: out std_logic;
            temp_low, temp_high, auto_temp,
            manual_heat, manual_cool, manual_fan : in std_logic );
    end component controlAC;


begin
    UUT : component controlAC port map
          (heater_on, cooler_on, fan_on, temp_low, temp_high,
           auto_temp, manual_heat, manual_cool, manual_fan);

vec_test : process is
      variable valor  : unsigned (5 downto 0);
    begin
       -- Generar todos los posibles valores de entrada
       for i in 0 to 63 loop
          valor := to_unsigned(i,6);
          temp_low <= std_logic(valor(5));
          temp_high <= std_logic(valor(4));
          auto_temp <= std_logic(valor(3));
          manual_heat <= std_logic(valor(2));
          manual_cool <= std_logic(valor(1));
          manual_fan <= std_logic(valor(0));
          wait for DELAY;
       end loop;
       wait;     -- Final de la simulación
   end process vec_test;
end architecture bp_control;
-------------------------------------------------
```

Código VHDL 1.14: Solución al Apartado 2.e: Diseño del banco de pruebas.

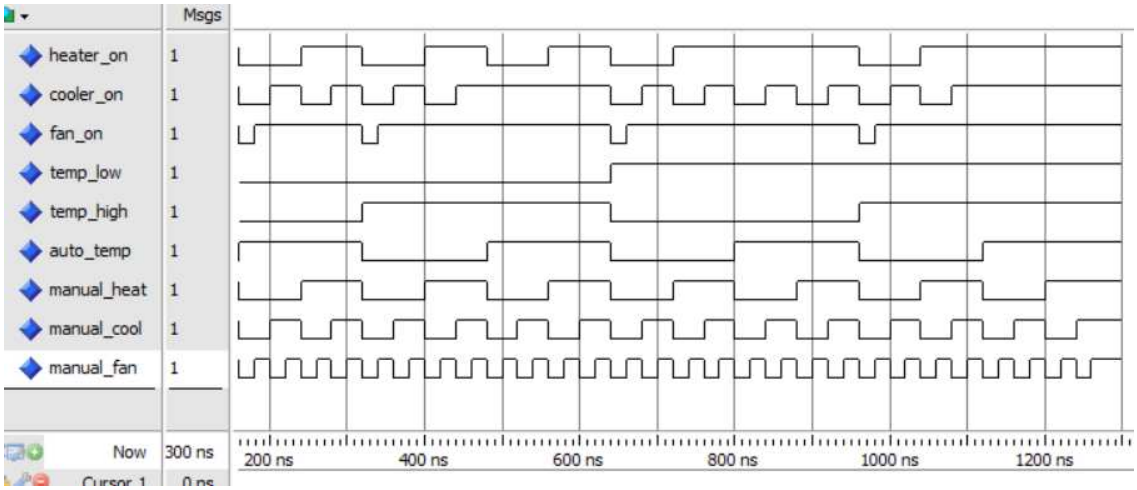

Figura 1.5: Cronograma obtenido al testear el circuito de control diseñado empleando un bloque **process**.

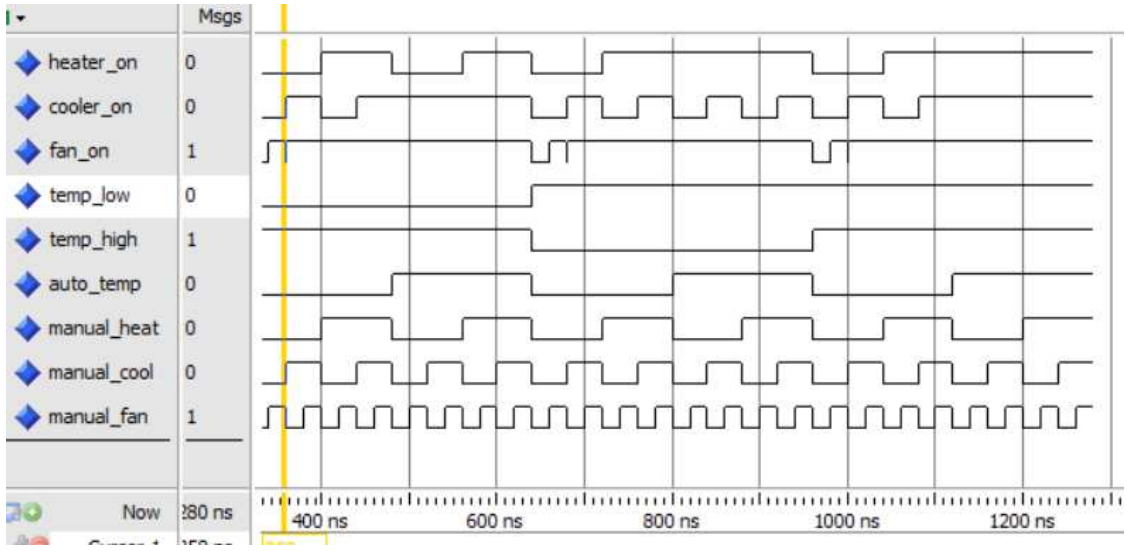

Figura 1.6: Cronograma obtenido al testear el circuito de control diseñado empleando sentencias de asignación concurrentes.

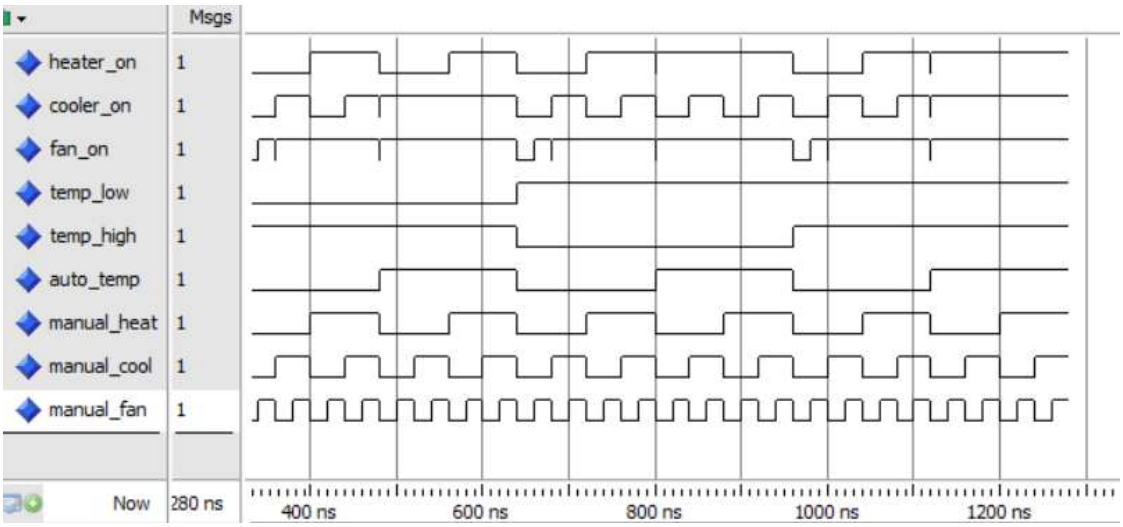

Figura 1.7: Cronograma obtenido al testear el circuito de control descrito empleando puertas lógicas.