INGENIERÍA DE COMPUTADORES 3

Solución al examen de Junio 2024, Primera Semana

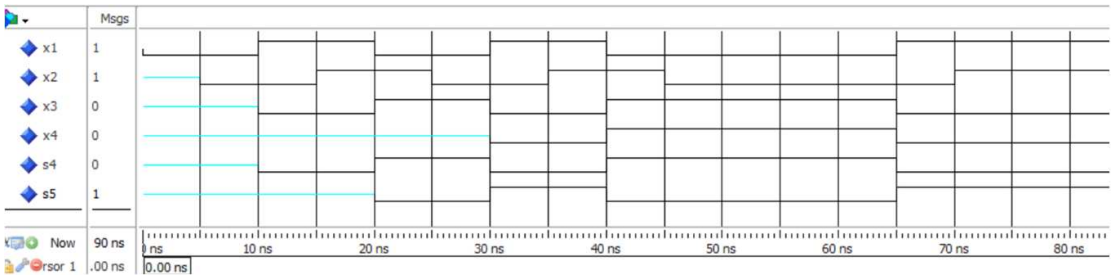
PREGUNTA 1 (2 puntos)

Tomando como base el siguiente código VHDL, dibuje el cronograma de evolución de las señales `x1`, `x2`, `x3`, `x4`, `s4` y `s5` entre los instantes 0 y 80 ns. En el cronograma se debe indicar para cada una de estas señales el instante de tiempo en que la señal cambia de valor así como su nuevo valor.

```
library IEEE;
use IEEE.std_logic_1164.all;
entity crono1 is
end entity crono1;
architecture crono1 of crono1 is
   signal x1, x2 : std_logic;
   signal x3, x4 : std_logic;
   signal s4, s5 : std_logic;
begin
  process (x1)
    variable v1, v2: std_logic;
  begin
     v1 := x2;
     v2 := v1;
     x3 <= v2;
     s4 <= x2;
     s5 <= s4;
     x4 <= s5;
  end process;
  x1 <= '0',  '1' after 10 ns, '0' after 20 ns,
        '1' after 30 ns, '0' after 40 ns,
        '1' after 65 ns;
  x2 <=  transport x1 after 5 ns;
end architecture crono1;
```

Solución a la Pregunta 1

En la siguiente figura se muestra el cronograma de evolución de las señales.



PREGUNTA 2 (3 puntos)

Escriba en VHDL la **architecture** que describe el comportamiento de un circuito contador de código Gray de dos bits con señal de reset asíncrono (`reset_n`) activa a nivel bajo que pone la cuenta al valor `"00"`. El circuito hace una cuenta ascendente y posteriormente una cuenta descendente. Es decir, la señal de salida (`count`) del contador de código Gray ha de pasar en cada flanco de subida de la señal de reloj (`clk`) de forma cíclica consecutivamente por los valores `"00"`, `"01"`, `"11"`, `"10"`, `"11"`,`"01"`. A continuación, se muestran la **entity** del circuito.

```
entity gcc_2 is
   port (  count          : out std_logic_vector(1 downto 0);
           reset_n, clk   : in  std_logic );
end entity gcc_2;
```

Solución a la Pregunta 2

El Código VHDL 1.1 muestra la **architecture** del circuito contador de código Gray.

```vhdl
library IEEE;
use IEEE.std_logic_1164.all;
architecture gcc_2 of gcc_2 is
begin
  process (reset_n, clk) is
    variable countv: std_logic_vector(1 downto 0);
    variable state : std_logic_vector(2 downto 0);
  begin
    if (reset_n = '0') then
      countv := "00";
      state  := "000";
    elsif rising_edge(clk) then
      case state is
        when "000" => countv := "00"; state := "001";
        when "001" => countv := "01"; state := "010";
        when "010" => countv := "11"; state := "011";
        when "011" => countv := "10"; state := "100";
        when "100" => countv := "11"; state := "101";
        when "101" => countv := "01"; state := "000";
        when others => countv := "00"; state := "001";
      end case;
    end if;
    count <= countv;
   end process;
end architecture gcc_2;
```

Código VHDL 1.1: Architecture del circuito contador de código Gray.

PREGUNTA 3 (2 puntos)

Escriba en VHDL un banco de pruebas que permita comprobar mediante inspección visual el comportamiento del circuito contador de la Pregunta 2. El banco de pruebas debe realizar consecutivamente los siguientes pasos:

- Primero, realiza un reseteo del circuito entre los instantes 25 ns y 125 ns.
- Después, genera los flancos necesarios de la señal de reloj para que el circuito cambie de valor 6 veces. Deben transcurrir 100 ns entre cada flanco de subida de la señal de reloj.
- No realiza más cambios en la señal de reloj y muestra el mensaje:

  `"Simulacion finalizada"`.

Solución a la Pregunta 3

El código VHDL del banco de pruebas del circuito contador se muestra en Código VHDL 1.2.

```
1  library IEEE;
2  use IEEE.std_logic_1164.all;
3  use IEEE.numeric_std.all;
4
5  entity bp_gcc2 is
6    constant PERIODO: time := 100 ns;
7  end entity bp_gcc2;
8
9  architecture bp_gcc2 of bp_gcc2 is
10   signal reset_n, clk: std_logic;
11   signal count : std_logic_vector(1 downto 0);
12   component gcc_2 is
13     port (  count          : out std_logic_vector(1 downto 0);
14             reset_n, clk   : in  std_logic );
15   end component gcc_2;
16 begin
17
18   UUT:component gcc_2 port map (count, reset_n, clk);
19   reset_n <= '1', '0' after (PERIODO/4),
20              '1' after (PERIODO+PERIODO/4);
21
22   main: process is
23   begin
24     clk <= '0';
25     wait for (PERIODO/2);
26     clk <= '1';
27     wait for (PERIODO/2);
28     if (now > (PERIODO/4+PERIODO*7)) then
29       report "Simulacion finalizada";
30       wait;
31     end if;
32   end process main;
33 end architecture bp_gcc2;
```

Código VHDL 1.2: Banco de pruebas del circuito contador.

PREGUNTA 4 (3 puntos)

Diseñe un circuito secuencial síncrono, que opere en el flanco de subida de la señal de reloj para el control de dos semáforos que están situados en un cruce, apuntando el semáforo 1 en una dirección y el semáforo 2 en otra dirección. El circuito tiene como señal de entrada la señal de reloj de 1 bit `clk` y la señal de 1 bit `stdby`. La señal de reloj de entrada al circuito tiene 60 Hz, por lo que en cada segundo se

contabilizan 60 flancos de subida de dicha señal. El circuito tiene 6 señales de salida de 1 bit: `r1`, `y1`, `g1`, `r2`, `y2`, `g2` para poner en rojo, amarillo o verde los semáforos 1 o 2, respectivamente.

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

```
entity regulador is
   port (  r1, y1, g1, r2, y2, g2 : out std_logic;
           clk, stdby   : in  std_logic );
end regulador;
```

El circuito tiene los cinco estados de operación descritos a continuación. En cada estado se indican las únicas señales de salida que han de tener valor `'1'` en dicho estado, teniendo que tener el resto de señales de salida el valor `'0'`.

- Estado YY: semáforo 1 en amarillo (`y1='1'`) y semáforo 2 en amarillo (`y2='1'`). Este es el estado con que se inicializa el circuito. Se pasa a este estado de modo asíncrono (desde cualquier otro estado) en cuanto se detecta que la señal `stdby` tiene el valor `'1'`. El circuito permanece en este estado mientras que la señal `stdby` tenga el valor `'1'`. Cuando la señal `stdby` toma el valor `'0'`, se pasa al estado RY.
- Estado RY: semáforo 1 en rojo (`r1='1'`) y semáforo 2 en amarillo (`y2='1'`). El circuito permanece en este estado 5 s, pasando transcurrido este tiempo al estado GR.
- Estado GR: semáforo 1 en verde (`g1='1'`) y semáforo 2 en rojo (`r2='1'`). El circuito permanece en este estado 45 s, pasando transcurrido este tiempo al estado YR.
- Estado YR: semáforo 1 en amarillo (`y1='1'`) y semáforo 2 en rojo (`r2='1'`). El circuito permanece en este estado 5 s, pasando transcurrido este tiempo al estado RG.
- Estado RG: semáforo 1 en rojo (`r1='1'`) y semáforo 2 en verde (`g2='1'`). El circuito permanece en este estado 30 s, pasando transcurrido este tiempo al estado RY.

El circuito se ha de diseñar como una máquina de Moore que opera en el flanco de subida de la señal de reloj. Escriba el código VHDL de la **architecture** que describe el comportamiento del circuito.

Solución a la Pregunta 4

El Código VHDL 1.3 es el diseño del circuito secuencial solución de la Pregunta 4.

```
1  library IEEE;
2  use IEEE.std_logic_1164.all;
3  use IEEE.numeric_std.all;
4  architecture fsm of regulador is
5    type state is (RG, RY, GR, YR, YY);
6    constant timeRG: integer := 1800;
7    constant timeRY: integer := 300;
8    constant timeGR: integer := 2700;
9    constant timeYR: integer := 300;
10   signal timer: integer range 0 to 2700;
11   signal pr_state, nx_state: state;
12 begin
13   process (clk, stdby) is
14     variable count: integer range 0 to 2700;
15   begin
16     if (stdby='1') then
17        pr_state <= YY;
18        count := 0;
19     elsif rising_edge(clk) then
20        count := count + 1;
21        if (count >= timer) then
22            pr_state <= nx_state;
23            count := 0;
24        end if;
25     end if;
26   end process;
27   process (pr_state)
28   begin
29      r1 <= '0'; y1 <= '0'; g1 <= '0';
30      r2 <= '0'; y2 <= '0'; g2 <= '0';
31      case pr_state is
32        when RG =>
33          r1 <= '1'; g2 <= '1';
34          nx_state <= RY;
35          timer <= timeRG;
36         when RY =>
37          r1 <= '1';  y1 <= '1';
38          nx_state <= GR;
39          timer <= timeRY;
40         when GR =>
41          g1 <= '1'; r2 <= '1';
42          nx_state <= YR;
43          timer <= timeGR;
44         when YR =>
45          y1 <= '1'; r2 <= '1';
46          nx_state <= RG;
47          timer <= timeYR;
```

```
48          when YY =>
49           y1 <= '1'; y2 <= '1';
50           nx_state <= RY;
51           timer <= timeYR; --Para evitar latches
52       end case;
53    end process;
54 end fsm;
```

Código VHDL 1.3: Circuito regulador de los semáforos de un cruce.