Código asignatura	Nombre asignatura
71012018	**Ingeniería de Computadores III**
Fecha alta y origen	Convocatoria
04/11/2015	Septiembre 2015 (Original)
Página Asignatura	

INGENIERÍA DE COMPUTADORES III

Solución al examen de Septiembre 2015

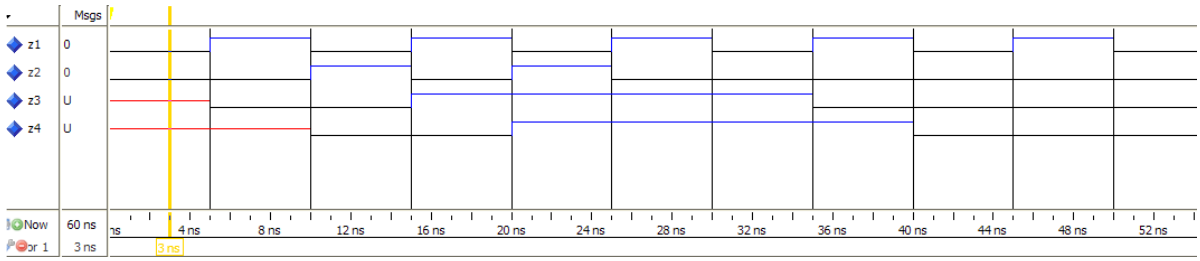
PREGUNTA 1 (2 puntos)

Tomando como base el siguiente código VHDL, dibuje el cronograma de evolución de las señales `z1`,`z2`, `z3` y `z4` entre los instantes 0 y 60 ns.

```
library IEEE;
use IEEE.std_logic_1164.all;
entity crono1 is
end entity crono1;
architecture crono1 of crono1 is
  constant PER : time :=10 ns;
  signal z1: std_logic:='0';
  signal z2, z3, z4: std_logic;
begin
  process is
  begin
    z2<='0';     wait until falling_edge(z1);
    z2<='1';     wait until rising_edge(z1);
    z2<='0';     wait until falling_edge(z1);
    z2<='1';     wait until rising_edge(z1);
    z2<='0';     wait until rising_edge(z1);
    wait;
  end process;
  process (z1)
  begin
    if(rising_edge(z1)) then
      z3<=z2;
    end if;
    z4<=z3;
  end process;
  z1<=not z1 after (PER/2);
end architecture crono1;
```

Solución a la Pregunta 1

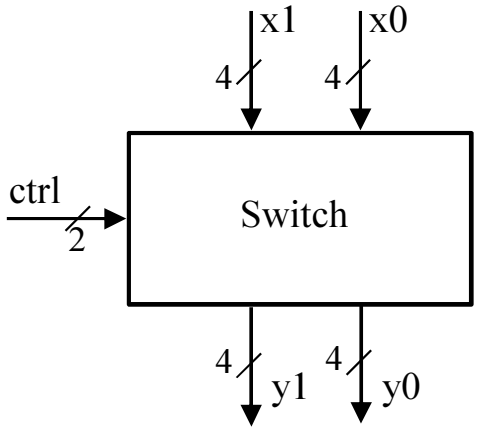
A continuación se muestra el cronograma de evolución de las señales `z1`, `z2`, `z3` y `z4` entre los instantes 0 y 60 ns.



PREGUNTA 2 (3 puntos)

Escriba en VHDL, de las cuatro formas que se detallan a continuación, la **architecture** que describe el comportamiento de un circuito combinacional cuya **entity**, símbolo lógico y tabla de operaciones se muestran a continuación.

```
entity switch is
 port (  y1, y0     : out std_logic_vector(3 downto 0);
         ctrl       : in  std_logic_vector(1 downto 0);
         x1, x0     : in std_logic_vector(3 downto 0) );
end entity switch;
```



ctrl(1)	ctrl(0)	y1	y0
0	0	x1	x0
0	1	x0	x1
1	0	x0	x0
1	1	x1	x1

2.a) (0.75 puntos) Empleando sentencias concurrentes condicionales (**when - else**).

2.b) (0.75 puntos) Empleando asignaciones concurrentes de selección (**with - select**).

2.c) (0.75 puntos) Empleando un bloque **process** con sentencias **if**.

2.d) (0.75 puntos) Empleando un bloque **process** con sentencias **case**.

Solución a la Pregunta 2

El código correspondiente a los apartados 2.a–2.d se muestra en Código VHDL 1.1–1.4.

```
-------------------------------------
architecture switchCond of switch is
begin
   y1   <= x1 when (ctrl="00") else
           x0 when (ctrl="01") else
           x0 when (ctrl="10") else
           x1;
   y0   <= x0 when (ctrl="00") else
           x1 when (ctrl="01") else
           x0 when (ctrl="10") else
           x1;
end architecture switchCond;
-------------------------------------
```

Código VHDL 1.1: Solución apartado 2a.

```
-------------------------------------
architecture switchSel of switch is
begin
     with ctrl select
          y1 <=  x1 when "00",
                 x0 when "01",
                 x0 when "10",
                 x1 when others;

     with ctrl select
          y0 <=  x0 when "00",
                 x1 when "01",
                 x0 when "10",
                 x1 when others;
end architecture switchSel;
-------------------------------------
```

Código VHDL 1.2: Solución apartado 2b.

```
-------------------------------------
architecture switchIf of switch is
begin
   process (x1, x0, ctrl) begin
      if (ctrl = "00") then
         y1<=x1;
         y0<=x0;
      elsif (ctrl = "01") then
         y1<=x0;
         y0<=x1;
      elsif (ctrl = "10") then
         y1<=x0;
         y0<=x0;
      else
         y1 <= x1;
         y0 <= x1;
      end if;
   end process;
end architecture switchIf;
-------------------------------------
```

Código VHDL 1.3: Solución apartado 2c.

```
-------------------------------------
architecture switchCase of switch is
begin
   process (x1, x0, ctrl) begin
      case ctrl is
            when "00"       =>
               y1<=x1;
               y0<=x0;
            when "01"       =>
               y1<=x0;
               y0<=x1;
            when "10"       =>
               y1<=x0;
               y0<=x0;
            when others =>
               y1 <= x1;
               y0 <= x1;
      end case;
   end process;
end architecture switchCase;
-------------------------------------
```

Código VHDL 1.4: Solución apartado 2d.

PREGUNTA 3 (3 puntos)

Diseñe un circuito secuencial síncrono capaz de detectar cuando los últimos 8 bits recibidos por su entrada serie (`DataSIN`) son iguales a un cierta palabra patrón de 8 bits almacenada en el circuito.

La palabra patrón es cargada a través de la entrada paralelo (`DataPIN`). La carga se produce en el flanco de subida de la señal de reloj, cuando la señal `Load` valga '1'.

Asimismo, el circuito posee una salida de 8 bits llamada `Patron`, que en todo momento muestra la palabra patrón internamente almacenada en el circuito.

Por otra parte, los últimos 8 bits recibidos por la entrada serie `DataSIN` van almacenándose en un registro interno del circuito llamado `RegCONT`, de manera que el último bit recibido por la entrada serie es el bit menos significativo de la palabra almacenada en el registro. Por ejemplo, si el valor de la señal de entrada `DataSIN` en los últimos 8 flancos de subida consecutivos de la señal de reloj (`Clk`) es '1', '1', '1', '1', '1', '1', '0', '0', entonces el contenido del registro `RegCONT` será "11111100".

El contenido del registro `RegCONT` es reseteado al valor "00000000" mediante la señal de reset síncrono (`Reset`) activa a nivel alto.

El circuito debe comparar si la palabra patrón coincide con la palabra del registro `RegCONT`. La salida `Y` debe valer '1' mientras ambas palabras de 8 bits sean iguales y debe valer '0' en caso contrario.

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

```
entity detector is
   port( Y     : out std_logic;
         Patron: out std_logic_vector(7 downto 0);
         Clk   : in std_logic;
         Reset : in std_logic;
         Load  : in std_logic;
         DataSIN: in std_logic;
         DataPIN: in std_logic_vector(7 downto 0) );
end entity detector;
```

Escriba en VHDL la **architecture** que describe el comportamiento del circuito.

Solución a la Pregunta 3

El código VHDL del circuito secuencial se muestra en Código VHDL 1.5.

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;
use IEEE.std_logic_unsigned.all;
architecture detector of detector is
   signal PatronLOC: std_logic_vector( 7 downto 0);
   signal RegCONT: std_logic_vector( 7 downto 0);
begin
    Reg: process (Clk)
    begin
        if rising_edge(Clk) then
            if (Reset = '1') then
                RegCONT <= (others =>'0');
            else
                RegCONT <= RegCONT(6 downto 0)&DataSIN;
            end if;
        end if;
    end process;

    LoadP: process (Clk)
    begin
        if rising_edge(Clk) then
            if (Load = '1') then
                PatronLOC <= DataPIN;
            end if;
        end if;
    end process;
    Patron <= PatronLOC;

    comp: process (RegCONT, PatronLOC)
    begin
        Y <= '0';
        if (RegCONT = PatronLOC) then
            Y<= '1';
        end if;
    end process;
end architecture detector;
```

Código VHDL 1.5: Banco de pruebas del circuito combinacional.

PREGUNTA 4 (2 puntos)

Programe en VHDL un banco de pruebas para el circuito que ha diseñado al contestar a la Pregunta 3. La señal de reloj (`Clk`) debe tener un periodo de 20 ns e inicialmente valer '0'. El programa de test debe realizar consecutivamente las acciones siguientes:

1. *Reset*. La señal de reset ha de tener el valor '1' durante 15 ns.
2. *Cargar el patrón "11111100"*. Para ello, la señal `Load` ha de valer '1' y la señal `DataPIN` ha de tener el valor "11111100".
3. *Introducir los valores adecuados por la entrada serie para que se reconozca el patrón de la entrada*. El banco de pruebas debe comprobar que la señal de salida `Y` va tomando los valores adecuados. En caso contrario, debe mostrar un mensaje indicándolo.

Solución a la Pregunta 4

El código del banco de pruebas se muestra en Código VHDL 1.6.

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
entity bp_detector is
    constant PERIODO     : time      := 20 ns; -- PERIODO clk
end entity bp_detector;

architecture bp_detector of bp_detector is
    signal Y: std_logic; --Salida UUT
    signal Patron   : std_logic_vector(7 downto 0); --Salida
                                                    UUT
    signal DataPIN    : std_logic_vector(7 downto 0); --
                                                    Entradas UUT
    signal DataSIN    : std_logic;
    signal Clk : std_logic := '0';
    signal Reset, Load: std_logic;
    component detector is
    port (   Y       : out std_logic;
             Patron: out std_logic_vector( 7 downto 0);
             Clk    : in std_logic;
             Reset  : in std_logic;
             Load   : in std_logic;
             DataSIN: in std_logic;
             DataPIN: in std_logic_vector(7 downto 0) );
    end component detector;
begin
 UUT : component detector port map
         (Y, Patron, Clk, Reset, Load, DataSIN, DataPIN);
 Clk <= not Clk after (PERIODO/2);
 reset <= '1' ,'0' after 15 ns;
 vec_test : process is
    variable temp   : std_logic_vector (7 downto 0);
  begin
    report "Comienza la simulación";
    load <= '1';
    DataPIN <= "11111100";
    DataSIN <= '0';
    wait for 2*PERIODO;
    temp := DataPIN;
    load <= '0';
    for i in 7 downto 0 loop
      DataSIN <= temp(i);
      wait for PERIODO;
      if i>0 then
          assert(Y='0') report("Valor de Y erroneo");
      else
          assert(Y='1') report("Valor de Y erroneo");
      end if;
    end loop;
    wait;   -- Termina la simulación
  end process vec_test;
end architecture bp_detector;
```

Código VHDL 1.6: Banco de pruebas.