INGENIERÍA DE COMPUTADORES III

INSTRUCCIONES
Por favor, entregue esta primera hoja de enunciado junto con el examen.
Dispone de 2 horas para realizar el examen.
MATERIAL PERMITIDO: Ninguno.

Pregunta 1 (2 puntos)

Tomando como base el siguiente código VHDL, dibuje el cronograma de evolución de las señales `x1`, `x2`, `x3`, `x4` entre los instantes 0 y 80 ns.

```
library IEEE;
use IEEE.std_logic_1164.all;
entity crono is
end entity crono;
architecture crono of crono is
  signal x1, x2, x3, x4 : std_logic;
begin
  x2 <= '0', '1' after 25 ns,
        '0' after 35 ns,'1' after 50 ns;
  Proc1: process
  begin
    x1 <= '0';
    wait for 10 ns;
    x1 <= '1';
    wait for 15 ns;
    x1 <= '0';
  end process Proc1;
  Proc2: process
  begin
    x1 <= '0';
    wait for 15 ns;
    x1 <= '1';
    wait for 10 ns;
    x1 <= '0';
    wait;
  end process Proc2;
  Proc3: process (x2)
  begin
    x3 <= x2;
    x4 <= x3;
  end process Proc3;
end architecture crono;
```

Pregunta 2 (2.5 puntos)

Diseñe un circuito combinacional que tenga una señal de entrada de 8 bits llamada `data` y una señal de salida de 4 bits llamada `zeros`. La señal `zeros` se interpreta como un número binario sin signo. Esta señal ha de tener como valor el número de ceros existentes en `data` antes de encontrar el primer '1', empezando a contar por el extremo izquierdo (bit más significativo) de `data` . Por ejemplo, el valor de la señal de entrada `data` "00110110" da como resultado un valor de la señal `zeros` de "0010" (valor 2 en notación decimal).

El diseño ha de tener la siguiente **entity**:

```
entity numceros is
   port( zeros   : out std_logic_vector ( 3 downto 0);
         data  : in std_logic_vector(7 downto 0));
end entity numceros;
```

Pregunta 3 (3.5 puntos)

Diseñe usando VHDL un contador binario ascendente de 8 bits que opere en el flanco de subida de la señal de reloj, con carga de datos síncrona y reset asíncrono. El contador tiene la siguiente **entity**.

```
entity contador is
   port (  count : out std_logic_vector( 7 downto 0);
           data: in std_logic_vector( 7 downto 0);
           clk, reset, load  : in  std_logic );
end contador;
```

Las entradas al circuito son la señal de reloj (`clk`), la señal de reset (`reset`) asíncrona y activa a nivel bajo, la señal de carga (`load`) y la señal `data`. El circuito tiene una única señal de salida llamada `count`.

Cuando la señal `reset` vale '0', todos los bits de la señal `count` se ponen a 0. En caso contrario, en cada flanco de subida de la señal de reloj, si la señal `load` vale '0' se incrementa en '1' el valor de la señal de salida y si `load` vale '1' se asigna a la señal de salida el valor de la señal `data`.

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 '1'. El programa de test debe realizar consecutivamente las acciones siguientes:

1. *Reset*. La señal de reset ha de tener el valor '0' durante 15 ns.
2. *Empezar la cuenta*. Comprobar que el circuito pasa consecutivamente por los valores "00000000" a "00000011".
3. *Cargar el valor "11111100"*. La señal `load` ha de valer '1' . La señal `data` ha de tener el valor "11111100".
4. *Seguir la cuenta*. Comprobar que el circuito pasa por los valores "11111100", "11111101", "11111110", "11111111" y "00000000".

El banco de pruebas debe comprobar que los valores de las señales de salida de la UUT coinciden con los esperados, mostrando el correspondiente mensaje en caso de que no coincidan. Al final del test, debe mostrarse un mensaje indicando el número total de errores.