INGENIERÍA DE COMPUTADORES III

Ejercicio de autocomprobación 6

INSTRUCCIONES:

1. Resuelva este ejercicio en las mismas condiciones en que realizará el examen: dos horas de tiempo y sin emplear ningún material.
2. Revise sus contestaciones, empleando para ello el texto y el simulador que esté usando para estudiar la asignatura.
3. Compare sus respuestas revisadas con la solución.

Pregunta 1 (2 puntos)

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

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

entity crono2 is
end entity crono2;

architecture crono2 of crono2 is
  signal z1: std_logic;
  signal z2: std_logic;
  signal z3: std_logic;
  signal x:  std_logic;
begin
  process is
    variable var1: std_logic;
  begin
    for i in 1 to 4 loop
      var1 := x;
      z1 <= var1;
      z2 <= z1;
      wait for 10 ns;
    end loop;
  end process;
  z3 <=  x after 10 ns;
  x  <= '0',             '1' after  5 ns,
        '0' after 15 ns, '1' after 20 ns,
        '0' after 30 ns, '1' after 35 ns,
        '0' after 40 ns;
end architecture crono2;
```

Pregunta 2 (3.5 puntos)

Escriba en VHDL la **architecture** que describe:

2.a) (0.5 puntos) El comportamiento de un multiplexor con un entrada de selección (`sel`), dos entradas de 8 bits (`x1`, `x2`) y una salida de 8 bits (`y`). Emplee para ello un bloque **process** con una sentencia **case**. La **entity** del circuito es la siguiente:

```
entity mux is port(
   y      :   out std_logic_vector(7 downto 0);
   x1, x2 :   in  std_logic_vector(7 downto 0);
   sel    :   in std_logic);
end entity mux;
```

2.b) (1 punto) El comportamiento de una unidad aritmética con dos entradas `A` y `B`. El tamaño de los dos operandos de entrada, `A` y `B`, es de 8 bits. La operación que realiza es especificada por la señal de entrada `sel`. Emplee en la programación del circuito una sentencia concurrente condicional (**when** - **else**). A continuación, se muestra la **entity** y la tabla de operaciones correspondiente al circuito.

```
entity uaritm is port(
   y      :   out std_logic_vector(7 downto 0);
   A, B   :   in  std_logic_vector(7 downto 0);
   sel    :   in  std_logic_vector(1 downto 0));
end entity uaritm;
```

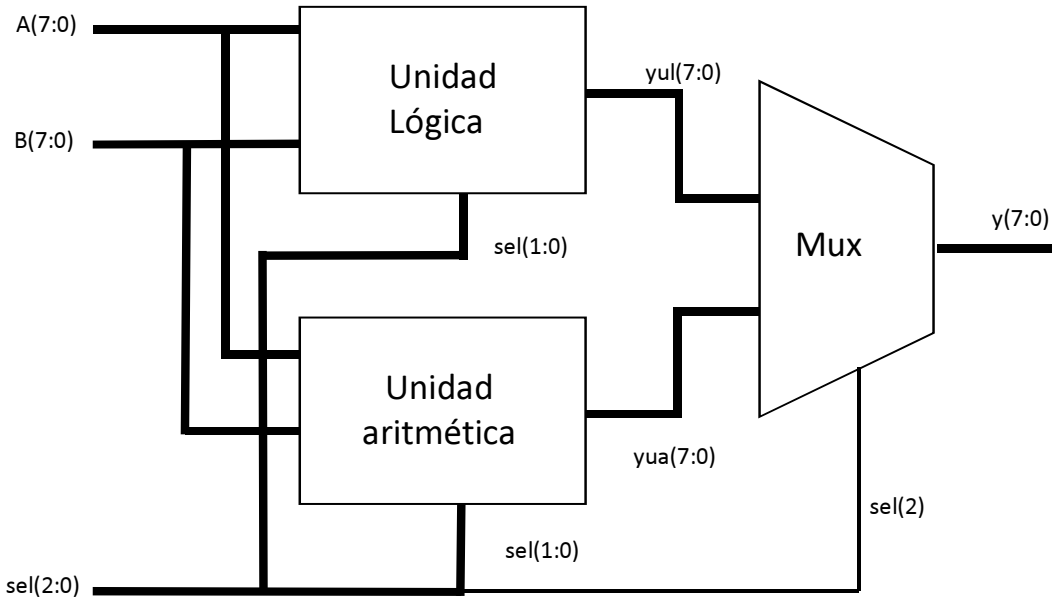
`sel`	Operación
0 0	*A*
0 1	*A* + *B*
1 0	*A* - *B*
1 1	-*A*

2.c) (1 punto) El comportamiento de una unidad lógica con dos entradas `A` y `B`. El tamaño de los dos operandos de entrada, `A` y `B`, es de 8 bits. La operación que realiza es especificada por la señal de entrada `sel`. Emplee en la programación del circuito un bloque **process** con una sentencia **if**. A continuación, se muestra la **entity** y la tabla de operaciones correspondiente al circuito.

```
entity ulogic is port(
   y      :   out std_logic_vector(7 downto 0);
   A, B   :   in  std_logic_vector(7 downto 0);
   sel    :   in  std_logic_vector(1 downto 0));
end entity ulogic;
```

`sel`	Operación
0 0	*A* or *B*
0 1	*A* nand *B*
1 0	*A* nor *B*
1 1	*A* xor *B*

2.d) (1 punto) La estructura de una unidad aritmético lógica (ALU). La **architecture** debe describir la estructura del circuito combinacional mostrado en la figura, instanciando y conectando adecuadamente los circuitos cuyo diseño ha realizado al contestar los tres apartados anteriores. Los dos bits menos significativos de la señal `sel` seleccionan la operación a realizar por la unidad lógica y por la unidad aritmética. El bit más significativo de la señal `sel` selecciona si la salida de la ALU es la salida de la unidad aritmética o la salida de la unidad lógica.



La **entity** del circuito es:

```
entity alu is port(
   y      :   out std_logic_vector(7 downto 0);
   A, B   :   in  std_logic_vector(7 downto 0);
   sel    :   in  std_logic_vector(2 downto 0));
end entity alu;
```

Pregunta 3 (2.5 puntos)

Diseñe usando VHDL un registro de desplazamiento de 4 bits conversor serie a paralelo. El circuito tiene las entradas siguientes: señal de reloj (`Clock`), señal de control de desplazamiento hacia la derecha (`Shift`), señal de reset asíncrono activo a nivel alto (`Reset`) y señal de entrada serie de datos (`Serial_in`). El circuito tiene una señal de 4 bits de salida paralelo de datos (`Q`). La **entity** del circuito es:

```
entity registro is port(
   Q : out std_logic_vector(3 downto 0);
   Clock, Shift, Serial_in, Reset : in std_logic);
end entity registro
```

Mientras `Reset` vale ’0’ y `Shift` vale ’1’, en cada flanco de subida de la señal de reloj se desplaza el contenido del registro un bit a la derecha, cargándose en el bit más significativo del registro el valor de `Serial_in`.

El diseño del registro en VHDL debe realizarse describiendo el comportamiento del circuito, empleando para ello un único bloque **process**.

Pregunta 4 (2 puntos)

Programe en VHDL un banco de pruebas para el registro que ha diseñado al contestar a la Pregunta 3. El programa de test debe primero resetear el registro y a continuación cargar en el registro, a través de la entrada serie y por este orden, los cuatro bits siguientes: ‘0’, ‘1’, ‘1’, ‘1’. Si los valores de la señal de salida de la UUT no coinciden con lo esperado, el programa de test debe mostrar el correspondiente mensaje.