

INGENIERÍA DE COMPUTADORES III

INSTRUCCIONES

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 y x4 entre los instantes 0 y 100 ns.

```
library IEEE;
use IEEE.std_logic_1164.all;
entity cronol is
end entity cronol;

architecture cronol of cronol is
    signal s1, s2, s3: std_logic := '0';
    signal x1, x2, x3, x4 : std_logic;
begin
    process (s2)
    begin
        if (rising_edge(s2)) then
            s3 <= x1;
            x3 <= s3;
            x4 <= x3;
        end if;
    end process;
    s1 <= '0', '1' after 5 ns, '0' after 10 ns,
        '1' after 20 ns, '0' after 40 ns;
    s2 <= not s2 after 10 ns;
    x1 <= transport s1 after 10 ns;
    x2 <= s1 after 10 ns;
end architecture cronol;
```

Pregunta 2 (3 puntos)

Escriba en VHDL cuatro **architecture** que describan lo siguiente:

- 2.a)** (0.5 puntos) La primera **architecture** debe describir 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*). En el diseño de esta **architecture** emplee 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)** (0.75 puntos) La segunda **architecture** debe describir 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;
```

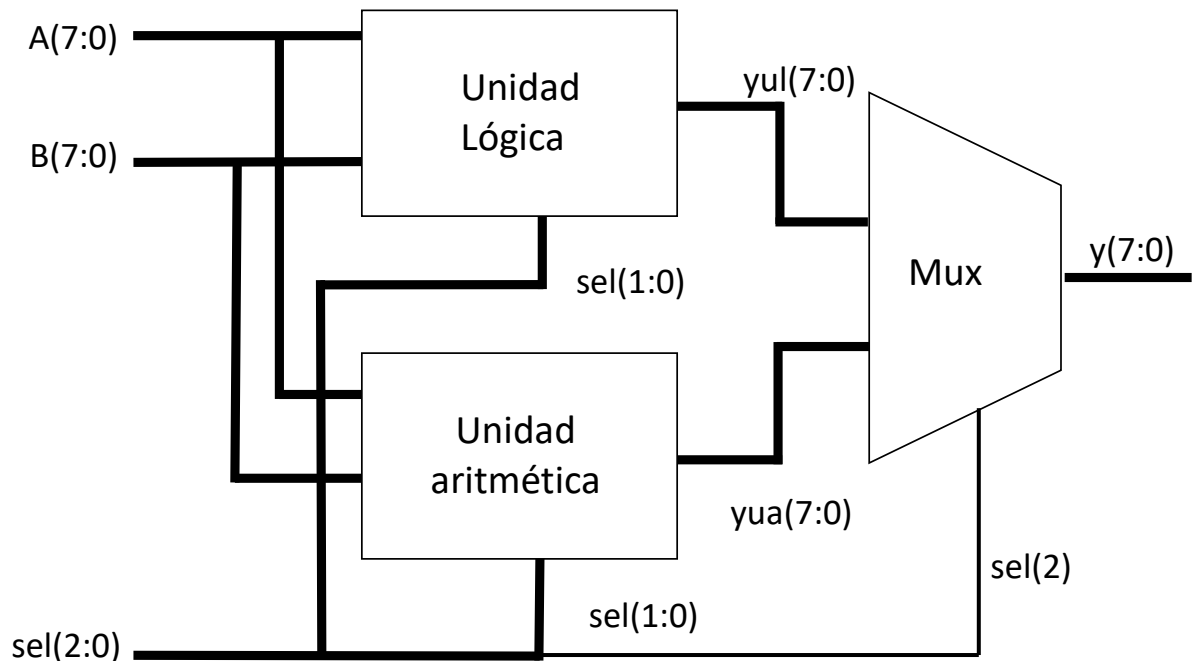
sel	Operación
0 0	A
0 1	$A + B$
1 0	$A - B$
1 1	$-A$

- 2.c)** (0.75 puntos) La tercera **architecture** debe describir 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	$A \text{ or } B$
0 1	$A \text{ nand } B$
1 0	$A \text{ nor } B$
1 1	$A \text{ xor } B$

- 2.d)** (1 punto) Por último, la cuarta **architecture** debe describir 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 (3 puntos)

Escriba en VHDL un circuito secuencial síncrono capaz de detectar si recibe la secuencia "0100" por su entrada X. La **entity** del circuito se muestra a continuación.

```
entity detector is
    port( Y      : out std_logic;
          state  : out std_logic_vector(2 downto 0);
          X      : in  std_logic;
          reset  : in  std_logic;
          clk    : in  std_logic );
end entity detector;
```

El circuito tiene una señal de reloj (clk), una entrada serie de un bit (X), una señal de reset asíncrona activa en '1' (reset), una señal que indica el estado en que se encuentra el circuito (state) y una señal de salida de un bit (Y).

La señal `Y` se pone a '1' si los últimos 4 bits recibidos por la entrada `X` se corresponden con la secuencia "0100". La máquina no vuelve al estado inicial tras haber reconocido la secuencia, sino que detecta secuencias solapadas. La señal `reset` pone el circuito en su estado inicial. Todos los cambios, excepto el reseteo del circuito, tienen lugar en el flanco de subida de la señal de reloj.

Escriba en VHDL la **architecture** que describe el comportamiento del circuito en términos de una máquina de Moore. Dibuje el diagrama de estados correspondiente al circuito que ha diseñado.

Pregunta 4 (2 puntos)

Programe en VHDL un banco de pruebas para el circuito secuencial que ha diseñado al contestar a la Pregunta 3. El programa de test debe primero resetear el circuito y a continuación cargar en el circuito, a través de la entrada `X`, los siete bits siguientes (en el orden indicado): '0', '1', '0', '0', '1', '0', '0'. Si los valores de las dos señales de salida de la UUT no coinciden con lo esperado, el programa de test debe mostrar el correspondiente mensaje.