INGENIERÍA DE COMPUTADORES 3

Solución al examen de Septiembre 2017

PREGUNTA 1 (2 puntos)

Tomando como base el siguiente código VHDL, dibuje el cronograma de evolución de las señales `x1`, `x2`, `x3`, `x4`, `x5` y `x6` entre los instantes 0 y 200 ns.

```
library IEEE;
use IEEE.std_logic_1164.all;
entity crono1 is
end entity crono1;
architecture crono1 of crono1 is
   signal x1, x2 : std_logic := '0';
   signal x3, x4, x5, x6 : std_logic;
begin
 process (x1)
    variable temp1, temp2, temp3: std_logic;
  begin
     if (rising_edge(x1)) then
     temp1 := x2;
     temp2 := temp1;
     temp3 := temp2;
     x3 <= temp3;
     x4 <= x2;
     x5 <= x4;
     x6 <= x5;
    end if;
  end process;
  x1 <= not x1 after 20 ns;
  x2 <= '0',  '1' after 30 ns, '0' after 90 ns;
end architecture crono1;
```

Solución a la Pregunta 1

En la Figura 1.1 se muestra el cronograma de evolución de las señales.

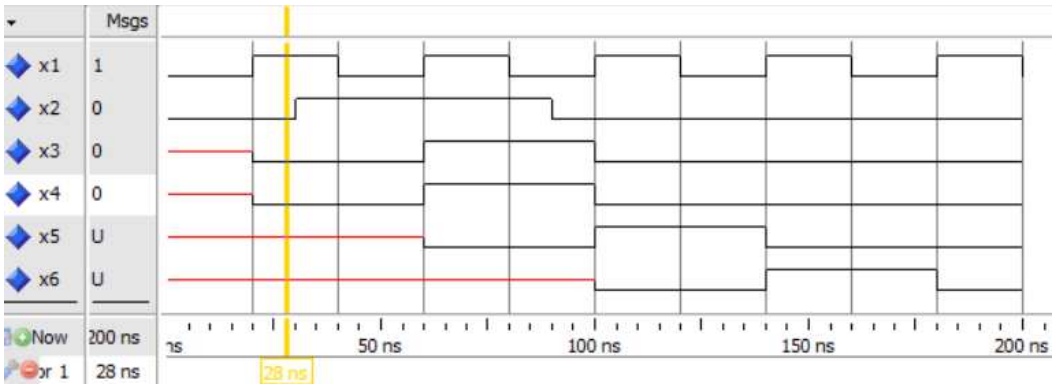
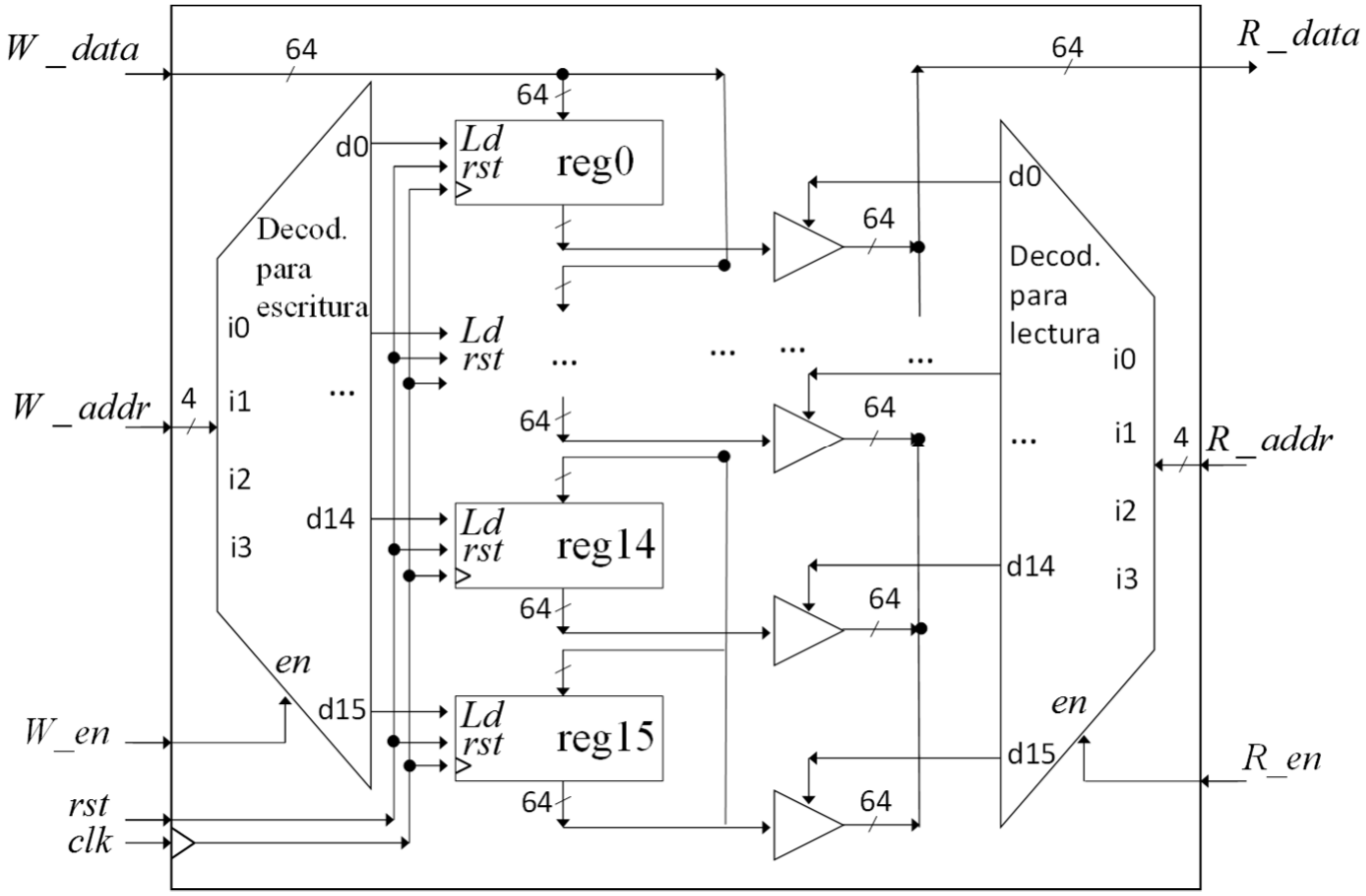


Figura 1.1: Cronograma de evolución de las señales.

PREGUNTA 2 (3 puntos)

Se pretende diseñar un *register file* de tamaño 16*64 que opera en el flanco de subida de la señal de reloj. Es decir, que contiene 16 registros, cada uno de los cuales tiene 64 bits. Este circuito se puede diseñar empleando dos decodificadores, 16 registros de 64 bits y 16 buffer triestado de 64 bits, tal como se muestra en la siguiente figura.



El circuito tiene una señal de salida `R_data` de 64 bits y una señal de entrada `W_data` de 64 bits.

Tiene asímismo una señal síncrona de reset a nivel alto llamada `rst` y una señal de reloj `clk`. La señal de reset pone todos los bits de todos los registros a cero.

Las señales de entrada `W_en` y `R_en` se emplean para habilitar la escritura y la lectura, respectivamente.

Las señales de entrada `W_addr` y `R_addr` se emplean para direccionar uno de los 16 registros.

Cuando la señal `rst` tiene valor '0' y la señal `W_en` tiene valor '1', el registro cuya dirección sea `W_addr` se carga en el flanco de subida de la señal de reloj con la señal de entrada `W_data`.

Cuando la señal `R_en` tiene valor '1', la señal de salida `R_data` toma el valor de la salida del registro cuya dirección sea `R_addr`. Por el contrario, cuando la señal `R_en` tiene valor '0', la señal de salida `R_data` toma el valor de alta impedancia.

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

```
entity RegFile16x64 is
   port ( R_data : out std_logic_vector(63 downto 0);
          W_data : in  std_logic_vector(63 downto 0);
          R_addr, W_addr: in std_logic_vector(3 downto 0);
          R_en, W_en: in std_logic;
          clk, rst: in std_logic );
end entity RegFile16x64;
```

Escriba en VHDL la **architecture** que describe el comportamiento del *register file* empleando para ello dos bloques **process**.

Solución a la Pregunta 2

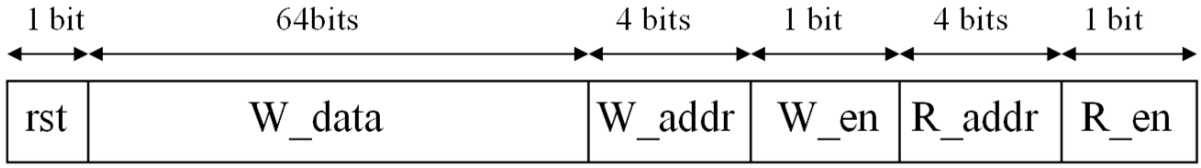
La **architecture** del *register file* se muestra en el Código 1.1.

```
-------------------------------------------------
-- Register file 16x64.
-- Descripción del comportamiento
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity RegFile16x64 is
    port ( R_data : out std_logic_vector(63 downto 0);
           W_data : in  std_logic_vector(63 downto 0);
           R_addr, W_addr: in std_logic_vector(3 downto 0);
           R_en, W_en: in std_logic;
           clk, rst: in std_logic );
end entity RegFile16x64;

architecture RegFile16x64_comp of RegFile16x64 is

    type regfile_type is
        array (0 to 15) of std_logic_vector(63 downto 0);
    signal regfile : regfile_type;

begin

    W_process : process (clk)
    begin
        if rising_edge(clk) then
            if ( rst = '1' ) then
               for i in 0 to 15 loop
                  regfile(i) <= (others=>'0');
               end loop; --bucle for
            elsif ( W_en = '1' ) then
                regfile(conv_integer(W_addr)) <= W_data;
            end if;
        end if;
    end process W_process;

    R_process: process (R_addr, R_en, regfile)
    begin
        if ( R_en = '1' ) then
            R_data <= regfile(conv_integer(R_addr));
        else
            R_data <= (others => 'Z');
        end if;
    end process R_process;

end architecture RegFile16x64_comp;
-------------------------------------------------
```

Código VHDL 1.1: Architecture del *register file*.

PREGUNTA 3 (2.5 puntos)

Diseñe un banco de pruebas para el *register file* que ha diseñado en la Pregunta 2 que permita realizar una inspección visual de las señales de entrada y de salida del circuito. El banco de pruebas debe generar una señal de reloj de periodo 10 ns y valor inicial '0'. El resto de valores de las señales de entrada los ha de leer de un fichero llamado *vectores.txt*. Cada una de las líneas de este fichero de texto consta de una palabra de 75 bits cuyo significado se muestra en la siguiente figura.



Solución a la Pregunta 3

El código VHDL del banco de pruebas se muestra en Código VHDL 1.2–1.3.

```
-------------------------------------------------
-- Banco de pruebas del register file
library IEEE;
use IEEE.std_logic_1164.all;
use std.textio.all;

entity bp_regFile16x64 is
    constant PERIODO : time         := 10 ns;
end entity bp_regFile16x64;

architecture bp_regFile16x64 of bp_regFile16x64 is
  signal R_data          : std_logic_vector(63 downto 0);
  signal W_data          : std_logic_vector(63 downto 0);
  signal R_addr, W_addr  : std_logic_vector(3 downto 0);
  signal R_en, W_en      : std_logic;
  signal rst             : std_logic;
  signal clk             : std_logic   := '0';

  component RegFile16x64 is
    port ( R_data          : out std_logic_vector(63 downto 0);
           W_data          : in  std_logic_vector(63 downto 0);
           R_addr, W_addr  : in  std_logic_vector(3 downto 0);
           R_en, W_en      : in  std_logic;
           clk, rst        : in  std_logic  );
  end component RegFile16x64;

begin
  -- Instanciar y conectar UUT
  uut : component RegFile16x64 port map
       (R_data, W_data, R_addr, W_addr, R_en, W_en, clk, rst);

  clk <= not clk after (PERIODO/2);

  gen_vec_test : process is
    file      fichVectores : text;
    variable inputLinea    : line;
    variable inputBit      : bit;
    constant numBits       : integer   := 74;   -- Num bits menos 1
    variable aux           : std_logic_vector(numBits downto 0);
```

Código VHDL 1.2: Diseño del banco de pruebas del *register file.*

```
  begin
    report "Comienza la simulación";
    -- Carga de vectores de test desde fichero
    file_open(fichVectores,"vectores.txt",read_mode);
    while not endfile(fichVectores) loop
        -- Lectura desde fichero
        readline(fichVectores,inputLinea);
        for i in numBits downto 0 loop
            read(inputLinea,inputBit);
            if inputBit = '1' then
              aux(i) := '1';
            else
              aux(i) := '0';
            end if;
        end loop; -- bucle for
        -- Asigna valor a W_data, W_addr, W_en, R_addr, R_en
        rst    <= aux(74);
        W_data <= aux(73 downto 10);
        W_addr <= aux(9 downto 6);
        W_en   <= aux(5);
        R_addr <= aux(4 downto 1);
        R_en   <= aux(0);
        wait for PERIODO;
    end loop;      -- bucle while
    file_close(fichVectores);
    wait until rising_edge(clk);
    report "Finaliza la simulación";
    wait;       -- Final del bloque process
  end process gen_vec_test;
end architecture bp_regFile16x64;
```

Código VHDL 1.3: Continuación del banco de pruebas del *register file*.

PREGUNTA 4 (2.5 puntos)

Diseñe un circuito secuencial síncrono capaz de detectar si el número de unos que le ha llegado por su entrada serie de un bit es par o impar. La **entity** del circuito se muestra a continuación.

```
entity detector is
   port( Y     : out std_logic;
         X     : in std_logic;
         reset : in std_logic;
         clk   : in std_logic);
end entity detector;
```

El circuito tiene una señal de reloj (`clk`), un entrada serie de un bit (`X`), una señal de reset asíncrona activa en '1' (`reset`) y una señal de salida de un bit (`Y`).

La señal `reset` pone el circuito en su estado inicial. Este estado inicial indica que no ha llegado ningún '1' por su entrada serie. Es decir, que el número de unos que ha llegado hasta el momento es un número par. Todos los cambios tienen lugar en el flanco de subida de la señal de reloj.

La señal `Y` se pone a '1' sólo si por la entrada `X` ha llegado un número impar de unos. Por el contrario, si el número de unos que ha llegado por la entrada `X` es par, la señal `Y` se pone a '0'.

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.

Solución a la Pregunta 4

En la Figura 1.2 se muestra el diagrama de estados del circuito detector de secuencias.

El código VHDL del detector se muestra en Código VHDL 1.4.

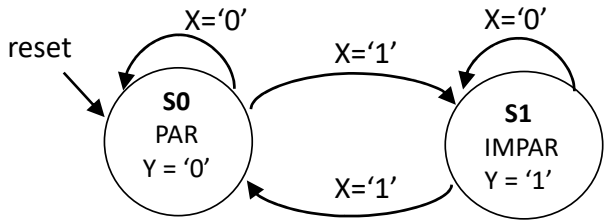


Figura 1.2: Diagrama de estados del circuito detector de secuencias.

```
-------------------------------------------------------
---Detector de impar/par
-------------------------------------------------------
library IEEE;
use IEEE.std_logic_1164.all;
entity detector is
    port( Y      : out std_logic;
          X      : in std_logic;
          reset  : in std_logic;
          clk    : in std_logic);
end entity detector;

architecture detector of detector is
       signal internal_state: std_logic;
begin
 Y <= internal_state;
 --Cálculo del próximo estado
 proximo_estado: process (clk, reset)
 begin
      if (reset = '1') then
           internal_state <= '0'; --PARA
      elsif (rising_edge(clk)) then
           case internal_state is
                when '0' =>
                     if X = '1' then
                          internal_state <= '1';
                     end if;
                when '1' =>
                     if X = '1' then
                          internal_state <= '0';
                     end if;
                 when others=>
                      internal_state <= '0';
           end case;
      end if;
end process proximo_estado;

end architecture detector;
-------------------------------------------
```

Código VHDL 1.4: Diseño del circuito detector.