Laboratorio VHDL

Hoja de respuestas del laboratorio “Sequential Circuits”

Asignatura: DSED

Número de grupo: 1

Nombres y apellidos de los miembros del grupo: Álvaro Escribano Vilar y Guillermo Pagés Scasso

Copia y pega el contenido del fichero lab3_2_1:

```
entity lab3_2_1 is
   Port ( d : in STD_LOGIC;
      clk : in STD_LOGIC;
      qa: out STD_LOGIC;
      qb: out STD_LOGIC;
      qc: out STD_LOGIC);
end lab3_2_1;
architecture Behavioral of lab3_2_1 is
begin
   process(clk, D)
   begin
      if rising_edge(clk) then
         qb <= D;
      elsif falling_edge(clk) then
         qc <= D;
      end if;


      if clk = '1' then
         qa <= D;
      end if;


   end process;
end Behavioral;
```

Copia y pega el contenido del fichero lab3_2_2:

```
entity lab3_2_2 is
  Port ( clk : in STD_LOGIC;
      d : in STD_LOGIC;
      rst : in STD_LOGIC;
      ce : in STD_LOGIC;
      q : out STD_LOGIC);
end lab3_2_2;
architecture Behavioral of lab3_2_2 is
begin
  process(clk)
  begin
    if rising_edge(clk) then
      if rst = '1' then
        q <='0';
      elsif ce = '1' then
        q <= d;
      end if;
    end if;
  end process;
end Behavioral;
```

Copia y pega el contenido del fichero lab3_2_3:

```
entity lab3_2_3 is
   Port ( clk : in STD_LOGIC;
          T : in STD_LOGIC;
          en : in STD_LOGIC;
          Q : out STD_LOGIC);
end lab3_2_3;
architecture Behavioral of lab3_2_3 is
   Signal tmp : STD_LOGIC := '0';
begin
   process(clk)
   begin
      if falling_edge(clk) then
         if (en = '1') and (T = '1') then
            tmp <= not tmp;
         else
            tmp <= tmp;
         end if;
      end if;
   end process;
   Q <= tmp;
end Behavioral;
```

Copia y pega el contenido del fichero lab3_3_1:

```
entity lab3_3_1 is
   Port ( clk : in STD_LOGIC;
          rst : in STD_LOGIC;
          D : in STD_LOGIC_VECTOR (3 downto 0);
          seta : in STD_LOGIC;
          load : in STD_LOGIC;
          Q : out STD_LOGIC_VECTOR (3 downto 0));
end lab3_3_1;
architecture Behavioral of lab3_3_1 is
   constant var: STD_LOGIC_VECTOR := "1010";
begin
   process(clk)
   begin
      if rising_edge(clk) then
         if rst = '1' then
            Q <= "0000";
         elsif seta = '1' then
            Q <= var;
         elsif load = '1' then
            Q <= D;
         end if;
      end if;
    end process;
end Behavioral;
```

Copia y pega el contenido del fichero lab3_4_1:

```
entity lab3_4_1 is
   Port ( clk : in STD_LOGIC;
          en : in STD_LOGIC;
          clear : in STD_LOGIC;
          Q : out STD_LOGIC_VECTOR (7 downto 0));
end lab3_4_1;
architecture Behavioral of lab3_4_1 is
   component lab3_2_3
      Port ( clk : in STD_LOGIC;
          T : in STD_LOGIC;
          clear: in STD_LOGIC;
          Q : out STD_LOGIC);
   end component;
   Signal aux : STD_LOGIC_VECTOR (7 downto 0);
   Signal tmp : STD_LOGIC_VECTOR (7 downto 0);
begin
   TFF0: lab3_2_3 Port map(clk,en,clear,aux(0));
   tmp(0) <= en and aux(0);
   TFF1: lab3_2_3 Port map(clk,tmp(0),clear,aux(1));
   tmp(1) <= tmp(0) and aux(1);
   TFF2: lab3_2_3 Port map(clk,tmp(1),clear,aux(2));
   tmp(2) <= tmp(1) and aux(2);
   TFF3: lab3_2_3 Port map(clk,tmp(2),clear,aux(3));
   tmp(3) <= tmp(2) and aux(3);
   TFF4: lab3_2_3 Port map(clk,tmp(3),clear,aux(4));
   tmp(4) <= tmp(3) and aux(4);
   TFF5: lab3_2_3 Port map(clk,tmp(4),clear,aux(5));
   tmp(5) <= tmp(4) and aux(5);
   TFF6: lab3_2_3 Port map(clk,tmp(5),clear,aux(6));
   tmp(6) <= tmp(5) and aux(6);
```

```vhdl
    TFF7: lab3_2_3 Port map(clk,tmp(6),clear,aux(7));

    tmp(7) <= tmp(6) and aux(7);

    Q <= aux;

end Behavioral;
```

Copia y pega el contenido del fichero lab3_4_2:

```
entity lab3_4_2 is
   Port ( clk : in STD_LOGIC;
          en : in STD_LOGIC;
          rst : in STD_LOGIC;
          Q : out STD_LOGIC_VECTOR (2 downto 0));
end lab3_4_2;

architecture Behavioral of lab3_4_2 is
   signal aux : STD_LOGIC_VECTOR (2 downto 0) := "000";
begin
   process(clk)
   begin
      if falling_edge(clk) then
         if rst = '1' then
            aux <= "000";
         elsif en = '1' then
            case aux is
               when "000" => aux <= "001";
               when "001" => aux <= "011";
               when "011" => aux <= "101";
               when "101" => aux <= "111";
               when "111" => aux <= "010";
               when "010" => aux <= "000";
               when others => aux <= "000";
             end case;
         end if;
       end if;
   end process;
   Q <= aux;
end Behavioral;
```