EL253 - Sistemas Digitales

Semestre 2020-2

Profesor Kalun José Lau Gan

Sesión de Teoría Semana 3

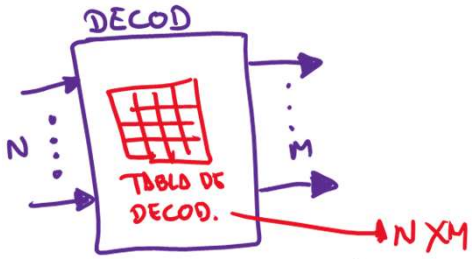
Agenda

- Modelo de decodificador en VHDL
- Revisión teórica del multiplexor
- Lógica basada en multiplexores
- Modelo de multiplexor en VHDL
- Modelo de sumador binario en VHDL
- Modelo de comparador de magnitud VHDL
- Ejercicios

Preguntas previas

- ¿Hay evaluación de laboratorio mañana?
  - Según sílabo si, está programado el LB1, lleguen puntuales, el que llegue tarde trabajará individualmente.
  - Los grupos son asignados de manera aleatoria para cada evaluación de laboratorio.
  - No se puede entregar la evaluación un día antes para garantizar la probidad de la evaluación según reglamento
- El process es como un algoritmo?
  - El process es el estilo de descripción algorítmico

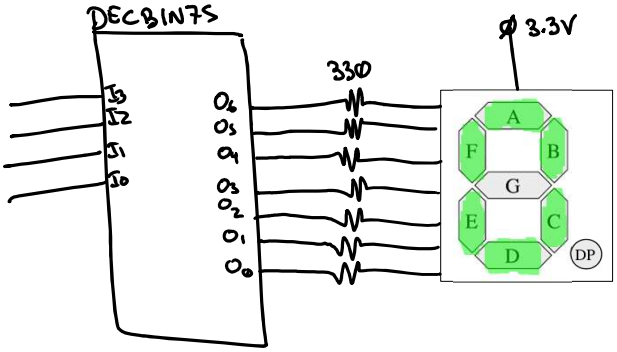
Modelo de decodificador en VHDL



Cada combinación de entrada se obtiene una salida.

El codificador hace la función inversa del decodificador

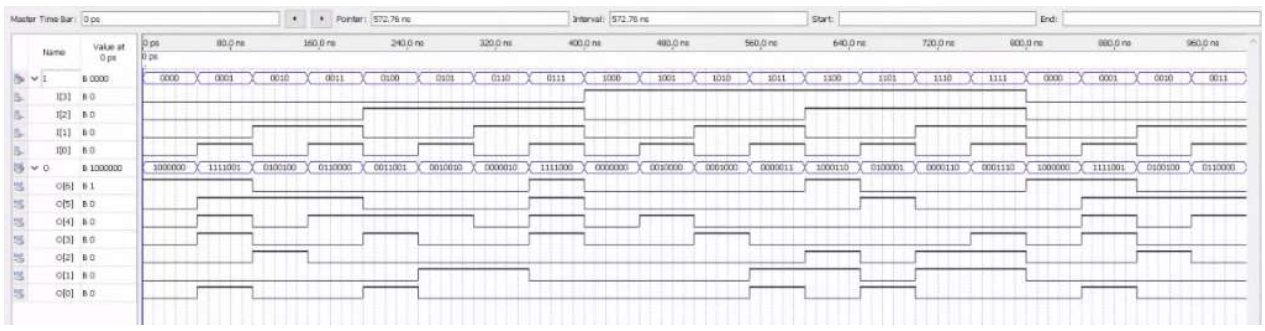
Decodificador Binario a siete segmentos




```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
USE ieee.std_logic_unsigned.all;

ENTITY decoder_test IS
   PORT(
         I  : IN    std_logic_vector  (3 DOWNTO 0);
         O  : OUT    std_logic_vector (6 DOWNTO 0)
      );
  attribute pin_assign : string;
  attribute pin_assign of I : signal is "19,18,14,13";
  attribute pin_assign of O : signal is "11,9,8,4,3,2,1";
END decoder_test ;
ARCHITECTURE Behavioral OF decoder_test IS
BEGIN
   decode:PROCESS(I)
   BEGIN
      CASE I IS            -- gfedcba            hex
         WHEN "0000" => O <= "1000000"; --   0   3f
         WHEN "0001" => O <= "1111001"; --   1   06
         WHEN "0010" => O <= "0100100"; --   2   5b
         WHEN "0011" => O <= "0110000"; --   3   4f
         WHEN "0100" => O <= "0011001"; --   4   66
         WHEN "0101" => O <= "0010010"; --   5   6d
         WHEN "0110" => O <= "0000010"; --   6   7d
         WHEN "0111" => O <= "1111000"; --   7   07
         WHEN "1000" => O <= "0000000"; --   8   7f
         WHEN "1001" => O <= "0010000"; --   9   67
         WHEN "1010" => O <= "0001000"; --   A   77
         WHEN "1011" => O <= "0000011"; --   B   7c
         WHEN "1100" => O <= "1000110"; --   C   39
         WHEN "1101" => O <= "0100001"; --   D   5e
         WHEN "1110" => O <= "0000110"; --   E   79
         WHEN "1111" => O <= "0001110"; --   F   71
         WHEN others => O <= "1111111"; --       00
      END CASE;
   END PROCESS decode;
END Behavioral;
```

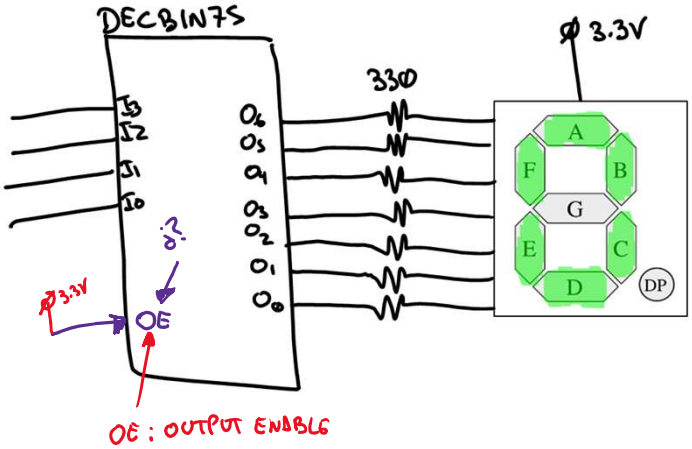
5

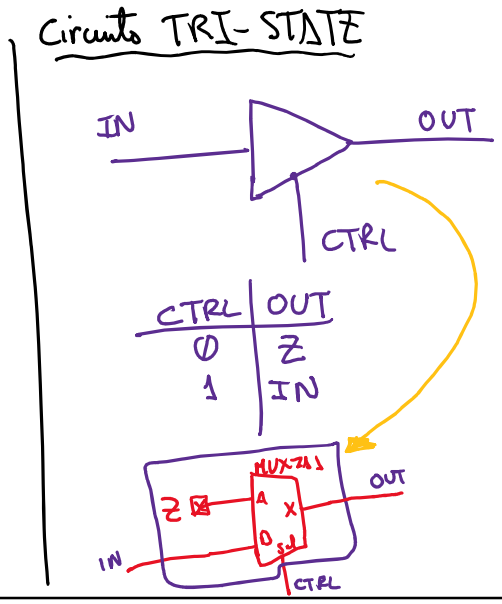
Simulación



6

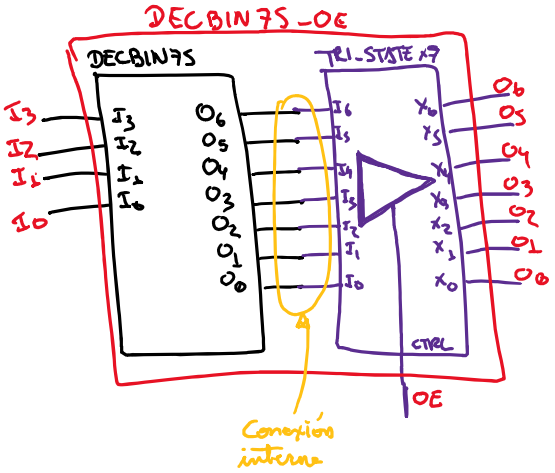
Señales de control ó habilitación de circuitos





7

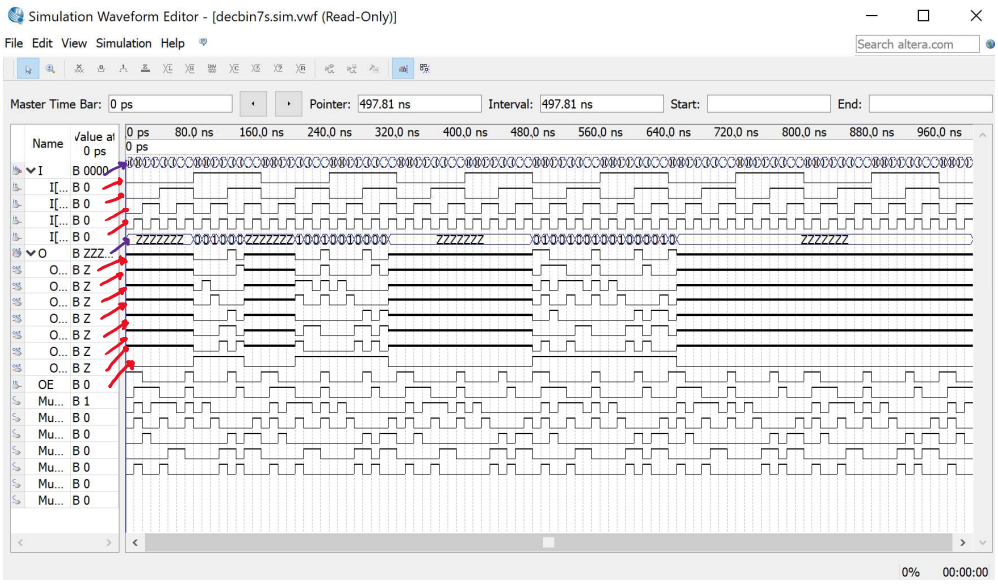
Implementando DECBIN7S con OE




```
1  LIBRARY IEEE;
2  USE IEEE.std_logic_1164.all;
3  USE IEEE.std_logic_arith.all;
4  USE IEEE.std_logic_unsigned.all;
5
6  ENTITY decbin7s IS
7          PORT (
8                  I: IN  std_logic_vector (3 downto 0);
9                  OE: IN std_logic;
10                 O: OUT std_logic_vector (6 downto 0)
11                 );
12 END decbin7s;

14 ARCHITECTURE Behavioral OF decbin7s IS
15 signal interno: std_logic_vector(6 downto 0);
16 BEGIN
17        with OE select
18           O <= interno when '1', (others => 'Z') when others;
19
20      decode: PROCESS(I)
21      BEGIN
22         CASE I IS
23                 WHEN "0000" => interno <= "1000000";
24                 WHEN "0001" => interno <= "1111001";
25                 WHEN "0010" => interno <= "0100100";
26                 WHEN "0011" => interno <= "0110000";
27                 WHEN "0100" => interno <= "0011001";
28                 WHEN "0101" => interno <= "0010010";
29                 WHEN "0110" => interno <= "0000010";
30                 WHEN "0111" => interno <= "1111000";
31                 WHEN "1000" => interno <= "0000000";
32                 WHEN "1001" => interno <= "0010000";
33                 WHEN "1010" => interno <= "0001000";
34                 WHEN "1011" => interno <= "0000011";
35                 WHEN "1100" => interno <= "1000110";
36                 WHEN "1101" => interno <= "0100001";
37                 WHEN "1110" => interno <= "0000110";
38                 WHEN "1111" => interno <= "0001110";
39                 WHEN others => interno <= "1111111";
40        END CASE;
41      END PROCESS decode;
42 END Behavioral;
```

8

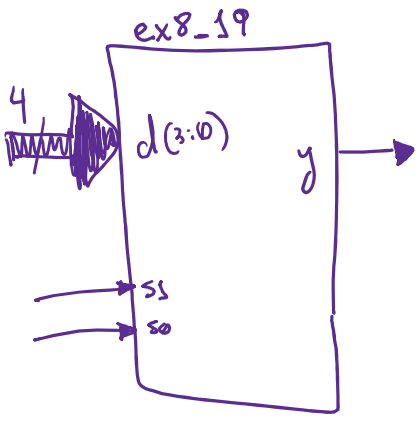
Simulación



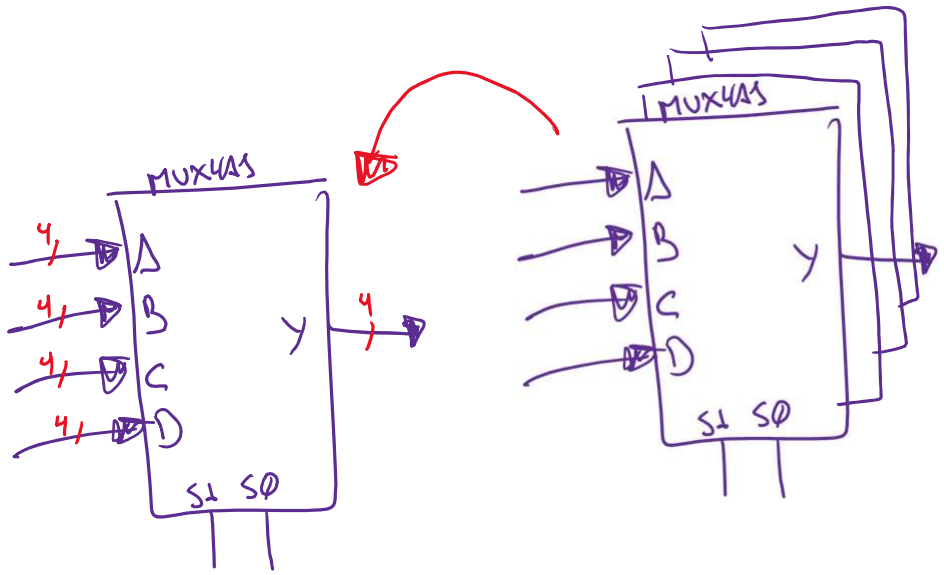
El multiplexor

Analicemos el siguiente código en VHDL:

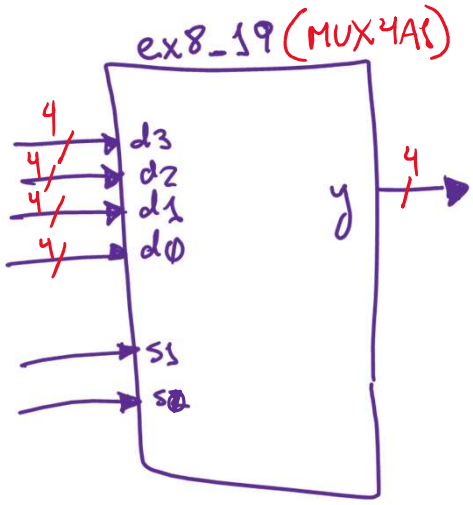
```
LIBRARY ieee;                       -- 4-line multiplexer
USE ieee.std_logic_1164.ALL;        --  using the
                                    -- selected signal assignment
ENTITY ex8_19 IS
    PORT(d  : IN    std_logic_vector (3 DOWNTO 0);
         s  : IN    std_logic_vector (1 DOWNTO 0);
         y  : OUT   std_logic);
END ex8_19 ;

ARCHITECTURE arc OF ex8_19 IS
BEGIN
    WITH s SELECT
                y<=d(3) WHEN "11",
                   d(2) WHEN "10",
                   d(1) WHEN "01",
                   d(0) WHEN "00",
                   '0'  WHEN OTHERS;
END arc;
```



MUX4A1 de 4 bits de ancho

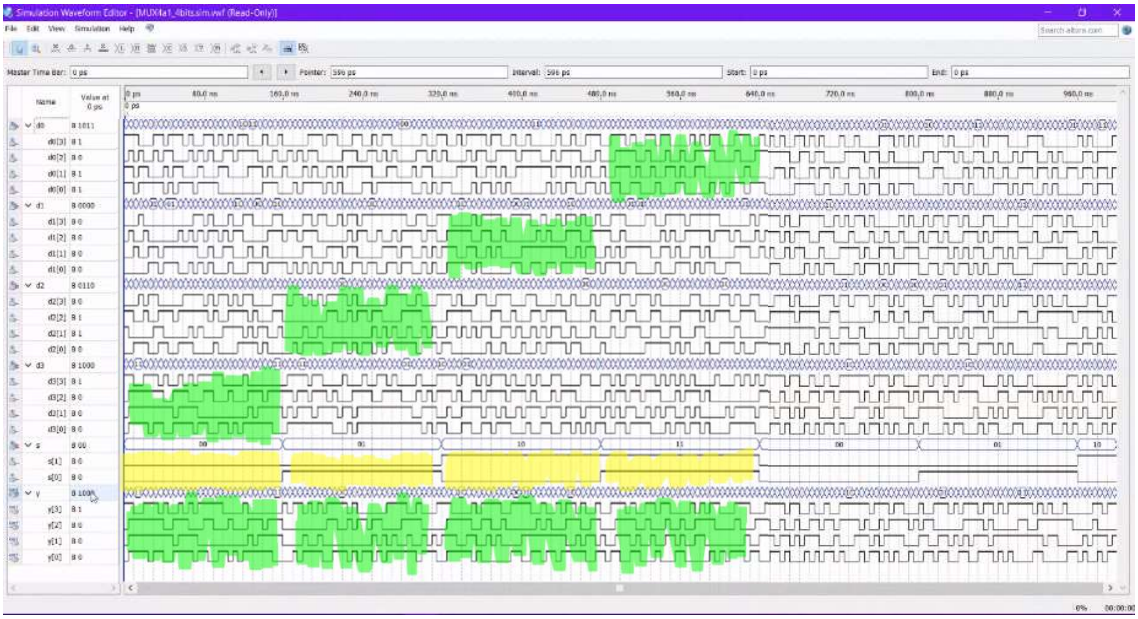


Modificando el anterior circuito para que sea de ancho de 4 bits en los datos

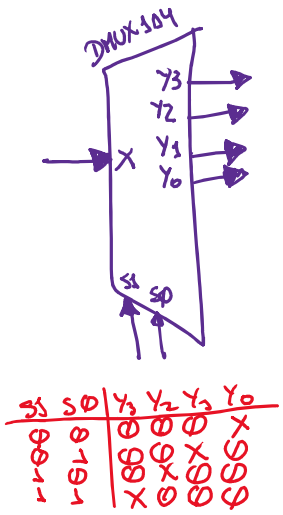



```
entity ex8_19 is
port (d3, d2, d1, d0: in std_logic_vector(3 downto 0);
        s: in std_logic_vector (1 downto 0);
        y: out std_logic_vector(3 downto 0));
end ex8_19;
architecture mas_grande of ex8_19 is
begin
                with s select
                                y <= d3 when "00",
                                     d2 when "01",
                                     d1 when "10",
                                     d0 when "11",
                                    (others => 'Z' ) when others;
end mas_grande;
```

Simulación



Demultiplexor: función inversa del MUX




```
entity dmux1a4
port(x: in std_logic;
      s: in std_logic_vector(1 downto 0);
      y: out std_logic_vector(3 downto 0));
end dmux1a4;

architecture flujazo2 of dmux1a4 is
begin
            with s select
              y(3) <= x when "11", '0' when others;
            with s select
              y(2) <= x when "10", '0' when others;
            with s select
              y(1) <= x when "01", '0' when others;
            with s select
              y(0) <= x when "00", '0' when others;
end flujazo2;
```

Ejercicio:

- Desarrollar un demultiplexor 1 a 8 donde la entrada de datos tenga un ancho de 8 bits al igual que las salidas de distribución, contemplar una entrada OE para controlar la habilitación de las salidas (‘1’ habilitadas las salidas, ‘0’ las salidas se mantendrán en ‘Z’)

- Desarrollar un decodificador binario-palabra con conexión a un display de siete segmentos del tipo cátodo común. El decodificador tendrá una tabla con las letras de la palabra “INGENIERIA” de manera ordenada. De tal manera que cuando se ingrese una cuenta de manera secuencia a la entrada de datos del decodificador, se mostrará la referida palabra a razón de una letra a la vez.