Diseño de Sistemas Electrónicos Digitales Avanzados
Sistemas combinacionales con VHDL
Índice

1  Introducción ..............................................................................................................3
2  Circuitos combinacionales .......................................................................................3
   2.1  Decodificadores ...............................................................................................3
   2.2  Decodificador BCD a 7 segmentos .................................................................5
   2.3  Multiplexores ...................................................................................................6
   2.4  Codificadores ....................................................................................................7
   2.5  Comparadores ..................................................................................................8
3  Diseño de un sistema electrónico digital .................................................................9
   3.1  Nivel de dos depósitos. ....................................................................................9
4  Entrega: ....................................................................................................................9
1 Introducción

Leer completa la práctica para luego realizar los apartados que en ella se solicitan. Esta práctica consiste en repasar los conceptos y circuitos fundamentales de los sistemas combinacionales.

2 Circuitos combinacionales

2.1 Decodificadores

Un decodificador es un circuito lógico que convierte un código binario de entrada de \( n \) bits en \( m \) líneas de salida de tal manera que exclusivamente una de esas líneas se activa (por uno o por cero) para una combinación de entrada. La Fig. 1 muestra un decodificador activo por \( \text{cero} \) y entrada de habilitación.

De tal manera que se cumple que \( 2^n = m \), por ejemplo un decodificador con 4 bits de entrada, debería tener 16 líneas de salida, \( 2^4 = 16 \)

A continuación se muestra la descripción VHDL de un decodificador de 3 bits de entrada y además una entrada de habilitación activa por uno. La salida es activa por cero.

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity DECO3A8 is
Port ( e0 : in  STD_LOGIC;
e1 : in  STD_LOGIC;
e2 : in  STD_LOGIC;
habilita : in  STD_LOGIC;
s0 : out  STD_LOGIC;
s1 : out  STD_LOGIC;
s2 : out  STD_LOGIC;
s3 : out  STD_LOGIC;
s4 : out  STD_LOGIC;
s5 : out  STD_LOGIC;
s6 : out  STD_LOGIC;
s7 : out  STD_LOGIC );
end DECO3A8;
```

**NOTA:**

1. Crear un proyecto y describir este circuito.
2. Escribir los vectores de test adecuados
3. Realizar la simulación funcional.
NOTA:
1. Crear un proyecto y describir el circuito deco3a8.
2. Escribir los vectores de test adecuados.
3. Realizar la simulación funcional.
2.2 Decodificador BCD a 7 segmentos
Este tipo de decodificador es una variante de los a los descritos en el párrafo anterior, en este caso la entrada es un código BCD, cuatro bits de entrada para indicar del 0 al 9, y la salida tiene 8 líneas de las cuales 7 se corresponden con los segmentos del display, y la otra al punto del display.

![fig. 2 a) Decodificador BCD a 7 segmentos](image)

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.numeric_std.all;

entity BCD7SEG is
PORT
    (BCD : in std_logic_vector(3 downto 0); -- código BCD entrada
     controlDisplay : out std_logic_vector (3 downto 0);
     SEGMENTOS : out std_logic_vector(7 downto 0)); -- 7segmentos + p end BCD7SEG;
architecture Behavioral of BCD7SEG is
BEGIN
    controlDisplay <= "1100"; -- control displays encendidos
    WITH BCD SELECT
        -----------gfedcbap
        SEGMENTOS <= "10000001" when "0000",
                     "11110011" when "0001",
                     "01001001" when "0010",
                     "01100001" when "0011",
                     "00110011" when "0100",
                     "00100101" when "0101",
                     "00000101" when "0110",
                     "11110001" when "0111",
                     "00000001" when "1000",
                     "00110001" when "1001",
                     "11111111" when others;
    end Behavioral;
```

b) Display de 7 segmentos

NOTA:
1. Crear un proyecto y describir el circuito decodificador BCD a 8 segmentos.
2. Escribir los vectores de test adecuados
3. Realizar la simulación funcional.
4. ¿Hay algo que no os cuadra en la simulación?
2.3 Multiplexores

Un multiplexor es un circuito lógico que posee \( n \) entradas de datos y una salida, por lo tanto necesitan \( m \) entradas de selección, donde \( m \) se puede calcular a partir de la expresión \( n=2^m \), en la figura se observa un multiplexor de 4 entradas a 1 salida, por lo tanto \( m=2 \), dos líneas de selección.

![Fig. 3](image_url)

Fig. 3 a) Multiplexor 4 a 1 de un bit  b) Multiplexor 4 a 1 de 8 bits

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity mux4a1de8 is
  generic (palabra: integer:=8);
  port (e0: in std_logic_vector(palabra-1 downto 0);
        e1: in std_logic_vector(palabra-1 downto 0);
        e2: in std_logic_vector(palabra-1 downto 0);
        e3: in std_logic_vector(palabra-1 downto 0);
        sel: in std_logic_vector(1 downto 0);
        y: out std_logic_vector(palabra-1 downto 0));
end mux4a1de8;
architecture comporta of mux4a1de8 is
begin
  with sel select
  y <= e0 when "00",
       e1 when "01",
       e2 when "10",
       e3 when others;
end comporta;

Descripción del multiplexor de la Fig. 3 b) multiplexor 4 a 1 de 8 bits, observad la utilización de `generic` para generalizar las descripciones, sería muy fácil implementar uno de 4 a 1 de 16 bits.

Observad que el `when` debe estar completo, por ello aparece la posibilidad del `when others`
NOTA:
1. Crear un proyecto y describir el circuito multiplexor 4a1 y 4a1 de 8bits.
2. Escribir los vectores de test adecuados
3. Realizar la simulación funcional.

2.4 Codificadores
Son dispositivos que realizan la operación inversa de los decodificadores. Generalmente, poseen $2^n$ entradas y $n$ salidas, es decir, proporciona a la salida el código binario de la entrada que está activada.

Podemos ver en la figura un codificador con prioridad de 8 entradas y una de habilitación, cuyo comportamiento responde a la tabla

<table>
<thead>
<tr>
<th>enable</th>
<th>I7</th>
<th>I6</th>
<th>I5</th>
<th>I4</th>
<th>I3</th>
<th>I2</th>
<th>I1</th>
<th>I0</th>
<th>S2</th>
<th>S1</th>
<th>S0</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

dentity COD8A3 is
  port(
    I  : in std_logic_vector(7 downto 0);
    enable : in std_logic;
    S   : out std_logic_vector(2 downto 0)
  );
end COD8A3;
architecture Behavioral of COD8A3 is
begin
  process (I, enable)
  begin
    if enable = '1' then
      if  (I(7)='1') then -- la prioridad la proporciona el IF
        S<="111";
      elsif (I(6)='1') then
        S<="110";
      elsif (I(5)='1') then
        S<="101";
      elsif (I(4)='1') then
        S<="100";
      elsif (I(3)='1') then
        S<="011";
      elsif (I(2)='1') then
        S<="010";
      elsif (I(1)='1') then
        S<="001";
      elsif (I(0)='1') then
        S<="000";
      end if;
    else
      S<="000";
    end if;
  end process;
end Behavioral;
NOTA:
1. Crear un proyecto y describir el circuito codificador 8a3.
2. Escribir los vectores de test adecuados
3. Realizar la simulación funcional.

2.5 Comparadores
Son circuitos que realizan la comparación entre los valores de las entradas indicando a la salida si son iguales o una mayor o menor que la otra.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity comparador is
  generic (nbits: integer:=8);
  Port (a: in STD_LOGIC_VECTOR (nbits-1 downto 0);
       b: in STD_LOGIC_VECTOR (nbits-1 downto 0);
       AmyB: out STD_LOGIC;
       AmnB: out STD_LOGIC;
       AigualB: out STD_LOGIC);
end comparador;
architecture Behavioral of comparador is
begin
  process (a,b)
  begin
    if (a > b) then
      AmyB <= '1';
      AigualB <= '0';
      AmnB <= '0';
    elsif (a = b) then
      AmyB <= '0';
      AigualB <= '1';
      AmnB <= '0';
    else
      AmyB <= '0';
      AigualB <= '0';
      AmnB <= '1';
    end if;
  end process;
end Behavioral;
```
3 Diseño de un sistema electrónico digital.

3.1 Nivel de dos depósitos.
En esta práctica se pide diseñar y describir en VHDL de forma estructurada un circuito que responda a las siguientes especificaciones:

Disponemos de dos depósitos y conectados a cada uno de ellos 4 sensores que codifican su contenido de la siguiente forma: cada sensor se activa (‘1’) cuando el nivel del contenido llega a su posición en el depósito. Esto supone cinco niveles de llenado 0000; 0001; 0011; 0111 y 1111. Para simular los sensores utilizaremos los "switches" de la placa Basys2 de la siguiente manera, sw7 a sw4 para los sensores del depósito A y sw3 a sw0 para el depósito B.

El circuito a diseñar y describir en VHDL debe indicar el menor valor de los dos depósitos en un display de 7 segmentos de la placa. Además, cuando los dos valores sean iguales el LED LD7, patilla G1, se encenderá para indicar que los valores de los dos depósitos son iguales.

Pensad como realizar el parpadeo del led LD7 a una frecuencia de 1Hz. Proponer la solución

Se debe realizar la descripción del circuito, la simulación funcional, generar el fichero *.bit y grabarlo en la placa Basys2 y comprobar su funcionamiento.

4 Entrega:
1. La entrega de la práctica se realiza a través de Campus Virtual
2. Proyecto completo ISE 14.7 convenientemente limpiado que compruebe el correcto funcionamiento del diseño.
3. Ficheros *.bit para cargar en la FPGA cuando sea conveniente