5. Decodificadores

La función de un decodificador es la siguiente: ante una combinación concreta binaria de entrada (correspondiente a una combinación de algún código binario), activar una salida correspondiente al número que identifica esa combiación.

5.1. Diseño

Veamos el diseño para dos decodificadores básicos:

Decodificador BCD a decimal = Decodificador de 4 a 10 líneas

Las combinaciones de entrada se corresponden con las combinaciones para un dígito del código BCD. Por lo tanto, el circuito dispondrá de cuatro bits de entrada y 10 bits de salida. La table de verdad correspondiente se muestra en la Tabla 27. En ella, se han representado las salidas activas a nivel bajo y no aparecen las combinaciones de entrada correspondientes a los valores entre 11 y 15 puesto que no se pueden.

Tabla de verdad

Entradas				Salidas
A3	A2	A1	A0	0 1 2 3 4 5 6 7 8 9
0	0	0	0	0 1 1 1 1 1 1 1 1 1
0	0	0	1	1 0 1 1 1 1 1 1 1 1
0	0	1	0	1 1 0 1 1 1 1 1 1 1
0	0	1	1	1 1 1 0 1 1 1 1 1 1
0	1	0	0	1 1 1 1 0 1 1 1 1 1
0	1	0	1	1 1 1 1 1 0 1 1 1 1
0	1	1	0	1 1 1 1 1 1 0 1 1 1
0	1	1	1	1 1 1 1 1 1 1 0 1 1
1	0	0	0	1 1 1 1 1 1 1 1 0 1
1	0	0	1	1 1 1 1 1 1 1 1 1 0

Tabla 27

Las funciones lógicas obteniedas a partir de la tabla de verdad son:

$$Salida\_0 = \overline{\overline{A_3}\,\overline{A_2}\,\overline{A_1}\,\overline{A_0}} = \overline{m_0} = M_0$$

$$Salida\_1 = \overline{\overline{A_3}\,\overline{A_2}\,\overline{A_1}\,A_0} = \overline{m_1} = M_1$$

....

El circuito correspondiente a las funciones ógicas es el representado en la Figura 97

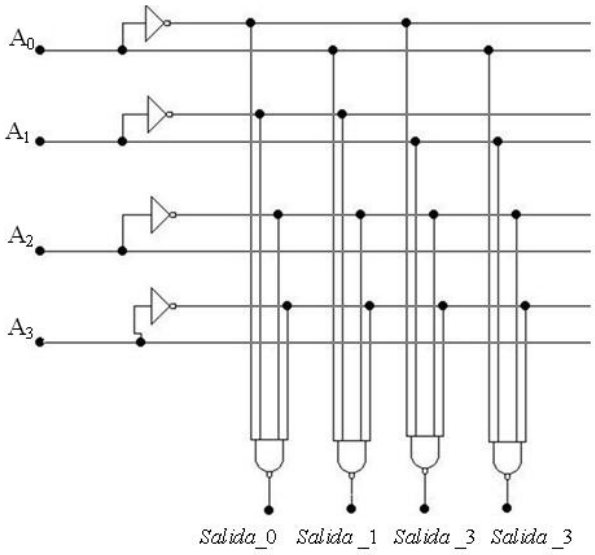

Figura 97

Decodificador de 4 a 16 líneas:

En este caso, las combinaciones de entrada se corresponden con el código binario natural, es decir, cuatro bits de entrada y por lo tanto, 16 bits de salida. Si las salidas se toman activas a baja, la tabla de verdad será la siguiente.

Tabla de Verdad

Digito Decimal	Entradas binarias $A_3A_2A_1A_0$	Función de decodificación	Salidas 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0000	$\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0001	$\overline{A}_3\overline{A}_2\overline{A}_1A_0$	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	0010	$\overline{A}_3\overline{A}_2A_1\overline{A}_0$	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
3	0011	$\overline{A}_3\overline{A}_2A_1A_0$	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
4	0100	$\overline{A}_3A_2\overline{A}_1\overline{A}_0$	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
5	0101	$\overline{A}_3A_2\overline{A}_1A_0$	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
6	0110	$\overline{A}_3A_2A_1\overline{A}_0$	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
7	0111	$\overline{A}_3A_2A_1A_0$	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
8	1000	$A_3\overline{A}_2\overline{A}_1\overline{A}_0$	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
9	1001	$A_3\overline{A}_2\overline{A}_1A_0$	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
10	1010	$A_3\overline{A}_2A_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
11	1011	$A_3\overline{A}_2A_1A_0$	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
12	1100	$A_3A_2\overline{A}_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
13	1101	$A_3A_2\overline{A}_1A_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
14	1110	$A_3A_2A_1\overline{A}_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
15	1111	$A_3A_2A_1A_0$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

5.2. Circuitos comerciales: 7442 y74154

El encapsulado correspondiente al circuito integrado 74154 se muestra en la Figura 98 y el diagrama lógico se puede analizar en la Figura 99. Las entradas de que dispone le circuito integrado son seis, cuatro correspondientes al número a decodificar y otras dos para activar el circuito. Las salidas son 16.

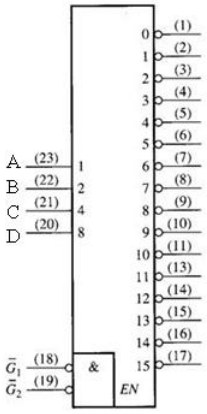

Figura 98

Cabe destacarla utilización de las dobles inversiones (marcado en rojo en la Figura 99). Se utilizan para asegurar el cumplimiento de fan-out. Si no exitiese esa doble inversión, cada entrada Ai tendría un fan-out de 7. Si esa entrada se llevara simultanemaente a otro decodificador (es habitual la utilización de varios decodificadores para decodifcar números mayores), el fan-out necesario ascendería a 14, corriendo el riesgo de superar el fan-out máximo.

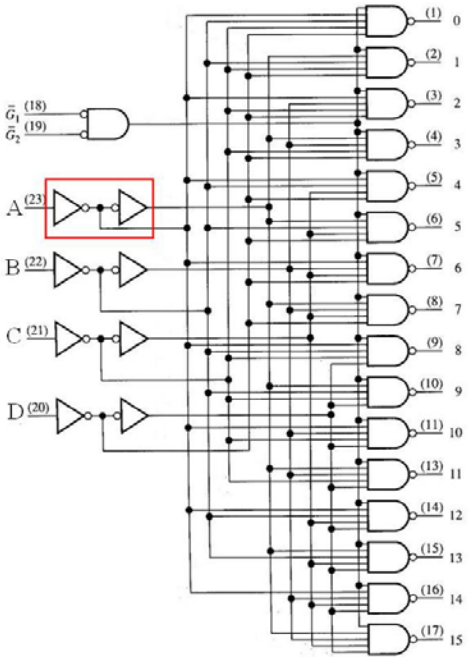

Figura 99

5.3. Aplicación

Un ejemplo de aplicación muy popular es la selección de entradas y salidas en las computadoras. Las computadoras se comunican con los perifericos para enviar datos. Para ello, previamente se debe seleccionar dicho periferico a través de un puerto de E/S y eso se lleva a cabo mediante un decodificador tal como puede observarse en la Figura 100. Cada puerto tiene asignada una dirección (número) que lo identifica.

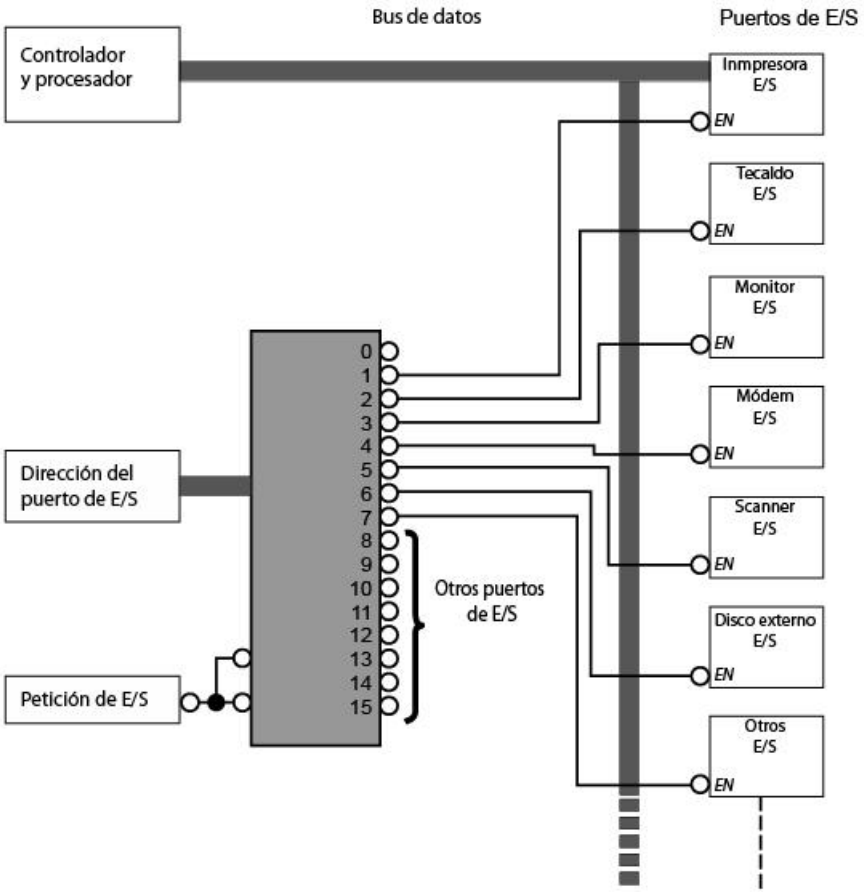

Figura 100

5.4. Descripción VHDL: 74138

```
library ieee;
use ieee.std_logic_1164.all;
entity decoder is
port(
entrada: in  std_logic_vector(2 downto 0);
g1, g2: in  std_logic;
salida: out std_logic_vector(7 downto 0)
);
end decoder;
1   architecture archidecoder of decoder is
2   begin
3   decoder: process (entrada,g1,g2)
4   begin
5   if g2='1' then salida<=(others=>'1');
6     elsif g2='0' and g1='0' then salida<=(others=>'1');
7      elsif g2='0' and g1='1' then
8
9       salida <= "01111111" when entrada="000" else
10                "10111111" when entrada="001" else
11                "11011111" when entrada="010" else
12                "11101111" when entrada="011" else
13                "11110111" when entrada="100" else
14                "11111011" when entrada="101" else
15                "11111101" when entrada="110" else
16                "11111110" when entrada="111" else
17                "11111111";
18  end archidecoder;
```

5.5. Decodificador BCD-7-Segmentos

Este tipo de decodificador tiene por entrada las combinacines asociadas al código BCD y siete salidas para activar los segmentos de un display de 7 segmentos para representar el dígito decimal correspondiente (Figura 101).

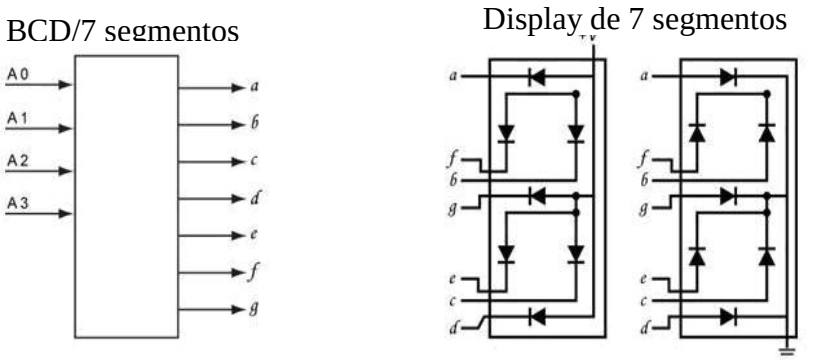

Figura 101

Tabla de Verdad

La Tabla de Verdad correspondiente será la Tabla 28

Entradas				Salidas
D	**C**	**B**	**A**	**a b c d e f g**
0	0	0	0	1 1 1 1 1 1 0
0	0	0	1	0 1 1 0 0 0 0
0	0	1	0	1 1 0 1 1 0 1
0	0	1	1	1 1 1 1 0 0 1
0	1	0	0	0 1 1 0 0 1 1
0	1	0	1	1 0 1 1 0 1 1
0	1	1	0	1 0 1 1 1 1 1
0	1	1	1	1 1 1 0 0 0 0
1	0	0	0	1 1 1 1 1 1 1
1	0	0	1	1 1 1 1 0 1 1
1	0	1	0	x x x x x x x
...	...	...	...	...
1	1	1	1	x x x x x x x

Tabla 28

Las funciones lógicas para cada una de las salidas son:

$a = \overline{A_2}\,\overline{A_0} + A_2 A_0 + A_1 + A_4$

$b = \overline{A_0} + \overline{A_2}$

$c = A_2 + A_0 + \overline{A_1}$

$d = \overline{A_2}\,\overline{A_0} + \overline{A_2} A_1 + A_2 \overline{A_1} A_0 + A_1 \overline{A_0} + A_4$

$e =$

$f =$

$g =$

Circuito comercial: 7447

Este es un ejemplo de dispositivo MSI decodificador BCD/7segmentos. Su encapsulado se muestra en la Figura 102

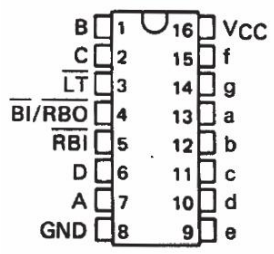

Figura 102

A parir de la tabla de verdad (Figura 103) pueden extraerse las siguientes conclusiones: En primer lugar, las salidas son activas a nivel bajo, por lo que debe conectarse a un display de ánodo común. En segunod lugar, cabe destacar la funcionalidad de la entrada LT, activa a nivel bajo, para comprobar el correcto funcionamiento de los leds deñ display. Finalmente, la entrada RBI y la terminal BI/RBO también tienen funiconalidades específicas como es la supresión de ceros no significativos.

DECIMAL OR FUNCTION	INPUTS						$\overline{BI}/\overline{RBO}$†	OUTPUTS							NOTE
	$\overline{LT}$	$\overline{RBI}$	D	C	B	A		a	b	c	d	e	f	g	
0	H	H	L	L	L	L	H	ON	ON	ON	ON	ON	ON	OFF	
1	H	X	L	L	L	H	H	OFF	ON	ON	OFF	OFF	OFF	OFF	
2	H	X	L	L	H	L	H	ON	ON	OFF	ON	ON	OFF	ON	
3	H	X	L	L	H	H	H	ON	ON	ON	ON	OFF	OFF	ON	
4	H	X	L	H	L	L	H	OFF	ON	ON	OFF	OFF	ON	ON	
5	H	X	L	H	L	H	H	ON	OFF	ON	ON	OFF	ON	ON	
6	H	X	L	H	H	L	H	OFF	OFF	ON	ON	ON	ON	ON	
7	H	X	L	H	H	H	H	ON	ON	ON	OFF	OFF	OFF	OFF	1
8	H	X	H	L	L	L	H	ON	ON	ON	ON	ON	ON	ON	
9	H	X	H	L	L	H	H	ON	ON	ON	OFF	OFF	ON	ON	
10	H	X	H	L	H	L	H	OFF	OFF	OFF	ON	ON	OFF	ON	
11	H	X	H	L	H	H	H	OFF	OFF	ON	ON	OFF	OFF	ON	
12	H	X	H	H	L	L	H	OFF	ON	OFF	OFF	OFF	ON	ON	
13	H	X	H	H	L	H	H	ON	OFF	OFF	ON	OFF	ON	ON	
14	H	X	H	H	H	L	H	OFF	OFF	OFF	ON	ON	ON	ON	
15	H	X	H	H	H	H	H	OFF	OFF	OFF	OFF	OFF	OFF	OFF	
BI	X	X	X	X	X	X	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	2
RBI	H	L	L	L	L	L	L	OFF	OFF	OFF	OFF	OFF	OFF	OFF	3
LT	L	X	X	X	X	X	H	ON	ON	ON	ON	ON	ON	ON	4

H = high level, L = low level, X = irrelevant

NOTES: 1. The blanking input ($\overline{BI}$) must be open or held at a high logic level when output functions 0 through 15 are desired. The ripple-blanking input ($\overline{RBI}$) must be open or high if blanking of a decimal zero is not desired.
2. When a low logic level is applied directly to the blanking input ($\overline{BI}$), all segment outputs are off regardless of the level of any other input.
3. When ripple-blanking input ($\overline{RBI}$) and inputs A, B, C, and D are at a low level with the lamp test input high, all segment outputs go off and the ripple-blanking output ($\overline{RBO}$) goes to a low level (response condition).
4. When the blanking input/ripple blanking output ($\overline{BI}/\overline{RBO}$) is open or held high and a low is applied to the lamp-test input, all segment outputs are on.

†$\overline{BI}/\overline{RBO}$ is wire AND logic serving as blanking input ($\overline{BI}$) and/or ripple-blanking output ($\overline{RBO}$).

Figura 103

Aplicación

En la Figura 104 se presentan ejemplos de supresión de cero mediante el decodificador BCD/7 segmentos. La clave está en que cuando el numero en BCD es el cero, dependiendo del valor de RBI, las salidas se desactivan o generan la combinación necesaria para mostrar el cero en el display. El pin BI/RBO se utiliza como salida y propagador de la información sobre la supresión del cero.

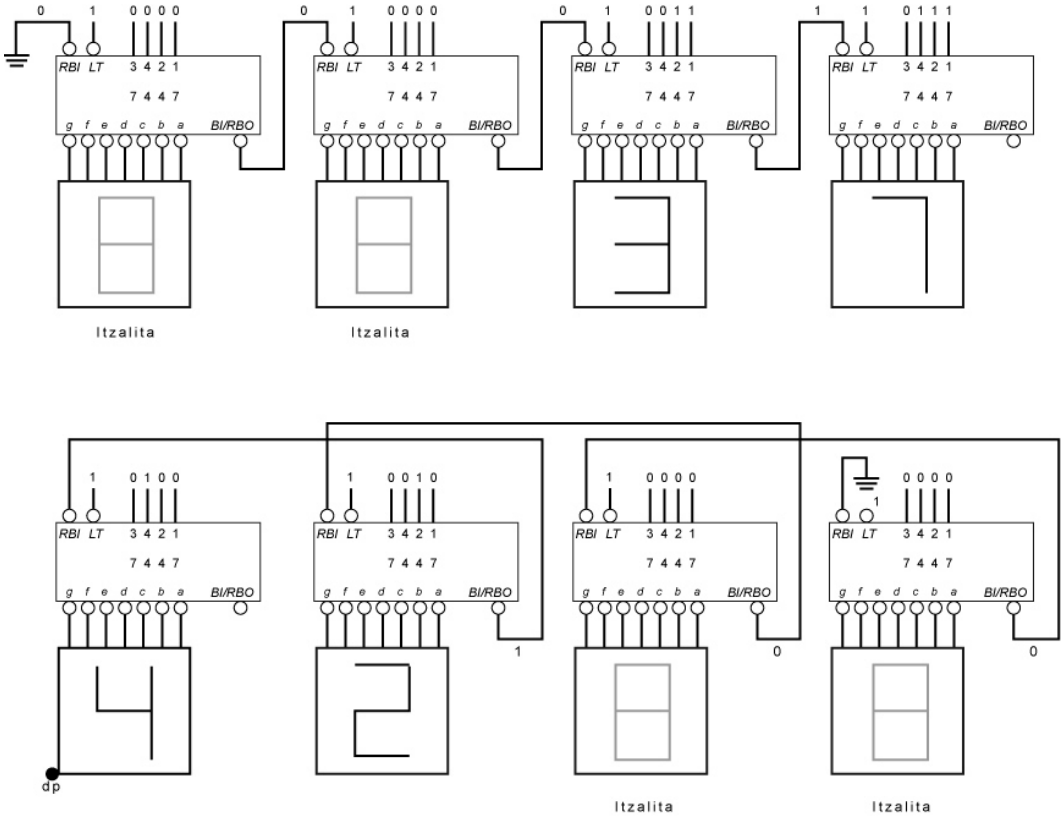

Figura 104

5.6. Descripción VHDL: BCD 7/segmentos

```
library ieee;
use ieee.std_logic_1164.all;
entity convertidor is
port(
bcd: in  bit_vector(3 downto 0);
led: out bit_vector(6 downto 0)
);
end convertidor;

1   architecture archiconv of convertidor is
2   begin
3   conv: process (bcd)
4   begin
5   case bcd is
6     when "0000" => LED <= "1111110";
7     when "0001" => LED <= "1100000";
8     when "0010" => LED <= "1011011";
9     when "0011" => LED <= "1110011";
10    when "0100" => LED <= "1100101";
11    when "0101" => LED <= "0110111";
12    when "0110" => LED <= "0111111";
13    when "0111" => LED <= "1100010";
14    when "1000" => LED <= "1111111";
15    when "1001" => LED <= "1110111";
16    when others => LED <= "0000000";
17  end case;
18  end process conv;
19  end archiconv;
```

5.7. Implementación de funciones con decodificadores

Los decodificadores pueden utilizarse para implementar una función lógica. Teniendo en cuenta las ecuaciones lógicas que implementan un decodificador:

$$Salida\_0 = \overline{\overline{A_3}\,\overline{A_2}\,\overline{A_1}\,\overline{A_0}} = \overline{m_0} = M_0$$

$$Salida\_1 = \overline{\overline{A_3}\,\overline{A_2}\,\overline{A_1}\,A_0} = \overline{m_1} = M_1$$

....

y que una función puede expresarse como suma de minterms o producto de maxterms

Función lógica como suma de productos

$$f(a,b,...,z) = m_i + m_j + ... + m_k$$

$$f(a,b,...,z) = \overline{\overline{m_i}.\overline{m_j}.....\overline{m_k}}$$

Salidas del decodificador

Función lógica como producto de sumas

$$f(a,b,...,z) = M_i.M_j.....M_k$$

$$f(a,b,...,z) = \overline{\overline{m_i}.\overline{m_j}.....\overline{m_k}}$$

Entonces se puede observar que basta con añadir a las salidas de un decodificador una puerta NAND si se expresa mendiante minterms la función lógica o una puerta AND si se expresa mediante maxterms la función lógica.