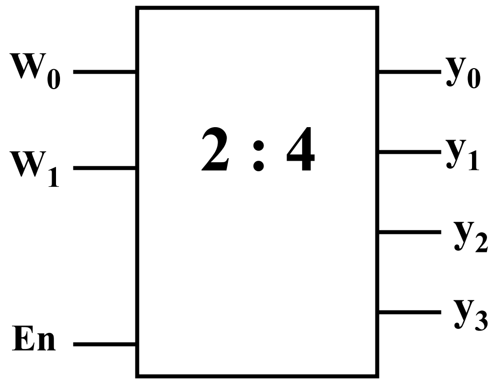
DECODIFICADORES

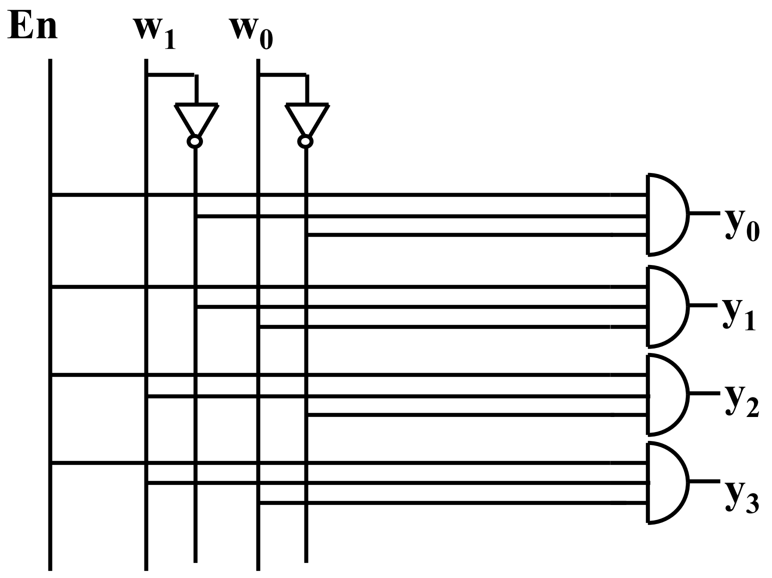
Un decodificador es un GENERADOR DE PRODUCTOS CANÓNICOS, en los cuales se presenta una sola salida activa para cada combinación de entrada.

Ejemplo:

Decodificador de 2 a 4



ENTRADAS			SALIDAS			
En	w_1	w_0	y_3	y_2	y_1	y_0
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0



$$y_0 = En \cdot \overline{w_1} \cdot \overline{w_0}$$

MODELO VHDL – DECODIFICADOR 2:4

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

ENTITY dec2to4 IS
    PORT ( w      : IN     STD_LOGIC_VECTOR(1DOWNTO 0) ; ;
           En     : IN     STD_LOGIC ;
           y      : OUT  STD_LOGIC_VECTOR(0 TO 3) ) ;
END dec2to4 ;

ARCHITECTURE Behavior OF dec2to4 IS
    SIGNAL Enw : STD_LOGIC_VECTOR(2 DOWNTO 0) ;

BEGIN
    Enw <= En & w ;   --  CONCATENATE
        WITH Enw SELECT
            y <=   "1000"  WHEN "100",  --Enw = 100 : Enw2= 1, Enw1= 0
                   "0100"  WHEN "101",  --y = 0100 : y0 = 0,  y1 = 1
                   "0010"  WHEN "110",
                   "0001"  WHEN "111",
                   "0000"  WHEN OTHERS ;
END Behavior ;
```

SEÑALES CONDICIONADAS

Similar a la asignación de señales de selección, la asignación de señales condicionales permite que una señal sea forzada a uno de varios valores.. Se usa una asignación condicional para especificar que *f* es asignada al valor w0, cuando (WHEN) *S* = 0 o de otro modo (ELSE) *f* es asignada a w1

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

ENTITY mux2to1 IS
        PORT ( w0, w1, s   : IN    STD_LOGIC ;
               f           : IN    STD_LOGIC- ;
END mux2to1 ;


ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
    f <= w0 WHEN s = ‘0’ ELSE w1 ;
END Behavior ;
```

Especificaciones para un mux 2 a 1 utilizando señales de asignacion condicionales.

DECODIFICADOR 2:4 – (PROCESS)

Hemos visto el código VHDL para un decodificador de 2 a 4. Una forma diferente de describir este circuito, es utilizando las sentencias secuenciales , tal como se muestra a continuación

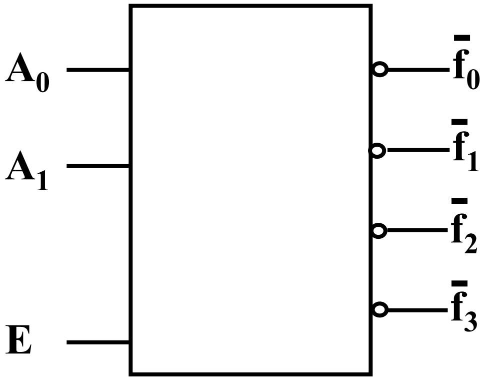
```
LIBRARY ieee;
USE ieee.std_logic_1164.all ;

ENTITY dec2to4 IS
        PORT ( w      : IN     STD_LOGIC_VECTOR(1DOWNTO 0) ; ;
                 En    : IN     STD_LOGIC- ;
                 y     : OUT  STD_LOGIC_VECTOR(0 TO 3) ) ;
END dec2to4 ;
```

DECODIFICADOR 2:4 – (PROCESS)

```
ARCHITECTURE Behavior OF dec2to4 IS
            BEGIN
                 PROCESS ( w, En )
                 BEGIN
                      IF En = `1` THEN
                        CASE w IS
                              WHEN “00” =>
                                    y  <= “1000” ;
                              WHEN “01” =>
                                    y  <= “0100” ;
                              WHEN “10” =>
                                    y  <= “0010” ;
                              WHEN OTHERS =>
                            y  <= “0001” ;
                       END CASE ;
                          ELSE
                           y <= “0000” ;
             END IF ;
             END PROCESS ;
            END Behavior ;
```

ENTRADAS			SALIDAS			
En	w_1	w_0	y_3	y_2	y_1	y_0
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

2 : 4 MAXTERM



ENTRADAS			SALIDAS			
E	A_1	A_0	f_3	f_2	f_1	f_0
0	x	x	1	1	1	1
1	0	0	1	1	1	0
1	0	1	1	1	0	1
1	1	0	1	0	1	1
1	1	1	0	1	1	1

MAXTERM

$$\overline{f_0} = \overline{E} + A_1 + A_0$$

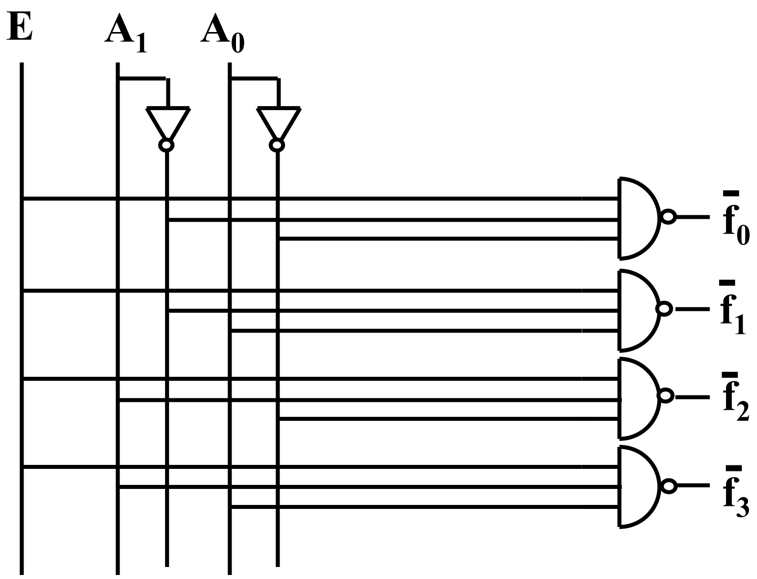
$$\overline{f_0} = \overline{\overline{\overline{E} + A_1 + A_0}}$$

$$\overline{f_0} = \overline{E \cdot \overline{A_1} \cdot \overline{A_0}}$$

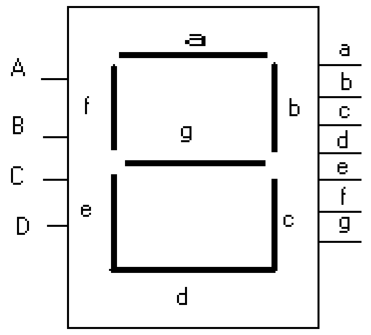
$$\overline{f_1} = \overline{E \cdot \overline{A_1} \cdot A_0}$$

$$\overline{f_2} = \overline{E \cdot A_1 \cdot \overline{A_0}}$$

$$\overline{f_3} = \overline{E \cdot A_1 \cdot A_0}$$



BCD a 7 SEGMENTOS



		ENTRADAS					SALIDAS					
Mj	mi	A	B	C	D	a	b	c	d	e	f	g
M_{15}	m_0	0	0	0	0	1	1	1	1	1	1	0
M_{14}	m_1	0	0	0	1	0	1	1	0	0	0	0
M_{13}	m_2	0	0	1	0	1	1	0	1	1	0	1
M_{12}	m_3	0	0	1	1	1	1	1	1	0	0	1
M_{11}	m_4	0	1	0	0	0	1	1	0	0	1	1
M_{10}	m_5	0	1	0	1	1	0	1	1	0	1	1
M_9	m_6	0	1	1	0	0	0	1	1	1	1	1
M_8	m_7	0	1	1	1	1	1	1	0	0	0	0
M_7	m_8	1	0	0	0	1	1	1	1	1	1	1
M_6	m_9	1	0	0	1	1	1	1	0	0	1	1
M_5	m_{10}	1	0	1	0	x	x	x	x	x	x	x
M_4	m_{11}	1	0	1	1	x	x	x	x	x	x	x
M_3	m_{12}	1	1	0	0	x	x	x	x	x	x	x
M_2	m_{13}	1	1	0	1	x	x	x	x	x	x	x
M_1	m_{14}	1	1	1	0	x	x	x	x	x	x	x
M_0	m_{15}	1	1	1	1	x	x	x	x	x	x	x

a = Σ 0,2,3,5,7,8,9,X10....X15

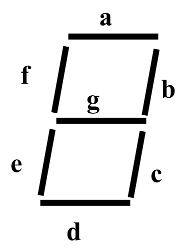
a = Π 9, 11, 14, X0,........X5

MINIMIZANDO

$a = (A + \overline{B} + D)\ (A + B + C + \overline{D})$

DECODIFICADOR BCD A 7 SEGMENTOS

```
LIBRARY ieee;
USE ieee.std_logic_1164.all ;
ENTITY seg7 IS
      PORT ( bcd    : IN     STD_LOGIC_VECTOR(3 DOWNTO 0) ;
                 leds   : OUT STD_LOGIC_VECTOR(1 TO 7) ) ;
END seg7 ;
ARCHITECTURE Behavior OF seg7 IS
BEGIN
      PROCESS ( bcd )
      BEGIN
            CASE bcd IS                              --      abcdefg
                  WHEN “0000”        => leds <= “1111110” ;
                  WHEN “0001”        => leds <= “0110000” ;
                  WHEN “0010”        => leds <= “1101101” ;
                  WHEN “0011”        => leds <= “1111001” ;
                  WHEN “0100”        => leds <= “0110011” ;
                  WHEN “0101”        => leds <= “1011011” ;
                  WHEN “0110”        => leds <= “1011110” ;
                  WHEN “0111”        => leds <= “1110000” ;
                  WHEN “1000”        => leds <= “1111111” ;
                  WHEN “1001”        => leds <= “1111011” ;
                  WHEN OTHERS     => leds <= “- - - - - - ” ;
            END CASE ;

 END PROCESS ;
```



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity dec7seg is
port(hex: in bit_vector(3 downto 0);
led: out bit_vector(6 downto 0));
end dec7seg;
architecture comportamiento of dec7seg is
begin
process(hex)
begin
case hex is
when "0001" => led <= "0010010"; --1
when "0010" => led <= "1011101"; --2
when "0011" => led <= "1011011"; --3
when "0100" => led <= "0111010"; --4
when "0101" => led <= "1101011"; --5            6
when "0110" => led <= "1101111"; --6           ---
when "0111" => led <= "1010010"; --7         5|   |4
when "1000" => led <= "1111111"; --8           ---   <- 3
when "1001" => led <= "1111011"; --9          2|   |1
when "1010" => led <= "1111110"; --A            ---
when "1011" => led <= "0101111"; --B             0
when "1100" => led <= "1100101"; --C
when "1101" => led <= "0011111"; --D
when "1110" => led <= "1101101"; --E
when "1111" => led <= "1101100"; --F
when others => led <= "1110111"; --0
end case;
end process;
end comportamiento;
```

```
--HEX-to-seven-segment decoder
  HEX:   in    STD_LOGIC_VECTOR (3 downto 0);
  LED:   out   STD_LOGIC_VECTOR (6 downto 0);

with HEX SELect
   LED <= "1111001" when "0001",   --1
          "0100100" when "0010",   --2
          "0110000" when "0011",   --3
          "0011001" when "0100",   --4
          "0010010" when "0101",   --5
          "0000010" when "0110",   --6
          "1111000" when "0111",   --7
          "0000000" when "1000",   --8
          "0010000" when "1001",   --9
          "0001000" when "1010",   --A
          "0000011" when "1011",   --b
          "1000110" when "1100",   --C
          "0100001" when "1101",   --d
          "0000110" when "1110",   --E
          "0001110" when "1111",   --F
          "1000000" when others;   --0
```

```
   0
  ---
5 |   | 1
  ---   <- 6
4 |   | 2
  ---
   3
```

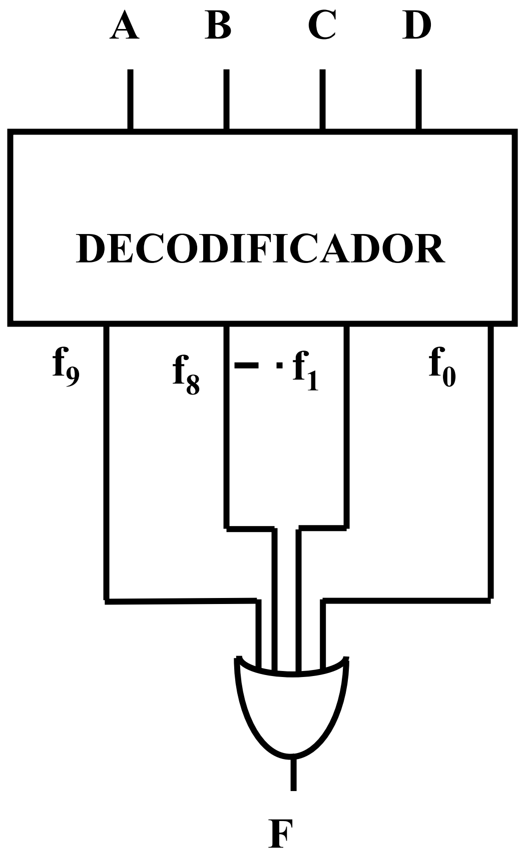
BCD A 7 SEG.   POR ECUACIONES

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity tg is
            Port (
                    A, B, C, D: in std_logic;
                    fa, fb, fc, fd, fe, ff, fg: out std_logic );
end tg;

architecture behavioral of tg is
begin
  fa<=((not(B) and not(D)) or (B and D) or (A) or (not(B) and C))
  fb<=((not(C) and not(D)) or (not(B)) or (C and D))
  fc<=((not(C)) or (D) or (B))
  fd<=((not(B) and not(D)) or (C and not(D)) or (B and not(C) and D) or (not(B) and C))
  fe<=((not(B) and not(D)) or (C and not(D)))
  ff<=((not(C) and not(D)) or (B and not(D)) or (B and not(C)) or (A))
  fg<=((B and not(C)) or (not(B) and C) or (A) or (B and not(D)))
end behavioral;
```

•*ESTE MODELO VHDL ES SOLO ILUSTRATIVO -*

DECODIFICADOR COMO GENERADOR DE FUNCIONES

$F = \Sigma\ 0, 1, 8, 9$


A
B
C
D
DECODIFICADOR
f_9
f_8
f_1
f_0
F